



## ScaleRichView

ScaleRichView © trichview.com

# Table of Contents

|  |           |
|--|-----------|
| <b>Part I ScaleRichView</b>                              | <b>18</b> |
| <b>1..Version history</b>                                | <b>19</b> |
| New after v12.0  | 20        |
| New in v12.0   | 20        |
| New in v11.0   | 21        |
| New in v10.0   | 22        |
| New in v9.0  | 23        |
| New in v8.0  | 25        |
| New in v7.0  | 26        |
| New in v6.0  | 27        |
| New in v5.0  | 28        |
| New in v4.0  | 30        |
| New in v3.0  | 33        |
| New in v2.0  | 34        |
| <b>2..Overview</b>                                       | <b>37</b> |
| Document parts in TSRichViewEdit                         | 37        |
| Notes and text boxes                                     | 38        |
| <b>3..Components</b>                                     | <b>39</b> |
| TSRichViewEdit   | 40        |
| Properties   | 44        |
| TSRichViewEdit.AcceptDragDropFormats, AcceptPasteFormats | 47        |
| TSRichViewEdit.ActiveEditor                              | 47        |
| TSRichViewEdit.AnimationMode                             | 48        |
| TSRichViewEdit.BackgroundProperty                        | 48        |
| TSRichViewEdit.BiDiMode                                  | 48        |
| TSRichViewEdit.BottomMarginPix                           | 48        |
| TSRichViewEdit.CanUpdate, CanScroll, CanUpdatePosition   | 49        |
| TSRichViewEdit.CanUpdateMargin                           | 49        |
| TSRichViewEdit.CaretBlinkTime                            | 49        |
| TSRichViewEdit.CaretPos                                  | 49        |
| TSRichViewEdit.CaretVisible                              | 50        |
| TSRichViewEdit.CPEventKind                               | 50        |
| TSRichViewEdit.CurControl                                | 50        |
| TSRichViewEdit.CurrentNote, CurrentNoteParentEditor      | 50        |
| TSRichViewEdit.CurrentPage                               | 51        |
| TSRichViewEdit.Document                                  | 51        |
| TSRichViewEdit.ExternalRVStyle, -Header, -Footer         | 51        |
| TSRichViewEdit.FirstPageNo                               | 52        |
| TSRichViewEdit.FixMarginsMode                            | 52        |
| TSRichViewEdit.HMaxScrollPos                             | 52        |
| TSRichViewEdit.HScrollBar                                | 53        |
| TSRichViewEdit.HScrollBarSchemeIndex                     | 53        |
| TSRichViewEdit.HScrollPos                                | 53        |
| TSRichViewEdit.HTMLReadProperties                        | 53        |
| TSRichViewEdit.HTMLSaveProperties                        | 53        |
| TSRichViewEdit.LastPageNo                                | 54        |
| TSRichViewEdit.LeftMarginPix                             | 54        |
| TSRichViewEdit.LineNumberProperty                        | 54        |
| TSRichViewEdit.MarkdownProperties                        | 54        |
| TSRichViewEdit.MaxPrintedItemNo, MaxPrintedOffsInItem    | 55        |

|   |    |
|---|----|
| TSRichViewEdit.MenuHButtons .....                           | 55 |
| TSRichViewEdit.MenuHorizontal .....                         | 55 |
| TSRichViewEdit.MenuVButtons .....                           | 56 |
| TSRichViewEdit.MenuVertical .....                           | 56 |
| TSRichViewEdit.MinPrintedItemNo, MinPrintedOffsInItem ..... | 56 |
| TSRichViewEdit.OffsetX .....                                | 57 |
| TSRichViewEdit.OffsetY .....                                | 57 |
| TSRichViewEdit.PageCount .....                              | 57 |
| TSRichViewEdit.PageHeightPix, PageHeight100Pix .....        | 57 |
| TSRichViewEdit.PageNoMouseDown .....                        | 58 |
| TSRichViewEdit.PageNoMouseMove .....                        | 58 |
| TSRichViewEdit.PagePosProperty .....                        | 58 |
| TSRichViewEdit.PageProperty .....                           | 58 |
| TSRichViewEdit.PageWidthPix, PageWidth100Pix .....          | 58 |
| TSRichViewEdit.RangeSearch .....                            | 58 |
| TSRichViewEdit.RangeSearchFirstY .....                      | 59 |
| TSRichViewEdit.RangeSearchLastY .....                       | 59 |
| TSRichViewEdit.ReadModeProperty .....                       | 59 |
| TSRichViewEdit.ReadOnly .....                               | 59 |
| TSRichViewEdit.RichViewEdit .....                           | 60 |
| TSRichViewEdit.RightMarginPix .....                         | 60 |
| TSRichViewEdit.RTFOptions .....                             | 60 |
| TSRichViewEdit.RTFReadProperties .....                      | 61 |
| TSRichViewEdit.RVBackgroundBitmap .....                     | 61 |
| TSRichViewEdit.RVBackgroundPicture .....                    | 61 |
| TSRichViewEdit.RVBackgroundProperties .....                 | 61 |
| TSRichViewEdit.RVBackgroundStyle .....                      | 61 |
| TSRichViewEdit.RVColor .....                                | 62 |
| TSRichViewEdit.RVEditorOptions .....                        | 62 |
| TSRichViewEdit.RVFooter .....                               | 62 |
| TSRichViewEdit.RVFOptions .....                             | 63 |
| TSRichViewEdit.RVFPaStylesReadMode .....                    | 63 |
| TSRichViewEdit.RVFTextStylesReadMode .....                  | 63 |
| TSRichViewEdit.RVHeader .....                               | 64 |
| TSRichViewEdit.RVNote .....                                 | 64 |
| TSRichViewEdit.RVOptions .....                              | 65 |
| TSRichViewEdit.ScrollBarControlStyle .....                  | 65 |
| TSRichViewEdit.ScrolledPage .....                           | 65 |
| TSRichViewEdit.SkinManager .....                            | 66 |
| TSRichViewEdit.SmartPopupProperties .....                   | 66 |
| TSRichViewEdit.SmartPopupVisible .....                      | 66 |
| TSRichViewEdit.StyleTemplateInsertMode .....                | 66 |
| TSRichViewEdit.Subdocuments .....                           | 66 |
| TSRichViewEdit.TabNavigation .....                          | 68 |
| TSRichViewEdit.TextEngine .....                             | 68 |
| TSRichViewEdit.TopMarginPix .....                           | 68 |
| TSRichViewEdit.UnitsProgram .....                           | 68 |
| TSRichViewEdit.UseDrawHyperlinksEvent .....                 | 69 |
| TSRichViewEdit.UseStyleTemplates .....                      | 69 |
| TSRichViewEdit.Version .....                                | 69 |
| TSRichViewEdit.ViewProperty .....                           | 69 |
| TSRichViewEdit.VMaxScrollPos .....                          | 69 |
| TSRichViewEdit.VScrollBar .....                             | 69 |
| TSRichViewEdit.VScrollBarSchemeIndex .....                  | 70 |
| TSRichViewEdit.VScrollPos .....                             | 70 |
| Methods .....   | 70 |
| TSRichViewEdit.BeginComponentsUpdate .....                  | 72 |
| TSRichViewEdit.CalculateAllPagePositions .....              | 72 |

|   |    |
|---|----|
| TSRichViewEdit.CalculatePageCount.....              | 72 |
| TSRichViewEdit.CalculatePagePosition.....           | 73 |
| TSRichViewEdit.Clear .....                          | 73 |
| TSRichViewEdit.ConvertRVtoSRV.....                  | 73 |
| TSRichViewEdit.ConvertSRVtoRV.....                  | 74 |
| TSRichViewEdit.ConvertTo.....                       | 75 |
| TSRichViewEdit.Copy .....                           | 75 |
| TSRichViewEdit.Create .....                         | 75 |
| TSRichViewEdit.Cut .....                            | 75 |
| TSRichViewEdit.DeletePage.....                      | 76 |
| TSRichViewEdit.Destroy .....                        | 76 |
| TSRichViewEdit.DrawMetafile.....                    | 76 |
| TSRichViewEdit.DrawPage.....                        | 77 |
| TSRichViewEdit.EndComponentsUpdate.....             | 77 |
| TSRichViewEdit.FindNextCheckpoint.....              | 78 |
| TSRichViewEdit.FindNextHeading .....                | 78 |
| TSRichViewEdit.FindNextHyperlink.....               | 79 |
| TSRichViewEdit.FindNextItem.....                    | 79 |
| TSRichViewEdit.FindPriorCheckpoint.....             | 80 |
| TSRichViewEdit.FindPriorHeading.....                | 80 |
| TSRichViewEdit.FindPriorHyperlink .....             | 81 |
| TSRichViewEdit.FindPriorItem.....                   | 81 |
| TSRichViewEdit.FirstCurPage.....                    | 82 |
| TSRichViewEdit.Format, Reformat.....                | 82 |
| TSRichViewEdit.GetCaretPosInUnits.....              | 83 |
| TSRichViewEdit.GetCurrentLineCol .....              | 83 |
| TSRichViewEdit.GetFirstItemVisibleRV.....           | 83 |
| TSRichViewEdit.GetFirstVisiblePage.....             | 84 |
| TSRichViewEdit.GetItemAt.....                       | 84 |
| TSRichViewEdit.GetItemBounds, GetItemBounds100..... | 84 |
| TSRichViewEdit.GetItemCoords100 .....               | 85 |
| TSRichViewEdit.GetItemPages.....                    | 86 |
| TSRichViewEdit.GetLastItemVisibleRV.....            | 86 |
| TSRichViewEdit.GetLastVisiblePage.....              | 87 |
| TSRichViewEdit.GetPageAt.....                       | 87 |
| TSRichViewEdit.GetPageClientRect.....               | 87 |
| TSRichViewEdit.GetPageLastItemNo .....              | 87 |
| TSRichViewEdit.GetPageNo.....                       | 88 |
| TSRichViewEdit.GetPageStartItemNo.....              | 88 |
| TSRichViewEdit.GetPageStartTableItem.....           | 88 |
| TSRichViewEdit.GetPageStartTableRow.....            | 89 |
| TSRichViewEdit.GetTableItem.....                    | 89 |
| TSRichViewEdit.GetTableItemRVData.....              | 89 |
| TSRichViewEdit.InsertPageBreak.....                 | 89 |
| TSRichViewEdit.InZoomRect.....                      | 90 |
| TSRichViewEdit.LastCurPage.....                     | 90 |
| TSRichViewEdit.LoadRVF .....                        | 90 |
| TSRichViewEdit.LoadRVFFromStream.....               | 90 |
| TSRichViewEdit.NextCurCheckpoint.....               | 90 |
| TSRichViewEdit.NextCurHeading.....                  | 91 |
| TSRichViewEdit.NextCurHyperlink.....                | 91 |
| TSRichViewEdit.NextCurItem.....                     | 91 |
| TSRichViewEdit.NextCurPage.....                     | 91 |
| TSRichViewEdit.OutZoomRect.....                     | 92 |
| TSRichViewEdit.Paste .....                          | 92 |
| TSRichViewEdit.PrintAll .....                       | 92 |
| TSRichViewEdit.PrintCurrent.....                    | 93 |
| TSRichViewEdit.PrintRange.....                      | 93 |



|   |     |
|---|-----|
| TSRichViewEdit.PriorCurCheckpoint.....                  | 94  |
| TSRichViewEdit.PriorCurHeading.....                     | 94  |
| TSRichViewEdit.PriorCurHyperlink.....                   | 94  |
| TSRichViewEdit.PriorCurItem.....                        | 95  |
| TSRichViewEdit.PriorCurPage.....                        | 95  |
| TSRichViewEdit.ProcessControls .....                    | 95  |
| TSRichViewEdit.ReturnToNote.....                        | 95  |
| TSRichViewEdit.ScrollToCaret, ScrollCaretToCenter ..... | 96  |
| TSRichViewEdit.ScrollToItem.....                        | 96  |
| TSRichViewEdit.SelectAll .....                          | 97  |
| TSRichViewEdit.Set*PropertyEd.....                      | 97  |
| TSRichViewEdit.SetMargin.....                           | 97  |
| TSRichViewEdit.SetMarginMM.....                         | 97  |
| TSRichViewEdit.SetMarginUnit.....                       | 98  |
| TSRichViewEdit.SetRVMargins.....                        | 98  |
| TSRichViewEdit.StartEditing.....                        | 99  |
| TSRichViewEdit.StartEditNote.....                       | 99  |
| TSRichViewEdit.UnitsPerInchH.....                       | 100 |
| TSRichViewEdit.UnitsPerInchV.....                       | 100 |
| TSRichViewEdit.UpdateBuffer, UpdateBufferAt.....        | 100 |
| TSRichViewEdit.ZoomCanvas, RestoreCanvasZoom.....       | 100 |
| TSRichViewEdit.ZoomRectIn, ZoomRectOut.....             | 101 |
| TSRichViewEdit.ZoomToFullPages .....                    | 101 |
| Events .....  | 101 |
| TSRichViewEdit.OnAssignImageFileName.....               | 103 |
| TSRichViewEdit.OnBeforeOleDrop, OnAfterOleDrop.....     | 104 |
| TSRichViewEdit.OnCaretGetOut.....                       | 104 |
| TSRichViewEdit.OnCaretMove, OnCaretMoving.....          | 104 |
| TSRichViewEdit.OnChange.....                            | 104 |
| TSRichViewEdit.OnChangeActiveEditor.....                | 104 |
| TSRichViewEdit.OnChangeCurrentControl .....             | 105 |
| TSRichViewEdit.OnChangeViewModeAfter.....               | 105 |
| TSRichViewEdit.OnChangeViewModeBefore.....              | 105 |
| TSRichViewEdit.OnChanging.....                          | 105 |
| TSRichViewEdit.OnCheckControl.....                      | 105 |
| TSRichViewEdit.OnCheckpointVisible.....                 | 106 |
| TSRichViewEdit.OnCheckStickingItems.....                | 106 |
| TSRichViewEdit.OnClickPage.....                         | 106 |
| TSRichViewEdit.OnContextPopup.....                      | 106 |
| TSRichViewEdit.OnControlAction.....                     | 106 |
| TSRichViewEdit.OnCopy.....                              | 106 |
| TSRichViewEdit.OnCurParaStyleChanged.....               | 107 |
| TSRichViewEdit.OnCurrentPageChange.....                 | 107 |
| TSRichViewEdit.OnCurTextStyleChanged.....               | 107 |
| TSRichViewEdit.OnDrawHyperlink.....                     | 107 |
| TSRichViewEdit.OnDropFile, OnDropFiles .....            | 108 |
| TSRichViewEdit.OnFullRedraw.....                        | 108 |
| TSRichViewEdit.OnGetPagePos.....                        | 108 |
| TSRichViewEdit.OnHMenuClickButton.....                  | 109 |
| TSRichViewEdit.OnHMenuEnterButton.....                  | 109 |
| TSRichViewEdit.OnHScrolled .....                        | 109 |
| TSRichViewEdit.OnHTMLSaveImage.....                     | 110 |
| TSRichViewEdit.OnImportPicture.....                     | 110 |
| TSRichViewEdit.OnItemAction .....                       | 110 |
| TSRichViewEdit.OnItemHint.....                          | 110 |
| TSRichViewEdit.OnItemResize.....                        | 110 |
| TSRichViewEdit.OnItemTextEdit.....                      | 110 |
| TSRichViewEdit.OnJump.....                              | 111 |

|   |     |
|---|-----|
| TSRichViewEdit.OnLoadCustomFormat.....      | 111 |
| TSRichViewEdit.OnLoadDocument.....          | 111 |
| TSRichViewEdit.OnMarginsChanged.....        | 111 |
| TSRichViewEdit.OnMessageControl .....       | 112 |
| TSRichViewEdit.OnNewDocument.....           | 113 |
| TSRichViewEdit.OnOleDragEnter.....          | 113 |
| TSRichViewEdit.OnOleDragLeave.....          | 113 |
| TSRichViewEdit.OnOleDragOver .....          | 113 |
| TSRichViewEdit.OnOleDrop.....               | 113 |
| TSRichViewEdit.OnPageCountChanged.....      | 114 |
| TSRichViewEdit.OnPageFormatChanged.....     | 114 |
| TSRichViewEdit.OnPageScrolled .....         | 114 |
| TSRichViewEdit.OnPaint.....                 | 114 |
| TSRichViewEdit.OnPaintComponent.....        | 114 |
| TSRichViewEdit.OnPaintPage.....             | 115 |
| TSRichViewEdit.OnParaStyleConversion.....   | 116 |
| TSRichViewEdit.OnPaste.....                 | 116 |
| TSRichViewEdit.OnPrinting.....              | 116 |
| TSRichViewEdit.OnProgress.....              | 117 |
| TSRichViewEdit.OnReadField.....             | 117 |
| TSRichViewEdit.OnReadHyperlink.....         | 117 |
| TSRichViewEdit.OnReadMergeField.....        | 117 |
| TSRichViewEdit.OnResize, OnResizing .....   | 117 |
| TSRichViewEdit.OnResizeDoc.....             | 117 |
| TSRichViewEdit.OnRVDbClick.....             | 117 |
| TSRichViewEdit.OnRVFControlNeeded .....     | 118 |
| TSRichViewEdit.OnRVFImageListNeeded.....    | 118 |
| TSRichViewEdit.OnRVFPictureNeeded.....      | 118 |
| TSRichViewEdit.OnRVMouseDown.....           | 118 |
| TSRichViewEdit.OnRVMouseMove.....           | 118 |
| TSRichViewEdit.OnRVMouseUp.....             | 118 |
| TSRichViewEdit.OnRVRightClick.....          | 119 |
| TSRichViewEdit.OnSaveComponentToFile.....   | 119 |
| TSRichViewEdit.OnSaveCustomFormat.....      | 119 |
| TSRichViewEdit.OnSaveDocXExtra .....        | 119 |
| TSRichViewEdit.OnSaveHTMLExtra .....        | 120 |
| TSRichViewEdit.OnSaveImage2 .....           | 120 |
| TSRichViewEdit.OnSaveItemToFile.....        | 120 |
| TSRichViewEdit.OnSaveRTFExtra.....          | 120 |
| TSRichViewEdit.OnSelect.....                | 120 |
| TSRichViewEdit.OnSmartPopupClick.....       | 120 |
| TSRichViewEdit.OnSpellingCheck.....         | 121 |
| TSRichViewEdit.OnStyleConversion.....       | 121 |
| TSRichViewEdit.OnTableIconClick.....        | 121 |
| TSRichViewEdit.OnTextFound .....            | 121 |
| TSRichViewEdit.OnURLNeeded.....             | 121 |
| TSRichViewEdit.OnVMMenuClickButton .....    | 121 |
| TSRichViewEdit.OnVMMenuEnterButton.....     | 122 |
| TSRichViewEdit.OnVScrolled.....             | 122 |
| TSRichViewEdit.OnWriteHyperlink.....        | 122 |
| TSRichViewEdit.OnWriteObjectProperties..... | 122 |
| TSRichViewEdit.OnZoomChanged .....          | 122 |
| TSRichViewEdit.OnZoomViewChanged.....       | 123 |
| Classes of properties.....                  | 123 |
| TSRVBackgroundProperty.....                 | 123 |
| Properties .....                            | 124 |
| TSRVBackgroundProperty.ColorBegin.....      | 124 |
| TSRVBackgroundProperty.ColorEnd.....        | 124 |

|   |     |
|---|-----|
| TSRVBackgroundProperty.ColorMiddle.....               | 124 |
| TSRVBackgroundProperty.FillType.....                  | 125 |
| TSRVBackgroundProperty.GlobalPageBackgroundColor..... | 125 |
| TSRVBackgroundProperty.PercentMiddle.....             | 126 |
| TSRVBackgroundProperty.Picture.....                   | 126 |
| TSRVBackgroundProperty.PicturePosition.....           | 126 |
| TSRVBackgroundProperty.TransparentColor.....          | 126 |
| TSRVBackgroundProperty.Visible.....                   | 127 |
| TSRVCustomPropertyTB.....                             | 127 |
| Properties.....                                       | 128 |
| TSRVCustomPropertyTB.AlignButton.....                 | 128 |
| TSRVCustomPropertyTB.BorderWidth.....                 | 128 |
| TSRVCustomPropertyTB.ButtonHeight.....                | 128 |
| TSRVCustomPropertyTB.Buttons.....                     | 129 |
| TSRVCustomPropertyTB.ButtonWidth.....                 | 129 |
| TSRVCustomPropertyTB.Color.....                       | 129 |
| TSRVCustomPropertyTB.ColorDisable.....                | 129 |
| TSRVCustomPropertyTB.ColorDown.....                   | 130 |
| TSRVCustomPropertyTB.ColorSelect.....                 | 130 |
| TSRVCustomPropertyTB.ImageList.....                   | 130 |
| TSRVCustomPropertyTB.Indent.....                      | 130 |
| TSRVCustomPropertyTB.PenFrame.....                    | 130 |
| TSRVCustomPropertyTB.ScaleImagesForDPI.....           | 131 |
| TSRVCustomPropertyTB.Spacer.....                      | 131 |
| TSRVCustomPropertyTB.SRVToolBar.....                  | 131 |
| TSRVLineNumberProperty.....                           | 131 |
| Properties.....                                       | 132 |
| TSRVLineNumberProperty.DistanceFromText.....          | 132 |
| TSRVLineNumberProperty.Font.....                      | 132 |
| TSRVLineNumberProperty.StartFrom.....                 | 132 |
| TSRVLineNumberProperty.Step.....                      | 132 |
| TSRVLineNumberProperty.TablesAsLines.....             | 133 |
| TSRVLineNumberProperty.Ticks.....                     | 133 |
| TSRVLineNumberProperty.Visible.....                   | 133 |
| TSRVPageProperty.....                                 | 133 |
| Properties.....                                       | 134 |
| TSRVPageProperty.AutoWidth.....                       | 135 |
| TSRVPageProperty.BorderPen.....                       | 135 |
| TSRVPageProperty.BottomMargin.....                    | 136 |
| TSRVPageProperty.CaretPen.....                        | 136 |
| TSRVPageProperty.FacingPages.....                     | 136 |
| TSRVPageProperty.FooterVisible.....                   | 136 |
| TSRVPageProperty.FooterY.....                         | 137 |
| TSRVPageProperty.HeaderVisible.....                   | 137 |
| TSRVPageProperty.HeaderY.....                         | 137 |
| TSRVPageProperty.LeftMargin.....                      | 138 |
| TSRVPageProperty.MirrorMargins.....                   | 138 |
| TSRVPageProperty.Orientation.....                     | 138 |
| TSRVPageProperty.PageFormat.....                      | 139 |
| TSRVPageProperty.PageHeight.....                      | 139 |
| TSRVPageProperty.PageNoFirst.....                     | 139 |
| TSRVPageProperty.PageNoFont.....                      | 139 |
| TSRVPageProperty.PageNoFromNumber.....                | 140 |
| TSRVPageProperty.PageNoHAlign.....                    | 140 |
| TSRVPageProperty.PageNoVAlign.....                    | 140 |
| TSRVPageProperty.PageNoVisible.....                   | 140 |
| TSRVPageProperty.PageViewMode.....                    | 141 |
| TSRVPageProperty.PageWidth.....                       | 141 |

|  |     |
|--|-----|
| TSRVPageProperty.PrintableAreaPen.....                     | 141 |
| TSRVPageProperty.RightMargin.....                          | 142 |
| TSRVPageProperty.TitlePage.....                            | 142 |
| TSRVPageProperty.TopMargin.....                            | 142 |
| TSRVPageProperty.UsePageOrders.....                        | 142 |
| Methods .....  | 143 |
| TSRVPageProperty.ConvertPageFormatToPageSize.....          | 143 |
| TSRVPageProperty.ConvertPageSizeToPageFormat.....          | 143 |
| TSRVPagePositionProperty.....                              | 143 |
| Properties .....   | 144 |
| TSRVPagePositionProperty.AlignPageH.....                   | 144 |
| TSRVPagePositionProperty.AlignPageV.....                   | 145 |
| TSRVPagePositionProperty.HiddenDistances .....             | 145 |
| TSRVPagePositionProperty.HPadding, VPadding.....           | 146 |
| TSRVPagePositionProperty.MaxPageColCount.....              | 147 |
| TSRVPagePositionProperty.PageHSpacing, PageVSpacing.....   | 147 |
| TSRVPropertyTBH .....                                      | 148 |
| Properties .....   | 149 |
| TSRVPropertyTBH.Align.....                                 | 149 |
| TSRVPropertyTBH.Visible.....                               | 149 |
| TSRVPropertyTBV .....                                      | 149 |
| Properties .....   | 150 |
| TSRVPropertyTBV.Align.....                                 | 150 |
| TSRVPropertyTBV.Visible.....                               | 151 |
| TSRVReadModeProperty.....                                  | 151 |
| Properties .....   | 152 |
| TSRVReadModeProperty.Active.....                           | 152 |
| TSRVReadModeProperty.HidePageBackgroundImage.....          | 152 |
| TSRVReadModeProperty.MinPageColCount, MaxPageColCount..... | 152 |
| TSRVReadModeProperty.PageColor .....                       | 153 |
| TSRVReadModeProperty.PageFlipEffect.....                   | 153 |
| TSRVViewProperty .....                                     | 154 |
| Properties .....   | 155 |
| TSRVViewProperty.AutoVScrollBarPosition.....               | 156 |
| TSRVViewProperty.DarkMode.....                             | 157 |
| TSRVViewProperty.DarkModeUI.....                           | 157 |
| TSRVViewProperty.FooterPen.....                            | 157 |
| TSRVViewProperty.FooterTitleColor.....                     | 157 |
| TSRVViewProperty.FooterTitleFont.....                      | 158 |
| TSRVViewProperty.FreePosPage.....                          | 158 |
| TSRVViewProperty.HeaderFooterShade.....                    | 158 |
| TSRVViewProperty.HeaderPen.....                            | 159 |
| TSRVViewProperty.HeaderTitleColor.....                     | 159 |
| TSRVViewProperty.HeaderTitleFont.....                      | 159 |
| TSRVViewProperty.HintFont.....                             | 160 |
| TSRVViewProperty.HintPrefixText.....                       | 160 |
| TSRVViewProperty.IconStyle.....                            | 160 |
| TSRVViewProperty.MainDocumentShade.....                    | 160 |
| TSRVViewProperty.MainPen.....                              | 161 |
| TSRVViewProperty.MainTitleColor.....                       | 161 |
| TSRVViewProperty.MainTitleFont.....                        | 161 |
| TSRVViewProperty.MarginsPen.....                           | 162 |
| TSRVViewProperty.MarginsRectVisible.....                   | 162 |
| TSRVViewProperty.MouseWheelZoom.....                       | 162 |
| TSRVViewProperty.PageHeight.....                           | 162 |
| TSRVViewProperty.PageWidth.....                            | 162 |
| TSRVViewProperty.ShowScrollHint.....                       | 163 |
| TSRVViewProperty.TableIconDelay.....                       | 163 |

|   |            |
|---|------------|
| TSRVViewProperty.TableIconPopupMenu.....    | 163        |
| TSRVViewProperty.Texts.....                 | 163        |
| TSRVViewProperty.UseTableIcons.....         | 164        |
| TSRVViewProperty.UseVCLThemes.....          | 165        |
| TSRVViewProperty.ViewMode.....              | 165        |
| TSRVViewProperty.WheelStep.....             | 165        |
| TSRVViewProperty.WorkArea.....              | 165        |
| TSRVViewProperty.ZoomFont.....              | 166        |
| TSRVViewProperty.ZoomMenu.....              | 166        |
| TSRVViewProperty.ZoomMin, ZoomMax.....      | 166        |
| TSRVViewProperty.ZoomMode.....              | 166        |
| TSRVViewProperty.ZoomModeEdit.....          | 167        |
| TSRVViewProperty.ZoomModeIN.....            | 167        |
| TSRVViewProperty.ZoomModeOUT.....           | 167        |
| TSRVViewProperty.ZoomPanelFont.....         | 167        |
| TSRVViewProperty.ZoomPanelUsesGradient..... | 168        |
| TSRVViewProperty.ZoomPanelVisible.....      | 168        |
| TSRVViewProperty.ZoomPercent.....           | 168        |
| TSRVViewProperty.ZoomPercentEdit.....       | 169        |
| TSRVViewProperty.ZoomPercentIN.....         | 169        |
| TSRVViewProperty.ZoomPercentOUT.....        | 169        |
| <b>TDBSRichViewEdit .....</b>               | <b>169</b> |
| Properties .....                            | 170        |
| TDBSRichViewEdit.AutoDisplay.....           | 173        |
| TDBSRichViewEdit.DataField.....             | 173        |
| TDBSRichViewEdit.DataSource.....            | 173        |
| TDBSRichViewEdit.ExternalRV.....            | 174        |
| TDBSRichViewEdit.FieldFormat.....           | 174        |
| TDBSRichViewEdit.IgnoreEscape.....          | 174        |
| TDBSRichViewEdit.RichViewEdit.....          | 174        |
| <b>TSRVPageScroll .....</b>                 | <b>174</b> |
| Properties .....                            | 175        |
| TSRVPageScroll.AlignText.....               | 176        |
| TSRVPageScroll.AutoScrolled.....            | 177        |
| TSRVPageScroll.AutoUpdate.....              | 177        |
| TSRVPageScroll.BackgroundColor.....         | 177        |
| TSRVPageScroll.CachedPagesCount.....        | 178        |
| TSRVPageScroll.CacheScrollLimit.....        | 178        |
| TSRVPageScroll.ChangeDelay.....             | 178        |
| TSRVPageScroll.Columns.....                 | 178        |
| TSRVPageScroll.MarginHorizontal.....        | 179        |
| TSRVPageScroll.MarginVertical.....          | 179        |
| TSRVPageScroll.PageBorderColor.....         | 179        |
| TSRVPageScroll.PageBreakHeight.....         | 179        |
| TSRVPageScroll.PageBreakWidth.....          | 180        |
| TSRVPageScroll.PageHeight.....              | 180        |
| TSRVPageScroll.PageNoFont.....              | 180        |
| TSRVPageScroll.PageNoVisible.....           | 181        |
| TSRVPageScroll.PageProportions.....         | 181        |
| TSRVPageScroll.PageWidth.....               | 181        |
| TSRVPageScroll.ScrollType.....              | 181        |
| TSRVPageScroll.SelectColor.....             | 182        |
| TSRVPageScroll.ShadowColor.....             | 182        |
| TSRVPageScroll.ShadowWidth.....             | 182        |
| TSRVPageScroll.SmoothScroll.....            | 182        |
| TSRVPageScroll.SmoothThumbnails.....        | 183        |
| TSRVPageScroll.SRichViewEdit.....           | 183        |
| Methods .....                               | 183        |

|   |            |
|---|------------|
| TSRVPageScroll.BeginUpdate.....         | 183        |
| TSRVPageScroll.EndUpdate.....           | 183        |
| TSRVPageScroll.First .....              | 184        |
| TSRVPageScroll.GetPageAt.....           | 184        |
| TSRVPageScroll.Last .....               | 184        |
| TSRVPageScroll.Next .....               | 184        |
| TSRVPageScroll.Prev .....               | 185        |
| TSRVPageScroll.ScrollToPage.....        | 185        |
| TSRVPageScroll.UpdateCache.....         | 185        |
| Events .....                            | 185        |
| TSRVPageScroll.OnDrawPageBorder.....    | 186        |
| TSRVPageScroll.OnDrawPageNumber.....    | 186        |
| TSRVPageScroll.OnDrawSelectedPage.....  | 187        |
| <b>TSclRVRuler .....</b>                | <b>187</b> |
| Properties .....                        | 187        |
| TSclRVRuler.SRichViewEdit.....          | 188        |
| Methods .....                           | 188        |
| TSclRVRuler.CheckMargins .....          | 188        |
| TSclRVRuler.GetPageProperties.....      | 188        |
| TSclRVRuler.Scrolled .....              | 189        |
| TSclRVRuler.SetPageProperties .....     | 189        |
| Events .....                            | 189        |
| TSclRVRuler.OnAfterMarginChanged.....   | 189        |
| <b>TSRVToolBar .....</b>                | <b>189</b> |
| Properties .....                        | 190        |
| TSRVToolBar.AlignButton.....            | 191        |
| TSRVToolBar.AlignText .....             | 191        |
| TSRVToolBar.AutoSize .....              | 192        |
| TSRVToolBar.BorderWidth.....            | 192        |
| TSRVToolBar.ButtonCol .....             | 192        |
| TSRVToolBar.ButtonHeight.....           | 192        |
| TSRVToolBar.Buttons .....               | 192        |
| TSRVToolBar.ButtonWidth.....            | 193        |
| TSRVToolBar.Color .....                 | 193        |
| TSRVToolBar.ColorDisable.....           | 193        |
| TSRVToolBar.ColorDown.....              | 193        |
| TSRVToolBar.ColorSelect.....            | 194        |
| TSRVToolBar.HintCancel.....             | 194        |
| TSRVToolBar.HintFont .....              | 194        |
| TSRVToolBar.HintHeight.....             | 194        |
| TSRVToolBar.ImageList .....             | 195        |
| TSRVToolBar.Indent .....                | 195        |
| TSRVToolBar.PenFrame .....              | 195        |
| TSRVToolBar.ScaleImagesForDPI.....      | 195        |
| TSRVToolBar.Spacer .....                | 196        |
| Events .....                            | 196        |
| TSRVToolBar.OnClickButton.....          | 196        |
| TSRVToolBar.OnDrawButtonBackground..... | 196        |
| TSRVToolBar.OnEnter .....               | 196        |
| TSRVToolBar.OnEnterButton.....          | 197        |
| TSRVToolBar.OnLeave .....               | 197        |
| <b>TSRVToolWindow .....</b>             | <b>197</b> |
| Properties .....                        | 198        |
| TSRVToolWindow.AlignText.....           | 198        |
| TSRVToolWindow.BorderWidth.....         | 199        |
| TSRVToolWindow.ButtonCol.....           | 199        |
| TSRVToolWindow.ButtonHeight.....        | 199        |
| TSRVToolWindow.Buttons.....             | 199        |

|                                       |            |
|---------------------------------------|------------|
| TSRVToolWindow.ButtonWidth.....       | 199        |
| TSRVToolWindow.Color.....             | 200        |
| TSRVToolWindow.ColorDisable.....      | 200        |
| TSRVToolWindow.ColorDown.....         | 200        |
| TSRVToolWindow.ColorSelect.....       | 201        |
| TSRVToolWindow.HintCancel.....        | 201        |
| TSRVToolWindow.HintFont.....          | 201        |
| TSRVToolWindow.HintHeight.....        | 201        |
| TSRVToolWindow.ImageList.....         | 202        |
| TSRVToolWindow.PenFrame.....          | 202        |
| TSRVToolWindow.ScaleImagesForDPI..... | 202        |
| TSRVToolWindow.Spacer.....            | 202        |
| Methods .....                         | 202        |
| TSRVToolWindow.Execute.....           | 203        |
| Events .....                          | 203        |
| TSRVToolWindow.OnClickButton.....     | 203        |
| TSRVToolWindow.OnEnter .....          | 203        |
| TSRVToolWindow.OnEnterButton.....     | 203        |
| TSRVToolWindow.OnLeave.....           | 204        |
| TSRVToolWindow.OnSelectEvent.....     | 204        |
| <b>TSRVPrint .....</b>                | <b>204</b> |
| Properties .....                      | 206        |
| TSRVPrint.AutoUpdate .....            | 207        |
| TSRVPrint.Center .....                | 207        |
| TSRVPrint.ClipMargins .....           | 208        |
| TSRVPrint.FrameCountX.....            | 208        |
| TSRVPrint.FrameCountY.....            | 209        |
| TSRVPrint.FrameHeightPix.....         | 209        |
| TSRVPrint.FrameWidthPix.....          | 210        |
| TSRVPrint.IntegerCount .....          | 210        |
| TSRVPrint.MetafileCompatibility.....  | 211        |
| TSRVPrint.NoMetafiles .....           | 211        |
| TSRVPrint.OffXPix .....               | 211        |
| TSRVPrint.OffYPix .....               | 212        |
| TSRVPrint.Orientation .....           | 212        |
| TSRVPrint.PageColCount.....           | 213        |
| TSRVPrint.PageFormat .....            | 213        |
| TSRVPrint.PageHeightPix.....          | 213        |
| TSRVPrint.PageRowCount.....           | 214        |
| TSRVPrint.PageWidthPix.....           | 214        |
| TSRVPrint.PrintMode .....             | 214        |
| TSRVPrint.RealHeight .....            | 215        |
| TSRVPrint.RealWidth .....             | 216        |
| TSRVPrint.ScaleImage .....            | 216        |
| TSRVPrint.SRichViewEdit.....          | 217        |
| TSRVPrint.TotalFrameHPix.....         | 217        |
| TSRVPrint.TotalFrameWPix.....         | 217        |
| TSRVPrint.UsePhysicalOffsets.....     | 217        |
| Methods .....                         | 218        |
| TSRVPrint.BeginUpdate .....           | 218        |
| TSRVPrint.CheckMargin.....            | 218        |
| TSRVPrint.EndUpdate .....             | 219        |
| TSRVPrint.OptimalOrientation.....     | 219        |
| TSRVPrint.PaperCodeToPageFormat.....  | 219        |
| TSRVPrint.Print .....                 | 219        |
| TSRVPrint.PrintFrame .....            | 220        |
| TSRVPrint.PrintFrames .....           | 220        |
| TSRVPrint.PrintFramesEx.....          | 221        |

|   |            |
|---|------------|
| TSRVPrint.PrintPages .....                | 221        |
| TSRVPrint.Update .....                    | 222        |
| Events .....                              | 222        |
| TSRVPrint.OnSendingToPrinter.....         | 222        |
| TSRVPrint.OnUpdateData .....              | 222        |
| <b>TSRVPreview .....</b>                  | <b>223</b> |
| Properties .....                          | 223        |
| TSRVPreview.BorderColor.....              | 225        |
| TSRVPreview.BorderWidth.....              | 225        |
| TSRVPreview.FrameBorderColor.....         | 225        |
| TSRVPreview.FrameBorderWidth.....         | 225        |
| TSRVPreview.PageBorderColor.....          | 226        |
| TSRVPreview.PageBorderStyle.....          | 226        |
| TSRVPreview.PageBorderWidth.....          | 226        |
| TSRVPreview.PageNo .....                  | 226        |
| TSRVPreview.SRVPrint .....                | 227        |
| Methods .....                             | 227        |
| TSRVPreview.First .....                   | 227        |
| TSRVPreview.Last .....                    | 227        |
| TSRVPreview.Next .....                    | 227        |
| TSRVPreview.Prev .....                    | 227        |
| Events .....                              | 227        |
| TSRVPreview.OnAfterPaint.....             | 228        |
| TSRVPreview.OnBeforePaint.....            | 228        |
| <b>TSRVSkinManager .....</b>              | <b>228</b> |
| Properties .....                          | 230        |
| TSRVSkinManager.CurrentSkin.....          | 230        |
| TSRVSkinManager.SkinIndex.....            | 230        |
| TSRVSkinManager.Skins.....                | 230        |
| Classes of properties .....               | 230        |
| TSRVSkin .....                            | 231        |
| Properties .....                          | 232        |
| TSRVSkin.BoxSchemes.....                  | 232        |
| TSRVSkin.Elements.....                    | 232        |
| TSRVSkin.Fonts .....                      | 232        |
| TSRVSkin.HorizontalScrollBarSchemes ..... | 232        |
| TSRVSkin.HorizontalTabSetSchemes .....    | 232        |
| TSRVSkin.Name.....                        | 233        |
| TSRVSkin.VerticalScrollBarSchemes .....   | 233        |
| TSRVSkin.VerticalTabSetSchemes .....      | 233        |
| TSRVBoxScheme .....                       | 233        |
| TSRVBoxSchemeCollection .....             | 234        |
| TSRVSkinCollection .....                  | 235        |
| TSRVSkinFont .....                        | 235        |
| TSRVSkinFontCollection.....               | 236        |
| TSRVSkinElement .....                     | 237        |
| TSRVSkinElementCollection.....            | 238        |
| TSRVScrollBarSkinScheme.....              | 238        |
| TSRVScrollBarSkinSchemeCollection .....   | 240        |
| TSRVTabSetSkinScheme .....                | 241        |
| TSRVTabSetSkinSchemeCollection .....      | 242        |
| <b>TSRVScrollBar .....</b>                | <b>243</b> |
| Properties .....                          | 244        |
| TSRVScrollBar.DecButtonWidth.....         | 245        |
| TSRVScrollBar.Flat .....                  | 245        |
| TSRVScrollBar.FocusBlinkDelay.....        | 245        |
| TSRVScrollBar.IncButtonWidth.....         | 246        |
| TSRVScrollBar.Kind .....                  | 246        |



|  |            |
|--|------------|
| TSRVScrollBar.PageSize .....                 | 246        |
| TSRVScrollBar.Position .....                 | 246        |
| TSRVScrollBar.ScrollingDelay .....           | 247        |
| TSRVScrollBar.SkinManager .....              | 247        |
| TSRVScrollBar.SkinSchemeIndex .....          | 247        |
| TSRVScrollBar.SmallChange, LargeChange ..... | 247        |
| TSRVScrollBar.SRVControlStyle .....          | 248        |
| TSRVScrollBar.UseXPThemes .....              | 248        |
| Events .....                                 | 248        |
| TSRVScrollBar.OnDraw... .....                | 249        |
| <b>TSRVTabSet .....</b>                      | <b>249</b> |
| Properties .....                             | 251        |
| TSRVTabSet.AutoTabWidth .....                | 252        |
| TSRVTabSet.CanMoveTabs .....                 | 253        |
| TSRVTabSet.CloseButton .....                 | 253        |
| TSRVTabSet.DarkMode, AutoDarkMode .....      | 253        |
| TSRVTabSet.FirstTabIndex .....               | 253        |
| TSRVTabSet.ImageList .....                   | 253        |
| TSRVTabSet.Indent .....                      | 254        |
| TSRVTabSet.Kind .....                        | 254        |
| TSRVTabSet.LastTabIndex .....                | 254        |
| TSRVTabSet.Margins .....                     | 254        |
| TSRVTabSet.MaxTabSize .....                  | 255        |
| TSRVTabSet.MinTabSize .....                  | 255        |
| TSRVTabSet.MirrorText .....                  | 255        |
| TSRVTabSet.OppositeTabPosition .....         | 256        |
| TSRVTabSet.ScrollButtonWidth .....           | 256        |
| TSRVTabSet.ScrrollerMenu .....               | 256        |
| TSRVTabSet.ScrrollerPosition .....           | 256        |
| TSRVTabSet.ScrollingDelay .....              | 257        |
| TSRVTabSet.SkinManager .....                 | 257        |
| TSRVTabSet.SkinSchemeIndex .....             | 257        |
| TSRVTabSet.Spacings .....                    | 257        |
| TSRVTabSet.SRVControlStyle .....             | 258        |
| TSRVTabSet.TabHeight .....                   | 258        |
| TSRVTabSet.TabIndex .....                    | 258        |
| TSRVTabSet.Tabs .....                        | 259        |
| TSRVTabSet.TextAlignH .....                  | 259        |
| TSRVTabSet.TextAlignV .....                  | 259        |
| Methods .....                                | 259        |
| TSRVTabSet.ScrollToTab .....                 | 259        |
| Events .....                                 | 260        |
| TSRVTabSet.OnChange .....                    | 260        |
| TSRVTabSet.OnCloseTab .....                  | 260        |
| TSRVTabSet.OnDraw... .....                   | 260        |
| TSRVTabSet.OnMoveTab .....                   | 261        |
| Classes of properties .....                  | 261        |
| TSRVCloseButton .....                        | 262        |
| Properties .....                             | 262        |
| TSRVCloseButton.AllowCloseLastTab .....      | 263        |
| TSRVCloseButton.Height .....                 | 263        |
| TSRVCloseButton.HorizontalAlign .....        | 263        |
| TSRVCloseButton.IndentX .....                | 263        |
| TSRVCloseButton.IndentY .....                | 263        |
| TSRVCloseButton.VerticalAlign .....          | 264        |
| TSRVCloseButton.Visibility .....             | 264        |
| TSRVCloseButton.Visible .....                | 264        |
| TSRVCloseButton.Width .....                  | 264        |

|   |            |
|---|------------|
| TSRVTabItem .....                         | 264        |
| Properties .....                          | 265        |
| TSRVTabItem.Enabled.....                  | 265        |
| TSRVTabItem.ImageIndex .....              | 265        |
| TSRVTabItem.SkinFontIndex.....            | 265        |
| TSRVTabItem.SkinSchemeIndex.....          | 266        |
| TSRVTabItem.Tag.....                      | 266        |
| TSRVTabItem.Text.....                     | 266        |
| TSRVTabItem.Visible.....                  | 266        |
| TSRVTabCollection .....                   | 267        |
| <b>4..Types .....</b>                     | <b>267</b> |
| <b>Custom drawing types .....</b>         | <b>268</b> |
| TSRVDrawScrollBarAreaEvent type.....      | 268        |
| TSRVDrawScrollBarEvent type.....          | 268        |
| TSRVDrawState, TSRVDrawStates types ..... | 269        |
| TSRVDrawTabEvent type.....                | 269        |
| TSRVDrawTabSetEvent                       |            |
| type .....                                | 270        |
| TSRVPOnPaint type.....                    | 270        |
| <b>ToolBar types .....</b>                | <b>271</b> |
| TSRVAlignButton type.....                 | 271        |
| TSRVAlignText type.....                   | 271        |
| TSRVButtonCollection.....                 | 271        |
| TSRVClickButtonEvent type.....            | 272        |
| TSRVTool Button.....                      | 272        |
| Properties .....                          | 272        |
| TSRVTool Button.AllowAllUp.....           | 273        |
| TSRVTool Button.Caption.....              | 273        |
| TSRVTool Button.Enabled.....              | 273        |
| TSRVTool Button.Down.....                 | 273        |
| TSRVTool Button.Hint.....                 | 273        |
| TSRVTool Button.GroupIndex.....           | 274        |
| TSRVTool Button.ImageIndexDisable.....    | 274        |
| TSRVTool Button.ImageIndexDown.....       | 274        |
| TSRVTool Button.ImageIndexMove.....       | 274        |
| TSRVTool Button.ImageIndexNormal .....    | 275        |
| TSRVTool Button.ShowHint.....             | 275        |
| TSRVTool Button.Tag.....                  | 275        |
| TSRVTool Button.Visible.....              | 275        |
| TSRVControlStyle .....                    | 276        |
| TSRVFloatProperty type .....              | 276        |
| TSRVIntProperty type .....                | 277        |
| TSRVMouseEvent type .....                 | 277        |
| TSRVPageFormat type .....                 | 278        |
| TSRVTHAlign, TSRVTVAlign types .....      | 281        |
| <b>5..Global variables .....</b>          | <b>281</b> |

## Part II ScaleRichView Actions 282

|   |            |
|---|------------|
| <b>1..Localization .....</b>                      | <b>283</b> |
| <b>2..Basic actions .....</b>                     | <b>284</b> |
| <b>TsrvAction .....</b>                           | <b>284</b> |
| Properties .....                                  | 285        |
| TsrvAction.Control .....                          | 285        |
| <b>TsrvCustomActionThatNeedsDocProps .....</b>    | <b>285</b> |
| Properties .....                                  | 286        |
| TsrvCustomActionThatNeedsDocProps.ActionSave..... | 286        |

|   |            |
|---|------------|
| <b>3..Print, preview, page setup .....</b>        | <b>286</b> |
| <b>TsrvActionCustomOrientation .....</b>          | <b>286</b> |
| <b>TsrvActionLineNumbers .....</b>                | <b>287</b> |
| <b>TsrvActionOrientationLandscape .....</b>       | <b>287</b> |
| <b>TsrvActionOrientationPortrait .....</b>        | <b>288</b> |
| <b>TsrvActionPageFormat .....</b>                 | <b>289</b> |
| Properties .....                                  | 289        |
| TsrvActionPageFormat.PageFormat.....              | 290        |
| TsrvActionPageFormat.PageHeight.....              | 290        |
| TsrvActionPageFormat.PageWidth.....               | 290        |
| TsrvActionPageFormat.Units .....                  | 290        |
| <b>TsrvActionPageSetup .....</b>                  | <b>291</b> |
| Properties .....                                  | 291        |
| TsrvActionPageSetup.PageFormats*.....             | 292        |
| Events .....                                      | 293        |
| TsrvActionPageSetup.OnChange.....                 | 293        |
| <b>TsrvActionPreview .....</b>                    | <b>293</b> |
| <b>TsrvActionPrint .....</b>                      | <b>293</b> |
| Properties .....                                  | 294        |
| TsrvActionPrint.Title .....                       | 294        |
| <b>TsrvActionQuickPrint .....</b>                 | <b>294</b> |
| Properties .....                                  | 295        |
| TsrvActionQuickPrint.Title.....                   | 295        |
| <b>TsrvCustomPrintAction .....</b>                | <b>296</b> |
| <b>4..View .....</b>                              | <b>296</b> |
| <b>TsrvActionCustomLayout .....</b>               | <b>296</b> |
| Events .....                                      | 298        |
| TsrvActionCustomLayout.OnExecuted.....            | 298        |
| <b>TsrvActionCustomZoom .....</b>                 | <b>299</b> |
| <b>TsrvActionLayoutDraft .....</b>                | <b>299</b> |
| Properties .....                                  | 301        |
| TsrvActionLayoutDraft.HPadding.....               | 301        |
| <b>TsrvActionLayoutPrint .....</b>                | <b>301</b> |
| Properties .....                                  | 303        |
| TsrvActionLayoutPrint.Color.....                  | 303        |
| TsrvActionLayoutPrint.FooterVisible.....          | 303        |
| TsrvActionLayoutPrint.HeaderVisible.....          | 303        |
| TsrvActionLayoutPrint.HPadding.....               | 304        |
| <b>TsrvActionLayoutRead .....</b>                 | <b>304</b> |
| Properties .....                                  | 305        |
| TsrvActionLayoutRead.Color.....                   | 306        |
| TsrvActionLayoutRead.HidePageBackgroundImage..... | 306        |
| TsrvActionLayoutRead.HPadding.....                | 306        |
| TsrvActionLayoutRead.PageColor.....               | 306        |
| TsrvActionLayoutRead.PageFlipEffect.....          | 306        |
| <b>TsrvActionLayoutSideToSide .....</b>           | <b>307</b> |
| Properties .....                                  | 308        |
| TsrvActionLayoutSideToSide.Color.....             | 309        |
| TsrvActionLayoutSideToSide.FooterVisible.....     | 309        |
| TsrvActionLayoutSideToSide.HeaderVisible.....     | 309        |
| TsrvActionLayoutSideToSide.HPadding.....          | 309        |
| TsrvActionLayoutSideToSide.NoPageSpacing.....     | 309        |
| TsrvActionLayoutSideToSide.PageFlipEffect.....    | 310        |
| <b>TsrvActionLayoutWeb .....</b>                  | <b>310</b> |
| <b>TsrvActionThumbnails .....</b>                 | <b>311</b> |
| Properties .....                                  | 312        |
| TsrvActionThumbnails.PageScroll.....              | 312        |

|   |            |
|---|------------|
| Events .....  | 312        |
| TsrvActionThumbnails.OnExecuted.....                  | 312        |
| <b>TsrvActionZoom .....</b>                           | <b>313</b> |
| Properties .....                                      | 313        |
| TsrvActionZoom.ZoomPercent.....                       | 313        |
| <b>TsrvActionZoomFullPage .....</b>                   | <b>314</b> |
| <b>TsrvActionZoomPageWidth .....</b>                  | <b>314</b> |
| <b>5. Editing .....</b>                               | <b>315</b> |
| TsrvActionEditHeader .....                            | 315        |
| TsrvActionEditFooter .....                            | 315        |
| TsrvActionEditMain .....                              | 316        |
| <b>6. Notes .....</b>                                 | <b>316</b> |
| TsrvActionCustomInsertSidenote .....                  | 317        |
| Properties .....                                      | 317        |
| TsrvActionCustomInsertSidenote.BoxPosition.....       | 318        |
| TsrvActionCustomInsertSidenote.BoxProperties.....     | 318        |
| TsrvActionCustomInsertSidenote.StyleTemplateName..... | 318        |
| TsrvActionEditNote .....                              | 319        |
| TsrvActionInsertEndnote .....                         | 319        |
| TsrvActionInsertFootnote .....                        | 320        |
| TsrvActionInsertNote .....                            | 321        |
| Properties .....                                      | 322        |
| TsrvActionInsertNote.Font.....                        | 322        |
| TsrvActionInsertSidenote .....                        | 322        |
| Properties .....                                      | 323        |
| TsrvActionInsertSidenote.RefStyleTemplateName.....    | 324        |
| TsrvActionInsertTextBox .....                         | 324        |
| TsrvActionReturnToNote .....                          | 325        |

## Part III SRVControls

**325**

|                                     |            |
|-------------------------------------|------------|
| <b>1. Ancestor classes .....</b>    | <b>327</b> |
| TCustomSRVEdit .....                | 327        |
| TCustomSRVGraphicControl .....      | 329        |
| TCustomSRVPanel .....               | 330        |
| TSRVCanvasControl .....             | 331        |
| TSRVCustomComboBox .....            | 332        |
| TSRVCustomControl .....             | 335        |
| TSRVCustomListBox .....             | 336        |
| TSRVEditControl .....               | 339        |
| TSRVGraphicControl .....            | 339        |
| TSRVWinControl .....                | 340        |
| <b>2. Main Controls .....</b>       | <b>342</b> |
| TSRVButton .....                    | 343        |
| TSRVCheckBox .....                  | 345        |
| TSRVComboBox .....                  | 346        |
| TSRVEdit .....                      | 349        |
| TSRVGroupBox .....                  | 352        |
| TSRVImagesScroll .....              | 354        |
| TSRVLabel .....                     | 358        |
| TSRVListBox .....                   | 359        |
| TSRVMemo .....                      | 362        |
| TSRVPaintBox .....                  | 364        |
| TSRVPanel .....                     | 365        |
| TSRVRadioButton .....               | 367        |
| <b>3. Data-Aware Controls .....</b> | <b>368</b> |
| TSRVDBCheckBox .....                | 369        |

---

|   |                |
|---|----------------|
| TSRVDBComboBox .....                                | 370            |
| TSRVDBEdit .....                                    | 371            |
| TSRVDBListBox .....                                 | 372            |
| TSRVDBMemo .....                                    | 373            |
| TSRVDBText .....                                    | 374            |
| <b>4. Classes of properties .....</b>               | <b>375</b>     |
| TSRVBoxItemCollection, TSRVLBoxItemCollection ..... | 375            |
| TSRVImageScrollCollection .....                     | 375            |
| TSRVBoxItem, TSRVLBoxItem .....                     | 375            |
| TSRVImageScrollItem .....                           | 379            |
| <br><b>Index .....</b>                              | <br><b>381</b> |

# 1 ScaleRichView

Version 11+ (What's new? <sup>20</sup>)

## Introduction

---

ScaleRichView is a package of Delphi/C++Builder/Lazarus components.

It uses:

- TRichView Package ([www.trichview.com](http://www.trichview.com))
- RichViewActions ([www.trichview.com/resources/actions/](http://www.trichview.com/resources/actions/)).

TRichView and RichViewActions must be installed in order to use ScaleRichView.

## Components

---



TSVRichViewEdit <sup>40</sup> allows editing rich text documents in WYSIWYG mode. It supports scaling and different view modes, including a page-view mode.



TDBSRichViewEdit <sup>169</sup> is a data-aware version of WYSIWYG editing component.



TSVRuler <sup>187</sup> is a horizontal or vertical ruler designed to work with TSVRichViewEdit. It is inherited from TRVRuler component included in RichViewActions.



TSVRToolBar <sup>189</sup> is a simplified analog of TToolBar. Instances of TSVRToolbar are used in TSVRichViewEdit: they allow adding buttons in scrollbar area.



TSVRToolWindow <sup>197</sup> uses the TSVRToolBar and displays it in a popup window.



TSVRPageScroll <sup>174</sup> displays thumbnails of pages for documents prepared in TSVRichViewEdit component.



TSVRPrint <sup>204</sup> and TSVRPreview <sup>223</sup> are components for advanced printing of documents prepared in TSVRichViewEdit. For example, they allow printing posters (e.g. A3 page on A4 printer).



TSVRSkinManager <sup>228</sup> allows applying skins to scrollbars of TSVRichViewEdit <sup>40</sup> /TDBSRichViewEdit <sup>169</sup>, to TSVRScrollBar <sup>243</sup>, to TSVRTabSet <sup>249</sup>.



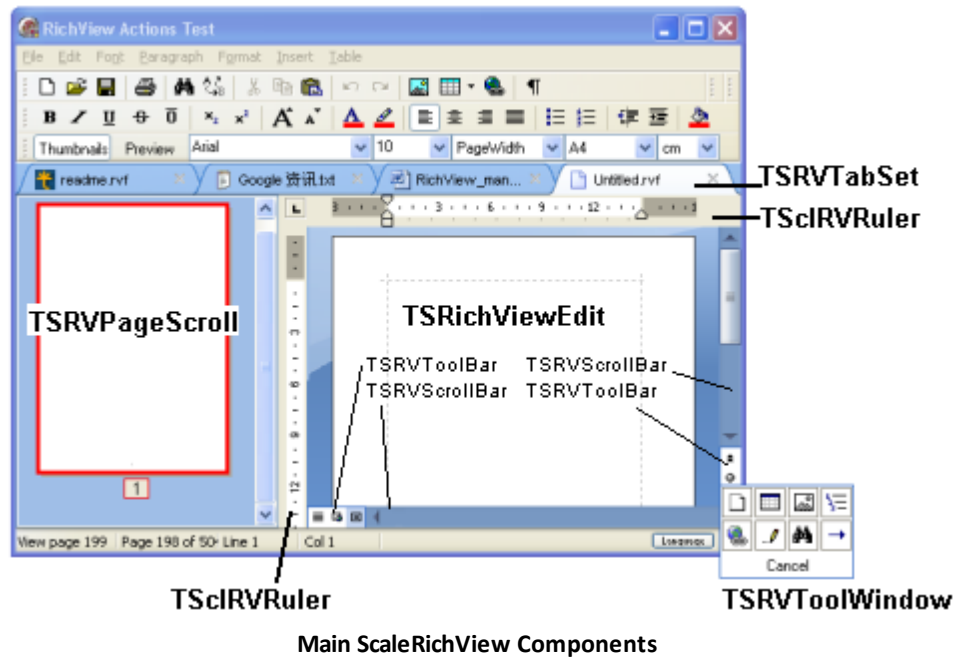
TSVRTabSet <sup>249</sup> – advanced skinnable tabs, can be used to implement a tabbed multiple document interface.



TSVRScrollBar <sup>243</sup> – skinnable scrollbar. This component is used for scrollbars in TSVRichViewEdit, but may be useful as a separate component as well.

## Example

---



Main ScaleRichView Components

In the screenshot above, you can see:

- `TSRichViewEdit`<sup>(40)</sup> component;
- `TSRVPPageScroll`<sup>(174)</sup> component;
- two `TScIRVRuler`<sup>(187)</sup> components;
- a window displayed by `TSRVToolWindow`<sup>(197)</sup>.

You can also see three instances of `TSRVToolBar`<sup>(189)</sup>: two toolbars embedded in a scrollbar area of `TSRichViewEdit`<sup>(40)</sup> component, and a toolbar inside a `TSRVToolWindow`<sup>(197)</sup>'s window. These three toolbars are not separate components, they are owned by `TSRichViewEdit`<sup>(40)</sup> and `TSRVToolWindow`<sup>(197)</sup>, and their properties are defined by their owners.

## 1.1 Version history

### Version History

- New after v12.0<sup>(20)</sup> (dark mode, GDI+ support in Lazarus)
- New in v12.0<sup>(20)</sup> (VCL themes, LiveBindings, HTML)
- New in v11.0<sup>(21)</sup> (improved zooming, share properties, end-user help)
- New in v10.0<sup>(22)</sup> (smooth scrolling, reading mode, improved bi-di, data-aware SRVControls)
- New in v9.0<sup>(23)</sup> (Lazarus, better zooming and page positioning, high DPI, major rewrite of SRVControls)
- New in v8.0<sup>(25)</sup> (printing options, system caret)
- New in v7.0<sup>(26)</sup> (Uniscribe, transparency, line numbers, setup)
- New in v6.0<sup>(27)</sup> (text boxes and sidenotes, smooth image scaling, touch screen support)
- New in v5.0<sup>(28)</sup> (style templates, zooming panel, animations, "smart popups")
- New in v4.0<sup>(30)</sup> (footnotes and endnotes, new actions, new printing modes)
- New in v3.0<sup>(33)</sup> (header and footer, printing actions, new SRVControls)
- New in v2.0<sup>(34)</sup> (new printing modes)

## 1.1.1 New after v12.0

### Compatibility issues

---

Changes in packages for Lazarus: designtime and db- packages are removed (their units are moved to main packages). Only two packages left: `srvpkg laz.lpk` and `srvcontrols laz.lpk` ("runtime + designtime" packages)

### Delphi and C++Builder

---

Delphi and C++Builder 12 and 13 are supported

Support of "Windows 64-bit (Modern)" platform in C++Builder 12+.

### Lazarus

---

SRVControls can use GDI+ drawing in Lazarus. See GDI+ notes <sup>(327)</sup>.

OnUTF8KeyPress event is supported.

### Background

---

The background properties of pages are now defined in a new way. Instead of `TSRichViewEdit.RVBackgroundBitmap` <sup>(61)</sup>, you should use `RVBackgroundPicture` <sup>(61)</sup>, which can contain any image type supported in Delphi. Instead of `TSRichViewEdit.RVBackgroundStyle` <sup>(61)</sup>, you should now use `RVBackgroundProperties` <sup>(61)</sup>, which specify the size, position, and repetition of the background image in a much more flexible way.

The old properties are supported for backward compatibility.

### Appearance

---

New property to implement a "dark mode": `TSRichViewEdit.ViewProperty` <sup>(69)</sup>.`DarkMode` <sup>(157)</sup>, `DarkModeUI` <sup>(157)</sup>.

You can define appearance of scrollbars in `TSRichViewEdit` using `ScrollBarControlStyle` <sup>(65)</sup> property.

All `SRVControls` have `DarkMode` and `AutoDarkMode` properties too. By default, their `AutoDarkMode` = `True`, so, when inserted in an editor (`TRichView` or `TSRichViewEdit`), they are drawn according to the editor's `DarkMode`. Otherwise, their own `DarkMode` property is used.

## 1.1.2 New in v12.0

### Compatibility issues

---

The default value of `TSRichViewEdit.Color` is changed from `$0099A8AC` to `clBtnShadow`.

The type of `Sender` parameter in `OnSaveCustomFormat` <sup>(119)</sup> and `OnLoadCustomFormat` <sup>(111)</sup> events is changed from `TDBSRichViewEdit` <sup>(169)</sup> to `TSRichViewEdit` <sup>(40)</sup>.



---

## Printing metafiles as bitmaps

---

New optional parameter NoMetafiles in TSRichViewEdit.DrawPage<sup>(77)</sup>, DrawMetafile<sup>(76)</sup>, PrintAll<sup>(92)</sup>, PrintRange<sup>(93)</sup>, PrintCurrent<sup>(93)</sup>.

New property: TSRVPrint.NoMetafiles<sup>(211)</sup>.

## VCL themes

---

TSRVScrollBar<sup>(243)</sup> supports VCL themes (in Delphi XE2 and newer). Improved VCL themes support in TSRichViewEdit.

## DataBindings

---

A new TSRichViewEdit.Document<sup>(51)</sup> property can be linked to a database field (TBlobField) using LiveBindings (in Delphi XE2 and newer). The following events are moved from TDBSRichViewEdit<sup>(169)</sup> to TSRichViewEdit<sup>(40)</sup> and work for LiveBindings as well: OnSaveCustomFormat<sup>(119)</sup>, OnLoadCustomFormat<sup>(111)</sup>, OnNewDocument<sup>(113)</sup>, OnLoadDocument<sup>(111)</sup>.

SRVControls support DataBindings as well: TSRVEdit<sup>(349)</sup> and TSRVCheckBox<sup>(345)</sup> as live binding editors; TSRVLabel<sup>(358)</sup>, TSRVButton<sup>(343)</sup>, TSRVPanel<sup>(365)</sup> as sources.

## HTML

---

ScaleRichView supports HTML loading, including storing documents as HTML in databases by TDBSRichViewEdit<sup>(169)</sup>.

### 1.1.3 New in v11.0

#### Compatibility issues

---

TSRichViewEdit.SelectAll<sup>(97)</sup> now selects in ActiveEditor<sup>(47)</sup> instead of RichViewEdit<sup>(60)</sup>.

The following properties of TSRichViewEdit were removed: ExternalRV, ExternalRVH, ExternalRVF, ExternalRVN. External TRVStyle components still can be used.

## RAD Studio 11 Alexandria

---

Delphi and C++Builder 11 Alexandria are supported.

## Improved zooming

---

If TSRichViewEdit.PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup> = *True*, TSRichViewEdit.ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> is applied to document content (not to pages). In this mode, to apply changed ZoomPercent, the editor needs to reformat the whole document. Because of this, for large documents, fast multiple changes of ZoomPercent may be a problem.

Since this version, when zooming is initiated by a mouse wheel<sup>(162)</sup> or a track bar on a zooming panel<sup>(168)</sup>, the editor enters in a special zooming mode. The zooming percent is displayed using TSRichViewEdit.ViewProperty<sup>(69)</sup>.ZoomFont<sup>(166)</sup>. ZoomPercent is assigned when the user finishes using a mouse wheel / trackbar.

## Shared TRVStyle properties

---

TRVStyle of internal TRichViewEdit objects in TSRichViewEdit component now share all properties (except for collections of text, paragraph, and list styles). If you change a property of one of these TRVStyle objects, you change this property in all of them.

For example, if you assign `SRichViewEdit1.RichViewEdit(60).Style.SelOpacity`, this change affects not only the main document editor, but also editors of headers, footers, and notes.

## End-user help

---

HelpContext and HelpKeyword are assigned to a page setup dialog<sup>(291)</sup>.

### 1.1.4 New in v10.0

This topic includes changes made in v10.1.

## UI

---

New UI translation: Slovenian.

## Smooth scrolling

---

A new optional parameter SmoothScroll is added to the following TSRichViewEdit methods:

- ScrollToItem<sup>(96)</sup>, ScrollToCaret, ScrollCaretToCenter<sup>(96)</sup>,
- StartEditNote<sup>(99)</sup>, ReturnToNote<sup>(95)</sup>,
- FirstCurPage<sup>(82)</sup>, NextCurPage<sup>(91)</sup>, PriorCurPage<sup>(95)</sup>, LastCurPage<sup>(90)</sup>.

It is *False* by default.

Since this version, the actions use smooth scrolling:

- when scrolling to a checkpoint,
- when editing a note or a text box (switching between a note document and a note itself)
- when activating headers/footers.

New property of TSRVPageScroll<sup>(174)</sup>: SmoothScroll<sup>(182)</sup>.

## Read mode


---


New property of TSRichViewEdit<sup>(40)</sup>: ReadModeProperty<sup>(59)</sup>. It allows displaying document in a way convenient for reading.

Features:

- always displaying full pages
- auto page size or auto zooming
- page flipping effect
- options to overriding page color and for hiding page background image

New actions for switching to a read mode:

 TsrVActionLayoutSideToSide<sup>(307)</sup> (auto page zooming)

 TsrVActionLayoutRead<sup>(304)</sup> (auto page size)

## Improved BiDiMode support

If `TSRichViewEdit`<sup>(40)</sup>.`BiDiMode`<sup>(48)</sup> = `rvbdRightToLeft`, the following changes are applied (in addition to placing a vertical scrollbar at the left side):

- when pages are arranged in columns, pages in each row are ordered from right to left
- line selection with the mouse works on the right page margin

## SRVControls

SRVControls are ported to Lazarus, Windows platform (except for `TSRVMediaPlayer`).

New data-aware versions of SRVControls<sup>(325)</sup>:



`TSrvDBCheckBox`<sup>(369)</sup> – data-aware check box;



`TSrvDBComboBox`<sup>(346)</sup> – data-aware combo box;



`TSrvDBEdit`<sup>(371)</sup> – data-aware plain-text one-line editor;



`TSrvDBListBox`<sup>(372)</sup> – data-aware scrollable list of items;



`TSrvDBMemo`<sup>(373)</sup> – data-aware plain-text multi-line editor;



`TSrvDBText`<sup>(374)</sup> – data-aware text label.

New property for `TSRVListBox`<sup>(359)</sup> and `TSRVComboBox`<sup>(346)</sup>: `UseSelectedImagesInHotState`. It specifies which image indexes are used for highlighted items.

## Other changes

### New events and properties

In `TSRichViewEdit`<sup>(40)</sup>:

- `OnTextFound`<sup>(121)</sup> event
- `MarkdownProperties`<sup>(54)</sup> property
- `MinPrintedOffInItem`<sup>(56)</sup> and `MaxPrintedOffInItem`<sup>(55)</sup> properties allow defining a range of items to print with higher precision

In `TSRVToolbar`:

- `OnDrawButtonBackground`<sup>(196)</sup> event

## 1.1.5 New in v9.0

### Compatibility issues

#### Changes in ScaleRichView

The following properties were moved:

- `AlignPageH`<sup>(144)</sup>, `AlignPageV`<sup>(145)</sup> from `TSRichViewEdit.ViewProperty`<sup>(69)</sup> to `PagePosProperty`<sup>(58)</sup>
- `MaxPageColCount`<sup>(147)</sup> from `TSRichViewEdit.ViewProperty`<sup>(69)</sup> to `PagePosProperty`<sup>(58)</sup>

The following properties were removed:

- from `TSRichViewEdit.PageProperty`<sup>(58)</sup>: `BoundLeftRight` (replaced by `PagePosProperty`<sup>(58)</sup>.`HPadding`<sup>(146)</sup>), `PageBreakHeight`, `MinPageBreakHeight`, `MaxPageBreakHeight` (replaced by `PagePosProperty`<sup>(58)</sup>.`PageVSpacing`<sup>(147)</sup>)
- from `TsrvActionLayoutDraft`<sup>(299)</sup>: `BoundLeftRight` (replaced by `HPadding`<sup>(301)</sup>)
- from `TsrvActionLayoutPrint`<sup>(301)</sup>: `BoundLeftRight` (replaced by `HPadding`<sup>(304)</sup>)

### Changes in SRVControls

`SRVControls`<sup>(325)</sup> and `TSRVTabSet`<sup>(249)</sup> have a new default appearance (see `SRVControlStyle` properties)

In `TSRVTabSet`<sup>(249)</sup>, `TextAlignH`<sup>(259)</sup> and `TextAlignV`<sup>(259)</sup> do not affect icons positions anymore; they depend only on `BiDiMode` and `MirrorText`<sup>(255)</sup>.

In `TSRVEdit`<sup>(349)</sup> and `TSRVComboBox`<sup>(346)</sup>, types of parameters are changed in `OnEditHints` (Text and `EditHints` parameters) and `OnCloseHints` (NewText parameter).

### New development environments

- RAD Studio 10.3 compatibility
- Lazarus compatibility (Windows 32 and Windows 64 platforms)

### Zooming, high DPI

`ScaleRichView` does not assign the global variable `RichViewPixelsPerInch = 96` anymore. Moreover, it's recommended to leave `RichViewPixelsPerInch` at the default state, to allow zooming relative to the actual screen DPI.

Zooming is changed: previously, pages were scaled relative to 96 DPI. Since this version, pages are scaled relative to the screen DPI. Since this version, `*100Pix` properties do not necessary correspond to 100% zoom: now they correspond to 100% zoom in 96 DPI.

Zooming panel appearance is changed in the mode `TSRichViewEdit`<sup>(40)</sup>.`ViewProperty`<sup>(69)</sup>.`IconStyle`<sup>(160)</sup> = `srvisFlat`.

`TSRichViewEdit` supports high-DPI display modes, "per monitor" and "per monitor v2" modes (if they are supported by an application).

You can specify whether toolbars should scale image according to the screen DPI: new `ScaleImagesForDPI` property is added to `TSRVToolbar`<sup>(189)</sup>, `TSRVToolWindow`<sup>(197)</sup>, `TSRichViewEdit`<sup>(40)</sup>.`MenuHorizontal`<sup>(55)</sup> and `TSRichViewEdit.MenuVertical`<sup>(56)</sup>.

New demo projects for RAD Studio 10.3 are added: `Delphi\ActionTestTabs_MultiRes` and `CBuilderUnicode\ActionTestTabs_MultiRes`. They use virtual image lists (containing 16x16, 32x32 and selected 64x64 images) and support "per monitor v2" mode.

### Page positioning

New `TSRichViewEdit` property is added: `PagePosProperty`<sup>(58)</sup>: `TSRVPagePositionProperty`<sup>(143)</sup>. It contains sub-properties controlling arrangement of pages in the editor.

### SRVControls

All additional components (that can be inserted in `TSRichViewEdit` or used on a form) are updated: `TSRVTabSet`<sup>(249)</sup>, `TSRVScrollBar`<sup>(243)</sup>, and all `SRVControls`<sup>(325)</sup>.

Main improvements:

- the components support high-dpi screen modes;
- the components support bi-directional texts (BiDiMode property is published and completely supported);
- all text properties are Unicode (TRVUnicodeString) in controls in all versions of Delphi;
- new default appearance: modern flat look, smooth antialiased lines (antialiasing requires Delphi XE2+);
- old appearance can be returned using SRVControlStyle property (the only control that has the old appearance by default is TSRVScrollBar<sup>(243)</sup>); but the old appearance is improved as well;
- TSRVMemo<sup>(362)</sup> and TSRVEdit<sup>(349)</sup> are rewritten completely; since this version, they are not based on standard memo and edit controls, they are implemented from scratch. TSRVMemo<sup>(362)</sup> supports scroll bars;
- SkinFontIndex property is added to TSRVEdit<sup>(349)</sup>, TSRVMemo<sup>(362)</sup>, TSRVButton<sup>(343)</sup>, TSRVCheckBox<sup>(345)</sup>, TSRVRadioButton<sup>(367)</sup>, TSRVPanel<sup>(365)</sup>, TSRVGroupBox<sup>(352)</sup> to allow using fonts<sup>(232)</sup> defined in TSRVSkinManager<sup>(228)</sup> component.

New property for TSRVTabSet<sup>(249)</sup>: OppositeTabPosition<sup>(256)</sup>

## Other changes

- Type of string properties is changed to TRVUnicodeString.
- TSRVPrint<sup>(204)</sup>.PrintMode<sup>(214)</sup> = *srvpTiles* works differently if ScaleRichView document does not fit the paper sheet. Now it scales down the page to fit paper, and tries to print multiple copies of this scaled down page.
- new method TSRichViewEdit<sup>(40)</sup>.ConvertToEMU<sup>(75)</sup>

### 1.1.6 New in v8.0

This topic includes changes added in v8.1.

## Compatibility issues

SRVItemResizeMode global variable is removed.

Since this version, colors of VCL themes are not used for page backgrounds only if TSRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup>.UseVCLThemes<sup>(165)</sup>.

Themes of Windows and VCL do not affect printing any more.

The default value TSRichViewEdit<sup>(40)</sup>.PageProperty<sup>(58)</sup>.CaretPen<sup>(136)</sup>.Width is changed from 2 to 0.

## Printing

New parameters are added in TSRVPrint<sup>(204)</sup>.Print<sup>(219)</sup> and PrintPages<sup>(221)</sup> methods. They allow printing odd/even pages, and printing in reverse order.

Themes of Windows and VCL do not affect printing any more (*clWindowText* is always printed black, *clWindow* is always printed white, etc.)

## Editor appearance

---

New properties of `TSRichViewEdit`<sup>(40)</sup>.`ViewProperty`<sup>(69)</sup>:

- `ZoomPanelUsesGradient`<sup>(168)</sup>.
- `UseVCLThemes`<sup>(165)</sup>

## Drag and drop

---

New properties of `TSRichViewEdit`: `AcceptDragDropFormats`, `AcceptPasteFormats`<sup>(47)</sup>.

New events of `TSRichViewEdit`: `OnBeforeOleDrop`, `OnAfterOleDrop`<sup>(104)</sup>.

## TSRVPageScroll<sup>(174)</sup>

---

More efficient repainting: new `ChangeDelay`<sup>(178)</sup> property. Ability to display smoothed thumbnails of page images, `SmoothThumbnails`<sup>(183)</sup>.

## Caret

---

Since this version, the editor uses the system caret width by default. You can change the caret thickness by assigning a positive value to `TSRichViewEdit`<sup>(40)</sup>.`PageProperty`<sup>(58)</sup>.`CaretPen`<sup>(136)</sup>.`Width`.

A wide caret is not centered at the insertion position any more, it is positioned according to the global `RichViewEditCaretPosition` variable (RVERVData unit); by default, the caret is positioned at the right side of the insertion position for left-to-right items, and to the left side if right-to-left items.

Corners of a wide caret are not rounded any more.

## Other changes

---

Packages are separated into runtime and designtime packages.

A new parameter `AllowFloating` is added to `TSRichViewEdit`'s methods `GetItemCoords100`<sup>(85)</sup>, `GetItemBounds`, `GetItemBounds100`<sup>(84)</sup>.

## 1.1.7 New in v7.0

### Compatibility issues

---

- A zooming panel<sup>(168)</sup> is always drawn using the parent `TSRichViewEdit`'s `BackgroundProperty`<sup>(48)</sup>. It does not use `TSRichViewEdit.Color` any more.
- Previously, the default values of `TSRichViewEdit`'s `MenuVertical`<sup>(56)</sup>.`ButtonHeight`<sup>(128)</sup> and `MenuHorizontal`<sup>(55)</sup>.`ButtonWidth`<sup>(129)</sup> were ignored, a size of buttons was autocalculated. In this update, autocalculation is removed.
- A new parameters for `TSRichViewEdit.DrawPage`<sup>(77)</sup>: `UseWordPainters`, `ForMetafile`.
- The default value of `TSRichViewEdit`'s `MenuVertical`<sup>(56)</sup>.`ButtonHeight`<sup>(128)</sup> is changed from 26 to 20.

## Uniscribe

---

Since this version, `TSRichViewEdit` draws and prints text using Uniscribe. You can switch back to Windows API using `TextEngine` <sup>(68)</sup> property.

Printing output may be incompatible with metafiles (so it may cause problems to some virtual printers). A metafile-compatible mode can be set by:

- new optional parameter of `TSRichViewEdit.PrintAll` <sup>(92)</sup>, `PrintCurrent` <sup>(93)</sup>, `PrintRange` <sup>(93)</sup>, `DrawPage` <sup>(77)</sup> methods;
- `TSRVPrint.MetafileCompatibility` <sup>(211)</sup> property
- `TRVAControlPanel.MetafileCompatibility` property (used in `TsrvActionPrint` <sup>(293)</sup> and `TsrvActionQuickPrint` <sup>(294)</sup>).

## Transparency

---

Since this version, `TSRichViewEdit` supports transparent objects, including pictures and semitransparent backgrounds of paragraphs, tables and text boxes.

Transparency is supported both on the screen and when printing. The most of printers do not support transparency, so `TSRichViewEdit` draws semitransparent areas in bitmaps and prints this bitmap.

Previously, `TSRichViewEdit` only were able to draw transparent pictures on the screen.

## Line numbers

---

`TSRichViewEdit` can show line numbers. New property: `LineNumberProperty` <sup>(54)</sup>.

New action:

 `TsrvActionLineNumbers` <sup>(287)</sup> shows or hides line numbers.

## Installation and directory structure

---

Starting from this update, `ScaleRichView` is installed automatically in Delphi and C++Builder IDE.

The new installer installs the components in Delphi and C++Builder, both for 32-bit and 64-bit platforms (if available). The installer adds all necessary paths to RAD Studio library.

Source code is moved to "Source" folder, inc-file is moved to "Source\Include" folder. Demo projects that use `SRVControls` are moved to "Demos" folder of `SRVControls`.

This help file is integrated in RAD Studio IDE (for XE8+)

### 1.1.8 New in v6.0

#### Compatibility issues

---

- Since this version, `TSRichViewEdit` must be formatted by its method `Format` <sup>(82)</sup>. You cannot format it by formatting `RichViewEdit` <sup>(60)</sup>, `RVHeader` <sup>(64)</sup> and `RVFooter` <sup>(62)</sup>, or by calling `SetRVMargins` <sup>(98)</sup>.
- Headers and footers are stored in Subdocuments <sup>(66)</sup>. `RVHeader` <sup>(64)</sup> and `RVFooter` <sup>(62)</sup> are used only for editing Subdocuments.

- It's not necessary to call `UpdateNote` or `UpdateNoteEx` after inserting a note or a text box.
- The following properties of `TSRVViewProperty`<sup>(154)</sup> are removed: `HeaderTitle`, `FooterTitle`, `MainTitle`. They are replaced with `Texts`<sup>(163)</sup>.
- When adding a page break between table rows, `TSRichViewEdit`<sup>(40)</sup> reserves space for table borders (previously, they might be truncated); so documents containing tables may be paginated differently.
- A different font is used in a zooming panel inside `TSRichViewEdit`: `TSRichViewEdit.ViewProperty`<sup>(69)</sup>.`ZoomPanelFont`<sup>(167)</sup> (`TSRichViewEdit.Font` was used in previous versions)

## Changes

`TSRichViewEdit` allows editing sidenotes (notes in a floating box) and text box items.

Different headers and footers for the first page, for odd and even pages. Headers and footers are stored in Subdocuments<sup>(66)</sup>. `RVHeader`<sup>(64)</sup> and `RVFooter`<sup>(62)</sup> are used only for editing Subdocuments.

Smooth image scaling. In addition to images inside documents, `TSRichViewEdit.BackgroundProperty`<sup>(48)</sup>.`Picture`<sup>(126)</sup> is scaled when displayed stretched.

`rvoHideReadOnlyCaret` option in `TSRichViewEdit.RVEditorOptions`<sup>(62)</sup> is supported.

Accurate drawing of table borders in places where a table is split across pages.

New properties of `TSRichViewEdit.ViewProperty`<sup>(69)</sup>: `ZoomPanelFont`<sup>(167)</sup> and `IconStyle`<sup>(160)</sup>.

New methods of `TSRichViewEdit`: `SetIntPropertyEd`, `SetFloatPropertyEd`<sup>(97)</sup>, `Reformat`<sup>(82)</sup>.

New properties of `TSRichViewEdit`: `CanUpdatePosition`<sup>(49)</sup>, `CurrentNoteParentEditor`<sup>(50)</sup>.

New events of `TSRichViewEdit`: `OnCaretMoving`<sup>(104)</sup>, `OnResizing`<sup>(117)</sup>

Touch screen support: panning in all visual components, two finger zooming and selection handles in `TSRichViewEdit`.

## Changes in actions

`TsrvActionPageSetup`<sup>(291)</sup>, `TsrvActionPageFormat`<sup>(289)</sup>, `TsrvActionCustomOrientation`<sup>(286)</sup> now applies changes as editing operations (undoable).

New actions:

- `TsrvActionInsertSidenote`<sup>(322)</sup>
- `TsrvActionInsertTextBox`<sup>(324)</sup>

`TsrvActionEditNote`<sup>(319)</sup> and `TsrvActionReturnToNote`<sup>(325)</sup> support sidenotes and text box items.

## 1.1.9 New in v5.0

### Compatibility issues

`TSRichViewEdit`<sup>(40)</sup>.`CurControl`<sup>(50)</sup> now works according to the documentation: this is a focused control inside `TSRichViewEdit`<sup>(40)</sup>. Previously, this property may be changed to the control at the



position of the caret. Use the new event `OnChangeCurrentControl`<sup>(105)</sup> to detect when this property is changed.

`TSRichViewEdit.ScrollToItem`<sup>(96)</sup> has new parameters. They are optional, but for old versions of C++Builder they must be specified.

A default value for `TSRVToolBar.Color`<sup>(193)</sup> and `TSRVToolWindow.Color`<sup>(200)</sup> is changed (from *c/White* to *c/Window*)

In the new version, pages can be arranged in columns. To return the old layout (one page in a row), you can assign `TSRichViewEdit.ViewProperty`<sup>(69)</sup>.`MaxPageColCount`<sup>(147)</sup> = 1.

`TSRichViewEdit.ViewProperty.ZoomPercent`<sup>(168)</sup> can be changed only in the range `TSRichViewEdit.ViewProperty.ZoomMin..ZoomMax`<sup>(166)</sup>.

The following code is not needed and must be removed from existing projects:

```
RVA_GetRichViewEditFromPopupComponent :=
    SRVGetRichViewEditFromPopupComponent;
RVA_GetRichViewEdit := SRVGetRichViewEdit
```

## Style templates

New properties of `TSRichViewEdit`: `UseStyleTemplates`<sup>(69)</sup> and `StyleTemplateInsertMode`<sup>(66)</sup>.

If `UseStyleTemplates`<sup>(69)</sup> = *True*, `TsrvActionInsertFootnote`<sup>(320)</sup> formats note characters using 'footnote reference' (or, if it does not exist, make it superscript like before), footnote text using 'footnote text' (or, if it does not exist, uses `TsrvActionInsertFootnote.Font`<sup>(322)</sup> like before).

`TsrvActionInsertEndnote`<sup>(319)</sup> uses 'endnote reference' and 'endnote text' styles.

## Simplification

You can assign `TSRichViewEdit` component to `Control` property of actions inherited from `TrvAction` (previously, only `TRichViewEdit` could be assigned). This assignment can be made at a design time.

Font combo-boxes (`TRVFontComboBox`, `TRVFontSizeComboBox`, `TRVFontCharsetComboBox`) do not require additional code any more. They work automatically, if `TSRichViewEdit` control is assigned to their `Editor` property (`TrvActionFontEx` action must be assigned to their `ActionFont` property as well).

You do not need to call the following code any more:

- `RVA_GetRichViewEditFromPopupComponent := SRVGetRichViewEditFromPopupComponent;`
- `RVA_GetRichViewEdit := SRVGetRichViewEdit;`
- `SRichViewEdit1.SetRVMargins` in `TRVAControlPanel.OnMarginsChanged`

## Animation

`TSRichViewEdit`<sup>(40)</sup> can play animated (gif) images. Animation is played in `ActiveEditor`<sup>(47)</sup>. New `AnimationMode`<sup>(48)</sup> property.

## Zooming and page arrangement

`TSRichViewEdit`<sup>(40)</sup> can display a zooming panel, if `ViewProperty`<sup>(69)</sup>.`ZoomPanelVisible`<sup>(168)</sup> = *True*.

In the new version, pages can be arranged in columns. `TSRichViewEdit.ViewProperty`<sup>(69)</sup>.`MaxPageColCount`<sup>(147)</sup> specifies the maximal count of columns.

## Page layout

---

TSRichViewEdit.PageProperty<sup>(58)</sup>.MirrorMargins<sup>(138)</sup> allow exchanging left and right margins on even pages.

## "Smart popups"

---

"Smart popups" are buttons that can be displayed for the current (at the position of the caret) item. New SmartPopupProperties<sup>(66)</sup>, SmartPopupVisible<sup>(66)</sup> property and OnSmartPopupClick<sup>(120)</sup> event.

## Other changes

---

New optional parameters for TSRichViewEdit.ScrollToItem<sup>(96)</sup>. New ScrollCaretToCenter<sup>(96)</sup> method.

New AlignButton<sup>(191)</sup> and Indent<sup>(195)</sup> properties of TSRVToolBar<sup>(189)</sup>.

New TSRichViewEdit.OnPrinting<sup>(116)</sup> event.

### 1.1.10 New in v4.0

#### Compatibility issues

---

- The type of Sender parameter was changed from TSRichViewEdit to TCustomRichViewEdit for TSRichViewEdit.OnStyleConversion<sup>(121)</sup> and OnParaStyleConversion<sup>(116)</sup> events.
- Two new parameters are added to TSRichViewEdit.OnPaintPage<sup>(115)</sup>: Prepaint and Printing.
- A new parameter is added to TSRichViewEdit.OnPaint<sup>(114)</sup>: PaintRect.
- The user can zoom the editor in/out using **Ctrl** + mouse wheel. So, if you assume that the editor has the fixed values of TSRichViewEdit.ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> and ViewProperty<sup>(69)</sup>.ZoomMode<sup>(166)</sup>, assign ViewProperty<sup>(69)</sup>.MouseWheelZoom<sup>(162)</sup> = *False*.
- TSRichViewEdit.ViewProperty<sup>(69)</sup>.HeaderFooterShade is renamed to ViewProperty<sup>(69)</sup>.MainDocumentShade<sup>(160)</sup>. ViewProperty<sup>(69)</sup>.HeaderFooterShade<sup>(158)</sup> now has a different meaning.
- TSRichViewEdit.PageProperty<sup>(58)</sup>.PageWidth<sup>(141)</sup> and PageHeight<sup>(139)</sup> do not depend on the document display modes any more. Properties of the same names were added in TSRichViewEdit.ViewProperty<sup>(69)</sup>.

## Units of measurement

---

All properties and parameters measured in TSRichViewEdit<sup>(40)</sup>.UnitsProgram<sup>(68)</sup> have changed their types from Extended to TRVLength. At design time, you can see units of measurement for these properties in the Object Inspector.

By default, properties of all RVStyles in TSRichViewEdit<sup>(40)</sup> are measured in pixels. To convert them to twips, call TSRichViewEdit<sup>(40)</sup>.ConvertToTwips<sup>(75)</sup>.

TsrvActionPageSetup<sup>(291)</sup> displays values in TRVAControlPanel.UnitsDisplay, not in TSRichViewEdit<sup>(40)</sup>.UnitsProgram<sup>(68)</sup>.

To do: a precision of drawing and printing is still up to one screen pixel. A higher precision is planned in newer version.

## Incremental printing

TSRichViewEdit<sup>(40)</sup> has new properties: MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup>. If these properties define a subrange of items, only this subrange (and only the pages containing it) is printed.

These properties allow implementing an incremental printing: when the next portion of the document is ready, it can be printed below the previously printed fragment. This feature can be used for printing accounting journals, logs, etc.

## New properties and methods

### New properties:

- TSRichViewEdit<sup>(40)</sup>.BackgroundProperty<sup>(48)</sup>.GlobalPageBackgroundColor<sup>(125)</sup> allows to define document background color without saving it in RVF files.
- TSRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup>.MouseWheelZoom<sup>(162)</sup> allows to turn off a mouse wheel zooming
- TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[][.BoxSchemes<sup>(232)</sup> defines schemes for applying skin elements to rectangles (such as items of a list box)
- TSRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup>: MainPen<sup>(161)</sup>, MainTitle, MainTitleColor<sup>(161)</sup>, MainTitleFont<sup>(161)</sup> allow drawing borders and label for the main document (like it is done for a header and a footer).
- TSRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup>: PageWidth<sup>(162)</sup>, PageHeight<sup>(162)</sup> return page size as it is shown on the screen.

### New methods of TSRichViewEdit<sup>(40)</sup>:

- GetPageNo<sup>(88)</sup> returns the page index for the specified position in the document.
- GetItemAt<sup>(84)</sup> returns the item at the specified coordinates.
- Clear<sup>(73)</sup>, Format<sup>(82)</sup> (not new but improved) clears and format the editor completely, including header and footer
- GetFirstVisiblePage<sup>(84)</sup>, GetLastVisiblePage<sup>(87)</sup> allow to get a range of visible pages (if a free page positioning mode is not used)

### New properties of TSRVScrollBar:

- UseXPThemes<sup>(248)</sup> allows/disallows using Windows themes (visual styles)
- Flat<sup>(245)</sup> makes the scrollbar look flat.

### Modified methods of TSRichViewEdit<sup>(40)</sup>:

GetItemPages<sup>(86)</sup>, GetItemBounds, GetItemBounds100<sup>(84)</sup>, GetItemCoords100<sup>(85)</sup> support item parts, so they can return more precise information.

## Advanced printing in TSRVPrint<sup>(204)</sup>

New PrintMode<sup>(214)</sup>: *srvpGrid*. It allows printing several pages on one paper sheet.

## Footnotes and endnotes

TSRichViewEdit can display and edit footnotes and endnotes.

New properties related to footnotes and endnotes:

- RVNote<sup>(64)</sup>

- CurrentNote<sup>(50)</sup>

New methods:

- StartEditNote<sup>(99)</sup>,
- ReturnToNote<sup>(95)</sup>,
- UpdateNote (is obsolete in version 6.0).

Additional information can be found in overview<sup>(38)</sup>.



















## Other features

Support of "keep with next" and "keep lines together" properties of paragraph style is restored (it was disabled in v3.0)

New TsrRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup>.AutoVScrollBarPosition<sup>(156)</sup> property controls the position of vertical scrollbar for right-to-left documents.

## Actions

### New actions<sup>(282)</sup>:

-  TsrvActionPreview<sup>(293)</sup> switches editing and preview modes.
-  TsrvActionThumbnails<sup>(311)</sup> shows/hides page thumbnails.
-  TsrvActionLayoutDraft<sup>(299)</sup> switches the editor to a draft layout.
-  TsrvActionLayoutWeb<sup>(310)</sup> switches the editor to a web layout.
-  TsrvActionLayoutPrint<sup>(301)</sup> switches the editor to a print layout.
-  TsrvActionZoom<sup>(313)</sup> applies the specified zooming percent to the editor.
-  TsrvActionZoomPageWidth<sup>(314)</sup> zooms the editor to fit the page width in the editor width.
-  TsrvActionZoomFullPage<sup>(314)</sup> zooms the editor to fit the full page in the editor.
-  TsrvActionOrientationPortrait<sup>(288)</sup> changes page orientation to "portrait".
-  TsrvActionOrientationLandscape<sup>(287)</sup> changes page orientation to "landscape".
-  TsrvActionPageFormat<sup>(289)</sup> changes page size.
-  TsrvActionEditHeader<sup>(315)</sup> starts editing a page header.
-  TsrvActionEditFooter<sup>(315)</sup> starts editing a page footer.
-  TsrvActionEditMain<sup>(316)</sup> starts editing the main document.
-  TsrvActionInsertFootnote<sup>(320)</sup> inserts a footnote.
-  TsrvActionInsertEndnote<sup>(319)</sup> inserts an endnote.
-  TsrvActionEditNote<sup>(319)</sup> starts editing a footnote/endnote.
-  TsrvActionReturnToNote<sup>(325)</sup> moves the caret to the parent footnote/endnote.

### Changes in actions:

-  TsrvActionPageSetup<sup>(291)</sup> can change HeaderY<sup>(137)</sup> and FooterY<sup>(137)</sup> properties.

Localization procedures<sup>(283)</sup> for the ScaleRichView actions.

## SRVControls

New controls: TsrVListBox<sup>(359)</sup>, TsrVComboBox<sup>(346)</sup>, TsrVImagesScroll<sup>(354)</sup>.

TsrVEdit<sup>(349)</sup> can show suggestions on editing.

New help topics about SRVControls<sup>(325)</sup>.

### 1.1.11 New in v3.0

#### Compatibility issues

##### Related to header and footer

Previously, TSRichViewEdit control contained only one RichViewEdit control (RichViewEdit<sup>(60)</sup>). Now, there are three of them: RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup> and RVFooter<sup>(62)</sup>. All of them can be edited. All editing operations must be applied to ActiveEditor<sup>(47)</sup>, not RichViewEdit<sup>(60)</sup>. However, file saving and loading operations still must be called for RichViewEdit<sup>(60)</sup>.

Before loading a file, you need to clear not only RichViewEdit, but also header and footer (loading RVF or RTF does it automatically, but loading text does not).

After loading a file, you need to format all editors:

```
SRichViewEdit1.RVHeader(64).Clear;
SRichViewEdit1.RVFooter(62).Clear;
SRichViewEdit1.RichViewEdit(60).Clear;
SRichViewEdit1.RichViewEdit.LoadRTF(FileName);
SRichViewEdit1.RVHeader.Format;
SRichViewEdit1.RVFooter.Format;
SRichViewEdit1.SetRVMargins(98);
```

(update: in newer version, you can use a simpler code:

```
SRichViewEdit1.Clear(73);
SRichViewEdit1.RichViewEdit.LoadRTF(FileName);
SRichViewEdit1.Format(82);
```

update 2: since version 6, you must use TSRichViewEdit.Clear and TSRichViewEdit.Format)

)

Alternatively, you can hide header and footer using TSRVPageProperty<sup>(133)</sup>'s HeaderVisible<sup>(137)</sup> and FooterVisible<sup>(136)</sup> properties.

The following events have TRichViewEdit control in Sender parameter: OnCurTextStyleChanged<sup>(107)</sup>, OnCurParaStyleChanged<sup>(107)</sup>, OnCaretMove<sup>(104)</sup> (previously, it was TSRichViewEdit). Now all events of TNotifyEvent type have RichViewEdit in Sender parameter, so you can distinguish if it was called from RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup> or RVFooter<sup>(62)</sup>.

In the new version, ConvertRVToSRV<sup>(73)</sup>, ConvertSRVToRV<sup>(74)</sup>, GetCurrentLineCol<sup>(83)</sup> methods work for ActiveEditor<sup>(47)</sup> instead of RichViewEdit<sup>(60)</sup>.

##### Formatting

A new text formatting procedure is used, so existing documents may look not exactly like before.

**The following properties of TSRVViewProperty<sup>(154)</sup> are renamed:**

- BoundTextPen → MarginsPen<sup>(162)</sup>
- BoundTextVisible → MarginsRectVisible<sup>(162)</sup>

##### New components:



TSRVSkinManager<sup>(228)</sup>

 **TSRVTabSet** <sup>(249)</sup>

 **TSRVScrollBar** <sup>(243)</sup>

## Header and footer

---

Two new properties of **TSRichViewEdit** control return **TRichViewEdit** controls used for editing/displaying header and footer: **RVHeader** <sup>(64)</sup>, **RVFooter** <sup>(62)</sup>.

**ActiveEditor** <sup>(47)</sup> returns the currently edited control (see the compatibility issues above). **OnChangeActiveEditor** <sup>(104)</sup> event occurs when it changed.

### New properties of **TSRVPageProperty** <sup>(133)</sup>:


- **HeaderY** <sup>(137)</sup>, **FooterY** <sup>(137)</sup>
- **HeaderVisible** <sup>(137)</sup>, **FooterVisible** <sup>(136)</sup>


### New properties of **TSRVViewProperty** <sup>(154)</sup>:

- **HeaderTitle**, **FooterTitle**
- **HeaderPen** <sup>(159)</sup>, **FooterPen** <sup>(157)</sup>
- **HeaderTitleFont** <sup>(159)</sup>, **FooterTitleFont** <sup>(158)</sup>
- **HeaderTitleColor** <sup>(159)</sup>, **FooterTitleColor** <sup>(157)</sup>

## New actions

---

 **TsrvActionQuickPrint** <sup>(294)</sup>

 **TsrvActionPrint** <sup>(293)</sup>

 **TsrvActionPageSetup** <sup>(291)</sup>

## Other changes

---

### New properties and methods of **TSRichViewEdit** <sup>(40)</sup>:

- **GetPageStartItemNo** <sup>(88)</sup>
- **SkinManager** <sup>(66)</sup>, **HScrollBarSchemeIndex** <sup>(53)</sup>, **VScrollBarSchemeIndex** <sup>(70)</sup>
- new optional parameter in **SetRVMargins** <sup>(98)</sup>

### New properties of **TSRVBackgroundProperty** <sup>(123)</sup>:

- **TransparentColor** <sup>(126)</sup>

### Headings

- **FindNextHeading** <sup>(78)</sup>/**FindPriorHeading** <sup>(80)</sup> search for headings.
- **NextCurHeading** <sup>(91)</sup>/**PriorCurHeading** <sup>(94)</sup> move the caret to the next/previous heading.

## 1.1.12 New in v2.0

### Compatibility issues

---

In several methods, the first letter is capitalized.

**The properties related to margins are renamed.** The pattern is "PageLeft → LeftMargin". This renaming includes both properties of `TSRichViewEdit`<sup>(40)</sup> (rarely used outside the component code) and properties of `TSRichViewEdit.PageProperty`<sup>(58)</sup>:`TSRVPageProperty`<sup>(133)</sup>:

- PageLeft → LeftMargin<sup>(138)</sup>
- PageTop → TopMargin<sup>(142)</sup>
- PageRight → RightMargin<sup>(142)</sup>
- PageBottom → BottomMargin<sup>(136)</sup>

**The properties related to page size are renamed.** The pattern is "FormatWidth → PageWidth". This renaming includes both properties of `TSRichViewEdit` (rarely used outside the component code) and properties of `TSRichViewEdit.PageProperty`:

- FormatWidth → PageWidth<sup>(141)</sup>
- FormatHeight → PageHeight<sup>(139)</sup>

**Other `TSRichViewEdit.PageProperty` properties and methods are renamed:**

- BreakPage → PageViewMode<sup>(141)</sup>
- FormatMode → PageFormat<sup>(139)</sup>
- ConvertPageSizeToFormatMode → ConvertPageSizeToPageFormat<sup>(143)</sup>
- getPaperSize → ConvertPageFormatToPageSize<sup>(143)</sup>

**All properties, methods and events related to table icons (rectangles displayed at the top left corners of tables) are renamed.** The pattern is "HintTable → TableIcon":

properties of `TSRichViewEdit`:

- getHintTableItem → GetTableIconItem<sup>(89)</sup>
- getHintTableRVData → GetTableIconRVData<sup>(89)</sup>
- OnHintTableClick → OnTableIconClick<sup>(121)</sup>

properties of `TSRVViewProperty`<sup>(154)</sup>:

- HintTableTimeDelay → TableIconDelay<sup>(163)</sup>
- HintTablePopupMenu → TableIconPopupMenu<sup>(163)</sup>
- HintTableVisible → UseTableIcons<sup>(164)</sup>

**In addition, the following properties of `TSRVViewProperty` are renamed:**

- HintPage → ShowScrollHint<sup>(163)</sup>

**The following methods and events of `TSRichViewEdit` are renamed:**

- calculatePosPages → CalculateAllPagePositions<sup>(72)</sup>
- calculatePosPagesFor → CalculatePagePosition<sup>(73)</sup>
- RectPage → GetPageClientRect<sup>(87)</sup>
- getPageMousePos → GetPageAt<sup>(87)</sup>
- OnCurParaStyleChange → OnCurParaStyleChanged<sup>(107)</sup>

**The following properties of `TSRVBackgroundProperty`<sup>(123)</sup> are renamed:**

- PictureType → PicturePosition<sup>(126)</sup>

**The following changes were made in `TSRVToolButton`<sup>(272)</sup>:**

- AlwaysUp property is removed
- AllowAllUp<sup>(273)</sup> property is added (now the behavior of these buttons is identical to `TSpeedButtons`)

**The following properties and methods of `TSRVPrint`<sup>(204)</sup> are renamed:**

- FormatMode → PageFormat<sup>(213)</sup>
- PaperCodeToFormatPaper → PaperCodeToPageFormat<sup>(219)</sup>
- UpdatePage → Update<sup>(222)</sup>

**The following methods of TSclRVRule<sup>(187)</sup> are renamed:**

getPageProperty → GetPageProperties<sup>(188)</sup>  
 setPageProperty → SetPageProperties<sup>(189)</sup>

**The following properties of TSRVPreview<sup>(223)</sup> are changed:**

- PageBorderColor → FrameBorderColor<sup>(225)</sup>
- PageBorderWidth → FrameBorderWidth<sup>(225)</sup>
- PageShadowColor, PageShadowWidth are removed (as well as "shadows" between frames)
- PageBoundsColor → PageBorderColor<sup>(226)</sup>
- PageBoundsWidth → PageBorderWidth<sup>(226)</sup>
- PageBoundsStyle → PageBorderStyle<sup>(226)</sup>
- BorderColor<sup>(225)</sup>, BorderWidth<sup>(225)</sup>, ShadowColor, ShadowWidth are added.

**The following properties, methods and events of TSRVPageScroll<sup>(174)</sup> are changed:**

- WidthBreakPage → PageBreakWidth<sup>(180)</sup>
- HeightBreakPage → PageBreakHeight<sup>(179)</sup>
- ShadowOff → ShadowWidth<sup>(182)</sup>
- CountPagesCache → CachedPagesCount<sup>(178)</sup>
- LimitCacheUpdate → CacheScrollLimit<sup>(178)</sup>
- GetPageNoToXY → GetPageAt<sup>(184)</sup>
- OnDrawSelect → OnDrawSelectedPage<sup>(187)</sup>
- OnDrawPageBorder<sup>(186)</sup>'s parameters are changed
- UpdateCache<sup>(185)</sup> is made public
- the component highlights the page containing the caret, instead of the first page visible in TSRichViewEdit.

## Other changes

Unnecessary and non working methods and events were removed (or made private).

TSRVFormatMode type is renamed to TSRVPageFormat<sup>(278)</sup>. TSRVPicture type is renamed to TSRVPicturePosition<sup>(126)</sup>.

Parameters of TSRichViewEdit.OnGetPagePos<sup>(108)</sup> event are changed.

The way how TSRVPrint.UsePhysicalOffsets<sup>(217)</sup> is processed is changed.

## New

New printing modes in TSRVPrint, see PrintMode<sup>(214)</sup> property.

New method TSRichViewEdit.GetItemPages<sup>(86)</sup> returns pages containing the specified item.

New method TSRichViewEdit.DeletePage<sup>(76)</sup> deletes a page as an editing operation.

New version is TSRVToolWindow.Execute<sup>(203)</sup> simplifies calling window which is invoked with a button.

Many small tweaks and fixes.



## 1.2 Overview

- Document parts in TSRichViewEdit<sup>(37)</sup>
- Footnotes and endnotes<sup>(38)</sup>

### 1.2.1 Document parts in TSRichViewEdit

#### Editors

TSRichViewEdit<sup>(40)</sup> control contains several TRichViewEdit controls inside. These controls are invisible, they are used to render and edit different document parts. These controls are available as properties of TSRichViewEdit:

- RichViewEdit<sup>(60)</sup> for the main document;
- RVHeader<sup>(64)</sup> for page headers;
- RVFooter<sup>(62)</sup> for page footers;
- RVNote<sup>(64)</sup> for notes and text box items.

RichViewEdit<sup>(60)</sup> stores the main document. Other editors are used to edit documents stored in different places:

- RVHeader<sup>(64)</sup> and RVFooter<sup>(62)</sup> edit Subdocuments<sup>(66)</sup>
- RVNote<sup>(64)</sup> edits Document of footnotes, endnotes, sidenotes, text boxes.

RichViewEdit<sup>(60)</sup> always contain the main document. Other editors contain valid documents only when they are active (equal to ActiveEditor<sup>(47)</sup>).

#### Loading

When you load a new document, call TSRichViewEdit.Format<sup>(82)</sup>.

This method copies page properties from TSRichViewEdit.RichViewEdit<sup>(60)</sup>.DocParameters to TSRichViewEdit.PageProperty<sup>(58)</sup> (when loading files, page properties are loaded in RichViewEdit.DocParameters), formats RVHeader<sup>(64)</sup>, RVFooter<sup>(62)</sup>, RichViewEdit<sup>(60)</sup>, notes and text boxes, paginates the document.

#### Example of loading RTF file:

```
SRichViewEdit1.Clear(73);
SRichViewEdit1.RichViewEdit(60).LoadRTF(FileName);
SRichViewEdit1.Format(82);
```

The same sequence must be called for other loading methods of TSRichViewEdit.RichViewEdit<sup>(60)</sup>: LoadRTFFromStream, LoadRVF, LoadRVFFromStream, LoadText, etc.

For RVF, you can use the methods of TSRichViewEdit: LoadRVF<sup>(90)</sup> and LoadRVFFromStream<sup>(90)</sup>. These methods load, format and assign all necessary data themselves.

If you do not want to display a header and a footer, assign:

```
SRichViewEdit1.PageProperty(58).HeaderVisible(137) := False;
SRichViewEdit1.PageProperty(58).FooterVisible(136) := False;
```

#### Switching between document parts


The active TRichViewEdit control (the control corresponding to the document part containing the caret) is returned in the TSRichViewEdit.ActiveEditor<sup>(47)</sup> property.


To activate editing of the main document, a header, or a footer, call `StartEditing`<sup>(99)</sup>.


To activate editing of a footnote or an endnote, call `StartEditNote`<sup>(99)</sup>.

The user can activate editing of the page header/footer by double clicking.

If you use `RichViewActions`, you can use the following actions to switch between document parts:

 `TsrvActionEditHeader`<sup>(315)</sup> starts editing a page header;

 `TsrvActionEditFooter`<sup>(315)</sup> starts editing a page footer;

 `TsrvActionEditMain`<sup>(316)</sup> to return to the main document.

Then `ActiveEditor`<sup>(47)</sup> is changed, `OnChangeActiveEditor`<sup>(104)</sup> event occurs.

## Editing

All editing operations must be applied to `ActiveEditor`<sup>(47)</sup>, for example:

```
SRichViewEdit1.ActiveEditor.Undo;
```

`ChangeStyleTemplates` must not be called for `ActiveEditor`. It must not be called if `ActiveEditor=RVNote`<sup>(64)</sup>.

Operations applied to the whole document (such as saving or loading) must be called for the main editor:

```
SRichViewEdit1.RichViewEdit(60).SaveRTF(...);
```

In the events of `TSRichViewEdit` (such as `OnStyleConversion`<sup>(121)</sup>, `OnParaStyleConversion`<sup>(116)</sup>, `OnCurTextStyleChanged`<sup>(107)</sup>, `OnCurParaStyleChanged`<sup>(107)</sup>) you can distinguish the `TRichViewEdit` control using the `Sender` parameter.

### 1.2.2 Notes and text boxes

`TSRichViewEdit` control can edit documents with footnotes, endnotes, sidenotes (notes in a floating box) and floating boxes.

When a note or a text box is being edited, its `Document` is loaded in `TSRichViewEdit.RVNote`<sup>(64)</sup>, `ActiveEditor`<sup>(47)</sup> becomes equal to `RVNote`<sup>(64)</sup>, `CurrentNote`<sup>(50)</sup> returns this note/text box item, `CurrentNoteParentEditor`<sup>(50)</sup> returns the editor containing this note.


Footnotes and endnotes can be inserted only in the main document. Sidenotes and text box items can be inserted in the main document, headers and footers.


#### Limitations:


- document of footnote/endnote must not be higher than one page;
- `ChangeStyleTemplates` must not be called when a note editor is active.


### Notes and text boxes in RichViewActions


 `TsrvActionInsertFootnote`<sup>(320)</sup> inserts a footnote.

 `TsrvActionInsertEndnote`<sup>(319)</sup> inserts an endnote.

 `TsrvActionInsertSidenote`<sup>(322)</sup> inserts a sidenote.

 `TsrvActionInsertTextBox`<sup>(324)</sup> inserts a text box item.

 `TsrvActionEditNote`<sup>(319)</sup> starts editing a note or a text box.

 `TsrvActionReturnToNote`<sup>(325)</sup> moves the caret to the parent footnote/endnote.

Use the actions above instead of their analogs included in RichViewActions for TRichViewEdit controls.

## Editing notes and text boxes without RichViewActions

Inserting a new footnote in SRichViewEdit1 (this example does not use StyleTemplates):

```
var Note: TRVFootnoteItemInfo;
    NoteRef: TRVNoteReferenceItemInfo;
    rve: TCustomRichViewEdit;

// we can insert a footnote only in the main document
rve := SRichViewEdit1.ActiveEditor47;
if rve<>SRichViewEdit1.RichViewEdit60 then
    exit;
// we will use item indices, so we require the top level editor
rve := rve.TopLevelEditor;
// creating a footnote
Note := TRVFootnoteItemInfo.CreateEx(rve.RVData,
    RVGetNoteTextStyleNo(rve.Style, rve.CurTextStyleNo), 1, False);
// filling Note.Document: adding a footnote number and a space character
NoteRef := TRVNoteReferenceItemInfo.CreateEx(Note.Document,
    RVGetNoteTextStyleNo(rve.Style, 0));
Note.Document.AddItem(' ', NoteRef);
Note.Document.AddNL(' ', 0, -1);
// inserting
if rve.InsertItem(' ', Note) then begin
    // starting editing the inserted footnote
    SRichViewEdit1.StartEditNote99(rve.RVData, rve.CurItemNo);
    // moving the caret to the end of the Note.Document
    rve := SRichViewEdit1.ActiveEditor47;
    // now rv = SRichViewEdit.RVNote64
    rve.SetSelectionBounds(
        rve.ItemCount-1, rve.GetOffsAfterItem(rve.ItemCount-1),
        rve.ItemCount-1, rve.GetOffsAfterItem(rve.ItemCount-1));
end;
```

Returning from RVNote<sup>64</sup> to the parent footnote/endnote:

```
SRichViewEdit1.ReturnToNote95;
```

## 1.3 Components

### Components




TSRichViewEdit<sup>40</sup> allows editing rich text documents in WYSIWYG mode. It supports scaling and different view modes, including a page-view mode.




TDBSRichViewEdit<sup>169</sup> is a data-aware version of WYSIWYG editing component.






TScIRVRuler<sup>187</sup> is a horizontal or vertical ruler designed to work with TSRichViewEdit. It is inherited from TRVRuler component included in RichViewActions.


 **TSRVToolBar**<sup>(189)</sup> is a simplified analog of TToolBar. Instances of TSRVToolBar are used in TSRichViewEdit: they allow adding buttons in scrollbar area.


 **TSRVToolWindow**<sup>(197)</sup> uses the TSRVToolBar and displays it in a popup window.

 **TSRVPageScroll**<sup>(174)</sup> displays thumbnails of pages for documents prepared in TSRichViewEdit component.

 **TSRVPrint**<sup>(204)</sup> and  **TSRVPreview**<sup>(223)</sup> are components for advanced printing of documents prepared in TSRichViewEdit. For example, they allow printing posters (e.g. A3 page on A4 printer).

 **TSRVSkinManager**<sup>(228)</sup> allows applying skins to scrollbars of TSRichViewEdit<sup>(40)</sup> / TDBSRichViewEdit<sup>(169)</sup>, to TSRVScrollBar<sup>(243)</sup>, to TSRVTabSet<sup>(249)</sup>.

 **TSRVTabSet**<sup>(249)</sup> – advanced skinnable tabs, can be used to implement a tabbed multiple document interface.

 **TSRVScrollBar**<sup>(243)</sup> – skinnable scrollbar. This component is used for scrollbars in TSRichViewEdit, but may be useful as a separate component as well.

### 1.3.1 TSRichViewEdit

TSRichViewEdit is a word processing component.

**Unit** SclRView.

#### Syntax

```
TSRichViewEdit = class (TCustomControl);
```

#### Hierarchy

---

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomControl

#### Description

---

The main features of the component are:

- zooming;
- WYSIWYG mode;
- real-time repagination;
- displaying and printing in different formats (e.g. the document in A5 format can be printed in A3 format);
- free positioning of page on the screen (optional);
- variety of document view modes (web mode, draft mode, page view mode, etc.).
- hint on scrolling containing page number and a short text;
- vertical and horizontal lines of buttons in the scroll bar area.

## Properties

### Main properties

RichViewEdit<sup>(60)</sup> returns a TRichViewEdit control used to render a document. Use properties and methods of this control to modify document or get information about it. By default, RichViewEdit<sup>(60)</sup> uses auto-created TRVStyle object. If you want to use an external TRVStyle component, use ExternalRVStyle<sup>(51)</sup> property.

Other editors are used to render different parts of document:

- RVHeader<sup>(64)</sup> for page header;
- RVFooter<sup>(62)</sup> for page footer;
- RVNote<sup>(64)</sup> for notes and text boxes.

The active editor is returned in ActiveEditor<sup>(47)</sup> property. Additional information about different part of documents is available in overview<sup>(37)</sup>.

Properties related to page attributes are available in PageProperty<sup>(58)</sup>.

Properties related to viewing modes are available in ViewProperty<sup>(69)</sup>.

Properties related to background are available in BackgroundProperty<sup>(48)</sup>.

Properties related to page arrangement inside the component window are available in PagePosProperty<sup>(58)</sup>.

Properties related to line numbering are available in LineNumberProperty<sup>(54)</sup>.

Properties related to a read-mode are available in ReadModeProperty<sup>(59)</sup>.

### Quick access to properties of RichViewEdit<sup>(60)</sup>

The following properties provide access to properties of RichViewEdit<sup>(60)</sup> of the same name (some of these properties are applied to ActiveEditor<sup>(47)</sup> or to all editors, see more information in the topics about these properties):

- AnimationMode<sup>(48)</sup>
- BiDiMode<sup>(48)</sup>
- CPEventKind<sup>(50)</sup>
- ReadOnly<sup>(59)</sup>
- RTFOptions<sup>(60)</sup>
- RTFReadProperties<sup>(61)</sup>
- RVBackgroundPicture<sup>(61)</sup> (access to BackgroundPicture)
- RVBackgroundProperties<sup>(61)</sup> (access to BackgroundProperties)
- RVColor<sup>(62)</sup> (access to Color)
- RVEditorOptions<sup>(62)</sup> (access to EditorOptions)
- RVFOptions<sup>(63)</sup>
- RVFParaStylesReadMode<sup>(63)</sup>
- RVFTextStylesReadMode<sup>(63)</sup>
- RVOptions<sup>(65)</sup> (access to Options)
- SmartPopupProperties<sup>(66)</sup>, SmartPopupVisible<sup>(66)</sup>
- TabNavigation<sup>(68)</sup>
- UseStyleTemplates<sup>(69)</sup>, StyleTemplateInsertMode<sup>(66)</sup>.

These properties (except for a public property `SmartPopupVisible`<sup>(66)</sup>) can be used to assign values at design time.

### Scrolling

`HScrollPos`<sup>(53)</sup> (0..`HMaxScrollPos`<sup>(52)</sup>) and `VScrollPos`<sup>(70)</sup> (0..`VMaxScrollPos`<sup>(69)</sup>) allow getting/setting the scrolling positions.

`HScrollBar`<sup>(53)</sup> and `VScrollBar`<sup>(69)</sup> hide/show scrollbars.

`OnHScrolled`<sup>(109)</sup> and `OnVScrolled`<sup>(122)</sup> events occur on scrolling.

`ScrolledPage`<sup>(65)</sup> returns the index of the first visible page.

### Menus (toolbars) in the scrollbar area

`MenuHButtons`<sup>(55)</sup> and `MenuHorizontal`<sup>(55)</sup> define properties of horizontal menu.

`MenuVButtons`<sup>(56)</sup> and `MenuVertical`<sup>(56)</sup> define properties of vertical menu.

### Skins

Skins can be applied to scrollbars. Skins are linked using `SkinManager`<sup>(66)</sup> property.

`HScrollBarSkinSchemeIndex`<sup>(53)</sup> and `HScrollBarSkinSchemeIndex`<sup>(70)</sup> are used to specify skin schemes for scrollbars.

### Converted properties of PageProperty<sup>(58)</sup>

Some properties return layout-related properties of `PageProperty`<sup>(58)</sup> converted to screen pixels and scaled according to `ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup>:

- `LeftMarginPix`<sup>(54)</sup>, `TopMarginPix`<sup>(68)</sup>, `RightMarginPix`<sup>(60)</sup>, `BottomMarginPix`<sup>(48)</sup>
- `PageHeightPix`<sup>(57)</sup>, `PageWidthPix`<sup>(58)</sup>

### Menus (toolbars) in scrollbars areas

Buttons are defined in `MenuHButtons`<sup>(55)</sup> and `MenuVButtons`<sup>(55)</sup>. Properties of menus are defined in `MenuHorizontal`<sup>(55)</sup> and `MenuVertical`<sup>(56)</sup>.

See also events below.

## Methods

### Coordinates

`ConvertRVToSRV`<sup>(73)</sup> and `ConvertSRVToRV`<sup>(74)</sup> allow translating coordinates between this `TSRichViewEdit` control and `RichViewEdit`<sup>(60)</sup>.

`GetPageClientRect`<sup>(87)</sup> returns the position of page, `GetItemBounds`<sup>(84)</sup> returns the position of item in the page. The page is returned by `GetItemPages`<sup>(86)</sup>.

`GetCaretPosInUnits`<sup>(83)</sup> returns the caret position on the page.

`GetPageAt`<sup>(87)</sup> returns the page at the specified coordinates, `GetItemAt`<sup>(84)</sup> returns the item at the specified coordinates.

`UnitsPerInchH`<sup>(100)</sup> and `UnitsPerInchV`<sup>(100)</sup> allow converting coordinates from units to units.

### Finding items

FindNextHyperlink<sup>(79)</sup>/FindPriorHyperlink<sup>(81)</sup>, FindNextCheckpoint<sup>(78)</sup>/FindPriorCheckpoint<sup>(80)</sup>, FindNextItem<sup>(79)</sup>/FindPriorItem<sup>(81)</sup>, FindNextHeading<sup>(78)</sup>/FindPriorHeading<sup>(80)</sup> search for hyperlinks, *checkpoints*, items, headings.

NextCurHyperlink<sup>(91)</sup>/PriorCurHyperlink<sup>(94)</sup>, NextCurCheckpoint<sup>(90)</sup>/PriorCurCheckpoint<sup>(94)</sup>, NextCurItem<sup>(91)</sup>/PriorCurItem<sup>(95)</sup> select the next/previous hyperlink, item with *checkpoint*, item.

NextCurHeading<sup>(91)</sup>/PriorCurHeading<sup>(94)</sup> moves the caret to the next/previous heading.

By default, the search is performed in the whole document. But you can search in the specified range of vertical coordinates, see RangeSearch<sup>(58)</sup> property.

FirstCurPage<sup>(82)</sup>, PriorCurPage<sup>(95)</sup>, NextCurPage<sup>(91)</sup>, LastCurPage<sup>(90)</sup> moves the caret to the beginning of the first/previous/next/last page.

## Printing

You can print the whole document (PrintAll<sup>(92)</sup>), the current page (PrintCurrent<sup>(93)</sup>) or a range of pages (PrintRange<sup>(93)</sup>).

With MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup> you can define a subrange of items for printing. Items outside of this range are not printed (but places are still reserved for them). Using these properties, you can implement an incremental printing: when the next portion of the document is ready, you can print it below the previously printed fragment.

## Drawing on external canvas

DrawPage<sup>(77)</sup>/DrawMetafile<sup>(76)</sup> draws page onto the specified canvas/metafile.

## Scrolling

ScrollToCaret<sup>(96)</sup>/ScrollToItem<sup>(96)</sup> makes the caret/item visible.

GetFirstItemVisibleRV<sup>(83)</sup> returns the first visible item.

## Footnotes and endnotes

Call StartEditNote<sup>(99)</sup> to edit a footnote/endnote, call ReturnToNote<sup>(95)</sup> to finish editing.

Additional information can be found in overview<sup>(38)</sup>.

## Events

The most of TScaleRichViewEdit events provides access to RichViewEdit<sup>(60)</sup>'s events of the same names.

### Events on changing properties

OnPageFormatChanged<sup>(114)</sup> occurs when PageProperty<sup>(58)</sup>.PageFormat<sup>(139)</sup> is changed.

OnChangeViewModeAfter<sup>(105)</sup> and OnChangeViewModeBefore<sup>(105)</sup> occur when ViewProperty<sup>(69)</sup>.ViewMode<sup>(165)</sup> is changed.

OnZoomChanged<sup>(122)</sup> occurs when ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> or .ZoomMode<sup>(166)</sup> are changed. OnZoomViewChanged<sup>(123)</sup> occurs when other properties of ViewProperty<sup>(69)</sup> related to zooming are changed.

### Events on scrolling, clicking and caret moving

OnHScrolled<sup>(109)</sup>, OnVScrolled<sup>(122)</sup>, OnPageScrolled<sup>(114)</sup> occur on scrolling.

OnCurrentPageChange<sup>(107)</sup> occurs when the caret is moved to another page.

OnPageCountChanged<sup>(114)</sup> occurs when the count of pages is changed.

OnClickPage<sup>(106)</sup> occurs when the user clicked on the page.

### Database

The following items occurs if Document<sup>(51)</sup> property is linked to a database field using LiveBindings, or in inherited TDBSRichViewEdit<sup>(169)</sup> component.

OnNewDocument<sup>(113)</sup> occurs when creating or before loading

OnLoadDocument<sup>(111)</sup> occurs after loading

OnSaveCustomFormat<sup>(119)</sup>, OnLoadCustomFormat<sup>(111)</sup> allow storing data in DB in custom format.

### Custom drawing

In addition to custom drawing events of TRVStyle, you can use OnPaintPage<sup>(115)</sup> for drawing on page, OnPaintComponent<sup>(114)</sup> for drawing inserted controls, and OnPaint<sup>(114)</sup> for drawing something else.

### Inserted controls

OnPaintComponent<sup>(114)</sup> allows drawing inserted controls (useful if the default procedure cannot draw them properly).

OnCheckControl<sup>(105)</sup> allows enumerating all inserted controls.

OnMessageControl<sup>(112)</sup> allows responding on messages from inserted controls.

### Free page position and ordering

OnGetPagePos<sup>(108)</sup> allows defining position, zooming and Z-order of pages

### Menus (toolbars) in scrollbars areas

OnHMenuClickButton<sup>(109)</sup> and OnVMenuClickButton<sup>(121)</sup> occur when the user clicked a button.

OnHMenuEnterButton<sup>(109)</sup> and OnVMenuEnterButton<sup>(122)</sup> occur when the mouse pointer is above a button.

### Table "icons"

When the user moves the mouse pointer above some table, a special icon is displayed at the top left corner of this table (if ViewProperty<sup>(69)</sup>.UseTableIcons<sup>(164)</sup>=True).

This icon is displayed for some time (specified in ViewProperty<sup>(69)</sup>.TableIconDelay<sup>(163)</sup>) after the mouse pointer leaves this table.

When the user clicks on this icon, OnTableIconClick<sup>(121)</sup> occurs.

## 1.3.1.1 Properties

### In TSRichViewEdit

- AcceptDragDropFormats<sup>(47)</sup>
- AcceptPasteFormats<sup>(47)</sup>
- ▶ ActiveEditor<sup>(47)</sup>
- AnimationMode<sup>(48)</sup>
- BackgroundProperty<sup>(48)</sup>



- BiDiMode <sup>48</sup>
- ▶ BottomMarginPix <sup>48</sup>
- CanScroll <sup>49</sup>
- CanUpdate <sup>49</sup>
- CanUpdatePosition <sup>49</sup>
- CanUpdateMargin <sup>49</sup>
- CaretBlinkTime <sup>49</sup>
- ▶ CaretPos <sup>49</sup>
- CaretVisible <sup>50</sup>
- CPEventKind <sup>50</sup>
- ▶ CurControl <sup>50</sup>
- ▶ CurrentNote <sup>50</sup>
- ▶ CurrentNoteParentEditor <sup>50</sup>
- CurrentPage <sup>51</sup>
- ExternalRVStyle <sup>51</sup>
- Document <sup>51</sup>
- ExternalRVStyleFooter <sup>51</sup>
- ExternalRVStyleHeader <sup>51</sup>
- ▶ FirstPageNo <sup>52</sup>
- FixMarginsMode <sup>52</sup>
- ▶ HMaxScrollPos <sup>52</sup>
- HScrollBar <sup>53</sup>
- HScrollBarSchemeIndex <sup>53</sup>
- HScrollPos <sup>53</sup>
- HTMLReadProperties <sup>53</sup>
- HTMLSaveProperties <sup>53</sup>
- ▶ LastPageNo <sup>54</sup>
- ▶ LeftMarginPix <sup>54</sup>
- MarkdownProperties <sup>54</sup>
- MaxPrintedItemNo <sup>55</sup>
- MaxPrintedOffsInItem <sup>55</sup>
- MenuHButtons <sup>55</sup>
- MenuHorizontal <sup>55</sup>
- MenuVButtons <sup>56</sup>
- MenuVertical <sup>56</sup>
- MinPrintedItemNo <sup>56</sup>
- MinPrintedOffsInItem <sup>56</sup>
- ▶ OffsetX <sup>57</sup>
- ▶ OffsetY <sup>57</sup>
- ▶ PageCount <sup>57</sup>
- ▶ PageHeightPix, PageHeight100Pix <sup>57</sup>
- ▶ PageNoMouseDown <sup>58</sup>
- ▶ PageNoMouseMove <sup>58</sup>
- PageProperty <sup>58</sup>
- ▶ PageWidthPix, PageWidth100Pix <sup>58</sup>
- RangeSearch <sup>58</sup>
- RangeSearchFirstY <sup>59</sup>
- RangeSearchLastY <sup>59</sup>

- ReadModeProperty <sup>59</sup>
- ReadOnly <sup>59</sup>
  - RichViewEdit <sup>60</sup>
- ▶ RightMarginPix <sup>60</sup>
- RTFOptions <sup>60</sup>
- RTFReadProperties <sup>61</sup>
- RVBackgroundPicture <sup>61</sup>
- RVBackgroundProperties <sup>61</sup>
- RVColor <sup>62</sup>
- RVEditorOptions <sup>62</sup>
  - RVFooter <sup>62</sup>
- RVFOptions <sup>63</sup>
- RVFParaStylesReadMode <sup>63</sup>
- RVFTextStylesReadMode <sup>63</sup>
  - RVHeader <sup>64</sup>
  - RVNote <sup>64</sup>
- RVOptions <sup>65</sup>
- ▶ ScrolledPage <sup>65</sup>
- SkinManager <sup>66</sup>
- SmartPopupProperties <sup>66</sup>
  - SmartPopupVisible <sup>66</sup>
- StyleTemplateInsertMode <sup>66</sup>
- TabNavigation <sup>68</sup>
- TextEngine <sup>68</sup>
- ▶ TopMarginPix <sup>68</sup>
- UnitsProgram <sup>68</sup>
- UseDrawHyperlinksEvent <sup>69</sup>
- UseStyleTemplates <sup>69</sup>
- Version <sup>69</sup>
- ViewProperty <sup>69</sup>
- ▶ VMaxScrollPos <sup>69</sup>
- VScrollBar <sup>69</sup>
- VScrollBarSchemeIndex <sup>70</sup>
  - VScrollPos <sup>70</sup>

## Derived from TCustomControl

- Align
- Anchors
- Constraints
- Ctl3D
- DragMode
- Enabled
- Height
- HelpContext
- HelpKeyword (D6+)
- HelpType (D6+)
- Hint
- Left

- Name
- ParentCtl3D
- ParentShowHint
- PopupMenu
- ShowHint
- TabOrder
- TabStop
- Tag
- Top
- Touch (D2010+)
- Visible
- Width

#### 1.3.1.1.1 TSRichViewEdit.AcceptDragDropFormats, AcceptPasteFormats

AcceptDragDropFormats lists formats that can be accepted when dropping data in TSRichViewEdit.

AcceptPasteFormats lists formats that can be accepted when pasting in TSRichViewEdit.

**property** AcceptDragDropFormats: TRVDragDropFormats;

**property** AcceptPasteFormats: TRVDragDropFormats;

These properties provide access to RichViewEdit<sup>(60)</sup>.AcceptDragDropFormats and AcceptPasteFormats properties.

Default values

- AcceptDragDropFormats: [*rvddRVF*, *rvddRTF*, *rvddText*, *rvddUnicodeText*, *rvddBitmap*, *rvddMetafile*, *rvddFiles*]
- AcceptPasteFormats: [*rvddRVF*, *rvddRTF*, *rvddText*, *rvddUnicodeText*, *rvddBitmap*, *rvddMetafile*, *rvddFiles*, *rvddHTML*]

#### 1.3.1.1.2 TSRichViewEdit.ActiveEditor

Returns the active editor belonging to this TSRichViewEdit control.

**property** ActiveEditor: TRichViewEdit;

This property returns the editor containing the caret. All editing operations must be applied to this editor.

This property can return one of the following values:

- RichViewEdit<sup>(60)</sup>
- RVHeader<sup>(64)</sup>
- RVFooter<sup>(62)</sup>
- RVNote<sup>(64)</sup>

When the user starts editing another RichViewEdit, OnChangeActiveEditor<sup>(104)</sup> event occurs.

**See also methods:**

- StartEditing<sup>(99)</sup>
- StartEditNote<sup>(99)</sup>
- Document parts in TSRichViewEdit<sup>(37)</sup>

### 1.3.1.1.3 TSRichViewEdit.AnimationMode

Specifies when an animation of images starts.

**property** AnimationMode: TRVAnimationMode;

Animation is played in ActiveEditor<sup>(47)</sup>. Animation in other editors is disabled.

To enable gif animation, include in your project:

- RVGifAnimate.pas (animation using the Anders Melander's TGifImage, for Delphi 4-2006);
- RVGifAnimate2007.pas (animation using VCL's TGifImage, for Delphi 2007 or newer);
- RVJvGifAnimate.pas (animation using the JVCL's TJvGifImage).

**Default value:**

*rvaniOnFormat* (animation is started automatically)

### 1.3.1.1.4 TSRichViewEdit.BackgroundProperty

Contains background settings.

**property** BackgroundProperty: TSRVBackgroundProperty<sup>(123)</sup>;

### 1.3.1.1.5 TSRichViewEdit.BiDiMode

Specifies the default bidirectional mode for the control.

**property** BiDiMode: TRVBiDiMode;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

A value of this property changes layout of the whole editor. If it is equal to *rvbdRightToLeft*, the following changes are applied:

- the vertical scrollbar is placed at the left side (unless you assign *False* to ViewProperty<sup>(69)</sup>.AutoVScrollBarPosition)<sup>(156)</sup>
- if pages are arranged in columns, pages in each row are ordered from right to left
- PagePosProperty<sup>(58)</sup>.AlignPageH<sup>(144)</sup> = *srvaphLeft* aligns pages to the right; *srvaphRight* aligns pages to the left.

### 1.3.1.1.6 TSRichViewEdit.BottomMarginPix

Returns the bottom margin, in pixels.

**property** BottomMarginPix: Integer;

**property** BottomMargin100Pix: Integer;

The bottom margin is defined in PageProperty<sup>(58)</sup>.BottomMargin<sup>(136)</sup>. These properties return this margin converted to pixels.

**BottomMarginPix** is scaled according to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> (so this property returns the margin as it is displayed on the screen).

**BottomMargin100Pix** is not scaled, it is the margin in 96 DPI and 100% zooming:

**BottomMarginPix** := **BottomMargin100Pix** \* (ZoomPercent / 100) \* (<screen DPI> / 96).

**See also:**

- LeftMarginPix <sup>(54)</sup>
- TopMarginPix <sup>(68)</sup>
- RightMarginPix <sup>(60)</sup>

#### 1.3.1.1.7 TSRichViewEdit.CanUpdate, CanScroll, CanUpdatePosition

Enables/disables repainting.

```
property CanUpdate: Boolean;  
property CanScroll: Boolean;  
property CanUpdatePosition: Boolean;
```

If CanUpdate=*False*, the component does not update its image on the screen and does not move the caret.

If CanScroll=*False*, the component does not scroll itself.

If CanUpdatePosition=*False*, the component does not scroll to make the caret visible.

**See also:**

- CalculatePageCount <sup>(72)</sup>

#### 1.3.1.1.8 TSRichViewEdit.CanUpdateMargin

Enables/disables margins changing.

```
property CanUpdateMargin: Boolean;
```

If this property is *False*:

- the component does not allow changing FormatWidth <sup>(141)</sup>, FormatHeight <sup>(139)</sup>, PageLeft <sup>(138)</sup>, PageTop <sup>(142)</sup>, PageRight <sup>(142)</sup>, PageBottom <sup>(136)</sup> properties;
- SetMargin <sup>(97)</sup> and SetMarginMM <sup>(97)</sup> functions do not set the margin values.

#### 1.3.1.1.9 TSRichViewEdit.CaretBlinkTime

Caret blinking interval.

```
property CaretBlinkTime: Cardinal;
```

This property defines the period when the caret remains in one of the states: displayed or not.

**Default value:**

a system default value returned by GetCaretBlinkTime function.

#### 1.3.1.1.10 TSRichViewEdit.CaretPos

Client coordinates of the caret.

```
property CaretPos: TPoint;
```

The caret coordinates relative to the top left corner of the component.

**See also:**

- GetCaretPosInUnit <sup>(83)</sup>

### 1.3.1.1.11 TSTRichViewEdit.CaretVisible

Enable/disable the caret display.

**property** `CaretVisible: Boolean;`

CursorVisible indicates whether to display caret or not, for example for implementing a read-only mode or a print preview.

By default the caret is visible.

### 1.3.1.1.12 TSTRichViewEdit.CPEventKind

Specifies the mode of generating OnCheckpointVisible<sup>(106)</sup> event.

**property** `CPEventKind: TCPEventKind;`

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

TSTRichViewEdit supports this property&event, despite of the fact that they are not supported by TRichViewEdit.

### 1.3.1.1.13 TSTRichViewEdit.CurControl

Returns the control (inserted in the editor) having the input focus.

**property** `CurControl: TControl;`

*Nil*, if no control has focus.

**See also:**

- OnChangeCurrentControl<sup>(105)</sup>

### 1.3.1.1.14 TSTRichViewEdit.CurrentNote, CurrentNoteParentEditor

The properties return the note (or a text box) item edited in RVNote<sup>(64)</sup>, and the editor containing this note.

**property** `CurrentNote: TCustomRVNoteItemInfo;`

**property** `CurrentNoteParentEditor: TCustomRichViewEdit;`

These properties return *nil* if a note is not being edited.

Use StartEditNote<sup>(99)</sup> to start editing a note, use ReturnToNote<sup>(95)</sup> to finish editing.

CurrentNote can be of one of the following types:

- footnotes (TRVFootnoteItemInfo)
- endnotes (TRVEndnoteItemInfo)
- sidenotes (TRVSidenoteItemInfo)
- text box items (TRVTextBoxItemInfo)

Footnotes and endnotes can be inserted only in RichViewEdit<sup>(60)</sup>. Sidenotes and text boxes can be inserted in RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup> and RVFooter<sup>(62)</sup>.

CurrentNoteParentEditor always returns a root editor (RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup> or RVFooter<sup>(62)</sup>), even if a note is inserted in a table cell).

**See also:**

- Footnotes and endnotes<sup>(38)</sup>

#### 1.3.1.1.15 TSRichViewEdit.CurrentPage

Index of the current page (from 1)

**property** `CurrentPage: Integer;`

This page contains the caret.

If you assign a value to this property, the caret is moved to the beginning of the specified page.

**See also:**

- `ScrolledPage` <sup>(65)</sup>
- `OnCurrentPageChange` <sup>(107)</sup>
- `FirstCurPage` <sup>(82)</sup>, `PriorCurPage` <sup>(95)</sup>, `NextCurPage` <sup>(91)</sup>, `LastCurPage` <sup>(90)</sup>

#### 1.3.1.1.16 TSRichViewEdit.Document

Allows linking this TSRichViewEdit component to a database field (Delphi XE2+)

**property** `Document: TRVDocumentProperty;`

You can use LiveBindings to link this property to a dataset field of TBlobField type.

This is an alternative way of using ScaleRichView components to view and edit database fields. Another way is using TDBSRichViewEdit <sup>(169)</sup> components.

This property contains sub-properties controlling this linking.

**Important note:** in order to link **Document** to TBlobField, DBRV unit must be included in the project (add it in "uses").

**Note:** while this property is available in TDBSRichViewEdit <sup>(169)</sup> as well, it's highly recommended to use LiveBindings only for TSRichViewEdit.

#### 1.3.1.1.17 TSRichViewEdit.ExternalRVStyle, -Header, -Footer

These properties allow using external TRVStyle components.

**property** `ExternalRVStyle: TRVStyle;`

**property** `ExternalRVStyleHeader: TRVStyle;`

**property** `ExternalRVStyleFooter: TRVStyle;`

Assigning these properties is optional. You can assign links to TRVStyle components to these properties.

If assigned (not *nil*):

- `ExternalRVStyle` is assigned to `RichViewEdit` <sup>(60)</sup>.`Style`
- `ExternalRVStyleHeader` is assigned to `RVHeader` <sup>(64)</sup>.`Style`
- `ExternalRVStyleFooter` is assigned to `RVFooter` <sup>(62)</sup>.`Style`

By default, these properties are not assigned (*nil*), and internal TRVStyle components are used.

All these components must be different, you cannot assign the same component to two or more of these properties.

These properties are useful if you want to define properties of TRVStyle at design time.

If `UseStyleTemplates` <sup>(69)</sup> = *True*, the component assigns:

- RichViewEdit<sup>(60)</sup>.Style.MainRVStyle = *nil*;
- RVHeader<sup>(64)</sup>.Style.MainRVStyle = RichViewEdit<sup>(60)</sup>.Style
- RVFooter<sup>(62)</sup>.Style.MainRVStyle = RichViewEdit<sup>(60)</sup>.Style

If some of these TRVStyle components are external, these assignments are made to their properties too.

#### 1.3.1.1.18 TSRichViewEdit.FirstPageNo

Returns the index of the first page that will be printed, from 1.

**property** FirstPageNo: Integer;

This property takes the properties MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup> (and the corresponding offsets in items) into account. By default, when these properties define the range including the whole document, FirstPageNo returns 1. If MinPrintedItemNo<sup>(56)</sup>>0, this property returns the index of the page containing the MinPrintedItemNo<sup>(56)</sup>-th item (see GetPageNo<sup>(88)</sup>). If MinPrintedItemNo<sup>(56)</sup>>=RichViewEdit<sup>(60)</sup>.ItemCount, FirstPageNo returns PageCount<sup>(57)</sup>+1 (and nothing will be printed).

#### 1.3.1.1.19 TSRichViewEdit.FixMarginsMode

Specifies how to work with too small margins (less than the printer supports).

**type**

TSRVFixMarginsMode = (srvfmmAutoCorrect, srvfmmIgnore);

**property** FixMarginsMode : TSRVFixMarginsMode;

This property is similar to TRVPrint's property of the same name.

| Mode                     | Meaning   |
|--------------------------|---|
| <i>srvfmmAutoCorrect</i> | Minimal supported margins are used (if the left and/or the top margin were too small, the output is moved to the right and/or down) |
| <i>srvfmmIgnore</i>      | The specified values of margins are used (document may be cropped)  |

**Default value:**

*srvfmmAutoCorrect*

**See also**

- PageProperty<sup>(58)</sup>.PrintableAreaPen<sup>(141)</sup>

#### 1.3.1.1.20 TSRichViewEdit.HMaxScrollPos

Defines the maximum position of horizontal scroll bar.

**property** HMaxScrollPos: Integer;

This value is used to calculate the maximum possible value for HScrollPos<sup>(53)</sup>.

This is a read-only property.

**See also:**



- VMaxScrollPos<sup>(69)</sup>

#### 1.3.1.1.21 TSRichViewEdit.HScrollBar

Shows/hides the horizontal scrollbar.

**property** HScroolBar: Boolean;

**Default value:**

True (visible)

#### 1.3.1.1.22 TSRichViewEdit.HScrollBarSchemeIndex

Determines which skin scheme is used for the horizontal scrollbar.

**property** HScrollBarSchemeIndex: Integer;

This is an index in SkinManager<sup>(66)</sup>.CurrentSkin<sup>(230)</sup>.HorizontalTabSetSchemes<sup>(232)</sup> collection.

If SkinManager<sup>(66)</sup> = *nil*, or SkinManager<sup>(66)</sup>.CurrentSkin<sup>(230)</sup> = *nil*, or this index is not valid for this collection (negative or too large), then this scrollbar has a default (no-skin) appearance.

**Default value**

0

**See also**

- VScrollBarSchemeIndex<sup>(70)</sup>
- ScrollBarControlStyle<sup>(65)</sup>

#### 1.3.1.1.23 TSRichViewEdit.HScrollPos

Defines the position of horizontal scroll bar, measured in screen pixels.

**property** HScrollPos: Integer;

You can use this property for scrolling to the specified position.

Value of this property is in the range from 0 to HMaxScrollPos<sup>(52)</sup> - ClientWidth + 1.

**See also:**

- VScrollPos<sup>(70)</sup>

#### 1.3.1.1.24 TSRichViewEdit.HTMLReadProperties

A set of properties controlling HTML import.

**property** HTMLReadProperties: TRVHTMLReaderProperties;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

#### 1.3.1.1.25 TSRichViewEdit.HTMLSaveProperties

A set of properties controlling HTML export.

**property** HTMLSaveProperties: TRVHTMLSaveProperties;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

**Note:** the default value for **HTMLSaveProperty**.ImageOptions is different for control types:

- for TSRichViewEdit<sup>(40)</sup>: []

- for TDBSRichViewEdit<sup>(169)</sup>: [rvhtmlsiolnlineImages]

#### 1.3.1.1.26 TSRichViewEdit.LastPageNo

Returns the index of the last page that will be printed, from 1.

**property** LastPageNo: Integer;

This property takes the properties MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup> into account. By default, when these properties define the range including the whole document, LastPageNo returns PageCount<sup>(57)</sup>. If MaxPrintedItemNo<sup>(55)</sup> is inside the items range (0..RichViewEdit<sup>(60)</sup>.ItemCount-1), this property returns the index of the page containing the MaxPrintedItemNo<sup>(55)</sup>-th item (see GetPageNo<sup>(88)</sup>).

#### 1.3.1.1.27 TSRichViewEdit.LeftMarginPix

Returns the left margin, in pixels.

**property** LeftMarginPix: Integer;

**property** LeftMargin100Pix: Integer;

**function** GetLeftMarginPix(PageNo : Integer) : Integer;

**function** GetLeftMargin100Pix(PageNo : Integer) : Integer;

The left margin is defined in PageProperty<sup>(58)</sup>.LeftMargin<sup>(138)</sup>.

##### Properties

These properties return this margin converted to pixels.

**LeftMarginPix** is scaled according to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> (so this property returns the margin as it is displayed on the screen).

**LeftMargin100Pix** is not scaled, it is the margin in 96 DPI and 100% zooming:

**LeftMarginPix** := **LeftMargin100Pix** \* (ZoomPercent / 100) \* (<screen DPI> / 96).

##### Methods

The methods return the left margin for the specified page, taking PageProperty<sup>(58)</sup>.MirrorMargins<sup>(138)</sup> into account.

##### See also:

- TopMarginPix<sup>(68)</sup>
- RightMarginPix<sup>(60)</sup>
- BottomMarginPix<sup>(48)</sup>

#### 1.3.1.1.28 TSRichViewEdit.LineNumberProperty

Contains settings for displaying line numbers

**property** LineNumberProperty: TSRVLineNumberProperty<sup>(131)</sup>;

#### 1.3.1.1.29 TSRichViewEdit.MarkdownProperties

A set of properties controlling Markdown export.

**property** MarkdownProperties: TRVMarkdownProperties;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

### 1.3.1.1.30 TSRichViewEdit.MaxPrintedItemNo, MaxPrintedOffsInItem

The properties define the ending position to print.

**property** MaxPrintedItemNo: Integer;

**property** MaxPrintedOffsInItem: Integer;

The component prints items from (MinPrintedItemNo, MinPrintedOffsInItem)<sup>(56)</sup> to (MaxPrintedItemNo, MaxPrintedOffsInItem). By default, this range includes all items.

Items outside of this range are not printed (but they still occupy their spaces). The pages that do not contain items in this range are skipped (the component prints pages from FirstPageNo<sup>(52)</sup> to LastPageNo<sup>(54)</sup>).

**MaxPrintedItemNo** is the index of item in RichViewEdit<sup>(60)</sup>, the last item to print. The value -1 is treated like RichViewEdit<sup>(60)</sup>.ItemCount-1, i.e. the document is printed to the end.

**MaxPrintedOffsInItem** is the offset of the last position in the **MaxPrintedItemNo**-th item to print. The value -1 is treated like RichViewEdit<sup>(60)</sup>.GetOffsAfterItem(**MaxPrintedItemNo**), i.e. the item is printed completely. If this is a text item, this property allows printing a part of it. The position is defined with up-to-line, not up-to-character precision (i.e., if the item is displayed as several lines, and the specified position is in the middle of a line, the part of item on this line is printed).

These properties allow implementing an incremental printing: when the next portion of the document is ready, it can be printed below the previously printed fragment. This feature can be used for printing accounting journals, logs, etc.

**Default value:**

-1

### 1.3.1.1.31 TSRichViewEdit.MenuHButtons

A collection of buttons for the horizontal menu.

**property** MenuHButtons: TSRVButtonCollection<sup>(271)</sup>;

A horizontal menu (horizontal toolbar) is placed in the horizontal scrollbar area.

Properties of horizontal menu are defined in MenuHorizontal<sup>(55)</sup> property.

**See also:**

- MenuVButtons<sup>(56)</sup>

### 1.3.1.1.32 TSRichViewEdit.MenuHorizontal

Properties of a horizontal menu (horizontal toolbar)

**property** MenuHorizontal: TSRVPropertyTBH<sup>(148)</sup>;

**See also:**

- MenuHButtons<sup>(55)</sup>
- OnHMenuClickButton<sup>(109)</sup>
- OnHMenuEnterButton<sup>(109)</sup>
- MenuVertical<sup>(56)</sup>

### 1.3.1.1.33 TSRichViewEdit.MenuVButtons

A collection of buttons for the vertical menu.

**property** MenuVButtons: TSRVButtonCollection<sup>(271)</sup>;

A vertical menu (vertical toolbar) is placed in the vertical scrollbar area.

Properties of vertical menu are defined in MenuVertical<sup>(56)</sup> property.

**See also:**

- MenuHButtons<sup>(55)</sup>

### 1.3.1.1.34 TSRichViewEdit.MenuVertical

Properties of a vertical menu (vertical toolbar)

**property** MenuVertical: TSRVPropertyTBV<sup>(149)</sup>;

**See also:**

- MenuVButtons<sup>(56)</sup>
- OnVMenuClickButton<sup>(121)</sup>
- OnVMenuEnterButton<sup>(122)</sup>
- MenuHorizontal<sup>(55)</sup>

### 1.3.1.1.35 TSRichViewEdit.MinPrintedItemNo, MinPrintedOffsInItem

The properties define the starting position to print.

**property** MinPrintedItemNo: Integer;

**property** MinPrintedOffsInItem: Integer;

The component prints items from (**MinPrintedItemNo**, **MinPrintedOffsInItem**) to (**MaxPrintedItemNo**, **MaxPrintedOffsInItem**<sup>(55)</sup>). By default, this range includes all items.

Items outside of this range are not printed (but they still occupy their spaces). The pages that do not contain items in this range are skipped (the component prints pages from FirstPageNo<sup>(52)</sup> to LastPageNo<sup>(54)</sup>).

**MinPrintedItemNo** is the index of item in RichViewEdit<sup>(60)</sup>, the first item to print.

**MinPrintedOffsInItem** is the offset of the first position in the **MinPrintedItemNo**-th item to print. If this is a text item, this property allows printing a part of it. The position is defined with up-to-line, not up-to-character precision (i.e., if the item is displayed as several lines, and the specified position is in the middle of a line, the part of item on this line is printed).

These properties allow implementing an incremental printing: when the next portion of the document is ready, it can be printed below the previously printed fragment. This feature can be used for printing accounting journals, logs, etc.

**Default value:**

0

### 1.3.1.1.36 TSRichViewEdit.OffsetX

A horizontal offset of document (page) image in the editor window.

**property** OffsetX: Single;

This property is a maximal value of:

- $(\text{ClientWidth} - \text{PageWidthPix}^{(58)}) / 2$
- $\text{PagePosProperty}^{(58)}.HPadding^{(146)}$  (taking  $\text{PagePosProperty}^{(58)}.HiddenDistances$  into account)

**See also:**

- OffsetY<sup>(57)</sup>

### 1.3.1.1.37 TSRichViewEdit.OffsetY

A vertical offset of document (page) image in the editor window.

**property** OffsetY: Single;

This property is a maximal value of:

- $(\text{ClientHeight} - \text{PageHeightPix}^{(57)}) \text{ div } 2$
- 0

**See also:**

- OffsetX<sup>(57)</sup>

### 1.3.1.1.38 TSRichViewEdit.PageCount

Returns the count of pages in the document.

**property** PageCount: Integer;

Accessing this property does not invoke a repagination.

Use  $\text{calculatePageCount}^{(72)}$  method for repagination.

### 1.3.1.1.39 TSRichViewEdit.PageHeightPix, PageHeight100Pix

The properties return a page height of document on the screen.

**property** PageHeightPix: Integer;

**property** PageHeight100Pix: Integer;

**PageHeightPix** returns the page height converted to screen pixels and scaled according to  $\text{ViewProperty}^{(69)}.ZoomPercent^{(168)}$  (i.e. a page height as it can be seen in the control window).

**PageHeight100Pix** returns the page height converted to screen pixels in 96 DPI and  $ZoomPercent=100$ :

$$\text{PageHeightPix} := \text{PageHeight100Pix} * (\text{ZoomPercent} / 100) * (<\text{screen DPI}> / 96).$$

The same value measured in  $\text{UnitsProgram}^{(68)}$  is returned in  $\text{ViewProperty}^{(69)}.PageHeight^{(162)}$ .

The page height is defined in  $\text{PageProperty}^{(58)}.PageHeight^{(139)}$ .

**See also:**

- PageWidthPix<sup>(58)</sup>

#### 1.3.1.1.40 TSRichViewEdit.PageNoMouseDown

Returns the index of page (from 1) where a mouse button was pressed last time.

**property** PageNoMouseDown: Integer;

#### 1.3.1.1.41 TSRichViewEdit.PageNoMouseMove

Returns the index of page (from 1) below the mouse pointer.

**property** PageNoMouseMove: Integer;

#### 1.3.1.1.42 TSRichViewEdit.PagePosProperty

Pages positioning properties.

**property** PagePosProperty: TSRVPagePositionProperty<sup>(143)</sup>;

**See also:**

- ViewProperty<sup>(69)</sup>.FreePosPage<sup>(158)</sup>

#### 1.3.1.1.43 TSRichViewEdit.PageProperty

Page properties.

**property** PageProperty: TSRVPageProperty<sup>(133)</sup>;

#### 1.3.1.1.44 TSRichViewEdit.PageWidthPix, PageWidth100Pix

The properties return a page width of document on the screen.

**property** PageWidthPix: Integer;

**property** PageWidth100Pix: Integer;

**PageWidthPix** returns the page width converted to screen pixels and scaled according to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> (i.e. a page width as it can be seen in the control window).

**PageWidth100Pix** returns the page height converted to screen pixels in 96 DPI and ZoomPercent=100:

**PageWidthPix** := **PageWidth100Pix** \* (ZoomPercent / 100) \* (<screen DPI> / 96).

The same value measured in UnitsProgram<sup>(68)</sup> is returned in ViewProperty<sup>(69)</sup>.PageWidth<sup>(162)</sup>.

The page width is defined in PageProperty<sup>(58)</sup>.PageWidth<sup>(141)</sup>.

**See also:**

- PageHeightPix<sup>(57)</sup>

#### 1.3.1.1.45 TSRichViewEdit.RangeSearch

Allows to narrow a search scope.

**RangeSearch**: Boolean;

If it is *True*, the search is performed only in items located between coordinates specified in RangeSearchFirstY<sup>(59)</sup>..RangeSearchLastY<sup>(59)</sup> properties.

This range affects the following methods:

- FindPriorItem<sup>(81)</sup>, FindNextItem<sup>(79)</sup>,

- FindPriorHyperlink<sup>(81)</sup>, FindNextHyperlink<sup>(79)</sup>,
- FindPriorCheckpoint<sup>(80)</sup>, FindNextCheckpoint<sup>(78)</sup>,
- PriorCurlItem<sup>(95)</sup>, NextCurlItem<sup>(91)</sup>,
- PriorCurHyperlink<sup>(94)</sup>, NextCurHyperlink<sup>(91)</sup>,
- PriorCurCheckpoint<sup>(94)</sup>, NextCurCheckpoint<sup>(90)</sup>.

This feature can be used, for example, to search only on the some pages.

**Example:** searching for the first visible hyperlink (if any hyperlink is visible):

```
var
  RVData: TCustomRVData;
  RVData2: TCustomRVFormattedData;
  ItemNo, OffsetInItem: Integer;
..
SRichViewEdit1.RangeSearch := True;
SRichViewEdit1.RangeSearchFirstY(59) := 0;
SRichViewEdit1.RangeSearchLastY(59) := SRichViewEdit1.ClientHeight;
SRichViewEdit1.GetFirstItemVisibleRV(83)(RVData2, ItemNo, OffsetInItem);
RVData := RVData2;
Dec(ItemNo);
if FindNextHyperlink(79)(RVData, ItemNo) then
  ...
```

#### 1.3.1.1.46 TSRichViewEdit.RangeSearchFirstY

Allows to narrow a search scope.

RangeSearchFirstY: Integer;

This is a client coordinate (relative to the top of the component window).

See RangeSeach<sup>(58)</sup> property.

#### 1.3.1.1.47 TSRichViewEdit.RangeSearchLastY

Allows to narrow a search scope.

RangeSearchLastY: Integer;

This is a client coordinate (relative to the top of the component window).

See RangeSeach<sup>(58)</sup> property.

#### 1.3.1.1.48 TSRichViewEdit.ReadModeProperty

Contains settings for a read mode and side-to-side page displaying mode.

**property** ReadModeProperty: TSRVReadModeProperty<sup>(151)</sup>;

#### 1.3.1.1.49 TSRichViewEdit.ReadOnly

When set to *True*, this property prevents user's input in the editor; it also blocks execution of all editing-style methods of RichViewEdit<sup>(60)</sup>.

**property** ReadOnly : Boolean;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

### 1.3.1.1.50 TSRichViewEdit.RichViewEdit

Returns TRichViewEdit component used to render documents.

**property** RichViewEdit: TRichViewEdit;

This editor is used to render and edit the main part of document.

Other editors are used to render different parts of document:

- RVHeader<sup>64</sup> for page header;
- RVFooter<sup>62</sup> for page footer;
- RVNote<sup>64</sup> for footnotes and endnotes.

The active editor is returned in ActiveEditor<sup>47</sup> property.

**See also:**

- Document parts in TSRichViewEdit<sup>37</sup>

### 1.3.1.1.51 TSRichViewEdit.RightMarginPix

Returns the right margin, in pixels.

**property** RightMarginPix: Integer;

**property** RightMargin100Pix: Integer;

**function** GetRightMarginPix(PageNo : Integer) : Integer;

**function** GetRightMargin100Pix(PageNo : Integer) : Integer

The right margin is defined in PageProperty<sup>58</sup>.RightMargin<sup>142</sup>.

#### Properties

These properties return this margin converted to pixels.

**RightMarginPix** is scaled according to ViewProperty<sup>69</sup>.ZoomPercent<sup>168</sup> (so this property returns the margin as it is displayed on the screen).

**RightMargin100Pix** is not scaled, it is the margin in 96 DPI and 100% zooming:

$$\text{RightMarginPix} := \text{RightMargin100Pix} * (\text{ZoomPercent} / 100) * (<\text{screen DPI}> / 96).$$

#### Methods

The methods return the right margin for the specified page, taking PageProperty<sup>58</sup>.MirrorMargins<sup>138</sup> into account.

**See also:**

- LeftMarginPix<sup>54</sup>
- TopMarginPix<sup>68</sup>
- BottomMarginPix<sup>48</sup>

### 1.3.1.1.52 TSRichViewEdit.RTFOptions

Options for saving RTF (Rich Text Format).

**property** RTFOptions: TRVRTFOptions;

This property provides access to RichViewEdit<sup>60</sup>'s property of the same name.

The initial value of this property for an internal TRichViewEdit includes *rvrtfSaveDocParameters* (in addition to the options included in the initial value of this property for TRichView)



#### 1.3.1.1.53 TSRichViewEdit.RTFReadProperties

A set of properties controlling RTF (Rich Text Format) import.

**property** RTFReadProperties: TRVRTFReaderProperties;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

The initial value of this property for an internal TRichViewEdit has:

- RTFReadProperties.ReadDocParameters = *True*.

#### 1.3.1.1.54 TSRichViewEdit.RVBackgroundBitmap

Background image for the the document

**property** RVBackgroundBitmap: TBitmap;

Deprecated, use RVBackgroundPicture<sup>(61)</sup> instead.

Provides access to RichViewEdit<sup>(60)</sup>.BackgroundBitmap property.

**See also:**

- RVBackgroundProperties<sup>(61)</sup>
- RVColor<sup>(62)</sup>

#### 1.3.1.1.55 TSRichViewEdit.RVBackgroundPicture

Background image for the document

**property** RVBackgroundPicture: TRVPicture;

Provides access to RichViewEdit<sup>(60)</sup>.BackgroundPicture property.

**See also:**

- RVBackgroundProperties<sup>(61)</sup>
- RVColor<sup>(62)</sup>

#### 1.3.1.1.56 TSRichViewEdit.RVBackgroundProperties

Defines how RVBackgroundPicture<sup>(61)</sup> is displayed: visibility, size, position, repetition.

**property** RVBackgroundProperties: TRVBackgroundProperties;

Provides access to RichViewEdit<sup>(60)</sup>.BackgroundProperties property.

**See also:**

- RVColor<sup>(62)</sup>

#### 1.3.1.1.57 TSRichViewEdit.RVBackgroundStyle

Defines how RVBackgroundPicture<sup>(61)</sup> is displayed

**property** RVBackgroundStyle: TBackgroundStyle;

Deprecated, use RVBackgroundProperties<sup>(61)</sup> instead.

Provides access to RichViewEdit<sup>(60)</sup>.BackgroundStyle property.

**See also:**

- RVColor<sup>(62)</sup>

### 1.3.1.1.58 TSVRichViewEdit.RVColor

Background color of document.

**property** RVColor: TColor;

Provides access to RichViewEdit<sup>(60)</sup>.Color property.

**See also:**

- RVBackgroundProperties<sup>(61)</sup>
- RVBackgroundPicture<sup>(61)</sup>

### 1.3.1.1.59 TSVRichViewEdit.RVEditorOptions

Options affecting editing in TRichViewEdit.

**property** RVEditorOptions: TRVEditorOptions;

Provides access to RichViewEdit<sup>(60)</sup>.EditorOptions property.

### 1.3.1.1.60 TSVRichViewEdit.RVFooter

Specifies the TRichViewEdit control used for editing footers.

**property** RVFooter: TRichViewEdit;

Footers are visible if PageProperty<sup>(58)</sup>.FooterVisible<sup>(136)</sup> = True.

When footer editing is activated, a document is loaded from Subdocuments<sup>(66)</sup> to RVFooter. Depending on the current page<sup>(51)</sup>, PageProperty<sup>(58)</sup>.TitlePage<sup>(142)</sup> and PageProperty<sup>(58)</sup>.FacingPages<sup>(136)</sup>, a document is loaded from one of the following Subdocuments.Items: *srvhftFirstPageFooter*, *srvhftEvenPagesFooter*, *srvhftNormalFooter*. An edited document is saved back when editing is finished, or when the editor needs to be saved or printed.

When this editor is active, it is returned in ActiveEditor<sup>(47)</sup> property. Only in this case (RVFooter=ActiveEditor) you can assume that the document in it is valid.

Distances from the page border to the footer are defined in PageProperty<sup>(58)</sup>:

- LeftMargin<sup>(138)</sup> (from the left side)
- RightMargin<sup>(142)</sup> (from the right side)
- FooterY<sup>(137)</sup> (from the bottom side)

Margins properties of this editor (RVFooter.LeftMargin, TopMargin, etc.) are ignored when calculating the footer position.

When the footer is being edited, it is visually indicated by the following properties of ViewProperty<sup>(69)</sup>:

- Texts<sup>(163)</sup>
- FooterPen<sup>(157)</sup>
- FooterTitleFont<sup>(158)</sup>
- FooterTitleColor<sup>(157)</sup>

#### Using an external editor (not recommended!)

You can assign an external TRichViewEdit control to this property. Take a note that width and height of this RichViewEdit are set by TSVRichViewEdit.

Assigning *nil* to this property makes `TSRichViewEdit` to use a hidden internal `TRichViewEdit` for the footer (this is a default behavior). Users cannot see it, it is responsible only for document rendering.

**See also:**

- `RVHeader`<sup>64</sup>
- `RVNote`<sup>64</sup>
- `RichViewEdit`<sup>60</sup>
- Document parts in `TSRichViewEdit`<sup>37</sup>

#### 1.3.1.1.61 `TSRichViewEdit.RVFOptions`

Options for reading and saving RVF (RichView Format)

**property** `RVFOptions`: `TRVFOptions`;

This property provides access to `RichViewEdit`<sup>60</sup>'s property of the same name.

**Initial value (for internal `RichViewEdit`):**

`[rvfoLoadDocProperties, rvfoSaveDocProperties,  
rvfoSavePicturesBody, rvfoSaveControlsBody,  
rvfoIgnoreUnknownPicFmt, rvfoIgnoreUnknownCtrls,  
rvfoConvUnknownStylesToZero, rvfoConvLargeImageIdxToZero,  
rvfoSaveBinary, rvfoSaveBack, rvfoLoadBack,  
rvfoSaveTextStyles, rvfoSaveParaStyles, rvfoSaveLayout,  
rvfoLoadLayout]`

**See also:**

- `RVFParaStylesReadMode`<sup>63</sup>
- `RVFTextStylesReadMode`<sup>63</sup>

#### 1.3.1.1.62 `TSRichViewEdit.RVFParaStylesReadMode`

Specifies how paragraph styles and list styles are loaded from RVF files or streams.

**property** `RVFParaStylesReadMode`: `TRVFReaderStyleMode`;

This property provides access to `RichViewEdit`<sup>60</sup>'s property of the same name.

**See also:**

- `RVFTextStylesReadMode`<sup>63</sup>
- `RVFOptions`<sup>63</sup>

#### 1.3.1.1.63 `TSRichViewEdit.RVFTextStylesReadMode`

Specifies how text styles are loaded from RVF files or streams.

**property** `RVFTextStylesReadMode`: `TRVFReaderStyleMode`;

This property provides access to `RichViewEdit`<sup>60</sup>'s property of the same name.

**See also:**

- `RVFParaStylesReadMode`<sup>63</sup>
- `RVFOptions`<sup>63</sup>

### 1.3.1.1.64 TSRichViewEdit.RVHeader

Specifies the TRichViewEdit control used for editing headers.

**property** RVHeader: TRichViewEdit;

Headers are visible if PageProperty<sup>(58)</sup>.HeaderVisible<sup>(137)</sup> = True.

When header editing is activated, a document is loaded from Subdocuments<sup>(66)</sup> to RVHeader. Depending on the current page<sup>(51)</sup>, PageProperty<sup>(58)</sup>.TitlePage<sup>(142)</sup> and PageProperty<sup>(58)</sup>.FacingPages<sup>(136)</sup>, a document is loaded from one of the following Subdocuments.Items: *srvhftFirstPageHeader*, *srvhftEvenPagesHeader*, *srvhftNormalHeader*. An edited document is saved back when editing is finished, or when the editor needs to be saved or printed.

When this editor is active, it is returned in ActiveEditor<sup>(47)</sup> property. Only in this case (RVHeader=ActiveEditor) you can assume that the document in it is valid.

Distances from the page border to the header are defined in PageProperty<sup>(58)</sup>:

- LeftMargin<sup>(138)</sup> (from the left side)
- RightMargin<sup>(142)</sup> (from the right side)
- HeaderY<sup>(137)</sup> (from the top side)

Margins properties of this editor (RVHeader.LeftMargin, TopMargin, etc.) are ignored when calculating the header position.

When the header is being edited, it is visually indicated by the following properties of ViewProperty<sup>(69)</sup>:

- Texts<sup>(163)</sup>
- HeaderPen<sup>(159)</sup>
- HeaderTitleFont<sup>(159)</sup>
- HeaderTitleColor<sup>(159)</sup>

#### Using an external editor (not recommended!)

You can assign an external TRichViewEdit control to this property. Take a note that width and height of this RichViewEdit are set by TSRichViewEdit.

Assigning *nil* to this property makes TSRichViewEdit to use a hidden internal TRichViewEdit for the header (this is a default behavior). Users cannot see it, it is responsible only for document rendering.

#### See also:

- RVFooter<sup>(62)</sup>
- RVNote<sup>(64)</sup>
- RichViewEdit<sup>(60)</sup>
- Document parts in TSRichViewEdit<sup>(37)</sup>

### 1.3.1.1.65 TSRichViewEdit.RVNote

Returns TRichViewEdit control used to edit notes and text boxes.

**property** RVNote: TRichViewEdit;

Unlike other similar properties (RVHeader<sup>(64)</sup>, RVFooter<sup>(62)</sup>, RichViewEdit<sup>(60)</sup>), the content of this editor is not persistent after document loading. This TRichViewEdit control is used to edit different

footnotes/endnotes (see `StartEditNote`<sup>(99)</sup>). The content of this editor is defined only if `CurrentNote`<sup>(50)</sup> `<>nil`.

When this editor is active, it is returned in `ActiveEditor`<sup>(47)</sup> property.

Documents of items of the following types can be edited:

- footnotes (`TRVFootnoteItemInfo`)
- endnotes (`TRVEndnoteItemInfo`)
- sidenotes (`TRVSidenoteItemInfo`)
- text box items (`TRVTextBoxItemInfo`)

**See also:**

- `RVHeader`<sup>(64)</sup>
- `RVFooter`<sup>(62)</sup>
- `RichViewEdit`<sup>(60)</sup>
- Document parts in `TSRichViewEdit`<sup>(37)</sup>
- Footnotes and endnotes<sup>(38)</sup>

### 1.3.1.1.66 TSRichViewEdit.RVOptions

Options for `TRichViewEdit`.

**property** `RVOptions: TRVOptions;`

Provides access to `RichViewEdit`<sup>(60)</sup>.Options property.

**Initial value (for internal RichViewEdit):**

*[rvoAllowSelection, rvoScrollToEnd, rvoTagsArePChars,  
rvoAutoCopyText, rvoAutoCopyRVF, rvoAutoCopyImage,  
rvoAutoCopyRTF, rvoFormatInvalidate, rvoDbClickSelectsWord,  
rvoRClickDeselects]*

### 1.3.1.1.67 TSRichViewEdit.ScrollBarControlStyle

Defines appearance of scrollbars

**property** `ScrollBarControlStyle: TSRVControlStyle`<sup>(276)</sup>;

Internally, scrollbars are represented by `TSRVScrollBar`<sup>(243)</sup> controls. Value of this property is assigned to `SRVControlStyle`<sup>(248)</sup> property of these scrollbars.

**Default value:**

*srvcClassic*

**See also**

- `ViewProperty`<sup>(69)</sup>.`DarkModeUI`<sup>(157)</sup>
- `HScrollBarSchemeIndex`<sup>(53)</sup>
- `VScrollBarSchemeIndex`<sup>(70)</sup>

### 1.3.1.1.68 TSRichViewEdit.ScrolledPage

Returns the index of the first visible page (from 1).

**property** `ScrolledPage: Integer;`

This value is not necessary the same as `CurrentPage`<sup>(51)</sup>.

### 1.3.1.1.69 TSRichViewEdit.SkinManager

Determines which skin manager is associated with this editor.

**property** SkinManager: TSRVSkinManager<sup>(228)</sup>;

SkinManager defines a visual appearance of scrollbars. If it is not specified, scrollbars have a default (no-skin) view.

**See also:**

- HScrollBarSchemeIndex<sup>(53)</sup>
- VScrollBarSchemeIndex<sup>(70)</sup>

### 1.3.1.1.70 TSRichViewEdit.SmartPopupProperties

Properties for "smart popups"

**property** SmartPopupProperties: TRVSmartPopupProperties;

Provides access to RichViewEdit<sup>(60)</sup>.SmartPopupProperties property.

TSRichViewEdit uses RichViewEdit<sup>(60)</sup>.SmartPopupProperties not only for "smart popups" in RichViewEdit<sup>(60)</sup>, but for "smart popups" in other editors as well (RVHeader<sup>(64)</sup>, RVFooter<sup>(62)</sup>, RVNote<sup>(64)</sup>). Values of SmartPopupProperties of these other editors are ignored.

You can show/hide a button using SmartPopupVisible<sup>(66)</sup> property.

**See also:**

- OnSmartPopupClick<sup>(120)</sup>

### 1.3.1.1.71 TSRichViewEdit.SmartPopupVisible

Shows/hides a "smart popup" button for the current item in ActiveEditor<sup>(47)</sup>.

**property** SmartPopupVisible: Boolean;

Properties of this button are defined in SmartPopupProperties<sup>(66)</sup>.

**See also:**

- OnSmartPopupClick<sup>(120)</sup>

### 1.3.1.1.72 TSRichViewEdit.StyleTemplateInsertMode

Controls how RichViewEdit<sup>(60)</sup>.Style.StyleTemplates affect insertion of RTF and RVF files or streams.

**property** StyleTemplateInsertMode: TRVStyleTemplateInsertMode;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

**Default value**

*rvstimUseTargetFormatting*

### 1.3.1.1.73 TSRichViewEdit.Subdocuments

A list of headers and footers.

**type**

TSRVSubDocuments = **class** (TPersistent)

**property** SubDocuments: TSRVSubDocuments;

The main property of TSRVSubDocuments is Items:

#### type

```
TSRVHeaderFooterType = (
    srvhftFirstPageHeader, srvhftEvenPagesHeader, srvhftNormalHeader,
    srvhftFirstPageFooter, srvhftEvenPagesFooter, srvhftNormalFooter);
TSRVSubDocumentRVData = class (TCustomRVData)
```

**property** Items[Index: TSRVHeaderFooterType]: TSRVSubDocumentRVData;

The Index specifies the header or footer type:

| Value                        | Meaning                         | Used if...  |
|------------------------------|---------------------------------|---|
| <i>srvhftFirstPageHeader</i> | Header for the first page       | PageProperty <sup>(58)</sup> .TitlePage <sup>(142)</sup><br>=True |
| <i>srvhftEvenPagesHeader</i> | Header for even pages           | PageProperty.FacingPages <sup>(136)</sup><br>=True                |
| <i>srvhftNormalHeader</i>    | Normal header (for other pages) | Always  |
| <i>srvhftFirstPageFooter</i> | Footer for the first page       | PageProperty.TitlePage=True                                       |
| <i>srvhftEvenPagesFooter</i> | Footer for even pages           | PageProperty.FacingPages=True                                     |
| <i>srvhftNormalFooter</i>    | Normal footer (for other pages) | Always  |

Items is the default property, so you can access (for example) the first page header as `SRichViewEdit.Subdocuments[srvhftFirstPageHeader]`.

Headers are shown and used only if `PageProperty(58).HeaderVisible(137)=True`. Footers are shown and used only if `PageProperty.FooterVisible(136)=True`.

Each subdocument is linked to its own TRVStyle object. You can access it as `SRichViewEdit.Subdocuments[srvhftFirstPageHeader].GetRVStyle`. `RichViewEdit(60).Style` is assigned as `MainRVStyle` to these TRVStyle objects, so they all use a common collection of StyleTemplates.

When an event occurs in some subdocument, `RichViewEdit(60)` is passed as the Sender parameter of this event.

### Editing

When a header is activated for editing (by double clicking or by calling `StartEditing(99)`), a document is loaded in `RVHeader(64)`. When a footer is activated for editing, a document is loaded in `RVFooter(62)`. An edited document is saved back to Subdocuments when the editing is finished, or when the main document needs to be saved or printed. The process of saving and loading data between Subdocuments and RVHeader/RVFooter is completely automatic and transparent for the user.

### Saving to files

Headers and footers can be saved in RVF, RTF, DocX files, loaded from RVF, RTF files.

`PageProperty.TitlePage` and `FacingPages` only turn on/off displaying of the specific headers and footers. They do not affect saving and loading: all headers and footers are stored.

Assigning `PageProperty.HeaderVisible=False` prevents saving and loading headers.

Assigning `PageProperty.FooterVisible=False` prevents saving and loading footers.

#### 1.3.1.1.74 TSRichViewEdit.TabNavigation

Mode of **Tab** and **Shift + Tab** navigation

**property** `TabNavigation: TRVTabNavigationType;`

This property provides access to `RichViewEdit`<sup>(60)</sup>'s property of the same name.

This property does nothing in `TSRichViewEdit`. Reserved for future.

#### 1.3.1.1.75 TSRichViewEdit.TextEngine

Specifies the text API which is used to draw text.

**property** `TextEngine: TRVTextEngine;`

Uniscribe provides reliable processing of bidirected text and complex scripts.

Windows API is faster.

**Default value:**

*rvteUniscribe*

#### 1.3.1.1.76 TSRichViewEdit.TopMarginPix

Returns the top margin, in pixels.

**property** `TopMarginPix: Integer;`

**property** `TopMargin100Pix: Integer;`

The top margin is defined in `PageProperty`<sup>(58)</sup>.`TopMargin`<sup>(142)</sup>. These properties return this margin converted to pixels.

**TopMarginPix** is scaled according to `ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup> (so this property returns the margin as it is displayed on the screen).

**TopMargin100Pix** is not scaled, it is the margin in 96 DPI and 100% zooming:

**TopMarginPix** := **TopMargin100Pix** \* (ZoomPercent / 100) \* (<screen DPI> / 96).

**See also:**

- `LeftMarginPix`<sup>(54)</sup>
- `RightMarginPix`<sup>(60)</sup>
- `BottomMarginPix`<sup>(48)</sup>

#### 1.3.1.1.77 TSRichViewEdit.UnitsProgram

Measurement units for page margins and size.

**property** `UnitsProgram: TRVUnits;`

This property defines measurement units for:

- page size and margins properties of `PageProperty`<sup>(58)</sup>,
- `PageWidth`<sup>(162)</sup> and `PageHeight`<sup>(162)</sup> properties of `ViewProperty`<sup>(69)</sup>.

Note: "pixels" unit assumes 96 dpi (used internally in `ScaleRichView`).

**Default value:**



*rvuMillimeters*

#### 1.3.1.1.78 TSRichViewEdit.UseDrawHyperlinksEvent

Specifies whether OnDrawHyperlink<sup>(107)</sup> event is used.

**property** UseDrawHyperlinksEvent: Boolean;

This property may be useful if you want to use this event only when drawing on external canvas (DrawMetafile<sup>(76)</sup>, DrawPage<sup>(77)</sup>).

**Default value:**

*True*

#### 1.3.1.1.79 TSRichViewEdit.UseStyleTemplates

Allows or disallows using RichViewEdit<sup>(60)</sup>.Style.StyleTemplates ("real styles").

**property** UseStyleTemplates: Boolean;

This property provides access to RichViewEdit<sup>(60)</sup>'s property of the same name.

**Default value:**

*False*

#### 1.3.1.1.80 TSRichViewEdit.Version

Returns the version of ScaleRichView.

**property** Version : TRVUnicodeString;

It is a string like 'v1.6.0'.

#### 1.3.1.1.81 TSRichViewEdit.ViewProperty

Contains settings for document displaying.

**property** ViewProperty: TSRVViewProperty<sup>(154)</sup>;

#### 1.3.1.1.82 TSRichViewEdit.VMaxScrollPos

Defines the maximum position of vertical scroll bar.

**property** VMaxScrollPos: Integer;

This value is used to calculate the maximum possible value for VScrollPos<sup>(70)</sup>.

This is a read-only property.

**See also:**

- HMaxScrollPos<sup>(52)</sup>

#### 1.3.1.1.83 TSRichViewEdit.VScrollBar

Shows/hides the vertical scrollbar.

**property** VScroolBar: Boolean;

**Default value:**

*True* (visible)

### 1.3.1.1.84 TSRichViewEdit.VScrollBarSchemeIndex

Determines which skin scheme is used for the vertical scrollbar.

**property** VScrollBarSchemeIndex: Integer;

This is an index in SkinManager<sup>(66)</sup>.CurrentSkin<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup> collection.

If SkinManager<sup>(66)</sup> = *nil*, or SkinManager<sup>(66)</sup>.CurrentSkin<sup>(230)</sup> = *nil*, or this index is not valid for this collection (negative or too large), then this scrollbar has a default (no-skin) appearance.

#### Default value

0

#### See also

- HScrollBarSchemeIndex<sup>(53)</sup>
- ScrollBarControlStyle<sup>(65)</sup>

### 1.3.1.1.85 TSRichViewEdit.VScrollPos

Defines the position of vertical scroll bar, measured in screen pixels.

**property** VScrollPos: Integer;

You can use this property for scrolling to the specified position.

Value of this property is in the range from 0 to VMaxScrollPos<sup>(69)</sup> - (ClientWidth div RichViewEdit<sup>(60)</sup>.VSmallStep) + 1

#### See also:

- HScrollPos<sup>(53)</sup>

## 1.3.1.2 Methods

### In TSRichViewEdit

BeginComponentsUpdate<sup>(72)</sup>  
 CalculateAllPagePositions<sup>(72)</sup>  
 CalculatePageCount<sup>(72)</sup>  
 CalculatePagePosition<sup>(73)</sup>  
 Clear<sup>(73)</sup>  
 ConvertRVtoSRV<sup>(73)</sup>  
 ConvertSRVtoRV<sup>(74)</sup>  
 ConvertToPixels, ConvertToTwips, ConvertToDifferentUnits<sup>(75)</sup>  
 Copy<sup>(75)</sup>  
 Create<sup>(75)</sup>  
 Cut<sup>(75)</sup>  
 DeletePage<sup>(76)</sup>  
 Destroy<sup>(76)</sup>  
 DrawMetafile<sup>(76)</sup>  
 DrawPage<sup>(77)</sup>  
 EndComponentsUpdate<sup>(77)</sup>  
 FindNextCheckpoint<sup>(78)</sup>  
 FindNextHeading<sup>(78)</sup>  
 FindNextHyperlink<sup>(79)</sup>

FindNextItem<sup>79</sup>  
FindPriorCheckpoint<sup>80</sup>  
FindPriorHeading<sup>80</sup>  
FindPriorHyperlink<sup>81</sup>  
FindPriorItem<sup>81</sup>  
FirstCurPage<sup>82</sup>  
Format<sup>82</sup>  
GetCaretPosInUnits<sup>83</sup>  
GetCurrentLineCol<sup>83</sup>  
GetFirstItemVisibleRV<sup>83</sup>  
GetFirstVisiblePage<sup>84</sup>  
GetItemAt<sup>84</sup>  
GetItemBounds, GetItemBounds100<sup>84</sup>  
GetItemCoords<sup>85</sup>  
GetItemPages<sup>86</sup>  
GetLastItemVisibleRV<sup>86</sup>  
GetLastVisiblePage<sup>87</sup>  
GetLeftMarginPix, GetLeftMargin100Pix<sup>54</sup>  
GetPageAt<sup>87</sup>  
GetPageClientRect<sup>87</sup>  
GetPageLastItemNo<sup>87</sup>  
GetPageNo<sup>88</sup>  
GetPageStartItemNo<sup>88</sup>  
GetPageStartTableItem<sup>88</sup>  
GetPageStartTableRow<sup>89</sup>  
GetRightMarginPix, GetRightMargin100Pix<sup>60</sup>  
GetTableIconItem<sup>89</sup>  
GetTableIconRVData<sup>89</sup>  
InsertPageBreak<sup>89</sup>  
InZoomRect<sup>90</sup>  
LastCurPage<sup>90</sup>  
LoadRVF<sup>90</sup>  
LoadRVFFromStream<sup>90</sup>  
NextCurCheckpoint<sup>90</sup>  
NextCurHeading<sup>91</sup>  
NextCurHyperlink<sup>91</sup>  
NextCurItem<sup>91</sup>  
NextCurPage<sup>91</sup>  
OutZoomRect<sup>92</sup>  
Paste<sup>92</sup>  
PrintAll<sup>92</sup>  
PrintCurrent<sup>93</sup>  
PrintRange<sup>93</sup>  
PriorCurCheckpoint<sup>94</sup>  
PriorCurHeading<sup>94</sup>  
PriorCurHyperlink<sup>94</sup>  
PriorCurItem<sup>95</sup>  
PriorCurPage<sup>95</sup>

ProcessControls<sup>(95)</sup>  
 Reformat<sup>(82)</sup>  
 ReturnToNote<sup>(95)</sup>  
 ScrollToCaret<sup>(96)</sup>  
 ScrollToItem<sup>(96)</sup>  
 SelectAll<sup>(97)</sup>  
 SetFloatPropertyEd<sup>(97)</sup>  
 SetIntPropertyEd<sup>(97)</sup>  
 SetMargin<sup>(97)</sup>  
 SetMarginMM<sup>(97)</sup>  
 SetMarginUnit<sup>(98)</sup>  
 SetRVMargins<sup>(98)</sup>  
 StartEditing<sup>(99)</sup>  
 StartEditNote<sup>(99)</sup>  
 UnitsPerInchH<sup>(100)</sup>  
 UnitsPerInchV<sup>(100)</sup>  
 UpdateBuffer, UpdateBufferAt<sup>(100)</sup>  
 ZoomCanvas, RestoreCanvasZoom<sup>(100)</sup>  
 ZoomRectIn, ZoomRectOut<sup>(101)</sup>  
 ZoomToFullPages<sup>(101)</sup>

#### 1.3.1.2.1 TSRichViewEdit.BeginComponentsUpdate

Locks redrawing of inserted controls.

**procedure** BeginComponentsUpdate;

Inserted components will not be repainted until EndComponentsUpdate<sup>(77)</sup> is called.

Calls to BeginComponentsUpdate and EndComponentsUpdate may be nested.

If redrawing of inserted controls is locked, repainting is faster.

#### 1.3.1.2.2 TSRichViewEdit.CalculateAllPagePositions

Recalculates positions, zooming and Z-orders for all pages.

**procedure** CalculateAllPagePositions;

This method calls OnGetPagePos<sup>(108)</sup> event for all pages.

This event is used only if ViewProperty<sup>(69)</sup>.FreePosPage<sup>(158)</sup> = *True*. Usually page positions are calculated automatically. Call this method if you want to recalculate them for existing pages.

Z-orders are recalculated only if PageProperty<sup>(58)</sup>.UsePageOrders<sup>(142)</sup> = *True*.

**See also:**

- CalculatePagePosition<sup>(73)</sup>

#### 1.3.1.2.3 TSRichViewEdit.CalculatePageCount

Repaginates and returns the page count.

**function** CalculatePageCount: Integer;

Usually, if a document is changed by editing-style methods, it is repaginated automatically.

Call this method in the following cases:

- if you modified a document using viewer-style methods (methods that do not call OnChange<sup>(104)</sup>)
- if you blocked updating with CanUpdate<sup>(49)</sup> property

**Return value:**

count of pages

**See also:**

- PageCount<sup>(57)</sup> property

#### 1.3.1.2.4 TSRichViewEdit.CalculatePagePosition

Recalculates position, zooming and Z-order for the specified page.

**function** CalculatePagePosition(PageNo : Integer) : Boolean;

**PageNo** is a page index, from 1.

This method calls OnGetPagePos<sup>(108)</sup> event for this page.

This event is used only if ViewProperty<sup>(69)</sup>.FreePosPage<sup>(158)</sup> = *True*. Usually page positions are calculated automatically. Call this method if you want to recalculate them for the existing page.

Z-orders are recalculated only if PageProperty<sup>(58)</sup>.UsePageOrders<sup>(142)</sup> = *True*.

**See also:**

- CalculateAllPagePositions<sup>(72)</sup>

#### 1.3.1.2.5 TSRichViewEdit.Clear

Clears the whole editor content and deletes all unused styles.

**procedure** Clear;

This method calls Clear and DeleteUnusedStyles(True,True,True) for:

- RichViewEdit<sup>(60)</sup>
- Subdocuments<sup>(66)</sup>
- RVHeader<sup>(64)</sup>
- RVFooter<sup>(62)</sup>

**See also:**

- Document parts in TSRichViewEdit<sup>(37)</sup>

#### 1.3.1.2.6 TSRichViewEdit.ConvertRVtoSRV

Converts client coordinates in ActiveEditor<sup>(47)</sup> to client coordinates of this TSRichViewEdit component.

**function** ConvertRVtoSRV(p: TPoint) : TPoint;

**Input parameters:**

**p** – coordinates relative to the top left corner of ActiveEditor<sup>(47)</sup>.

**Return value:**

coordinates relative to the top left corner of TSRichViewEdit's window.

**See also**

- ConvertSRVtoRV<sup>(74)</sup>

#### 1.3.1.2.7 TSRichViewEdit.ConvertSRVtoRV

Converts client coordinates in this TSRichViewEdit component to client coordinates of ActiveEditor<sup>(47)</sup>.

**function** ConvertSRVtoRV(p: TPoint; **var** bInTextArea, bInLeft, bInPageArea: Boolean)

**Input parameters:**

**p** – coordinates relative to the top left corner of TSRichViewEdit's window.

**Output parameters:**

**bInTextArea** – *True* if **p** is inside of some page's content area.

**bInLeft** – *True* if **p** is to the left of some page (in the "line selection" area).

**InPageArea** – *True* if **p** is inside of some page's area (above content or above margins).

**Return value:**

coordinates relative to the top left corner of ActiveEditor<sup>(47)</sup>.

**See also:**

- ConvertRVToSRV<sup>(73)</sup>

**Example:**

This code assigns text of the item at the position of the mouse pointer to Label1.Caption

*// OnMouseMove event*

```
procedure TForm3.SRichViewEdit1MouseMove(Sender: TObject;
  Shift: TShiftState; X, Y: Integer);
var pt: TPoint;
    RVData: TCustomRVFormattedData;
    InPageArea, InLeftArea, InTextArea: Boolean;
    ItemNo, Offs: Integer;
begin
  pt := SRichViewEdit1.ConvertSRVtoRV(Point(X, Y), InTextArea,
    InLeftArea, InPageArea);
  if InTextArea then begin
    pt := SRichViewEdit1.ActiveEditor(47).ClientToDocument(pt);
    SRichViewEdit1.ActiveEditor(47).GetItemAt(pt.X, pt.Y,
      RVData, ItemNo, Offs, True);
    if ItemNo >= 0 then
      Label1.Caption := RVData.GetItemText(ItemNo)
    else // in page, but not in item
      Label1.Caption := '';
    end
  else // not in page
    Label1.Caption := '';
end;
```

### 1.3.1.2.8 TSVRichViewEdit.ConvertTo...

The methods convert lengths in the documents and RVStyle components to twips, pixels or EMU.

```
procedure ConvertDocToTwips;
```

```
procedure ConvertDocToPixels;
```

```
procedure ConvertDocToEMU;
```

```
procedure ConvertDocToDifferentUnits(NewUnits: TRVStyleUnits);
```

The methods convert documents and RVStyles of the following properties to new units:

- RichViewEdit<sup>60</sup>
- Subdocuments<sup>66</sup>
- RVFooter<sup>62</sup>
- RVHeader<sup>64</sup>
- RVNote<sup>64</sup> (if a note is being edited)

The methods convert documents in these editors (using ConvertDocTo\* methods) and their RVStyles (using Style.ConvertTo\* methods).

If you do not change RVStyles of these editors (and *rvfoCanChangeUnits* is not included in their RVFOptions), it is enough to call a conversion method one time, unit of measurement will be kept.

### 1.3.1.2.9 TSVRichViewEdit.Copy

Copies the selected fragment to the Clipboard. Copies in all supported formats (such as text, RVF, RTF and images).

```
procedure Copy;
```

This procedure does nothing if the selection is empty.

This procedure empties the Clipboard before copying.

This method should be called only when document is formatted.

This method calls RichViewEdit<sup>60</sup>.CopyDef.

**See also:**

- OnCopy<sup>106</sup>

### 1.3.1.2.10 TSVRichViewEdit.Create

A constructor, creates a new instance of TSVRichViewEdit.

```
constructor Create(AOwner: TComponent);
```

**AOwner** is another component, typically the form, that is responsible for freeing the control. It becomes the value of the Owner property.

Controls placed on form at design time are created automatically.

### 1.3.1.2.11 TSVRichViewEdit.Cut

Cuts the selected fragment to the Clipboard.

```
procedure Cut;
```

This procedure does nothing if the selection is empty.

This procedure empties the Clipboard before copying.

This method should be called only when document is formatted.

This method calls RichViewEdit<sup>(60)</sup>.CutDef.

**See also:**

- OnCopy<sup>(106)</sup>

#### 1.3.1.2.12 TSRichViewEdit.DeletePage

Deletes content of the **PageNo**-th page, as an editing operation.

**procedure** DeletePage(PageNo : Integer);

It's not guaranteed that after calling this method other pages will not be changed. It's not even guaranteed that page count will be decreased exactly by 1.

#### 1.3.1.2.13 TSRichViewEdit.Destroy

Disposes of the component.

**destructor** Destroy; **override**;

Do not call Destroy directly. Call Free instead. Free verifies that the component is not nil, and only then calls Destroy.

Never explicitly free a component in one of its own event handlers, nor free a component from the event handler of a component that it owns or contains.

Note: A form owns all the controls and non-visual components that are placed on it in design mode. When it is freed, all of these components are automatically freed as well.

#### 1.3.1.2.14 TSRichViewEdit.DrawMetafile

Paints the specified page in the metafile.

**procedure** DrawMetafile(PageNo : Integer; PicturePage : TMetafile;  
PageNoVisible, ClipMargins, Printing : Boolean;  
NoMetafiles: Boolean = False);

**PageNo** – index of page, from 1.

**PicturePage** – a metafile to draw. Page image will be scaled to **PicturePage.Width** x **PicturePage.Height**.

**PageNoVisible** specifies whether to print page numbers (see the properties of PageProperty<sup>(58)</sup>).

**ClipMargins** specifies whether all drawing outside the document area (i.e. drawing on margins) will be cropped.

**Printing** specifies whether this is a drawing for printing (a special method for drawing pictures is used in this case).

The parameter **NoMetafiles** is used only if **Printing = True**. If **NoMetafiles = False**, metafiles are drawn as they are. If **True**, metafiles are drawn as bitmaps. This option is useful when drawing on canvases of components (such as PDF generators) that do not handle embedded metafiles properly.

**See also:**



- DrawPage<sup>(77)</sup>
- OnDrawHyperlink<sup>(107)</sup>

#### 1.3.1.2.15 TSRichViewEdit.DrawPage

Draws the **PageNo**-th page onto the specified canvas.

```
procedure DrawPage(PageNo, PageWd, PageHt, OffX, OffY: Integer;  
    CanvasPage: TCanvas; PageNoVisible, ClipMargins, Printing: Boolean;  
    UseWordPainters: Boolean; ForMetafile: Boolean = False;  
    NoMetafiles: Boolean = False);
```

**PageNo** – index of page, from 1.

**CanvasPage** – a canvas where to draw.

Bounds(**OffX**, **OffY**, **PageWd**, **PageHt**) defines a rectangle on **CanvasPage** where the page will be drawn. Page image will be scaled to **PageWd** x **PageHt**.

**PageNoVisible** specifies whether to print page numbers (see the properties of PageProperty<sup>(58)</sup>).

**ClipMargins** specifies whether all drawing outside the document area (i.e. drawing on margins) will be cropped.

**Printing** specifies whether this is a drawing for printing (a special method for drawing pictures is used in this case).

If **UseWordPainters**=*True*, live spelling underlines will be drawn.

If **ForMetafile**=*False*, printing output may be incompatible with metafiles; the component may use font glyph indexes instead of character codes when printing text. Such metafiles will be displayed incorrectly on computers without this font or with another version of this font.

If **ForMetafile**=*True*, the component always uses character codes when printing text, to create metafile-compatible output.

The parameter **NoMetafiles** is used only if **Printing** = *True*. If **NoMetafiles** = *False*, metafiles are drawn as they are. If *True*, metafiles are drawn as bitmaps. This option is useful when drawing on canvases of components (such as PDF generators) that do not handle embedded metafiles properly.

**See also:**

- DrawMetafile<sup>(76)</sup>
- OnDrawHyperlink<sup>(107)</sup>

#### 1.3.1.2.16 TSRichViewEdit.EndComponentsUpdate

Unlocks redrawing of inserted controls.

```
procedure EndComponentsUpdate;
```

Unlocks redrawing of inserted controls, locked by BeginComponentsUpdate<sup>(72)</sup>.

Calls to BeginComponentsUpdate and EndComponentsUpdate may be nested.

### 1.3.1.2.17 TSRichViewEdit.FindNextCheckpoint

Finds the location of the next *checkpoint*.

```
function FindNextCheckpoint(var RVData : TCustomRVData;  
    var ItemNo : Integer; CheckRaiseEvent : Boolean) : Boolean;
```

Unlike TCustomRichView's methods for *checkpoints* enumeration, this method correctly processes *checkpoints* in table cells.

The search can be limited to the specified range of coordinates, see RangeSearch<sup>(58)</sup>.

#### Input parameters:

**RVData**, **ItemNo** – starting position for the search. The search starts from the next item. **RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

**CheckRaiseEvent** – if *True*, searches only for *checkpoints* with "raise event" flag.

#### Output parameters:

**RVData**, **ItemNo** – position of the found *checkpoint*. Valid only if the method returns *True*.

#### Return value:

*True* if found.

#### See also:

- FindPriorCheckpoint<sup>(80)</sup>
- NextCurCheckpoint<sup>(90)</sup>
- FindNextHyperlink<sup>(79)</sup>
- FindNextItem<sup>(79)</sup>

### 1.3.1.2.18 TSRichViewEdit.FindNextHeading

Finds the location of the next heading.

```
function FindNextHeading(var RVData: TCustomRVData;  
    var ItemNo: Integer) : Boolean;
```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>(58)</sup>.

Heading is a paragraph with OutlineLevel property > 0.

#### Input parameters:

**RVData**, **ItemNo** – starting position for the search. The search starts from the next item. **RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

#### Output parameters:

**RVData**, **ItemNo** – position of the found item. Valid only if the method returns *True*.

#### Return value:

*True* if found.

#### See also:

- FindPriorItem<sup>(81)</sup>
- NextCurlItem<sup>(91)</sup>

- FindNextCheckpoint<sup>(78)</sup>
- FindNextHyperlink<sup>(79)</sup>

### 1.3.1.2.19 TSVRichViewEdit.FindNextHyperlink

Finds the location of the next hyperlink.

```
function FindNextHyperlink(var RVData: TCustomRVData;  
    var ItemNo: Integer): Boolean;
```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>(58)</sup>.

#### Input parameters:

**RVData**, **ItemNo** – starting position for the search. The search starts from the next item. **RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

#### Output parameters:

**RVData**, **ItemNo** – position of the found hyperlink. Valid only if the method returns *True*.

#### Return value:

*True* if found.

#### See also:

- FindPriorHyperlink<sup>(81)</sup>
- NextCurHyperlink<sup>(91)</sup>
- FindNextCheckpoint<sup>(78)</sup>
- FindNextItem<sup>(79)</sup>

### 1.3.1.2.20 TSVRichViewEdit.FindNextItem

Finds the location of the next item of the specified styles (types).

```
function FindNextItem(StyleNo: array of Integer;  
    var RVData: TCustomRVData; var ItemNo: Integer) : Boolean;
```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>(58)</sup>.

#### Input parameters:

**StyleNo** – a list of styles (types) for searching.

**RVData**, **ItemNo** – starting position for the search. The search starts from the next item. **RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

#### Output parameters:

**RVData**, **ItemNo** – position of the found item. Valid only if the method returns *True*.

#### Return value:

*True* if found.

#### Example:

```
var  
    RVData: TCustomRVData;  
    ItemNo: Integer;
```

```

...
// searching for the first image in the document
RVData := SRichViewEdit1.RichViewEdit60.RVData;
ItemNo := -1;
if SRichViewEdit1.FindNextItem(
    [rvsPicture, rvsHotPicture, rvsBullet, rvsHotspot],
    RVData, ItemNo) then
...

```

#### See also:

- FindPriorItem<sup>81</sup>
- NextCurItem<sup>91</sup>
- FindNextCheckpoint<sup>78</sup>
- FindNextHyperlink<sup>79</sup>

#### 1.3.1.2.21 TSRichViewEdit.FindPriorCheckpoint

Finds the location of the previous *checkpoint*.

```

function FindNextCheckpoint(var RVData : TCustomRVData;
    var ItemNo : Integer; CheckRaiseEvent : Boolean) : Boolean;

```

Unlike TCustomRichView's methods for *checkpoints* enumeration, this method correctly processes *checkpoints* in table cells.

The search can be limited to the specified range of coordinates, see RangeSearch<sup>58</sup>.

#### Input parameters:

**RVData**, **ItemNo** – starting position for the search. The search starts from the previous item.

**RVData** is a document (RichViewEdit<sup>60</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

**CheckRaiseEvent** – if *True*, searches only for *checkpoints* with "raise event" flag.

#### Output parameters:

**RVData**, **ItemNo** – position of the found *checkpoint*. Valid only if the method returns *True*.

#### Return value:

*True* if found.

#### See also:

- FindNextCheckpoint<sup>78</sup>
- PriorCurCheckpoint<sup>94</sup>
- FindPriorHyperlink<sup>81</sup>
- FindPriorItem<sup>81</sup>

#### 1.3.1.2.22 TSRichViewEdit.FindPriorHeading

Finds the location of the next heading.

```

function FindPriorHeading(var RVData: TCustomRVData;
    var ItemNo: Integer): Boolean;

```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>58</sup>.

Heading is a paragraph with OutlineLevel property > 0.

**Input parameters:**

**RVData, ItemNo** – starting position for the search. The search starts from the previous item.

**RVData** is a document (RichViewEdit<sup>60</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

**Output parameters:**

**RVData, ItemNo** – position of the found item. Valid only if the method returns *True*.

**Return value:**

*True* if found.

**See also:**

- FindNextItem<sup>79</sup>
- PriorCurlItem<sup>95</sup>
- FindPriorCheckpoint<sup>80</sup>
- FindPriorHyperlink<sup>81</sup>

### 1.3.1.2.23 TSRichViewEdit.FindPriorHyperlink

Finds the location of the previous hyperlink.

```
function FindPriorHyperlink(var RVData: TCustomRVData;  
    var ItemNo: Integer): Boolean;
```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>58</sup>.

**Input parameters:**

**RVData, ItemNo** – starting position for the search. The search starts from the previous item.

**RVData** is a document (RichViewEdit<sup>60</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

**Output parameters:**

**RVData, ItemNo** – position of the found hyperlink. Valid only if the method returns *True*.

**Return value:**

*True* if found.

**See also:**

- FindNextHyperlink<sup>79</sup>
- PriorCurHyperlink<sup>94</sup>
- FindPriorCheckpoint<sup>80</sup>
- FindPriorItem<sup>81</sup>

### 1.3.1.2.24 TSRichViewEdit.FindPriorItem

Finds the location of the next item of the specified styles (types).

```
function FindPriorItem(StyleNo: array of Integer;  
    var RVData: TCustomRVData; var ItemNo: Integer) : Boolean;
```

The search can be limited to the specified range of coordinates, see RangeSearch<sup>58</sup>.

**Input parameters:**

**StyleNo** – a list of styles (types) for searching.

**RVData, ItemNo** – starting position for the search. The search starts from the previous item.

**RVData** is a document (RichViewEdit<sup>60</sup>.RVData, table cell, or cell inplace editor's RVData; if the cell is edited, it must be an inplace editor's RVData), **ItemNo** is an index of item in this document.

**Output parameters:**

**RVData, ItemNo** – position of the found item. Valid only if the method returns *True*.

**Return value:**

*True* if found.

**Example:**

**var**

```
RVData: TCustomRVData;
ItemNo: Integer;
...
// searching for the last image in the document
RVData := SRichViewEdit1.RichViewEdit60.RVData;
ItemNo := SRichViewEdit1.RichViewEdit60.ItemCount;
if SRichViewEdit1.FindPriorItem(
    [rvsPicture, rvsHotPicture, rvsBullet, rvsHotspot],
    RVData, ItemNo) then
    ...
```

**See also:**

- FindNextItem<sup>79</sup>
- PriorCurItem<sup>95</sup>
- FindPriorCheckpoint<sup>80</sup>
- FindPriorHyperlink<sup>81</sup>

### 1.3.1.2.25 TSRichViewEdit.FirstCurPage

Moves the caret to the beginning of the first page.

**procedure** FirstCurPage(SmoothScroll: Boolean = False);

The same affect as CurrentPage<sup>51</sup> := 1, but may apply a smooth scrolling effect.

**See also:**

- PriorCurPage<sup>95</sup>, NextCurPage<sup>91</sup>, LastCurPage<sup>90</sup>

### 1.3.1.2.26 TSRichViewEdit.Format, Reformat

The methods format the document completely. Only formatted documents can be displayed or edited. In addition, this methods copy properties (document size, margins, etc.) from RichViewEdit<sup>60</sup>.DocParameters to PageProperty<sup>58</sup>.

**procedure** Format;

**procedure** Reformat;

The difference between these methods is the following. **Format** does not require document to be formatted before its call, and it moves the caret to the beginning. **Reformat** can be called only if the document is already formatted, and it keeps the selection and the caret position.

**Format/Reformat** do the following work:

- loads header and footer from Subdocuments<sup>66</sup> to RVHeader<sup>64</sup> and RVFooter<sup>62</sup>
- SetRVMargins<sup>98</sup>(*True/False*).
- paginates the document

Call **Format** after loading files in RichViewEdit<sup>60</sup>.

Call **Reformat** after calling SetIntPropertyEd<sup>97</sup>(..., False)/SetFloatPropertyEd<sup>97</sup>(...False).

**See also:**

- Document parts in TSRichViewEdit<sup>37</sup>

#### 1.3.1.2.27 TSRichViewEdit.GetCaretPosInUnits

Returns the caret coordinates relative to the top left corner of the page (CurrentPage<sup>51</sup>).

```
function GetCaretPosInUnits(Units: TRVUnits;
  var CaretPosX, CaretPosY: TRVLength) : Boolean;
```

**Units** define the measurement units for the output coordinates. Note: "pixels" unit assumes 96 dpi (used internally in ScaleRichView).

The returned values are counted in zoom = 100%.

The coordinates are returned in **CaretPosX**, **CaretPosY**.

**Return value:**

*True* if the caret position is successfully returned.

**See also:**

- CaretPos<sup>49</sup>
- OnCaretMove<sup>104</sup>

#### 1.3.1.2.28 TSRichViewEdit.GetCurrentLineCol

Returns a line and column at the caret position in ActiveEditor<sup>47</sup>.

```
procedure GetCurrentLineCol(var Line, Column: Integer);
```

Number of the first line on the page is 1. Number of the leftmost caret position in the line is 1.

If the caret is in a table cell, the method returns values for this table.

**See also:**

- LineNumberProperty<sup>54</sup>

#### 1.3.1.2.29 TSRichViewEdit.GetFirstItemVisibleRV

Returns the first visible item.

```
procedure GetFirstItemVisibleRV(var RVData: TCustomRVFormattedData;
  var ItemNo, OffsetInItem: Integer);
```

This function returns the position where the caret would be placed if the user clicked at the top left corner of `TSRichViewEdit` control's window.

**RVData** is a document (`RichViewEdit`<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document, **OffsetInItem** is a position inside this item.

**See also:**

- `GetLastItemVisibleRV`<sup>(86)</sup>

#### 1.3.1.2.30 `TSRichViewEdit.GetFirstVisiblePage`

Returns the index of the first page visible on the screen (at least partially), from 1

```
function GetFirstVisiblePage: Integer;
```

**See also:**

- `GetLastVisiblePage`<sup>(87)</sup>

#### 1.3.1.2.31 `TSRichViewEdit.GetItemAt`

Returns the `ActiveEditor`<sup>(47)</sup>'s item at the specified coordinates.

```
function GetItemAt(X, Y : Integer; var RVData: TCustomRVFormattedData;
var ItemNo, OffsetInItem: Integer; Strict: Boolean): Boolean;
```

**Input parameters:**

**X, Y** – coordinates relative to the client area of `TSRichViewEdit` control.

If **Strict**=*True*, the method returns the item only if it is located exactly at the specified position (if it exists). If *False*, the method returns the closest item for any position above a page.

**Output parameters:**

Output parameters are valid only if this method returns *True*.

**RVData** – object containing this item (`ActiveEditor`<sup>(47)</sup>.RVData or table cell or RVData of cell inplace editor)

**ItemNo** – index of item (in the items list of **RVData**), or -1 if no item was found.

**OffsetInItem** – position of this item:

- for non-text: 0 (closer to the beginning of the item) or 1 (to the end);
- for text: position is before the **OffsetInItem**-th character of text; characters are indexed from 1, the last position is `Length(text)+1`.

**Return value:**

"item found?"

#### 1.3.1.2.32 `TSRichViewEdit.GetItemBounds, GetItemBounds100`

The methods return the coordinates and size of the specified item or its part.

```
function GetItemBounds(RVData: TCustomRVData; ItemNo: Integer;
Part: Integer = -1; AllowFloating: Boolean = True) : TRect;
```

**RVData** is a document (`RichViewEdit`<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document.



If **Part**=-1, the result is returned for the whole item. Otherwise, it is returned for the specified item part, counted from 0. A text item may have more than one part if drawn on several lines. To enumerate all parts, start calling this method from **Part**=0 and increase this value until the method returns Rect(-1, -1, -1, -1). You can use the function RichViewEdit.RVData.GetItemPart to get the item part for the position in the document.

The returned coordinates are relative to the top left corner of the item's (or part's) page (see GetItemPages<sup>(86)</sup>). If the item spans across several pages (it is possible only if **Part**=-1), the top left corner is relative to the first item page, the bottom left corner is relative to the last item page.

Coordinates are scaled (see ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup>).

To get the client coordinates of the item, add top and left coordinates returned by GetPageClientRect<sup>(87)</sup>.

**AllowFloating** defines which coordinates are returned for left- and right-aligned items (for all other items, this parameter is ignored). If *True*, the method returns coordinates of the object (such as an image or an control). If *False*, the method returns coordinates of a placeholder, i.e. coordinates of the point of insertion of this object in a line.

```
function GetItemBounds100 (RVData: TCustomRVFormattedData;
  ItemNo: Integer; var ARect: TRect; var PageNoB, PageNoE: Integer;
  Part: Integer = -1; AllowFloating: Boolean = True) : Boolean;
```

This method is similar, but:

- the resulting coordinates are returned in **ARect** parameter; they are is valid only if this method returns *True*;
- the coordinates are not scaled (they correspond to zooming = 100%)
- this method returns *False* if an incorrect part index is specified; all output parameters are valid only if this method returns *True*;
- this method returns the fist and the last item (or part) pages in **PageNoB** and **PageNoE** parameters, so it is not necessary to call GetItemPages<sup>(86)</sup>;

**See also:**

- GetItemCoords100<sup>(85)</sup>

### 1.3.1.2.33 TSVRichViewEdit.GetItemCoords100

The methods return the coordinates of the specified item or its part.

```
function GetItemCoords100 (RVData: TCustomRVFormattedData;
  ItemNo: Integer; var ALeft, ATop: Integer;
  var PageNo: Integer; Part : Integer = -1;
  AllowFloating: Boolean = True): Boolean;
```

#### Input parameters

**RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document.

If **Part**=-1, the result is returned for the whole item. Otherwise, it is returned for the specified item part, counted from 0. A text item may have more than one part if drawn on several lines. To enumerate all parts, start calling this method from **Part**=0 and increase this value until the method

returns *False*. You can use the function `RichViewEdit.RVData.GetItemPart` to get the item part for the position in the document.

The returned coordinates are relative to the top left corner of the item's (or part's) page (see `GetItemPages`<sup>(86)</sup>). If the item spans across several pages (it is possible only if **Part**=-1), the top left corner is relative to the first item page, the bottom left corner is relative to the last item page.

Coordinates are not scaled (they correspond to zooming = 100%).

**AllowFloating** defines which coordinates are returned for left- and right-aligned items (for all other items, this parameter is ignored). If *True*, the method returns coordinates of an object (such as an image or an control). If *False*, the method returns coordinates of a placeholder, i.e. coordinates of the point of insertion of this object in a line.

#### See also:

- `GetItemBounds`, `GetItemBounds100`<sup>(84)</sup>

#### 1.3.1.2.34 TSRichViewEdit.GetItemPages

Returns pages containing the specified item.

```
function GetItemPages(RVData: TCustomRVFormattedData; ItemNo: Integer;
  var FirstPageNo, LastPageNo: Integer; Part : Integer = -1): Boolean;
```

##### Input parameters:

**RVData** is a document (`RichViewEdit`<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document.

If **Part**=-1, the result is returned for the whole item. Otherwise, it is returned for the specified item part, counted from 0. A text item may have more than one part if drawn on several lines. To enumerate all parts, start calling this method from **Part**=0 and increase this value until the method returns *False*. You can use the function `RichViewEdit.RVData.GetItemPart` to get the item part for the position in the document.

##### Output parameters:

**FirstPageNo** – the first page where this item (or its part) is located, from 1.

**LastPageNo** – the last page where this item (or its part) is located, from 1.

##### Return value:

True if the specified item part exists.

#### 1.3.1.2.35 TSRichViewEdit.GetLastItemVisibleRV

Returns the last visible item.

```
procedure GetFirstItemVisibleRV(var RVData: TCustomRVFormattedData;
  var ItemNo, OffsetInItem: Integer);
```

This function returns the position where the caret would be placed if the user clicked at the bottom left corner of `TSRichViewEdit` control's window.

**RVData** is a document (`RichViewEdit`<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document, **OffsetInItem** is a position inside this item.

**See also:**

- `GetFirstItemVisibleRV` <sup>(83)</sup>

**1.3.1.2.36 TSVRichViewEdit.GetLastVisiblePage**

Returns the index of the last page visible on the screen (at least partially), from 1

```
function GetLastVisiblePage: Integer;
```

**See also:**

- `GetFirstVisiblePage` <sup>(84)</sup>

**1.3.1.2.37 TSVRichViewEdit.GetPageAt**

Returns the page at the specified coordinates.

```
function GetPageAt(X, Y: Integer; var PageNo: Integer) : Boolean;
```

**Input parameters:**

**X, Y** – client coordinates (relative to the top left corner of TSVRichViewEdit component's window)

**Output parameter:**

**PageNo** – index of page, from 1. Valid only if the method returns *True*.

**Return value:**

*True* if the position is inside a page.

**1.3.1.2.38 TSVRichViewEdit.GetPageClientRect**

Returns client coordinates of the specified page.

```
function GetPageClientRect(PageNo: Integer): TRect;
```

**NoPage** is the index of page (from 1).

**Return value:**

page area (TRect) relative to the top left corner of the component's client area.

**1.3.1.2.39 TSVRichViewEdit.GetPageLastItemNo**

Returns the index of the last item at the end of the **PageNo**-th page.

```
procedure GetPageLastItemNo(PageNo: Integer; var ItemNo, Offs: Integer);
```

**Input parameter**

**PageNo** is a page index, from 1.

**Output parameters**

**ItemNo** – index of the item ending this page. This value is in range 0..RichViewEdit<sup>(60)</sup>.ItemCount-1.

**Offs** – defines the position in the **ItemNo**-th item, where the last line on the page ends. For all items except for tables, this is an offset in item:

- for text items, this is the index of the character ending the page (from 1)+1,
- for non-text items (except for tables), this parameter equals to 1;
- for tables, this is the index of the last row on this page (from 0).

**See also:**

- [GetPageStartItemNo](#)<sup>(88)</sup>

**1.3.1.2.40 TSRichViewEdit.GetPageNo**

Returns the index of page for the specified position in the document.

```
function GetPageNo (RVData: TCustomRVFormattedData;  
    ItemNo, Offs : Integer): Integer;
```

**Parameters:**

The parameters define a position in the source document,

**RVData** is a document/subdocument. It can be either RVData of the source TRichView control, or table cell, or RVData of a cell inplace editor.

**ItemNo** is the index of item in this **RVData**.

**OffsetInItem** is the position inside this item. For non-text items, 0 is a position before the item, 1 is a position after the item. For text items, offsets are counted from 1 (1 - before the first character, 2 - before the second character, ... Length+1 - after the last character).

**Return value:**

Index of the page for this position, from 1.

**1.3.1.2.41 TSRichViewEdit.GetPageStartItemNo**

Returns the index of the first item at the beginning of the **PageNo**-th page.

```
procedure GetPageStartItemNo (PageNo: Integer; var ItemNo, Offs: Integer);
```

**Input parameter**

**PageNo** is a page index, from 1.

**Output parameters**

**ItemNo** – index of the item starting this page. This value is in range 0..[RichViewEdit](#)<sup>(60)</sup>.ItemCount-1.

**Offs** – defines the position in the **ItemNo**-th item, where the first line on the page starts. For all items except for tables, this is an offset in item:

- for text items, this is the index of the character starting the page (from 1);
- for non-text items (except for tables), this parameter equals to 0;
- for tables, this is the index of the first row on this page (from 0) (see [GetPageStartTableItem](#)<sup>(88)</sup> and [GetPageStartTableRow](#)<sup>(89)</sup>).

**See also:**

- [GetPageLastItemNo](#)<sup>(87)</sup>

**1.3.1.2.42 TSRichViewEdit.GetPageStartTableItem**

Returns the index of table item (TRVTableItemInfo) at the beginning of the **PageNo**-th page.

```
function GetPageStartTableItem (PageNo: Integer) : Integer;
```

**PageNo** is a page index, from 1.

**Return value:**

item index of table in RichViewEdit<sup>(60)</sup>, or -1 if the page is not started from a table.

**See also:**

- GetPageStartTableRow<sup>(89)</sup>

#### 1.3.1.2.43 TSRichViewEdit.GetPageStartTableRow

Returns the index of table row at the beginning of the **PageNo**-th page.

**function** GetPageStartTableRow(PageNo : Integer) : Integer;

**PageNo** is a page index, from 1.

**Return value:**

index of row in the table, or -1 if the page is not started from a table.

The table is TRVTableItemInfo(RichViewEdit<sup>(60)</sup>.GetItem(GetPageStartTableItem<sup>(88)</sup>)), if GetPageStartTableItem<sup>(88)</sup> >= 0.

#### 1.3.1.2.44 TSRichViewEdit.GetTableIconItem

The function returns the table for which the "icon" (rectangle in the top left corner) is displayed.

**function** GetTableIconItem: TRVTableItemInfo;

This method can be used in OnTableIconClick<sup>(121)</sup> event.

If a table icon is not displayed, the function returns *nil*.

Table icon is displayed when the user moves the mouse pointer above the table, if ViewProperty<sup>(69)</sup>.UseTableIcons<sup>(164)</sup> = *True*.

#### 1.3.1.2.45 TSRichViewEdit.GetTableIconRVData

The function returns a document containing the table for which the "icon" (rectangle in the top left corner) is displayed.

**function** GetTableIconRVData: TCustomRVFormattedData;

The returned value may be RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData.

This method can be used in OnTableIconClick<sup>(121)</sup> event.

If a table icon is not displayed, the function returns *nil*.

Table icon is displayed when the user moves the mouse pointer above the table, if ViewProperty<sup>(69)</sup>.UseTableIcons<sup>(164)</sup> = *True*.

#### 1.3.1.2.46 TSRichViewEdit.InsertPageBreak

Inserts a page break in the position of the caret.

**procedure** InsertPageBreak;

The same as RichViewEdit<sup>(89)</sup>.InsertPageBreak, but keeps the selection.

### 1.3.1.2.47 TSVRichViewEdit.InZoomRect

Zoom the rectangle in by the specified **Ratio**.

**function** InZoomRect(r: TRect; Ratio: Single): TRect;

This method multiplies all coordinates in **r** by **Ratio** and returns the modified rectangle.

This method may be useful for calculating coordinates in TSVRichViewEdit component.

**See also:**

- OutZoomRect<sup>(92)</sup>
- ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup>

### 1.3.1.2.48 TSVRichViewEdit.LastCurPage

Moves the caret to the beginning of the last page.

**procedure** LastCurPage(SmoothScroll: Boolean = False);

The same affect as CurrentPage<sup>(51)</sup> := PageCount<sup>(57)</sup>, but may apply a smooth scrolling effect.

**See also:**

- FirstCurPage<sup>(82)</sup>, PriorCurPage<sup>(95)</sup>, NextCurPage<sup>(91)</sup>

### 1.3.1.2.49 TSVRichViewEdit.LoadRVF

Load data in RVF (RichView Format) from the file **FileName**

**function** LoadRVF(FileName: TRVUnicodeString): Boolean;

This method calls RichViewEdit.LoadRVF, then formats a header and a footer and calls SetRVMargins<sup>(98)</sup>.

**Return value:**

*True* if loading was successful

### 1.3.1.2.50 TSVRichViewEdit.LoadRVFFromStream

Load data in RVF (RichView Format) from **Stream**.

**function** LoadRVFFromStream(Stream: TStream): Boolean;

This method calls RichViewEdit.LoadRVFFromStream, then formats a header and a footer and calls SetRVMargins<sup>(98)</sup>.

**Return value:**

*True* if loading was successful

### 1.3.1.2.51 TSVRichViewEdit.NextCurCheckpoint

Selects the next item having a *checkpoint*.

**function** NextCurCheckpoint: Boolean;

The search is started from the next item after the caret position (RichViewEdit<sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit<sup>(60)</sup>.TopLevelEditor.CurItemNo).

If a *checkpoint* is found, the method selects its item and returns *True*.

**See also**

- PriorCurCheckpoint <sup>(94)</sup>

#### 1.3.1.2.52 TSRichViewEdit.NextCurHeading

Moves the caret to the beginning of the next heading.

**function** NextCurHeading: Boolean;

The search is started from the next item after the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If an item is found, the method moves the caret and returns *True*.

Heading is a paragraph with OutlineLevel property > 0.

**See also:**

- PriorCurHeading <sup>(94)</sup>

#### 1.3.1.2.53 TSRichViewEdit.NextCurHyperlink

Selects the next hyperlink.

**function** NextCurHyperlink : Boolean;

The search is started from the next item after the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If a hyperlink is found, the method selects it and returns *True*.

**See also**

- PriorCurHyperlink <sup>(94)</sup>

#### 1.3.1.2.54 TSRichViewEdit.NextCurItem

Selects the next item of one of the specified styles (types).

**function** NextCurItem(StyleNo: **array of** Integer) : Boolean;

**StyleNo** – a list of styles (types) for searching. If this array is empty, this method searches for any item.

The search is started from the next item after the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If an item is found, the method selects it and returns *True*.

This method can be used for the command "Find Picture" and "Find Table" in the horizontal toolbar <sup>(55)</sup>.

**See also:**

- PriorCurItem <sup>(95)</sup>

#### 1.3.1.2.55 TSRichViewEdit.NextCurPage

Moves the caret to the beginning of the next page.

**procedure** NextCurPage(SmoothScroll: Boolean = False);

The same affect as

**if** CurrentPage <sup>(51)</sup> < PageCount <sup>(57)</sup> **then**

```
CurrentPage(51) := CurrentPage(51) + 1;
```

The method may apply a smooth scrolling effect.

**See also:**

- FirstCurPage<sup>(82)</sup>, PriorCurPage<sup>(95)</sup>, LastCurPage<sup>(90)</sup>
- NextCurHyperlink<sup>(91)</sup>
- NextCurCheckpoint<sup>(90)</sup>
- NextCurItem<sup>(91)</sup>

### 1.3.1.2.56 TSRichViewEdit.OutZoomRect

Zoom the rectangle out by the specified **Ratio**.

```
function OutZoomRect(r: TRect; Ratio: Single): TRect;
```

This method divides all coordinates in **r** by **Ratio** and returns the modified rectangle.

This method may be useful for calculating coordinates in TSRichViewEdit component.

**See also:**

- OutZoomRect<sup>(92)</sup>
- ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup>

### 1.3.1.2.57 TSRichViewEdit.Paste

Pastes data from the Clipboard in the caret position.

```
procedure Paste;
```

This method calls RichViewEdit<sup>(60)</sup>.Paste.

**See also:**

- OnPaste<sup>(116)</sup>

### 1.3.1.2.58 TSRichViewEdit.PrintAll

Prints the document.

```
procedure PrintAll(MetafileCompatibility: Boolean = False;  
NoMetafiles: Boolean = False);
```

This method takes MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup> properties into account.

The printer is defined in Printer.PrinterIndex.

If **MetafileCompatibility=False**, printing output may be incompatible with metafiles; the component may use font glyph indexes instead of character codes when printing text. Such metafiles will be displayed incorrectly on computers without this font or with another version of this font.

If **MetafileCompatibility=True**, the component always uses character codes when printing text, to create metafile-compatible output.

If **NoMetafiles = False**, metafiles are drawn as they are. If **True**, metafiles are drawn as bitmaps. This option is useful when drawing on canvases of components (such as PDF generators) that do not handle embedded metafiles properly.

**See also:**

- PrintCurrent<sup>(93)</sup>



- [PrintRange](#)<sup>(93)</sup>
- [TSRVPrint](#)<sup>(204)</sup> component
- [TSRVPrint.MetafileCompatibility](#)<sup>(211)</sup>
- [TRVAControlPanel.MetafileCompatibility](#) (in the RichViewActions help file)

### 1.3.1.2.59 TSRichViewEdit.PrintCurrent

Prints the current page.

```
procedure PrintCurrent(MetafileCompatibility: Boolean=False;
  NoMetafiles: Boolean = False);
```

The printer is defined in Printer.PrinterIndex.

If **MetafileCompatibility**=*False*, printing output may be incompatible with metafiles; the component may use font glyph indexes instead of character codes when printing text. Such metafiles will be displayed incorrectly on computers without this font or with another version of this font.

If **MetafileCompatibility**=*True*, the component always uses character codes when printing text, to create metafile-compatible output.

If **NoMetafiles** = *False*, metafiles are drawn as they are. If *True*, metafiles are drawn as bitmaps. This option is useful when drawing on canvases of components (such as PDF generators) that do not handle embedded metafiles properly.

**See also:**

- [CurrentPage](#)<sup>(51)</sup>
- [PrintAll](#)<sup>(92)</sup>
- [PrintRange](#)<sup>(93)</sup>
- [TSRVPrint](#)<sup>(204)</sup> component
- [TSRVPrint.MetafileCompatibility](#)<sup>(211)</sup>
- [TRVAControlPanel.MetafileCompatibility](#) (in the RichViewActions help file)

### 1.3.1.2.60 TSRichViewEdit.PrintRange

Prints all pages from **StartPageNo** to **EndPageNo**.

```
procedure PrintRange(StartPageNo, EndPageNo: Integer;
  MetafileCompatibility: Boolean=False;
  NoMetafiles: Boolean = False);
```

This method takes [MinPrintedItemNo](#)<sup>(56)</sup> and [MaxPrintedItemNo](#)<sup>(55)</sup> properties into account.

The printer is defined in Printer.PrinterIndex.

If **MetafileCompatibility**=*False*, printing output may be incompatible with metafiles; the component may use font glyph indexes instead of character codes when printing text. Such metafiles will be displayed incorrectly on computers without this font or with another version of this font.

If **MetafileCompatibility**=*True*, the component always uses character codes when printing text, to create metafile-compatible output.

If **NoMetafiles** = *False*, metafiles are drawn as they are. If *True*, metafiles are drawn as bitmaps. This option is useful when drawing on canvases of components (such as PDF generators) that do not handle embedded metafiles properly.

**See also:**

- PrintAll <sup>(92)</sup>
- PrintCurrent <sup>(93)</sup>
- TSRVPrint <sup>(204)</sup> component
- TSRVPrint.MetafileCompatibility <sup>(211)</sup>
- TRVAControlPanel.MetafileCompatibility (in the RichViewActions help file)

**1.3.1.2.61 TSRichViewEdit.PriorCurCheckpoint**

Selects the previous item having a *checkpoint*.

**function** PriorCurCheckpoint: Boolean;

The search is started from the item before the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If a *checkpoint* is found, the method selects its item and returns *True*.

**See also**

- NextCurCheckpoint <sup>(90)</sup>

**1.3.1.2.62 TSRichViewEdit.PriorCurHeading**

Moves the caret to the beginning of the previous heading.

**function** PriorCurHeading: Boolean;

The search is started from the item before the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If an item is found, the method moves the caret and returns *True*.

Heading is a paragraph with OutlineLevel property > 0.

**See also:**

- NextCurHeading <sup>(91)</sup>

**1.3.1.2.63 TSRichViewEdit.PriorCurHyperlink**

Selects the next hyperlink.

**function** PriorCurHyperlink : Boolean;

The search is started from the item before the caret position (RichViewEdit <sup>(60)</sup>.TopLevelEditor.RVData, RichViewEdit <sup>(60)</sup>.TopLevelEditor.CurItemNo).

If a hyperlink is found, the method selects it and returns *True*.

**See also**

- NextCurHyperlink <sup>(91)</sup>

#### 1.3.1.2.64 TSRichViewEdit.PriorCurItem

Selects the previous item of one of the specified styles (types).

**function** NextCurItem(StyleNo: **array of** Integer) : Boolean;

**StyleNo** – a list of styles (types) for searching. If this array is empty, this method searches for any item.

The search is started from the item before the caret position (RichViewEdit<sup>60</sup>.TopLevelEditor.RVData, RichViewEdit<sup>60</sup>.TopLevelEditor.CurItemNo).

If an item is found, the method selects it and returns *True*.

This method can be used for the command "Find Picture" and "Find Table" in the horizontal toolbar<sup>55</sup>.

**See also:**

- NextCurItem<sup>91</sup>

#### 1.3.1.2.65 TSRichViewEdit.PriorCurPage

Moves the caret to the beginning of the previous page.

**procedure** PriorCurPage(SmoothScroll: Boolean = False);

The same affect as

```
if CurrentPage51 > 1 then
  CurrentPage51 := CurrentPage51 - 1;
```

The method may apply a smooth scrolling effect.

**See also:**

- FirstCurPage<sup>82</sup>, NextCurPage<sup>91</sup>, LastCurPage<sup>90</sup>
- PriorCurHyperlink<sup>94</sup>
- PriorCurCheckpoint<sup>94</sup>
- PriorCurItem<sup>95</sup>

#### 1.3.1.2.66 TSRichViewEdit.ProcessControls

Calls OnCheckControl<sup>105</sup> event for all controls inserted in the document.

**procedure** ProcessControls;

#### 1.3.1.2.67 TSRichViewEdit.ReturnToNote

Ends editing a note's Document and moves the caret to the location of this note in the main document.

**function** ReturnToNote(SmoothScroll: Boolean = False): Boolean;

This method does nothing if a note is not being edited (i.e. when ActiveEditor<sup>47</sup> <> RVNote<sup>64</sup>).

When the caret is moved, the main document becomes active (ActiveEditor<sup>47</sup> becomes equal to RichViewEdit<sup>60</sup>).

If **SmoothScroll** = *True*, a smooth scrolling effect is applied.

**Return value:**

*True* if the caret was moved successfully

**See also:**

- StartEditNote <sup>99</sup>
- Footnotes and endnotes <sup>38</sup>

### 1.3.1.2.68 TSRichViewEdit.ScrollToCaret, ScrollCaretToCenter

The methods scroll to make the caret visible.

```
function ScrollToCaret(SmoothScroll: Boolean = False): Boolean;
function ScrollCaretToCenter(SmoothScroll: Boolean = False): Boolean;
```

**ScrollToCaret** does nothing if the caret is already completely visible. **ScrollCaretToCenter** always scrolls.

**ScrollToCaret** scrolls to show the top of the caret in the middle of the editor. **ScrollCaretToCenter** scrolls to show the middle of the caret in the middle of the editor.

If **SmoothScroll** = *True*, a smooth scrolling effect is applied.

**See also**

- ScrollToItem <sup>96</sup>

### 1.3.1.2.69 TSRichViewEdit.ScrollToItem

Scrolls to make the specified item visible. Optionally moves the caret to this item.

```
function ScrollToItem(RVData: TCustomRVFormattedData; ItemNo: Integer;
  SetCaret: Boolean = True; ToItemEnd: Boolean = False;
  ToItemCenter: Boolean = False; SmoothScroll: Boolean = False): Boolean;
```

**The item location**

**RVData** is a document (RichViewEdit <sup>60</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document.

**Moving the caret**

If **SetCaret**=*True*, the method not only scrolls the window, but also moves the caret to the specified item. The caret is moved either to the beginning of this item (if **ToItemEnd**=*False*) or to the end of it (if **ToItemEnd**=*True*).

**Scrolling**

If **ToItemCenter**=*False*, the method scrolls so that the item top is shown at the top of the editor window,

If **ToItemCenter**=*True*, the method scrolls so that the item middle is shown at the middle of the editor window.

If **SmoothScroll** = *True*, a smooth scrolling effect is applied.

**See also:**

- ScrollToCaret, ScrollCaretToCenter <sup>96</sup>

### 1.3.1.2.70 TSVRichViewEdit.SelectAll

Selects the whole document in ActiveEditor<sup>(47)</sup>.

```
procedure SelectAll;
```

### 1.3.1.2.71 TSVRichViewEdit.Set\*PropertyEd

Editing-style methods for changing properties of the editor.

```
procedure SetIntPropertyEd(Prop: TSVRVIntProperty(277); Value: Integer;  
    AutoReformat: Boolean);
```

```
procedure SetFloatPropertyEd(Prop: TSVRVFloatProperty(276); Value: TRVLength;  
    AutoReformat: Boolean);
```

#### Parameters:

**Prop** identifies the property to change.

**Value** – new property value. If the property is a property of enumerated type, use Ord(). If the property is a boolean property, any non-zero value means *True*.

If **AutoReformat=True**, the editor is reformatted (if necessary), and RichViewEdit<sup>(60)</sup>.Change is called. Normally, this method is called with **AutoReformat=True** if you change a single property. If you call these methods multiple times for changing multiple properties, reformat<sup>(82)</sup> (if necessary) and call RichViewEdit<sup>(60)</sup>.Change yourself.

### 1.3.1.2.72 TSVRichViewEdit.SetMargin

Assigns margins.

```
procedure SetMargin(MarginLeft, MarginTop, MarginRight,  
    MarginBottom: Integer);
```

**MarginLeft, MarginTop, MarginRight, MarginBottom** are new values of margins, measured in pixels.

Before assigning, OnMarginsChanged<sup>(111)</sup> occurs.

Only parameters <> -1 are assigned.

This method changes:

- PageProperty<sup>(58)</sup>'s LeftMargin<sup>(138)</sup>, TopMargin<sup>(142)</sup>, RightMargin<sup>(142)</sup>, BottomMargin<sup>(136)</sup>
- LeftMarginPix<sup>(54)</sup>, TopMarginPix<sup>(68)</sup>, RightMarginPix<sup>(60)</sup>, BottomMarginPix<sup>(48)</sup>

#### See also:

- SetMarginMM<sup>(97)</sup>
- SetMarginUnit<sup>(98)</sup>
- SetRVMargins<sup>(98)</sup>

### 1.3.1.2.73 TSVRichViewEdit.SetMarginMM

Assigns margins.

```
procedure SetMarginMM(MarginLeftMM, MarginTopMM, MarginRightMM,  
    MarginBottomMM : Extended);
```

**MarginLeftMM, MarginTopMM, MarginRightMM, MarginBottomMM** are new values of margins, measured in millimeters.

Before assigning, OnMarginsChanged<sup>(111)</sup> occurs.

Only parameters <> -1 are assigned.

This method changes:

- PageProperty<sup>(58)</sup>'s LeftMargin<sup>(138)</sup>, TopMargin<sup>(142)</sup>, RightMargin<sup>(142)</sup>, BottomMargin<sup>(136)</sup>
- LeftMarginPix<sup>(54)</sup>, TopMarginPix<sup>(68)</sup>, RightMarginPix<sup>(60)</sup>, BottomMarginPix<sup>(48)</sup>

**See also:**

- SetMargin<sup>(97)</sup>
- SetMarginUnit<sup>(98)</sup>
- SetRVMargins<sup>(98)</sup>

#### 1.3.1.2.74 TSRichViewEdit.SetMarginUnit

Assigns margins.

```
procedure SetMarginUnit (MarginLeft, MarginTop, MarginRight,
    MarginBottom: TRVLength);
```

**MarginLeft, MarginTop, MarginRight, MarginBottom** are new values of margins, measured in UnitsProgram<sup>(68)</sup>.

Before assigning, OnMarginsChanged<sup>(111)</sup> occurs.

Only parameters <> -1 are assigned.

This method changes:

- PageProperty<sup>(58)</sup>'s LeftMargin<sup>(138)</sup>, TopMargin<sup>(142)</sup>, RightMargin<sup>(142)</sup>, BottomMargin<sup>(136)</sup>
- LeftMarginPix<sup>(54)</sup>, TopMarginPix<sup>(68)</sup>, RightMarginPix<sup>(60)</sup>, BottomMarginPix<sup>(48)</sup>

**See also:**

- SetMarginMM<sup>(97)</sup>
- SetMargin<sup>(97)</sup>
- SetRVMargins<sup>(98)</sup>

#### 1.3.1.2.75 TSRichViewEdit.SetRVMargins

Assigns page properties from RichViewEdit<sup>(60)</sup>.DocParameters to PageProperty<sup>(58)</sup>, formats the editor.

```
procedure SetRVMargins (CompleteFormat: Boolean = True);
```

**It's highly recommended to call call Format<sup>(82)</sup> or Reformat<sup>(82)</sup> instead of this method!**

This method does the following work:

- copies page properties from RichViewEdit<sup>(60)</sup>.DocParameters to PageProperty<sup>(58)</sup>
- resets formatting for notes and text boxes (so that they will be reformatted on the next pagination)
- formats headers and footers (if CompleteFormat=False, the caret position is kept in RVHeader<sup>(64)</sup> and RVFooter<sup>(62)</sup>)
- formats RichViewEdit<sup>(60)</sup> (if CompleteFormat=False, the caret position is kept)
- (if RVNote<sup>(64)</sup> <> nil) formats RVNote<sup>(64)</sup> (if CompleteFormat=False, the caret position is kept)

### 1.3.1.2.76 TSRichViewEdit.StartEditing

Begins editing the specified subdocument.

**type**

```
TSRVRichViewEditType = (srvrveMain, srvrveHeader, srvrveFooter);
```

**function** StartEditing(Editor: TSRVRichViewEditType): Boolean;

| Editor              | Begins editing...  |
|---------------------|--|
| <i>srvrveMain</i>   | RichViewEdit <sup>(60)</sup> (main document)                                 |
| <i>srvrveHeader</i> | RVHeader <sup>(64)</sup> (page header for the current page <sup>(51)</sup> ) |
| <i>srvrveFooter</i> | RVFooter <sup>(62)</sup> (page footer for the current page)                  |

This method changes value of ActiveEditor<sup>(47)</sup> property.

**See also:**

- StartEditNote<sup>(99)</sup>

### 1.3.1.2.77 TSRichViewEdit.StartEditNote

Begins editing the Document of the specified note or a text box item.

**function** StartEditNote(RVData: TCustomRVFormattedData;

```
ItemNo: Integer; ScrollToNote: Boolean = True;
```

```
SmoothScroll: Boolean = False): TRichViewEdit;
```

**RVData** and **ItemNo** must define the location of a note item. The following item types are supported:

- footnotes (TRVFootnoteItemInfo)
- endnotes (TRVEndnoteItemInfo)
- sidenotes (TRVSidenoteItemInfo)
- text box items (TRVTextBoxItemInfo)

This method changes value of ActiveEditor<sup>(47)</sup> property (it becomes equal to RVNote<sup>(64)</sup>).

If **ScrollToNote**=*True*, the method scrolls the control to show the edited note. If **SmoothScroll** = *True*, this scrolling is smooth.

**Return value:**

ActiveEditor<sup>(47)</sup> if succeed. *Nil* if **RVData** and **ItemNo** do not define a location of a note or a text box item.

**See also:**

- StartEditing<sup>(99)</sup>
- ReturnToNote<sup>(95)</sup>
- Footnotes and endnotes<sup>(38)</sup>

### 1.3.1.2.78 TSVRichViewEdit.UnitsPerInchH

Converts one inch to the specified units. Used for horizontal distances.

**function** UnitsPerInchH(Units: TRVUnits): TRVLength;

Since in ScaleRichView both vertical and horizontal resolution are assumed equal to 96 dpi, this function returns the same value as UnitsPerInchH<sup>(100)</sup>.

This function can be used to convert one units to another, for example:

```
// converting horizontal size from millimeters to UnitsProgram(68)
Size := SizeMM * UnitsPerInchH(UnitsProgram(68)) /
    UnitsPerInchH(rvuMillimeters)
```

### 1.3.1.2.79 TSVRichViewEdit.UnitsPerInchV

Converts one inch to the specified units. Used for vertical distances.

**function** UnitsPerInchV(Units: TRVUnits): TRVLength;

Since in ScaleRichView both vertical and horizontal resolution are assumed equal to 96 dpi, this function returns the same value as UnitsPerInchH<sup>(100)</sup>.

This function can be used to convert one units to another, for example:

```
// converting vertical size from millimeters to UnitsProgram(68)
Size := SizeMM * UnitsPerInchV(UnitsProgram(68)) /
    UnitsPerInchV(rvuMillimeters)
```

### 1.3.1.2.80 TSVRichViewEdit.UpdateBuffer, UpdateBufferAt

These methods redraw the internal buffer or its area.

**procedure** UpdateBuffer;

**procedure** UpdateBufferAt(PaintRect: TRect);

When drawing on the screen, TSVRichViewEdit control uses an internal buffer (a bitmap). When you call Invalidate or Repaint method, the control simply copies from this buffer to the screen.

These methods update this buffer (or the area PaintRect specified in the client coordinates) and the corresponding area on the screen.

Normally, the buffer is updated when document is changed, so you do not need to call these methods. However, if you want to update some drawing made in OnPaintPage<sup>(115)</sup>, use these methods.

### 1.3.1.2.81 TSVRichViewEdit.ZoomCanvas, RestoreCanvasZoom

Applies and restores zooming to the **Canvas**.

**function** ZoomCanvas(Canvas : TCanvas;  
ScaleX, ScaleY : Single) : TSVRVZoomInfo;

**procedure** RestoreCanvasZoom(Canvas : TCanvas;  
OldInfo : TSVRVZoomInfo);

**type**

TSRVZoomInfo = **record** ...



After calling `ZoomCanvas`, all subsequent drawing on **Canvas** will be scaled by **ScaleX** horizontally and by **ScaleY** vertically. Store the returned value and then call `RestoreCanvasZoom` to restore the original zooming.

These methods may be useful in `OnPaint`<sup>(114)</sup> event, if you want to draw something in the same zooming ratio as pages (use `ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup> \* 0.01 as **ScaleX** and **ScaleY** parameters).

#### 1.3.1.2.82 TSRichViewEdit.ZoomRectIn, ZoomRectOut

Returns a scaled rectangle.

```
function ZoomRectIn(r: TRect; Ratio: Single): TRect;
function ZoomRectOut(r: TRect; Ratio: Single): TRect;
```

`ZoomRectIn` returns the rectangle **r** zoomed in by **Ratio**.

`ZoomRectOut` returns the rectangle **r** zoomed in by `1/Ratio`.

#### 1.3.1.2.83 TSRichViewEdit.ZoomToFullPages

Changes zooming (`ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup>) to display pages in **ColCount** columns, if possible.

```
procedure ZoomToFullPages(ColCount: Integer);
```

The procedure changes `ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup> to display **ColCount** pages horizontally. It may not always be possible, because zooming must be in the range `ViewProperty`<sup>(69)</sup>.`ZoomMin`..`ZoomMax`<sup>(166)</sup>

### 1.3.1.3 Events

#### In TSRichViewEdit

- `OnAfterOleDrop`<sup>(104)</sup>
- `OnAssignImageFileName`<sup>(103)</sup>
- `OnBeforeOleDrop`<sup>(104)</sup>
- `OnCaretGetOut`<sup>(104)</sup>
- `OnCaretMove`<sup>(104)</sup>
- `OnCaretMoving`<sup>(104)</sup>
- `OnChange`<sup>(104)</sup>
- `OnChangeActiveEditor`<sup>(104)</sup>
- `OnChangeCurrentControl`<sup>(105)</sup>
- `OnChangeViewModeAfter`<sup>(105)</sup>
- `OnChangeViewModeBefore`<sup>(105)</sup>
- `OnChanging`<sup>(105)</sup>
- `OnCheckControl`<sup>(105)</sup>
- `OnCheckpointVisible`<sup>(106)</sup>
- `OnCheckStickingItems`<sup>(106)</sup>
- `OnClickPage`<sup>(106)</sup>
- `OnContextPopup`<sup>(106)</sup>
- `OnControlAction`<sup>(106)</sup>
- `OnCopy`<sup>(106)</sup>

- OnCurParaStyleChanged<sup>107</sup>
- OnCurrentPageChange<sup>107</sup>
- OnCurTextStyleChanged<sup>107</sup>
- OnDrawHyperlink<sup>107</sup>
- OnDropFiles<sup>108</sup>
- OnFullRedraw<sup>108</sup>
- OnGetPagePos<sup>108</sup>
- OnHMenuClickButton<sup>109</sup>
- OnHMenuEnterButton<sup>109</sup>
- OnHScrolled<sup>109</sup>
- OnHTMLSaveImage<sup>110</sup>
- OnImportPicture<sup>110</sup>
- OnItemAction<sup>110</sup>
- OnItemHint<sup>110</sup>
- OnItemResize<sup>110</sup>
- OnItemTextEdit<sup>110</sup>
- OnLoadCustomFormat<sup>111</sup>
- OnLoadDocument<sup>111</sup>
- OnJump<sup>111</sup>
- OnMarginsChanged<sup>111</sup>
- OnMessageControl<sup>112</sup>
- OnNewDocument<sup>113</sup>
- OnOleDragEnter<sup>113</sup>
- OnOleDragLeave<sup>113</sup>
- OnOleDragOver<sup>113</sup>
- OnOleDrop<sup>113</sup>
- OnPageCountChanged<sup>114</sup>
- OnPageFormatChanged<sup>114</sup>
- OnPageScrolled<sup>114</sup>
- OnPaint<sup>114</sup>
- OnPaintComponent<sup>114</sup>
- OnPaintPage<sup>115</sup>
- OnParaStyleConversion<sup>116</sup>
- OnPaste<sup>116</sup>
- OnPrinting<sup>116</sup>
- OnProgress<sup>117</sup>
- OnReadField<sup>117</sup>
- OnReadHyperlink<sup>117</sup>
- OnReadMergeField<sup>117</sup>
- OnResize<sup>117</sup>
- OnResizing<sup>117</sup>
- OnResizeDoc<sup>117</sup>
- OnRVDbClick<sup>117</sup>
- OnRVFControlNeeded<sup>118</sup>
- OnRVFImageListNeeded<sup>118</sup>
- OnRVFPictureNeeded<sup>118</sup>
- OnRVMouseDown<sup>118</sup>
- OnRVMouseMove<sup>118</sup>

- OnRVMouseUp <sup>(118)</sup>
- OnRVRightClick <sup>(119)</sup>
- OnSaveComponentToFile <sup>(119)</sup>
- OnSaveCustomFormat <sup>(119)</sup>
- OnSaveDocXExtra <sup>(119)</sup>
- OnSaveHTMLExtra <sup>(120)</sup>
- OnSaveImage2 <sup>(120)</sup>
- OnSaveItemToFile <sup>(120)</sup>
- OnSaveRTFExtra <sup>(120)</sup>
- OnSelect <sup>(120)</sup>
- OnSmartPopupClick <sup>(120)</sup>
- OnSpellingCheck <sup>(121)</sup>
- OnStyleConversion <sup>(121)</sup>
- OnTableIconClick <sup>(121)</sup>
- OnTextFound <sup>(121)</sup>
- OnURLNeeded <sup>(121)</sup>
- OnVMenuClickButton <sup>(121)</sup>
- OnVMenuEnterButton <sup>(122)</sup>
- OnVScrolled <sup>(122)</sup>
- OnWriteHyperlink <sup>(122)</sup>
- OnWriteObjectProperties <sup>(122)</sup>
- OnZoomChanged <sup>(122)</sup>
- OnZoomViewChanged <sup>(123)</sup>

### Derived from TCustomControl

- OnClick
- OnDbClick
- OnDragDrop
- OnDragOver
- OnEndDrag
- OnEnter
- OnExit
- OnGesture (D2010+)
- OnKeyDown
- OnKeyPress
- OnKeyUp
- OnMouseMove
- OnMouseWheel
- OnStartDrag

#### 1.3.1.3.1 TSVRichViewEdit.OnAssignImageFileName

Occurs when the component assigns background image file names; allows to modify file names before assigning.

**property** OnAssignImageFileName : TRVAssignImageFileNameEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- ActiveEditor<sup>47</sup>

#### 1.3.1.3.2 TSRichViewEdit.OnBeforeOleDrop, OnAfterOleDrop

OLE drag&drop events. They occur when the user completes drag&drop operation into the editor. These events occur before and after insertion.

**property** OnBeforeOleDrop: TNotifyEvent;

**property** OnAfterOleDrop: TNotifyEvent;

See the events of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.3 TSRichViewEdit.OnCaretGetOut

Occurs when user presses arrow keys at the beginning/end of document in "outside" direction.

**property** OnCaretGetOut : TRVOnCaretGetOutEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

**See also:**

- ActiveEditor<sup>47</sup>

#### 1.3.1.3.4 TSRichViewEdit.OnCaretMove, OnCaretMoving

The events occur when the caret is moved to the another position within the document (TCustomRichViewEdit(Sender)).

**property** OnCaretMove: TNotifyEvent;

**property** OnCaretMoving: TNotifyEvent;

OnCaretMoving occurs before TSRichViewEdit processes a caret movement (and scrolls to move the caret visible).

OnCaretMove occurs after that.

See "OnCaretMove" in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.5 TSRichViewEdit.OnChange

Occurs when document is changed.

**property** OnChange: TNotifyEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.6 TSRichViewEdit.OnChangeActiveEditor

Occurs when value of ActiveEditor<sup>47</sup> property is changed.

**type**

TSRVChangeActiveEditorEvent = **procedure** (Sender: TSRichViewEdit<sup>40</sup>;  
ActiveEditor: TRichViewEdit) **of object**;

**property** OnChangeActiveEditor : TSRVChangeActiveEditorEvent;

### 1.3.1.3.7 TSRichViewEdit.OnChangeCurrentControl

Occurs when value of CurControl<sup>(50)</sup> property is changed.

#### type

```
TSRVChangeCurrentControlEvent = procedure(Sender: TSRichViewEdit(40);
    AControl: TControl) of object;
```

**property** OnChangeCurrentControl : TSRVChangeCurrentControlEvent;

### 1.3.1.3.8 TSRichViewEdit.OnChangeViewModeAfter

The event occurs after changing a viewing mode (ViewProperty<sup>(69)</sup>.ViewMode<sup>(165)</sup>).

#### type

```
TSRVChangeViewModeEvent = procedure (Sender: TSRichViewEdit(40);
    Value: TSRVViewMode(154)) of object;
```

**property** OnChangeViewModeAfter: TSRVChangeViewModeEvent;

**Value** is a new value of ViewMode<sup>(165)</sup>.

#### See also:

- OnChangeViewModeBefore<sup>(105)</sup>

### 1.3.1.3.9 TSRichViewEdit.OnChangeViewModeBefore

The event occurs before changing a viewing mode (ViewProperty<sup>(69)</sup>.ViewMode<sup>(165)</sup>).

#### type

```
TSRVChangeViewModeEvent = procedure (Sender: TSRichViewEdit(40);
    Value: TSRVViewMode(154)) of object;
```

**property** OnChangeViewModeBefore: TSRVChangeViewModeEvent;

**Value** is a new value of ViewMode<sup>(165)</sup>.

#### See also:

- OnChangeViewModeAfter<sup>(105)</sup>

### 1.3.1.3.10 TSRichViewEdit.OnChanging

Occurs before editing operations, allows preventing editing.

**property** OnChanging: TRVChangingEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

### 1.3.1.3.11 TSRichViewEdit.OnCheckControl

This event is called by ProcessControls<sup>(95)</sup> method.

#### type

```
TSRVCheckControl = procedure (Sender: TSRichViewEdit;
    AControl : TControl) of object;
```

**property** OnCheckControl: TSRVCheckControl;

This event is called for each control (**AControl**) inserted in the current document.

### 1.3.1.3.12 TSRichViewEdit.OnCheckpointVisible

Occurs when *checkpoint* (with RaiseEvent flag = *True*) becomes visible as a result of vertical scrolling.

#### type

```
TSRVCheckpointVisibleEvent = procedure (Sender: TSRichViewEdit40;  
    RVData: TCustomRVData; ItemNo: Integer) of object;
```

**property** OnCheckpointVisible: TSRVCheckpointVisibleEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

This event has different parameters comparing with the event of TCustomRichView: instead of CheckpointData: TCheckpointData, there are two parameters defining the location of the *checkpoint*: **RVData** and **ItemNo**. You can get the *checkpoint* as **RVData.GetItemCheckpoint(ItemNo)**.

### 1.3.1.3.13 TSRichViewEdit.OnCheckStickingItems

Allows preventing insertion between two items, or at the beginning of the document, or at the end of the document.

**property** OnCheckStickingItems: TRVStickingItemsEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

### 1.3.1.3.14 TSRichViewEdit.OnClickPage

Occurs when the user clicked on a page.

**property** OnClickPage: TSRVMouseEvent<sup>277</sup>;

### 1.3.1.3.15 TSRichViewEdit.OnContextPopup

Occurs when the user right-clicks the control or otherwise invokes the popup menu (such as using the keyboard).

**property** OnContextPopup: TContextPopupEvent;

### 1.3.1.3.16 TSRichViewEdit.OnControlAction

Occurs when some important operation is performed on control inserted in Sender.

**property** OnControlAction: TRVControlActionEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

### 1.3.1.3.17 TSRichViewEdit.OnCopy

Occurs when copying to the Clipboard

**property** OnCopy: TNotifyEvent;

This event allows copying to the Clipboard in your own formats (occurs between Clipboard.Open and Clipboard.Close).

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- ActiveEditor<sup>(47)</sup>

#### 1.3.1.3.18 TSRichViewEdit.OnCurParaStyleChanged

Occurs when the index of current paragraph style (value of TCustomRichViewEdit(Sender).CurParaStyleNo property) is changed.

**property** OnCurParaStyleChanged: TNotifyEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.19 TSRichViewEdit.OnCurrentPageChange

Occurs when the caret is moved to another page

**property** OnCurrentPageChange: TNotifyEvent;

This event occurs after CurrentPage<sup>(51)</sup> property is changed.

**See also:**

- OnPageCountChanged<sup>(114)</sup>
- OnPageScrolled<sup>(114)</sup>

#### 1.3.1.3.20 TSRichViewEdit.OnCurTextStyleChanged

Occurs when the index of current text style (value of TCustomRichViewEdit(Sender).CurTextStyleNo property) is changed.

**property** OnCurTextStyleChanged: TNotifyEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.21 TSRichViewEdit.OnDrawHyperlink

Occurs when a hyperlink is drawn.

**type**

```
TSRVDataDrawHyperlinkEvent = procedure (Sender: TSRichViewEdit(40);
  Canvas : TCanvas; RVData: TCustomRVData; PageNo, ItemNo: Integer;
  RepaintRect, R: TRect) of object;
```

**property** OnDrawHyperlink : TSRVDataDrawHyperlinkEvent;

This event can be used for the following tasks:

- storing locations of hyperlinks when calling DrawPage<sup>(77)</sup> or DrawMetafile<sup>(76)</sup>
- custom drawing (see the demo **Demos\Basic Demos\DrawHyperlink\**)

**Parameters:**

**Canvas** – a canvas where the page is drawn.

**PageNo** – a page containing this hyperlink, from 1.

**RVData, ItemNo** – location of the hyperlink. **RVData** is a document (RichViewEdit<sup>(60)</sup>.RVData, table cell, or cell inplace editor's RVData), **ItemNo** is an index of item in this document.

**RepaintRect** – page area (the same as returned by GetPageClientRect<sup>(87)</sup>).

**R** – hyperlink area

### 1.3.1.3.22 TSVRichViewEdit.OnDropFile, OnDropFiles

The events occur when dropping files in the editor (as a result of drag&drop operation, for example from Windows Explorer)

**property** OnDropFile: TRVDropFileEvent;

**property** OnDropFiles: TRVDropFilesEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

### 1.3.1.3.23 TSVRichViewEdit.OnFullRedraw

Occurs when the component is completely repainted.

**property** OnFullRedraw: TNotifyEvent;

Mainly for internal use.

### 1.3.1.3.24 TSVRichViewEdit.OnGetPagePos

Allows to define page position, size and Z-order. Occurs if ViewProperty<sup>(69)</sup>.FreePosPage<sup>(158)</sup> = True.

**type**

```
TSRVGetPagePosEvent = procedure (Sender: TSVRichViewEdit;
  PageNo : Integer;
  StartItemNo, EndItemNo, OffsetInStartItem,
  StartTableRow, EndTableRow: Integer;
  var Position : TPoint; var ZoomPercent : Single;
  var ZOrder : Integer) of object;
```

**property** OnGetPagePos: TSRVGetPagePosEvent;

**PageNo** – page index (from 1).

**StartItemNo** – index of the first item on the page.

**EndItemNo** – index of the last item on the page.

**OffsetInStartItem** – offset in the first item (for non-text items: 0; for text items: index of the first character on this page, from 1).

**StartTableRow** – index of the first table row on the page, if the table is the first item on the page; -1 otherwise.

**EndTableRow** – index of the last table row on the page, if the table is the last item on the page; -1 otherwise.

**Position** – page position, coordinates of the top left page corner. By default, it is calculated by adding offset to the previous page. The coordinates are relative to the top left corner of a scrollable area in TSVRichViewEdit's window.

**ZoomPercent** – page zooming. By default, it is equal to zoom percent for the previous page. For the first page, it is equal to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup>.

**ZOrder** – the page's Z-order. By default it is the same as for the previous page. For the first page, it is 0. Pages having the same Z-order value are drawn in the order of their numeration. Pages with larger Z-order value are drawn above the pages with smaller Z-order values. If PageProperty<sup>(58)</sup>.UsePageOrders<sup>(142)</sup> = False, this parameter is ignored.



**See also:**

- CalculatePagePosition <sup>(73)</sup>
- CalculateAllPagePositions <sup>(72)</sup>

**Example:**

```
// Arranging pages in 4 columns, 10 pixels between rows,
// 30 pixels between columns
procedure TFColumns.SRichViewEdit1GetPagePos(Sender: TFSRichViewEdit;
  PageNo : Integer; StartItemNo, EndItemNo, OffsetInStartItem,
  StartTableRow, EndTableRow: Integer;
  var Position : TPoint; var ZoomPercent : Single;
  var ZOrder : Integer);
begin
  Position.X := 10 + ((PageNo - 1) mod 4) * (Sender.PageWidthPix + 30);
  Position.Y := 10 + ((PageNo - 1) div 4) * (Sender.PageHeightPix + 10);
end;
```

**1.3.1.3.25 TFSRichViewEdit.OnHMenuClickButton**

Occurs when the user clicks one of buttons of the horizontal menu (toolbar).

**property** OnHMenuClickButton: TFSRVClickButtonEvent <sup>(272)</sup>;

**ToolButton** – the clicked button (one of MenuHButtons <sup>(55)</sup>), or *nil*, if the user clicked outside any button.

**See also:**

- OnVMenuClickButton <sup>(121)</sup>
- OnHMenuEnterButton <sup>(109)</sup>
- MenuHorizontal <sup>(55)</sup>

**1.3.1.3.26 TFSRichViewEdit.OnHMenuEnterButton**

The event occurs when the user moves the mouse pointer to one of the buttons of the horizontal menu.

**property** OnHMenuEnterButton: TFSRVClickButtonEvent <sup>(272)</sup>;

**ToolButton** – the button under the mouse pointer, one of MenuHButtons <sup>(55)</sup>.

**See also:**

- OnVMenuEnterButton <sup>(122)</sup>
- OnHMenuClickButton <sup>(109)</sup>
- MenuHorizontal <sup>(55)</sup>

**1.3.1.3.27 TFSRichViewEdit.OnHScrolled**

Occurs on changing position of horizontal scrollbar.

**property** OnHScrolled: TNotifyEvent;

This event occurs when the user changes position of horizontal scrollbar, or when new value is assigned to HScrollPos <sup>(53)</sup> property.

**See also:**

- OnVScrolled<sup>(122)</sup>

**1.3.1.3.28 TRichViewEdit.OnHTMLSaveImage**

Occurs when RichViewEdit<sup>(60)</sup> saves item containing image to HTML file or stream. This event allows to modify saving procedure for all (or some) images.

**property** OnHTMLSaveImage: TRVHTMLSaveImageEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- OnSaveImage2<sup>(120)</sup>

**1.3.1.3.29 TRichViewEdit.OnImportPicture**

Occurs when loading RTF file or stream containing links to image files. This event also can be used by [TRVHtmlImporter](#).

**property** OnImportPicture: TRVImportPictureEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**1.3.1.3.30 TRichViewEdit.OnItemAction**

Occurs on insertion, deletion and some other modifications of item in Sender's document.

**property** OnItemAction: TRVItemActionEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**1.3.1.3.31 TRichViewEdit.OnItemHint**

Allows to define custom popup hints for items in Sender's document.

**property** OnItemHint: TRVItemHintEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**1.3.1.3.32 TRichViewEdit.OnItemResize**

Occurs after resizing image, control or table with the mouse.

**property** OnItemResize : TRVItemResizeEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

**1.3.1.3.33 TRichViewEdit.OnItemTextEdit**

Occurs when text item's text is changed as a result of editing operation.

**property** OnItemTextEdit: TRVItemTextEditEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.34 TSVRichViewEdit.OnJump

Occurs when the user clicks on hypertext item.

**property** OnJump: TJumpEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- OnRVMouseMove<sup>(118)</sup>

#### 1.3.1.3.35 TSVRichViewEdit.OnLoadCustomFormat

This event allows loading data from field in your own format.

**type**

```
TSRVCustomFormatEvent = procedure (Sender: TSVRichViewEdit(40);  
    Stream: TStream; var DoDefault: Boolean) of object;
```

**property** OnLoadCustomFormat: TDBSRVCustomFormatEvent;

See the event of the same name in the TRichView help file (event of TRichView).

This event may be called:

- in TDBSRichViewEdit<sup>(169)</sup>
- if Document<sup>(51)</sup> is linked to a database field using LiveBindings

**See also:**

- OnSaveCustomFormat<sup>(119)</sup>

#### 1.3.1.3.36 TSVRichViewEdit.OnLoadDocument

Occurs after loading document from the database field

**property** OnLoadDocument: TNotifyEvent;

See the event of the same name in the TRichView help file (event of TRichView).

This event allows to preprocess the loaded document before displaying.

This event may be called:

- in TDBSRichViewEdit<sup>(169)</sup>
- if Document<sup>(51)</sup> is linked to a database field using LiveBindings

**See also:**

- OnNewDocument<sup>(113)</sup>

#### 1.3.1.3.37 TSVRichViewEdit.OnMarginsChanged

Occurs before margins are changed.

**type**

```
TSRVMarginsChangedEvent = procedure (Sender: TSVRichViewEdit;  
    var NewLeft, NewTop, NewRight, NewBottom : Integer;  
    OldLeft, OldTop, OldRight, OldBottom : Integer) of object;
```

**property** OnMarginsChanged : TSRVMarginsChanged;

**NewLeft, NewTop, NewRight, NewBottom** – new margins, pixels.

**OldLeft, OldTop, OldRight, OldBottom** – old margins, pixels.

Unlike values of `LeftMarginPix`<sup>(54)</sup>, `TopMarginPix`<sup>(68)</sup>, `RightMarginPix`<sup>(60)</sup>, `BottomMarginPix`<sup>(48)</sup> properties, these parameters contain unscaled values (as if `ViewProperty`<sup>(69)</sup>.`ZoomPercent`<sup>(168)</sup>=100).

If one of **Old\*\*\*** parameters is equal to -1, this margin was not changed; the corresponding **New\*\*\*** parameter must be ignored.

If you assign -1 to one of **New\*\*\*** parameters, this margin will not be changed.

This event occurs:

- when assigning new values to `LeftMargin`<sup>(138)</sup>, `RightMargin`<sup>(142)</sup>, `TopMargin`<sup>(142)</sup>, `BottomMargin`<sup>(136)</sup> properties of `PageProperty`<sup>(58)</sup>, or
- when calling `SetMargin`<sup>(97)</sup>, `SetMarginMM`<sup>(97)</sup> or `SetMarginUnit`<sup>(98)</sup> methods.

**Example:**

```
// SRVPrint: TSRVPrint(204) is placed on the form
procedure TForm1.SRichViewEdit1MarginsChanged(
  Sender: TSVRichViewEdit; var NewLeft, NewTop, NewRight,
  NewBottom: Integer; OldLeft, OldTop, OldRight, OldBottom: Integer);
var
  ValueLeft, ValueRight, ValueTop, ValueBottom : Integer;
begin
  if SRVPrint1.CheckMargin(218)(ValueLeft, ValueTop, ValueRight, ValueBottom,
    NewLeft, NewTop, NewRight, NewBottom) then begin
    NewLeft := ValueLeft;
    NewTop := ValueTop;
    NewRight := ValueRight;
    NewBottom := ValueBottom;
  end;
end;
```

### 1.3.1.3.38 TSVRichViewEdit.OnMessageControl

Occurs when a message from **AControl** is received.

```
type
  TSVRMessageControl = procedure (Sender: TSVRichViewEdit(40);
    AControl : TControl; var Message: TMessage) of object;
```

```
property OnMessageControl : TSVRMessageControl;
```

**AControl** – the control that sent the message. This control was inserted in the editor.

**Message** – the message from the control.

You can use this method to process messages from controls inserted in the editor. This is an alternative way to using events of these controls.

**Example:**

```
procedure TForm1.SRichViewEdit1MessageControl(
  Sender: TSVRichViewEdit(40); AControl: TControl; var Message: TMessage);
begin
  // if 'Ready' button is pressed
  if (AControl.Name = 'Ready') and (Message.Msg = WM_LBUTTONDOWN) then
```

```
...  
end;
```

#### 1.3.1.3.39 TScaleRichViewEdit.OnNewDocument

Occurs when creating or before loading document

**property** OnNewDocument: TNotifyEvent;

See the event of the same name in the TScaleRichView help file (event of TScaleRichView).

This event may be called:

- in TDBScaleRichViewEdit<sup>169</sup>
- if Document<sup>51</sup> is linked to a database field using LiveBindings

Occurs:

- on creating new document (before displaying field from newly added record);
- before loading document from the field.

Use this event to assign default values to properties (such as background, margins, styles).

**See also:**

- OnLoadDocument<sup>111</sup>.

#### 1.3.1.3.40 TScaleRichViewEdit.OnOleDragEnter

OLE drag&drop event. It occurs when the user drags the mouse pointer into the editor. Indicates whether a drop can be accepted, and, if so, the effect of the drop.

**property** OnOleDragEnter: TRVOleDragEnterEvent;

See the event of the same name in the TScaleRichView help file (event of TCustomScaleRichViewEdit).

#### 1.3.1.3.41 TScaleRichViewEdit.OnOleDragLeave

OLE drag&drop event. It occurs when the user drags the mouse cursor out of the editor or cancels the current drag&drop operation.

**property** OnOleDragLeave: TNotifyEvent;

See the event of the same name in the TScaleRichView help file (event of TCustomScaleRichViewEdit).

#### 1.3.1.3.42 TScaleRichViewEdit.OnOleDragOver

OLE drag&drop event. It occurs when the user moves the mouse pointer over the editor. Called only if the dragging was accepted in OnOleDragEnter<sup>113</sup> (or automatically by the editor).

**property** OnOleDragOver: TRVOleDragOverEvent;

See the event of the same name in the TScaleRichView help file (event of TCustomScaleRichViewEdit).

#### 1.3.1.3.43 TScaleRichViewEdit.OnOleDrop

OLE drag&drop event. It occurs when the user completes drag&drop operation into the editor. This event allows you to insert data in your own format.

**property** OnOleDrop: TRVOleDropEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

#### 1.3.1.3.44 TSRichViewEdit.OnPageCountChanged

Occurs after PageCount<sup>(57)</sup> is changed.

**property** OnPageCountChanged: TNotifyEvent;

**See also:**

- OnCurrentPageChange<sup>(107)</sup>

#### 1.3.1.3.45 TSRichViewEdit.OnPageFormatChanged

Occurs after PageProperty<sup>(58)</sup>.PageFormat<sup>(139)</sup> is changed.

**property** OnPageFormatChanged: TNotifyEvent;

#### 1.3.1.3.46 TSRichViewEdit.OnPageScrolled

Occurs when a different page becomes visible.

**property** OnPageScrolled: TNotifyEvent;

The event may occur on changing position of vertical scrollbar. If a new page comes into the user's field of view (the middle of the screen), the event occurs. Index of this page is assigned to ScrolledPage<sup>(65)</sup>.

#### 1.3.1.3.47 TSRichViewEdit.OnPaint

Occurs on repainting.

**type**

```
TSRVPaintEvent = procedure (Sender: TSRichViewEdit(40); Canvas: TCanvas;
  Prepaint: Boolean; PaintRect: TRect) of object;
```

**property** OnPaint: TSRVPaintEvent;

The event occurs after repainting. You can draw something on top of default drawing.

**Canvas** – canvas where to draw.

**Prepaint** reserved for future use. This event occurs when the document is already painted.

**PaintRect** specifies the area that needs to be repainted. You can use this parameter to optimize drawing.

**See also:**

OnPaintPage<sup>(115)</sup>

OnPaintComponent<sup>(114)</sup>

#### 1.3.1.3.48 TSRichViewEdit.OnPaintComponent

Allows custom painting for controls inserted in the editor.

**type**

```
TSRVComponentEvent = procedure (Sender: TSRichViewEdit(40);
  PaintMe: TControl; Canvas: TCanvas; ComponentRect: TRect;
```

```
var isPaint: Boolean) of object;
```

```
property OnPaintComponent: TSRVComponentEvent;
```

All controls are inserted in RichViewEdit<sup>(60)</sup>. TSRichViewEdit component creates images of controls and uses them when it needs to display them.

Not all controls can be drawn by the default procedure. This event allows you to draw controls yourself.

**PaintMe** – control to paint.

**Canvas** – canvas where to paint.

**ComponentRect** – coordinates where to paint.

**isPaint** - set to *True* if you drew this control (otherwise it will be drawn by the default procedure).

#### 1.3.1.3.49 TSRichViewEdit.OnPaintPage

Occurs before and after the page is drawn.

**type**

```
TSRVPaintPage = procedure (Sender: TObject; PageNo: Integer;  
    PageRect, R: TRect; Canvas: TCanvas; Prepaint, Printing: Boolean)  
    of object;
```

```
property OnPaintPage: TSRVPaintPage;
```

You can draw something on below or above the default drawing.

**PageNo** – index of the page, from 1.

**PageRect** – rectangle where to draw the page. Its size is the page size in 96 DPI and 100% zooming.

**R** – client coordinates of area that needs to be repainted.

**Canvas** – canvas where to paint.

**Prepaint** is *True* if this event is called before the default drawing (to draw a background), and *False* otherwise.

**Printing** is *True* if this event is called for printing the page.

You should paint as if zooming is 100%. When drawing on the screen, all painting made in this event is automatically scaled according to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup>.

#### Example: drawing page numbers

```
procedure TForm1.SRichVewEdit1PaintPage(Sender: TObject; PageNo: Integer;  
    PageRect, R: TRect; Canvas: TCanvas; Prepaint, Printing: Boolean);  
var  
    H : Integer;  
    Text: String;  
begin  
    if Prepaint then  
        exit;  
    Canvas.Brush.Style := bsClear;  
    Canvas.Font.Assign(SRichViewEdit1.RichViewEdit(60).Style.TextStyles[0]);
```

```

H := SRichViewEdit1.TopMargin100Pix68;
Text := 'Page ' + IntToStr(PageNo);
Canvas.TextOut(
    (PageRect.Left + PageRect.Right - Canvas.TextWidth(Text)) div 2,
    PageRect.Top+H div 2, Text);
end;

```

**See also:**

OnPaint<sup>114</sup>

### 1.3.1.3.50 TSRichViewEdit.OnParaStyleConversion

Occurs while executing ApplyParaStyleConversion method of RichViewEdit<sup>60</sup>, RVHeader<sup>64</sup>, RVFooter<sup>62</sup>, or RVNote<sup>64</sup> (these TRichViewEdit controls are specified in the Sender parameter)

**property** OnParaStyleConversion: TRVStyleConversionEvent;

This event allows you to create custom conversion procedure for styles of the selected paragraphs. This event is called for all selected items allowing to change their paragraph style.

It can be useful for implementing commands like "change paragraph alignment", "change paragraph indents", etc.

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

**See also:**

- OnStyleConversion<sup>121</sup>

### 1.3.1.3.51 TSRichViewEdit.OnPaste

Occurs when pasting from the Clipboard. Allows pasting data in custom formats.

**property** OnPaste: TRVPasteEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

**See also:**

- ActiveEditor<sup>47</sup>

### 1.3.1.3.52 TSRichViewEdit.OnPrinting

Occurs on printing.

**type**

```

TSRVPrintingEvent = procedure (Sender: TSRichViewEdit40;
    PageCompleted: Integer; Step: TRVPrintingStep) of object;

```

**property** OnPrinting : TSRVPrintingProgressEvent;

This is the analog of TRVPrint.OnSendingToPrinter, see the TRichView help file.

**See also:**

- TSRVPrint.OnSendingToPrinter<sup>222</sup>



#### 1.3.1.3.53 TSVRichViewEdit.OnProgress

Occurs on long operations.

**property** OnProgress : TRVProgressEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.54 TSVRichViewEdit.OnReadField

Occurs when reading a field from RTF or DocX, allows to insert custom content.

**property** OnReadField: TRVReadFieldEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.55 TSVRichViewEdit.OnReadHyperlink

Occurs when reading hyperlinks from RTF, DocX, HTML, Markdown, or when dragging hyperlink into the editor.

**property** OnReadHyperlink: TRVReadHyperlink;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- OnWriteHyperlink<sup>(122)</sup>

#### 1.3.1.3.56 TSVRichViewEdit.OnReadMergeField

Occurs when reading content of merge fields from RTF, allows to modify it.

**property** OnReadHyperlink: TRVReaMergeFieldEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.57 TSVRichViewEdit.OnResize, OnResizing

The events occur when the component is resized.

**property** OnResize: TNotifyEvent;

**property** OnResizing: TNotifyEvent;

OnResizing occurs before TSVRichViewEdit processes resizing itself.

OnResize occurs after that.

#### 1.3.1.3.58 TSVRichViewEdit.OnResizeDoc

Occurs when RichViewEdit<sup>(60)</sup> is resized.

**property** OnResizeDoc: TNotifyEvent;

#### 1.3.1.3.59 TSVRichViewEdit.OnRVDbClick

Occurs when user double-clicks the primary mouse button when the mouse pointer is above the item in RichView (or on single left-click, if *rvsSingleClick* is in RichViewEdit<sup>(60)</sup>.Options)

**property** OnRVDbClick: TRVDbClickEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.60 TSVRichViewEdit.OnRVFControlNeeded

Occurs when reading control (inserted in Sender) from RVF file or stream, and this control was saved without its "body".

**property** OnRVFControlNeeded : TRVFControlNeededEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.61 TSVRichViewEdit.OnRVFImageListNeeded

Occurs when reading *bullet*, *hotspot* or list level with imagelist picture from RVF.

**property** OnRVFImageListNeeded : TRVFImageListNeededEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.62 TSVRichViewEdit.OnRVFPictureNeeded

Occurs when reading picture or *hot-picture* from RVF file or stream, and this picture was saved without its "body"

**property** OnRVFPictureNeeded : TRVFPictureNeededEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.63 TSVRichViewEdit.OnRVMouseDown

Occurs when the user presses a mouse button with the mouse pointer over the control.

**property** OnRVMouseDown : TRVMouseEvent;

**X** and **Y** parameters in this events are coordinates relative to RichViewEdit<sup>(60)</sup>. If you need coordinates of this TSVRichViewEdit control, use OnMouseDown event.

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- ConvertRVToSRV<sup>(73)</sup>
- OnRVMouseUp<sup>(118)</sup>

#### 1.3.1.3.64 TSVRichViewEdit.OnRVMouseMove

Occurs when mouse pointer is moved to the area of hypertext link.

**property** OnRVMouseMove: TRVMouseMoveEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- OnJump<sup>(111)</sup>

#### 1.3.1.3.65 TSVRichViewEdit.OnRVMouseUp

Occurs when the user releases a mouse button that was pressed with the mouse pointer over the component.

**property** OnRVMouseUp : TRVMouseEvent;

**X** and **Y** parameters in this events are coordinates relative to RichViewEdit<sup>(60)</sup>. If you need coordinates of this TSRichViewEdit control, use OnMouseUp event.

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- ConvertRVToSRV<sup>(73)</sup>
- OnRVMouseDown<sup>(118)</sup>

#### 1.3.1.3.66 TSRichViewEdit.OnRVRightClick

Occurs when the user releases the right mouse button over the RichView component.

**property** OnRVRightClick: TRVRightClickEvent;

**X** and **Y** parameters in this events are coordinates relative to RichViewEdit<sup>(60)</sup>. If you need coordinates of this TSRichViewEdit control, use OnMouseUp event.

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.67 TSRichViewEdit.OnSaveComponentToFile

Occurs when RichViewEdit<sup>(60)</sup> wants to save inserted control in text, RTF or HTML file.

**property** OnSaveComponentToFile: TRVSaveComponentToFileEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.68 TSRichViewEdit.OnSaveCustomFormat

This event allows saving data to field in your own format.

**type**

```
TSRVCustomFormatEvent = procedure (Sender: TSRichViewEdit(40);  
    Stream: TStream; var DoDefault: Boolean) of object;
```

**property** OnSaveCustomFormat: TDBSRVCustomFormatEvent;

This event may be called:

- in TDBSRichViewEdit<sup>(169)</sup>
- if Document<sup>(51)</sup> is linked to a database field using LiveBindings

See the event of the same name in the TRichView help file (event of TRichViewEdit).

**See also:**

- OnLoadCustomFormat<sup>(119)</sup>

#### 1.3.1.3.69 TSRichViewEdit.OnSaveDocXExtra

Allows saving additional information in DocX files.

**property** OnSaveDocXExtra: TRVSaveDocXExtraEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.70 TSVRichViewEdit.OnSaveHTMLExtra

Allows saving additional information in HTML.

**property** OnSaveHTMLExtra: TRVSaveHTMLExtraEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.71 TSVRichViewEdit.OnSaveImage2

Occurs when RichViewEdit<sup>60</sup> saves image to HTML file or stream.

**property** OnSaveImage2 : TRVSaveImageEvent2;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**See also:**

- OnHTMLSaveImage<sup>110</sup>

#### 1.3.1.3.72 TSVRichViewEdit.OnSaveItemToFile

Allows you to change how document items are saved in text, RTF or HTML file or stream.

**property** OnSaveItemToFile: TRVSaveItemToFileEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.73 TSVRichViewEdit.OnSaveRTFExtra

Allows saving additional information in RTF.

**property** OnSaveRTFExtra: TRVSaveRTFExtraEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.74 TSVRichViewEdit.OnSelect

Occurs when selection in the document was changed (not contents, but bounds)

**property** OnSelect: TNotifyEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

#### 1.3.1.3.75 TSVRichViewEdit.OnSmartPopupClick

Occurs when user clicks "smart popup" button.

**property** OnSmartPopupClick: TRVSmartPopupClickEvent;

This event allows implementing your own effect when clicking "smart popup" button (or on pressing its ShortCut). If you use menu, it's not necessary to process this event.

**See also:**

- SmartPopupProperties<sup>66</sup>
- SmartPopupVisible<sup>66</sup>

### 1.3.1.3.76 TSVRichViewEdit.OnSpellingCheck

Allows to mark misspelled words.

**property** OnSpellingCheck: TRVSpellingCheckEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

### 1.3.1.3.77 TSVRichViewEdit.OnStyleConversion

Occurs while executing ApplyStyleConversion method of RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup>, RVFooter<sup>(62)</sup>, or RVNote<sup>(64)</sup> (these TSVRichViewEdit controls are specified in the Sender parameter)

**property** OnStyleConversion: TRVStyleConversionEvent;

This event allows you to create custom conversion procedure of text styles for the selected text. This event is called for all selected text items allowing to change their text style.

It can be useful to implement commands like "make bold", "apply font", "change text color", etc.

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

**See also:**

- OnParaStyleConversion<sup>(116)</sup>

### 1.3.1.3.78 TSVRichViewEdit.OnTableIconClick

Occurs when the user clicked a table "icon" (rectangle in the top left corner of table)

**property** OnTableIconClick: TSRVMouseEvent<sup>(277)</sup>;

Table icon is displayed when the user moves the mouse pointer above the table, if ViewProperty<sup>(69)</sup>.UseTableIcons<sup>(164)</sup>=True.

This table is returned by GetTableIconItem<sup>(89)</sup>. A document where this table is located is returned by GetTableIconRVData<sup>(89)</sup>.

### 1.3.1.3.79 TSVRichViewEdit.OnTextFound

Occurs as a result of RichViewEdit<sup>(60)</sup>.SearchText.

**property** OnTextFound: TRVTextFoundEvent;

See the event of the same name in the TRichView help file (event of TCustomRichViewEdit).

### 1.3.1.3.80 TSVRichViewEdit.OnURLNeeded

An obsolete version of OnWriteHyperlink<sup>(122)</sup>

**property** OnURLNeeded: TRVURLNeededEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

### 1.3.1.3.81 TSVRichViewEdit.OnVMenuClickButton

Occurs when the user clicks one of buttons of the vertical menu (toolbar).

**property** OnVMenuClickButton: TSRVClickButtonEvent<sup>(272)</sup>;

**ToolButton** – the clicked button (one of MenuVButtons<sup>(56)</sup>), or *nil*, if the user clicked outside any button.

**See also:**

- OnHMenuClickButton<sup>(109)</sup>
- OnVMenuEnterButton<sup>(122)</sup>
- MenuVertical<sup>(56)</sup>

**1.3.1.3.82 TSRichViewEdit.OnVMenuEnterButton**

The event occurs when the user moves the mouse pointer to one of the buttons of the vertical menu.

**property** OnVMenuEnterButton: TSRVClickButtonEvent<sup>(272)</sup>;

**ToolButton** – the button under the mouse pointer, one of MenuVButtons<sup>(56)</sup>.

**See also:**

- OnHMenuEnterButton<sup>(109)</sup>
- OnVMenuClickButton<sup>(121)</sup>
- MenuVertical<sup>(56)</sup>

**1.3.1.3.83 TSRichViewEdit.OnVScrolled**

Occurs on changing position of vertical scrollbar.

**property** OnVScrolled: TNotifyEvent;

This event occurs when the user changes position of vertical scrollbar, or when new value is assigned to VScrollPos<sup>(70)</sup> property.

**See also:**

- OnHScrolled<sup>(109)</sup>

**1.3.1.3.84 TSRichViewEdit.OnWriteHyperlink**

Occurs when RichViewEdit<sup>(60)</sup> wants to save hypertext link to HTML, RTF, or DocX file.

**property** OnWriteHyperlink: TRVWriteHyperlink;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**1.3.1.3.85 TSRichViewEdit.OnWriteObjectProperties**

Occurs when RichViewEdit<sup>(60)</sup> wants saves and object (for example, image) to HTML, RTF, or DocX file. This event allows saving additional object properties.

**property** OnWriteObjectProperties: TRVWriteObjectPropertiesEvent;

See the event of the same name in the TRichView help file (event of TCustomRichView).

**1.3.1.3.86 TSRichViewEdit.OnZoomChanged**

Occurs after ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> or .ZoomMode<sup>(166)</sup> are changed.

**property** OnZoomChanged: TNotifyEvent;

**See also:**

- OnZoomViewChanged<sup>(123)</sup>

### 1.3.1.3.87 TSRichViewEdit.OnZoomViewChanged

Occurs when zooming properties are changed.

**property** OnZoomViewChanged: TNotifyEvent;

This even occurs when one of the following properties of ViewProperty<sup>(69)</sup> is changed:

- ZoomPercentEdit<sup>(169)</sup>
- ZoomModeEdit<sup>(167)</sup>
- ZoomPercentIN<sup>(169)</sup>
- ZoomModeIN<sup>(167)</sup>
- ZoomPercentOUT<sup>(169)</sup>
- ZoomModeOUT<sup>(167)</sup>

**See also:**

- OnZoomChanged<sup>(122)</sup>

## 1.3.1.4 Classes of properties

### Classes of TSRichViewEdit<sup>(40)</sup> Properties

- TSRVBackgroundProperty<sup>(123)</sup> – properties of the editor's background; the class of BackgroundProperty<sup>(48)</sup> property.
- TSRVPropertyTBH<sup>(148)</sup> (inherited from TSRVCustomPropertyTB<sup>(127)</sup>) – properties of the toolbar in the horizontal scrollbar area; the class of MenuHorizontal<sup>(55)</sup> property;
- TSRVPropertyTBV<sup>(149)</sup> (inherited from TSRVCustomPropertyTB<sup>(127)</sup>) – properties of the toolbar in the vertical scrollbar area; the class of MenuVertical<sup>(56)</sup> property;
- TSRVLineNumberProperty<sup>(131)</sup> – properties of line numbers; the class of LineNumberProperty<sup>(54)</sup>;
- TSRVPageProperty<sup>(133)</sup> – page properties; most of them affect page appearance both on the screen and on paper; the class of PageProperty<sup>(58)</sup> property.
- TSRVViewProperty<sup>(154)</sup> – visual page properties; most of them affect page appearance only on the screen; the class of ViewProperty<sup>(69)</sup> property.

### 1.3.1.4.1 TSRVBackgroundProperty

TSRVBackgroundProperty is the class for TSRichViewEdit<sup>(40)</sup>.BackgroundProperty<sup>(48)</sup> property.

**Unit** SclRView.

#### Syntax

```
TSRVBackgroundProperty = class (TPersistent)
```

#### Hierarchy

TObject

TPersistent

### Properties for area around pages

This class defines properties for an advanced background filling. An advanced background is a combination of a gradient filling and a picture. It is activated if Visible<sup>(127)</sup> = True.

A gradient is defined in FillType<sup>(125)</sup>, ColorBegin<sup>(124)</sup>, ColorMiddle<sup>(124)</sup>, ColorEnd<sup>(124)</sup>, PercentMiddle<sup>(126)</sup> properties.

A picture is defined in Picture<sup>(126)</sup>, PicturePosition<sup>(126)</sup>, and TransparentColor<sup>(126)</sup> properties.

## Properties for pages

GlobalPageBackgroundColor<sup>(125)</sup> allows changing color of pages.

### 1.3.1.4.1.1 Properties

#### In TSRVBackgroundProperty

- ColorBegin<sup>(124)</sup>
- ColorEnd<sup>(124)</sup>
- ColorMiddle<sup>(124)</sup>
- FillType<sup>(125)</sup>
- PercentMiddle<sup>(126)</sup>
- Picture<sup>(126)</sup>
- PicturePosition<sup>(126)</sup>
- TransparentColor<sup>(126)</sup>
- Visible<sup>(127)</sup>

Specifies the initial color of gradient filling.

**property** ColorBegin: TColor;

If FillType<sup>(125)</sup> = *svtFlat*, this property defines a background color.

This color is also used for background of zooming panel, if it does not use a gradient.

If VCL theme is active (in Delphi XE2 or newer), and this color is a system color, a corresponding theme color is used.

**Default value:**

**\$00E7BE9F**

Specifies the ending color of gradient filling

**property** ColorEnd: TColor;

If FillType<sup>(125)</sup> = *svtFlat*, this property is not used.

If VCL theme is active (in Delphi XE2 or newer), and this color is a system color, a corresponding theme color is used.

**Default value:**

**\$00CD9165**

Specifies the middle color of gradient filling

**property** ColorMiddle : TColor;

If FillType<sup>(125)</sup> = *svtFlat*, this property is not used.



If **ColorMiddle** = *clNone*, this color is ignored, a gradient is drawn from **ColorBegin**<sup>(124)</sup> to **ColorEnd**<sup>(124)</sup>.

If VCL theme is active (in Delphi XE2 or newer), and this color is a system color, a corresponding theme color is used.

**Default value:**

\$00BD865D

**See also:**

- **PercentMiddle**<sup>(126)</sup>

Specifies the background filling mode.

**type**

```
TSRVFillType = (srvtfFlat, srvtfGradientH, srvtfGradientV);
```

**property** FillType : TSRVFillType;

| Value                 | Meaning   |
|-----------------------|---|
| <i>srvtfFlat</i>      | A plain colored background, the color is <b>ColorBegin</b> <sup>(124)</sup> .   |
| <i>srvtfGradientH</i> | A horizontal gradient, the background is filled with horizontal gradient using three colors: from <b>ColorBegin</b> <sup>(124)</sup> through <b>ColorMiddle</b> <sup>(124)</sup> to <b>ColorEnd</b> <sup>(124)</sup> .<br><br>If <b>ColorMiddle</b> = <i>clNone</i> , only two colors are used. |
| <i>srvtfGradientV</i> | A vertical gradient, the background is filled with vertical gradient using three colors: from <b>ColorBegin</b> <sup>(124)</sup> through <b>ColorMiddle</b> <sup>(124)</sup> to <b>ColorEnd</b> <sup>(124)</sup> .<br><br>If <b>ColorMiddle</b> = <i>clNone</i> , only two colors are used.     |

**ColorMiddle**<sup>(124)</sup> is shifted by the percent specified in **PercentMiddle**<sup>(126)</sup>.

**Default value:**

*srvtfFlat*

Specifies a page background color.

**property** GlobalPageBackgroundColor: TColor;

Unlike all other properties of **BackgroundProperty**<sup>(48)</sup>, this property affects pages, not the area around them.

This property (if not equal to *c/None*) overrides the value of RichViewEdit<sup>(60)</sup>.Color. Unlike RichViewEdit<sup>(60)</sup>.Color, this property is not stored in RVF files.

This property has a lower priority than the editor's ReadModeProperty<sup>(59)</sup>.PageColor<sup>(153)</sup>.

**Default value:**

*c/None*

Specifies the shift of the middle color, percent of the gradient width.

**property** PercentMiddle: Single;

This value defines the position where a gradient color becomes equal to ColorMiddle<sup>(124)</sup>.

**Default value:**

50.0

Specifies the picture to display above the gradient.

**property** Picture: TPicture;

The position of this picture is defined in PicturePosition<sup>(126)</sup>.

Specifies the position of Picture<sup>(126)</sup>.

**type**

```
TSRVPicturePosition = (srvtpNone, srvtpTopLeft, srvtpCenter,
    srvtpTile, srvtpStretch, srvtpScale);
```

**property** PicturePosition: TSRVPicturePosition;

| Value               | Meaning   |
|---------------------|---|
| <i>srvtpNone</i>    | picture is not displayed                                |
| <i>srvtpTopLeft</i> | picture is in the top left corner                       |
| <i>srvtpCenter</i>  | picture in the center                                   |
| <i>srvtpTile</i>    | picture is tiled  |
| <i>srvtpStretch</i> | picture is stretched to the window size                 |
| <i>srvtpScale</i>   | picture is stretched to the window size, proportionally |

**Default value:**

*srvtpNone*

Specifies a color of Picture<sup>(126)</sup> that will be transparent when it is drawn.

**property** ColorBegin: TColor;

This property is used only if Picture<sup>(126)</sup> contains TBitmap.

**Default value:**

*c/None* (no transparency)

Disables/enables background displaying.

**property** `Visible : Boolean;`

If the value is *False*, all properties of this class are ignored and background is filled with `TSRichViewEdit(40).Color`.

A zooming panel<sup>(168)</sup> is always filled according to `BackgroundProperty`, even if **Visible**=*False*.

**Default value:**

*False*

#### 1.3.1.4.2 TSRVCustomPropertyTB

TSRVCustomPropertyTB is the base class for `TSRVPropertyTBH(148)` and `TSRVPropertyTBV(149)`.

**Unit** `SRVToolBar`.

**Syntax**

```
TSRVCustomPropertyTB = class (TPersistent)
```

#### Hierarchy

*TObject*

*TPersistent*

#### Description

This class is not used directly. It is a base class for types of `MenuHorizontal(55)` and `MenuVertical(56)` properties of `TSRichViewEdit(40)`.

This class defines properties of toolbars. All properties are protected.

Some of these properties make sense only for toolbars displayed in tool windows, these properties are not published in `TSRVPropertyTBH(148)` and `TSRVPropertyTBV(149)`.

#### Properties

This class contains almost the same set of properties as `TSRVToolBar(189)` and `TSRVToolWindow(197)` components, but a hint area is hidden.

Buttons are defined in `Buttons(129)` collection (or in external collections, like in case of `MenuHorizontal(55)` and `MenuVertical(56)` properties). A size of buttons is defined in `ButtonWidth(129)` and `ButtonHeight(128)` properties. A spacing between buttons is specified in `Spacer(131)`. A horizontal positions of buttons is defined by `AlignButton(128)` and `Indent(130)` properties.

The toolbar can have a border, its width is specified in `BorderWidth(128)`.

The toolbar color is `Color(129)`. Colors of disabled, pressed and highlighted buttons are specified in `ColorDisable(129)`, `ColorDown(130)`, and `ColorSelect(130)` properties. A pen of frame around the toolbar is defined in `PenFrame(130)`.

## 1.3.1.4.2.1 Properties

**In TSRVCustomPropertyTB**

- [BorderWidth](#) <sup>(128)</sup>
- [BorderWidth](#) <sup>(128)</sup>
- [ButtonHeight](#) <sup>(128)</sup>
- [Buttons](#) <sup>(129)</sup>
- [ButtonWidth](#) <sup>(129)</sup>
- [Color](#) <sup>(129)</sup>
- [ColorDisable](#) <sup>(129)</sup>
- [ColorDown](#) <sup>(130)</sup>
- [ColorSelect](#) <sup>(130)</sup>
- [ImageList](#) <sup>(130)</sup>
- [Indent](#) <sup>(130)</sup>
- [PenFrame](#) <sup>(130)</sup>
- [ScaleImagesForDPI](#) <sup>(131)</sup>
- [Spacer](#) <sup>(131)</sup>
- ▶ [SRVToolBar](#) <sup>(131)</sup>

Defines the position of buttons in the toolbar window.

**property** `AlignButton: TSRVAlignButton` <sup>(271)</sup>;

**See also**

[Indent](#) <sup>(130)</sup>

**Default value:**

*srvabCenter*

Specifies a width of the toolbar border (spacing around the toolbar)

**property** `BorderWidth: Integer`;

A border (see [PenFrame](#) <sup>(130)</sup>) is drawn in this area. The rest of space (if `BorderWidth`>the pen's width) is painted with [Color](#) <sup>(129)</sup>.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

1

**See also:**

- [Spacer](#) <sup>(131)</sup>

Sets the height of tool buttons ([TSRVToolButton](#) <sup>(272)</sup>)

**property** `ButtonHeight: TRVPixel96Length`;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

20

**See also:**

- ButtonWidth<sup>(129)</sup>

Collection of tool buttons (TSRVToolButton<sup>(272)</sup>).

**property** Buttons: TSRVButtonCollection<sup>(271)</sup>;

For horizontal menu<sup>(55)</sup> and vertical menu<sup>(56)</sup> of TSRichViewEdit<sup>(40)</sup>, this property is not published. For them, buttons are available in MenuHButtons<sup>(55)</sup> and MenuVButtons<sup>(56)</sup> properties of TSRichViewEdit<sup>(40)</sup>.

Width of tool buttons (TSRVToolButton<sup>(272)</sup>).

**property** ButtonWidth: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

26

**See also:**

- ButtonHeight<sup>(128)</sup>

Specifies the toolbar background color.

**property** Color: TColor;

This color is used for background of:

- the border around the toolbar (see BorderWidth<sup>(128)</sup>)
- area between the buttons (see Spacer<sup>(131)</sup>)
- tool buttons (if the button is in a normal state, i.e. not highlighted, pressed or disabled).

**Default value:**

*c/White*

**See also:**

- ColorDisable<sup>(129)</sup>
- ColorDown<sup>(130)</sup>
- ColorSelect<sup>(130)</sup>

Background color of disabled tool buttons.

**property** ColorDisable: TColor;

This color is used for buttons having Enabled<sup>(273)</sup>=False.

**Default value:**

*c/Silver*

**See also:**

- Color<sup>(129)</sup>
- ColorDown<sup>(130)</sup>
- ColorSelect<sup>(130)</sup>

Background color of pressed tool buttons.

**property** ColorDown: TColor;

**Default value:**

\$00EED2C1

**See also:**

- Color<sup>(129)</sup>
- ColorDisable<sup>(129)</sup>
- ColorSelect<sup>(130)</sup>

Background color of highlighted tool buttons (buttons under the mouse pointer).

**property** ColorSelect: TColor;

**Default value:**

\$00E2B598

**See also:**

- Color<sup>(129)</sup>
- ColorDisable<sup>(129)</sup>
- ColorDown<sup>(130)</sup>

A reference to image list for images in tool buttons<sup>(129)</sup>.

**property** ImageList: TCustomImageList;

This image list must contain buttons glyphs for all buttons states (normal, pressed, disabled) (you can use the same image for all of them).

When a toolbar displayed on a monitor that has pixel density (DPI) higher than 96, the toolbar draws images stretched, proportionally to the screen DPI. To turn off stretching, assign ScaleImagesForDPI<sup>(131)</sup> = *False*.

Changes the position of buttons in the toolbar window.

**property** Indent: Integer;

If AlignButton<sup>(128)</sup> = *svabLeft*, buttons are shifted by Indent to the right.

If AlignButton<sup>(128)</sup> = *svabRight*, buttons are shifted by Indent to the left.

If AlignButton<sup>(128)</sup> = *svabCenter*, buttons are shifted by Indent/2 to the left.

**Default value:**

0

Specifies a pen for drawing lines in the tool bar.

**property** PenFrame: TPen;

This pen is used for drawing border around the toolbar. The width of this border is defined as a pen's width, not in BorderWidth<sup>(128)</sup>.

**Default value:**

Style=*psSolid*, Width=1, Color=\$00C56A31;

Specify whether images from ImageList<sup>(130)</sup> are displayed stretched according to the screen pixel density (DPI).

**property** ScaleImagesForDPI: Boolean;

If *True*, toolbar images are stretched from 96 DPI to the screen DPI.

If *False*, toolbar images are displayed as they are. This mode is useful if ImageList<sup>(130)</sup> contains images that corresponds to the current DPI.

#### Default value

*True*

Specifies the distance between tool buttons<sup>(129)</sup>.

**property** Spacer: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

#### Default value:

0

Returns a reference to the toolbar.

SRVToolBar: TSRVToolBar<sup>(189)</sup>;

This property can be used to get the toolbar position. Please do not modify properties of this toolbar.

### 1.3.1.4.3 TSRVLineNumberProperty

TSRVLineNumberProperty is the class for TSVRichViewEdit<sup>(40)</sup>.LineNumberProperty<sup>(54)</sup> property.

**Unit** SclRView.

#### Syntax

TSRVLineNumberProperty = **class**(TPersistent)

#### Hierarchy

*TObject*

*TPersistent*

#### Description

Line numbers are shown (both when displaying on the screen and when printing) if Visible<sup>(133)</sup>=*True*.

Numbers are counted on each page from StartFrom<sup>(132)</sup>. Every Step<sup>(132)</sup>-th line number is shown. Other line numbers are not shown, or shown as ticks, depending Ticks<sup>(133)</sup> property.

Line numbers are displayed to the left side of the main text area (or to the right side, if BiDiMode<sup>(48)</sup>=*rvbdRightToLeft*). A distance from the main text area is defined in DistanceFromText<sup>(132)</sup>.

The font is defined in Font<sup>(132)</sup>.

## 1.3.1.4.3.1 Properties

**In TSRVLineNumberProperty**

- DistanceFromText <sup>(132)</sup>
- Font <sup>(132)</sup>
- StartFrom <sup>(132)</sup>
- Step <sup>(132)</sup>
- TablesAsLines <sup>(133)</sup>
- Ticks <sup>(133)</sup>
- Visible <sup>(133)</sup>

Specifies a distance between line numbers and the main text area.

**property** Visible: Boolean;

This value is measured in pixels in the resolution 96 dpi.

**Default value:**

5

Specifies the font for drawing line numbers.

**type**

```
TSRVLineNumberFont = class(TFont);
```

**property** Font: TSRVLineNumberFont;

**Default value:**

'Tahoma', 8, *cIBtnShadow*

Specifies the value of the first number on a page and the increment.

**property** StartFrom: Integer;

**Default value:**

1

**property** Step: Integer;

Only every **Step**-th line number is drawn as a number. Other numbers are not drawn, or drawn as ticks, depending on Ticks <sup>(133)</sup> property.

If Ticks <sup>(133)</sup> = *True*, the first number on the page is always shown.

**Examples:**

- Step=1; line numbers: 1 2 3 4 5, etc.
- Step=10, Ticks=*False*; line numbers: 10 20 30, etc. (other line numbers are not shown)
- Step=10, Ticks=*True*; line numbers: 1 . . . | . . . . 10 . . . . | . . . . 20, etc.

**Default value:**

1



Specifies whether tables are counted as lines.

**property** TablesAsLines: Boolean;

If *True*, each table is counted as one line (although line numbers are not drawn for tables). This method is compatible with the results returned by GetCurrentLineCol<sup>(83)</sup>.

If *False*, tables are ignored when counting lines (like in Microsoft Word)

**Default value:**

*True*

Shows or hides ticks.

**property** Ticks: Boolean;

If *True*, and Step<sup>(132)</sup>>1, all intermediate line numbers are drawn as ticks. Additionally, a vertical line is drawn between text and line numbers.

If *False*, intermediate line numbers are not drawn.

**Default value:**

*False*

Shows or hides line numbers.

**property** Visible: Boolean;

**Default value:**

*False*

#### 1.3.1.4.4 TSRVPageProperty

TSRVPageProperty is the class for TSRichViewEdit<sup>(40)</sup>.PageProperty<sup>(58)</sup> property.

**Unit** SclRView.

**Syntax**

TSRVPageProperty = **class**(TPersistent)

#### Hierarchy

TObject

TPersistent

#### Properties

##### Page sizes and margins

The following properties define the page size: PageFormat<sup>(139)</sup> and Orientation<sup>(138)</sup>.

PageWidth<sup>(141)</sup> and PageHeight<sup>(139)</sup> are usually calculated automatically, but they can define a page size if PageFormat<sup>(139)</sup>=*srvfmCustom*.

The following properties define margins: LeftMargin<sup>(138)</sup>, TopMargin<sup>(142)</sup>, RightMargin<sup>(142)</sup>, BottomMargin<sup>(136)</sup>.

MirrorMargins<sup>(138)</sup> allows switching left and right margins on even pages.

All sizes are measured in `TSRichViewEdit.UnitsProgram` <sup>(68)</sup>.

### Viewing modes

The following properties are used to implement different viewing modes: `AutoWidth` <sup>(135)</sup> and `PageViewMode` <sup>(141)</sup>.

Typical modes:

- "Normal mode" (one long page, width depends on `PageWidth` <sup>(141)</sup>): `AutoWidth` <sup>(135)</sup> = `False`, `PageViewMode` <sup>(141)</sup> = `False`.
- "Web mode" (one long page, width depends on the window size): `AutoWidth` <sup>(135)</sup> = `True`, `PageViewMode` <sup>(141)</sup> = `False`.
- "Page View Mode": `AutoWidth` <sup>(135)</sup> = `False`, `PageViewMode` <sup>(141)</sup> = `True`.

A border can be drawn around pages, see `BorderPen` <sup>(135)</sup>. A printable area can be shown, see `PrintableAreaPen` <sup>(141)</sup> (the current printer cannot print beyond this area).

The caret is drawn using `CaretPen` <sup>(136)</sup>.

### Page numbering

Page numbers are printed, if `PageNoVisible` <sup>(140)</sup> = `True`. This is a simple option to display page numbering. The alternative (recommended) option is inserting "page number" field (`TRVPageNumberItemInfo` item) in a header or a footer <sup>(66)</sup>.

They are printed at the top or the bottom of pages, depending on `PageNoVAlign` <sup>(140)</sup> property. A horizontal alignment is specified in `PageNoHAlign` <sup>(140)</sup>, font is specified in `PageNoFont` <sup>(139)</sup>.

The numbering is started from `PageNoFromNumber` <sup>(140)</sup>. You can omit page numbers on one or more first pages, using `PageNoFirst` <sup>(139)</sup>.

`PageNoFromNumber` <sup>(140)</sup> is also used as a starting value for "page number" fields.

### Header and footer

Headers <sup>(66)</sup> are displayed, if `HeaderVisible` <sup>(137)</sup> = `True`. Headers position is defined in `HeaderY` <sup>(137)</sup>.

Footers <sup>(66)</sup> are displayed, if `FooterVisible` <sup>(136)</sup> = `True`. Footers position is defined in `FooterY` <sup>(137)</sup>.

If `TitlePage` <sup>(142)</sup> = `True`, special headers are used for the first page. If `FacingPages` <sup>(136)</sup> = `True`, special headers are used for even pages.

### See also:

`TSRVViewProperty` <sup>(154)</sup>.

#### 1.3.1.4.4.1 Properties

### In `TSRVPageProperty`

- `AutoWidth` <sup>(135)</sup>
- `BorderPen` <sup>(135)</sup>
- `BottomMargin` <sup>(136)</sup>
- `CaretPen` <sup>(136)</sup>
- `FacingPages` <sup>(136)</sup>
- `FooterVisible` <sup>(136)</sup>
- `FooterY` <sup>(137)</sup>

- HeaderVisible <sup>(137)</sup>
- HeaderY <sup>(137)</sup>
- LeftMargin <sup>(138)</sup>
- MirrorMargins <sup>(138)</sup>
- Orientation <sup>(138)</sup>
- PageFormat <sup>(139)</sup>
- PageHeight <sup>(139)</sup>
- PageNoFirst <sup>(139)</sup>
- PageNoFont <sup>(139)</sup>
- PageNoFromNumber <sup>(140)</sup>
- PageNoHAlign <sup>(140)</sup>
- PageNoVAlign <sup>(140)</sup>
- PageNoVisible <sup>(140)</sup>
- PageViewMod <sup>(141)</sup>
- PageWidth <sup>(141)</sup>
- PrintableAreaPen <sup>(141)</sup>
- RightMargin <sup>(142)</sup>
- TitlePage <sup>(142)</sup>
- TopMargin <sup>(142)</sup>
- UsePageOrders <sup>(142)</sup>

Specifies whether page width is automatically calculated depending on the editor's client width.

**property** AutoWidth: Boolean;

If this property is *True*, a document occupies all available width (except for the double PagePosProperty <sup>(58)</sup>.HPadding <sup>(146)</sup>), a horizontal scrollbar is never enabled. In a read mode <sup>(151)</sup>, document also occupies all available height (except for the double PagePosProperty <sup>(58)</sup>.VPadding <sup>(146)</sup>). In this mode, zooming <sup>(168)</sup> is applied to a document content and is not applied to sizes of pages. Page size <sup>(141)</sup> and format <sup>(139)</sup> specified are ignored in this mode.

This property may be used when implementing custom view modes, for example, "web mode" and "read mode".

**Default value:**

*False*

Specifies pen properties for drawing border around pages.

**type**

```
TSRVBorderPen = class (TPen);
```

**property** BorderPen: TSRVBorderPen;

To disable a page border drawing, assign BorderPen.Style = *psClear*.

**Default value:**

Color = \$003E3E3E, Width = 1.

Specifies a bottom margin of pages.

**property** BottomMargin: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

**Default value:**

25

Specifies pen properties for drawing the caret.

**type**

TSRV caretPen = **class** (TPen)

**property** CaretPen: TSRVCaretPen;

If **CaretPen.Width** = 0 (default), the system caret thickness is used (defined in the "Ease of Access" of the Settings app (or the Control Panel, depending in the Windows version)).

**CaretPen.Color** and **Mode** are ignored (the caret always uses **Mode** = *pmNot*)

Specifies whether special headers and footers are used for even pages.

**property** FacingPages: Boolean;

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetIntPropertyEd<sup>(97)</sup>.

A header for even pages is stored in SRichViewEdit.Subdocuments<sup>(66)</sup> [*srvhftEvenPagesHeader*].

A footer for even pages is stored in SRichViewEdit.Subdocuments<sup>(66)</sup> [*srvhftEvenPagesFooter*].

FacingPages turns on/off displaying headers and footers for even pages, but does not affect saving: headers and footers are loaded and saved regardless of this property.

Headers are shown only if HeaderVisible<sup>(137)</sup>=*True*, footers are shown only if FooterVisible<sup>(136)</sup>=*True*.

**Default value:**

*False*

**See also:**

- TitlePage<sup>(142)</sup>
- MirrorMargins<sup>(138)</sup>

Shows or hides footers.

**property** FooterVisible: Boolean;

Footers are stored in TSRichViewEdit<sup>(40)</sup>.Subdocuments<sup>(66)</sup> and edited in TSRichViewEdit.RVFooter<sup>(62)</sup>.

If FooterVisible=*False*, footers are not displayed, not loaded and not saved to files.

**Default value:**

*True*

**See also:**

- FooterY<sup>(137)</sup>
- HeaderVisible<sup>(137)</sup>
- TitlePage<sup>(142)</sup>
- FacingPages<sup>(136)</sup>

Specifies a distance between the bottom of the page and the bottom of the footer.

**property** FooterY: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

A footer is shown only if FooterVisible<sup>(136)</sup>=*True*.

Footers are stored in TSRichViewEdit<sup>(40)</sup>.Subdocuments and edited in TSRichViewEdit.RVFooter<sup>(62)</sup>.

This value is used for all types of footers.

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

**Default value:**

10

**See also**

- HeaderY<sup>(137)</sup>

Shows or hides headers.

**property** HeaderVisible: Boolean;

Headers are stored in TSRichViewEdit<sup>(40)</sup>.Subdocuments<sup>(66)</sup> and edited in TSRichViewEdit.RVHeader<sup>(64)</sup>.

If HeaderVisible=*False*, headers are not displayed, not loaded and not saved to files.

**Default value:**

*True*

**See also:**

- HeaderY<sup>(137)</sup>
- FooterVisible<sup>(136)</sup>
- TitlePage<sup>(142)</sup>
- FacingPages<sup>(136)</sup>

Specifies a distance between the top of the page and the top of the header.

**property** HeaderY: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

A header is shown only if HeaderVisible<sup>(137)</sup>=*True*.

Headers are stored in TSRichViewEdit<sup>(40)</sup>.Subdocuments and edited in TSRichViewEdit.RVHeader<sup>(64)</sup>.

This value is used for all types of headers.

This property can be changed as an editing (undoable) operation, see `TSRichViewEdit`<sup>(40)</sup>.  
`.SetFloatPropertyEd`<sup>(97)</sup>.

**Default value:**

10

**See also**

- `FooterY`<sup>(137)</sup>

Specifies a left margin of pages.

**property** `LeftMargin: TRVLength;`

This value is measured in `UnitsProgram`<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see `TSRichViewEdit`<sup>(40)</sup>.  
`.SetFloatPropertyEd`<sup>(97)</sup>.

**Default value:**

35

Instructs to exchange left and right margins on even pages.

**property** `MirrorMargins: Boolean;`

If `MirrorMargins = True`, then:

- for odd pages, `LeftMargin`<sup>(138)</sup> defines the left margin, `RightMargin`<sup>(142)</sup> defines the right margin; for even pages, `LeftMargin`<sup>(138)</sup> defines the right margin, `RightMargin`<sup>(142)</sup> defines the left margin;
- if page numbering is visible<sup>(140)</sup>, left and right alignment<sup>(140)</sup> of page numbers are exchanged on even pages.

This property can be changed as an editing (undoable) operation, see `TSRichViewEdit`<sup>(40)</sup>.  
`.SetIntPropertyEd`<sup>(97)</sup>.

**Default value:**

*False*

**See also:**

- `FacingPages`<sup>(136)</sup>

Specifies a page orientation

**property** `Orientation: TPrinterOrientation;`

Portrait or landscape.

Assigning to this property sets the proper values to `PageWidth`<sup>(141)</sup> and `PageHeight`<sup>(139)</sup> properties, if `PageFormat`<sup>(139)</sup> `<>srvfmCustom`.

This property can be changed as an editing (undoable) operation, see `TSRichViewEdit`<sup>(40)</sup>.  
`.SetIntPropertyEd`<sup>(97)</sup>.

**Default value:**

*poPortrait*

Specifies a page format (size).

**property** PageFormat: TSRVPageFormat<sup>(278)</sup>;

Assigning to this property changes values of PageWidth<sup>(141)</sup> and PageHeight<sup>(139)</sup> properties as well (if the new value <> *srvfmCustom*). This change depends on Orientation<sup>(138)</sup> property.

When this property is changed, OnPageFormatChanged<sup>(114)</sup> occurs.

If this value is equal to *srvfmCustom* (user-defined paper format), page size is defined in PageWidth<sup>(141)</sup> and PageHeight<sup>(139)</sup> properties.

This property can be changed as an editing (undoable) operation, see TSVRichViewEdit<sup>(40)</sup>.SetIntPropertyEd<sup>(97)</sup>.

#### Default value

*srvfmA4*

#### See also:

- ConvertPageFormatToPageSize<sup>(143)</sup>
- ConvertPageSizeToPageFormat<sup>(143)</sup>

Specifies a page height.

**property** PageHeight: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see TSVRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

#### Default value:

297 (corresponds to A4 in portrait Orientation<sup>(138)</sup>)

#### See also:

- PageWidth<sup>(141)</sup>
- PageFormat<sup>(139)</sup>
- ViewProperty<sup>(69)</sup>.PageHeight<sup>(162)</sup>

Specifies the index of the first page where page numbers are drawn, from 1.

**property** PageNoFirst: Integer;

#### Default value:

1 (all pages have page numbers, if PageNoVisible<sup>(140)</sup>=*True*).

Specifies the font for drawing page numbers.

#### type

TSRVPageNoFont = **class** (TFont)

**property** PageNoFont: TSRVPageNoFont;

#### Default value:

'Arial', 8

#### See also

- PageNoVisible <sup>(140)</sup>

Specifies the number of the first page.

**property** PageNoFromNumber: Integer;

This is the starting value for page numbering.

It is used:

- for drawing page numbers, if PageNoVisible <sup>(140)</sup> = *True*
- for "page number" fields (TRVPageNumberItemInfo items)

**Default value:**

1

Specifies a horizontal alignment of page numbers.

**type**

TSRVHAlign = (srvhaLeft, srvhaCenter, srvhaRight);

**property** PageNoHAlign: TSRVHAlign;

If MirrorMargins <sup>(138)</sup> = *True*, this property defines the alignment for odd pages; for even pages, left and right alignments are exchanged.

**Default value:**

*srvhaCenter*

**See also**

- PageNoVisible <sup>(140)</sup>
- PageNoVAlign <sup>(140)</sup>

Specifies a vertical position of page numbers.

**type**

TSRVVAlign = (srvvaUp, srvvaDown);

**property** PageNoVAlign: TSRVVAlign;

**Default value:**

*srvvaDown* (bottom of pages)

**See also**

- PageNoVisible <sup>(140)</sup>
- PageNoHAlign <sup>(140)</sup>

Shows/hides page numbers.

**property** PageNoVisible: Boolean;

This is a simple option to display page numbering. These page numbers are not saved to RTF, RVF or DocX.

The alternative (recommended) option is inserting "page number field (TRVPageNumberItemInfo item) in a header or a footer <sup>(66)</sup>.



**Default value:***True***See also**

- PageNoVAlign<sup>(140)</sup>
- PageNoHAlign<sup>(140)</sup>
- PageNoFirst<sup>(139)</sup>
- PageNoFromNumber<sup>(140)</sup>

Disables/enables page-view mode.

**property** PageViewMode: Boolean;

If this property is *False*, documents are displayed without separation on pages. PageHeight<sup>(139)</sup> is calculated to fit DocumentHeight of RichViewEdit component.

This property may be used when implementing custom view modes.

This property must be *True* in a read mode<sup>(151)</sup>.

**Default value:***True*

Specifies a page width.

**property** PageWidth: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

**Default value:**

210 (corresponds to A4 in portrait Orientation<sup>(138)</sup>)

**See also:**

- PageHeight<sup>(139)</sup>
- PageFormat<sup>(139)</sup>
- ViewProperty<sup>(69)</sup>.PageWidth<sup>(162)</sup>

This pen is used for drawing a rectangle around the printable area on the page.

**type**

```
TSRVPAPen = class (TPen)
```

**property** PrintableAreaPen : TSRVPAPen;

Printer cannot print beyond the area shown on this rectangle, see WinAPI function GetDeviceCaps, PHYSICAL\*\*\* constants.

This property is similar to the property of TRVPrintPreview of the same name.

**Default value:**

Color=**clRed**, Style=*psClear* (rectangle is invisible)

Specifies a right margin of pages.

**property** RightMargin: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

**Default value:**

15

Specifies whether special headers and footers are used for the first page.

**property** TitlePage: Boolean;

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetIntPropertyEd<sup>(97)</sup>.

A header for the first page is stored in SRichViewEdit.Subdocuments<sup>(66)</sup> [srvhftFirstPageHeader].

A footer for the first page is stored in SRichViewEdit.Subdocuments<sup>(66)</sup> [srvhftFirstPageFooter].

TitlePage turns on/off displaying headers and footers for the first page, but does not affect saving: headers and footers are loaded and saved regardless of this property.

Headers are shown only if HeaderVisible<sup>(137)</sup>=True, footers are shown only if FooterVisible<sup>(136)</sup>=True.

**Default value:**

False

**See also:**

- FacingPages<sup>(136)</sup>

Specifies a top margin of pages.

**property** TopMargin: TRVLength;

This value is measured in UnitsProgram<sup>(68)</sup> (mm by default).

This property can be changed as an editing (undoable) operation, see TSRichViewEdit<sup>(40)</sup>.SetFloatPropertyEd<sup>(97)</sup>.

**Default value:**

25

Specifies whether to use custom Z-order when drawing pages.

**property** UsePageOrders: Boolean;

Assigning True to this property calls CalculateAllPagePositions<sup>(72)</sup>, if TSRichViewEdit.ViewProperty<sup>(69)</sup>.FreePosPages<sup>(158)</sup>=True.

**Default value:**

False

#### 1.3.1.4.4.2 Methods

##### In TSRVPageProperty

ConvertPageSizeToPageFormat<sup>(143)</sup>  
 ConvertPageFormatToPageSize<sup>(143)</sup>

Converts **PageFormat** to **PaperWidth** and **PaperHeight**, in portrait orientation.

**procedure** ConvertPageFormatToPageSize(PageFormat: TSRVPageFormat<sup>(278)</sup>;  
 UnitsPaper: TRVUnits; **var** PaperWidth, PaperHeight: TRVLength);

**PaperWidth** and **PaperHeight** are measured in **UnitsPaper** ("pixels" assume 96 dpi).

If **PageFormat**=*srvfmCustom*, PageWidth<sup>(141)</sup> and PageHeight<sup>(139)</sup> are returned (always in landscape orientation)

Returns a page format by the specified values of **PaperWidth** and **PaperHeight**.

**function** ConvertPageSizeToPageFormat(UnitsPaper: TRVUnits;  
 PaperWidth, PaperHeight: TRVLength): TSRVPageFormat<sup>(278)</sup>;

**PaperWidth** and **PaperHeight** must be specified in portrait orientation.

**PaperWidth** and **PaperHeight** are measured in **UnitsPaper** ("pixels" assume 96 dpi).

If a standard format of this size cannot be found, *srvfmCustom* is returned.

#### 1.3.1.4.5 TSRVPagePositionProperty

TSRVPagePositionProperty is the class for TSRichViewEdit<sup>(40)</sup>.PagePosProperty<sup>(58)</sup> property  
**Unit** SclRView.

##### Syntax

TSRVPagePositionProperty = **class** (TPersistent)

##### Hierarchy

TObject

TPersistent

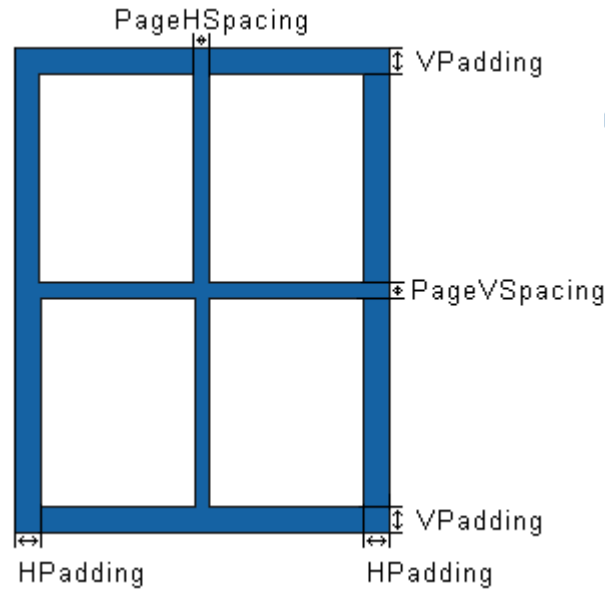
##### Properties

This class contains properties controlling arrangement of pages in TSRichViewEdit<sup>(40)</sup>, if  
 ViewProperty<sup>(69)</sup>.FreePosPage<sup>(158)</sup> = *False*.

HPadding and VPadding<sup>(146)</sup> define distances between sides of a scrollable area and outermost  
 pages.

PageHSpacing and PageVSpacing<sup>(147)</sup> define distances between pages.

You can remove some of these distances using HiddenDistances<sup>(145)</sup> property.



You can align pages (columns of pages) horizontally using `AlignPageH` <sup>(144)</sup> property.

You can align pages (rows of pages) vertically using `AlignPageV` <sup>(145)</sup> property.

By default, each column have as many pages as a page width and `ZoomPercent` <sup>(168)</sup> allow. You can limit it using `MaxPageColCount` <sup>(147)</sup> property.

#### 1.3.1.4.5.1 Properties

##### In `TSRVPagePositionProperty`

- `AlignPageH` <sup>(144)</sup>
- `AlignPageV` <sup>(145)</sup>
- `HiddenDistances` <sup>(145)</sup>
- `HPadding` <sup>(146)</sup>
- `MaxPageColCount` <sup>(147)</sup>
- `PageHSpacing` <sup>(147)</sup>
- `PageVSpacing` <sup>(147)</sup>
- `VPadding` <sup>(146)</sup>

Specifies the horizontal alignment of pages.

##### type

```
TSRVAlignPageH =
    (srvaphLeft, srvaphRight, srvaphCenter);
```

**property** `AlignPageH` : `TSRVAlignPageH`;

A minimum distance from the left and the right sides of the scrollable area to a page is specified in `HPadding` <sup>(146)</sup>. If a page width is smaller than the rest of space, pages are aligned according to the value of this property (if there are multiple columns of pages, columns as a whole are aligned according to this property).

If `TSRichViewEdit(40).PageProperty(58).AutoWidth(135)=True`, the page occupies all available window width (excluding the double `HPadding(146)`), so this alignment makes sense only if `TSRichViewEdit(40).PageProperty(58).AutoWidth(135)=False`.

if `TSRichViewEdit(40).BiDiMode(48) = rvbdRightToLeft`, left and right values of this property are exchanged.

Call `TSRichViewEdit(40).Format(82)` after changing value of this property.

#### Default value:

*srvaphCenter*

Specifies the vertical alignment of pages.

#### type

```
TSRVAlignPageV = (srvapvTop, srvapvCenter);
```

**property** `AlignPageV : TSRVAlignPageV;`

A minimum distance from the top and the bottom sides of the scrollable area to a page is specified in `VPadding(146)`. If a page height is smaller than the rest of space, pages are aligned according to the value of this property (if there are multiple rows of pages, rows as a whole are aligned according to this property).

Call `TSRichViewEdit(40).Format(82)` after changing value of this property.

Allows showing/hiding distances between pages.

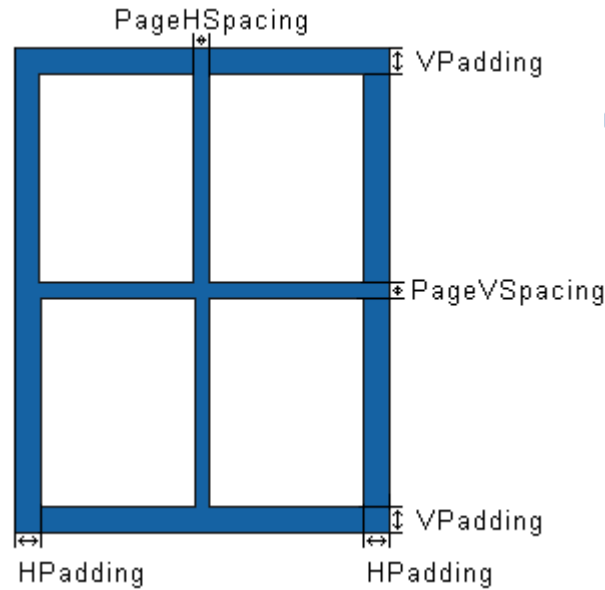
#### type

```
TSRVPagePositionDistance = (
    srvppdHPadding, srvppdVPadding,
    srvppdPageHSpacing, srvppdPageVSpacing);
TSRVPagePositionDistances = set of TSRVPagePositionDistance;
```

**property** `HiddenDistances: TSRVPagePositionDistances;`

Include values in this property to hide the corresponding distance in `TSRichViewEdit(40)`.

| Value                     | Distance                                  |
|---------------------------|---|
| <i>srvppdHPadding</i>     | <code>HPadding<sup>(146)</sup></code>     |
| <i>srvppdVPadding</i>     | <code>VPadding<sup>(146)</sup></code>     |
| <i>srvppdPageHSpacing</i> | <code>PageHSpacing<sup>(147)</sup></code> |
| <i>srvppdPageVSpacing</i> | <code>PageVSpacing<sup>(147)</sup></code> |



Call `TSRichViewEdit(40).Format(82)` after changing value of this property.

**Default value:**

[]

Minimal distance between outermost pages and sides of the editor's scrollable area, horizontal and vertical.

**property** `HPadding`: `TRVPixel96Length`;

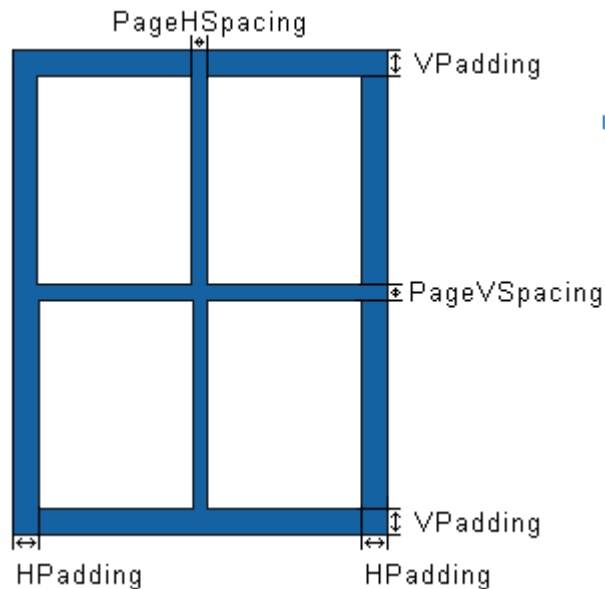
**property** `VPadding`: `TRVPixel96Length`;

`HPadding` is a minimal possible distance from the left/right side of the scrollable area to a page.

`VPadding` is a minimal possible distance from the top/bottom side of the scrollable area to a page.

These values are measured in pixels in 96 DPI (i.e. 1 pixel = 1/96 of an inch). When displaying on the screen, the actual distance is recalculated according to the screen DPI. Zooming does not affect the result.

To remove these distances, it's not necessary to assign 0 to these properties. Instead, you can include `srvppdHPadding` and/or `srvppdVPadding` in `HiddenDistances(145)`.



Call `TSRichViewEdit(40).Format(82)` after changing values of these properties.

**Default value:**

15

Specifies the maximum count of columns for page arrangement.

**property** `MaxPageColCount: Integer;`

If zooming<sup>(168)</sup> allows, the editor tries to display several pages in a row. This value defines the maximum count of pages in a row (0 means unlimited).

This property is ignored in a read mode<sup>(151)</sup>.

**Default value:**

0

Distance between pages, horizontal and vertical.

**property** `PageHSpacing: TRVPixel96Length;`

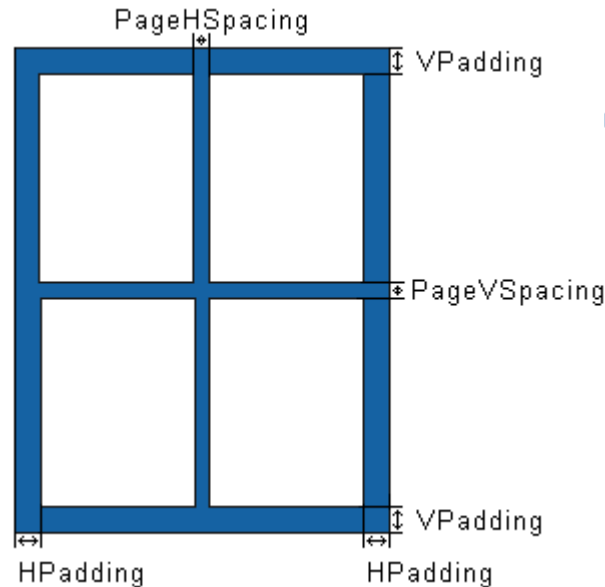
**property** `PageVSpacing: TRVPixel96Length;`

`PageHSpacing` is a distance between columns of pages.

`PageVSpacing` is a distance between rows of pages.

These values are measured in pixels in 96 DPI (i.e. 1 pixel = 1/96 of an inch). When displaying on the screen, the actual distance is recalculated according to the screen DPI. Zooming does not affect the result.

To remove these distances, it's not necessary to assign 0 to these properties. Instead, you can include `srvppdPageHSpacing` and/or `srvppdPageVSpacing` in `HiddenDistances(145)`.



Call `TSRichViewEdit(40).Format(82)` after changing values of these properties.

**Default value:**

12

#### 1.3.1.4.6 TSRVPropertyTBH

TSRVPropertyTBH is the class for `MenuHorizontal(55)` property of `TSRichViewEdit(40)`

**Unit** SclRView.

**Syntax**

```
TSRVPropertyTBH = class (TSRVCustomPropertyTB(127))
```

#### Hierarchy

*TObject*

*TPersistent*

*TSRVCustomPropertyTB<sup>(127)</sup>*

#### Description

This class describes properties of "horizontal menu" (toolbar in the area of horizontal scrollbar).

For example, this toolbar can be used to define viewing modes ("Web mode", "Normal mode", "Print mode").

This toolbar can be placed to the left or to the right of the scrollbar, depending on `Align(149)` property. It is shown only if `Visible(149) = True`.

Since this is a toolbar, not a tool window, all properties related to a hint area are not available. A height of buttons is calculated automatically.

Buttons are defined in `MenuHButtons(55)` property of `TSRichViewEdit(40)`.



When the user moves the mouse to a button, `TSRichViewEdit.OnHMenuEnterButton` <sup>(109)</sup> event occurs. When a button is clicked, `TSRichViewEdit.OnHMenuClickButton` <sup>(109)</sup> event occurs.

**See also:**

- `TSRVPropertyTBV` <sup>(149)</sup>

### 1.3.1.4.6.1 Properties

#### In `TSRVPropertyTBH`

- `Align` <sup>(149)</sup>
- `Visible` <sup>(149)</sup>

#### Derived from `TSRVCustomPropertyTB` <sup>(127)</sup>

- `BorderWidth` <sup>(128)</sup>
- `ButtonHeight` <sup>(128)</sup>
- `Buttons` <sup>(129)</sup>
- `ButtonWidth` <sup>(129)</sup>
- `Color` <sup>(129)</sup>
- `ColorDisable` <sup>(129)</sup>
- `ColorDown` <sup>(130)</sup>
- `ColorSelect` <sup>(130)</sup>
- `ImageList` <sup>(130)</sup>
- `PenFrame` <sup>(130)</sup>
- `ScaleImagesForDPI` <sup>(131)</sup>
- `Spacer` <sup>(131)</sup>

Defines the position of the toolbar relative to the horizontal scrollbar.

**type**

```
TSRVTBAlignH = (srvtbaLeft, srvtbaRight);
```

**property** `Align` : `TSRVTBAlignH`;

**Default value:**

*srvtbaLeft* (to the left of the scrollbar)

Specifies whether the toolbar is visible.

**property** `Visible`: `Boolean`;

**Default value:**

*False* (the toolbar is hidden)

### 1.3.1.4.7 `TSRVPropertyTBV`

`TSRVPropertyTBV` is the class for `MenuVertical` <sup>(56)</sup> property of `TSRichViewEdit` <sup>(40)</sup>

**Unit** `SclRView`.

**Syntax**

```
TSRVPropertyTBV = class (TSRVCustomPropertyTB (127))
```

## Hierarchy

*TObject*

*TPersistent*

*TSRVCustomPropertyTB* <sup>(127)</sup>

## Description

This class describes properties of "vertical menu" (toolbar in the area of vertical scrollbar).

For example, this toolbar can be used for searching in the document.

This toolbar can be placed above or below the scrollbar, depending on `Align` <sup>(150)</sup> property. It is shown only if `Visible` <sup>(151)</sup> = `True`.

Since this is a toolbar, not a tool window, all properties related to a hint area are not available. A width of buttons is calculated automatically.

Buttons are defined in `MenuVButtons` <sup>(56)</sup> property of `TSRichViewEdit` <sup>(40)</sup>.

When the user moves the mouse to a button, `TSRichViewEdit.OnVMenuEnterButton` <sup>(122)</sup> event occurs. When a button is clicked, `TSRichViewEdit.OnVMenuClickButton` <sup>(121)</sup> event occurs.

### See also:

- `TSRVPropertyTBVH` <sup>(148)</sup>

### 1.3.1.4.7.1 Properties

## In TSRVPropertyTBV

■ `Align` <sup>(150)</sup>

■ `Visible` <sup>(151)</sup>

## Derived from TSRVCustomPropertyTB <sup>(127)</sup>

■ `BorderWidth` <sup>(128)</sup>

■ `ButtonHeight` <sup>(128)</sup>

Buttons <sup>(129)</sup>

■ `ButtonWidth` <sup>(129)</sup>

■ `Color` <sup>(129)</sup>

■ `ColorDisable` <sup>(129)</sup>

■ `ColorDown` <sup>(130)</sup>

■ `ColorSelect` <sup>(130)</sup>

■ `ImageList` <sup>(130)</sup>

■ `PenFrame` <sup>(130)</sup>

■ `ScaleImagesForDPI` <sup>(131)</sup>

■ `Spacer` <sup>(131)</sup>

Defines the position of the toolbar relative to the vertical scrollbar.

### type

```
TSRVTBAlignV = (srvtbaTop, srvtbaBottom);
```

**property** `Align` : `TSRVTBAlignV`;

**Default value:**

*srvtbaTop* (above the the scrollbar)

Specifies whether the toolbar is visible.

**property** Visible: Boolean;

**Default value:**

*False* (the toolbar is hidden)

### 1.3.1.4.8 TSRVReadModeProperty

TSRVReadModeProperty is the class for ReadModeProperty<sup>(59)</sup> property of TSRichViewEdit<sup>(40)</sup>

**Unit** SclRView.

**Syntax**

```
TSRVReadModeProperty = class (TPersistent)
```

**Hierarchy**

*TObject*

*TPersistent*

**Description**

This class defines properties for displaying pages in a way convenient for reading.

It is applied when you assign Active<sup>(152)</sup> = *True*. Before activating a read mode, assign the parent editor's PageProperty<sup>(141)</sup>.PageViewMode<sup>(141)</sup> = *True*.

The results depend on the parent editor's PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup>.

**Side to side mode**

If AutoWidth<sup>(135)</sup> = *False*, pages are displayed at their normal size, i.e. pages look identically to pages displayed without a read mode.

However, there are the following differences:

- all pages are arranged in a horizontal row (the parent editor's PagePosProperty<sup>(58)</sup>.MaxPageColCount<sup>(147)</sup> is ignored)
- the horizontal scroll bar scrolls by whole pages; the vertical scroll bar is disabled; so all visible pages are shown completely
- the parent editor's ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> becomes read-only; it is auto-calculated to show at least MinPageColCount<sup>(152)</sup> pages completely.
- no more than MaxPageColCount<sup>(152)</sup> pages are displayed at the same time.

**Normal read mode**

If AutoWidth<sup>(135)</sup> = *True*, each page occupies all available space in the editor (excluding the parent editor's PagePosProperty<sup>(58)</sup>.HPadding and VPadding<sup>(146)</sup>).

Additionally:

- all pages are arranged in a horizontal row; only one page is visible (all column count properties are ignored)

- the horizontal scroll bar scrolls by whole pages; the vertical scroll bar is disabled; so a visible page is shown completely
- headers and footers<sup>(66)</sup> are not displayed

### Visual properties

In a read mode, special page transition effects are available. They are defined in PageFlipEffect<sup>(153)</sup> property.

You can hide a page background picture<sup>(152)</sup> and define a special page background color<sup>(153)</sup>.

#### 1.3.1.4.8.1 Properties

### In TSRVReadModeProperty

- Active<sup>(152)</sup>
- HidePageBackgroundImage<sup>(152)</sup>
- MaxPageColCount<sup>(152)</sup>
- MinPageColCount<sup>(152)</sup>
- PageColor<sup>(153)</sup>
- PageFlipEffect<sup>(153)</sup>

Activates or deactivates a read mode.

**property** Active: Boolean;

Before assigning True to this property, assign the parent editor's PageProperty<sup>(141)</sup>.PageViewMode<sup>(141)</sup> = True.

**Default value:**

False

Allows hiding a background images of pages in a read mode.

**property** HidePageBackgroundImage: Boolean;

If **HidePageBackgroundImage** = True, the editor's RichViewEdit<sup>(60)</sup>.BackgroundBitmap is not shown when a read mode is active<sup>(152)</sup>.

This property is not applied when printing pages from a read mode.

**See also**

- PageColor<sup>(153)</sup>

**Default value:**

c/None

Defines a number of visible pages in a read mode.

**property** MinPageColCount: Integer;

**property** MaxPageColCount: Integer;

These properties are applied when a read mode is active<sup>(152)</sup>.

**MinPageColCount** defines the minimal count of visible pages.

If **MaxPageColCount** > 0, it defines the maximal allowed count of visible pages.

For example, if you want to implement a two-page "book" view, assign 2 to the both properties.

**Default value:**

- MinPageColCount: 1
- MaxPageColCount: 0

Allows overriding background color of pages in a read mode.

**property** PageColor: TColor;

If **PageColor** <> *c/None*, it is used as a background page color when a read mode is active <sup>(152)</sup>.

This property has higher priority than the editor's BackgroundProperty <sup>(48)</sup>.  
.GlobalPageBackgroundColor <sup>(125)</sup>. However, it is not applied when printing pages from a read mode.

**See also**

- HidePageBackgroundImage <sup>(152)</sup>

**Default value:**

*c/None*

Defines a page transition animation in a read mode.

**type**

```
TSRVPageFlipEffect = (srvpfeNone, srvpfeScroll,  
    srvpfeStack, srvpfePopUp, srvpfeFade);
```

**property** PageFlipEffect: TSRVPageFlipEffect;

| Value               | Meaning   |
|---------------------|---|
| <i>srvpfeNone</i>   | No animation. Fast page transition.                             |
| <i>srvpfeScroll</i> | Pages slide from/to outside                                     |
| <i>srvpfeStack</i>  | Pages slide as if they are stacked in outermost visible columns |
| <i>srvpfePopUp</i>  | Like <i>srvpfeStack</i> , but page pop-up effect is amplified.  |
| <i>srvpfeFade</i>   | Pages gradually appear on new places (requires Delphi 2009+).   |

Page transition animation is used:

- when pages are scrolled using a mouse wheel
- when pages are scrolled using a horizontal scrollbar by 1 position

**Default value:**

*srvpfeStack*

### 1.3.1.4.9 TSRVViewProperty

TSRVViewProperty is the class for TSRichViewEdit<sup>(40)</sup>.ViewProperty<sup>(69)</sup> property.

**Unit** SclRView.

#### Syntax

```
TSRVViewProperty = class (TPersistent)
```

#### Hierarchy

TObject

TPersistent

#### Description

##### Free page positioning mode

Free page positioning mode is activated in FreePosPage<sup>(158)</sup> property. In this mode, you can define position, zooming and Z-order for specific pages.

##### Zooming and page arrangement

The current zooming is defined in ZoomMode<sup>(166)</sup> and ZoomPercent<sup>(168)</sup> properties.

In a preview mode (ViewMode<sup>(165)</sup>=*srvvmPreviewMode*), clicking switches these properties between ZoomModeIN<sup>(167)</sup>, ZoomPercentIN<sup>(169)</sup> and ZoomModeOUT<sup>(167)</sup>, ZoomPercentOUT<sup>(169)</sup>.

When switching back to an editing mode (ViewMode<sup>(165)</sup>=*srvvmEditMode*), they are set to ZoomModeEdit<sup>(167)</sup>, ZoomPercentEdit<sup>(169)</sup>.

The user can change zooming:

- using a mouse wheel, if MouseWheelZooming<sup>(162)</sup>=*True*.
- using a special panel, if ZoomPanelVisible<sup>(168)</sup>=*True*.

ZoomPercent<sup>(168)</sup> can be assigned in the range ZoomMin..ZoomMax<sup>(166)</sup>.

If zooming allows, pages may be arranged in columns, according to MaxPageColCount<sup>(147)</sup> property.

##### Table icons

A table "icon" (a rectangle at the top left corner) is displayed when the user moves the mouse pointer inside a table (TRVTableItemInfo), if UseTableIcons<sup>(164)</sup>=*True*.

Even when the user moves the mouse pointer outside a table, this icon is still visible for TableIconDelay<sup>(163)</sup>.

When the user right-clicks this icon, TableIconPopupMenu<sup>(163)</sup> is displayed. Besides, TSRichViewEdit.OnTableIconClick<sup>(121)</sup> occurs on clicking.

The table for the displayed icon is returned by TSRichViewEdit.GetTableIconItem<sup>(89)</sup>. A document where this table is located is returned by TSRichViewEdit.GetTableIconRVData<sup>(89)</sup>.

##### Rectangle around the document area

You can display a rectangle between the document area and margins, if you set MarginsRectVisible<sup>(162)</sup>=*True*. This rectangle is drawn using MarginsPen<sup>(162)</sup>.

##### Hints on vertical scrolling

A hint appear on scrolling, if `ShowScrollHint`<sup>(163)</sup> = `True`. This hint contains two lines describing the page at the scrolling position:

- `HintPrefixText`<sup>(160)</sup> followed by the page number
- text from the beginning of page

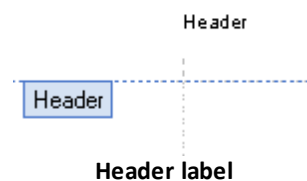
This text is drawn with `HintFont`<sup>(160)</sup>.

## Header and footer

When the header is being edited, a line is shown above the main document. This line uses `HeaderPen`<sup>(159)</sup>. Additionally, a label is displayed.

## VCL Themes

`UseVCLThemes`<sup>(165)</sup> allows using colors of the current VCL theme.



This label has text `Texts`<sup>(163)</sup>.`NormalPageHeader`, font `HeaderTitleFont`<sup>(159)</sup>, color `HeaderTitleColor`<sup>(159)</sup>, border color `HeaderPen`<sup>(159)</sup>.`Color`. If `Texts`<sup>(163)</sup>.`NormalPageHeader` is empty, this label is not displayed.

A footers and the main document have similar properties. For the main document, a line and a label are invisible by default.

When a footer or a header is being edited, the main document can be shaded, if `MainDocumentShade`<sup>(160)</sup> = `True`. When the main document is edited, a header and a footer can be shaded if `HeaderFooterShade`<sup>(158)</sup> = `True`.

## Page size

The page size, *as it is shown on the screen*, is returned in `PageWidth`<sup>(162)</sup> and `PageHeight`<sup>(162)</sup> properties.

They are measured in `SRichViewEdit`'s `UnitsProgram`<sup>(68)</sup> and do not depend on scaling. The same values in pixels are returned in `PageWidth100Pix`<sup>(58)</sup> and `PageHeight100Pix`<sup>(57)</sup> properties of `SRichViewEdit`.

However, these properties depend on viewing modes. If `PageProperty`<sup>(58)</sup>.`AutoWidth`<sup>(135)</sup> = `False` and `PageProperty`<sup>(58)</sup>.`PageViewMode`<sup>(141)</sup> = `True`, these properties return the same values as `PageProperty`<sup>(58)</sup>'s `PageWidth`<sup>(141)</sup> and `PageHeight`<sup>(139)</sup>. When `AutoWidth` or `PageViewMode` are changed, the screen view does not correspond to the paper size, and these properties become different.

### 1.3.1.4.9.1 Properties

#### In `TSRVViewProperty`

- `AutoVScrollBarPosition`<sup>(156)</sup>
- `DarkMode`<sup>(157)</sup>

- DarkModeUI <sup>157</sup>
- FooterPen <sup>157</sup>
- FooterTitleColor <sup>157</sup>
- FooterTitleFont <sup>158</sup>
- FreePosPage <sup>158</sup>
- HeaderFooterShade <sup>158</sup>
- HeaderPen <sup>159</sup>
- HeaderTitleColor <sup>159</sup>
- HeaderTitleFont <sup>159</sup>
- HintFont <sup>160</sup>
- HintPrefixText <sup>160</sup>
- IconStyle <sup>160</sup>
- MainDocumentShade <sup>160</sup>
- MainPen <sup>161</sup>
- MainTitleColor <sup>161</sup>
- MainTitleFont <sup>161</sup>
- MarginsPen <sup>162</sup>
- MarginsRectVisible <sup>162</sup>
- MouseWheelZoom <sup>162</sup>
- ▶ PageHeight <sup>162</sup>
- ▶ PageWidth <sup>162</sup>
- ShowScrollHint <sup>163</sup>
- TableIconDelay <sup>163</sup>
- TableIconPopupMenu <sup>163</sup>
- Texts <sup>163</sup>
- UseTableIcons <sup>164</sup>
- UseVCLThemes <sup>165</sup>
- ViewMode <sup>165</sup>
- WheelStep <sup>165</sup>
- WorkArea <sup>165</sup>
- ZoomFont <sup>166</sup>
- ZoomMenu <sup>166</sup>
- ZoomMin, ZoomMax <sup>166</sup>
- ZoomMode <sup>166</sup>
- ZoomModeEdit <sup>167</sup>
- ZoomModeIN <sup>167</sup>
- ZoomModeOUT <sup>167</sup>
- ZoomPanelFont <sup>167</sup>
- ZoomPanelVisible <sup>168</sup>
- ZoomPercent <sup>168</sup>
- ZoomPercentEdit <sup>169</sup>
- ZoomPercentIN <sup>169</sup>
- ZoomPercentOUT <sup>169</sup>

Specifies where a vertical scrollbar is located if a right-to-left document is loaded

**property** AutoVScrollBarPosition: Boolean;

By default, if BiDiMode <sup>48</sup> = *rvbdRightToLeft*, a vertical scrollbar is moved to the left side.



If you assign *False* to this property, this scrollbar will always be at the right side.

**Default value:**

*True*

Inverts luminance of all colors in document.

**property** `DarkMode: Boolean;`

If *True*, luminance of all colors in the document is inverted. Dark colors become light colors, black becomes white, white becomes black.

Warning: be careful with this property: it affects both drawing on the screen and printing. Black backgrounds causes overuse of printer ink.

This property does not affect to drawing controls around pages, see `DarkModeUI` <sup>(157)</sup>.

**Default value:**

*False*

Inverts luminance of all colors in user interface

**property** `DarkModeUI: Boolean;`

If *True*, luminance of all colors in controls around pages is inverted.

This property does not affect drawing on pages, see `DarkMode` <sup>(157)</sup>.

Scroll bars use “dark mode” drawing only if you change `ScrollBarControlStyle` <sup>(65)</sup> to *svcsSimple*.

**Default value:**

*False*

A pen for drawing a line between the main document and the footer, when a footer (`TSRichViewEdit.RVFooter` <sup>(62)</sup>) is being edited.

**type**

```
TSRVHeaderFooterPen = class (TPen);
```

**property** `FooterPen: TSRVHeaderFooterPen;`

The color of this pen is also used for drawing border around header label, see `Texts` <sup>(163)</sup>.

In Delphi XE2 or newer, if a system color is specified for `FooterPen.Color`, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

1, *psDot*, *\$00CA784B*

**See also:**

- `HeaderPen` <sup>(159)</sup>
- `MainPen` <sup>(161)</sup>

A background color of label displayed when a footer (`TSRichViewEdit.RVFooter` <sup>(62)</sup>) is being edited.

**property** `FooterTitleColor: TColor;`

A border around this label is drawn using `FooterPen` <sup>(157)</sup>.`Color`.

A text of this label (Texts <sup>(163)</sup>.NormalPageFooter, FirstPageFooter, EvenPageFooter, OddPageFooter) is drawn using FooterTitleFont <sup>(158)</sup>.

In Delphi XE2 or newer, if a system color is specified for this property, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

\$00F3E2D5

**See also:**

- HeaderTitleColor <sup>(159)</sup>
- MainTitleColor <sup>(161)</sup>

A font of label title displayed when a footer (TSRichViewEdit.RVFooter <sup>(62)</sup>) is being edited.

**type**

```
TSRVHeaderFooterFont = class (TFont);
```

**property** FooterTitleFont : TSRVHeaderFooterFont;

This font is used to draw a footer title (Texts <sup>(163)</sup>.NormalPageFooter, FirstPageFooter, EvenPageFooter, OddPageFooter) on a background filled with FooterTitleColor <sup>(157)</sup>.

In Delphi XE2 or newer, if a system color is specified for FooterTitleFont.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

'Tahoma', 8, [], \$009A480B

**See also:**

- HeaderTitleFont <sup>(159)</sup>
- MainTitleFont <sup>(161)</sup>

Enables/disables free page positioning.

**property** FreePosPage: Boolean;

If *False*, positions, scaling and Z-order of all pages are calculated automatically.

If *True*, they are requested in OnGetPagePos <sup>(108)</sup> event.

**Default value:**

*False*

Determines whether a header and a footer are shaded when the main document is being edited.

**property** HeaderFooterShade: Boolean;

**Default value:**

*True*

**See also**

- MainDocumentShade <sup>(160)</sup>

A pen for drawing a line between the main document and the header, when a header (TSRichViewEdit.RVHeader<sup>64</sup>) is being edited.

**type**

```
TSRVHeaderFooterPen = class (TPen);
```

**property** HeaderPen: TSRVHeaderFooterPen;

The color of this pen is also used for drawing border around header label, see Texts<sup>163</sup>.

In Delphi XE2 or newer, if a system color is specified for HeaderPen.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

1, *psDot*, \$00CA784B

**See also:**

- FooterPen<sup>157</sup>
- MainPen<sup>161</sup>

A background color of label displayed when a header (TSRichViewEdit.RVHeader<sup>64</sup>) is being edited.

**property** HeaderTitleColor: TColor;

A border around this label is drawn using HeaderPen<sup>159</sup>.Color.

A text of this label (Texts<sup>163</sup>.NormalPageHeader, FirstPageHeader, EvenPageHeader, OddPageHeader) is drawn using HeaderTitleFont<sup>159</sup>.

In Delphi XE2 or newer, if a system color is specified for this property, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

\$00F3E2D5

**See also:**

- FooterTitleColor<sup>157</sup>
- MainTitleColor<sup>161</sup>

A font of label title displayed when a header (TSRichViewEdit.RVHeader<sup>64</sup>) is being edited.

**type**

```
TSRVHeaderFooterFont = class (TFont);
```

**property** HeaderTitleFont : TSRVHeaderFooterFont;

This font is used to draw a header title (Texts<sup>163</sup>.NormalPageHeader, FirstPageHeader, EvenPageHeader, OddPageHeader) on a background filled with HeaderTitleColor<sup>159</sup>.

In Delphi XE2 or newer, if a system color is specified for HeaderTitleFont.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

'Tahoma', 8, [], \$009A480B

**See also:**

- FooterTitleFont<sup>158</sup>

## • MainTitleFont<sup>(161)</sup>

A font for hints appearing on vertical scrolling.

### type

```
TSRVHintFont = class (TFont);
```

**property** HintFont: TSRVHintFont;

This hint appears if ShowScrollHint<sup>(163)</sup> = *True*.

In Delphi XE2 or newer, if a system color is specified for HintFont.Color, and custom styles are applied, then a corresponding style color is used instead.

### Default value:

'Arial', 10, clInfoText

Specifies the text for the first line of a popup hint appearing on vertical scrolling.

**property** HintPrefixText: TRVUnicodeString;

HintPrefixText is displayed before the page number.

This hint appears if ShowScrollHint<sup>(163)</sup> = *True*.

### Default value:

'Page '



Allows to choose an set of icons used in TSRichViewEdit<sup>(40)</sup> control.

### type

```
TSRVIconStyle = (srvisNormal, srvisFlat);
```

**property** IconStyle: TSRVIconStyle;

TSRichViewEdit allows choosing different icons for drawing in a zooming panel (if ZoomPanelVisible<sup>(168)</sup> = *True*)

| Value              | Meaning  |
|--------------------|--|
| <i>srvisNormal</i> | Standard icons<br>              |
| <i>srvisFlat</i>   | Simplified flat white icons<br> |

### Default value:

*srvisNormal*

Determines whether the main document is shaded when a header or a footer is being edited.

**property** MainDocumentShade: Boolean;

### Default value:

*False*

### See also

- HeaderFooterShade<sup>(158)</sup>

A pen for drawing lines between the main document and the header and the footer, when the main document (TSRichViewEdit.RichViewEdit<sup>(60)</sup>) is being edited.

**type**

```
TSRVMainPen = class (TPen);
```

**property** MainPen: TSRVMainPen;

The color of this pen is also used for drawing border around the main document label, see Texts<sup>(163)</sup>.

In Delphi XE2 or newer, if a system color is specified for MainPen.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

1, psClear, \$00CA784B (invisible line)

**See also:**

- HeaderPen<sup>(159)</sup>
- FooterPen<sup>(157)</sup>

A background color of a label displayed when the main document (TSRichViewEdit.RichViewEdit<sup>(60)</sup>) is being edited.

**property** MainTitleColor: TColor;

A border around this label is drawn using MainPen<sup>(161)</sup>.Color.

A text of this label (Texts<sup>(163)</sup>.MainDocument) is drawn using MainTitleFont<sup>(161)</sup>.

In Delphi XE2 or newer, if a system color is specified for this property, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

\$00F3E2D5

**See also:**

- HeaderTitleColor<sup>(159)</sup>
- FooterTitleColor<sup>(157)</sup>

A font of label title displayed when the main document (TSRichViewEdit.RichViewEdit<sup>(60)</sup>) is being edited.

**type**

```
TSRVHeaderFooterFont = class (TFont);
```

**property** MainTitleFont : TSRVHeaderFooterFont;

This font is used to draw Texts<sup>(163)</sup>.MainDocument on a background filled with MainTitleColor<sup>(161)</sup>.

In Delphi XE2 or newer, if a system color is specified for MainTitleFont.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

'Tahoma', 8, [], \$009A480B

**See also:**

- HeaderTitleFont<sup>(159)</sup>
- FooterTitleFont<sup>(158)</sup>

Specifies a pen properties for drawing rectangle around the document area.

**property** BoundTextPen: TSRVMarginsPen;

This rectangle shows margins. It is drawn only if MarginsRectVisible<sup>(162)</sup>=*True*.

In Delphi XE2 or newer, if a system color is specified for MarginsPen.Color, and custom styles are applied, then a corresponding style color is used instead.

**Default value:**

*psDot, clSilver*

Specifies whether the component draws a rectangle around the document area.

**property** MarginsRectVisible: Boolean;

This rectangle is drawn using MarginsPen<sup>(162)</sup>.

Specifies whether pages can be zoomed in/out using the mouse wheel (with pressed **Ctrl** key)

**property** MouseWheelZoom: Boolean;

If *True*, **Ctrl**+mouse wheel changes ZoomMode<sup>(166)</sup> to *rvzmCustom* and increases/decreases ZoomPercent<sup>(168)</sup>.

**Default value:**

*True*

**See also:**

- MouseWheelZoomMin, MouseWheelZoomMax, MouseWheelZoomSpeed global variables<sup>(281)</sup>

Returns the page height as it is shown on the screen.

**property** PageHeight: TRVLength;

This property is measured in SRichViewEdit's UnitsProgram<sup>(68)</sup>. The same value in pixels is returned in PageHeight100Pix<sup>(57)</sup> property of SRichViewEdit.

This property does not depend on scaling, but depends on viewing modes (PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup> and PageProperty<sup>(58)</sup>.PageViewMode<sup>(141)</sup>). This property is equal to PageProperty<sup>(58)</sup>.PageHeight<sup>(139)</sup> only if AutoWidth=*False* and PageViewMode=*True*.

Returns the page width as it is shown on the screen.

**property** PageWidth: TRVLength;

This property is measured in SRichViewEdit's UnitsProgram<sup>(68)</sup>. The same value in pixels is returned in PageWidth100Pix<sup>(58)</sup> property of SRichViewEdit.

This property does not depend on scaling, but depends on viewing modes (PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup> and PageProperty<sup>(58)</sup>.PageViewMode<sup>(141)</sup>). This property is equal to PageProperty<sup>(58)</sup>.PageWidth<sup>(141)</sup> only if AutoWidth=*False* and PageViewMode=*True*.

Specifies whether a popup hint appears on scrolling.

**property** ShowScrollHint: Boolean;

This hint displays on the first line: HintPrefixText<sup>(160)</sup>, then the page number.

On the second line, a text from the beginning of this page is displayed.

HintFont<sup>(160)</sup> is used.

**Default value:**

*True*

Specifies the period of time for which a table "icon" is still visible after the user moved the mouse pointer outside a table.

**property** TableIconDelay: LongWord;

A table "icon" (a rectangle at the top left corner) is displayed when the user moves the mouse pointer inside a table (TRVTableItemInfo), if UseTableIcons<sup>(164)</sup>=*True*.

Even when the user moves the mouse pointer outside a table, this icon is still visible for this period of time.

This value is measured in milliseconds.

**Default value:**

3000

**See also:**

- TSRichViewEdit.OnTableIconClick<sup>(121)</sup>
- TableIconPopupMenu<sup>(163)</sup>

A reference to the context menu of a table "icon".

**property** TableIconPopupMenu: TPopupMenu;

A table "icon" (a rectangle at the top left corner) is displayed when the user moves the mouse pointer inside a table (TRVTableItemInfo), if UseTableIcons<sup>(164)</sup>=*True*.

**See also:**

- TSRichViewEdit.OnTableIconClick<sup>(121)</sup>
- TableIconDelay<sup>(163)</sup>

A list of text strings displayed in TSRichViewEdit<sup>(40)</sup>.

**type**

TSRVTexts = **class** (TPersistent)

**property** Texts: TSRVTexts;

TSRVTexts has the properties listed in the table below.

| Property         | Meaning   | Default Value | Related Properties of<br>TSRVViewProperty                                 |
|------------------|---|---------------|---|
| NormalPageHeader | Title for the normal header, if<br>PageProperty <sup>(58)</sup> | 'Header'      | HeaderTitleFont <sup>(159)</sup> ,<br>HeaderTitleColor <sup>(159)</sup> , |

|                  |   |                     |  |
|------------------|---|---------------------|--|
|                  | .FacingPages <sup>136</sup> = <i>False</i>                |                     | HeaderPen <sup>159</sup> .Color  |
| NormalPageFooter | Title for the normal footer, if FacingPages= <i>False</i> | 'Footer'            | FooterTitleFont <sup>158</sup> ,<br>FooterTitleColor <sup>157</sup> ,<br>FooterPen <sup>157</sup> .Color |
| FirstPageHeader  | Title for the first page header                           | 'First Page Header' | HeaderTitleFont <sup>159</sup> ,<br>HeaderTitleColor <sup>159</sup> ,<br>HeaderPen <sup>159</sup> .Color |
| FirstPageFooter  | Title for the first page footer                           | 'First Page Footer' | FooterTitleFont <sup>158</sup> ,<br>FooterTitleColor <sup>157</sup> ,<br>FooterPen <sup>157</sup> .Color |
| EvenPageHeader   | Title for the even pages header                           | 'Even Page Header'  | HeaderTitleFont <sup>159</sup> ,<br>HeaderTitleColor <sup>159</sup> ,<br>HeaderPen <sup>159</sup> .Color |
| EvenPageFooter   | Title for the even pages footer                           | 'Even Page Footer'  | FooterTitleFont <sup>158</sup> ,<br>FooterTitleColor <sup>157</sup> ,<br>FooterPen <sup>157</sup> .Color |
| OddPageHeader    | Title for the normal header, if FacingPages= <i>True</i>  | 'Odd Page Header'   | HeaderTitleFont <sup>159</sup> ,<br>HeaderTitleColor <sup>159</sup> ,<br>HeaderPen <sup>159</sup> .Color |
| OddPageFooter    | Title for the normal footer, if FacingPages= <i>True</i>  | 'Odd Page Footer'   | FooterTitleFont <sup>158</sup> ,<br>FooterTitleColor <sup>157</sup> ,<br>FooterPen <sup>157</sup> .Color |
| MainDocument     | Title for the main document                               | "                   | MainTitleFont <sup>161</sup> ,<br>MainTitleColor <sup>161</sup> , MainPen <sup>161</sup> .Color          |

These strings are used to draw labels when editing a header, a footer, or a main document. If a string is empty, the corresponding label is not displayed.

Specifies whether table "icons" should be displayed for tables.

**property** UseTableIcons: Boolean;

A table "icon" (a rectangle at the top left corner) is displayed when the user moves the mouse pointer inside a table (TRVTableItemInfo), if UseTableIcons=*True*.

Even when the user moves the mouse pointer outside a table, this icon is still visible for TableIconDelay<sup>163</sup>.

When the user right-clicks this icon, TableIconPopupMenu<sup>163</sup> is displayed. Besides, TSRichViewEdit.OnTableIconClick<sup>121</sup> occurs on clicking.

The table for the displayed icon is returned by TSRichViewEdit.GetTableIconItem<sup>89</sup>. A document where this table is located is returned by TSRichViewEdit.GetTableIconRVData<sup>89</sup>.



**Default value:**

*True*

Allows using VCL theme colors (for Delphi XE2 and newer).

**property** UseVCLThemes: Boolean;

If *True*, VCL theme colors are used instead of system colors in documents. This property is ignored for Delphi versions prior to XE2.

This property does not affect user interface objects in the editor. They always use VCL themes, if available.

**Default value:**

*False*

Switches between editing and preview modes.

**type**

```
TSRVViewMode = (srvvmEditMode, srvvmPreviewMode);
```

**property** ViewMode: TSRVViewMode;

When this property is changed, `TSRichViewEdit.OnChangeViewModeAfter`<sup>(105)</sup> and `TSRichViewEdit.OnChangeViewModeBefore`<sup>(105)</sup> events occur.

**Default value:**

*srvvmEditMode*

Defines how much the document will be scrolled when the user turns a mouse wheel.

**property** WheelStep: Integer;

This property provides access to the property of the same name of `TSRichViewEdit`<sup>(40)</sup>.  
`.RichViewEdit`<sup>(60)</sup>.

**Default value:**

*3*

Restricts the region available for drawing page content.

**type**

```
TSRVWorkArea = (srvwaTextRect, srvwaPageRect);
```

**property** WorkArea : TSRVWorkArea;

| Value                | Meaning  |
|----------------------|--|
| <i>srvwaTextRect</i> | The output area is limited by the document area (nothing can be drawn on margins)  |
| <i>srvwaPageRect</i> | The output area is limited by the page area (nothing can be drawn beyond the page) |

**Default value:**

*srvwaPageRect*

A font for drawing text (zooming percent) in a zooming mode.

**type**

```
TSRVZoomPanelFont = (TFont)
```

**property** ZoomPanelFont: TSRVZoomPanelFont;

A zooming mode is initiated if `TSRichViewEdit.PageProperty(58).AutoWidth(135) = True`, by a mouse wheel<sup>(162)</sup> or a track bar on a zooming panel<sup>(168)</sup>.

In this mode, the editor content is shaded with white color, and a current zooming percent is displayed using this font.

The specified font size corresponds to 100% zoom. The text size is increased/decreased according to the zooming percent.

**Default value:**

'Tahoma', 20, clWindowText

A reference to the popup menu shown when the user clicks on the text area of a zooming panel.

**property** ZoomMenu: TPopupMenu;

**See also**

- ZoomPanelVisible<sup>(168)</sup>
- ZoomPanelFont<sup>(167)</sup>
- TableIconMenu<sup>(163)</sup>

The properties define the range of zooming.

**property** ZoomMin: Integer;

**property** ZoomMax: Integer;

ZoomPercent<sup>(168)</sup> can be set to a value in this range.

**Default value:**

20..500

The current zooming mode.

**property** ZoomMode: TRVZoomMode;

Full page / Page width / Custom. See also ZoomPercent<sup>(168)</sup>.

This value is switched between ZoomModeIN<sup>(167)</sup>, ZoomModeOUT<sup>(167)</sup>, ZoomModeEdit<sup>(167)</sup>.

When this property is changed, `TSRichViewEdit.OnZoomChanged(122)` occurs.

**Default value:**

*rvzmPageWidth*

This value is assigned to ZoomMode<sup>(166)</sup> when switching ViewMode<sup>(165)</sup> to *srvvmEditMode*.

**property** ZoomModeEdit: TRVZoomMode;

If this property is equal to *rvzmCustom*, the corresponding zooming is defined in ZoomPercentEdit<sup>(169)</sup>.

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value

*rvzmPageWidth*

This value is assigned to ZoomMode<sup>(166)</sup> when switching ViewMode<sup>(165)</sup> to *srvvmPreviewMode*, or in preview mode when the user clicks the page and the cursor looks like "-" magnifier.

**property** ZoomModeIN: TRVZoomMode;

In preview mode the mouse pointer looks like a magnifier. On clicking, ZoomMode<sup>(166)</sup> is switched between ZoomModeIN/ZoomModeOUT<sup>(167)</sup>.

If this property is equal to *rvzmCustom*, the corresponding zooming is defined in ZoomPercentIN<sup>(169)</sup>.

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value:

*rvzmFullPage*.

This value is assigned to ZoomMode<sup>(166)</sup> in preview mode when the user clicks the page and the cursor looks like "+" magnifier.

**property** ZoomModeOUT: TRVZoomMode;

In preview mode the mouse pointer looks like a magnifier. On clicking, ZoomMode<sup>(166)</sup> is switched between ZoomModeIN<sup>(167)</sup>/ZoomModeOUT.

If this property is equal to *rvzmCustom*, the corresponding zooming is defined in ZoomPercentOUT<sup>(169)</sup>.

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value:

*rvzmCustom*

A font for drawing text (zooming percent) in a zooming panel.

#### type

```
TSRVZoomPanelFont = (TFont)
```

**property** ZoomPanelFont: TSRVZoomPanelFont;

A zooming panel is shown if ZoomPanelVisible<sup>(168)</sup> = *True*.

#### Default value:

'Tahoma', 8, clWindowText

Disables a gradient background fill in a zooming panel.

**property** ZoomPanelUsesGradient: Boolean;

The zooming panel background has a vertical gradient, if the editor's BackgroundProperty<sup>(48)</sup> defines it. You can disable a gradient fill by assigning *False* to this property.

A flat fill uses BackgroundProperty<sup>(48)</sup>.ColorBegin<sup>(124)</sup>.

**Default value:**

*True*

**See also**

- ZoomPanelVisible<sup>(168)</sup>

Shows/hides a zooming panel.

**property** ZoomPanelVisible: Boolean;

This panel is located in the bottom right corner, to the right of the horizontal scrollbar. It contains a track bar to change ZoomPercent<sup>(168)</sup> in the range ZoomMin..ZoomMax<sup>(166)</sup>.

A background of this panel is drawn using the parent TSRichViewEdit<sup>(40)</sup>'s BackgroundProperty<sup>(48)</sup>. If a gradient fill is enabled, it is always vertical in the zooming panel. Even if a gradient fill is enabled in BackgroundProperty<sup>(48)</sup>, you can disable it by assigning *False* to ZoomPanelUsesGradient<sup>(168)</sup> property. A flat fill uses BackgroundProperty<sup>(48)</sup>.ColorBegin<sup>(124)</sup>.

If ZoomMenu<sup>(166)</sup> is assigned, it is displayed when the user clicks a text area of the panel. This text is drawn using ZoomPanelFont<sup>(167)</sup>.

The appearance of the track bar depends on the value of IconStyle<sup>(160)</sup> property.

**Default value:**

*False*

**See also**

- ZoomMenu<sup>(166)</sup>
- ZoomPanelFont<sup>(167)</sup>
- IconStyle<sup>(160)</sup>

The page zooming percent.

**property** ZoomPercent: Single;

If ZoomMode<sup>(166)</sup> = *rvzmCustom*, this value is assigned by the user. Otherwise it is calculated automatically.

Assigning to this property resets ZoomMode<sup>(166)</sup> to *rvzmCustom*.

When this property is changed, TSRichViewEdit.OnZoomChanged<sup>(122)</sup> occurs.

This property can be changed in the range ZoomMin..ZoomMax<sup>(166)</sup>. If the parent editor's PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup> = *True*, there is an additional auto-calculated upper limit, to prevent displaying too large content on too small pages.

If the parent editor's PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup> = *True*, zooming is applied to the document content, while sizes of pages depends on the editor size. Otherwise, zooming is applied to pages (both page size and content).

The result of zooming depend on the screen pixel depth (DPI). First, page sizes are calculated at 96 DPI. Next, the actual sizes are calculated as:

$$\text{SizePix} := \text{Size100Pix} * (\text{ZoomPercent} / 100) * (<\text{Screen DPI}> / 96)$$

If the application supports multiple monitors having different DPI (in Delphi 10.1+), <Screen DPI> is the DPI of the monitor containing the form that contains this editor. Otherwise, this is Screen.PixelsPerInch (i.e. DPI of the primary monitor). This value can be overridden by RichViewPixelsPerInch global variable.

#### Default value:

100

#### See also:

- MouseWheelZoom<sup>(162)</sup>
- ZoomPanelVisible<sup>(168)</sup>

This value is assigned to ZoomPercent<sup>(168)</sup> when switching ViewMode<sup>(165)</sup> to *srvvmEditMode*, if ZoomModeEdit<sup>(167)</sup> = *rvzmCustom*.

**property** ZoomPercentEdit: Single;

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value:

100

This value is assigned to ZoomPercent<sup>(168)</sup> when switching ViewMode<sup>(165)</sup> to *srvvmPreviewMode*, or in preview mode when the user clicks the page and the cursor looks like "-" magnifier, if ZoomModeIN<sup>(167)</sup> = *rvzmCustom*.

**property** ZoomPercentIN: Single;

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value:

100

This value is assigned to ZoomPercent<sup>(168)</sup> when the user clicks the page and the cursor looks like "+" magnifier, if ZoomModeOUT<sup>(167)</sup> = *rvzmCustom*.

**property** ZoomPercentOUT: Single;

When this property is changed, TSRichViewEdit.OnZoomViewChanged<sup>(123)</sup> event occurs.

#### Default value:

100

## 1.3.2 TDBSRichViewEdit

TDBSRichViewEdit is a word processing component linked to a database field.

**Unit** DBSRVE.

#### Syntax

TDBSRichViewEdit = **class** (TSRichViewEdit<sup>(40)</sup>);

## Hierarchy

---

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TSRichViewEdit* <sup>(40)</sup>

## Description

---

Use a *TDBSRichViewEdit* object to enable users to edit a database field in a word processing control.

If the application doesn't require the data-aware capabilities of *TDBSRichViewEdit*, use a *TSRichViewEdit* <sup>(40)</sup> component instead, to conserve system resources.

**Alternative way:** use *LiveBindings* to link *TSRichViewEdit.Document* <sup>(51)</sup> to a database field.

## Properties

---

*TDBSRichViewEdit* overrides *RichViewEdit* <sup>(174)</sup> and *ExternalRV* <sup>(174)</sup> properties of *TSRichViewEdit* <sup>(40)</sup>. They have the same meaning, but *TDBSRichViewEdit* type.

Other properties and events provide access to properties and events of *RichViewEdit* <sup>(174)</sup>.

### 1.3.2.1 Properties

#### In *TDBSRichViewEdit*

---

- *AutoDisplay* <sup>(173)</sup>
- *DataField* <sup>(173)</sup>
- *DataSource* <sup>(173)</sup>
- *ExternalRV* <sup>(174)</sup>
- *FieldFormat* <sup>(174)</sup>
- *IgnoreEscape* <sup>(174)</sup>
- *RichViewEdit* <sup>(174)</sup>

#### Derived from *TSRichViewEdit* <sup>(40)</sup>

---

- *AcceptDragDropFormats* <sup>(47)</sup>
- *AcceptPasteFormats* <sup>(47)</sup>
- ▶ *ActiveEditor* <sup>(47)</sup>
- *AnimationMode* <sup>(48)</sup>
- *BackgroundProperty* <sup>(48)</sup>
- *BiDiMode* <sup>(48)</sup>
- ▶ *BottomMarginPix* <sup>(48)</sup>
- *CanScroll* <sup>(49)</sup>
- *CanUpdate* <sup>(49)</sup>
- *CanUpdatePosition* <sup>(49)</sup>
- *CanUpdateMargin* <sup>(49)</sup>

- CaretBlinkTime<sup>49</sup>
- ▶ CaretPos<sup>49</sup>
- CaretVisible<sup>50</sup>
- CPEventKind<sup>50</sup>
- ▶ CurControl<sup>50</sup>
- ▶ CurrentNote<sup>50</sup>
- ▶ CurrentNoteParentEditor<sup>50</sup>
- CurrentPage<sup>51</sup>
- ExternalRVStyle<sup>51</sup>
- ▶ FirstPageNo<sup>52</sup>
- FixMarginsMode<sup>52</sup>
- ▶ HMaxScrollPos<sup>52</sup>
- HScrollBar<sup>53</sup>
- HScrollBarSchemeIndex<sup>53</sup>
- HScrollPos<sup>53</sup>
- HTMLReadProperties<sup>53</sup>
- HTMLSaveProperties<sup>53</sup>
- ▶ LastPageNo<sup>54</sup>
- ▶ LeftMarginPix<sup>54</sup>
- MarkdownProperties<sup>54</sup>
- MaxPrintedItemNo<sup>55</sup>
- MaxPrintedOffsInItem<sup>55</sup>
- MenuHButtons<sup>55</sup>
- MenuHorizontal<sup>55</sup>
- MenuVButtons<sup>56</sup>
- MenuVertical<sup>56</sup>
- MinPrintedItemNo<sup>56</sup>
- MinPrintedOffInItem<sup>56</sup>
- ▶ OffsetX<sup>57</sup>
- ▶ OffsetY<sup>57</sup>
- ▶ PageCount<sup>57</sup>
- ▶ PageHeightPix, PageHeight100Pix<sup>57</sup>
- ▶ PageNoMouseDown<sup>58</sup>
- ▶ PageNoMouseMove<sup>58</sup>
- PageProperty<sup>58</sup>
- ▶ PageWidthPix, PageWidth100Pix<sup>58</sup>
- RangeSearch<sup>58</sup>
- RangeSearchFirstY<sup>59</sup>
- RangeSearchLastY<sup>59</sup>
- ReadModeProperty<sup>59</sup>
- ReadOnly<sup>59</sup>
- RichViewEdit<sup>60</sup>
- ▶ RightMarginPix<sup>60</sup>
- RTFOptions<sup>60</sup>
- RTFReadProperties<sup>61</sup>
- RVBackgroundPicture<sup>61</sup>
- RVBackgroundProperties<sup>61</sup>
- RVColor<sup>62</sup>

- RVEditorOptions <sup>62</sup>
- RVFooter <sup>62</sup>
- RVFOptions <sup>63</sup>
- RVFParaStylesReadMode <sup>63</sup>
- RVFTextStylesReadMode <sup>63</sup>
- RVHeader <sup>64</sup>
- RVNote <sup>64</sup>
- RVOptions <sup>65</sup>
- ▶ ScrolledPage <sup>65</sup>
- SkinManager <sup>66</sup>
- SmartPopupProperties <sup>66</sup>
- SmartPopupVisible <sup>66</sup>
- StyleTemplateInsertMode <sup>66</sup>
- TabNavigation <sup>68</sup>
- TextEngine <sup>68</sup>
- ▶ TopMarginPix <sup>68</sup>
- UnitsProgram <sup>68</sup>
- UseDrawHyperlinksEvent <sup>69</sup>
- UseStyleTemplates <sup>69</sup>
- Version <sup>69</sup>
- ViewProperty <sup>69</sup>
- ▶ VMaxScrollPos <sup>69</sup>
- VScrollBar <sup>69</sup>
- VScrollBarSchemeIndex <sup>70</sup>
- VScrollPos <sup>70</sup>

## Derived from TCustomControl

- Align
- Anchors
- Constraints
- Ctl3D
- DragMode
- Enabled
- Height
- HelpContext
- HelpKeyword (D6+)
- HelpType (D6+)
- Hint
- Left
- Name
- ParentCtl3D
- ParentShowHint
- PopupMenu
- ShowHint
- TabOrder
- TabStop
- Tag
- Top



- Touch (D2010+)
- Visible
- Width

#### 1.3.2.1.1 TDBSRichViewEdit.AutoDisplay

Determines whether to automatically display the contents of field in the TDBSRichViewEdit control.

**property** AutoDisplay: Boolean;

This property provides access to the RichViewEdit<sup>(174)</sup>'s property of the same name.

If AutoDisplay is *True*, the TDBSRichViewEdit control automatically displays new data when the underlying BLOB field changes (such as when moving to a new record).

If AutoDisplay is *False*, the control shows only the field name whenever the underlying BLOB field changes. To display the data, the user can double-click on the control.

The effect of AutoDisplay is not purely cosmetic. When AutoDisplay is *False*, if the data changes, the content of TDBSRichViewEdit document changes to the name of the field (written in the 0th text and paragraph style). Thus, if AutoDisplay is *False*, applications should be cautious about using the methods for saving contents to ascertain the value of the underlying field.

Calling the RichViewEdit<sup>(174)</sup>.LoadField method causes component to update to the current value of the BLOB field. This change will also be reflected in the appearance of the control on screen.

Change the value of AutoDisplay to *False* if the automatic loading of BLOB fields seems to take too long.

#### 1.3.2.1.2 TDBSRichViewEdit.DataField

Specifies the field from which the TDBSRichViewEdit control displays data.

**property** DataField: string;

This property provides access to the RichViewEdit<sup>(174)</sup>'s property of the same name.

Use DataField to bind the TDBSRichViewEdit control to the field in the dataset. To fully specify database field, both the dataset and the field within that dataset must be defined. The DataSource<sup>(173)</sup> property of the TDBSRichViewEdit control specifies the dataset which contains the DataField.

#### 1.3.2.1.3 TDBSRichViewEdit.DataSource

Links the TDBSRichViewEdit control to the dataset that contains the field it represents.

**property** DataSource : TDataSource;

This property provides access to the RichViewEdit<sup>(174)</sup>'s property of the same name.

Use DataSource to link the TDBSRichViewEdit control to the dataset in which the data can be found. To fully specify database field for the TDBSRichViewEdit control, both the dataset and the field within that dataset must be defined. Use the DataField<sup>(173)</sup> property to specify the particular field within the dataset.

#### 1.3.2.1.4 TDBSRichViewEdit.ExternalRV

Reference to an external TDBRichViewEdit component used to render documents in this TDBRichViewEdit component.

**property** ExternalRV: TDBRichViewEdit;

*Nil* by default (invisible internal RichViewEdit<sup>(174)</sup> is used).

#### 1.3.2.1.5 TDBSRichViewEdit.FieldFormat

Format for saving data in the database field

**property** FieldFormat: TRVDBFieldFormat;

This property allows changing the field saving format to RTF or plain text.

This property provides access to the RichViewEdit<sup>(174)</sup>'s property of the same name.

**Default value:**

*rvdbRVF*

#### 1.3.2.1.6 TDBSRichViewEdit.IgnoreEscape

Disallows processing of **Escape** key

**property** IgnoreEscape: Boolean;

This property provides access to the RichViewEdit<sup>(174)</sup>'s property of the same name.

By default, **Escape** cancels editing in TDBSRichViewEdit.

If you set this property to *True*, **Escape** will do nothing.

**Default value:**

*False*

#### 1.3.2.1.7 TDBSRichViewEdit.RichViewEdit

Returns TDBRichViewEdit component used to render documents.

**property** RichViewEdit: TDBRichViewEdit;

TDBSRichViewEdit can use two types of TDBRichViewEdit:

- internal: TDBRichViewEdit component is created inside TDBSRichViewEdit. Users cannot see it, it is responsible only for document rendering.
- external: refers to external TDBRichViewEdit component (ExternalRV<sup>(174)</sup>). Take a note that width and height of this DBRichViewEdit are set by TDBSRichViewEdit.

### 1.3.3 TSRVPageScroll

TSRVPageScroll is a component for displaying thumbnails of pages for a document from TDBRichViewEdit<sup>(40)</sup> component.

**Unit** SRVPageScroll.

**Syntax**

```
TSRVPageScroll = class (TRVScroller)
```

## Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TRVScroller*

## Description

The source document is specified in `SRichViewEdit` <sup>(183)</sup>.

Thumbnails are arranged vertically or horizontally, depending on `ScrollType` <sup>(181)</sup>. A number of columns (or rows) is specified in `Columns` <sup>(178)</sup>.

A size of thumbnails is defined in `PageWidth` <sup>(181)</sup>, `PageHeight` <sup>(180)</sup>, and `PageProportions` <sup>(181)</sup> properties.

Distances between and around thumbnails are specified in `PageBreakHeight` <sup>(179)</sup>, `PageBreakWidth` <sup>(180)</sup>, `MarginVertical` <sup>(179)</sup>, `MarginHorizontal` <sup>(179)</sup> properties.

In addition to thumbnails, the component can display rectangles containing page numbers. They are displayed if `PageNoVisible` <sup>(181)</sup> = `True`. Their properties are `AlignText` <sup>(176)</sup>, `BackgroundTextColor` <sup>(177)</sup>, `PageNoFont` <sup>(180)</sup>.

A background is filled with `Color`. Thumbnails have borders drawn using `PageBorderColor` <sup>(179)</sup> and `SelectColor` <sup>(182)</sup>. A shadow can be displayed using the properties `ShadowColor` <sup>(182)</sup> and `ShadowWidth` <sup>(182)</sup>.

Thumbnail quality may be increased by assigning `SmoothToombnails` <sup>(183)</sup> = `True` (however, it slows down repainting).

### 1.3.3.1 Properties

#### In `TSRVPageScroll`

- `AlignText` <sup>(176)</sup>
- `AutoScrolled` <sup>(177)</sup>
- `AutoUpdate` <sup>(177)</sup>
- `BackgroundTextColor` <sup>(177)</sup>
- `CachedPagesCount` <sup>(178)</sup>
- `CacheScrollLimit` <sup>(178)</sup>
- `CacheScrollLimit` <sup>(178)</sup>
- `Columns` <sup>(178)</sup>
- `MarginHorizontal` <sup>(179)</sup>
- `MarginVertical` <sup>(179)</sup>
- `PageBorderColor` <sup>(179)</sup>
- `PageBreakHeight` <sup>(179)</sup>
- `PageBreakWidth` <sup>(180)</sup>
- `PageHeight` <sup>(180)</sup>
- `PageNoFont` <sup>(180)</sup>

- PageNoVisible <sup>181</sup>
- PageProportions <sup>181</sup>
- PageWidth <sup>181</sup>
- ScrollType <sup>181</sup>
- SelectColor <sup>182</sup>
- ShadowColor <sup>182</sup>
- ShadowWidth <sup>182</sup>
- SmoothScroll <sup>182</sup>
- SmoothThumbnails <sup>183</sup>
- SRichViewEdit <sup>183</sup>

### Derived from TRVScroller

---

- BorderStyle
- ▶ HScrollMax
  - HScrollPos
- HScrollVisible
- ▶ InplaceEditor (not used in this component)
  - NoVScroll
- Tracking
- UseXPThemes
- ▶ VScrollMax
  - VScrollPos
- VScrollVisible
- WheelStep

### Derived from TCustomControl

---

- Align
- Anchors
- Color
- Constraints
- Ctl3D
- Height
- HelpContext
- Hint
- ParentCtl3D
- PopupMenu
- ShowHint
- TabOrder
- TabStop
- Touch (D2010+)
- Visible
- Width

#### 1.3.3.1.1 TSRVPageScroll.AlignText

Specifies the position of page number relative to the page.

##### type

```
TPSAlignText = (psatLeft, psatRight, psatTop, psatBottom);
```

```
property AlignText : TPSAlignText ;
```

Page numbers are displayed if PageNoVisible<sup>(181)</sup> = *True*.

**Default value:**

*psatBottom* (below pages)

**See also:**

- BackgroundTextColor<sup>(177)</sup>
- PageNoFont<sup>(180)</sup>
- OnDrawPageNumber<sup>(186)</sup>

#### 1.3.3.1.2 TSRVPageScroll.AutoScrolled

Specifies whether the component should be scrolled when another page becomes visible in SRichViewEdit<sup>(183)</sup>.

```
property AutoScrolled: Boolean;
```

If *True*, the component is scrolled to make SRichViewEdit.ScrolledPage<sup>(65)</sup> visible.

**Default value:**

*True*

#### 1.3.3.1.3 TSRVPageScroll.AutoUpdate

Enables/disables automatic update of thumbnails when the content of SRichViewEdit<sup>(183)</sup> is changed.

```
property AutoUpdate: Boolean;
```

Assign *False* to this value if you know that the content of SRichViewEdit<sup>(183)</sup> cannot be changed, or to make editing in SRichViewEdit<sup>(183)</sup> faster.

If *False*, call UpdateCache<sup>(185)</sup> to update thumbnails.

**Default value:**

*True*

#### 1.3.3.1.4 TSRVPageScroll.BackgroundColor

Specifies the background color below page numbers.

```
property BackgroundTextColor : TColor;
```

Page numbers are displayed if PageNoVisible<sup>(181)</sup> = *True*.

**Default value:**

*cIBtnFace*

**See also:**

- AlignText<sup>(176)</sup>
- PageNoFont<sup>(180)</sup>
- OnDrawPageNumber<sup>(186)</sup>

### 1.3.3.1.5 TSRVPageScroll.CachedPagesCount

Specifies the maximal page cache size.

**property** `CachedPagesCount: Integer;`

The cache stores page images. If the cache is large, more memory is used, but drawing is faster.

**Default value:**

10

**See also:**

- `CacheScrollLimit`<sup>(178)</sup>

### 1.3.3.1.6 TSRVPageScroll.CacheScrollLimit

Specifies how the page cache is updated when the component is scrolled.

**property** `CacheScrollLimit: Integer;`

When the cache is completely filled (see `CachedPageCount`<sup>(178)</sup>), it is updated when the component is scrolled by at least `CacheScrollLimit` number of pages.

**Default value:**

1

### 1.3.3.1.7 TSRVPageScroll.ChangeDelay

Redrawing delay, in milliseconds.

**property** `ChangeDelay: Integer;`

This property specifies the delay between when `SRichViewEdit`<sup>(183)</sup> is updated and when this `TSRVPageScroll` component is repainted.

0 means immediate repainting, it may slow down response time.

**Default value:**

50

### 1.3.3.1.8 TSRVPageScroll.Columns

Specifies the number of columns (if `ScrollType`<sup>(181)</sup> = `psstVertical`) or rows (if `ScrollType`<sup>(181)</sup> = `psstHorizontal`),

**property** `Columns: Integer;`

For example, if `SRichViewEdit`<sup>(183)</sup>.`PageCount`<sup>(57)</sup> = 5 and `Columns` = 2, thumbnails are displayed in the order shown below.

If `ScrollType`<sup>(181)</sup> = `psstVertical`:

1 2

3 4

5

If `ScrollType`<sup>(181)</sup> = `psstHorizontal`:

1 3 5

2 4

A distance between columns is defined in `PageBreakWidth`<sup>(180)</sup>, a distance between rows is defined in `PageBreakHeight`<sup>(179)</sup>.

**Default value:**

1

#### 1.3.3.1.9 `TSRVPageScroll.MarginHorizontal`

Specifies the left and the right margins.

**property** `MarginHorizontal` : `Integer`;

This is a distance from the left and from the right sides of the scrollable area to thumbnails.

**Default value:**

15

**See also:**

`MarginVertical`<sup>(179)</sup>

`PageBreakWidth`<sup>(180)</sup>

#### 1.3.3.1.10 `TSRVPageScroll.MarginVertical`

Specifies the top and the bottom margins.

**property** `MarginVertical`: `Integer`;

This is a distance from the top and from the bottom sides of the scrollable area to thumbnails.

**Default value:**

15

**See also:**

`MarginHorizontal`<sup>(179)</sup>

`PageBreakHeight`<sup>(179)</sup>

#### 1.3.3.1.11 `TSRVPageScroll.PageBorderColor`

Defines the page border color.

**property** `PageBorderColor`: `TColor`;

This is a border color for all thumbnails, except for `SRichViewEdit.CurrentPage`<sup>(51)</sup>. For this page, `SelectColor`<sup>(182)</sup> is used.

You can draw this border yourself, using `OnDrawPageBorder`<sup>(186)</sup> event.

#### 1.3.3.1.12 `TSRVPageScroll.PageBreakHeight`

Specifies the vertical distance between page thumbnails.

**property** `PageBreakHeight`: `Integer`;

Thumbnails are displayed in several rows in the following cases:

- `ScrollType`<sup>(181)</sup> = `psstVertical`, or
- `ScrollType`<sup>(181)</sup> = `pssHorizontal` and `Columns`<sup>(178)</sup> > 1

**Default value:**

20

**See also:**

- PageBreakWidth<sup>(180)</sup>
- MarginVertical<sup>(179)</sup>

**1.3.3.1.13 TSRVPageScroll.PageBreakWidth**

Specifies the horizontal distance between page thumbnails.

**property** PageBreakWidth: Integer;

Thumbnails are displayed in several columns in the following cases:

- ScrollType<sup>(181)</sup> = *pssHorizontal*, or
- ScrollType<sup>(181)</sup> = *psstVertical* and Columns<sup>(178)</sup> > 1

**Default value:**

25

**See also:**

- PageBreakHeight<sup>(179)</sup>
- MarginHorizontal<sup>(179)</sup>

**1.3.3.1.14 TSRVPageScroll.PageHeight**

Specifies the [maximal] height of page thumbnails.

**property** PageHeight: Integer;

If PageProportions<sup>(181)</sup> = *False*, all thumbnails are created with the following size: PageWidth<sup>(181)</sup> x PageHeight.

If PageProportions<sup>(181)</sup> = *True*, thumbnails have the proportions of the original pages:

- ScrollType<sup>(181)</sup> = *pssHorizontal*, thumbnails' width is calculated automatically by PageHeight;
- ScrollType<sup>(181)</sup> = *psstVertical*, thumbnails' height is calculated automatically by PageWidth<sup>(181)</sup>.

**Default value**

200

**1.3.3.1.15 TSRVPageScroll.PageNoFont**

Specifies the font for displaying page numbers.

**type**

```
TSRVPageNoFont = class (TFont)
```

**property** PageNoFont: TSRVPageNoFont;

Page numbers are displayed if PageNoVisible<sup>(181)</sup> = *True*.

**Default value:**

'Arial', 8

**See also:**

- BackgroundTextColor<sup>(177)</sup>
- AlignText<sup>(176)</sup>
- OnDrawPageNumber<sup>(186)</sup>



### 1.3.3.1.16 TSRVPageScroll.PageNoVisible

Shows or hides page numbers

**property** PageNoVisible: Boolean;

**Default value:**

*True*

**See also:**

- BackgroundTextColor<sup>(177)</sup>
- PageNoFont<sup>(180)</sup>
- AlignText<sup>(176)</sup>
- OnDrawPageNumber<sup>(186)</sup>

### 1.3.3.1.17 TSRVPageScroll.PageProportions

Specifies whether thumbnails keep proportions of the original pages.

**property** PageProportions : Boolean;

**Default value:**

*True*

**See also:**

- PageWidth<sup>(181)</sup>
- PageHeight<sup>(180)</sup>

### 1.3.3.1.18 TSRVPageScroll.PageWidth

Specifies the [maximal] width of page thumbnails.

**property** PageWidth: Integer;

If PageProportions<sup>(181)</sup>=*False*, all thumbnails are created with the following size: PageWidth x PageHeight<sup>(180)</sup>.

If PageProportions<sup>(181)</sup>=*True*, thumbnails have the proportions of the original pages:

- ScrollType<sup>(181)</sup>=*pssHorizontal*, thumbnails' width is calculated automatically by PageHeight<sup>(180)</sup>;
- ScrollType<sup>(181)</sup>=*psstVertical*, thumbnails' height is calculated automatically by PageWidth.

**Default value**

150

### 1.3.3.1.19 TSRVPageScroll.ScrollType

Specifies how page thumbnails are arranged.

**type**

TPSScrollType = (psstHorizontal, psstVertical);

**property** ScrollType: TPSScrollType;

| Mode                  | Meaning  |
|-----------------------|--|
| <i>psstHorizontal</i> | Thumbnails are displayed in columns, each column has Columns <sup>(178)</sup> rows |

*psstVertical*Thumbnails are displayed in rows, each row has Columns<sup>(178)</sup> columns**Default value:***psstVertical*

#### 1.3.3.1.20 TSRVPageScroll.SelectColor

Defines the border color for the first visible page in SRichViewEdit<sup>(183)</sup>.**property** SelectColor: TColor;This is a border color for SRichViewEdit.CurrentPage<sup>(51)</sup>. Other pages use PageBorderColor<sup>(179)</sup>.**Default value:***clRed***See also:**

- AutoScrolled<sup>(177)</sup>

#### 1.3.3.1.21 TSRVPageScroll.ShadowColor

Specifies the color of shadow cast by page thumbnails.

**property** ShadowColor: TColor;A shadow is displayed if ShadowWidth<sup>(182)</sup>>0.**Default value:***\$0099A8AC*

#### 1.3.3.1.22 TSRVPageScroll.ShadowWidth

Specifies the width of shadow cast by page thumbnails.

**property** ShadowWidth: Integer;A shadow color is ShadowColor<sup>(182)</sup>.**Default value:**

3

#### 1.3.3.1.23 TSRVPageScroll.SmoothScroll

Enables/disables smooth scrolling to the page in SRichViewEdit<sup>(183)</sup>.**property** SmoothScroll: Boolean;

Then the user clicks on a page thumbnail, the linked editor is scrolled show this page. If

**SmoothScroll** = *True*, scrolling will be smooth.**Default value:***True*

#### 1.3.3.1.24 TSRVPageScroll.SmoothThumbnails

Enables/disables smoothed thumbnails of page images.

**property** SmoothThumbnails: Boolean;

Assign *True* to this properties to generate smoothed page images. They look better, but the component is updated slower.

**Default value:**

*False*

#### 1.3.3.1.25 TSRVPageScroll.SRichViewEdit

Specifies the link to the editor containing the source document.

**property** SRichViewEdit: TSRichViewEdit<sup>(40)</sup>;

Thumbnails can be displayed only if this property is assigned.

### 1.3.3.2 Methods

#### In TSRVPageScroll

BeginUpdate<sup>(183)</sup>  
EndUpdate<sup>(183)</sup>  
First<sup>(184)</sup>  
GetPageNoToXY<sup>(184)</sup>  
Last<sup>(184)</sup>  
Next<sup>(184)</sup>  
Prev<sup>(185)</sup>  
ScrollToPage<sup>(185)</sup>  
UpdateCache<sup>(185)</sup>

#### 1.3.3.2.1 TSRVPageScroll.BeginUpdate

Disables repainting until EndUpdate<sup>(183)</sup> is called.

**procedure** BeginUpdate;

Calls to BeginUpdate and EndUpdate<sup>(183)</sup> may be nested.

Call BeginUpdate before multiple changes in SRichViewEdit<sup>(183)</sup> to make them faster.

#### 1.3.3.2.2 TSRVPageScroll.EndUpdate

Enables repainting blocked by BeginUpdate<sup>(183)</sup>.

**procedure** EndUpdate;

### 1.3.3.2.3 TSRVPageScroll.First

Moves to the first page.

**procedure** First;

This method changes `SRichViewEdit(183).CurrentPage(51)`. This page becomes highlighted in this component, but not necessary the component is scrolled to this page (because it scrolls to show `SRichViewEdit(183).ScrolledPage(65)`, not `SRichViewEdit(183).CurrentPage(51)`). Call `ScrollToPage(185)` (`SRichViewEdit(183).CurrentPage(51)`) to scroll to the page containing the caret.

**See also:**

- Prev<sup>(185)</sup>
- Next<sup>(184)</sup>
- Last<sup>(184)</sup>

### 1.3.3.2.4 TSRVPageScroll.GetPageAt

Returns the index of page displayed at the specified position.

**function** GetPageAt(X, Y: Integer; isNearest : Boolean) : Integer;

**X, Y** – client coordinates (relative to the top left corner of the client area of the component's window).

if **isNearest=False**, the method returns the index of page at the specified coordinates (from 1), or 0 if these coordinates are outside of any page.

if **isNearest=True**, the method returns the index of the nearest page to the specified coordinates (from 1).

### 1.3.3.2.5 TSRVPageScroll.Last

Moves to the last page.

**procedure** Last;

This method changes `SRichViewEdit(183).CurrentPage(51)`. This page becomes highlighted in this component, but not necessary the component is scrolled to this page (because it scrolls to show `SRichViewEdit(183).ScrolledPage(65)`, not `SRichViewEdit(183).CurrentPage(51)`). Call `ScrollToPage(185)` (`SRichViewEdit(183).CurrentPage(51)`) to scroll to the page containing the caret.

**See also:**

- First<sup>(184)</sup>
- Prev<sup>(185)</sup>
- Next<sup>(184)</sup>

### 1.3.3.2.6 TSRVPageScroll.Next

Moves to the next page.

**procedure** Next;

This method changes `SRichViewEdit(183).CurrentPage(51)`. This page becomes highlighted in this component, but not necessary the component is scrolled to this page (because it scrolls to show `SRichViewEdit(183).ScrolledPage(65)`, not `SRichViewEdit(183).CurrentPage(51)`). Call `ScrollToPage(185)` (`SRichViewEdit(183).CurrentPage(51)`) to scroll to the page containing the caret.

**See also:**

- First<sup>(184)</sup>
- Prev<sup>(185)</sup>
- Last<sup>(184)</sup>

### 1.3.3.2.7 TSRVPageScroll.Prev

Moves to the previous page.

**procedure** Prev;

This method changes SRichViewEdit<sup>(183)</sup>.CurrentPage<sup>(51)</sup>. This page becomes highlighted in this component, but not necessary the component is scrolled to this page (because it scrolls to show SRichViewEdit<sup>(183)</sup>.ScrolledPage<sup>(65)</sup>, not SRichViewEdit<sup>(183)</sup>.CurrentPage<sup>(51)</sup>). Call ScrollToPage<sup>(185)</sup>(SRichViewEdit<sup>(183)</sup>.CurrentPage<sup>(51)</sup>) to scroll to the page containing the caret.

**See also:**

- First<sup>(184)</sup>
- Next<sup>(184)</sup>
- Last<sup>(184)</sup>

### 1.3.3.2.8 TSRVPageScroll.ScrollToPage

Scrolls to the specified page.

**procedure** ScrollToPage(PageNo : Integer);

This method scrolls this component, without scrolling SRichViewEdit<sup>(183)</sup>.

### 1.3.3.2.9 TSRVPageScroll.UpdateCache

Updates the cached images of page thumbnails.

**procedure** UpdateCache;

Normally, the cache is updated automatically. This method is useful if AutoUpdate<sup>(177)</sup>=False.

## 1.3.3.3 Events

### In TSRVPageScroll

- OnDrawPageBorder<sup>(186)</sup>
- OnDrawPageNumber<sup>(186)</sup>
- OnDrawSelectedPage<sup>(187)</sup>

### Derived from TRVScroller

- OnHScrolled
- OnVScrolled

### Derived from TCustomControl

- OnDbClick
- OnClick
- OnDbClick

- OnDragDrop
- OnDragOver
- OnEndDrag
- OnEnter
- OnExit
- OnGesture
- OnKeyPress
- OnMouseMove

#### 1.3.3.3.1 TSRVPageScroll.OnDrawPageBorder

Allows a custom drawing of borders of page thumbnails.

##### type

```
TPSDrawPageBorderEvent = procedure (Sender: TObject; Canvas: TCanvas;
    const R: TRect; PageNo: Integer) of object;
```

**property** OnDrawPageBorder : TPSDrawPageBorderEvent ;

**Canvas** – canvas where to draw.

**R** – location of the page thumbnail.

**PagNo** – index of the thumbnails' page, from 1.

By default, all thumbnails have a border of PageBorderColor<sup>(179)</sup> color. If this event is assigned, a default border is not drawn. But the selected page is still highlighted with SelectColor<sup>(182)</sup> color, use OnDrawSelectedPage<sup>(187)</sup> to override it.

#### 1.3.3.3.2 TSRVPageScroll.OnDrawPageNumber

Allows a custom drawing of page numbers.

##### type

```
TPSDrawPageNumberEvent = procedure (Sender: TObject; Canvas: TCanvas;
    const R: TRect; PageNo : Integer; isSelect : Boolean) of object;
```

**property** OnDrawPageNumber: TPSDrawPageNumberEvent;

By default, page numbers are displayed if PageNoVisible<sup>(181)</sup> = *True*.

They are displayed in the location specified in AlignText<sup>(176)</sup>, drawing using BackgroundTextColor<sup>(177)</sup> and PageNoFont<sup>(180)</sup> properties.

If this event is assigned, the default drawing is not performed.

**Canvas** – canvas where to draw.

**R** – location of the page number, calculated according to AlignText<sup>(176)</sup> and PageNoFont<sup>(180)</sup>.

**PagNo** – page number to draw.

**isSelect** specifies whether this page is highlighted.

### 1.3.3.3 TSRVPageScroll.OnDrawSelectedPage

Allows a custom drawing of highlighting of the selected thumbnail.

#### type

```
TPSDrawSelectedPageEvent = procedure (Sender: TObject; Canvas: TCanvas;  
    const R: TRect; PageNo: Integer) of object;
```

**property** OnDrawSelectedPage: TPSDrawSelectedPageEvent;

**Canvas** – canvas where to draw.

**R** – location of the page thumbnail.

**PagNo** – index of the thumbnails' page, from 1.

This drawing occurs after drawing the default page border, see OnDrawPageBorder<sup>(186)</sup>.

By default, highlighted page have a border of SelectColor<sup>(182)</sup> color. If this event is assigned, this default border is not drawn

## 1.3.4 TScIRVRuler

TScIRVRuler is a horizontal or vertical ruler.

**Unit** ScIRVRuler.

#### Syntax

```
TScIRVRuler = class (TRVRuler);
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TCustomRuler*  
*TRuler*  
*TRVRuler*

#### Description

The class is based on the TRVRuler class and adapts the ruler for work with TSRichViewEdit<sup>(40)</sup>.

It introduces new properties and methods for work as a vertical ruler and for assigning margins.

TRVRuler is included in RichViewActions.

The ruler works with the editor specified in SRichViewEdit<sup>(188)</sup> property.

### 1.3.4.1 Properties

#### In TScIRVRuler

■ SRichViewEdit<sup>(188)</sup>

## Derived from TRVRuler

---

...

### 1.3.4.1.1 TSclRVRuler.SRichViewEdit

A reference to the editor.

**property** SRichViewEdit: TSRichViewEdit<sup>40</sup>;

The ruler works with this editor.

### 1.3.4.2 Methods

#### In TSclRVRuler

---

GetPageProperties<sup>188</sup>

Scrolled<sup>189</sup>

SetPageProperties<sup>189</sup>

#### Derived from TRVRuler

---

...

### 1.3.4.2.1 TSclRVRuler.CheckMargins

Assigns the ruler properties from the linked SRichViewEdit<sup>188</sup>.

**procedure** CheckMargins;

### 1.3.4.2.2 TSclRVRuler.GetPageProperties

The procedure returns margins and page size in the specified units. The returned values depend on the ruler type (horizontal/vertical).

**procedure** GetPageProperties(Units: TRulerUnits;  
var AMargin1, AMargin2, APageWidth: Extended);

#### Input parameters

**Units** – type of units: centimeters, millimeters, pixels etc. The output parameters are measured in these units.

#### Output parameters

**AMargin1** – left margin (LeftMargin) for horizontal ruler, or top margin (TopMargin) for vertical ruler.

**AMargin2** – right margin (RightMargin) for horizontal ruler, bottom margin (BottomMargin) for vertical ruler.

**APageWidth** – page width (PageWidth) for horizontal ruler, page height (PageHeight) for vertical ruler.



### 1.3.4.2.3 TScIRVRuler.Scrolled

Updates the ruler's margins.

**procedure** Scrolled;

This method must be called if `SRichViewEdit189.CanUpdate49` = *False*. Otherwise, margins are updated automatically.

### 1.3.4.2.4 TScIRVRuler.SetPageProperties

The procedure changes margins and page size. The changed properties depend on the ruler type (horizontal/vertical).

**procedure** SetPageProperties(Units: TRulerUnits;  
AMargin1, AMargin2, APageWidth: Extended);

**Units** – type of units: centimeters, millimeters, pixels etc. Other parameters are measured in these units.

**AMargin1** – left margin (LeftMargin) for horizontal ruler, or top margin (TopMargin) for vertical ruler.

**AMargin2** – right margin (RightMargin) for horizontal ruler, bottom margin (BottomMargin) for vertical ruler.

**APageWidth** – page width (PageWidth) for horizontal ruler, page height (PageHeight) for vertical ruler.

## 1.3.4.3 Events

### In TScIRVRuler

■ OnAfterMarginChanged<sup>189</sup>

### Derived from TRVRuler

...

### 1.3.4.3.1 TScIRVRuler.OnAfterMarginChanged

Occurs after the ruler's margins are changed.

**property** OnAfterMarginChanged: TNotifyEvent;

## 1.3.5 TSVTToolBar

TSRVToolBar is a simple toolbar component.

**Unit** SRVToolBar.

### Syntax

TSRVToolBar = **class** (TCustomControl)

### Hierarchy

TObject

TPersistent

*TComponent*

*TControl*

*TWinControl*

*TCustomControl*

## Description

TSRVToolbar is a container for buttons (TSRVToolButton<sup>(272)</sup>).

The main features:

- all buttons in the toolbar have the same height and width;
- buttons are arranged in rows and columns;
- buttons can have different images for different states (normal, highlighted, disabled);
- button captions can be displayed in the bottom or the top part of the toolbar.

This component can be used directly, or inside a floating tool window (TSRVToolWindow<sup>(197)</sup>), or inside a scrollbar area of TSVRichViewEdit<sup>(40)</sup> component (see TSRVPropertyTBH<sup>(148)</sup> and TSRVPropertyTBV<sup>(149)</sup>).

## Properties

This class contains almost the same set of properties as TSRVCustomPropertyTB<sup>(127)</sup> class and TSRVToolWindow<sup>(197)</sup> component.

A toolbar consists of buttons and a hint area.

The hint area can be placed at the top or at the bottom of the toolbar, depending on AlignText<sup>(191)</sup> property. A height of this area is defined in HintHeight<sup>(194)</sup> property. This area can work as a "Cancel" button. When none of buttons is highlighted, it displays HintCancel<sup>(194)</sup>. Otherwise, it displays Caption<sup>(273)</sup> of tool button. A font for text displayed in this area is specified in HintFont<sup>(194)</sup>.

Buttons are defined in Buttons<sup>(192)</sup> collection. Buttons are arranged in rows and columns, the number of columns is specified in ButtonsCol<sup>(192)</sup>. A size of buttons is defined in ButtonWidth<sup>(193)</sup> and ButtonHeight<sup>(192)</sup> properties. A spacing between buttons is specified in Spacer<sup>(196)</sup>. A horizontal positions of buttons is defined by AlignButton<sup>(191)</sup> and Indent<sup>(195)</sup> properties.

The toolbar can have a border, its width is specified in BorderWidth<sup>(192)</sup>.

The toolbar color is Color<sup>(193)</sup>. Colors of disabled, pressed and highlighted buttons are specified in ColorDisable<sup>(193)</sup>, ColorDown<sup>(193)</sup>, and ColorSelect<sup>(194)</sup> properties. A pen of frame around the toolbar is defined in PenFrame<sup>(195)</sup>.

### 1.3.5.1 Properties

#### In TSRVToolBar

- AlignButton<sup>(191)</sup>
- AlignText<sup>(191)</sup>
- AutoSize<sup>(192)</sup>
- BorderWidth<sup>(192)</sup>
- ButtonsCol<sup>(192)</sup>
- ButtonHeight<sup>(192)</sup>
- Buttons<sup>(192)</sup>

- ButtonWidth <sup>(193)</sup>
- Color <sup>(193)</sup>
- ColorDisable <sup>(193)</sup>
- ColorDown <sup>(193)</sup>
- ColorSelect <sup>(194)</sup>
- HintCancel <sup>(194)</sup>
- HintFont <sup>(194)</sup>
- HintHeight <sup>(194)</sup>
- ImageList <sup>(195)</sup>
- Indent <sup>(195)</sup>
- PenFrame <sup>(195)</sup>
- ScaleImagesForDPI <sup>(195)</sup>
- Spacer <sup>(196)</sup>

## Derived from TCustomControl

---

- Align
- Anchors
- Cursor
- Height
- HelpContext
- HelpKeyword (D6+)
- HelpType (D6+)
- Hint
- Left
- Name
- Tag
- Top
- Width

### 1.3.5.1.1 TSRVToolBar.AlignButton

Defines the position of buttons in the toolbar window.

**property** AlignButton: TSRVAlignButton <sup>(271)</sup>;

#### See also

Indent <sup>(195)</sup>

#### Default value:

*srvabCenter*

### 1.3.5.1.2 TSRVToolBar.AlignText

Defines the position of text with the captions <sup>(273)</sup> of the highlighted button (button below the mouse pointer)

**property** AlignText: TSRVAlignText <sup>(271)</sup>;

When none of the buttons are highlighted, HintCancel <sup>(194)</sup> is displayed in this area.

#### Default value:

*srvatBottom*

### 1.3.5.1.3 TSRVToolBar.AutoSize

Allows autocalculation of the toolbar size.

**property** AutoSize: Boolean;

**Default value:**

*False*

### 1.3.5.1.4 TSRVToolBar.BorderWidth

Specifies a width of the toolbar border (spacing around the toolbar)

**property** BorderWidth: TRVPixel96Length;

A border (see PenFrame<sup>(195)</sup>) is drawn in this area. The rest of space (if BorderWidth>the pen's width) is painted with Color<sup>(193)</sup>.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

1

**See also:**

- Spacer<sup>(196)</sup>
- HintHeight<sup>(194)</sup>

### 1.3.5.1.5 TSRVToolBar.ButtonCol

Specifies the maximal number of columns.

**property** ButtonCol: Integer;

ButtonCol specifies the number of buttons what may be placed in a line.

**Default value:**

-1 (unlimited)

### 1.3.5.1.6 TSRVToolBar.ButtonHeight

Sets the height of tool buttons (TSRVToolButton<sup>(272)</sup>), in pixels.

**property** ButtonHeight: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

26

**See also:**

- ButtonWidth<sup>(193)</sup>

### 1.3.5.1.7 TSRVToolBar.Buttons

Collection of tool buttons (TSRVToolButton<sup>(272)</sup>).

**property** Buttons: TSRVButtonCollection<sup>(271)</sup>;

#### 1.3.5.1.8 TSRVToolBar.ButtonWidth

Sets the width of tool buttons (TSRVToolButton<sup>(272)</sup>), in pixels.

**property** ButtonWidth: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

26

**See also:**

- ButtonHeight<sup>(192)</sup>

#### 1.3.5.1.9 TSRVToolBar.Color

Specifies the toolbar background color.

**property** Color: TColor;

This color is used for background of:

- the border around the toolbar (see BorderWidth<sup>(192)</sup>)
- area between the buttons (see Spacer<sup>(196)</sup>)
- hint area (see HintHeight<sup>(194)</sup>)
- tool buttons (if the button is in a normal state, i.e. not highlighted, pressed or disabled).

**Default value:**

*c/Window*

**See also:**

- ColorDisable<sup>(193)</sup>
- ColorDown<sup>(193)</sup>
- ColorSelect<sup>(194)</sup>

#### 1.3.5.1.10 TSRVToolBar.ColorDisable

Background color of disabled tool buttons<sup>(192)</sup>.

**property** ColorDisable: TColor;

This color is used for buttons having Enabled<sup>(273)</sup>=False.

**Default value:**

*clSilver*

**See also:**

- Color<sup>(193)</sup>
- ColorDown<sup>(193)</sup>
- ColorSelect<sup>(194)</sup>

#### 1.3.5.1.11 TSRVToolBar.ColorDown

Background color of pressed tool buttons<sup>(192)</sup>.

**property** ColorDown: TColor;

**Default value:**

\$00EED2C1

**See also:**

- Color<sup>(193)</sup>
- ColorDisable<sup>(193)</sup>
- ColorSelect<sup>(194)</sup>

#### 1.3.5.1.12 TSRVToolBar.ColorSelect

Background color of highlighted tool buttons<sup>(192)</sup> (buttons under the mouse pointer).

**property** ColorSelect: TColor;

**Default value:**

\$00E2B598

**See also:**

- Color<sup>(193)</sup>
- ColorDisable<sup>(193)</sup>
- ColorDown<sup>(193)</sup>

#### 1.3.5.1.13 TSRVToolBar.HintCancel

Specifies the text to display in the hint area when none of the buttons is highlighted.

**property** HintCancel: TRVUnicodeString;

When this text is displayed, clicking on the hint area closes the tool window (and OnClickButton<sup>(196)</sup> occurs with *nil* parameter).

**Default value:**

'Cancel'

#### 1.3.5.1.14 TSRVToolBar.HintFont

Specifies the font for the hint area.

**type**

TSRVTBFont = **class**(TFont);

**property** HintFont: TSRVTBFont;

This font is used for captions of tool buttons and HintCancel<sup>(194)</sup>.

**Default value:**

default properties of TFont

**See also:**

- HintHeight<sup>(194)</sup>

#### 1.3.5.1.15 TSRVToolBar.HintHeight

Specifies the height of the hint area.

**property** HintHeight: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**See also:**

- AlignText<sup>(194)</sup>
- HintFont<sup>(194)</sup>

#### 1.3.5.1.16 TSRVToolBar.ImageList

A reference to image list for images in tool buttons<sup>(192)</sup>.

**property** ImageList: TCustomImageList;

This image list must contain buttons glyphs for all buttons states (normal, pressed, disabled) (you can use the same image for all of them).

When a toolbar displayed on a monitor that has pixel density (DPI) higher than 96, the toolbar draws images stretched, proportionally to the screen DPI. To turn off stretching, assign ScaleImagesForDPI<sup>(195)</sup> = *False*.

#### 1.3.5.1.17 TSRVToolBar.Indent

Changes the position of buttons in the toolbar window.

**property** Indent: Integer;

If AlignButton<sup>(191)</sup> = *srvabLeft*, buttons are shifted by Indent to the right.

If AlignButton<sup>(191)</sup> = *srvabRight*, buttons are shifted by Indent to the left.

If AlignButton<sup>(191)</sup> = *srvabCenter*, buttons are shifted by Indent/2 to the left.

**Default value:**

0

#### 1.3.5.1.18 TSRVToolBar.PenFrame

Specifies a pen for drawing lines in the tool bar.

**property** PenFrame: TPen;

This pen is used for drawing border around the toolbar. The width of this border is defined as a pen's width, not in BorderWidth<sup>(192)</sup>.

**Default value:**

Style=*psSolid*, Width=1, Color=*\$00C56A31*;

#### 1.3.5.1.19 TSRVToolBar.ScaleImagesForDPI

Specify whether images from ImageList<sup>(195)</sup> are displayed stretched according to the screen pixel density (DPI).

**property** ScaleImagesForDPI: Boolean;

If *True*, toolbar images are stretched from 96 DPI to the screen DPI.

If *False*, toolbar images are displayed as they are. This mode is useful if ImageList<sup>(195)</sup> contains images that corresponds to the current DPI.

**Default value**

*True*

### 1.3.5.1.20 TSRVToolBar.Spacer

Specifies the distance between tool buttons <sup>(192)</sup>.

**property** Spacer: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

0

## 1.3.5.2 Events

### In TSRVToolBar

- OnClickButton <sup>(196)</sup>
- OnDrawButtonBackground <sup>(196)</sup>
- OnEnter <sup>(196)</sup>
- OnEnterButton <sup>(197)</sup>
- OnLeave <sup>(197)</sup>

### 1.3.5.2.1 TSRVToolBar.OnClickButton

Occurs when user clicks inside the toolbar.

**property** OnClickButton: TSRVClickButtonEvent <sup>(272)</sup>;

**ToolButton** – the clicked button (one of Buttons <sup>(192)</sup>), or *nil*, if the user clicked outside any button (for example, in a hint area).

### 1.3.5.2.2 TSRVToolBar.OnDrawButtonBackground

Allows drawing a background of a toolbar button

**type**

```
TSRVDDrawButtonEvent = procedure (Sender: TObject; ToolButton: TSRVToolButton (272);  
    ACanvas: TCanvas; const ARect: TRect; Selected: Boolean;  
    var DoDefault: Boolean) of object;
```

**property** OnDrawButtonBackground: TSRVDDrawButtonEvent;

**ToolButton** – the button to draw (one of Buttons <sup>(192)</sup>)

The button must be drawn in the rectangle **ARect** on **ACanvas**.

You can check if the button is pressed: **ToolButton.Down**. If the button is below the mouse pointer, **Selected** = *True*.

If you draw the button background yourself, assign **DoDefault** = *False* to cancel default drawing by the toolbar.

### 1.3.5.2.3 TSRVToolBar.OnEnter

Occurs when user moves the mouse pointer into the toolbar area.

**property** OnEnter: TNotifyEvent;

**See also:**



- OnLeave<sup>(197)</sup>

#### 1.3.5.2.4 TSRVToolBar.OnEnterButton

Occurs when user moves the mouse pointer into the button area.

**property** OnEnterButton: TSRVClickButtonEvent<sup>(272)</sup>;

**ToolBar** – the button below the mouse pointer, one of Buttons<sup>(192)</sup>.

#### 1.3.5.2.5 TSRVToolBar.OnLeave

Occurs when user moves the mouse pointer out of the toolbar area.

**property** OnLeave: TNotifyEvent;

**See also:**

- OnEnter<sup>(196)</sup>

### 1.3.6 TSRVToolWindow

TSRVToolWindow shows a toolbar inside a popup window.

**Unit** SRVToolWindow.

#### Syntax

TSRVToolWindow = **class** (TComponent)

#### Hierarchy

TObject

TPersistent

TComponent

#### Description

TSRVToolWindow shows a toolbar inside a popup window, when you call Execute<sup>(203)</sup> method. The window is closed without any action when it loses focus or when user clicks outside any button on the toolbar

TSRVToolWindow uses TSRVToolBar<sup>(189)</sup> inside.

Usually TSRVToolWindow is used as a submenu for buttons in TSVRichViewEdit<sup>(40)</sup>'s MenuVertical<sup>(56)</sup>.

#### Properties

This class contains almost the same set of properties as TSRVCustomPropertyTB<sup>(127)</sup> class and TSRVToolBar<sup>(189)</sup> component.

A toolbar consists of buttons and a hint area.

The hint area can be placed at the top or at the bottom of the toolbar, depending on AlignText<sup>(198)</sup> property. A height of this area is defined in HintHeight<sup>(201)</sup> property. This area can work as a "Cancel" button. When none of buttons is highlighted, it displays HintCancel<sup>(201)</sup>. Otherwise, it displays Caption<sup>(273)</sup> of tool button. A font for text displayed in this area is specified in HintFont<sup>(201)</sup>.

Buttons are defined in Buttons<sup>(199)</sup> collection. Buttons are arranged in rows and columns, the number of columns is specified in ButtonsCol<sup>(199)</sup>. A size of buttons is defined in ButtonWidth<sup>(199)</sup> and ButtonHeight<sup>(199)</sup> properties. A spacing between buttons is specified in Spacer<sup>(202)</sup>.

The toolbar can have a border, its width is specified in BorderWidth<sup>(199)</sup>.

The toolbar color is Color<sup>(200)</sup>. Colors of disabled, pressed and highlighted buttons are specified in ColorDisable<sup>(200)</sup>, ColorDown<sup>(200)</sup>, and ColorSelect<sup>(201)</sup> properties. A pen of frame around the toolbar is defined in PenFrame<sup>(202)</sup>.

### 1.3.6.1 Properties

#### In TSRVToolWindow

- AlignText<sup>(198)</sup>
- BorderWidth<sup>(199)</sup>
- ButtonCol<sup>(199)</sup>
- ButtonHeight<sup>(199)</sup>
- Buttons<sup>(199)</sup>
- ButtonWidth<sup>(199)</sup>
- Color<sup>(200)</sup>
- ColorDisable<sup>(200)</sup>
- ColorDown<sup>(200)</sup>
- ColorSelect<sup>(201)</sup>
- HintCancel<sup>(201)</sup>
- HintFont<sup>(201)</sup>
- HintHeight<sup>(201)</sup>
- ImageList<sup>(202)</sup>
- PenFrame<sup>(202)</sup>
- ScaleImagesForDPI<sup>(202)</sup>
- Spacer<sup>(202)</sup>

#### Derived from TComponent

- Name
- Tag

##### 1.3.6.1.1 TSRVToolWindow.AlignText

Defines the position of text with the captions<sup>(273)</sup> of the highlighted button (button below the mouse pointer)

**property** AlignText: TSRVAlignText<sup>(271)</sup>;

When none of the buttons are highlighted, HintCancel<sup>(201)</sup> is displayed in this area.

**Default value:**

*srvatBottom*

#### 1.3.6.1.2 TSRVToolWindow.BorderWidth

Specifies a width of the toolbar border (spacing around the toolbar)

**property** BorderWidth: TRVPixel96Length;

A border (see PenFrame<sup>(202)</sup>) is drawn in this area. The rest of space (if BorderWidth>the pen's width) is painted with Color<sup>(200)</sup>.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

1

**See also:**

- Spacer<sup>(202)</sup>
- HintHeight<sup>(201)</sup>

#### 1.3.6.1.3 TSRVToolWindow.ButtonCol

Specifies the maximal number of columns.

**property** ButtonCol: Integer;

ButtonCol specifies the number of buttons what may be placed in a line.

**Default value:**

-1 (unlimited)

#### 1.3.6.1.4 TSRVToolWindow.ButtonHeight

Sets the height of tool buttons (TSRVToolButton<sup>(272)</sup>), in pixels.

**property** ButtonHeight: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

26

**See also:**

- ButtonWidth<sup>(199)</sup>

#### 1.3.6.1.5 TSRVToolWindow.Buttons

Collection of tool buttons (TSRVToolButton<sup>(272)</sup>).

**property** Buttons: TSRVButtonCollection<sup>(271)</sup>;

#### 1.3.6.1.6 TSRVToolWindow.ButtonWidth

Sets the width of tool buttons (TSRVToolButton<sup>(272)</sup>), in pixels.

**property** ButtonWidth: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

26

**See also:**

- ButtonHeight<sup>(199)</sup>

### 1.3.6.1.7 TSRVToolWindow.Color

Specifies the toolbar background color.

**property** Color: TColor;

This color is used for background of:

- the border around the toolbar (see BorderWidth<sup>(199)</sup>)
- area between the buttons (see Spacer<sup>(202)</sup>)
- hint area (see HintHeight<sup>(201)</sup>)
- tool buttons (if the button is in a normal state, i.e. not highlighted, pressed or disabled).

**Default value:**

clWindow

**See also:**

- ColorDisable<sup>(200)</sup>
- ColorDown<sup>(200)</sup>
- ColorSelect<sup>(201)</sup>

### 1.3.6.1.8 TSRVToolWindow.ColorDisable

Background color of disabled tool buttons<sup>(199)</sup>.

**property** ColorDisable: TColor;

This color is used for buttons having Enabled<sup>(273)</sup>=False.

**Default value:**

clSilver

**See also:**

- Color<sup>(200)</sup>
- ColorDown<sup>(200)</sup>
- ColorSelect<sup>(201)</sup>

### 1.3.6.1.9 TSRVToolWindow.ColorDown

Background color of pressed tool buttons<sup>(199)</sup>.

**property** ColorDown: TColor;

**Default value:**

\$00EED2C1

**See also:**

- Color<sup>(200)</sup>
- ColorDisable<sup>(200)</sup>
- ColorSelect<sup>(201)</sup>

#### 1.3.6.1.10 TSRVToolWindow.ColorSelect

Background color of highlighted tool buttons <sup>(199)</sup> (buttons under the mouse pointer).

**property** ColorSelect: TColor;

**Default value:**

\$00E2B598

**See also:**

- Color <sup>(200)</sup>
- ColorDisable <sup>(200)</sup>
- ColorDown <sup>(200)</sup>

#### 1.3.6.1.11 TSRVToolWindow.HintCancel

Specifies the text to display in the hint area when none of the buttons is highlighted.

**property** HintCancel: TRVUnicodeString;

When this text is displayed, clicking on the hint area closes the tool window (and OnClickButton <sup>(203)</sup> occurs with *nil* parameter).

**Default value:**

'Cancel'

#### 1.3.6.1.12 TSRVToolWindow.HintFont

Specifies the font for the hint area.

**type**

```
TSRVTBFont = class(TFont);
```

**property** HintFont: TSRVTBFont;

This font is used for captions of tool buttons and HintCancel <sup>(201)</sup>.

**Default value:**

default properties of TFont

**See also:**

- HintHeight <sup>(201)</sup>

#### 1.3.6.1.13 TSRVToolWindow.HintHeight

Specifies the height of the hint area.

**property** HintHeight: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**See also:**

- AlignText <sup>(201)</sup>
- HintFont <sup>(201)</sup>

#### 1.3.6.1.14 TSRVToolWindow.ImageList

A reference to image list for images in tool buttons <sup>(199)</sup>.

**property** ImageList: TCustomImageList;

This image list must contain buttons glyphs for all buttons states (normal, pressed, disabled) (you can use the same image for all of them).

When a toolbar displayed on a monitor that has pixel density (DPI) higher than 96, the toolbar draws images stretched, proportionally to the screen DPI. To turn off stretching, assign ScaleImagesForDPI <sup>(202)</sup> = *False*.

#### 1.3.6.1.15 TSRVToolWindow.PenFrame

Specifies a pen for drawing lines in the tool bar.

**property** PenFrame: TPen;

This pen is used for drawing border around the toolbar. The width of this border is defined as a pen's width, not in BorderWidth <sup>(199)</sup>.

**Default value:**

Style=*psSolid*, Width=1, Color=**\$00C56A31**;

#### 1.3.6.1.16 TSRVToolWindow.ScaleImagesForDPI

Specify whether images from ImageList <sup>(202)</sup> are displayed stretched according to the screen pixel density (DPI).

**property** ScaleImagesForDPI: Boolean;

If *True*, toolbar images are stretched from 96 DPI to the screen DPI.

If *False*, toolbar images are displayed as they are. This mode is useful if ImageList <sup>(202)</sup> contains images that corresponds to the current DPI.

**Default value**

*True*

#### 1.3.6.1.17 TSRVToolWindow.Spacer

Specifies the distance between tool buttons <sup>(199)</sup>.

**property** Spacer: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

0

### 1.3.6.2 Methods

#### In TSRVToolWindow

Execute <sup>(203)</sup>

### 1.3.6.2.1 TSRVToolWindow.Execute

The methods show a window with a toolbar.

```
procedure Execute(X, Y : Integer); overload;  
procedure Execute(R : TRect); overload;
```

The window remains open until it loses focus or one of buttons is clicked.

**X,Y** – screen coordinates of the top left corner of the window. The window will be shown exactly at these coordinates.

**R** – screen coordinates specifying a rectangle of the button which called this window. The window will be positioned close to this button without overlapping it, and without going beyond the screen. This method sets AlignText<sup>(198)</sup> property depending on the window position.

## 1.3.6.3 Events

### In TSRVToolWindow

- OnClickButton<sup>(203)</sup>
- OnEnter<sup>(203)</sup>
- OnEnterButton<sup>(203)</sup>
- OnLeave<sup>(204)</sup>
- OnSelectEvent<sup>(204)</sup>

### 1.3.6.3.1 TSRVToolWindow.OnClickButton

Occurs when user clicks inside the toolbar.

```
property OnClickButton: TSRVClickButtonEvent(272);
```

**ToolButton** – the clicked button (one of Buttons<sup>(199)</sup>), or *nil*, if the user clicked outside any button (for example, in a hint area).

### 1.3.6.3.2 TSRVToolWindow.OnEnter

Occurs when user moves the mouse pointer into the toolbar area.

```
property OnEnter: TNotifyEvent;
```

**See also:**

- OnLeave<sup>(204)</sup>

### 1.3.6.3.3 TSRVToolWindow.OnEnterButton

Occurs when user moves the mouse pointer into the button area.

```
property OnEnterButton: TSRVClickButtonEvent(272);
```

**ToolButton** – the button below the mouse pointer, one of Buttons<sup>(199)</sup>.

#### 1.3.6.3.4 TSRVToolWindow.OnLeave

Occurs when user moves the mouse pointer out of the toolbar area.

**property** OnLeave: TNotifyEvent;

**See also:**

- OnEnter<sup>(203)</sup>

#### 1.3.6.3.5 TSRVToolWindow.OnSelectEvent

Occurs when user clicked a tool button or closed the window.

**property** OnSelectEvent: TSRVClickButtonEvent<sup>(272)</sup>;

The window is closed (and the action is canceled) when the user clicks on the window outside any button, or when it loses the input focus.

### 1.3.7 TSRVPrint

TSRVPrint allows printing documents from TSVRichViewEdit<sup>(40)</sup> in several modes, including posters.

**Unit** SRVPrint.

**Syntax**

TSRVPrint = **class** (TComponent)

#### Hierarchy

TObject

TPersistent

TComponent

#### Description

The main property of TRVPrint is PrintMode<sup>(214)</sup>. It defines how the document in SRichViewEdit<sup>(217)</sup> is printed.

- *srvpAuto* – the printer paper size and orientation are set automatically;
- *srvpStretchToFit* – the current printer paper size and orientation are used;
- *srvpStretchIfLarger* is like *srvpStretchToFit* if the document page size is larger than the printer paper size. Otherwise, the document is printed at the top left corner of paper.
- *srvpTiles* – if possible, prints several copies of the document page on the same paper sheet;
- *srvpPoster* – allows printing posters: each page can be printed on several paper sheets; this is the most complex mode, it is described below.
- *srvpGrid* – allows printing several document pages on the same paper sheet.

The component uses the current printer (specified in Printer.PrinterIndex, where Printer is a function defined in the Printers unit). When the parameters of the current printer are changed, or the current printer is changed, call Update<sup>(222)</sup> to recalculate properties according to the printer.

#### Posters

#### Sizes



A document is prepared in `TSRichViewEdit`<sup>(40)</sup> component . Its size is defined in `TSRichViewEdit`<sup>(40)</sup>'s page properties<sup>(58)</sup> .

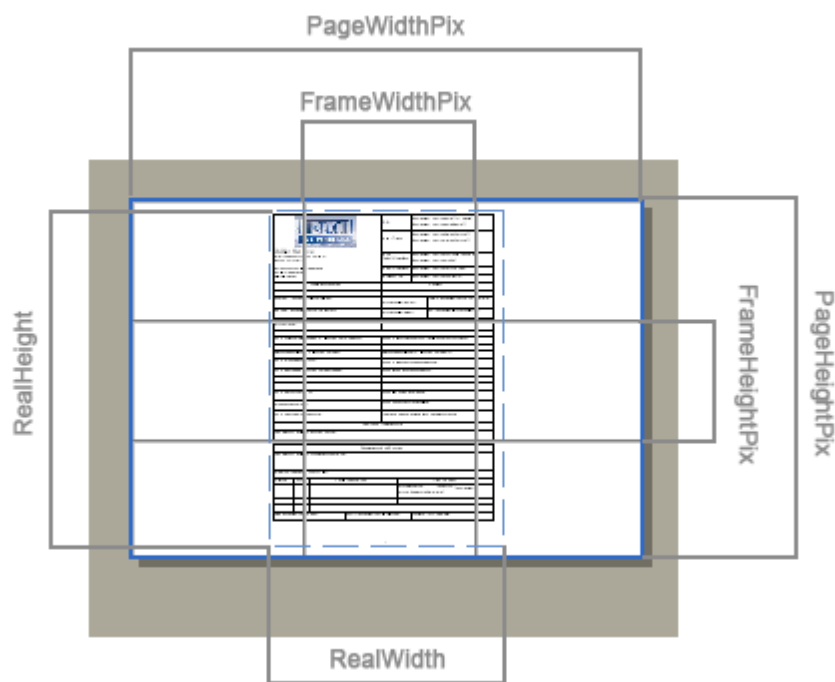
Each page of this document is scaled to `RealWidth`<sup>(216)</sup> x `RealHeight`<sup>(215)</sup> (see below how they are calculated).

Each page of this document is printed as a poster. The poster consists of frames. Frame is one paper sheet, it has size of the printer's paper (`FrameWidthPix`<sup>(210)</sup> x `FrameHeightPix`<sup>(209)</sup>, orientation `Orientation`<sup>(212)</sup>).

The poster consists of `Ceil(FrameCountX`<sup>(208)</sup>) x `Ceil(FrameCountY`<sup>(209)</sup>) frames (so, each page of the original document will be printed on `Ceil(FrameCountX`<sup>(208)</sup>) \* `Ceil(FrameCountY`<sup>(209)</sup>) printer pages).

The poster size in pixels is `PageWidthPix`<sup>(214)</sup> x `PageHeightPix`<sup>(213)</sup> (i.e. `Ceil(FrameCountX`<sup>(208)</sup>) \* `FrameWidthPix`<sup>(210)</sup> x `Ceil(FrameCountY`<sup>(209)</sup>) \* `FrameHeightPix`<sup>(209)</sup>).

`FrameCountX`<sup>(208)</sup> and `FrameCountY`<sup>(209)</sup> can be fractional values (if `IntegerCount`<sup>(210)</sup> = *False*), it affects calculation of `RealWidth`<sup>(216)</sup> and `RealHeight`<sup>(215)</sup> (see below).



Sizes in posters

### How the document page is scaled

As it was said above, each page of the original document is scaled to `RealWidth`<sup>(216)</sup> x `RealHeight`<sup>(215)</sup> .

If `ScaleImage`<sup>(216)</sup> = *True*, pages are scaled to the full poster size (`RealWidth`<sup>(216)</sup> = `PageWidthPix`<sup>(214)</sup>, `RealHeight`<sup>(215)</sup> = `PageHeightPix`<sup>(213)</sup>), without keeping the original proportions.

If `ScaleImage`<sup>(216)</sup> = `False`, pages are scaled proportionally to the maximal size that fits in `TotalFrameWPix`<sup>(217)</sup> x `TotalFrameHPix`<sup>(217)</sup> (so `RealWidth`<sup>(216)</sup> <= `TotalFrameWPix`<sup>(217)</sup>, `RealHeight`<sup>(215)</sup> <= `TotalFrameHPix`<sup>(217)</sup>).

`TotalFrameWPix`<sup>(217)</sup> and `TotalFrameHPix`<sup>(217)</sup> are similar to `PageWidthPix`<sup>(214)</sup> and `PageHeightPix`<sup>(213)</sup>, but calculated without rounding, i.e. they are equal to `FrameCountX`<sup>(208)</sup> \* `FrameWidthPix`<sup>(210)</sup> x `FrameCountY`<sup>(209)</sup> \* `FrameHeightPix`<sup>(209)</sup>.

If `ScaleImage`<sup>(216)</sup> = `False`, the document page can be printed in the top left corner or in the center of the poster, depending on the `Center`<sup>(207)</sup> property.

See also: `UsePhysicalOffsets`<sup>(217)</sup>.

## Printing and previewing

The main printing methods are `Print`<sup>(219)</sup> and `PrintPages`<sup>(221)</sup>. See also `PrintFrame`<sup>(220)</sup>, `PrintFrames`<sup>(220)</sup>, `PrintFramesEx`<sup>(221)</sup>.

A print preview is displayed by `TSRVPreview`<sup>(223)</sup> component.

You can provide a visual indication of printing progress using `OnSendingToPrinter`<sup>(222)</sup> event.

To make printing output compatible with metafiles, assign `MetafileCompatibility`<sup>(211)</sup> = `True` (and probably, `NoMetafiles`<sup>(211)</sup> = `True`). It may be necessary for some virtual printers

### 1.3.7.1 Properties

#### In TSRVPrint

- `AutoUpdate`<sup>(207)</sup>
- `Center`<sup>(207)</sup>
- `ClipMargins`<sup>(208)</sup>
- `FrameCountX`<sup>(208)</sup>
- `FrameCountY`<sup>(209)</sup>
- ▶ `FrameHeightPix`<sup>(209)</sup>
- ▶ `FrameWidthPix`<sup>(210)</sup>
- `IntegerCount`<sup>(210)</sup>
- `MetafileCompatibility`<sup>(211)</sup>
- `NoMetafiles`<sup>(211)</sup>
- ▶ `OffXPix`<sup>(211)</sup>
- ▶ `OffYPix`<sup>(212)</sup>
- `Orientation`<sup>(212)</sup>
- `PageColCount`<sup>(213)</sup>
- `PageFormat`<sup>(213)</sup>
- `PageHeightPix`<sup>(213)</sup>
- `PageRowCount`<sup>(214)</sup>
- `PageWidthPix`<sup>(214)</sup>
- `PrintMode`<sup>(214)</sup>
- ▶ `RealHeight`<sup>(215)</sup>
- ▶ `RealWidth`<sup>(216)</sup>

- ScaleImage<sup>(216)</sup>
- SRichViewEdit<sup>(217)</sup>
- ▶ TotalFrameHPix<sup>(217)</sup>
- ▶ TotalFrameWPix<sup>(217)</sup>
- UsePhysicalOffsets<sup>(217)</sup>

## Derived from TComponent

- Name
- Tag

### 1.3.7.1.1 TSRVPrint.AutoUpdate

Enables an autoupdate mode.

**property** AutoUpdate: Boolean;

If *True*, TSRVPrint recalculates its parameters when the content of SRichViewEdit<sup>(217)</sup> is changed. Recalculation occurs on OnChange<sup>(104)</sup> and OnPageCountChanged<sup>(114)</sup> events.

If the document was changed when *AutoUpdate=False*, call Update<sup>(222)</sup>.

**Default value:**

*False*

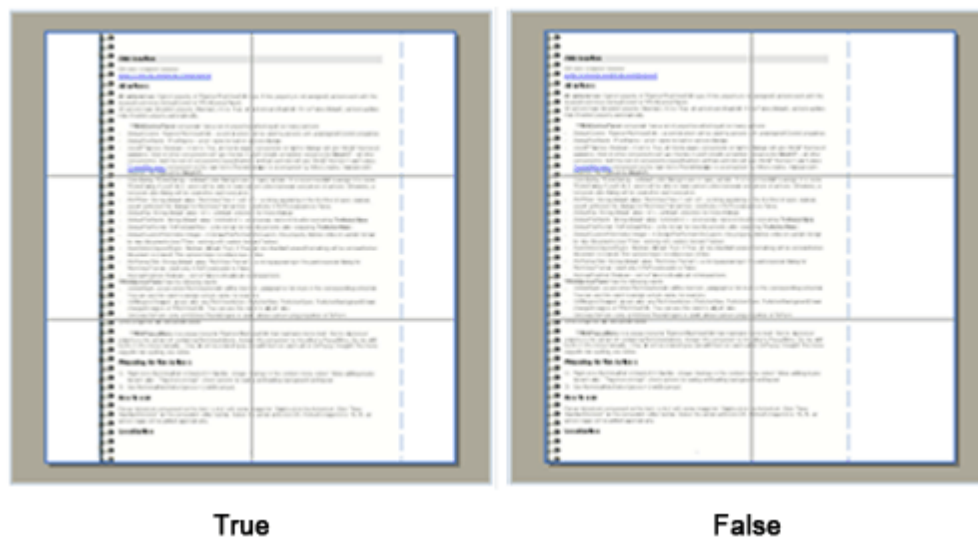
### 1.3.7.1.2 TSRVPrint.Center

In the poster mode: the property places the document image in the center relative to the poster.

In the grid mode: the property places the document pages in the center of the paper sheet.

**property** Center: Boolean;

Center



A posters is printed if PrintMode<sup>(214)</sup> = *srvpPoster*. For posters, this property makes sense only if ScaleImage<sup>(216)</sup> = *False*.

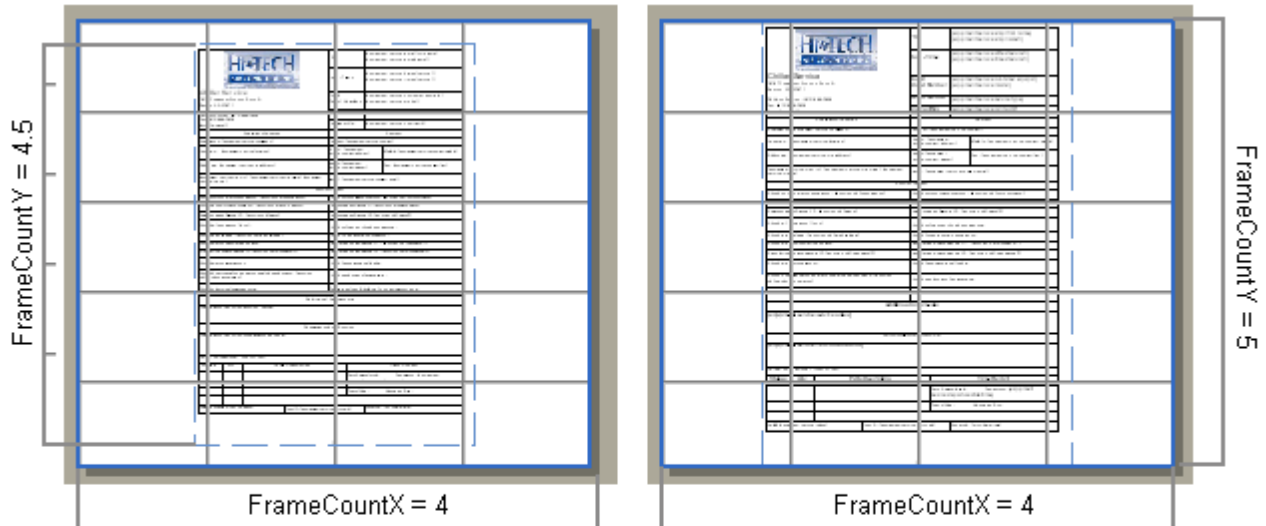
A grid is printed if PrintMode<sup>(214)</sup> = *srvpGrid*.

**Default value:***True***1.3.7.1.3 TSRVPrint.ClipMargins**

Restricts the output area.

**property** ClipMargins: Boolean;If this property is *True*, printing on margins is not allowed (clipped).**Default value:***True***1.3.7.1.4 TSRVPrint.FrameCountX**

Count of frames in a poster, horizontally.

**property** FrameCountX: Extended;Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.Assigning to this property changes the poster width (PageWidthPix<sup>(214)</sup>) and may change RealWidth<sup>(216)</sup> and RealHeight<sup>(215)</sup>.Assigning to this property resets PageFormat<sup>(213)</sup> to *srvfmCustom* and IntegerCount<sup>(210)</sup> to *False* (value may be fractional).**Frame Count**

PageFormat = A1  
 Printer page format = A4  
 Orientation = poLandscape  
 IntegerCount = False  
 ScaleImage = False

**Frame Count**

### 1.3.7.1.5 TSRVPrint.FrameCountY

Count of frames in a poster, vertically.

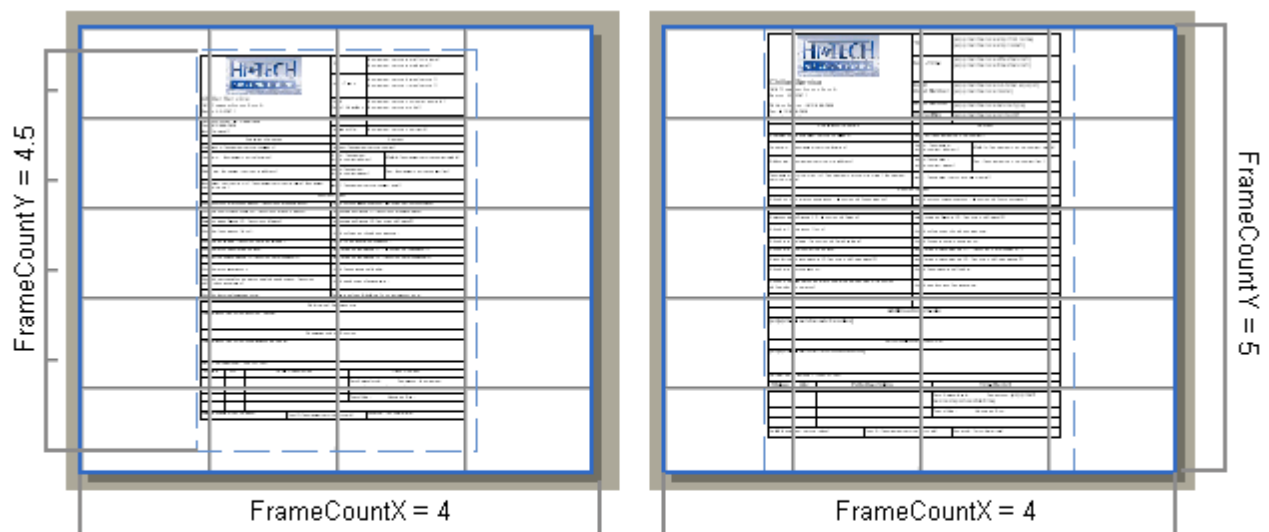
**property** `FrameCountY`: `Extended`;

Posters are printed if `PrintMode`<sup>(214)</sup> = `srvpPoster`.

Assigning to this property changes the poster height (`PageHeightPix`<sup>(213)</sup>) and may change `RealWidth`<sup>(216)</sup> and `RealHeight`<sup>(215)</sup>.

Assigning to this property resets `PageFormat`<sup>(213)</sup> to `srvfmCustom` and `IntegerCount`<sup>(210)</sup> to `False` (value may be fractional).

#### Frame Count



PageFormat = A1  
 Printer page format = A4  
 Orientation = poLandscape  
 IntegerCount = False  
 ScaleImage = False

#### Frame Count

### 1.3.7.1.6 TSRVPrint.FrameHeightPix

Returns the height of one frame of a poster, in pixels.

**property** `FrameHeightPix`: `Integer`;

Posters are printed if `PrintMode`<sup>(214)</sup> = `srvpPoster`.

Each page of a poster is printed as several frames.

If the default printer is set, the printer paper height is used.

**Default value:**

1

**See also:**

- `FrameWidthPix`<sup>(210)</sup>
- the picture in the topic about `TSRVPrint`<sup>(204)</sup>

### 1.3.7.1.7 TSRVPrint.FrameWidthPix

Returns the height of one frame of a poster, in pixels.

**property** FrameWidthPix : Integer;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

Each page of a poster is printed as several frames.

If the default printer is set, the printer paper width is used.

**Default value:**

1

**See also:**

- FrameHeightPix<sup>(209)</sup>
- the picture in the topic about TSRVPrint<sup>(204)</sup>

### 1.3.7.1.8 TSRVPrint.IntegerCount

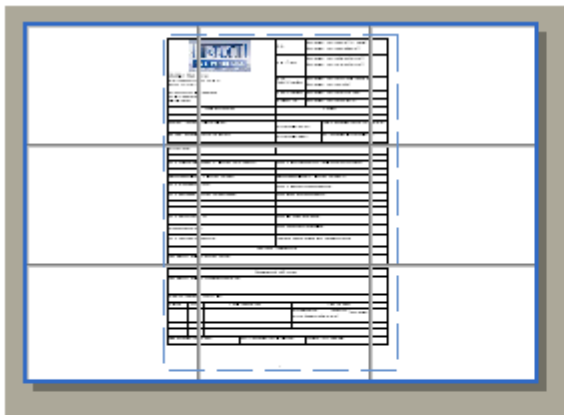
Specifies whether to round values of FrameCountX<sup>(208)</sup> and FrameCountY<sup>(209)</sup> up.

**property** IntegerCount : Boolean;

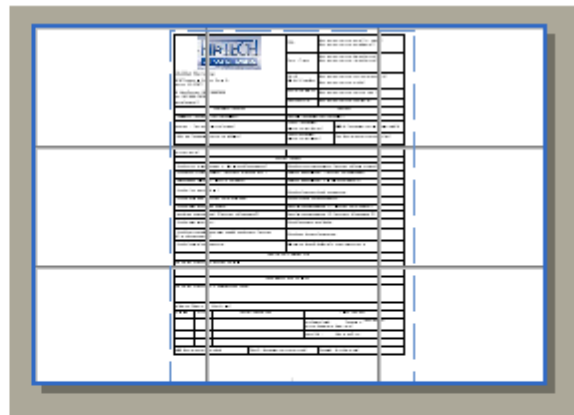
Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

This property allows to use integer numbers of frame counts for a poster.

#### IntegerCount



IntegerCount = False  
FrameCountX = 2.8  
FrameCountY = 2.8



IntegerCount = True  
FrameCountX = 3  
FrameCountY = 3

PageFormat = A1  
Printer page format = A4  
Orientation = poLandscape  
ScaleImage = False

#### IntegerCount Example

**Default value:**

*False*

#### 1.3.7.1.9 TSRVPrint.MetafileCompatibility

Specifies whether the component's printing output is compatible with metafiles.

**property** MetafileCompatibility: Boolean;

If **MetafileCompatibility**=*False*, printing output may be incompatible with metafiles; the component may use font glyph indexes instead of character codes when printing text. Such metafiles will be displayed incorrectly on computers without this font or with another version of this font.

If **MetafileCompatibility**=*True*, the component always uses character codes when printing text, to create metafile-compatible output.

**Default value:**

*False*

**See also**

- TRVAControlPanel.MetafileCompatibility (in the RichViewActions help file)

#### 1.3.7.1.10 TSRVPrint.NoMetafiles

Specifies whether the component prints metafiles as bitmaps.

**property** MetafileCompatibility: Boolean;

If **NoMetafiles**=*False*, metafiles in documents are printed as they are.

If **NoMetafiles**=*True*, metafiles are printed as bitmaps. In this mode, printing quality may be lower (pixelated graphics), however, it may be useful for printing on printers that do not handle embedded metafiles properly.

**Default value:**

*False*

#### 1.3.7.1.11 TSRVPrint.OffXPix

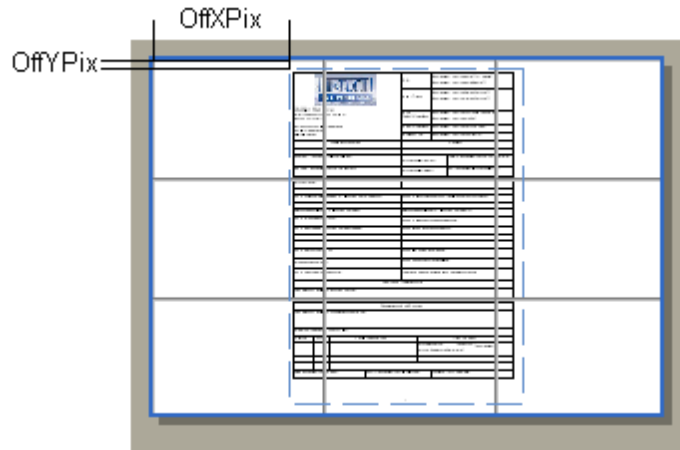
Returns the horizontal shift of the document page on the poster, pixels.

**property** OffXPix: Extended;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

The shift is nonzero if

- the document page is not scaled to the full poster size (ScaleImage<sup>(216)</sup> = *False*) and
- the page is printed at the center of the poster (Center<sup>(207)</sup> = *True*) and
- the page width (RealWidth<sup>(216)</sup>) is smaller than the poster width (PageWidthPix<sup>(214)</sup>).



OffXPix and OffYPix

#### 1.3.7.1.12 TSRVPrint.OffYPix

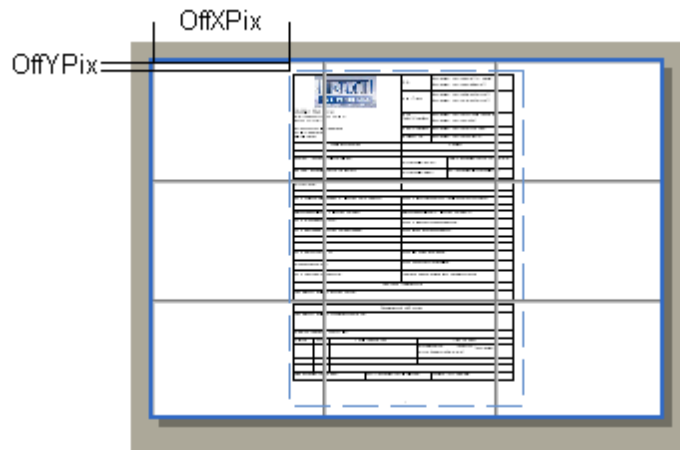
Returns the vertical shift of the document page on the poster, pixels.

**property** OffYPix: Extended;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

The shift is nonzero if

- the document page is not scaled to the full poster size (ScaleImage<sup>(216)</sup> = *False*) and
- the page is printed at the center of the poster (Center<sup>(207)</sup> = *True*) and
- the page height (RealHeight<sup>(215)</sup>) is smaller than the poster height (PageHeightPix<sup>(213)</sup>).



OffXPix and OffYPix

#### 1.3.7.1.13 TSRVPrint.Orientation

Specifies the orientation of frames in a poster.

**property** Orientation: TPrinterOrientation;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

**Default value:**

*poPortrait*



#### 1.3.7.1.14 TSRVPrint.PageColCount

Number of document pages on one paper sheet in horizontal direction, in *srvpGrid* mode.

**property** PageCount: Integer;

This property is used if PrintMode<sup>(214)</sup> = *srvpGrid*.

In this mode, PageRowCount<sup>(214)</sup> x PageColCount document pages are printed on a single paper sheet.

**Default value:**

1

**See also:**

Center<sup>(207)</sup>

#### 1.3.7.1.15 TSRVPrint.PageFormat

The poster's page format.

**property** PageFormat: TSRVPageFormat<sup>(278)</sup>;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

The poster consists of frames, each frame has size of the printer paper.

This is the page format as of the whole poster, containing all frames.

Assigning to this property changes PageWidthPix<sup>(214)</sup> and PageHeightPix<sup>(213)</sup>. Values of FrameCountX<sup>(208)</sup> and FrameCountY<sup>(209)</sup> are recalculated to fit the given format (taking the printer's paper size into account). If IntegerCount<sup>(210)</sup>=True, they are calculated as integer values. This calculation depends on UsePhysicalOffsets<sup>(217)</sup> property.

When you assign value to FrameCountX<sup>(208)</sup>, FrameCountY<sup>(209)</sup>, PageWidthPix<sup>(214)</sup>, or PageHeightPix<sup>(213)</sup>, this property is reset to *srvfmCustom*.

**Default value:**

*srvfmA4*

#### 1.3.7.1.16 TSRVPrint.PageHeightPix

The poster height, pixels

**property** PageHeightPix: Extended;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

A value of this property can be assigned or calculated. The formula PageHeightPix = Ceil(FrameCountY<sup>(209)</sup>\*FrameHeightPix<sup>(209)</sup> is used.

Assigning to this property resets PageFormat<sup>(213)</sup> to *srvfmCustom*.

Assigning any value (except for *srvfmCustom*) to PageFormat<sup>(213)</sup> changes a value of this property.

Assigning to FrameCountY<sup>(209)</sup> changes value of this property.

**See also:**

- PageWidthPix<sup>(214)</sup>
- the picture in the topic about TSRVPrint<sup>(204)</sup>

### 1.3.7.1.17 TSRVPrint.PageRowCount

Number of document pages on one paper sheet in vertical direction, in *srvpGrid* mode.

**property** PageCount: Integer;

This property is used if PrintMode<sup>(214)</sup> = *srvpGrid*.

In this mode, PageRowCount x PageColCount<sup>(213)</sup> document pages are printed on a single paper sheet.

**Default value:**

1

**See also:**

Center<sup>(207)</sup>

### 1.3.7.1.18 TSRVPrint.PageWidthPix

The poster height, pixels

**property** PageWidthPix: Extended;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

A value of this property can be assigned or calculated. The formula PageWidthPix = Ceil(FrameCountX<sup>(208)</sup>\*FrameWidthPix<sup>(210)</sup> is used.

Assigning to this property resets PageFormat<sup>(213)</sup> to *srvfmCustom*.

Assigning any value (except for *srvfmCustom*) to PageFormat<sup>(213)</sup> changes a value of this property.

Assigning to FrameCountX<sup>(208)</sup> changes value of this property.

**See also:**

- PageHeightPix<sup>(213)</sup>
- the picture in the topic about TSRVPrint<sup>(204)</sup>

### 1.3.7.1.19 TSRVPrint.PrintMode

Specifies how the document from SRichViewEdit<sup>(217)</sup> will be printed.

**type**

```
TSRVPrintMode = (srvpAuto, srvpStretchToFit, srvpStretchIfLarger,
  srvpTiles, srvpPoster, srvpGrid);
```

**property** PrintMode: TSRVPrintMode;

| Value           | Meaning  |
|-----------------|--|
| <i>srvpAuto</i> | A printer paper size and orientation are set automatically; if the printer supports the page size specified for the document, it is used (orientation is set automatically too). If the printer does not support the document page size, this mode works like <i>srvpStretchIfLarger</i> mode. |

|                            |   |
|----------------------------|---|
| <i>srvpStretchToFit</i>    | The current printer paper size and orientation are used; on printing, the document is stretched proportionally to fit the page size and the orientation chosen for the printer.   |
| <i>srvpStretchIfLarger</i> | This mode is like <i>srvpStretchToFit</i> if the document page size is larger than the printer paper size. Otherwise, the document is printed at the top left corner of paper sheet.  |
| <i>srvpTiles</i>           | The current printer paper size and orientation are used. If possible, prints several copies of the document page on the same paper sheet; for example, if the document page size is A6, and the printer page size is A4, it prints 4 copies (if orientations are the same).<br><br>If document page is larger than paper (possibly because of different orientation), the component shrinks it to fit the paper size, and tries to print several copies of this shrunk document page. |
| <i>srvpPoster</i>          | Allows printing posters (one page on several paper sheets).<br><br>It's recommended to assign <code>UsePhysicalOffsets<sup>(217)</sup> = False</code> in this mode.   |
| <i>srvpGrid</i>            | The current printer paper size and orientation are used.<br><br><code>PageRowCount<sup>(214)</sup> x PageColCount<sup>(213)</sup></code> document pages are printed on each paper sheet as a grid. These pages are scaled proportionally to fit; unlike <i>srvpTiles</i> mode, these pages are not copies of the same page.   |

**Default value:***srvpPoster***1.3.7.1.20 TSRVPrint.RealHeight**

Returns the height of the scaled page of the source document.

**property** `RealHeight`: `Extended`;Posters are printed if `PrintMode(214) = srvpPoster`.If `ScaleImage(216) = True`, each page of the source document is scaled to the full poster size without keeping proportions, so `RealHeight = PageHeightPix(213)`.If `ScaleImage(216) = False`, each page is scaled proportionally to the maximal size that fits in `TotalFrameWPix(217) x TotalFrameHPix(217)`, so `RealHeight ≤ TotalFrameHPix(217)`.**See also:**

- `RealWidth(216)`
- the picture in the topic about `TSRVPrint(204)`

### 1.3.7.1.21 TSRVPrint.RealWidth

Returns the width of the scaled page of the source document.

**property** RealWidth: Extended;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

If ScaleImage<sup>(216)</sup> = *True*, each page of the source document is scaled to the full poster size without keeping proportions, so RealWidth=PageWidthPix<sup>(214)</sup>.

If ScaleImage<sup>(216)</sup> = *False*, each page is scaled proportionally to the maximal size that fits in TotalFrameWPix<sup>(217)</sup> x TotalFrameHPix<sup>(217)</sup>, so RealWidth<=TotalFrameWPix<sup>(217)</sup>.

**See also:**

- RealHeight<sup>(215)</sup>
- the picture in the topic about TSRVPrint<sup>(204)</sup>

### 1.3.7.1.22 TSRVPrint.ScaleImage

Specifies whether the source page image must be stretched to the full poster size.

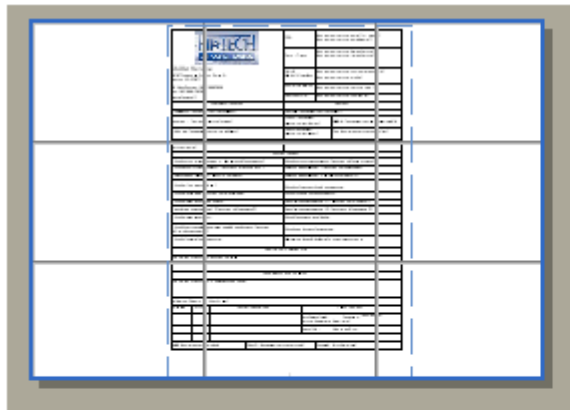
**property** ScaleImage: Boolean;

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

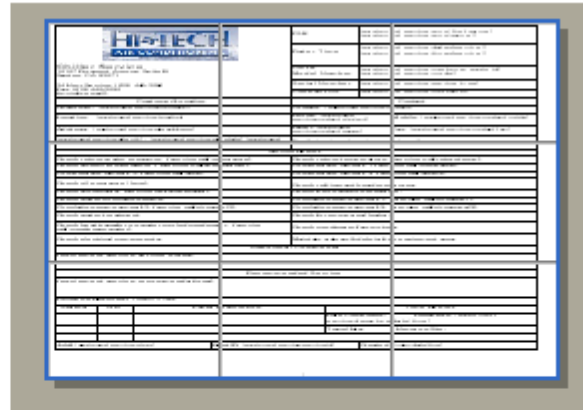
If ScaleImage = *True*, pages are scaled to the full poster size (RealWidth<sup>(216)</sup> will be PageWidthPix<sup>(214)</sup>, RealHeight<sup>(215)</sup> will be PageHeightPix<sup>(213)</sup>), without keeping the original proportions.

If ScaleImage = *False*, pages are scaled proportionally to the maximal size that fits in TotalFrameWPix<sup>(217)</sup> x TotalFrameHPix<sup>(217)</sup> (RealWidth<sup>(216)</sup> will be <=TotalFrameWPix<sup>(217)</sup>, RealHeight<sup>(215)</sup> <=TotalFrameHPix<sup>(217)</sup>).

#### ScaleImage



ScaleImage = False



ScaleImage = True

PageFormat = A1  
Printer page format = A4  
Orientation = poLandscape  
FrameCountX = 3  
FrameCountY = 3

#### ScaleImage Example

**Default value:**

*False*

#### 1.3.7.1.23 TSRVPrint.SRichViewEdit

Specifies a link to the editor containing the document for printing.

```
property SRichViewEdit: TSRichViewEdit(40);
```

TSRVPrint component can print only if an editor is specified.

#### 1.3.7.1.24 TSRVPrint.TotalFrameHPix

Returns  $\text{FrameCountY}^{(209)} * \text{FrameHeightPix}^{(209)}$ .

```
property TotalFrameHPix: Extended;
```

Posters are printed if  $\text{PrintMode}^{(214)} = \text{srvpPoster}$ .

If  $\text{ScaleImage}^{(216)} = \text{False}$ , pages of the source document are scaled proportionally to the maximal size that fits in  $\text{TotalFrameWPix}^{(217)} \times \text{TotalFrameHPix}$ .

#### 1.3.7.1.25 TSRVPrint.TotalFrameWPix

Returns  $\text{FrameCountX}^{(208)} * \text{FrameWidthPix}^{(210)}$ .

```
property TotalFrameWPix: Extended;
```

Posters are printed if  $\text{PrintMode}^{(214)} = \text{srvpPoster}$ .

If  $\text{ScaleImage}^{(216)} = \text{False}$ , pages of the source document are scaled proportionally to the maximal size that fits in  $\text{TotalFrameWPix} \times \text{TotalFrameHPix}^{(217)}$ .

#### 1.3.7.1.26 TSRVPrint.UsePhysicalOffsets

Specifies whether the component attempts to print on "physical margins".

```
property UsePhysicalOffsets: Boolean;
```

Physical margins are areas at the left, top, right and bottom sides of paper where the printer cannot print.

If *True*, TSRVPrint does not print on physical margins. It also affects how a number of frames for a poster is calculated. For example, you want to print A4 poster on A4 paper. If you set the current printer paper size to A4, and set  $\text{PageFormat}^{(213)}$  to *srvfmA4*, you may expect that the poster will be printed on 1 paper sheet. But you will find that a poster is printed on 2 x 2 paper sheets, because  $\text{FrameCountX}^{(208)}$  and  $\text{FrameCountY}^{(209)}$  are greater than 1. It happens because the poster size is a pure A4 size, and the printer's page size is A4 minus physical offsets, so the poster does not fit the paper. If you want to define a poster size more precisely, assign  $\text{FrameCountX}^{(208)}$  and  $\text{FrameCountY}^{(209)}$  properties directly, instead of changing  $\text{PageFormat}^{(213)}$ .

If *False*, TSRVPrint prints even on physical margins. In the example above (A4 poster on A4 paper), a poster will be printed on one paper sheet. But this value is highly not recommended for posters, because everything printed on these margins will be blank. This value is ok for non-poster printing (because usually document margins are larger than physical margins).

**Default value:**

*True*

## 1.3.7.2 Methods

### In TSRVPrint

BeginUpdate<sup>(218)</sup>  
 CheckMargin<sup>(218)</sup>  
 EndUpdate<sup>(219)</sup>  
 OptimalOrientation<sup>(219)</sup>  
 PaperCodeToFormatPage<sup>(219)</sup>  
 Print<sup>(219)</sup>  
 PrintFrame<sup>(220)</sup>  
 PrintFrames<sup>(220)</sup>  
 PrintFramesEx<sup>(221)</sup>  
 PrintPages<sup>(221)</sup>  
 Update<sup>(222)</sup>

#### 1.3.7.2.1 TSRVPrint.BeginUpdate

Disables recalculation of parameters until EndUpdate<sup>(219)</sup> is called.

**procedure** BeginUpdate;

Each call to BeginUpdate must be followed by the call to EndUpdate<sup>(219)</sup>. Calls may be nested.

Between calls of BeginUpdate and EndUpdate<sup>(219)</sup>, assignments to TSRVPrint's properties do not invoke recalculations of other properties, and OnUpdateData<sup>(222)</sup> event is not called. For example, assignment to FrameCountX<sup>(208)</sup> will not lead to recalculation of PageWidthPix<sup>(214)</sup>.

#### 1.3.7.2.2 TSRVPrint.CheckMargin

Compares margins with minimal possible values and returns adjusted values.

**function** CheckMargin(**var** NewLeft, NewTop, NewRight, NewBottom: Integer;  
 ALeft, ATop, ARight, ABottom: Integer) : Boolean;

This methods compares the margins specified in **ALeft**, **ATop**, **ARight**, **ABottom** parameters with minimal margins supported by the current printer.

The results are written to **NewLeft**, **NewTop**, **NewRight**, **NewBottom** parameters. If the corresponding margin is too small, a minimal supported value is returned. Otherwise, the value of the original parameter is returned.

All values are in pixels.

Any of the input parameters (**ALeft**, **ATop**, **ARight**, **ABottom**) can be equal to -1. Such parameters are ignored.

This method can be used in TSRichViewEdit.OnMarginsChanged<sup>(111)</sup> event.

### 1.3.7.2.3 TSRVPrint.EndUpdate

Enables automatic recalculation of parameters that was disabled by BeginUpdate<sup>(218)</sup>.

**procedure** EndUpdate;

Each call to BeginUpdate<sup>(218)</sup> must be followed by call to EndUpdate. Calls may be nested.

If EndUpdate is called as many times as BeginUpdate<sup>(218)</sup>, the last call performs all necessary recalculation of properties (by calling Update<sup>(222)</sup>).

### 1.3.7.2.4 TSRVPrint.OptimalOrientation

Returns an optimal frame orientation.

**function** OptimalOrientation : TPrinterOrientation;

Returns which value of Orientation<sup>(212)</sup> would be the best (less count of frames in a poster).

Posters are printed if PrintMode<sup>(214)</sup> = *srvpPoster*.

### 1.3.7.2.5 TSRVPrint.PaperCodeToPageFormat

This method converts DMPAPER\_\*\*\* values (defined in the Printers unit) to TSRVPageFormat<sup>(278)</sup> values.

**function** PaperCodeToPageFormat (PaperSizeCode: Integer): TSRVPageFormat<sup>(278)</sup>;

**PaperSizeCode** must be one of constants like DMPAPER\_A2, DMPAPER\_A3, DMPAPER\_A4, etc.

### 1.3.7.2.6 TSRVPrint.Print

Prints the whole document.

**procedure** Print (Title: **String**; Copies: Integer; Collate: Boolean;  
PageSet: TRVPageSet = rvpsAll; Ascending: Boolean = True);

The method prints the document from SRichViewEdit<sup>(217)</sup> according to PrintMode<sup>(214)</sup>.

The properties SRichViewEdit<sup>(217)</sup>.MinPrintedItemNo<sup>(56)</sup> and MaxPrintedItemNo<sup>(55)</sup> are taken into account.

**Title** – document title for the Print Manager.

**Copies** – count of copies to print.

**Collate** – if more than one copy is printed, defines whether to print them together or not (if **Collate**, pages are printed like 1, 2, 3, 1, 2, 3. If not **Collate**, pages are printed like 1, 1, 2, 2, 3, 3).

**PageSet** – set of pages for printing (all/odd/even).

**Ascending** switches normal/reverse order of printing (*True*: 1, 2, 3; *False*: 3, 2, 1).

Print (Title, Copies, Collate, PageSet, Ascending) is equivalent to PrintPages<sup>(221)</sup> (1, SRichViewEdit<sup>(57)</sup>.PageCount<sup>(57)</sup>, Title, Copies, Collate, PageSet, Ascending).

**About the grid printing mode (multiple pages on the same paper sheet)**

Printing odd and even pages, and printing in reverse order are not supported if `PrintMode`<sup>(214)</sup> = `srvpGrid`.

### About the poster printing mode (each page on the multiple paper sheets)

For posters (`PrintMode`<sup>(214)</sup> = `srvpPoster`), if **Collate** = `False`, printing order depends on the printer capabilities. In this mode, if **Collate** = `False`, pages are never collated. If the printer itself supports printing multiple copies, fragments of each page are not collated as well (page 1 fragment 1, page 1 fragment 1, page 1 fragment 2, page 1 fragment 2, then the same for page 2). But if the printer does not support multiple copies, TSRVPrint collates page fragments (page 1 fragment 1, page 1 fragment 2, page 1 fragment 1, page 1 fragment 2, then the same for page 2).

For posters, fragments of each page are always printed in normal order, **Ascending** may only reverse the order for pages of the source editor (`SRichViewEdit`<sup>(217)</sup>). "Odd/even pages" means pages of the source editor; all fragments of pages specified in **PageSet** are always printed. Because of this, you cannot use the procedure described below for printing posters on both sides of paper sheets.

### How to print on two sides of paper:

1. Print odd pages in normal order.
2. Flip the sheets over, then reinsert them into the printer. If page count is odd, do not insert the last page.
3. Print even pages in reverse order.

### See also:

- `OnSendingToPrinter`<sup>(222)</sup>

#### 1.3.7.2.7 TSRVPrint.PrintFrame

Prints the specified frame of the specified page, in the current printing job.

```
function PrintFrame(PageNo, FrameIndex : Integer) : Boolean;
```

**PageNo** – index of the page (from 1).

**FrameIndex** – index of the frame to print, in range from 1 to `Ceil(FrameCountX`<sup>(208)</sup> `*Ceil(FrameCountY`<sup>(209)</sup>).

This method does not start a new printing job.

Posters are printed if `PrintMode`<sup>(214)</sup> = `srvpPoster`. In other modes, the method prints the specified page (and **FrameIndex** must be 1).

This method is rarely needed. The main printing methods are `Print`<sup>(219)</sup> and `PrintPages`<sup>(221)</sup>.

The method calls `OnSendingToPrinter`<sup>(222)</sup> (`SRichViewEdit`<sup>(40)</sup>, **PageNo**, **FrameIndex**, *rvpsProceeding*).

### Return value:

*True* if the frame was printed. *False* if **FrameIndex** is not in the proper range.

#### 1.3.7.2.8 TSRVPrint.PrintFrames

Prints one page.

```
procedure PrintFrames(Title: String; PageNo: Integer);
```

**Title** – document title for the Print Manager.



**PageNo** – index of the page (from 1).

This method starts a new printing job, calls `PrintFramesEx`<sup>(221)</sup>, finishes the printing job.

This method is rarely needed. The main printing methods are `Print`<sup>(219)</sup> and `PrintPages`<sup>(221)</sup>.

#### 1.3.7.2.9 TSRVPrint.PrintFramesEx

Prints one page in the current printing job.

```
procedure PrintFramesEx(PageNo: Integer);
```

**PageNo** – index of the page (from 1).

If `PrintMode`<sup>(214)</sup> = *srvpPoster*, the method prints all frames of this page by calling `PrintFrame`<sup>(220)</sup>. For other modes, it prints the specified page.

This method is rarely needed. The main printing methods are `Print`<sup>(219)</sup> and `PrintPages`<sup>(221)</sup>.

#### 1.3.7.2.10 TSRVPrint.PrintPages

Prints the specified range of pages.

```
procedure PrintPages(firstPgNo, lastPgNo: Integer; Title: String;
  Copies: Integer; Collate: Boolean;
  PageSet: TRVPageSet = rvpsAll; Ascending: Boolean = True);
```

The method prints the document from `SRichViewEdit`<sup>(217)</sup> according to `PrintMode`<sup>(214)</sup>.

The properties `SRichViewEdit`<sup>(217)</sup>.`MinPrintedItemNo`<sup>(56)</sup> and `MaxPrintedItemNo`<sup>(55)</sup> are taken into account.

**firstPgNo** – index of the first page to print (from 1).

**lastPgNo** – index of the last page to print (from 1).

**Title** – document title for the Print Manager.

**Copies** – count of copies to print.

**PageSet** – set of pages for printing (all/odd/even).

**Ascending** switches normal/reverse order of printing (*True*: 1, 2, 3; *False*: 3, 2, 1). Printing in reverse order is not supported if `PrintMode`<sup>(214)</sup> = *srvpGrid*.

**Collate** – if more than one copy is printed, defines whether to print them together or not (if **Collate**, pages are printed like 1, 2, 3, 1, 2, 3. If not **Collate**, pages are printed like 1, 1, 2, 2, 3, 3).

In the topic about the `Print`<sup>(219)</sup> method, you can read the notes about using these parameters in grid and poster modes.

#### How to print on two sides of paper:

1. Print odd pages in normal order.
2. Flip the sheets over, then reinsert them into the printer. If page count is odd, do not insert the last page.
3. Print even pages in reverse order.

#### See also:

- `OnSendingToPrinter`<sup>(222)</sup>

### 1.3.7.2.11 TSRVPrint.Update

Performs calculations necessary for printing or previewing.

**procedure** Update;

Normally, all necessary calculations are performed automatically, when you modify properties of this component.

An explicit call is necessary in the following cases:

- the current printer or its properties were changed;
- a document in `SRichViewEdit`<sup>(217)</sup> was changed (and `AutoUpdate`<sup>(207)</sup>=`False`, or this change cannot be detected by `TSRVPrint`).

This method is called by `EndUpdate`<sup>(219)</sup>, when it enables recalculations disabled by `BeginUpdate`<sup>(218)</sup>.

This method calls `OnUpdateData`<sup>(222)</sup> event.

## 1.3.7.3 Events

### In TSRVPrint

- `OnSendingToPrinter`<sup>(222)</sup>
- `OnUpdateData`<sup>(222)</sup>

#### 1.3.7.3.1 TSRVPrint.OnSendingToPrinter

Occurs on printing.

**type**

```
TSRVPrintingEvent = procedure (Sender: TSRichViewEdit;
    PageCompleted, FrameCompleted: Integer;
    Step: TRVPrintingStep) of object;
```

**property** OnSendingToPrinter: TSRVPrintingEvent;

This event is called by the printing methods.

The main printing methods are `Print`<sup>(219)</sup> and `PrintPages`<sup>(221)</sup>. Initially, they call this event with the following parameters: (`SRichViewEdit`<sup>(217)</sup>, 0, 0, `rvpsStarting`). Then they call this event after printing each frame (non-poster modes assume one frame per page), with **Step**=`rvpsProceeding`. Finally, they call it with the parameters: (`SRichViewEdit`<sup>(217)</sup>, 0, 0, `rvpsFinished`).

**PageCompleted** (index of page containing the completed frame, from 1) and **FrameCompleted** (index of the completed frame, from 1) are used if **Step**=`rvpsProceeding`.

**See also:**

- `TSRichViewEdit.OnPrinting`<sup>(116)</sup>

#### 1.3.7.3.2 TSRVPrint.OnUpdateData

Occurs when `TSRVPrint`'s properties are recalculated according the the changes of other properties, changes in document in `SRichViewEdit`<sup>(217)</sup>, or a call to `Update`<sup>(222)</sup>.

**property** OnUpdateData: TNotifyEvent;

### 1.3.8 TSRVPreview

TSRVPreview displays a print preview for TSRVPrint<sup>(204)</sup>.

**Unit** SRVPreview.

#### Syntax

```
TSRVPreview = class (TCustomRVPrintPreview)
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TRVScroller*  
*TCustomRVPrintPreview*

#### Description

The source TSRVPrint<sup>(204)</sup> component is specified in SRVPrint<sup>(227)</sup> property. TSRVPreview displays a single page of the source document, specified in PageNo<sup>(226)</sup> property. This page may consist of several frames (paper sheets), if a poster is printed.

When SRVPrint<sup>(227)</sup> prints a poster, this component displays all frames (paper sheets) comprising this page. A border around all frames is shown using BorderColor<sup>(225)</sup> and BorderWidth<sup>(225)</sup> properties. Borders between frames are shown using FrameBorderColor<sup>(225)</sup> and FrameBorderWidth<sup>(225)</sup>. A position of the original page is shown using PageBorderColor<sup>(226)</sup>, PageBorderWidth<sup>(226)</sup>, PageBorderStyle<sup>(226)</sup> properties.

When SRVPrint prints in another mode, this component displays a single paper sheet. A border around it is shown using BorderColor<sup>(225)</sup> and BorderWidth<sup>(225)</sup> properties. A position of the original page is shown using PageBorderColor<sup>(226)</sup>, PageBorderWidth<sup>(226)</sup>, PageBorderStyle<sup>(226)</sup> properties (one sheet may contain several pages, if tiles are printed).

Paper sheets inside the main border (drawn using BorderColor<sup>(225)</sup> and BorderWidth<sup>(225)</sup>) drop a shadow with the parameters ShadowColor and ShadowWidth. The rest of the component's window is filled with Color.

#### 1.3.8.1 Properties

##### In TSRVPreview

- BorderColor<sup>(225)</sup>
- BorderWidth<sup>(225)</sup>
- FrameBorderColor<sup>(225)</sup>
- FrameBorderWidth<sup>(225)</sup>
- PageBorderColor<sup>(226)</sup>
- PageBorderStyle<sup>(226)</sup>
- PageBorderWidth<sup>(226)</sup>
- PageNo<sup>(226)</sup>

- SRVPrint<sup>227</sup>

### Derived from TCustomRVPrintPreview

---

- ClickMode
- Color
- MarginsPen
- PrintableAreaPen
- ShadowColor
- ShadowWidth
- ZoomInCursor
- ZoomMode
- ZoomOutCursor
- ZoomPercent

### Derived from TRVScroller

---

- BorderStyle
- ▶ HScrollMax
- HScrollPos
- HScrollVisible
- ▶ InplaceEditor (not used in this component)
- NoVScroll
- Tracking
- UseXPThemes
- ▶ VScrollMax
- VScrollPos
- VScrollVisible
- WheelStep

### Derived from TCustomControl

---

- Align
- Anchors
- Constraints
- Ctl3D
- Height
- HelpContext
- Hint
- ParentCtl3D
- PopupMenu
- ShowHint
- TabOrder
- TabStop
- Touch (D2010+)
- Visible
- Width

#### 1.3.8.1.1 TSRVPreview.BorderColor

Specifies the color of border around the paper.

**property** BorderColor: TColor;

When previewing posters, this is the color of border around all frames (all paper sheets).

When previewing in other modes, this is the color of border around the paper sheet.

The border's width is BorderWidth<sup>(225)</sup>.

**Default value:**

*clHighlight*

#### 1.3.8.1.2 TSRVPreview.BorderWidth

Specifies the width of border around the paper.

**property** BorderWidth: Integer;

When previewing posters, this is the width of border around all frames (all paper sheets).

When previewing in other modes, this is the width of border around the paper sheet.

The border's color is BorderColor<sup>(225)</sup>.

**Default value:**

2

#### 1.3.8.1.3 TSRVPreview.FrameBorderColor

Specifies the color of borders between poster frames.

**property** FrameBorderColor: TColor;

When previewing posters, this is a color of lines between the poster's frames (paper sheets).

Not used in other modes.

The width of these lines is FrameBorderWidth<sup>(225)</sup>.

**Default value:**

*clBtnShadow*

#### 1.3.8.1.4 TSRVPreview.FrameBorderWidth

Specifies the width of borders between poster frames.

**property** FrameBorderWidth: Integer;

When previewing posters, this is a width of lines between the poster's frames (paper sheets).

Not used in other modes.

The color of these lines is FrameBorderColor<sup>(225)</sup>.

**Default value:**

1

#### 1.3.8.1.5 TSRVPreview.PageBorderColor

Specifies the color of border around the page of the source document.

**property** PageBorderColor: TColor;

This border shows the location of the page of the source document on the paper sheet(s).

When previewing a poster, this page can occupy several frames (paper sheets).

The width of this border is PageBorderWidth<sup>(226)</sup>, its style is PageBorderStyle<sup>(226)</sup>.

**Default value:**

\$00CD9165

#### 1.3.8.1.6 TSRVPreview.PageBorderStyle

Specifies the style of border around the page of the source document.

**property** PageBorderStyle: TPenStyle;

This border shows the location of the page of the source document on the paper sheet(s).

When previewing a poster, this page can occupy several frames (paper sheets).

The width of this border is PageBorderWidth<sup>(226)</sup>, its color is PageBorderColor<sup>(226)</sup>.

**Default value:**

psDash

#### 1.3.8.1.7 TSRVPreview.PageBorderWidth

Specifies the width of border around the page of the source document.

**property** PageBorderWidth: Integer;

This border shows the location of the page of the source document on the paper sheet(s).

When previewing a poster, this page can occupy several frames (paper sheets).

The style of this border is PageBorderStyle<sup>(226)</sup>, its color is PageBorderColor<sup>(226)</sup>.

**Default value:**

1

#### 1.3.8.1.8 TSRVPreview.PageNo

Specifies the index of the page to display.

**property** PageNo: Integer;

This value must be in range 1..SRVPrint<sup>(227)</sup>.SRichViewEdit<sup>(217)</sup>.PageCount<sup>(57)</sup>.

**See also:**

- First<sup>(227)</sup>
- Prev<sup>(227)</sup>
- Next<sup>(227)</sup>
- Last<sup>(227)</sup>

### 1.3.8.1.9 TSRVPreview.SRVPrint

Contains a link to a TSRVPrint<sup>(204)</sup> component.

**property** SRVPrint: TSRVPrint<sup>(204)</sup>;

This property provides a connection to TSRVPrint<sup>(204)</sup> component.

TSRVPreview component can display a preview only if the this TSRVPrint<sup>(204)</sup> component is ready.

## 1.3.8.2 Methods

### In TSRVPreview

---

First<sup>(227)</sup>

Last<sup>(227)</sup>

Next<sup>(227)</sup>

Prev<sup>(227)</sup>

### 1.3.8.2.1 TSRVPreview.First

Switches the current page (PageNo<sup>(226)</sup>) to the first page.

**procedure** First;

### 1.3.8.2.2 TSRVPreview.Last

Switches the current page (PageNo<sup>(226)</sup>) to the last page.

**procedure** Last;

### 1.3.8.2.3 TSRVPreview.Next

Switches the current page (PageNo<sup>(226)</sup>) to the next page.

**procedure** Next;

### 1.3.8.2.4 TSRVPreview.Prev

Switches the current page (PageNo<sup>(226)</sup>) to the previous page.

**procedure** Prev;

## 1.3.8.3 Events

### In TSRVPreview

---

■ OnAfterPaint<sup>(228)</sup>

■ OnBeforePaint<sup>(228)</sup>

### Derived from TCustomRVPrintPreview

---

■ OnHScrolled

■ OnVScrolled

■ OnZoomChanged

## Derived from TCustomControl

### ■ OnGesture (D2010+)

#### 1.3.8.3.1 TSRVPreview.OnAfterPaint

Occurs when the component paints itself, after all the default painting.

**property** OnAfterPaint: TSRVOnPaint<sup>(270)</sup>;

This event allows drawing additional information in this component.

**See also:**

- OnBeforePaint<sup>(228)</sup>

#### 1.3.8.3.2 TSRVPreview.OnBeforePaint

Occurs when the component paints itself, before painting the content of pages.

**property** OnBeforePaint: TSRVOnPaint<sup>(270)</sup>;

This event allows drawing additional information in this component.

This event allows drawing only beyond the area specified in **ARect** parameter, because any drawing inside this area will be overridden. This event may be considered redundant, because OnAfterPaint<sup>(228)</sup> allows drawing both inside and outside of this area.

Since a clipping region is **ARect**, you need to create a new clipping region to draw outside **ARect**.

**Example:**

```
procedure TForm1.SRVPreview1BeforePaint(Sender: TObject; ACanvas: TCanvas;
  ARect: TRect);
var
  HRgn: Cardinal;
begin
  // creating a clipping region for drawing anywhere in SRVPreview1
  HRgn := CreateRectRgn(0, 0, SRVPreview1.Width, SRVPreview1.Height);
  SelectClipRgn(ACanvas.Handle, HRgn);
  // <drawing here>
  DeleteObject(HRgn);
  // restoring the original region
  HRgn := CreateRectRgn(ARect.Left, ARect.Top, ARect.Right, ARect.Bottom);
  SelectClipRgn(ACanvas.Handle, HRgn);
  DeleteObject(HRgn);
end;
```

### 1.3.9 TSRVSkinManager

TSRVSkinManager contains a collection of skins and allows applying them to visual components from the ScaleRichView package.

**Unit** SRVSkinManager.

**Syntax**

```
TSRVSkinManager = class (TComponent)
```



## Hierarchy

*TObject*

*TPersistent*

*TComponent*

## Description

This component can apply skins to:

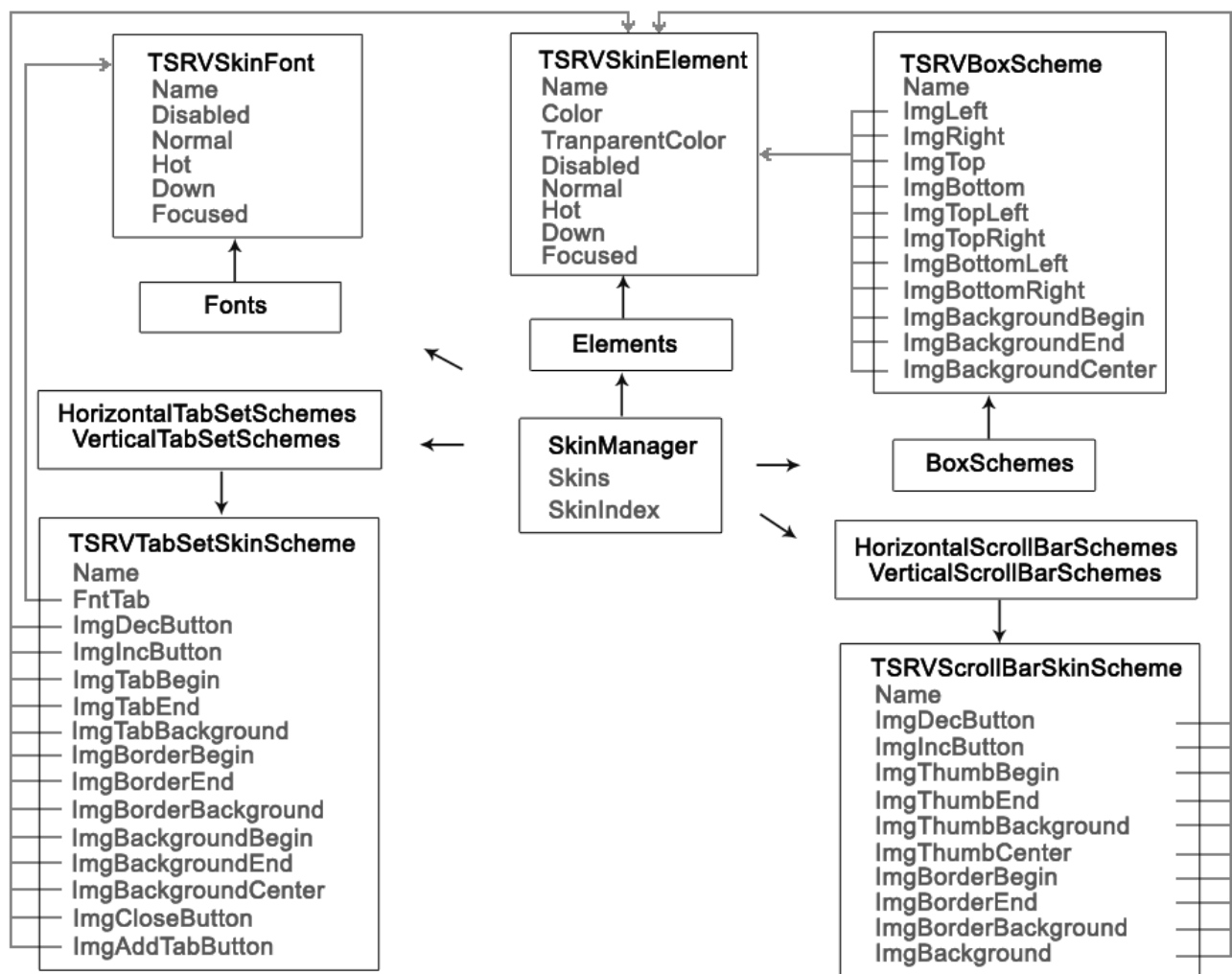
- scrollbars of `TSRichViewEdit`<sup>(40)</sup> and `TDBSRichViewEdit`<sup>(169)</sup>;
- `TSRVScrollBar`<sup>(243)</sup>;
- `TSRVTabSet`<sup>(249)</sup>.

## Properties

`Skins`<sup>(230)</sup> is a collection of skins. `SkinIndex`<sup>(230)</sup> allows choosing a skin from this collection. This skin is returned in the `CurrentSkin`<sup>(230)</sup> property and applied to all controls linked to this skin manager.

## Scheme

This scheme shows a skin manager component and its properties.



The main properties of each skin (i.e. of each item in Skins<sup>(230)</sup> collection) are schemes:

- HorizontalScrollBarSchemes<sup>(232)</sup> – scheme for horizontal scrollbars;
- VerticalScrollBarSchemes<sup>(233)</sup> – scheme for vertical scrollbars;
- HorizontalTabSetSchemes<sup>(232)</sup> – scheme for horizontal tab sets;
- VerticalTabSetSchemes<sup>(233)</sup> – scheme for vertical tab sets;
- BoxSchemes<sup>(232)</sup> – scheme for rectangular areas of different controls.

Each scheme contains references to Elements<sup>(232)</sup> (collection of images) and Fonts<sup>(232)</sup> (collection of fonts).

### 1.3.9.1 Properties

#### In TSRVSkinManager

- ▶ CurrentSkin<sup>(230)</sup>
- SkinIndex<sup>(230)</sup>
- Skins<sup>(230)</sup>

##### 1.3.9.1.1 TSRVSkinManager.CurrentSkin

Returns the active skin.

**property** CurrentSkin: TSRVSkin<sup>(231)</sup>;

If SkinIndex<sup>(230)</sup> is in the range 0..Skins<sup>(230)</sup>.Count-1, this property returns Skins<sup>(230)</sup>[SkinIndex<sup>(230)</sup>]. Otherwise, it returns *nil*.

##### 1.3.9.1.2 TSRVSkinManager.SkinIndex

Specifies the active skin.

**property** SkinIndex: Integer;

Zero or positive value defines the index in Skins<sup>(230)</sup> collection.

-1 means default (no-skin) mode.

**Default value:**

-1

##### 1.3.9.1.3 TSRVSkinManager.Skins

A collection of skins.

**property** Skins: TSRVSkinCollection<sup>(235)</sup>;

**See also:**

SkinIndex<sup>(230)</sup>

### 1.3.9.2 Classes of properties

#### Classes of TSRVSkinManager<sup>(228)</sup> Properties

TSRVSkinCollection<sup>(235)</sup> is collection of TSRVSkin<sup>(231)</sup> items. This is a type of Skins<sup>(230)</sup> property.

## Classes of TSRVSkin<sup>(231)</sup> Properties

TSRVSkin has basic properties of the following classes:

- TSRVSkinElementCollection<sup>(238)</sup> – collection of TSRVSkinElement<sup>(237)</sup> items; a type of Elements<sup>(232)</sup> property;
- TSRVSkinFontCollection<sup>(236)</sup> – collection of TSRVSkinFont<sup>(235)</sup> items; a type of Fonts<sup>(232)</sup> property;

All other properties of TSRVSkin contains indexes of items in the collections above (an index in Elements, if it needs to fill some area; index in Fonts, if it needs to draw some text).

TSRVSkin has scheme properties of the following classes:

- TSRVBoxSchemeCollection<sup>(234)</sup> – collection of TSRVBoxScheme<sup>(233)</sup> items; a type of BoxSchemes<sup>(232)</sup> property; schemes defining skins for different types of controls;
- TSRVScrollBarSkinSchemeCollection<sup>(240)</sup> – collection of TSRVScrollBarSkinScheme<sup>(238)</sup> items; a type of HorizontalScrollBarSchemes<sup>(232)</sup> and VerticalScrollBarSchemes<sup>(233)</sup> properties; schemes defining skins for TSRVScrollBar<sup>(243)</sup>;
- TSRVTabSetSkinSchemeCollection<sup>(242)</sup> – collection of TSRVTabSetSkinScheme<sup>(241)</sup> items; a type of HorizontalTabSetSchemes<sup>(232)</sup> and VerticalTabSetSchemes<sup>(233)</sup> properties; schemes defining skins for TSRVTabSet<sup>(249)</sup>;

### 1.3.9.2.1 TSRVSkin

TSRVSkin contains properties defining a visual appearance of skinnable controls.

This is a class of items in the collection TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>.

**Unit** SRVSkinManager.

#### Syntax

```
TSRVSkin = class (TCollectionItem)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*.

#### Properties

Name<sup>(233)</sup> is the skin name.

Elements<sup>(232)</sup> is a collection of images, Fonts<sup>(232)</sup> is a collection of fonts.

Other properties are collections of "schemes". A scheme defines which Elements<sup>(232)</sup> and Fonts<sup>(232)</sup> are used for different parts of skinnable objects:

- HorizontalScrollBarSchemes<sup>(232)</sup> – for horizontal scrollbars;
- VerticalScrollBarSchemes<sup>(233)</sup> – for vertical scrollbars;
- HorizontalTabSetSchemes<sup>(232)</sup> – for horizontal tab sets;
- VerticalTabSetSchemes<sup>(233)</sup> – for vertical tab sets;
- BoxSchemes<sup>(232)</sup> – for rectangular areas of different controls.

## 1.3.9.2.1.1 Properties

**In TSRVSkin**

- BoxSchemes<sup>(232)</sup>
- Elements<sup>(232)</sup>
- Fonts<sup>(232)</sup>
- HorizontalScrollBarSchemes<sup>(232)</sup>
- HorizontalTabSetSchemes<sup>(232)</sup>
- Name<sup>(233)</sup>
- VerticalScrollBarSchemes<sup>(233)</sup>
- VerticalTabSetSchemes<sup>(233)</sup>

Specifies which Elements<sup>(232)</sup> are used to draw specific parts of rectangular areas of different controls.

**property** BoxSchemes: TSRVBoxSchemeCollection<sup>(234)</sup>;

This is a collection of TSRVBoxScheme<sup>(233)</sup> items. Each item has properties defining which Elements<sup>(232)</sup> are used to draw specific parts of a rectangular area of a control.

A collection of images used in skinnable objects.

**property** Elements: TSRVSkinElementCollection<sup>(238)</sup>;

**See also**

- TSRVSkinElement<sup>(237)</sup> (class of items)

A collection of fonts used in skinnable objects.

**property** Fonts: TSRVSkinFontCollection<sup>(236)</sup>;

**See also**

- TSRVSkinFont<sup>(235)</sup> (class of items)

Specifies which Elements<sup>(232)</sup> are used to draw specific parts of a horizontal scrollbar.

**property** HorizontalScrollBarSchemes: TSRVScrollBarSkinSchemeCollection<sup>(240)</sup>;

This is a collection of TSRVScrollBarSkinScheme<sup>(238)</sup> items. Each item has properties defining which Elements<sup>(232)</sup> are used to draw specific parts of a horizontal scrollbar.

TSRVScrollBar<sup>(243)</sup> has SkinSchemeIndex<sup>(247)</sup> property specifying the index in this collection (used if TSRVScrollBar<sup>(243)</sup>.Kind = sbHorizontal).

TSRUIViewEdit<sup>(40)</sup> has HScrollBarSchemeIndex<sup>(53)</sup> property specifying the index in this collection for applying to horizontal scrollbar.

Specifies which Elements<sup>(232)</sup> are used to draw specific parts of a horizontal tab set control.

**property** HorizontalTabSetSchemes: TSRVTabSetSkinSchemeCollection<sup>(242)</sup>;

This is a collection of TSRVTabSetSkinScheme<sup>(241)</sup> items. Each item has properties defining which Elements<sup>(232)</sup> are used to draw specific parts of a horizontal tab set control.

TSRVTabSet<sup>(249)</sup> has SkinSchemeIndex<sup>(257)</sup> property specifying the index in this collection (used if TSRVTabSet<sup>(249)</sup>.Kind = *srvtskHorizontal*).

Specifies the skin name.

**property** Name: TRVUnicodeString;

This text is shown at design time when editing TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup> collection.

Specifies which Elements<sup>(232)</sup> are used to draw specific parts of a vertical scrollbar.

**property** VerticalScrollBarSchemes: TSRVScrollBarSkinSchemeCollection<sup>(240)</sup>;

This is a collection of TSRVScrollBarSkinScheme<sup>(238)</sup> items. Each item has properties defining which Elements<sup>(232)</sup> are used to draw specific parts of a vertical scrollbar.

TSRVScrollBar<sup>(243)</sup> has SkinSchemeIndex<sup>(247)</sup> property specifying the index in this collection (used if TSRVScrollBar<sup>(243)</sup>.Kind = *sbVertical*).

TSRRichViewEdit<sup>(40)</sup> has VScrollBarSchemeIndex<sup>(70)</sup> property specifying the index in this collection for applying to vertical scrollbar.

Specifies which Elements<sup>(232)</sup> are used to draw specific parts of a vertical tab set control.

**property** VerticalTabSetSchemes: TSRVTabSetSkinSchemeCollection<sup>(242)</sup>;

This is a collection of TSRVTabSetSkinScheme<sup>(241)</sup> items. Each item has properties defining which Elements<sup>(232)</sup> are used to draw specific parts of a vertical tab set control.

TSRVTabSet<sup>(249)</sup> has SkinSchemeIndex<sup>(257)</sup> property specifying the index in this collection (used if TSRVTabSet<sup>(249)</sup>.Kind = *srvtskVertical*).

### 1.3.9.2.2 TSRVBoxScheme

TSRVBoxScheme is a class defining a visual appearance of rectangular areas of different controls.

This is a class of items in the following collection:

- TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].BoxSchemes<sup>(232)</sup>;

**Unit** SRVSkinManager.

#### Syntax

```
TSRVBoxScheme = class (TCollectionItem)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*

#### Properties

**Name:** TRVUnicodeString – a name of this scheme. This text is displayed at design time when editing collections.

Other properties define indices in `TSRVSkinManager(228).Skins(230)[i].Elements(232)` collection. All these properties are Integers, their default value is -1.

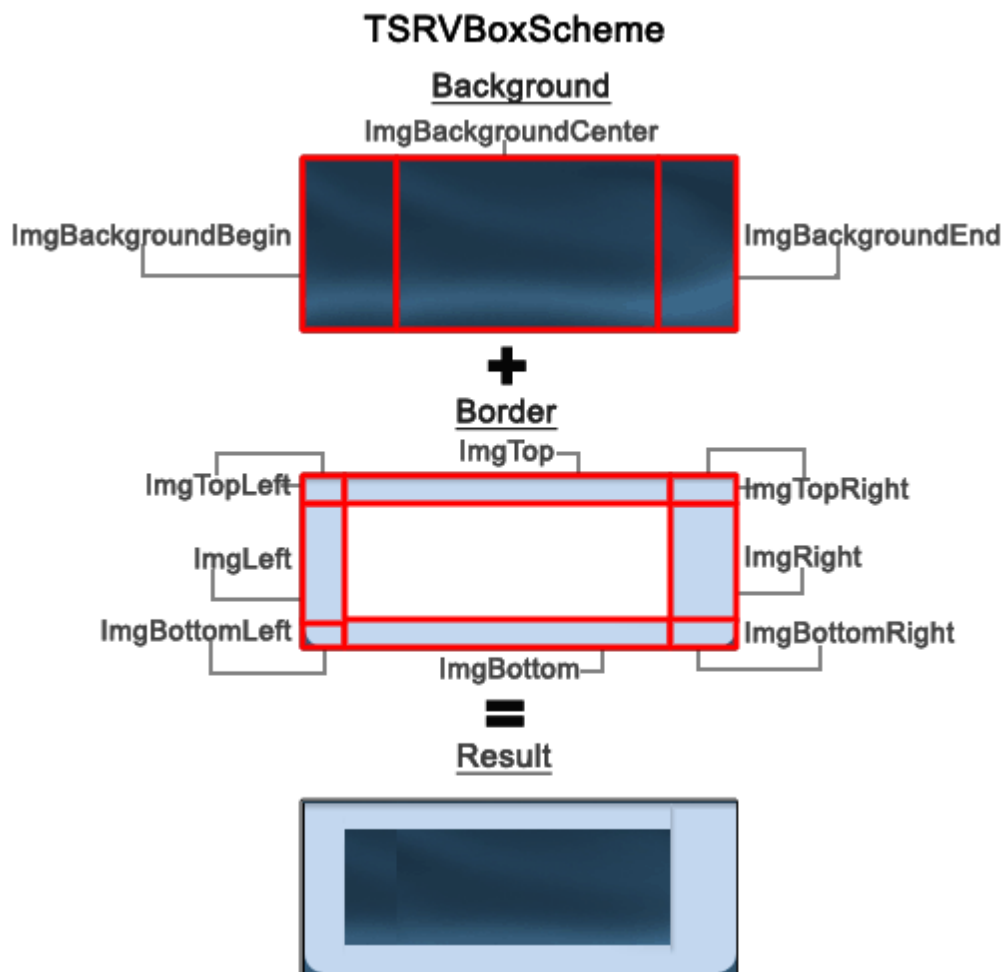
Background:

- **ImgBackgroundBegin** – left side;
- **ImgBackgroundEnd** – right side;
- **ImgBackgroundCenter** – middle of background (stretched).

Border:

- **ImgLeft**, **ImgRight**, **ImgTop**, **ImgBottom** – left/right/top/bottom sides (stretched);
- **ImgTopLeft**, **ImgTopRight**, **ImgBottomLeft**, **ImgBottomRight** – corners.

## Example



### 1.3.9.2.3 TSRVBoxSchemeCollection

`TSRVBoxSchemeCollection` is a class of a collection of skin schemes for rectangular elements of different controls.

This is a class of the following collections:

- `TSRVSkinManager(228).Skins(230)[i].BoxSchemes(232)`;

**Unit** SRVSkinManager.

### Syntax

```
TSRVBoxSchemeCollection = class(TCollection)
```

### Hierarchy

---

*TObject*

*TPersistent*

*TCollection*

### Properties

---

- **Items**[Index: Integer]: TSRVBoxScheme<sup>(233)</sup>;
- **Count**

#### 1.3.9.2.4 TSRVSkinCollection

TSRVSkinCollection is a class of a collection of skins.

This is a class of the collection TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>.

**Unit** SRVSkinManager.

### Syntax

```
TSRVSkinCollection = class(TCollection)
```

### Hierarchy

---

*TObject*

*TPersistent*

*TCollection*

### Properties

---

- **Items**[Index: Integer]: TSRVSkin<sup>(231)</sup>;
- **Count**

#### 1.3.9.2.5 TSRVSkinFont

TSRVSkinFont is a class defining properties for fonts used in skinnable objects.

This is a class of items in the collection TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Fonts<sup>(232)</sup>.

**Unit** SRVSkinManager.

### Syntax

```
TSRVSkinFont = class(TCollectionItem)
```

### Hierarchy

---

*TObject*

*TPersistent*

*TCollectionItem*

## Properties

- **Name:** TRVUnicodeString – a caption of this item. This text is displayed at design time when editing TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Fonts<sup>(232)</sup> collection.
- **Disabled:** TSRVFont – a font used when the object is disabled (where TSRVFont = class(TFont)).
- **Normal:** TSRVFont – a font used when the object is in a normal state.
- **Hot:** TSRVFont – a font used when the object is below the mouse pointer.
- **Down:** TSRVFont – a font used when the object is clicked.
- **Focused:** TSRVFont – a font used when the object is in a focused state.

## Example

### TSRVSkinFont



#### 1.3.9.2.6 TSRVSkinFontCollection

TSRVSkinFontCollection is a class of a collection of font items for applying to skinnable objects.

This is a class of the collection TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Fonts<sup>(232)</sup>.

**Unit** SRVSkinManager.

### Syntax

```
TSRVSkinFontCollection = class (TCollection)
```

### Hierarchy

*TObject*

*TPersistent*

*TCollection*

### Properties

- **Items**[Index: Integer]: TSRVSkinFont<sup>(235)</sup>;
- **Count**



### 1.3.9.2.7 TSRVSkinElement

TSRVSkinElement is a class defining properties for skinnable objects. Font properties are defined separately in TSRVSkinFont<sup>(235)</sup>.

This is a class of items in the collection TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Elements<sup>(232)</sup>.

**Unit** SRVSkinManager.

#### Syntax

```
TSRVSkinElement = class (TCollectionItem)
```

#### Hierarchy

---

*TObject*

*TPersistent*

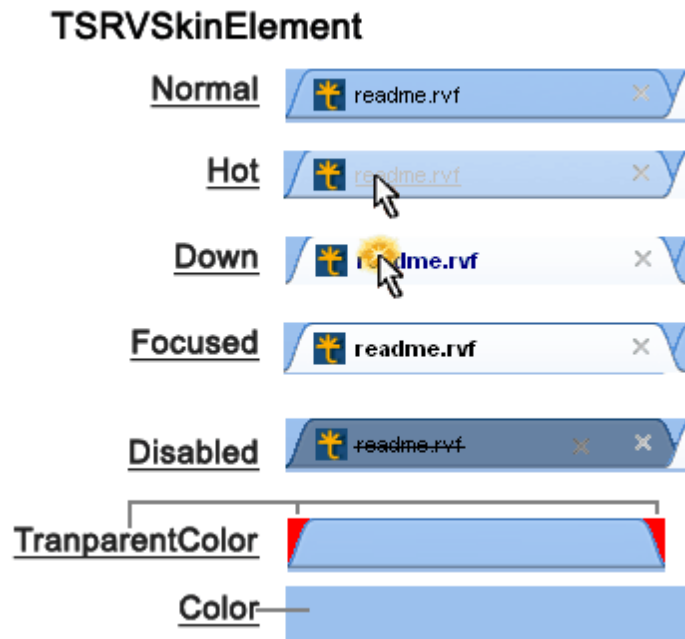
*TCollectionItem*

#### Properties

---

- **Name:** TRVUnicodeString – a caption of this item. This text is displayed at design time when editing TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Elements<sup>(232)</sup> collection.
- **Color:** TColor – a background color (usually used when images are not defined). Default value: *c/White*.
- **TransparentColor** – color of images that will be transparent when images are drawn. Used for Disabled, Normal, Hot, Down, Focused pictures, if they contain TBitmap. Default value: *c/None* (no bitmap transparency).
- **Disabled:** TPicture – a picture used when the object is disabled.
- **Normal:** TPicture – a picture used when the object is in a normal state.
- **Hot:** TPicture – a picture used when the object is below the mouse pointer.
- **Down:** TPicture – a picture used when the object is clicked.
- **Focused:** TPicture – a picture used when the object is in a focused state.

## Example



### 1.3.9.2.8 TSVSkinElementCollection

TSVSkinElementCollection is a class of a collection of image items for applying to skinnable objects.

This is a class of the collection TSVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Elements<sup>(232)</sup>.

**Unit** SRVSkinManager.

#### Syntax

```
TSVSkinElementCollection = class (TCollection)
```

#### Hierarchy

TObject

TPersistent

TCollection

#### Properties

- **Items**[Index: Integer]: TSVSkinElement<sup>(237)</sup>;
- Count

### 1.3.9.2.9 TSVScrollBarSkinScheme

TSVScrollBarSkinScheme is a class defining a visual appearance of scrollbars.

This is a class of items in the following collections:

- TSVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].HorizontalScrollBarSchemes<sup>(232)</sup>;
- TSVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].VerticalScrollBarSchemes<sup>(233)</sup>.

**Unit** SRVSkinManager.

### Syntax

```
TSRVScrollBarSkinScheme = class(TCollectionItem)
```

### Hierarchy

---

*TObject*

*TPersistent*

*TCollectionItem*

### Properties

---

**Name:** TRVUnicodeString – a name of this scheme. This text is displayed at design time when editing collections.

Other properties define indices in TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Elements<sup>(232)</sup> collection. All these properties are Integers, their default value is -1.

Buttons:

- **ImgDecButton** for the left/top scrollbar button;
- **ImgIncButton** for the right/bottom scrollbar button.

Thumbnail:

- **ImgThumbBegin** for the left/top side of thumbnail;
- **ImgThumbEnd** for the right/bottom side of thumbnail;
- **ImgThumbBackground** for the thumbnail background;
- **ImgThumbCenter** for drawing in the middle of thumbnail, above the **ImgThumbBackground**.

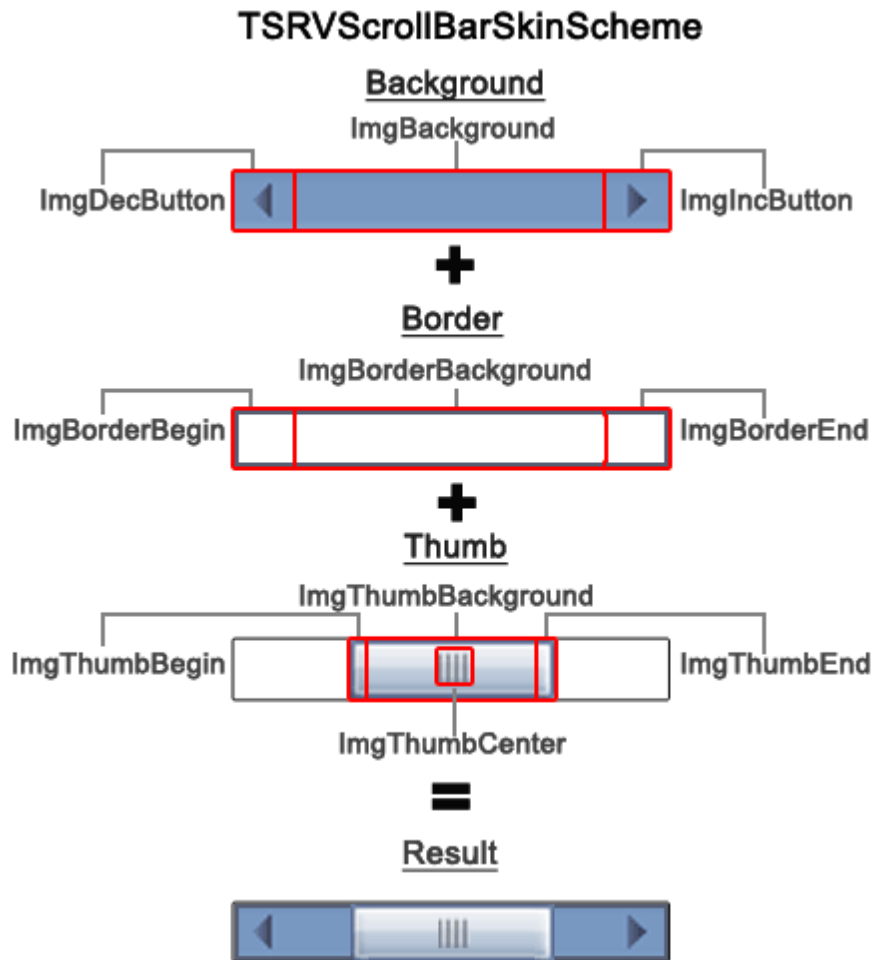
Border (the border is drawn above all other parts):

- **ImgBorderBegin** for the left/top side of the scrollbar border;
- **ImgBorderEnd** for the right/bottom side of the scrollbar border;
- **ImgBorderBackground** for the middle part of the scrollbar border (since the border is drawn after all other content, this image must have a transparent area in the middle).

Background:

- **ImgBackground** for the area between the buttons and the thumbnail.

## Example



### 1.3.9.2.10 TSRVScrollBarSkinSchemeCollection

TSRVScrollBarSkinSchemeCollection is a class of a collection of skin schemes for scrollbars.

This is a class of the following collections:

- `TSRVSkinManager(228).Skins(230)[i].HorizontalScrollBarSchemes(232)`;
- `TSRVSkinManager(228).Skins(230)[i].VerticalScrollBarSchemes(233)`.

**Unit** SRVSkinManager.

#### Syntax

```
TSRVScrollBarSkinSchemeCollection = class (TCollection)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollection*

#### Properties

- `Items[Index: Integer]: TSRVScrollBarSkinScheme(238)`;

- Count

### 1.3.9.2.11 TSRVTabSetSkinScheme

TSRVTabSetSkinScheme is a class defining a visual appearance of tab sets.

This is a class of items in the following collections:

- TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].HorizontalTabSetSchemes<sup>(232)</sup>;
- TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].VerticalTabSetSchemes<sup>(233)</sup>.

**Unit** SRVSkinManager.

#### Syntax

```
TSRVTabSetSkinScheme = class (TCollectionItem)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*

#### Properties

**Name:** TRVUnicodeString – a name of this scheme. This text is displayed at design time when editing collections.

**FontTab:** Integer specifies the index in the TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Fonts<sup>(232)</sup> collection. It defines the font for text in the tab. If you use a vertical tab, this font must be a TrueType/OpenType. The default value is -1.

Other properties define indices in TSRVSkinManager<sup>(228)</sup>.Skins<sup>(230)</sup>[i].Elements<sup>(232)</sup> collection. All these properties are Integers, their default value is -1.

Buttons on the tab set:

- **ImgDecButton** for the button to scroll to the beginning;
- **ImgIncButton** for the button to scroll to the end;
- **ImgAddTabButton** for the button for adding a new tab.

Tab:

- **ImgTabBegin** for the left/top side of tab;
- **ImgTabEnd** for the right/bottom side of tab;
- **ImgTabBackground** for the tab background;
- **ImgCloseButton** for a button for closing the tab.

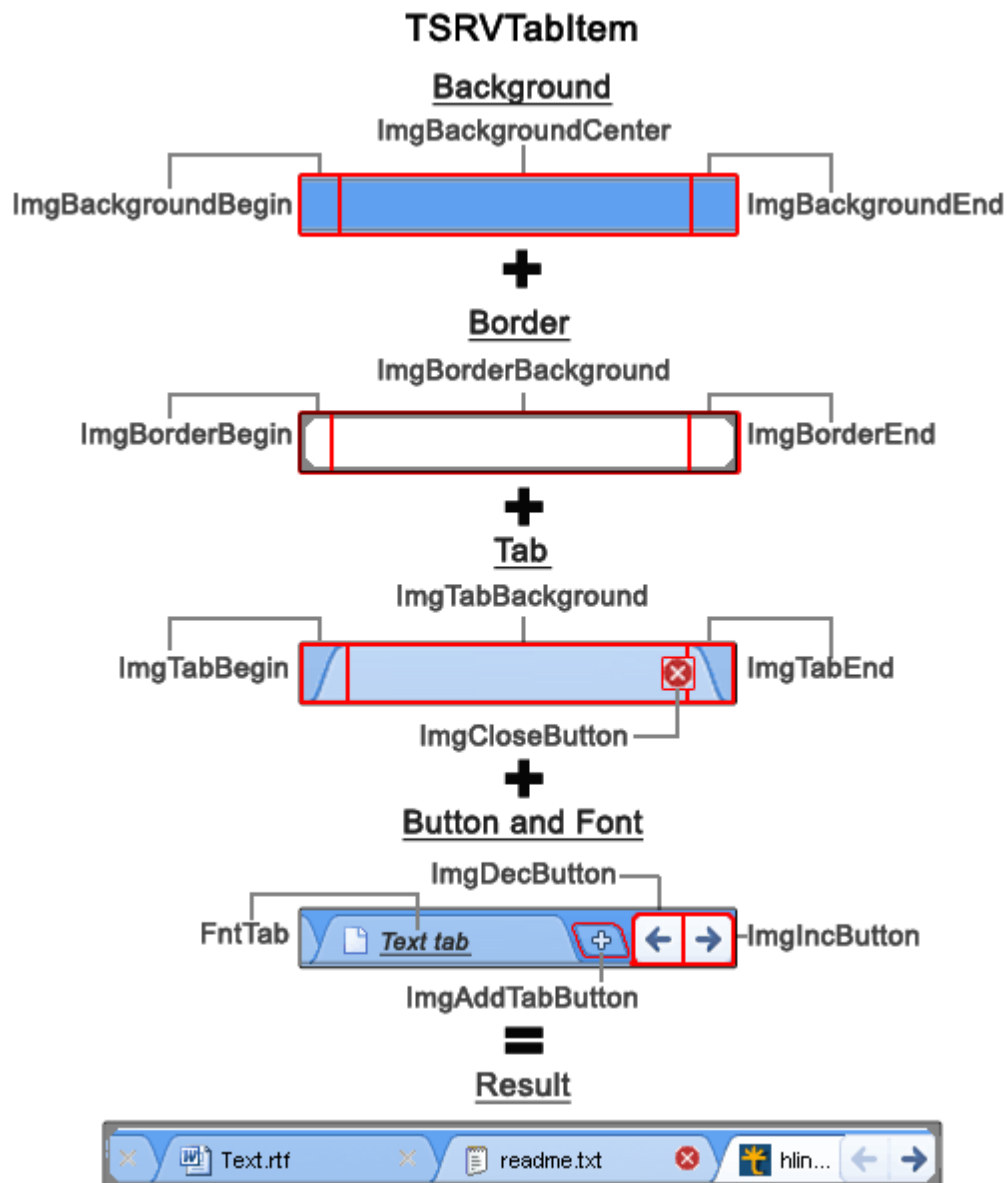
Border (the border is drawn above all other parts):

- **ImgBorderBegin** for the left/top side of the control border;
- **ImgBorderEnd** for the right/bottom side of the control border;
- **ImgBorderBackground** for the middle part of the control border (since the border is drawn after all other content, this image must have a transparent area in the middle).

Background:

- **ImgBackgroundBegin** for the left/top side of the control background;
- **ImgBackgroundEnd** for the right/bottom side of the control background;
- **ImgBackgroundCenter** for the middle part of the control background.

## Example



### 1.3.9.2.12 TSRVTabSetSkinSchemeCollection

TSRVTabSetSkinSchemeCollection is a class of a collection of skin schemes for tab set controls.

This is a class of the following collections:

- `TSRVSkinManager(228).Skins(230)[i].HorizontalTabSetSchemes(232)`;
- `TSRVSkinManager(228).Skins(230)[i].VerticalTabSetSchemes(233)`.

**Unit** SRVSkinManager.

#### Syntax

```
TSRVTabSetSkinSchemeCollection = class (TCollection)
```

## Hierarchy

*TObject*  
*TPersistent*  
*TCollection*

## Properties

- **Items**[Index: Integer]: *TSRVTabSetSkinScheme*<sup>(241)</sup>;
- **Count**

### 1.3.10 TSRVScrollBar

*TSRVScrollBar* is a scrollbar, which is used to scroll the contents of a window, form, or control.

**Unit** *SRVSBAR*.

#### Syntax

```
TSRVScrollBar = class (TSRVCustomControl(335))
```

## Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TSRVCustomControl*<sup>(335)</sup>

## Description

This is a skinnable version of *TScrollBar*. These scrollbars are used in *TSRichViewEdit*<sup>(40)</sup>.

## Visual appearance

If skins are not defined, the scrollbar appearance depends on *SRVControlStyle*<sup>(248)</sup> property:

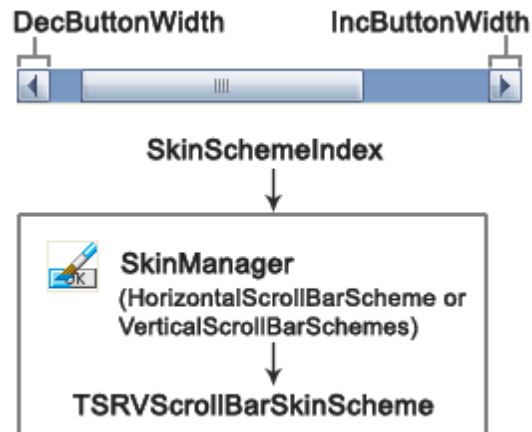
**srvcsSimple** (if *RVControlsPainter.Theme* = *rvctPaleBlue*):



**srvcsClassic:**

Appearance depends on *UseXPThemes*<sup>(248)</sup>, *Flat*<sup>(245)</sup> and *Color* properties.

*TSRVScrollBar* is a skinnable component. Skins are used when *SkinManager*<sup>(247)</sup> and *SkinSchemeIndex*<sup>(247)</sup> properties are assigned. You can draw various parts of the control yourself using custom drawing events<sup>(249)</sup>.



If this control is not inserted in `TSRichViewEdit`<sup>(40)</sup> control, we recommend to assign *True* to its `DoubleBuffered` property to prevent flickering.

### 1.3.10.1 Properties

#### In `TSRVScrollBar`

- `DecButtonWidth`<sup>(245)</sup>
- `Flat`<sup>(245)</sup>
- `FocusBlinkDelay`<sup>(245)</sup>
- `IncButtonWidth`<sup>(246)</sup>
- `Kind`<sup>(246)</sup>
- `LargeChange`<sup>(247)</sup>
- `PageSize`<sup>(246)</sup>
- `Position`<sup>(246)</sup>
- `ScrollingDelay`<sup>(247)</sup>
- `SmallChange`<sup>(247)</sup>
- `UseXPThemes`<sup>(248)</sup>

#### Inherited from `TSRVCustomControl`<sup>(335)</sup>

- `AutoDarkMode`
- `DarkMode`
- ▶ `MouseIn`
- `SkinManager`<sup>(247)</sup>
- `SkinSchemeIndex`<sup>(247)</sup>
- `SRVControlStyle`<sup>(248)</sup>

#### Inherited from `TCustomControl`

- `Align`
- `Anchors`
- `BiDiMode`
- `Color`
- `Constraints`
- `DoubleBuffered`



- DragCursor
- DragKind
- DragMode
- Enabled
- Kind
- LargeChange
- Max
- Min
- ParentBiDiMode
- ParentCtl3D
- ParentShowHint
- PopupMenu
- ShowHint
- SmallChange
- TabOrder
- TabStop
- Visible

#### 1.3.10.1.1 TSRVScrollBar.DecButtonWidth

Specifies width of the button for scrolling to the beginning.

**property** DecButtonWidth: TRVPixel96Length;

If value of this property is 0, a system default value is used. Values less than 5 are treated as 5.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

##### Default value

0

#### 1.3.10.1.2 TSRVScrollBar.Flat

Makes the scrollbar "flat".

**property** Flat: Boolean;

If *True*, the scrollbar looks flat and does not has a frame around it.

This flat appearance is used if:

- Windows themes are not active (see UseXPThemes<sup>(248)</sup>), and
- a skin is not applied (see SkinManager<sup>(335)</sup>, SkinSchemeIndex<sup>(335)</sup>), and
- SRVControlStyle<sup>(248)</sup> = *srvcsClassic*

##### Default value

*False*

#### 1.3.10.1.3 TSRVScrollBar.FocusBlinkDelay

Specifies the delay for blinking when the scroll bar is focused.

**property** FocusBlinkDelay: Integer;

This property is used only in a skin mode, when a focused image is defined for the thumb tab.

**Default value**

500

**1.3.10.1.4 TSRVScrollBar.IncButtonWidth**

Specifies width of the button for scrolling to the end.

**property** IncButtonWidth: TRVPixel96Length;

If value of this property is 0, a system default value is used. Values less than 5 are treated as 5.

This value is used only in a skin mode.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value**

0

**1.3.10.1.5 TSRVScrollBar.Kind**

Specifies whether the scroll bar is horizontal or vertical.

**property** Kind: TScrollBarKind;

Set Kind to indicate the orientation of the scroll bar.

Possible values: *sbHorizontal*, *sbVertical*.

**Default value**

sbHorizontal

**1.3.10.1.6 TSRVScrollBar.PageSize**

Specifies the size of the thumb tab.

**property** PageSize: Integer;

PageSize is the size of the thumb tab, measured in the same units as Position<sup>246</sup>, Min, and Max (not pixels).

0 means the default size.

**Default value**

0

**1.3.10.1.7 TSRVScrollBar.Position**

Indicates the current position of the scroll bar.

**property** Position: Integer;

This value is in range Min..Max-PageSize<sup>246</sup>.

**Default value**

0

### 1.3.10.1.8 TSRVScrollBar.ScrollingDelay

Defines a scrolling speed.

**property** ScrollingDelay: Integer;

This is an interval between scrolling events, ms. Successive scrolling events occur, for example, when the user presses a scrolling button.

#### Default value

40

### 1.3.10.1.9 TSRVScrollBar.SkinManager

Determines which skin manager is associated with the scrollbar.

**property** SkinManager: TSRVSkinManager<sup>(228)</sup>;

SkinManager defines a visual appearance of this control. If it is not specified, the control has a default (no-skin) view.

#### See also:

- SkinSchemeIndex<sup>(257)</sup>

### 1.3.10.1.10 TSRVScrollBar.SkinSchemeIndex

Determines which skin scheme is used for this scrollbar.

**property** SkinSchemeIndex: Integer;

If Kind=*sbHorizontal*, this is an index in SkinManager<sup>(247)</sup>.CurrentSkin<sup>(230)</sup>.HorizontalScrollBarSchemes<sup>(232)</sup> collection.

If Kind=*sbVertical*, this is an index in SkinManager<sup>(247)</sup>.CurrentSkin<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup> collection.

If SkinManager<sup>(247)</sup> = *nil*, or SkinManager<sup>(247)</sup>.CurrentSkin<sup>(230)</sup> = *nil*, or this index is not valid for this collection (negative or too large), then this scrollbar has a default (no-skin) appearance.

#### Default value

0

### 1.3.10.1.11 TSRVScrollBar.SmallChange, LargeChange

The properties defines changes of the current position of the scroll bar.

**property** SmallChange: TScrollBarInc;

**property** LargeChange: TScrollBarInc;

**SmallChange** determines how much Position<sup>(246)</sup> changes when the user clicks the arrow buttons on the scroll bar or presses the arrow keys on the keyboard.

**LargeChange** determines how much Position<sup>(246)</sup> changes when the user clicks the scroll bar on either side of the thumb tab or presses PgUp or PgDn.

#### Default value

1

### 1.3.10.1.12 TSRVScrollBar.SRVControlStyle

Define appearance of this scrollbar.

**property** SRVControlStyle: TSRVControlStyle<sup>(276)</sup>;

This property is used if a skin is not assigned (see SkinManager<sup>(247)</sup> and SkinSchemeIndex<sup>(247)</sup>).

**srvcSimple** (an example view if RVControlsPainter.Theme = *rvctPaleBlue*):



**srvcClassic:**

If UseXPThemes<sup>(248)</sup> = *True*, the scrollbar uses VCL themes (if available) or Windows themes (if available). Otherwise, the scrollbar has 3d look. When the scrollbar is inserted in a TSRichViewEdit<sup>(40)</sup> document, theme drawing is scaled not completely correctly. We recommend using *srvcSimple* instead.

**Default value:**

*srvcClassic*

This is the only control in ScaleRichView/SRVControls that has **SRVControlStyle** = *srvcClassic* by default.

### 1.3.10.1.13 TSRVScrollBar.UseXPThemes

Specifies whether the scrollbar uses VCL and Windows themes (visual styles), if they are available.

**property** UseXPThemes: Boolean;

VCL/Windows themes are used by the scrollbar if:

- it is not skinned (SkinManager<sup>(247)</sup> = nil, or SkinManager<sup>(247)</sup>.CurrentSkin<sup>(230)</sup> = nil, SkinSchemeIndex<sup>(247)</sup> is not valid) and
- SRVControlStyle<sup>(248)</sup> = *srvcClassic*

If a VCL theme is active (in Delphi XE2 or newer), it is used for drawing. Otherwise, Windows theme is used.

Note: when this scrollbar is inserted in a TSRichViewEdit<sup>(40)</sup> document, theme drawing is scaled not completely correctly. We recommend using SRVControlStyle<sup>(248)</sup> = *srvcSimple* instead.

**Default value**

*True*

## 1.3.10.2 Events

### In TSRVScrollBar

- OnDrawArea<sup>(249)</sup>
- OnDrawBorder<sup>(249)</sup>
- OnDrawDecButton<sup>(249)</sup>
- OnDrawIncButton<sup>(249)</sup>
- OnDrawThumbTab<sup>(249)</sup>

### Inherited from TScrollBar

- OnChange

- OnContextPopup
- OnDragDrop
- OnDragOver
- OnEndDock
- OnEndDrag
- OnEnter
- OnExit
- OnKeyDown
- OnKeyPress
- OnKeyUp
- OnScroll
- OnStartDock
- OnStartDrag

### 1.3.10.2.1 TSRVScrollBar.OnDraw...

Events for custom drawing of the scrollbar.

```

property OnDrawBorder: TSRVDrawScrollBarEvent268;
property OnDrawDecButton: TSRVDrawScrollBarEvent268;
property OnDrawIncButton: TSRVDrawScrollBarEvent268;
property OnDrawThumbTab: TSRVDrawScrollBarEvent268;
property OnDrawArea: TSRVDrawScrollBarAreaEvent268;

```

Use OnDrawArea to draw the scrollbar background (areas between the "inc/dec buttons" and the thumb tab).

Use OnDrawBackground to draw the control background. This event is called before all other custom drawing events.

Use OnDrawThumbTab to draw the thumb tab.

Use OnDrawDecButton to draw the button for scrolling to the beginning, and OnDrawIncButton to draw the button for scrolling to the end.

These events occur only if `SRVControlStyle248 = srvcClassic`.

## 1.3.11 TSRVTabSet

The TSRVTabSet component presents horizontal or vertical tabs users can click to initiate actions.

**Unit** SRVTabSet.

### Syntax

```
TSRVTabSet = class (TCustomControl);
```

### Hierarchy

```

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomControl

```

## Tabs

You create a set of tabs for the tab set control when you specify a list of items as the value of the `Tabs` <sup>(259)</sup> property. One tab is created for each item.

To determine which tab is currently selected or to use code to select a tab, use the `TabIndex` <sup>(258)</sup> property. To find out which tab is the first/last visible tab in the tab set control use the `FirstTabIndex` <sup>(253)</sup> / `LastTabIndex` <sup>(254)</sup> properties. To make the tab visible, use `ScrollToTab` <sup>(259)</sup>.

The user can close tabs, if closing buttons are displayed. Properties of these buttons are specified in `CloseButton` <sup>(253)</sup> property.

Tabs can have icons from `ImageList` <sup>(253)</sup>.

## Layout

Tabs are displayed horizontally or vertically, depending on the value of `Kind` <sup>(254)</sup> property.

By default, tabs are positioned at the top (for a horizontal ruler) / the left (for a vertical ruler) of the control. To move them to bottom / right, assign `True` to `OppositeTabPosition` <sup>(256)</sup>.

By default, left-to-right text is displayed from left to right (for a horizontal ruler) / from bottom to top (for a vertical ruler). To display it upside down / from top to bottom, assign `True` to `MirrorText` <sup>(255)</sup>.

By default, tabs are ordered from left to right. Using `BiDiMode` property, you can order them from from right to left (`BiDiMode` does not affect tab order in a vertical ruler).

The following properties affect the size of tabs: `AutoWidth` <sup>(252)</sup>, `MinTabSize` <sup>(255)</sup>, `MaxTabSize` <sup>(255)</sup>, `TabHeight` <sup>(258)</sup>.

The following properties affect the layout of each tab: `Indent` <sup>(254)</sup>, `Margins` <sup>(254)</sup>, `MirrorText` <sup>(255)</sup>, `TextAlignH` <sup>(259)</sup>, `TextAlignV` <sup>(259)</sup>.

## Visual appearance

If skins are not defined, the tab set appearance depends on `SRVControlStyle` <sup>(258)</sup> property:

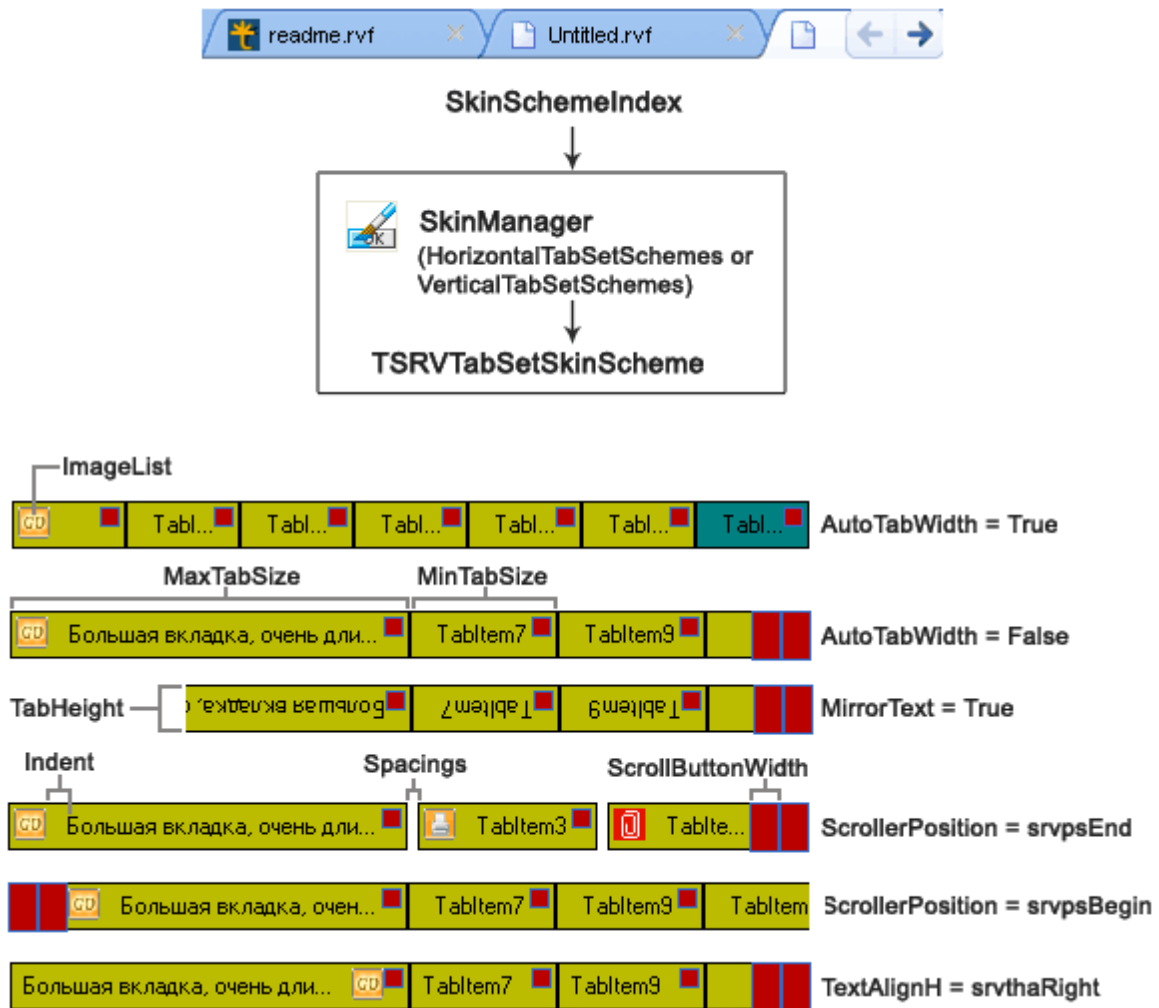
**srvcsSimple** (if `RVControlsPainter.Theme = rvctPaleBlue`):



**srvcsClassic:**



`TSRVTabSet` is a skinnable component. Skins are used when `SkinManager` <sup>(257)</sup> and `SkinSchemeIndex` <sup>(257)</sup> properties are assigned. You can draw various parts of the control yourself using custom drawing events <sup>(260)</sup>.



## Rearranging tabs

Tabs can be moved with the mouse, if `CanMoveTabs`<sup>(253)</sup> is `True`. When moving, `OnMoveTab`<sup>(261)</sup> event occurs.

### 1.3.11.1 Properties

#### In TSRVTabSet

- `AutoDarkMode`<sup>(253)</sup>
- `AutoTabWidth`<sup>(252)</sup>
- `CanMoveTabs`<sup>(253)</sup>
- `CloseButton`<sup>(253)</sup>
- `DarkMode`<sup>(253)</sup>
- ▶ `FirstTabIndex`<sup>(253)</sup>
- `ImageList`<sup>(253)</sup>
- `Indent`<sup>(254)</sup>
- `Kind`<sup>(254)</sup>
- ▶ `LastTabIndex`<sup>(254)</sup>

- Margins <sup>(254)</sup>
- MaxTabSize <sup>(255)</sup>
- MinTabSize <sup>(255)</sup>
- MirrorText <sup>(255)</sup>
- OppositeTabPosition <sup>(256)</sup>
- ScrollButtonWidth <sup>(256)</sup>
- ScrollerMenu <sup>(256)</sup>
- ScrollerPosition <sup>(256)</sup>
- ScrollingDelay <sup>(257)</sup>
- SkinManager <sup>(257)</sup>
- SkinSchemeIndex <sup>(257)</sup>
- Spacings <sup>(257)</sup>
- SRVControlStyle <sup>(258)</sup>
- TabHeight <sup>(258)</sup>
- TabIndex <sup>(258)</sup>
- Tabs <sup>(259)</sup>
- TextAlignH <sup>(259)</sup>
- TextAlignV <sup>(259)</sup>

## Inherited from TCustomControl

---

- Align
- Anchors
- Color
- Constraints
- Cursor
- Enabled
- Font
- Height
- Hint
- Left
- PopupMenu
- ShowHint
- TabStop
- Top
- Width

### 1.3.11.1.1 TSRVTabSet.AutoTabWidth

Specifies how widths of tabs are calculated.

**property** AutoTabWidth: Boolean;

If *True*, all tabs have widths wide enough to show their content (icon and text).

If *False*, widths of all tabs are calculated as  $\text{SRVTabSet.Width} / \text{SRVTabSet.Tabs}^{(259)}.Count$ .

In the both modes, MinTabSize <sup>(255)</sup> and MaxTabSize <sup>(255)</sup> are taken into account.

#### Default value

*True*



#### 1.3.11.1.2 TSRVTabSet.CanMoveTabs

Allows rearranging tabs with the mouse.

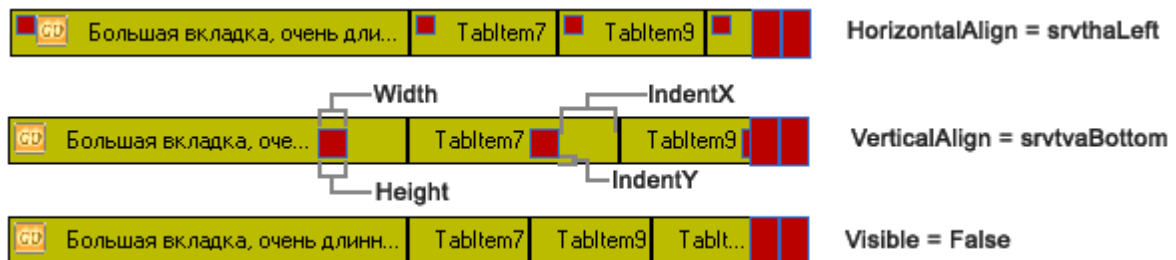
**property** CanMoveTabs: Boolean;

When moving tab, OnMoveTab<sup>(261)</sup> event occurs (when two tabs are exchanged).

#### 1.3.11.1.3 TSRVTabSet.CloseButton

Contains properties of tab closing buttons.

**property** CloseButton: TSRVCloseButton<sup>(262)</sup>;



#### 1.3.11.1.4 TSRVTabSet.DarkMode, AutoDarkMode

The property allow implementing a "dark mode".

**property** AutoTabWidth: Boolean;

These properties are used if SRVControlStyle<sup>(258)</sup> = *srvcSimple*.

If **AutoDarkMode** = *True*, and the control is inserted in a dark-mode editor (TSRichViewEdit<sup>(40)</sup> or TRichView), luminance of all colors is inverted.

Otherwise, **DarkMode** property is used (if *True*, luminance of all colors are inverted).

**Default values:**

AutoDarkMode: *True*

DarkMode: *False*

#### 1.3.11.1.5 TSRVTabSet.FirstTabIndex

Returns the index of the first visible tab.

**property** FirstTabIndex: Integer;

**See also:**

LastTabIndex<sup>(254)</sup>

#### 1.3.11.1.6 TSRVTabSet.ImageList

Determines the image list for drawing icons in tabs of the tab set control.

**property** ImageList: TCustomImageList;

Use **ImageList** to provide a customized list of bitmaps that can be displayed close to a tab's text. Individual tabs specify the image from this list that should appear by setting their ImageIndex<sup>(265)</sup> property.

### 1.3.11.1.7 TSRVTabSet.Indent

Adds spacing before text in each tab.

**property** Indent: TRVPixel96Length;

If the tab has an icon, this is a distance between the icon and the text.

If the tab does not have an icon, this is a distance from the tab's Margin<sup>(254)</sup> to text.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

5

### 1.3.11.1.8 TSRVTabSet.Kind

Specifies whether the tab set control is horizontal or vertical.

**type**

```
TSRVTabSetKind = (srvtskHorizontal, srvtskVertical);
```

**property** Kind: TSRVTabSetKind;

A vertical tab is rotated by 90° counterclockwise from a horizontal position.

**Default value:**

*srvtskHorizontal*

**See also:**

- MirrorText<sup>(255)</sup>
- OppositeTabPosition<sup>(256)</sup>

### 1.3.11.1.9 TSRVTabSet.LastTabIndex

Returns the index of the last visible tab.

**property** LastTabIndex: Integer;

**See also:**

FirstTabIndex<sup>(253)</sup>

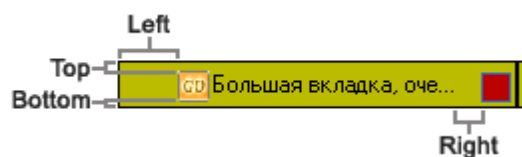
### 1.3.11.1.10 TSRVTabSet.Margins

Specifies margins for each tab.

**property** Margins: TSRVMargins;

The class TSRVMargins has Left, Right, Top, Bottom: TRVPixel96Length properties (0 by default).

These values are measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.



**See also:**

- TabHeight<sup>(258)</sup>
- Indent<sup>(254)</sup>

#### 1.3.11.1.11 TSRVTabSet.MaxTabSize

Specifies the maximum allowed tab width.

**property** MaxTabSize: TRVPixel96Length;

For horizontal<sup>(254)</sup> tab set controls, this property defines a maximum width of tabs.

For vertical<sup>(254)</sup> tab set controls, this property defines a maximum height of tabs.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

200

**See also:**

- AutoTabWidth<sup>(252)</sup>
- MinTabSize<sup>(255)</sup>

#### 1.3.11.1.12 TSRVTabSet.MinTabSize

Specifies the minimum allowed tab width.

**property** MinTabSize: TRVPixel96Length;

For horizontal<sup>(254)</sup> tab set controls, this property defines a minimum width of tabs.

For vertical<sup>(254)</sup> tab set controls, this property defines a minimum height of tabs.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

0

**See also:**

- AutoTabWidth<sup>(252)</sup>
- MaxTabSize<sup>(255)</sup>

#### 1.3.11.1.13 TSRVTabSet.MirrorText

Allows flipping text in all tabs (vertically for horizontal tab set control, horizontally otherwise)

**property** MirrorText: Boolean;

Note: this property is named inaccurately. Actually, the text is not mirrored by rotated by 180°.

This property also inverts TextAlignH<sup>(259)</sup>, TextAlignV<sup>(259)</sup>, and the icons positions relative to texts.

**Default value:**

*False*

**See also:**

- Kind<sup>(254)</sup>

#### 1.3.11.1.14 TSRVTabSet.OppositeTabPosition

Displays tabs at the opposite side of the tab set control.

**property** OppositeTabPosition: Boolean;

By default, tabs are located at the top side (for a horizontal <sup>254</sup> tab set) or at the left side (for a vertical <sup>254</sup> tab set) of the control. This property moves them to the opposite side (bottom / right).

**Default value:**

*False*

**See also:**

- Kind <sup>254</sup>
- MirrorText <sup>255</sup>

#### 1.3.11.1.15 TSRVTabSet.ScrollButtonWidth

Defines the width of scrolling buttons.

**property** ScrollButtonWidth: TRVPixel96Length;

Scrolling buttons appear when not all tabs can be shown without scrolling.

For horizontal <sup>254</sup> tab set controls, this property defines a width of the buttons.

For vertical <sup>254</sup> tab set controls, this property defines a height of the buttons.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

15

**See also:**

- ScrollerPosition <sup>256</sup>

#### 1.3.11.1.16 TSRVTabSet.ScrollerMenu

This property is reserved for future use.

**property** ScrollerMenu: TPopupMenu;

#### 1.3.11.1.17 TSRVTabSet.ScrollerPosition

Defines the position of scrolling buttons.

**type**

```
TSRVScrollerPosition = (srvpsBegin, srvpsEnd);
```

**property** ScrollerPosition: TSRVScrollerPosition;

For horizontal <sup>254</sup> tab set controls, RTL BiDiMode inverts value of this property.

Scrolling buttons appear when not all tabs can be shown without scrolling.

Depending on this property, they appear before or after tabs.

### 1.3.11.1.18 TSRVTabSet.ScrollingDelay

Defines the tab scrolling speed.

**property** ScrollingDelay: Integer;

This property sets a pause between redrawing on scrolling, in ms.

**Default value:**

100

### 1.3.11.1.19 TSRVTabSet.SkinManager

Determines which skin manager is associated with the tab set control.

**property** SkinManager: TSRVSkinManager<sup>(228)</sup>;

SkinManager defines a visual appearance of this tab set control. If it is not specified, the control has a default (no-skin) view.

**See also:**

- SkinSchemeIndex<sup>(257)</sup>

### 1.3.11.1.20 TSRVTabSet.SkinSchemeIndex

Determines which skin scheme is used for this tab set control.

**property** SkinSchemeIndex: Integer;

If Kind<sup>(254)</sup> = *srvtskHorizontal*, this is an index in SkinManager<sup>(257)</sup>.CurrentSkin<sup>(230)</sup>.HorizontalTabSetSchemes<sup>(232)</sup> collection.

If Kind<sup>(254)</sup> = *srvtskVertical*, this is an index in SkinManager<sup>(257)</sup>.CurrentSkin<sup>(230)</sup>.VerticalTabSetSchemes<sup>(233)</sup> collection.

If SkinManager<sup>(257)</sup> = *nil*, or SkinManager<sup>(257)</sup>.CurrentSkin<sup>(230)</sup> = *nil*, or this index is not valid for this collection (negative or too large), then this tab set control has a default (no-skin) appearance.

Individual tabs can override this property with their own SkinSchemeIndex<sup>(266)</sup>.

**Default value**

0

### 1.3.11.1.21 TSRVTabSet.Spacings

Specifies a distance between tabs.

**property** Spacings: TRVPixel96Length;

This value can be negative, if you want tabs to overlap.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

0

### 1.3.11.1.22 TSRVTabSet.SRVControlStyle

Define appearance of this tab set.

**property** SRVControlStyle: TSRVControlStyle<sup>(276)</sup>;

This property is used if a skin is not assigned (see SkinManager<sup>(257)</sup> and SkinSchemeIndex<sup>(257)</sup>).

**srvcsSimple** (if RVControlsPainter.Theme = *rvctPaleBlue*):



**srvcsClassic:**



**Default value:**

*srvcsSimple*

### 1.3.11.1.23 TSRVTabSet.TabHeight

Defines the height of tabs.

**property** TabHeight: TRVPixel96Length;

For horizontal<sup>(254)</sup> tab set controls, this property defines the height of tabs.

For vertical<sup>(254)</sup> tab set controls, this property defines the widths of tabs.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**See also:**

- Margins<sup>(254)</sup>
- Indent<sup>(254)</sup>
- TextAlignV<sup>(259)</sup>

### 1.3.11.1.24 TSRVTabSet.TabIndex

Determines which tab of a tab set control is currently selected.

**property** TabIndex: Integer;

The value of TabIndex can be from -1 to one less than the number of Tabs<sup>(259)</sup>. TabIndex is an index in the Tabs<sup>(259)</sup> collection property: a value of 0 indicates the first tab in the tab set control, 1 is the second tab, and so on. If no tabs are selected, TabIndex has a value of -1.

When a value is assigned to TabIndex, OnChange<sup>(260)</sup> event occurs, just as if the user had clicked on a tab.

### 1.3.11.1.25 TSRVTabSet.Tabs

Contains the list of tabs displayed in the tab set control.

**property** Tabs: TSRVTabCollection<sup>(267)</sup>;

Properties of each tab is defined in TSRVTabItem<sup>(264)</sup> class.

### 1.3.11.1.26 TSRVTabSet.TextAlignH

Defines a horizontal alignment of text and icon in each tab.

**property** TextAlignH: TSRVTHAlign<sup>(281)</sup>;

Meaning of this property is not changed if Kind<sup>(254)</sup> = *srvtskVertical*: this property always defines a horizontal alignment.

MirrorText<sup>(255)</sup> inverts value of this property. For a horizontal ruler, RTL BiDiMode inverts it as well.

**Default value:**

*srvthaLeft*

**See also:**

TextAlignV<sup>(259)</sup>

### 1.3.11.1.27 TSRVTabSet.TextAlignV

Defines a vertical alignment of text and icon in each tab.

**property** TextAlignV: TSRVTVAlign<sup>(281)</sup>;

Meaning of this property is not changed if Kind<sup>(254)</sup> = *srvtskVertical*: this property always defines a vertical alignment.

MirrorText<sup>(255)</sup> inverts value of this property. For a vertical ruler, RTL BiDiMode inverts it as well.

**Default value:**

*srvtvaCenter*

**See also:**

TextAlignH<sup>(259)</sup>

TabHeight<sup>(258)</sup>

## 1.3.11.2 Methods

### In TSRVTabSet

ScrollToTab<sup>(259)</sup>

### 1.3.11.2.1 TSRVTabSet.ScrollToTab

Scrolls tabs in the control to make Tabs<sup>(259)</sup> [TabIndex] visible.

**procedure** ScrollToTab (TabIndex : Integer) ;

### 1.3.11.3 Events

#### In TSRVTabSet

- OnChange<sup>(260)</sup>
- OnCloseTab<sup>(260)</sup>
- OnDrawBackground<sup>(260)</sup>
- OnDrawBorder<sup>(260)</sup>
- OnDrawCloseButton<sup>(260)</sup>
- OnDrawDecButton<sup>(260)</sup>
- OnDrawIncButton<sup>(260)</sup>
- OnDrawTab<sup>(260)</sup>
- OnMoveTab<sup>(261)</sup>

#### Inherited from TCustomControl

- OnClick
- OnEnter
- OnExit
- OnMouseDown
- OnMouseMove
- OnMouseUp
- OnKeyDown
- OnKeyUp

#### 1.3.11.3.1 TSRVTabSet.OnChange

Occurs after a new tab is selected.

**property** OnChange: TNotifyEvent;

This event occurs when value of TabIndex<sup>(258)</sup> property is changed.

#### 1.3.11.3.2 TSRVTabSet.OnCloseTab

Occurs when the user attempts to close a tab.

**type**

```
TSRVCloseTabEvent = procedure (Sender: TSRVTabSet(249); TabIndex: Integer;
    var CanClose : Boolean) of object;
```

**property** OnCloseTab: TSRVCloseTabEvent;

**Parameters**

**TabIndex** – the index of the tab to close (in the Tabs<sup>(259)</sup> collection).

Set **CanClose** to *False* to prevent closing.

When a tab is closed, its object is removed from Tabs<sup>(259)</sup> and freed.

#### 1.3.11.3.3 TSRVTabSet.OnDraw...

Events for custom drawing of the tab set control.

**property** OnDrawBackground: TSRVDrawTabSetEvent<sup>(270)</sup>;

**property** OnDrawBorder: TSRVDrawTabSetEvent<sup>(270)</sup>;



```

property OnDrawDecButton: TSRVDrawTabSetEvent(270);
property OnDrawIncButton: TSRVDrawTabSetEvent(270);
property OnDrawTab: TSRVDrawTabEvent(269);
property OnDrawCloseButton: TSRVDrawTabEvent(269);

```

### Drawing the tab set control

Use OnDrawBackground to draw the control background. This event is called before all other custom drawing events.

Use OnDrawBorder to draw the control border. This event is called after all other custom drawing events.

Use OnDrawDecButton to draw the button for scrolling tabs to the beginning, and OnDrawIncButton to draw the button for scrolling tabs to the end.

### Drawing tabs

Use OnDrawTab to draw a tab, and OnDrawCloseButton to draw a tab closing button.

#### 1.3.11.3.4 TSRVTabSet.OnMoveTab

Occurs when two tabs exchange their places as a result of tab moving.

```

type
  TSRVTabMoveEvent = procedure (Sender: TSRVTabSet(249); NewTabIndex, OldTabIndex :
property OnTabMove: TSRVTabMoveEvent;

```

**NewTabIndex, OldTabIndex** – indexes of exchanged tabs (in the Tabs<sup>(259)</sup> collection).

(this event can be interpreted as moving a tab from **OldTabIndex** to **NewTabIndex** position; but since this event occurs only for adjacent tabs, it is convenient to think about it as about exchanging places).

This event occurs only when tab rearranging is allowed, see CanMoveTabs<sup>(253)</sup> property.

### Example

Assuming that each tab corresponds to one item in Editors array. When moving tabs, we need to update this array:

```

procedure TForm3.SRVTabSet1TabMove(Sender: TSRVTabSet(249); NewTabIndex,
  OldTabIndex: Integer);
var
  tmp: TSRichViewEdit(40);
begin
  tmp := Editors[OldTabIndex];
  Editors[OldTabIndex] := Editors[NewTabIndex];
  Editors[NewTabIndex] := tmp;
end;

```

#### 1.3.11.4 Classes of properties

##### Classes of TSRVTabSet<sup>(249)</sup> properties

- TSRVTabCollection<sup>(267)</sup> – collection of TSRVTabItem<sup>(264)</sup> items; a type of Tabs<sup>(259)</sup> property;
- TSRVCloseButton<sup>(262)</sup> – properties of a tab closing button.

### 1.3.11.4.1 TSRVCloseButton

TSRVCloseButton contains properties of closing buttons that can be displayed on all tabs of TSRVTabSet<sup>(249)</sup> or TSRVImagesScroll<sup>(354)</sup>.

**Unit** SRVTabSet.

#### Syntax

```
TSRVCloseButton = class (TPersistent)
```

#### Hierarchy

*TObject*

*TPersistent*

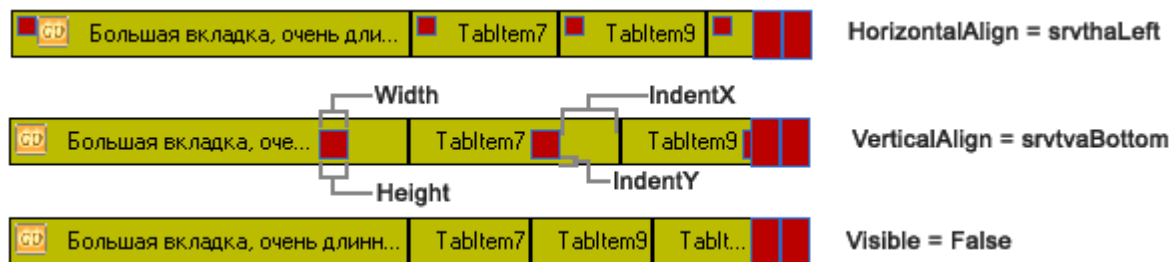
#### Properties

Closing buttons are shown if Visible<sup>(264)</sup> = *True*. If the tab set control has only a single tab, a closing button is displayed only if AllowCloseLastTab<sup>(263)</sup> = *True*.

Closing buttons can be shown either on all tabs, or only on a highlighted tab, or only on an active tab, depending on Visibility<sup>(264)</sup>.

Positions of closing buttons are defined with the properties HorizontalAlign<sup>(263)</sup>, VerticalAlign<sup>(264)</sup>, IndentX<sup>(263)</sup>, IndentY<sup>(263)</sup>.

Size of closing buttons is defined with the properties Width<sup>(264)</sup> and Height<sup>(263)</sup>.



#### 1.3.11.4.1.1 Properties

##### In TSRVCloseButton

- AllowCloseLastTab<sup>(263)</sup>
- Height<sup>(263)</sup>
- HorizontalAlign<sup>(263)</sup>
- IndentX<sup>(263)</sup>
- IndentY<sup>(263)</sup>
- VerticalAlign<sup>(264)</sup>
- Visibility<sup>(264)</sup>
- Visible<sup>(264)</sup>
- Width<sup>(264)</sup>

Specifies whether a closing button is displayed on the last tab (of [TSRVTabSet](#)<sup>(249)</sup> control) or on the last image (of [TSRVImagesScroll](#)<sup>(354)</sup> control).

**property** AllowCloseLastTab: Boolean;

If *True* (and [Visible](#)<sup>(264)</sup> is *True* too), a closing button is displayed even if the control contains a single tab / image.

**Default value:**

*True*

Specifies the height of the closing button.

**property** Height: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

10

Specifies the horizontal position of the tab closing button.

**property** HorizontalAlign: [TSRVTHAlign](#)<sup>(281)</sup>;

Meaning of this property is not changed if the tab set's [Kind](#)<sup>(254)</sup> = *srvtskVertical*: this property always defines a horizontal position.

For a horizontal ruler, RTL tab set's [BiDiMode](#) inverts value of this property.

**Default value:**

*srvthaRight*

Specifies the horizontal indent added to the button position.

**property** IndentX: TRVPixel96Length;

If [HorizontalAlign](#)<sup>(263)</sup> = *srvthaRight*, the button is shifted to the left by this value. Otherwise, it is shifted to the right.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

3

Specifies the vertical indent added to the button position.

**property** IndentY: TRVPixel96Length;

If [VerticalAlign](#)<sup>(264)</sup> = *srvthaBottom*, the button is shifted to the top by this value. Otherwise, it is shifted to the bottom.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

3

Specifies the vertical position of the tab closing button.

**property** VerticalAlign: TSRVTVAlign<sup>(281)</sup>;

Meaning of this property is not changed if the tab set's Kind<sup>(254)</sup> = *srvtSkVertical*: this property always defines a vertical position.

For a vertical ruler, RTL tab set's BiDiMode inverts value of this property.

**Default value:**

*srvtvaTop*

Specifies when tab closing buttons are displayed.

**type**

TSRVTSCloseButtonVisibility = (srvcbvAlways, srvcbvHot, srvcbvActive);

**property** Visibility: TSRVTSCloseButtonVisibility;

Closing buttons are displayed only if Visible<sup>(264)</sup> is *True*.

| Visibility          | Meaning   |
|---------------------|---|
| <i>srvcbvAlways</i> | Buttons are displayed on all tabs   |
| <i>srvcbvHot</i>    | A button is displayed only on the tab below the mouse pointer                                     |
| <i>srvcbvActive</i> | A button is displayed only on the active tab (Tabs <sup>(259)</sup> [TabIndex <sup>(258)</sup> ]) |

**Default value:**

*srvcbvAlways*

Specifies whether tabs have closing buttons.

**property** Visible: Boolean;

**Default value:**

*True*

Specifies the width of the closing button.

**property** Width: TRVPixel96Length;

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

**Default value:**

10

#### 1.3.11.4.2 TSRVTabItem

TSRVTabItem is an individual tab in TSRVTabSet<sup>(249)</sup> control.

This is a class of items in the collection TSRVTabSet<sup>(249)</sup>.Tabs<sup>(259)</sup>.

**Unit** SRVTabSet.

## Syntax

```
TSRVTabItem = class (TCollectionItem)
```

## Hierarchy

*TObject*

*TPersistent*

*TCollectionItem.*

## Properties

The tab has Text<sup>(266)</sup> and an optional icon, defined in ImageIndex<sup>(265)</sup>.

Normally, in a skin mode, appearance of all tabs is defined in the tab set control. You can override these settings with SkinSchemeIndex<sup>(266)</sup> and/or SkinFontIndex<sup>(265)</sup> properties.

The tab can be hidden (Visible<sup>(266)</sup> property) or disabled (Enabled<sup>(265)</sup> property).

### 1.3.11.4.2.1 Properties

#### In TSRVTabItem

- Enabled<sup>(265)</sup>
- ImageIndex<sup>(265)</sup>
- SkinFontIndex<sup>(265)</sup>
- SkinSchemeIndex<sup>(266)</sup>
- Tag<sup>(266)</sup>
- Text<sup>(266)</sup>
- Visible<sup>(266)</sup>

Controls whether the tab responds to mouse and keyboard events.

**property** Enabled: Boolean;

**Default value:**

*True*

Determines which image is displayed as the icon for the tab.

**property** ImageIndex: Integer;

Set ImageIndex to associate the tab with one of the images in the ImageList<sup>(253)</sup> property of the tab set control.

**Default value:**

-1 (no image)

Specifies which font is used for this tab in the skin mode.

**property** SkinFontIndex: Integer;

In a no-skin mode, a tab's font is defined in the Font property of the tab set control.

In a skin mode, a tab's fonts are defined in TabSet.SkinManager<sup>(257)</sup>.CurrentSkin<sup>(230)</sup>.Fonts<sup>(232)</sup> [FontIndex],

where *FontIndex* is a value of *FntTab*<sup>(241)</sup> property of *SkinManager*<sup>(257)</sup>.*CurrentSkin*<sup>(230)</sup>.*HorizontalTabSetSchemes*<sup>(232)</sup>/*VerticalTabSetSchemes*<sup>(233)</sup>[*SkinSchemeIndex*],

where *SkinSchemeIndex* is *SkinSchemeIndex*<sup>(266)</sup> property of the tab (if not negative) or *TabSet.SkinSchemeIndex*<sup>(257)</sup>.

**Default value:**

-1

Determines which skin scheme is used for this tab.

**property** *SkinSchemeIndex*: *Integer*;

If value of this property is equal to -1, *SkinSchemeIndex*<sup>(257)</sup> of the tab set control is used for this tab.

If *TabSet.Kind*<sup>(254)</sup>=*srvtSkHorizontal*, this is an index in *TabSet.SkinManager*<sup>(257)</sup>.*CurrentSkin*<sup>(230)</sup>.*HorizontalTabSetSchemes*<sup>(232)</sup> collection.

If *TabSet.Kind*<sup>(254)</sup>=*srvtSkVertical*, this is an index in *TabSet.SkinManager*<sup>(257)</sup>.*CurrentSkin*<sup>(230)</sup>.*VerticalTabSetSchemes*<sup>(233)</sup> collection.

If *TabSet.SkinManager*<sup>(257)</sup> = *nil*, or *TabSet.SkinManager*<sup>(257)</sup>.*CurrentSkin*<sup>(230)</sup> = *nil*, or the index is not valid for this collection (negative or too large), then this tab has a default (no-skin) appearance.

**Default value:**

-1

Stores an integer value as part of a tab.

**property** *Tag*: *Integer*;

Tag has no predefined meaning. The Tag property is provided for the convenience of developers. It can be used for storing an additional integer value or it can be typecast to any 32-bit value such as a component reference or a pointer.

**Default value:**

0

Specifies the text that appears in the tab.

**property** *Text*: *TRVUnicodeString*;

**Default value:**

" (empty string)

Allows hiding the tab without destroying it.

**property** *Visible*: *Boolean*;

**Default value:**

*True*

### 1.3.11.4.3 TSRVTabCollection

TSRVTabCollection maintains the collection of tabs that appear in TSRVTabSet<sup>(249)</sup> control.

This is a class of TSRVTabSet<sup>(249)</sup>.Tabs<sup>(259)</sup> collection.

**Unit** SRVTabSet.

#### Syntax

```
TSRVTabCollection = class (TCollection)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollection*

#### Properties

- **Items**[Index: Integer]: TSRVTabItem<sup>(264)</sup>;
- **Count**

## 1.4 Types

### Types Related to Custom Drawing

Events:

- TSRVDrawScrollBarAreaEvent<sup>(268)</sup> for drawing TSRVScrollBar<sup>(243)</sup> background area;
- TSRVDrawScrollBarEvent<sup>(268)</sup> for drawing TSRVScrollBar<sup>(243)</sup> elements;
- TSRVDrawTabEvent<sup>(269)</sup> for drawing tabs and a closing button in TSRVTabSet<sup>(249)</sup>;
- TSRVDrawTabSetEvent<sup>(270)</sup> for drawing TSRVTabSet<sup>(249)</sup> elements.

Other types:

- TSRVDrawState, TSRVDrawStates<sup>(269)</sup> – states of a painted object.

### Types of Properties of ToolBars and ToolWindows

Types of properties:

- TSRVButtonCollection<sup>(271)</sup> – collection of TSRVToolButton<sup>(272)</sup>
- TSRVAlignButton<sup>(271)</sup> defines the position of buttons;
- TSRVAlignText<sup>(271)</sup> defines the position of text.

Types of events:

- TSRVClickButtonEvent<sup>(272)</sup> occurs when a toolbar button is clicked.

### Other Types

Types of events:

- TSRVMouseEvent<sup>(277)</sup> – a type of mouse events in TSRichViewEdit<sup>(40)</sup>;

Types of properties:

TSRVControlStyle<sup>(276)</sup> defines a visual appearance of controls (if skins<sup>(228)</sup> are not applied);

TSRVFloatProperty<sup>(276)</sup> identifies a TSRichViewEdit<sup>(40)</sup> of an floating point type;

TSRVIntProperty<sup>(277)</sup> identifies a TSRichViewEdit<sup>(40)</sup> of an floating point type;  
 TSRVPageFormat<sup>(278)</sup> – a page size;  
 TSRVTHAlign, TSRVTVAlign<sup>(281)</sup> – a position in TSRVTabSet<sup>(249)</sup>.

## 1.4.1 Custom drawing types

### Types Related to Custom Drawing

Events:

- TSRVDrawScrollBarAreaEvent<sup>(268)</sup> for drawing TSRVScrollBar<sup>(243)</sup> background area;
- TSRVDrawScrollBarEvent<sup>(268)</sup> for drawing TSRVScrollBar<sup>(243)</sup> elements;
- TSRVDrawTabEvent<sup>(269)</sup> for drawing tabs and a closing button in TSRVTabSet<sup>(249)</sup>;
- TSRVDrawTabSetEvent<sup>(270)</sup> for drawing TSRVTabSet<sup>(249)</sup> elements.

Other types:

- TSRVDrawState, TSRVDrawStates<sup>(269)</sup> – states of a painted object.

#### 1.4.1.1 TSRVDrawScrollBarAreaEvent type

A type of the following events of TSRVScrollBar<sup>(243)</sup>:

- OnDrawArea<sup>(249)</sup>

Unit SRVSBBar.

**type**

```
TSRVDrawScrollBarAreaEvent = procedure (Sender: TSRVScrollBar(243);
  Canvas: TCanvas; ARect, PaintRect: TRect; isFirstArea: Boolean;
  State: TSRVDrawStates(269); var DoDefault : Boolean) of object;
```

**Parameters**

**Canvas** – a canvas where to draw.

**ARect** – a rectangle defining the position of the object to draw.

**APaintRect** – a rectangle that needs to be redrawn. You can use this parameter to optimize drawing.

**isFirstArea** – *True* if this event is called for the background area between the "dec button" and the thumb tab.

**State** – a set describing the current object state.

Set **DoDefault** to *False* to prevent the default drawing of this object.

#### 1.4.1.2 TSRVDrawScrollBarEvent type

A type of the following events of TSRVScrollBar<sup>(243)</sup>:

- OnDrawBorder<sup>(249)</sup>
- OnDrawDecButton
- OnDrawThumbTab



**Unit SRVScrollBar.**

**type**

```
TSRVDrawScrollBarEvent = procedure (Sender: TSRVScrollBar(243);
  Canvas: TCanvas; ARect, PaintRect: TRect;
  State: TSRVDrawStates(269); var DoDefault : Boolean) of object;
```

**Parameters**

**Canvas** – a canvas where to draw.

**ARect** – a rectangle defining the position of the object to draw.

**APaintRect** – a rectangle that needs to be redrawn. You can use this parameter to optimize drawing.

**State** – a set describing the current object state.

Set **DoDefault** to *False* to prevent the default drawing of this object.

### 1.4.1.3 TSRVDrawState, TSRVDrawStates types

Indicates state information that can influence how an object is drawn.

**Unit SRVSkinManager.**

**type**

```
TSRVDrawState = (srvdsHot, srvdsDown, srvdFocused, srvdsEnabled);
TSRVDrawStates = set of TSRVDrawState;
```

| State               | Meaning   |
|---------------------|---|
| <i>srvdsHot</i>     | the object is below the mouse pointer                     |
| <i>srvdsDown</i>    | the object is clicked                                     |
| <i>srvdFocused</i>  | the object is focused (or belongs to the focused control) |
| <i>srvdsEnabled</i> | the object is not disabled                                |

### 1.4.1.4 TSRVDrawTabEvent type

A type of the following events of TSRVTabSet<sup>(249)</sup>:

- OnDrawTab<sup>(260)</sup>
- OnDrawCloseButton

**Unit SRVPreview.**

**type**

```
TSRVDrawTabEvent = procedure (Sender: TSRVTabSet(249); Canvas: TCanvas;
  ARect, PaintRect: TRect; TabIndex: Integer; State: TSRVDrawStates(269);
  var DoDefault: Boolean) of object;
```

**Parameters**

**Canvas** – a canvas where to draw.

**ARect** – a rectangle defining the position of the object to draw.

**APaintRect** – a rectangle that needs to be redrawn. You can use this parameter to optimize drawing.

**TabIndex** – the index of the tab (in Tabs<sup>(259)</sup>) where this drawing should occur.

**State** – a set describing the current tab state.

Set **DoDefault** to *False* to prevent the default drawing of this tab.

#### 1.4.1.5 TSRVDrawTabSetEvent type

A type of the following events of TSRVTabSet<sup>(249)</sup>:

- OnDrawBorder<sup>(260)</sup>
- OnDrawDecButton
- OnDrawIncButton
- OnDrawBackground

**Unit** SRVTabSet.

**type**

```
TSRVDrawTabSetEvent = procedure (Sender: TSRVTabSet(249); Canvas: TCanvas;
  ARect, PaintRect: TRect; State: TSRVDrawStates(269);
  var DoDefault: Boolean) of object;
```

**Parameters**

**Canvas** – a canvas where to draw.

**ARect** – a rectangle defining the position of the object to draw.

**APaintRect** – a rectangle that needs to be redrawn. You can use this parameter to optimize drawing.

**State** – a set describing the current object state.

Set **DoDefault** to *False* to prevent the default drawing of this object.

#### 1.4.1.6 TSRVOnPaint type

A type of OnBeforePaint<sup>(228)</sup> and OnAfterPaint<sup>(228)</sup> events of TSRVPreview<sup>(223)</sup>.

**Unit** SRVPreview.

**type**

```
TSRVOnPaint = procedure (Sender: TObject; ACanvas: TCanvas;
  ARect: TRect) of object;
```

**ACanvas** – canvas where to draw.

**ARect** – a location of page(s). This area includes TSRVPreview.BorderWidth<sup>(225)</sup>, but does not include a shadow area (TSRVPreview.ShadowWidth).

When these events are called, a background color (beyond **ARect**) and a shadow is already drawn, and the clipping region is set to **ARect**.

## 1.4.2 ToolBar types

### Types of Properties of ToolBars and ToolWindows

Types of properties:

- `TSRVButtonCollection`<sup>(271)</sup> – collection of `TSRVToolButton`<sup>(272)</sup>
- `TSRVAlignButton`<sup>(271)</sup> defines the position of buttons;
- `TSRVAlignText`<sup>(271)</sup> defines the position of text.

Types of events:

- `TSRVClickButtonEvent`<sup>(272)</sup> occurs when a toolbar button is clicked.

#### 1.4.2.1 TSRVAlignButton type

Defines the position of buttons in `TSRVToolBar`<sup>(189)</sup>.

**Unit** `SRVToolBar`.

**type**

```
TSRVAlignButton = (srvabLeft, srvabRight, srvabCenter);
```

#### 1.4.2.2 TSRVAlignText type

Defines the position of text in `TSRVToolBar`<sup>(189)</sup>.

**Unit** `SRVToolBar`.

**type**

```
TSRVAlignText = (srvatTop, srvatBottom);
```

A text with the description of the button under the mouse pointer can be displayed at the top (above the buttons) or at the bottom (below the buttons) of the toolbar.

#### 1.4.2.3 TSRVButtonCollection

`TSRVButtonCollection` is a collection of `TSRVToolButton`<sup>(272)</sup>.

**Unit** `SRVToolBar`.

**Syntax**

```
TSRVButtonCollection = class (TCollection)
```

### Hierarchy

*TObject*

*TPersistent*

*TCollection*

### Description

This is a type of `TSRVToolBar.Buttons`<sup>(192)</sup>, `TSRVToolWindow.Buttons`<sup>(199)</sup>, `TSRICHViewEdit.MenuHButtons`<sup>(55)</sup>, `TSRICHViewEdit.MenuVButtons`<sup>(56)</sup> properties.

#### 1.4.2.4 TSRVClickButtonEvent type

A type for events happened to a toolbar button.

**Unit** SRVToolBar.

**type**

```
TSRVClickButtonEvent = procedure (Sender: TObject;  
    ToolButton: TSRVToolButton(272)) of object;
```

#### 1.4.2.5 TSRVToolButton

TSRVToolButton is a class containing properties of a toolbar button.

**Unit** SRVToolBar.

**Syntax**

```
TSRVToolButton = class (TCollectionItem)
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*

#### Description

This is a type of item in the TSRVToolBar.Buttons<sup>(192)</sup>, TSRVToolWindow.Buttons<sup>(199)</sup>, TSRichViewEdit.MenuHButtons<sup>(55)</sup>, TSRichViewEdit.MenuVButtons<sup>(56)</sup> collections properties.

Size and colors of buttons are defined in properties of their toolbars/tool windows.

#### 1.4.2.5.1 Properties

##### In TSRVToolButton

- AllowAllUp<sup>(273)</sup>
- Caption<sup>(273)</sup>
- Down<sup>(273)</sup>
- Enabled<sup>(273)</sup>
- GroupIndex<sup>(274)</sup>
- Hint<sup>(273)</sup>
- ImageIndexDisable<sup>(274)</sup>
- ImageIndexDown<sup>(274)</sup>
- ImageIndexMove<sup>(274)</sup>
- ImageIndexNormal<sup>(275)</sup>
- ShowHint<sup>(275)</sup>
- Tag<sup>(275)</sup>
- Visible<sup>(275)</sup>

#### 1.4.2.5.1.1 TSRVToolButton.AllowAllUp

Specifies whether all buttons in the group that contains this button can be unselected at the same time.

**property** AllowAllUp: Boolean;

Use AllowAllUp with a group of buttons that belong to the same group. Buttons belong to the same group if they have the same value on their GroupIndex<sup>(274)</sup> property. Buttons in the same group behave in a mutually exclusive fashion: when one button in the group is selected (Down<sup>(273)</sup>=True), all other buttons in the group become unselected (Down<sup>(273)</sup>=False).

If AllowAllUp is True, all of the buttons in a group can be unselected at the same time. Clicking the single selected button in the group will deselect that button without selecting another. If AllowAllUp is False, exactly one of the buttons in the group must be selected at any time. Clicking a down button won't return the button to its up state.

If GroupIndex<sup>(274)</sup> is zero, AllowAllUp has no effect.

#### 1.4.2.5.1.2 TSRVToolButton.Caption

Specifies the caption for the button.

**property** Caption: TRVUnicodeString;

If the toolbar has a hint area (above or below the buttons), buttons captions are shown there.

#### 1.4.2.5.1.3 TSRVToolButton.Enabled

Controls whether the button responds to mouse and keyboard events.

**property** Enabled : Boolean;

Use Enabled to change the availability of the button to the user. To disable a button, set Enabled to False.

**Default value:**

False

#### 1.4.2.5.1.4 TSRVToolButton.Down

Specifies whether this button is pressed.

**property** Down: Boolean;

If True, this button is pressed. Only buttons with GroupIndex<sup>(274)</sup> <> 0 can be permanently pressed.

#### 1.4.2.5.1.5 TSRVToolButton.Hint

Contains the text string that can appear when the user moves the mouse over the control.

**property** Hint: TRVUnicodeString;

Use the Hint property to provide a string of help text either as a help hint, or as help text on a particular location such as a status bar. A help hint is displayed only if ShowHint<sup>(275)</sup>=True.

If the toolbar has a hint area (below or above the buttons), the main property explaining the button for users is Caption<sup>(273)</sup>.

#### 1.4.2.5.1.6 TSrvToolButton.GroupIndex

Allows buttons to work together as a group.

**property** GroupIndex : Integer;

If GroupIndex is 0, the button works as a command button. When the user clicks such a button, the button appears pressed (in its clicked state) and then returns to its normal up state when the user releases the mouse button.

When GroupIndex is greater than 0, the button works like a checkbox or as a radio button, depending on AllowAllUp<sup>(273)</sup> property. If AllowAllUp<sup>(273)</sup> is *True*, the button works like a checkbox. If AllowAllUp<sup>(273)</sup>=*False*, buttons with the same GroupIndex property value (other than 0) work together as a group of radio buttons. Checked buttons have Down<sup>(273)</sup>=*True*.

**Default value:**

0

#### 1.4.2.5.1.7 TSrvToolButton.ImageIndexDisable

Specifies the index of the image displayed in the button, if Enabled<sup>(273)</sup>=*False*.

**property** ImageIndexDisable: Integer;

The image list is defined in the corresponding property of the component containing this button:

- TSrvToolBar.ImageList<sup>(195)</sup>
- TSrvToolWindow.ImageList<sup>(202)</sup>
- TSrvCustomPropertyTB.ImageList<sup>(130)</sup>

**Default value:**

-1 (no image)

#### 1.4.2.5.1.8 TSrvToolButton.ImageIndexDown

Specifies the index of the image displayed in the button, if it is pressed (or checked)

**property** ImageIndexDown: Integer;

The image list is defined in the corresponding property of the component containing this button:

- TSrvToolBar.ImageList<sup>(195)</sup>
- TSrvToolWindow.ImageList<sup>(202)</sup>
- TSrvCustomPropertyTB.ImageList<sup>(130)</sup>

**Default value:**

-1 (no image)

#### 1.4.2.5.1.9 TSrvToolButton.ImageIndexMove

Specifies the index of the image displayed in the button, if it is highlighted (below the mouse pointer)

**property** ImageIndexMove: Integer;

The image list is defined in the corresponding property of the component containing this button:

- TSrvToolBar.ImageList<sup>(195)</sup>
- TSrvToolWindow.ImageList<sup>(202)</sup>
- TSrvCustomPropertyTB.ImageList<sup>(130)</sup>

**Default value:**

-1 (no image)

**1.4.2.5.1.10 TSRVToolButton.ImageIndexNormal**

Specifies the index of the image displayed in the button.

**property** ImageIndexNormal: Integer;

This image is displayed in a normal state (when the button is not disabled, is not highlighted, is not pressed).

The image list is defined in the corresponding property of the component containing this button:

- TSRVToolBar.ImageList<sup>(195)</sup>
- TSRVToolWindow.ImageList<sup>(202)</sup>
- TSRVCustomPropertyTB.ImageList<sup>(130)</sup>

**Default value:**

-1 (no image)

**1.4.2.5.1.11 TSRVToolButton.ShowHint**

Determines whether a toolbar displays a help hint when the mouse pointer is above this button.

**property** ShowHint: Boolean;

The help hint's text is defined in Hint<sup>(273)</sup> property.

**Default value:**

*False*

**See also:**

- Caption<sup>(273)</sup>

**1.4.2.5.1.12 TSRVToolButton.Tag**

Stores an integer value as part of a button.

**property** Tag: Longint;

Tag has no predefined meaning. The Tag property is provided for the convenience of developers. It can be used for storing an additional integer value or it can be typecast to any 32-bit value such as a component reference or a pointer.

**Default value:**

0

**1.4.2.5.1.13 TSRVToolButton.Visible**

Specifies whether the button is visible on the toolbar.

**property** Visible: Boolean;

**Default value:**

*True*

### 1.4.3 TSRVControlStyle

Define appearance of controls.

**Unit** SRVControl.

**type**

```
TSRVControlStyle = (srvcsSimple, srvcsClassic);
```

**srvcsSimple:**

A modern, minimalistic and consistent appearance of controls.

It can be customized using RVControlsPainter class from RVControls unit. This class is described in the TRichView help file. The main its property is Theme: TRVControlTheme, where

```
TRVControlTheme = (rvctPaleBlue, rvctSpringGreen, rvctSienna, rvctHighContrast);
```

By changing Theme, you change colors of all controls having SRVControlStyle = *srvcsSimple*.

In Delphi XE2 and newer, this mode uses GDI+ for drawing, so lines are anti-aliased.

In Lazarus, GDI+ drawing is optional, see GDI+ notes <sup>(327)</sup>.

**srvcsClassic:**

A classic appearance of controls.

They look like in ScaleRichView 8 and older.

This is the type of the following properties:

- TSRVScrollBar<sup>(243)</sup>.SRVControlStyle<sup>(248)</sup>;
- TSRVTabSet<sup>(249)</sup>.SRVControlStyle<sup>(258)</sup>;
- TSRVCustomControl.SRVControlStyle<sup>(335)</sup> (published in TSRVButton<sup>(343)</sup>, TSRVCheckBox<sup>(345)</sup>, TSRVGroupBox<sup>(352)</sup>, TSRVPanel<sup>(365)</sup>, TSRVRadioButton<sup>(367)</sup>);
- TSRVWinControl.SRVControlStyle<sup>(340)</sup> (published in TSRVEdit<sup>(349)</sup>, TSRVMemo<sup>(362)</sup>).
- TSRichViewEdit<sup>(40)</sup>.ScrollBarControlStyle<sup>(65)</sup>

### 1.4.4 TSRVFloatProperty type

**Unit** ScIRView.

Identifies a property of TSRichViewEdit<sup>(40)</sup> of an integer or a floating point type.

**type**

```
TSRVFloatProperty = (srvfpPPLeftMargin, srvfpPPRightMargin,  
    srvfpPPTopMargin, srvfpPPBottomMargin,  
    srvfpPPHeaderY, srvfpPPFooterY,  
    srvfpPPPPageWidth, srvfpPPPPageHeight);
```

| Value                     | Corresponding property of TSRichViewEdit                  |
|---------------------------|---|
| <i>srvfpPPLeftMargin</i>  | PageProperty <sup>(58)</sup> .LeftMargin <sup>(138)</sup> |
| <i>srvfpPPRightMargin</i> | PageProperty.RightMargin <sup>(142)</sup>                 |
| <i>srvfpPPTopMargin</i>   | PageProperty.TopMargin <sup>(142)</sup>                   |



| Value                      | Corresponding property of TSRichViewEdit   |
|----------------------------|--|
| <i>srvfpPPBottomMargin</i> | PageProperty.BottomMargin <sup>(136)</sup> |
| <i>srvfpPPHeaderY</i>      | PageProperty.HeaderY <sup>(137)</sup>      |
| <i>srvfpPPFooterY</i>      | PageProperty.FooterY <sup>(137)</sup>      |
| <i>srvfpPPPPageWidth</i>   | PageProperty.PageWidth <sup>(141)</sup>    |
| <i>srvfpPPPPageHeight</i>  | PageProperty.PageHeight <sup>(139)</sup>   |

A parameter of this type is used in TSRichViewEdit <sup>(40)</sup>.SetFloatPropertyEd <sup>(97)</sup>.

See also:

- TSRVIntProperty <sup>(277)</sup>

### 1.4.5 TSRVIntProperty type

Unit SclRView.

Identifies a property of TSRichViewEdit <sup>(40)</sup> of an integer or an ordinal type.

**type**

```
TSRVIntProperty = (srvipPPOrientation, srvipPPPPageFormat,  
    srvipPPMirrorMargins, srvipPPFacingPages, srvipPPTitlePage);
```

| Value                       | Corresponding property of TSRichViewEdit                   |
|-----------------------------|--|
| <i>srvipPPOrientation</i>   | PageProperty <sup>(58)</sup> .Orientation <sup>(138)</sup> |
| <i>srvipPPPPageFormat</i>   | PageProperty.PageFormat <sup>(139)</sup>                   |
| <i>srvipPPMirrorMargins</i> | PageProperty.MirrorMargins <sup>(138)</sup>                |
| <i>srvipPPFacingPages</i>   | PageProperty.FacingPages <sup>(136)</sup>                  |
| <i>srvipPPTitlePage</i>     | PageProperty.TitlePage <sup>(142)</sup>                    |

A parameter of this type is used in TSRichViewEdit <sup>(40)</sup>.SetIntPropertyEd <sup>(97)</sup>.

See also:

- TSRVFloatProperty <sup>(276)</sup>

### 1.4.6 TSRVMouseEvent type

A type for several events of TSRichViewEdit <sup>(40)</sup>.

Unit SclRView.

```
TSRVMouseEvent = procedure (Sender: TSRichViewEdit (40); Button: TMouseButton;  
    Shift: TShiftState; X, Y: Integer; PageNo : Integer) of object;
```

**X** and **Y** are the pixel coordinates of the mouse pointer in the client area of the **Sender**.

**Shift** contains a set of pressed keys. **Button** identifies a mouse button.

**PageNo** is a page below the mouse pointer, from 1.

### 1.4.7 TSRVPageFormat type

A type of page format (size).

Unit SclRView.

**type**

```
TSRVPageFormat = (
    srvfmCustom, srvfm4A0, srvfm2A0, srvfmA0, srvfmA1, srvfmA2,
    srvfmA3, srvfmA4, srvfmA5, srvfmA6, srvfmA7, srvfmA8,
    srvfmA9, srvfmA10, srvfmB0, srvfmB1, srvfmB2, srvfmB3,
    srvfmB4, srvfmB5, srvfmB6, srvfmB7, srvfmB8, srvfmB9,
    srvfmB10, srvfmC0, srvfmC1, srvfmC2, srvfmC3, srvfmC4,
    srvfmC5, srvfmC6, srvfmC7, srvfmC8, srvfmC9, srvfmC10,
    srvfmLetter, srvfmLegal, srvfmLedger, srvfmTabloid,
    srvfmStatement, srvfmQuarto, srvfmFoolscap, srvfmFolio,
    srvfmExecutive, srvfmMonarch, srvfmGovernmentLetter,
    srvfmPost, srvfmCrown, srvfmLargePost, srvfmDemy,
    srvfmMedium, srvfmRoyal, srvfmElephant, srvfmDoubleDemy,
    srvfmQuadDemy, srvfmIndexCard3_5, srvfmIndexCard4_6,
    srvfmIndexCard5_8, srvfmInternationalBusinessCard,
    srvfmUSBusinessCard, srvfmEmperor, srvfmAntiquarian,
    srvfmGrandEagle, srvfmDoubleElephant, srvfmAtlas,
    srvfmColombier, srvfmImperial, srvfmDoubleLargePost,
    srvfmPrincess, srvfmCartridge, srvfmSheet, srvfmHalfPost,
    srvfmDoublePost, srvfmSuperRoyal, srvfmCopyDraught,
    srvfmPinchedPost, srvfmSmallFoolscap, srvfmBrief, srvfmPott,
    srvfmPA0, srvfmPA1, srvfmPA2, srvfmPA3, srvfmPA4, srvfmPA5,
    srvfmPA6, srvfmPA7, srvfmPA8, srvfmPA9, srvfmPA10, srvfmF4,
    srvfmA0a, srvfmJISB0, srvfmJISB1, srvfmJISB2, srvfmJISB3,
    srvfmJISB4, srvfmJISB5, srvfmJISB6, srvfmJISB7, srvfmJISB8,
    srvfmJISB9, srvfmJISB10, srvfmJISB11, srvfmJISB12,
    srvfmANSI_A, srvfmANSI_B, srvfmANSI_C, srvfmANSI_D,
    srvfmANSI_E, srvfmArch_A, srvfmArch_B, srvfmArch_C,
    srvfmArch_D, srvfmArch_E, srvfmArch_E1);
```

#### Custom Format

*srvfmCustom* – user-defined paper format.

#### ISO "A" Paper Sizes (mm x mm)

*srvfmA0* – paper format A0, 841 x 1189

*srvfmA1* – paper format A1, 594 x 841

*srvfmA2* – paper format A2, 420 x 594

*srvfmA3* – paper format A3, 297 x 420

*srvfmA4* – paper format A4, 210 x 297

*srvfmA5* – paper format A5, 148 x 210

*srvfmA6* – paper format A6, 105 x 148

*srvfmA7* – paper format A7, 74 x 105

*srvfmA8* – paper format A8, 52 x 74  
*srvfmA9* – paper format A9, 37 x 52  
*srvfmA10* – paper format A10, 26 x 37

**ISO "A" Paper Sizes, German Extensions (mm x mm)**

*srvfm4A0* – paper format 4A0, 1682 x 2378  
*srvfm2A0* – paper format 2A0, 1189 x 1682

**ISO "B" Paper Sizes (mm x mm)**

*srvfmB0* – paper format B0, 1000 x 1414  
*srvfmB1* – paper format B1, 707 x 1000  
*srvfmB2* – paper format B2, 500 x 707  
*srvfmB3* – paper format B3, 353 x 500  
*srvfmB4* – paper format B4, 250 x 353  
*srvfmB5* – paper format B5, 176 x 250  
*srvfmB6* – paper format B6, 125 x 176  
*srvfmB7* – paper format B7, 88 x 125  
*srvfmB8* – paper format B8, 62 x 88  
*srvfmB9* – paper format B9, 44 x 62  
*srvfmB10* – paper format B10, 31 x 44

**ISO "C" Paper Sizes (mm x mm)**

*srvfmC0* – paper format C0, 917 x 1297  
*srvfmC1* – paper format C1, 648 x 917  
*srvfmC2* – paper format C2, 458 x 648  
*srvfmC3* – paper format C3, 324 x 458  
*srvfmC4* – paper format C4, 229 x 324  
*srvfmC5* – paper format C5, 162 x 229  
*srvfmC6* – paper format C6, 114 x 162  
*srvfmC7* – paper format C7, 81 x 114  
*srvfmC8* – paper format C8, 57 x 81  
*srvfmC9* – paper format C9, 40 x 57  
*srvfmC10* – paper format C10, 28 x 40

**North American Paper Sizes (inch x inch)**

*srvfmLetter* – letter paper format, 8.5 x 11  
*srvfmLegal* – legal paper format, 8.5 x 14  
*srvfmLedger* – tabloid (ledger) paper format, 11 x 17  
*srvfmStatement* – statement (half letter) paper format, 5.5 x 8.5  
*srvfmQuarto* – quarto paper format, 8 x 10  
*srvfmFoolscap* – foolscap (folio) paper format, 8 x 13  
*srvfmFolio* – foolscap (folio) paper format, 8 x 13  
*srvfmExecutive* – executive (monarch) paper format, 7.25 x 10.5  
*srvfmMonarch* – executive (monarch) paper format, 7.25 x 10.5  
*srvfmGovernmentLetter* – government-letter paper format, 8 x 10.5  
*srvfmPost* – post paper format, 15.5 x 19.5  
*srvfmCrown* – crown paper format, 15 x 20  
*srvfmLargePost* – large post paper format, 16.5 x 21  
*srvfmDemy* – demy paper format, 17.5 x 22.5  
*srvfmMedium* – medium paper format, 18 x 23

*srvfmRoyal* – royal paper format, 20 x 25  
*srvfmElephant* – elephant paper format, 23 x 28  
*srvfmDoubleDemy* – double demy paper format, 22.5 x 35  
*srvfmQuadDemy* – quad demy paper format, 35 x 45

#### **Index and Business Cards (inch x inch)**

*srvfmIndexCard3\_5* – index card 3 x 5  
*srvfmIndexCard4\_6* – index card 4 x 6  
*srvfmIndexCard5\_8* – index card 5 x 8  
*srvfmInternationalBusinessCard* – international business card, 2.125 x 3.37  
*srvfmUSBusinessCard* – US business card, 2 x 3.5

#### **Traditional Paper Sizes (inch x inch)**

*srvfmEmperor* – emperor paper format, 48 x 72  
*srvfmAntiquarian* – antiquarian paper format, 31 x 53  
*srvfmGrandEagle* – grand eagle paper format, 28.75 x 42  
*srvfmDoubleElephant* – double elephant paper format, 26.75 x 40  
*srvfmAtlas* – atlas paper format, 26 x 34  
*srvfmColombier* – colombier paper format, 23.5 x 34.5  
*srvfmImperial* – imperial paper format, 22 x 30  
*srvfmDoubleLargePost* – double large post paper format, 21 x 33  
*srvfmPrincess* – princess paper format, 21.5 x 28  
*srvfmCartridge* – cartridge paper format, 21 x 26  
*srvfmSheet* – sheet (half post) paper format, 19.5 x 23.5  
*srvfmHalfPost* – sheet (half post) paper format, 19.5 x 23.5  
*srvfmDoublePost* – double post paper format, 19 x 30.5  
*srvfmSuperRoyal* – super royal paper format, 19 x 27  
*srvfmCopyDraught* – copy draught paper format, 16 x 20  
*srvfmPinchedPost* – pinched post paper format, 14.75 x 18.5  
*srvfmSmallFoolscap* – small foolscap paper format, 13.25 x 16.5  
*srvfmBrief* – brief paper format, 13.5 x 16  
*srvfmPott* – pott paper format, 12.5 x 15

#### **PA Series (ISO 216) (mm x mm)**

*srvfmPA0* – paper format PA0, 840 x 1120  
*srvfmPA1* – paper format PA1, 560 x 840  
*srvfmPA2* – paper format PA2, 420 x 560  
*srvfmPA3* – paper format PA3, 280 x 420  
*srvfmPA4* – paper format PA4, 210 x 280  
*srvfmPA5* – paper format PA5, 140 x 210  
*srvfmPA6* – paper format PA6, 105 x 140  
*srvfmPA7* – paper format PA7, 70 x 105  
*srvfmPA8* – paper format PA8, 52 x 70  
*srvfmPA9* – paper format PA9, 35 x 52  
*srvfmPA10* – paper format PA10, 26 x 35

#### **Other Paper Sizes (mm x mm)**

*srvfmF4* – 210 x 330  
*srvfmA0a* – paper format A0a, 1000 x 1370

**JIS B-Series (Japanese) (mm x mm)**

*srvfmJISB0* – paper format JIS B0, 1030 x 1456  
*srvfmJISB1* – paper format JIS B1, 728 x 1030  
*srvfmJISB2* – paper format JIS B2, 515 x 728  
*srvfmJISB3* – paper format JIS B3, 364 x 515  
*srvfmJISB4* – paper format JIS B4, 257 x 364  
*srvfmJISB5* – paper format JIS B5, 182 x 257  
*srvfmJISB6* – paper format JIS B6, 128 x 182  
*srvfmJISB7* – paper format JIS B7, 91 x 128  
*srvfmJISB8* – paper format JIS B8, 64 x 91  
*srvfmJISB9* – paper format JIS B9, 45 x 64  
*srvfmJISB10* – paper format JIS B10, 32 x 45  
*srvfmJISB11* – paper format JIS B11, 22 x 32  
*srvfmJISB12* – paper format JIS B12, 16 x 22

**ANSI Paper Sizes (inch x inch)**

*srvfmANSI\_A* – paper format ANSI A, 8.5 x 11  
*srvfmANSI\_B* – paper format ANSI B, 11 x 17  
*srvfmANSI\_C* – paper format ANSI C, 17 x 22  
*srvfmANSI\_D* – paper format ANSI D, 22 x 34  
*srvfmANSI\_E* – paper format ANSI E, 34 x 44

**Architectural Sizes (inch x inch)**

*srvfmArch\_A* – paper size Arch A, 12 x 9  
*srvfmArch\_B* – paper size Arch B, 18 x 12  
*srvfmArch\_C* – paper size Arch C, 24 x 18  
*srvfmArch\_D* – paper size Arch D, 36 x 24  
*srvfmArch\_E* – paper size Arch E, 48 x 36  
*srvfmArch\_E1* – paper size Arch E1, 42 x 30

**1.4.8 TSRVTHAlign, TSRVTVAlign types**

Define the position of tab closing buttons in *TSRVTabSet* <sup>(249)</sup>.

**Unit** *SRVTabSet*.

**type**

```

TSRVTHAlign = (srvthaLeft, srvthaCenter, srvthaRight);
TSRVTVAlign = (srvtvaTop, srvtvaCenter, srvtvaBottom);

```

These are the types of the following properties:

- *TSRVCloseButton.HorizontalAlign* <sup>(263)</sup>

**1.5 Global variables**

**Unit** *SclRView*;

**var**

```

MouseWheelZoomMin: Integer = 10;
MouseWheelZoomMax: Integer = 500;
MouseWheelZoomSpeed : Single = 1.0;

```

The following variables defines parameters for zooming using the mouse wheel <sup>(162)</sup>:

**MouseWheelZoomMin** – minimal value of ZoomPercent <sup>(168)</sup>

**MouseWheelZoomMax** – maximal value of ZoomPercent <sup>(168)</sup>


**MouseWheelZoomSpeed** defines how fast ZoomPercent <sup>(168)</sup> changes.


## 2 ScaleRichView Actions

Almost all RichViewActions designed for TRichViewEdit can be used with TScaleRichViewEdit <sup>(40)</sup>.

However, printing actions (TrvActionPrint, TrvActionQuickPrint, TrvActionPrintPreview, TrvActionPageSetup) cannot be used. Also, it is highly preferred to use ScaleRichView versions of the note actions (do not use TrvActionInsertFootnote, TrvActionInsertEndnote, TrvActionInsertSidenote, TrvActionInsertTextBox, TrvActionEditNote).

### Print and preview:

 TsrvActionQuickPrint <sup>(294)</sup>


 TsrvActionPrint <sup>(293)</sup>


 TsrvActionPreview <sup>(293)</sup>


### Page setup:

 TsrvActionPageSetup <sup>(291)</sup>

 TsrvActionOrientationPortrait <sup>(288)</sup> changes page orientation to "portrait".

 TsrvActionOrientationLandscape <sup>(287)</sup> changes page orientation to "landscape".

 TsrvActionPageFormat <sup>(289)</sup> changes page size.

 TsrvActionLineNumbers <sup>(287)</sup> shows or hides line numbers.

### View:

 TsrvActionThumbnails <sup>(311)</sup> shows/hides page thumbnails.

 TsrvActionLayoutDraft <sup>(299)</sup> switches the editor to a draft layout.


 TsrvActionLayoutWeb <sup>(310)</sup> switches the editor to a web layout.


 TsrvActionLayoutPrint <sup>(301)</sup> switches the editor to a print layout.


 TsrvActionLayoutSideToSide <sup>(307)</sup> switches the editor to a side-to-side layout.

 TsrvActionLayoutRead <sup>(304)</sup> switches the editor to a read mode layout.


### Zoom:


 TsrvActionZoom <sup>(313)</sup> applies the specified zooming percent to the editor.


 TsrvActionZoomPageWidth <sup>(314)</sup> zooms the editor to fit the page width in the editor width.

 TsrvActionZoomFullPage <sup>(314)</sup> zooms the editor to fit the full page in the editor.

### Header and footer:


 TsrvActionEditHeader <sup>(315)</sup> starts editing a page header.


 TsrvActionEditFooter <sup>(315)</sup> starts editing a page footer.




 TsrvActionEditMain <sup>(316)</sup> closes a header/footer editor and returns the main document.

### Notes and text boxes:

 TsrvActionInsertFootnote <sup>(320)</sup> inserts a footnote.

 TsrvActionInsertEndnote <sup>(319)</sup> inserts an endnote.

 TsrvActionInsertSidenote <sup>(322)</sup> inserts a sidenote (a note in a floating box).

-  TsrvActionInsertTextBox<sup>(324)</sup> inserts a text box item.
-  TsrvActionEditNote<sup>(319)</sup> starts editing a note or a text box.
-  TsrvActionReturnToNote<sup>(325)</sup> moves the caret to the parent note.

**See also:**

translating actions<sup>(283)</sup>

## 2.1 Localization

### General information

RichViewActions require localization, even if you use only one UI language.

Initially, the language specified in TRVAControlPanel.Language is used. If TRVAControlPanel component is not used, 'English (US)' is used. This language will be used in dialogs displayed by RichViewActions, but Captions and Hints of the actions themselves are not localized yet. Call RVA\_LocalizeForm to localize them.

For changing UI language, use RVA\_ChooseLanguage function. The topic about this function contains an example.

Other localization functions are declared in RVALocalize and SRVALocalize units.

### If you want to add a translation to you language (for example, Greek):

1. First, a translation RichViewActions for TRichViewEdit must be done.
2. Open SRVAL\_EngUS.pas, rename it to SRVAL\_Greek.pas. Add SRVAL\_Greek in "uses" of SRVALocalize.pas. (If you use Delphi/C++Builder 2009+, save this unit as UTF-8).
3. Translate all text in this unit (do not translate comments!)
4. Change the call of SRVA\_RegisterLanguage to SRVA\_RegisterLanguage('Greek', @Messages).
5. Send the translated file to [svt@trichview.com](mailto:svt@trichview.com) for including in the main installation of ScaleRichView. Your file will be maintained (it will be corrected when new messages will be added in the future versions of RichViewActions and ScaleRichView)

### Procedures and functions

The unit SRVALocalize contains the following procedures and functions:

```
function SRVA_GetS(MsgID: TSRVAMessageID;
  ControlPanel: TRVAControlPanel = nil): TRVALocString;
function SRVA_GetPC(MsgID: TSRVAMessageID;
  ControlPanel: TRVAControlPanel): PChar;
procedure SRVA_LocalizeToolWindow(ToolWindow: TSRVToolWindow(197);
  ControlPanel: TRVAControlPanel = nil);
procedure SRVA_LocalizeSRichViewEdit(Edit: TSRichViewEdit(40);
  ControlPanel: TRVAControlPanel = nil);
function SRVA_GetUnitsName(Units: TRVUnits;
  ControlPanel: TRVAControlPanel): TRVALocString;
```

SRVA\_GetS and SRVA\_GetPC return the text of the message by its identifier.

SRVA\_LocalizeToolWindow localizes a typical search tool window, assuming that it has buttons in the following order:

- go to next/previous page,
- find table
- find picture
- find heading
- find hyperlink
- find text

The following constants are defined:

|                       |      |
|-----------------------|------|
| SRV_TOOLWIN_PAGE      | = 0; |
| SRV_TOOLWIN_TABLE     | = 1; |
| SRV_TOOLWIN_PICTURE   | = 2; |
| SRV_TOOLWIN_HEADLINE  | = 3; |
| SRV_TOOLWIN_HYPERLINK | = 4; |
| SRV_TOOLWIN_TEXT      | = 5; |

SRVA\_LocalizeSRichViewEdit localizes a TSRichViewEdit control. It changes the following properties of ViewProperty<sup>(69)</sup>:

- HintPrefixText<sup>(160)</sup>;
- Texts<sup>(163)</sup>

SRVA\_GetUnitsName returns name of the given measuring unit.

## 2.2 Basic actions

### Basic Actions

The following classes are ancestors for actions:

- TsrvAction<sup>(284)</sup> – the parent action for all ScaleRichView actions;
- TsrvCustomActionThatNeedsDocProps<sup>(285)</sup> – the parent action for ScaleRichView actions than need access to file properties.

### 2.2.1 TsrvAction

TrvAction is the base class for all RichViewActions working with TSRichViewEdit<sup>(40)</sup>.

**Unit** SRVActions;

#### Syntax

```
TsrvAction = class (TrvCustomAction)
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*



## Description

This action is not used directly, but all RichViewActions working with TSRichViewEdit<sup>(40)</sup> are inherited from it.

### 2.2.1.1 Properties

#### In TsrvAction

■ Control<sup>(285)</sup>

#### Inherited from TrvCustomAction

■ Disabled

#### 2.2.1.1.1 TsrvAction.Control

Defines the editor for this action.

```
property Control: T(40)SRichViewEdit;
```

If T<sup>(40)</sup>SRichViewEdit component is assigned to this property, the action works with this component.

Otherwise, if T<sup>(40)</sup>SRichViewEdit component is focused, the action works with it.

Otherwise, it works with the first found T<sup>(40)</sup>SRichViewEdit component.

### 2.2.2 TsrvCustomActionThatNeedsDocProps

TsrvCustomActionThatNeedsDocProps is the basic class for actions that needs access to the document properties maintained by TrvActionSave.

**Unit** SRVActions;

#### Syntax

```
TsrvCustomActionThatNeedsDocProps = class (TsrvAction(284))
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>

## Description

This class is not used directly. The following actions are inherited from it:

- TsrvActionPrint<sup>(293)</sup>
- TsrvActionQuickPrint<sup>(294)</sup>

- TsrvActionPageSetup <sup>(291)</sup>
- TsrvActionPageFormat <sup>(289)</sup>
- TsrvActionLayoutDraft <sup>(299)</sup>
- TsrvActionLayoutPrint <sup>(301)</sup>
- TsrvActionLayoutWeb <sup>(310)</sup>

### 2.2.2.1 Properties

#### In TsrvCustomActionThatNeedsDocProps

- ActionSave <sup>(286)</sup>

#### Inherited from TsrvAction <sup>(284)</sup>

- Control <sup>(285)</sup>

#### Inherited from TrvCustomAction

- Disabled

#### 2.2.2.1.1 TsrvCustomActionThatNeedsDocProps.ActionSave




A link to TrvActionSave.

**property** ActionSave: TrvActionSave;






If this property is not assigned, the action searches for the first TrvActionSave action on the same form/datamodule, and uses it.

## 2.3 Print, preview, page setup

### Print and preview:

-  TsrvActionQuickPrint <sup>(294)</sup>
-  TsrvActionPrint <sup>(293)</sup>
-  TsrvActionPreview <sup>(293)</sup>

### Page setup:

-  TsrvActionPageSetup <sup>(291)</sup>
-  TsrvActionOrientationPortrait <sup>(288)</sup> changes page orientation to "portrait".
-  TsrvActionOrientationLandscape <sup>(287)</sup> changes page orientation to "landscape".
-  TsrvActionPageFormat <sup>(289)</sup> changes page size.
-  TsrvActionLineNumbers <sup>(287)</sup> shows or hides line numbers.

### 2.3.1 TsrvActionCustomOrientation

TsrvActionCustomOrientation is the base class for actions changing page orientation.

**Unit** SRVActions;

#### Syntax

TsrvActionCustomOrientation = **class** (TsrvAction <sup>(284)</sup>)

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>284</sup>

## Description

This action is not used directly.

The following actions are inherited from it:

- *TsrvActionOrientationPortrait* <sup>288</sup>
- *TsrvActionOrientationLandscape* <sup>287</sup>

### 2.3.2 TsrvActionLineNumbers

*TsrvActionLineNumbers* toggles visibility of line numbers

**Unit** SRVActions;

#### Syntax

```
TsrvActionLineNumbers = class(TsrvAction 284)
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>284</sup>

## Description

This action changes value of *LineNumberProperty* <sup>54</sup>.*Visible* <sup>133</sup> property.

### 2.3.3 TsrvActionOrientationLandscape

*TsrvActionOrientationLandscape* applies a landscape page orientation the the document in the target editor.

**Unit** SRVActions;

#### Syntax

```
TsrvActionOrientationLandscape = class(TsrvAction 284)
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>

### Description

This action assigns `PageProperty(58).Orientation(138) := poLandscape`.

If `PageProperty(58).PageFormat(139) = srvfmCustom`, and `PageProperty(58).PageWidth(141) < PageProperty(58).PageHeight(139)`, the action exchanges values of these properties.

The properties are changed as an editing operation (undoable).

#### See also:

- `TsrvActionOrientationPortrait` <sup>(288)</sup>

## 2.3.4 TsrvActionOrientationPortrait

`TsrvActionOrientationPortrait` applies a portrait page orientation the the document in the target editor.

**Unit** SRVActions;

#### Syntax

```
TsrvActionOrientationPortrait = class (TsrvAction(284))
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>

### Description

This action assigns `PageProperty(58).Orientation(138) := poPortrait`.

If `PageProperty(58).PageFormat(139) = srvfmCustom`, and `PageProperty(58).PageWidth(141) > PageProperty(58).PageHeight(139)`, the action exchanges values of these properties.

The properties are changed as an editing operation (undoable).

**See also:**

- [TsrvActionOrientationLandscape](#)<sup>(287)</sup>

## 2.3.5 TsrvActionPageFormat

[TsrvActionPageFormat](#) changes page size for the document in the target editor.

**Unit** SRVActions;

**Syntax**

```
TsrvActionPageFormat = class (TsrvCustomActionThatNeedsDocProps(285))
```

**▼ Hierarchy**

[TObject](#)  
[TPersistent](#)  
[TComponent](#)  
[TBasicAction](#)  
[TContainedAction](#)  
[TCustomAction](#)  
[TAction](#)  
[TsvCustomAction](#)  
[TsrvAction](#)<sup>(284)</sup>  
[TsrvCustomActionThatNeedsDocProps](#)<sup>(285)</sup>

### Description

This value changes the following properties of the target editor:

- [PageProperty](#)<sup>(58)</sup>.[PageFormat](#)<sup>(139)</sup> (see [PageFormat](#)<sup>(290)</sup>)
- [PageProperty](#)<sup>(58)</sup>.[PageWidth](#)<sup>(141)</sup> (see [PageWidth](#)<sup>(290)</sup>)
- [PageProperty](#)<sup>(58)</sup>.[PageHeight](#)<sup>(139)</sup> (see [PageHeight](#)<sup>(290)</sup>)

The properties are changed as an editing operation (undoable).

### 2.3.5.1 Properties

#### In TsrvActionLayoutPrint

- [PageFormat](#)<sup>(290)</sup>
- [PageHeight](#)<sup>(290)</sup>
- [PageWidth](#)<sup>(290)</sup>
- [Units](#)<sup>(290)</sup>

#### Inherited from TsrvCustomActionThatNeedsDocProps

- [ActionSave](#)<sup>(286)</sup>

#### Inherited from TsrvAction<sup>(284)</sup>

- [Control](#)<sup>(285)</sup>

## Inherited from TrvCustomAction

---

■ Disabled

### 2.3.5.1.1 TsrvActionPageFormat.PageFormat

Specifies a page format to apply to the document.

**property** PageFormat: TSRVPageFormat<sup>(278)</sup>;

If PageFormat=*srvfmCustom*, PageWidth<sup>(290)</sup> and PageHeight<sup>(290)</sup> properties are used.

When this property is assigned, Caption and Hints are changed.

**Default value:**

*srvfmA4*

### 2.3.5.1.2 TsrvActionPageFormat.PageHeight

Specifies a page height to apply to the document.

**property** PageHeight: Extended;

This property is used if PageFormat<sup>(290)</sup>=*srvfmCustom*.

This property is measured in Units<sup>(290)</sup>.

When this property is assigned, Caption and Hints are changed.

**See also:**

- PageWidth<sup>(290)</sup>.

**Default value:**

297.0 (height of A4 in mm)

### 2.3.5.1.3 TsrvActionPageFormat.PageWidth

Specifies a page width to apply to the document.

**property** PageWidth: Extended;

This property is used if PageFormat<sup>(290)</sup>=*srvfmCustom*.

This property is measured in Units<sup>(290)</sup>.

When this property is assigned, Caption and Hints are changed.

**See also:**

- PageHeight<sup>(290)</sup>.

**Default value:**

210.0 (width of A4 in mm)

### 2.3.5.1.4 TsrvActionPageFormat.Units

Specifies measuring units for PageWidth<sup>(290)</sup> and PageHeight<sup>(290)</sup> properties.

**property** PageWidth: Extended;

When this property is assigned, Caption and Hints are changed.

**Default value:**

*rvuMillimeters*

## 2.3.6 TsrvActionPageSetup

TsrvActionPageSetup is the action for "File | Page Setup" command.

**Unit** SRVActions;

**Syntax**

```
TsrvActionPageSetup = class (TsrvCustomActionThatNeedsDocProps285)
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>284</sup>  
*TsrvCustomActionThatNeedsDocProps*<sup>285</sup>

## Description

This action displays a page setup dialog.

It changes the following properties of TSRichViewEdit.PageProperty<sup>58</sup>:

- LeftMargin<sup>138</sup>
- RightMargin<sup>142</sup>
- TopMargin<sup>142</sup>
- HeaderY<sup>137</sup>
- FooterY<sup>137</sup>
- BottomMargin<sup>136</sup>
- Orientation<sup>138</sup>

The properties above are changed as an editing operation (undoable).

Additionally, it allows changing a paper source for the current printer (if available).

### 2.3.6.1 Properties

#### In TsrvActionPageSetup

- PageFormats1..PageFormats6<sup>292</sup>

#### Inherited from TsrvCustomActionThatNeedsDocProps

- ActionSave<sup>286</sup>

## Inherited from TsrvAction<sup>(284)</sup>

■ Control<sup>(285)</sup>

## Inherited from TrvCustomAction

■ Disabled

### 2.3.6.1.1 TsrvActionPageSetup.PageFormats\*

Lists of page formats initially available in the page setup dialog.

**type**

```
// ISO A & B
TSRVPageFormatAB = srvfm4A0..srvfmB10;
// ISO C
TSRVPageFormatC = srvfmC0..srvfmC10;
// North American
TSRVPageFormatUSA = srvfmLetter..srvfmQuadDemy;
// Index/business cards & Traditional
TSRVPageFormatCardTrad = srvfmIndexCard3_5..srvfmPott;
// PA & Other & JIS B
TSRVPageFormatPAOtherJISB = srvfmPA0..srvfmJISB12;
// ANSI & Architectural
TSRVPageFormatANSIArch = srvfmANSI_A..srvfmArch_E1;

TSRVPageFormats = set of TSRVPageFormat(278);
TSRVPageFormatsAB = set of TSRVPageFormatAB;
TSRVPageFormatsC = set of TSRVPageFormatC;
TSRVPageFormatsUSA = set of TSRVPageFormatUSA;
TSRVPageFormatsCardTrad = set of TSRVPageFormatCardTrad;
TSRVPageFormatsPAOtherJISB = set of TSRVPageFormatPAOtherJISB;
TSRVPageFormatsANSIArch = set of TSRVPageFormatANSIArch;
```

```
property PageFormats1: TSRVPageFormatsAB;
property PageFormats2: TSRVPageFormatsC;
property PageFormats3: TSRVPageFormatsUSA;
property PageFormats4: TSRVPageFormatsCardTrad;
property PageFormats5: TSRVPageFormatsPAOtherJISB;
property PageFormats6: TSRVPageFormatsANSIArch;
```

All these types are subsets of TSRVPageFormat<sup>(278)</sup> type. Formats listed in these properties are added in the combo box on the page setup dialog. The user can choose any page format using a menu.

**Default values:**

[*srvfmA4*, *srvfmA5*, *srvfmA5*] in PageFormats1  
 [*srvfmLetter*, *srvfmLegal*] in PageFormats3  
 [] in all other



## 2.3.6.2 Events

### In TsrvActionPageSetup

■ OnChange<sup>(293)</sup>

#### 2.3.6.2.1 TsrvActionPageSetup.OnChange

Occurs after the changes made in the page setup dialog are applied.

**property** OnChange: TNotifyEvent;

## 2.3.7 TsrvActionPreview

TsrvActionPreview switches the editor to the preview mode and back.

**Unit** SRVActions;

### Syntax

TsrvActionPreview = **class** (TsrvAction<sup>(284)</sup>)

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>

### Description

This action toggles the value of *SRichViewEdit.ViewProperty*<sup>(69)</sup>.*ViewMode*<sup>(165)</sup>

## 2.3.8 TsrvActionPrint

TsrvActionPrint is the action for "File | Print..." command.

**Unit** SRVActions;

### Syntax

TsrvActionPrint = **class** (TsrvCustomPrintAction<sup>(296)</sup>)

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*

*TrvCustomAction*

*TsrvAction* <sup>(284)</sup>

*TsrvCustomActionThatNeedsDocProps* <sup>(285)</sup>

*TsrvCustomPrintAction* <sup>(296)</sup>

## Description

This action displays a printing dialog (TPrintDialog) and prints the document from TSRichViewEdit <sup>(40)</sup> component.

The action can print the whole document or the range of pages.

This action is enabled only if at least one printer is installed in the system (Printer.Printers.Count>0).

### See also:

- TsrvActionQuickPrint <sup>(294)</sup>
- TRVAControlPanel.MetafileCompatibility (in the RichViewActions help file)

## 2.3.8.1 Properties

### In TsrvActionPrint

■ Title <sup>(294)</sup>

### Inherited from TsrvCustomActionThatNeedsDocProps <sup>(285)</sup>

■ ActionSave <sup>(286)</sup>

### Inherited from TsrvAction <sup>(284)</sup>

■ Control <sup>(285)</sup>

### Inherited from TrvCustomAction

■ Disabled

#### 2.3.8.1.1 TsrvActionPrint.Title

Determines the text that is listed in the Print Manager.

**property** Title: **string**;

### See also:

TsrvActionQuickPrint.Title <sup>(295)</sup>

## 2.3.9 TsrvActionQuickPrint

TsrvActionQuickPrint is the action for the "quick print" command (usually assigned to a toolbar button).

**Unit** SRVActions;

### Syntax

TsrvActionQuickPrint = **class** (TsrvCustomPrintAction <sup>(296)</sup>)

## ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>  
*TsrvCustomActionThatNeedsDocProps* <sup>(285)</sup>  
*TsrvCustomPrintAction* <sup>(296)</sup>

## Description

This action prints the document from *TSRichViewEdit* <sup>(40)</sup> component without displaying a printing dialog.

This action is enabled only if at least one printer is installed in the system (Printer.Printers.Count>0).

### See also:

- *TsrvActionPrint* <sup>(293)</sup>
- *TRVAControlPanel.MetafileCompatibility* (in the RichViewActions help file)

## 2.3.9.1 Properties

### In *TsrvActionQuickPrint*

■ Title <sup>(295)</sup>

### Inherited from *TsrvCustomActionThatNeedsDocProps* <sup>(285)</sup>

■ ActionSave <sup>(286)</sup>

### Inherited from *TsrvAction* <sup>(284)</sup>

■ Control <sup>(285)</sup>

### Inherited from *TrvCustomAction*

■ Disabled

#### 2.3.9.1.1 *TsrvActionQuickPrint*.Title

Determines the text that is listed in the Print Manager.

**property** Title: **string**;

### See also:

*TsrvActionPrint*.Title <sup>(294)</sup>

### 2.3.10 TsrvCustomPrintAction

TsrvCustomPrintAction is the base class for all printing RichViewActions working with TSRichViewEdit<sup>(40)</sup>.

**Unit** SRVActions;

#### Syntax

```
TsrvCustomPrintAction = class (TsrvCustomActionThatNeedsDocProps(285))
```

#### ▼ Hierarchy

TObject  
 TPersistent  
 TComponent  
 TBasicAction  
 TContainedAction  
 TCustomAction  
 TAction  
 TrvCustomAction  
 TsrvAction<sup>(284)</sup>  
 TsrvCustomActionThatNeedsDocProps<sup>(285)</sup>







#### Description

This action is not used directly, but all printing RichViewActions working with TSRichViewEdit<sup>(40)</sup> are inherited from it:




- TsrvActionQuickPrint<sup>(294)</sup>
- TsrvActionPrint<sup>(293)</sup>

## 2.4 View

#### View:

-  TsrvActionThumbnails<sup>(311)</sup> shows/hides page thumbnails.
-  TsrvActionLayoutDraft<sup>(299)</sup> switches the editor to a draft layout.
-  TsrvActionLayoutWeb<sup>(310)</sup> switches the editor to a web layout.
-  TsrvActionLayoutPrint<sup>(301)</sup> switches the editor to a print layout.
-  TsrvActionLayoutSideToSide<sup>(307)</sup> switches the editor to a side-to-side layout.
-  TsrvActionLayoutRead<sup>(304)</sup> switches the editor to a read mode layout.

#### Zoom:

-  TsrvActionZoom<sup>(313)</sup> applies the specified zooming percent to the editor.
-  TsrvActionZoomPageWidth<sup>(314)</sup> zooms the editor to fit the page width in the editor width.
-  TsrvActionZoomFullPage<sup>(314)</sup> zooms the editor to fit the full page in the editor.

### 2.4.1 TsrvActionCustomLayout

TsrvActionCustomLayout is a base class for actions changing view modes for TSRichViewEdit<sup>(40)</sup> control.

**Unit** SRVActions;

## Syntax

```
TsrvActionCustomLayout = class (TsrvCustomActionThatNeedsDocProps285)
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>284</sup>  
*TsrvCustomActionThatNeedsDocProps*<sup>285</sup>

## Description

This action is not used directly.

The following actions are inherited from it:

- TsrvActionLayoutDraft<sup>299</sup>
- TsrvActionLayoutWeb<sup>310</sup>
- TsrvActionLayoutPrint<sup>301</sup>
- TsrvActionLayoutSideToSide<sup>307</sup>
- TsrvActionLayoutRead<sup>304</sup>

After the action changes a layout, OnExecuted<sup>298</sup> event occurs.

These action use TrvActionSave to store additional information for the document in the target editor.

## Comparison of layouts

| Parameter  | Draft | Web layout | Print layout | Side to side | Read mode |
|--|-------|------------|--------------|--------------|-----------|
| Multiple pages   |       |            | +            | +            | +         |
| Pages are displayed at their original size (with zoom applied) |       |            | +            | +            |           |
| Page width depends on editor width                             |       | +          |              |              | +         |
| Page height depends on editor height                           |       |            |              |              | +         |

|  |  |   |  |   |   |
|--|--|---|--|---|---|
| Auto zoom  |  |   |  | + |   |
| Zoom scales content without scaling pages<br>(zoom does not affect page width) |  | + |  |   | + |
| Page flip effects  |  |   |  | + | + |

### 2.4.1.1 Events

#### In TsrvActionCustomLayout

■ Control<sup>(285)</sup>

##### 2.4.1.1.1 TsrvActionCustomLayout.OnExecuted

Occurs after the action applies the layout

**property** OnExecuted: TNotifyEvent;

This event may be useful if you have buttons for document view modes in the scrollbar area of TSRichViewEdit<sup>(40)</sup> control.

You can use this event to synchronize buttons with the actions.

**Example:**

(assuming that the buttons are in the following order: draft, web layout, print layout)

```
// SRichViewEdit1.OnHMenuClickButton(109)
procedure TfrmMain.SRichViewEdit1HMenuClickButton(Sender: TObject;
  ToolButton: TSrvToolButton(272));
begin
  if ToolButton = nil then Exit;
  SRichViewEdit1.CanUpdate(49) := False;
  case (ToolButton.Index) of
    0 : srvActionLayoutDraft1.Execute;
    1 : srvActionLayoutWeb1.Execute;
    2 : srvActionLayoutPrint1.Execute;
    3 : srvActionLayoutSideToSide1.Execute;
    4 : srvActionLayoutRead1.Execute;
  end;
  ActiveEditor.CanUpdate(49) := True;
end;
// After doc view actions are executed
// TsrvActionLayoutDraft(299)
procedure TfrmMain.srvActionLayoutDraft1Executed(Sender: TObject);
begin
  TSrvToolButton(272)(SRichViewEdit1.MenuHButtons(55).Items(271)[0]).Down(273) :=
    True;
```

```

end;
// TsrvActionLayoutWeb310
procedure TfrmMain.srvActionLayoutWeb1Executed(Sender: TObject);
begin
    TsrvToolButton272(SRichViewEdit1.MenuHButtons55.Items271[1]).Down273 :=
        True;
end;
// TsrvActionLayoutPrint301
procedure TfrmMain.srvActionLayoutPrint1Executed(Sender: TObject);
begin
    TsrvToolButton272(SRichViewEdit1.MenuHButtons55.Items271[2]).Down273 :=
        True;
end;

```

## 2.4.2 TsrvActionCustomZoom

TsrvActionCustomZoom is a base class for actions zooming TsrRichViewEdit<sup>40</sup> control.

**Unit** SRVActions;

### Syntax

```
TsrvActionCustomZoom = class (TsrvAction284)
```

### ▼ Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction
TsrvAction284

```

### Description

This action is not used directly.

The following actions are inherited from it:

- TsrvActionZoom<sup>313</sup>
- TsrvActionZoomPageWidth<sup>314</sup>
- TsrvActionZoomFullPage<sup>314</sup>

## 2.4.3 TsrvActionLayoutDraft

TsrvActionLayoutDraft switches TsrRichViewEdit<sup>40</sup> control to a "draft" layout.

**Unit** SRVActions;

### Syntax

```
TsrvActionLayoutDraft = class (TsrvActionCustomLayout296)
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>284</sup>  
*TsrvCustomActionThatNeedsDocProps*<sup>285</sup>  
*TsrvActionCustomLayout*<sup>296</sup>

### Description

In this mode, a document is shown using the target editor's PageProperty<sup>58</sup>.PageWidth<sup>141</sup>, but as a long single page.

The main changes made by this action for TSRichViewEdit control:

| Property  | Assigned value |
|---|----------------|
| PageProperty <sup>58</sup> .PageViewMode <sup>141</sup> | <i>False</i>   |
| PageProperty <sup>58</sup> .AutoWidth <sup>135</sup>    | <i>False</i>   |
| ReadModeProperty <sup>59</sup> .Active <sup>152</sup>   | <i>False</i>   |

This action also changes the following properties:

| Property  | Assigned value          |
|---|-------------------------|
| Color   | <i>c!White</i>          |
| PageProperty <sup>58</sup> .BorderPen <sup>135</sup> .Style | <i>psClear</i>          |
| PagePosProperty <sup>58</sup> .HPadding <sup>146</sup>      | HPadding <sup>301</sup> |
| BackgroundProperty <sup>48</sup> .Visible <sup>127</sup>    | <i>False</i>            |
| ViewProperty <sup>69</sup> .ShowScrollHint <sup>163</sup>   | <i>True</i>             |
| PagePosProperty <sup>58</sup> .AlignPageH <sup>144</sup>    | <i>srvaphLeft</i>       |
| PagePosProperty <sup>58</sup> .AlignPageV <sup>145</sup>    | <i>srvapvTop</i>        |
| PageProperty <sup>58</sup> .HeaderVisible <sup>137</sup>    | <i>False</i>            |
| PageProperty <sup>58</sup> .FooterVisible <sup>136</sup>    | <i>False</i>            |



|   |  |
|---|--|
| PagePosProperty <sup>58</sup> .HiddenDistances <sup>145</sup> | excludes [ <i>srvppdHPadding</i> ,<br><i>srvppdVPadding</i> ,<br><i>srvppdPageHSpacing</i> ] |
|---|--|

After the layout is changed, TRVAControlPanel.OnViewChanged event occurs.

#### See also:

- TsrvActionLayoutWeb<sup>310</sup>
- TsrvActionLayoutPrint<sup>301</sup>
- TsrvActionLayoutSideToSide<sup>307</sup>
- TsrvActionLayoutRead<sup>304</sup>

## 2.4.3.1 Properties

### In TsrvActionLayoutDraft

- HPadding<sup>301</sup>

### Inherited from TsrvCustomActionThatNeedsDocProps<sup>285</sup>

- ActionSave<sup>286</sup>

### Inherited from TsrvAction<sup>284</sup>

- Control<sup>285</sup>

### Inherited from TrvCustomAction

- Disabled

#### 2.4.3.1.1 TsrvActionLayoutDraft.HPadding

A value to assign to SRichViewEdit.PagePosProperty<sup>58</sup>.HPadding<sup>146</sup>.

**property** HPadding: TRVPixel196Length;

The action assigns this value to PagePosProperty<sup>58</sup>.HPadding<sup>146</sup> of the target editor.

#### Default value

15

## 2.4.4 TsrvActionLayoutPrint

TsrvActionLayoutPrint switches TSVRichViewEdit<sup>40</sup> control to a "print layout".

**Unit** SRVActions;

#### Syntax

```
TsrvActionLayoutPrint = class (TsrvActionCustomLayout296)
```

#### ▼ Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>  
*TsrvCustomActionThatNeedsDocProps* <sup>(285)</sup>  
*TsrvActionCustomLayout* <sup>(296)</sup>

## Description

In this mode, a document is shown using the target editor's page size and orientation (defined in *PageProperty* <sup>(58)</sup>). Pages are arranged in rows and columns (no more than the target editor's *PagePosProperty* <sup>(58)</sup>.*MaxPageColCount* <sup>(147)</sup> pages in a row), scrolled vertically.

The main changes made by this action for *TSRichViewEdit* control:

| Property   | Assigned value |
|--|----------------|
| <i>PageProperty</i> <sup>(58)</sup> . <i>PageViewMode</i> <sup>(141)</sup> | <i>True</i>    |
| <i>PageProperty</i> <sup>(58)</sup> . <i>AutoWidth</i> <sup>(135)</sup>    | <i>False</i>   |
| <i>ReadModeProperty</i> <sup>(59)</sup> . <i>Active</i> <sup>(152)</sup>   | <i>False</i>   |

This action also changes the following properties:

| Property   | Assigned value   |
|--|--|
| Color  | Color <sup>(303)</sup>   |
| <i>PageProperty</i> <sup>(58)</sup> . <i>BorderPen</i> <sup>(135)</sup> . <i>Style</i> | <i>psSolid</i>   |
| <i>PagePosProperty</i> <sup>(58)</sup> . <i>HPadding</i> <sup>(146)</sup>              | <i>HPadding</i> <sup>(304)</sup>   |
| <i>BackgroundProperty</i> <sup>(48)</sup> . <i>Visible</i> <sup>(127)</sup>            | <i>True</i>  |
| <i>ViewProperty</i> <sup>(69)</sup> . <i>ShowScrollHint</i> <sup>(163)</sup>           | <i>True</i>  |
| <i>PagePosProperty</i> <sup>(58)</sup> . <i>AlignPageH</i> <sup>(144)</sup>            | <i>srvaphCenter</i>  |
| <i>PagePosProperty</i> <sup>(58)</sup> . <i>AlignPageV</i> <sup>(145)</sup>            | <i>srvaphCenter</i>  |
| <i>PageProperty</i> <sup>(58)</sup> . <i>HeaderVisible</i> <sup>(137)</sup>            | <i>HeaderVisible</i> <sup>(303)</sup>  |
| <i>PageProperty</i> <sup>(58)</sup> . <i>FooterVisible</i> <sup>(136)</sup>            | <i>FooterVisible</i> <sup>(303)</sup>  |
| <i>PagePosProperty</i> <sup>(58)</sup> . <i>HiddenDistances</i> <sup>(145)</sup>       | excludes [ <i>srvppdHPadding</i> ,<br><i>srvppdVPadding</i> ,<br><i>srvppdPageHSpacing</i> ] |

After the layout is changed, *TRVAControlPanel.OnViewChanged* event occurs.

**See also:**

- TsrvActionLayoutWeb <sup>(310)</sup>
- TsrvActionLayoutDraft <sup>(299)</sup>
- TsrvActionLayoutSideToSide <sup>(307)</sup>
- TsrvActionLayoutRead <sup>(304)</sup>

## 2.4.4.1 Properties

### In TsrvActionLayoutPrint

- Color <sup>(303)</sup>
- FooterVisible <sup>(303)</sup>
- HeaderVisible <sup>(303)</sup>
- HPadding <sup>(304)</sup>

### Inherited from TsrvCustomActionThatNeedsDocProps <sup>(285)</sup>

- ActionSave <sup>(286)</sup>

### Inherited from TsrvAction <sup>(284)</sup>

- Control <sup>(285)</sup>

### Inherited from TrvCustomAction

- Disabled

#### 2.4.4.1.1 TsrvActionLayoutPrint.Color

A value to assign to SRichViewEdit.Color.

**property** Color: TColor;

The action assigns this value to Color property of the target editor.

#### Default value

\$99A8AC

#### 2.4.4.1.2 TsrvActionLayoutPrint.FooterVisible

A value to assign to SRichViewEdit.PageProperty <sup>(58)</sup>.FooterVisible <sup>(136)</sup>.

**property** FooterVisible: Boolean;

The action assigns this value to PageProperty <sup>(58)</sup>.FooterVisible <sup>(136)</sup> of the target editor.

#### Default value

True

#### 2.4.4.1.3 TsrvActionLayoutPrint.HeaderVisible

A value to assign to SRichViewEdit.PageProperty <sup>(58)</sup>.HeaderVisible <sup>(137)</sup>.

**property** HeaderVisible: Boolean;

The action assigns this value to PageProperty <sup>(58)</sup>.HeaderVisible <sup>(137)</sup> of the target editor.

#### Default value

True

#### 2.4.4.1.4 TsrvActionLayoutPrint.HPadding

A value to assign to `SRichViewEdit.PagePosProperty(58).HPadding(146)`.

**property** `HPadding: TRVPixel96Length;`

The action assigns this value to `PagePosProperty(58).HPadding(146)` of the target editor.

##### Default value

15

### 2.4.5 TsrvActionLayoutRead

`TsrvActionLayoutRead` switches `TSRichViewEdit(40)` control to a "read mode" layout.

**Unit** `SRVActions`;

##### Syntax

```
TsrvActionLayoutRead = class (TsrvActionCustomLayout(296))
```

##### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction<sup>(284)</sup>*  
*TsrvCustomActionThatNeedsDocProps<sup>(285)</sup>*  
*TsrvActionCustomLayout<sup>(296)</sup>*

### Description

In this mode width and height of pages are calculated automatically to fit the editor. Pages are shown completely, are arranged in one row, are scrolled horizontally with a special transition effect.

The main changes made by this action for `TSRichViewEdit` control:

| Property  | Assigned value |
|---|----------------|
| <code>PageProperty<sup>(58)</sup>.PageViewMode<sup>(141)</sup></code> | <i>True</i>    |
| <code>PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup></code>    | <i>True</i>    |
| <code>ReadModeProperty<sup>(59)</sup>.Active<sup>(152)</sup></code>   | <i>True</i>    |

This action also changes the following properties:

| Property           | Assigned value                     |
|--------------------|------------------------------------|
| <code>Color</code> | <code>Color<sup>(306)</sup></code> |

|   |   |
|---|---|
| PageProperty <sup>58</sup> .BorderPen <sup>135</sup> .Style   | <i>psSolid</i>  |
| PagePosProperty <sup>58</sup> .HPadding <sup>146</sup>        | HPadding <sup>304</sup>   |
| BackgroundProperty <sup>48</sup> .Visible <sup>127</sup>      | <i>True</i>   |
| ViewProperty <sup>69</sup> .ShowScrollHint <sup>163</sup>     | <i>True</i>   |
| PagePosProperty <sup>58</sup> .AlignPageH <sup>144</sup>      | <i>srvaphCenter</i>   |
| PagePosProperty <sup>58</sup> .AlignPageV <sup>145</sup>      | <i>srvapvCenter</i>   |
| PageProperty <sup>58</sup> .HeaderVisible <sup>137</sup>      | <i>False</i>  |
| PageProperty <sup>58</sup> .FooterVisible <sup>136</sup>      | <i>False</i>  |
| PagePosProperty <sup>58</sup> .HiddenDistances <sup>145</sup> | excludes [srvppdHPadding,<br>srvppdVPadding,<br>srvppdPageHSpacing] |

This action also changes the following properties specific for a read mode:

| Property   | Assigned value                         |
|--|--|
| ReadModeProperty <sup>59</sup> .PageFlipEffect <sup>153</sup>          | PageFlipEffect <sup>306</sup>          |
| ReadModeProperty <sup>59</sup> .PageColor <sup>153</sup>               | PageColor <sup>306</sup>               |
| ReadModeProperty <sup>59</sup> .HidePageBackgroundImage <sup>152</sup> | HidePageBackgroundImage <sup>306</sup> |

After the layout is changed, TRVAControlPanel.OnViewChanged event occurs.

#### See also:

- TsrvActionLayoutDraft<sup>299</sup>
- TsrvActionLayoutWeb<sup>310</sup>
- TsrvActionLayoutPrint<sup>301</sup>
- TsrvActionLayoutSideToSide<sup>307</sup>

## 2.4.5.1 Properties

### In TsrvActionLayoutRead

- Color<sup>306</sup>
- HidePageBackgroundImage<sup>306</sup>
- HPadding<sup>306</sup>
- PageColor<sup>306</sup>
- PageFlipEffect<sup>306</sup>

### Inherited from TsrvCustomActionThatNeedsDocProps<sup>285</sup>

- ActionSave<sup>286</sup>

## Inherited from TsrvAction <sup>(284)</sup>

■ Control <sup>(285)</sup>

## Inherited from TrvCustomAction

■ Disabled

### 2.4.5.1.1 TsrvActionLayoutRead.Color

A value to assign to SRichViewEdit.Color.

**property** Color: TColor;

The action assigns this value to Color property of the target editor.

#### Default value

\$99A8AC

### 2.4.5.1.2 TsrvActionLayoutRead.HidePageBackgroundImage

A value to assign to SRichViewEdit.ReadModeProperty <sup>(59)</sup>.HidePageBackgroundImage <sup>(152)</sup>

**property** HidePageBackgroundImage: Boolean;

The action assigns this value to ReadModeProperty <sup>(59)</sup>.HidePageBackgroundImage <sup>(152)</sup> property of the target editor.

#### Default value

False

### 2.4.5.1.3 TsrvActionLayoutRead.HPadding

A value to assign to SRichViewEdit.PagePosProperty <sup>(58)</sup>.HPadding <sup>(146)</sup>.

**property** HPadding: TRVPixel96Length;

The action assigns this value to PagePosProperty <sup>(58)</sup>.HPadding <sup>(146)</sup> of the target editor.

#### Default value

15

### 2.4.5.1.4 TsrvActionLayoutRead.PageColor

A value to assign to SRichViewEdit.ReadModeProperty <sup>(59)</sup>.PageColor <sup>(153)</sup>

**property** PageColor: TColor;

The action assigns this value to ReadModeProperty <sup>(59)</sup>.PageColor <sup>(153)</sup> property of the target editor.

#### Default value

c/None

### 2.4.5.1.5 TsrvActionLayoutRead.PageFlipEffect

A value to assign to SRichViewEdit.ReadModeProperty <sup>(59)</sup>.PageFlipEffect <sup>(153)</sup>.

**property** PageFlipEffect: TSRVPageFlipEffect;

The action assigns this value to ReadModeProperty <sup>(59)</sup>.PageFlipEffect <sup>(153)</sup> of the target editor.

**Default value***srvpfeScroll*

## 2.4.6 TsrvActionLayoutSideToSide

TsrvActionLayoutSideToSide switches TSRichViewEdit<sup>(40)</sup> control to a "side to side" layout.

**Unit** SRVActions;

**Syntax**

```
TsrvActionLayoutSideToSide = class (TsrvActionCustomLayout(296))
```

**▼ Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>  
*TsrvCustomActionThatNeedsDocProps*<sup>(285)</sup>  
*TsrvActionCustomLayout*<sup>(296)</sup>

### Description

In this mode, a document is shown using the target editor's page size and orientation (defined in PageProperty)<sup>(58)</sup>. Pages are shown completely, are arranged in one row, are scrolled horizontally with a special transition effect.

The main changes made by this action for TSRichViewEdit control:

| Property  | Assigned value |
|---|----------------|
| PageProperty <sup>(58)</sup> .PageViewMode <sup>(141)</sup> | <i>True</i>    |
| PageProperty <sup>(58)</sup> .AutoWidth <sup>(135)</sup>    | <i>False</i>   |
| ReadModelProperty <sup>(59)</sup> .Active <sup>(152)</sup>  | <i>True</i>    |

This action also changes the following properties:

| Property  | Assigned value            |
|---|---------------------------|
| Color   | Color <sup>(309)</sup>    |
| PageProperty <sup>(58)</sup> .BorderPen <sup>(135)</sup> .Style | <i>psSolid</i>            |
| PagePosProperty <sup>(58)</sup> .HPadding <sup>(146)</sup>      | HPadding <sup>(309)</sup> |
| BackgroundProperty <sup>(48)</sup> .Visible <sup>(127)</sup>    | <i>True</i>               |

|   |  |
|---|--|
| ViewProperty <sup>69</sup> .ShowScrollHint <sup>163</sup>     | <i>True</i>  |
| PagePosProperty <sup>58</sup> .AlignPageH <sup>144</sup>      | <i>srvaphCenter</i>  |
| PagePosProperty <sup>58</sup> .AlignPageV <sup>145</sup>      | <i>srvapvCenter</i>  |
| PageProperty <sup>58</sup> .HeaderVisible <sup>137</sup>      | HeaderVisible <sup>309</sup>   |
| PageProperty <sup>58</sup> .FooterVisible <sup>136</sup>      | FooterVisible <sup>309</sup>   |
| PagePosProperty <sup>58</sup> .HiddenDistances <sup>145</sup> | excludes [ <i>srvppdHPadding</i> ,<br><i>srvppdVPadding</i> ],<br>excluded/includes<br>[ <i>srvppdPageHSpacing</i> ]<br>depending on<br>NoPageSpacing <sup>309</sup> |

This action also changes the following properties specific for a read mode:

| Property   | Assigned value                |
|--|-------------------------------|
| ReadModeProperty <sup>59</sup> .PageFlipEffect <sup>153</sup>          | PageFlipEffect <sup>310</sup> |
| ReadModeProperty <sup>59</sup> .PageColor <sup>153</sup>               | <i>c/None</i>                 |
| ReadModeProperty <sup>59</sup> .HidePageBackgroundImage <sup>152</sup> | <i>False</i>                  |

After the layout is changed, TRVAControlPanel.OnViewChanged event occurs.

#### See also:

- TsrvActionLayoutDraft<sup>299</sup>
- TsrvActionLayoutWeb<sup>310</sup>
- TsrvActionLayoutPrint<sup>301</sup>
- TsrvActionLayoutRead<sup>304</sup>

## 2.4.6.1 Properties

### In TsrvActionLayoutSideToSide

- Color<sup>309</sup>
- FooterVisible<sup>309</sup>
- HeaderVisible<sup>309</sup>
- HPadding<sup>309</sup>
- NoPageSpacing<sup>309</sup>
- PageFlipEffect<sup>310</sup>

### Inherited from TsrvCustomActionThatNeedsDocProps<sup>285</sup>

- ActionSave<sup>286</sup>



---

## Inherited from TsrvAction <sup>(284)</sup>

---

■ Control <sup>(285)</sup>

## Inherited from TrvCustomAction

---

■ Disabled

### 2.4.6.1.1 TsrvActionLayoutSideToSide.Color

A value to assign to SRichViewEdit.Color.

```
property Color: TColor;
```

The action assigns this value to Color property of the target editor.

#### Default value

\$99A8AC

### 2.4.6.1.2 TsrvActionLayoutSideToSide.FooterVisible

A value to assign to SRichViewEdit.PageProperty <sup>(58)</sup>.FooterVisible <sup>(136)</sup>.

```
property FooterVisible: Boolean;
```

The action assigns this value to PageProperty <sup>(58)</sup>.FooterVisible <sup>(136)</sup> of the target editor.

#### Default value

True

### 2.4.6.1.3 TsrvActionLayoutSideToSide.HeaderVisible

A value to assign to SRichViewEdit.PageProperty <sup>(58)</sup>.HeaderVisible <sup>(137)</sup>.

```
property HeaderVisible: Boolean;
```

The action assigns this value to PageProperty <sup>(58)</sup>.HeaderVisible <sup>(137)</sup> of the target editor.

#### Default value

True

### 2.4.6.1.4 TsrvActionLayoutSideToSide.HPadding

A value to assign to SRichViewEdit.PagePosProperty <sup>(58)</sup>.HPadding <sup>(146)</sup>.

```
property HPadding: TRVPixel96Length;
```

The action assigns this value to PagePosProperty <sup>(58)</sup>.HPadding <sup>(146)</sup> of the target editor.

#### Default value

15

### 2.4.6.1.5 TsrvActionLayoutSideToSide.NoPageSpacing

Turns spacing between pages on/off.

```
property NoPageSpacing: Boolean;
```

Depending on the value of this property, the action excludes/includes *srvppdPageHSpacing* in PagePosProperty <sup>(58)</sup>.HiddenDistances <sup>(145)</sup> of the target editor.

**Default value***False***2.4.6.1.6 TsrvActionLayoutSideToSide.PageFlipEffect**

A value to assign to `SRichViewEdit.ReadModeProperty(59).PageFlipEffect(153)`.

**property** `PageFlipEffect`: `TSRVPageFlipEffect`;

The action assigns this value to `ReadModeProperty(59).PageFlipEffect(153)` of the target editor.

**Default value***srvpfeStack***2.4.7 TsrvActionLayoutWeb**

`TsrvActionLayoutWeb` switches `TSRichViewEdit(40)` control to a "web layout".

**Unit** `SRVActions`;

**Syntax**

```
TsrvActionLayoutWeb = class (TsrvActionCustomLayout(296))
```

**▼ Hierarchy**

`TObject`  
`TPersistent`  
`TComponent`  
`TBasicAction`  
`TContainedAction`  
`TCustomAction`  
`TAction`  
`TrvCustomAction`  
`TsrvAction(284)`  
`TsrvCustomActionThatNeedsDocProps(285)`  
`TsrvActionCustomLayout(296)`

**Description**

In this mode, a document is shown in a single page that occupies all available width of the target editor.

The main changes made by this action for `TSRichViewEdit` control:

| Property  | Assigned value |
|---|----------------|
| <code>PageProperty<sup>(58)</sup>.PageViewMode<sup>(141)</sup></code> | <i>False</i>   |
| <code>PageProperty<sup>(58)</sup>.AutoWidth<sup>(135)</sup></code>    | <i>True</i>    |
| <code>ReadModeProperty<sup>(59)</sup>.Active<sup>(152)</sup></code>   | <i>False</i>   |

This action also changes the following properties:

| Property  | Assigned value   |
|---|--|
| Color   | <i>clWhite</i>   |
| PageProperty <sup>(58)</sup> .BorderPen <sup>(135)</sup> .Style   | <i>psClear</i>   |
| BackgroundProperty <sup>(48)</sup> .Visible <sup>(127)</sup>      | <i>False</i>   |
| ViewProperty <sup>(69)</sup> .ShowScrollHint <sup>(163)</sup>     | <i>False</i>   |
| PagePosProperty <sup>(58)</sup> .AlignPageH <sup>(144)</sup>      | <i>srvaphLeft</i>  |
| PagePosProperty <sup>(58)</sup> .AlignPageV <sup>(145)</sup>      | <i>srvapvTop</i>   |
| PageProperty <sup>(58)</sup> .HeaderVisible <sup>(137)</sup>      | <i>False</i>   |
| PageProperty <sup>(58)</sup> .FooterVisible <sup>(136)</sup>      | <i>False</i>   |
| PagePosProperty <sup>(58)</sup> .HiddenDistances <sup>(145)</sup> | includes [ <i>srvppdHPadding</i> ,<br><i>srvppdVPadding</i> ], excludes<br>[ <i>srvppdPageHSpacing</i> ] |

After the layout is changed, TRVAControlPanel.OnViewChanged event occurs.

#### See also:

- TsrvActionLayoutPrint<sup>(301)</sup>
- TsrvActionLayoutDraft<sup>(299)</sup>
- TsrvActionLayoutSideToSide<sup>(307)</sup>
- TsrvActionLayoutRead<sup>(304)</sup>

## 2.4.8 TsrvActionThumbnails

TsrvActionThumbnails shows/hides page thumbnails for the editor.

**Unit** SRVActions;

#### Syntax

```
TsrvActionThumbnails = class (TsrvAction(284))
```

#### ▼ Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction
TsrvAction(284)

```

## Description

This action links/unlinks PageScroll<sup>(312)</sup> to TSRichViewEdit<sup>(40)</sup> control.

It assigns PageScroll<sup>(312)</sup>.SRichViewEdit<sup>(183)</sup> property, then calls OnExecuted<sup>(312)</sup>. In this event, you can perform additional operations, for example showing/hiding PageScroll<sup>(312)</sup>.

### 2.4.8.1 Properties

#### In TsrvActionThumbnails

■ PageScroll<sup>(312)</sup>

#### Inherited from TsrvAction<sup>(284)</sup>

■ Control<sup>(285)</sup>

#### Inherited from TrvCustomAction

■ Disabled

#### 2.4.8.1.1 TsrvActionThumbnails.PageScroll

A link to TSRVPageScroll<sup>(174)</sup> component.

```
property PageScroll: TSRVPageScroll(174);
```

### 2.4.8.2 Events

#### In TsrvActionThumbnails

■ OnExecuted<sup>(312)</sup>

#### 2.4.8.2.1 TsrvActionThumbnails.OnExecuted

Occurs after the action links/unlinks PageScroll<sup>(312)</sup> to TSRichViewEdit<sup>(40)</sup> control.

```
property OnExecuted: TNotifyEvent;
```

#### Example:

```
procedure TForm3.srvActionThumbnails1Executed(Sender: TObject);  
begin  
    Splitter1.Visible := TsrvActionThumbnails(311)(Sender).Checked;  
    SRVPageScroll11.Visible := TsrvActionThumbnails(311)(Sender).Checked;  
end;
```

## 2.4.9 TsrvActionZoom

TsrvActionZoom zooms TSVRichViewEdit<sup>(40)</sup> control to the specified percent.

**Unit** SRVActions;

### Syntax

```
TsrvActionZoom = class (TsrvActionCustomZoom(299))
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>  
*TsrvActionCustomZoom*<sup>(299)</sup>

### Description

This action assigns the following properties of the target editor:

- ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> := ZoomPercent<sup>(313)</sup>;
- ViewProperty<sup>(69)</sup>.ZoomMode<sup>(166)</sup> := *rvzmCustom*.

### See also:

- TsrvActionZoomPageWidth<sup>(314)</sup>
- TsrvActionZoomFullPage<sup>(314)</sup>

### 2.4.9.1 Properties

#### Inherited from TsrvActionCustomZoom<sup>(299)</sup>

- ZoomPercent<sup>(313)</sup>

#### Inherited from TsrvAction<sup>(284)</sup>

- Control<sup>(285)</sup>

#### Inherited from TrvCustomAction

- Disabled

#### 2.4.9.1.1 TsrvActionZoom.ZoomPercent

Zooming percent.

**property** ZoomPercent: Single;

This value is assigned to ViewProperty<sup>(69)</sup>.ZoomPercent<sup>(168)</sup> of the target editor.

When you change value of this property, Caption and Hint are updated.

**Default value:**

100

## 2.4.10 TsrvActionZoomFullPage

TsrvActionZoomFullPage zooms TSVRichViewEdit<sup>(40)</sup> control to fit the full page in the editor.

**Unit** SRVActions;

**Syntax**

```
TsrvActionZoomFullPage = class (TsrvActionCustomZoom(299))
```

▼ **Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>  
*TsrvActionCustomZoom*<sup>(299)</sup>

## Description

This action assigns the following property of the target editor:

- ViewProperty<sup>(69)</sup>.ZoomMode<sup>(166)</sup> := *rvzmFullPage*

**See also:**

- TsrvActionZoom<sup>(313)</sup>
- TsrvActionZoomPageWidth<sup>(314)</sup>

## 2.4.11 TsrvActionZoomPageWidth

TsrvActionZoomPageWidth zooms TSVRichViewEdit<sup>(40)</sup> control to fit the page width in the editor width.

**Unit** SRVActions;

**Syntax**

```
TsrvActionZoomPageWidth = class (TsrvActionCustomZoom(299))
```

▼ **Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*

*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>284</sup>  
*TsrvActionCustomZoom* <sup>299</sup>

## Description

This action assigns the following property of the target editor:




- `ViewProperty` <sup>69</sup>.`ZoomMode` <sup>166</sup> := *rvzmPageWidth*.

**See also:**

- *TsrvActionZoom* <sup>313</sup>
- *TsrvActionZoomFullPage* <sup>314</sup>

## 2.5 Editing

**Header and footer:**

-  *TsrvActionEditHeader* <sup>315</sup> starts editing a page header.
-  *TsrvActionEditFooter* <sup>315</sup> starts editing a page footer.
-  *TsrvActionEditMain* <sup>316</sup> closes a header/footer editor and returns the main document.

### 2.5.1 TsrvActionEditHeader

Starts editing a page header.

**Unit** SRVActions;

**Syntax**

```
TsrvActionEditHeader = class (TsrvAction 284)
```

▼ **Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>284</sup>

## Description

This action calls `StartEditing` <sup>99</sup> (*srvrveHeader*)

### 2.5.2 TsrvActionEditFooter

Starts editing a page footer.

**Unit** SRVActions;

**Syntax**

```
TsrvActionEditFooter = class (TsrvAction284)
```

**▼ Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>284</sup>

**Description**

This action calls StartEditing<sup>99</sup> (*srvveFooter*)

**2.5.3 TsrvActionEditMain**

Closes header/footer editors and starts editing the main document.

**Unit** SRVActions;

**Syntax**

```
TsrvActionEditMain = class (TsrvAction284)
```





**▼ Hierarchy**

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>284</sup>

**Description**

This action calls StartEditing<sup>99</sup> (*srvveMain*)

**2.6 Notes****Notes and text boxes:**

-  TsrvActionInsertFootnote<sup>320</sup> inserts a footnote.
-  TsrvActionInsertEndnote<sup>319</sup> inserts an endnote.
-  TsrvActionInsertSidenote<sup>322</sup> inserts a sidenote (a note in a floating box).
-  TsrvActionInsertTextBox<sup>324</sup> inserts a text box item.



-  `TsrvActionEditNote`<sup>(319)</sup> starts editing a note or a text box.
-  `TsrvActionReturnToNote`<sup>(325)</sup> moves the caret to the parent note.

## 2.6.1 TsrvActionCustomInsertSidenote

`TsrvActionCustomInsertSidenote` is the base class for actions for inserting sidenotes and text box items.

**Unit** `SRVActions`;

### Syntax

```
TsrvActionCustomInsertSidenote = class(TsrvActionInsertNote(321))
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>  
*TsrvActionInsertNote*<sup>(321)</sup>

### Description

This action introduces properties that are assigned to properties of an inserted sidenote (or a text box item):

- `BoxProperties`<sup>(318)</sup>
- `BoxPosition`<sup>(318)</sup>

Text in the note's Document is formatted either according to `StyleTemplateName`<sup>(318)</sup> (if style templates are used) or according to `Font`<sup>(322)</sup> (otherwise).

This class is not used directly. The following actions are inherited from `TrvActionCustomInsertSidenote`:

- `TrvActionInsertSidenote`<sup>(322)</sup>
- `TrvActionInsertTextBox`<sup>(324)</sup>

### 2.6.1.1 Properties

#### In `TsrvActionCustomInsertSidenote`

- `BoxPosition`<sup>(318)</sup>
- `BoxProperties`<sup>(318)</sup>
- `StyleTemplateName`<sup>(318)</sup>

#### Inherited from `TsrvActionInsertNote`<sup>(321)</sup>

- `Font`<sup>(322)</sup>

## Inherited from TsrvAction <sup>(284)</sup>

■ Control <sup>(285)</sup>

## Inherited from TrvCustomAction

■ Disabled

### 2.6.1.1.1 TsrvActionCustomInsertSidenote.BoxPosition

Specifies position properties for the floating box.

**property** BoxPosition: TRVABoxPosition;

This property is assigned to BoxPosition property of the inserted sidenote / text box item.

Default values are different in inherited actions.

**TsrvActionInsertSidenote** <sup>(322)</sup>:

- HorizontalAnchor=rvhanMainTextArea;
- HorizontalPositionKind=rvfpkAlignment;
- HorizontalAlignment=rvfphalRight;
- VerticalAnchor=rvvanParagraph;
- VerticalPositionKind=rvfpkAbsPosition;
- VerticalOffset=0.

**TsrvActionInsertTextBox** <sup>(324)</sup>:

- HorizontalAnchor=rvhanMainTextArea;
- HorizontalPositionKind=rvfpkAlignment;
- HorizontalAlignment=rvfphalCenter;
- VerticalAnchor=rvvanLine;
- VerticalPositionKind=rvfpkAbsPosition;
- VerticalOffset=20.

### 2.6.1.1.2 TsrvActionCustomInsertSidenote.BoxProperties

Specifies size, background, border, vertical alignment properties for the floating box.

**property** BoxProperties: TRVABoxProperties;

This property is assigned to BoxProperties property of the inserted sidenote / text box item.

#### Default values

- WidthType=rvbwtRelPage;
- Width=20000;

### 2.6.1.1.3 TsrvActionCustomInsertSidenote.StyleTemplateName

Specifies the name of a style template that will be applied to a paragraph of a sidenote's Document.

**property** StyleTemplateName: TRVStyleTemplateName;

This property is used only if the target editor's UseStyleTemplates <sup>(69)</sup>=True. Otherwise, the 0-th paragraph style and Font <sup>(322)</sup> property are used.

Default values are different in inherited actions:

**TsrvActionInsertSidenote**<sup>(322)</sup>:

'Sidenote Text' (note: this style template is standard for RichViewActions, but it is not standard for Microsoft Word)

**TsrvActionInsertTextBox**<sup>(324)</sup>:

'Normal'

## 2.6.2 TsrvActionEditNote

TsrvActionEditNote starts editing the current or the next footnote/endnote/sidenote/text box item.

**Unit** SRVActions;

### Syntax

```
TsrvActionEditNote = class (TsrvAction(284))
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>

### Description

If the current item in TSRichViewEdit.RichViewEdit<sup>(60)</sup> is a note, this action starts editing it.

Otherwise, the action searches for the next note from the caret position. If found, this action starts editing it.

Use this action instead of TrvActionEditNote.

## 2.6.3 TsrvActionInsertEndnote

TsrvActionInsertEndnote inserts an endnote in the caret position.

**Unit** SRVActions;

### Syntax

```
TsrvActionInsertEndnote = class (TsrvActionInsertNote(321))
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*

*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>  
*TsrvActionInsertNote* <sup>(321)</sup>

## Description

If `TSRichViewEdit.RichViewEdit` <sup>(60)</sup> is active, this action inserts an endnote (`TRVEndnoteItemInfo` item).

If `TSRichViewEdit.RVNote` <sup>(64)</sup> is active (editing a `endnote.Document`), this action inserts a reference to the parent endnote (`TRVNoteReferenceItemInfo` item).

Otherwise, this action is disabled.

When inserting an endnote, this action adds in its `Document` the following items:

- reference to this endnote;
- space character.

Attributes of inserted items depend on the target editor's `UseStyleTemplates` <sup>(69)</sup> property.

If style templates are used, the endnote and the endnote reference characters are formatted using "endnote reference" style template. Otherwise, or if this style template does not exist, they are formatted as superscript text. In the both cases, *rvprDoNotAutoSwitch* is included in its `Protection`.

If style templates are used, endnote paragraph and text are formatted using "endnote text" style template. Otherwise, or if this style template does not exist, endnote text is formatted using `Font` <sup>(322)</sup> property.

**Note:** "endnote reference" and "endnote text" style templates are added by default to `TrvActionNew.StyleTemplates`, so they are included in new documents.

Use this action instead of `TrvActionInsertEndnote`.

### 2.6.4 TsrvActionInsertFootnote

`TsrvActionInsertFootnote` inserts a footnote in the caret position.

**Unit** `SRVActions`;

#### Syntax

```
TsrvActionInsertFootnote = class (TsrvActionInsertNote (321))
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>

*TsrvActionInsertNote*<sup>(321)</sup>

## Description

If *TSRichViewEdit.RichViewEdit*<sup>(60)</sup> is active, this action inserts a footnote (*TRVFootnoteItemInfo* item).

If *TSRichViewEdit.RVNote*<sup>(64)</sup> is active (editing a *footnote.Document*), this action inserts a reference to the parent footnote (*TRVNoteReferenceItemInfo* item).

Otherwise, this action is disabled.

When inserting a footnote, this action adds in its *Document* the following items:

- reference to this footnote;
- space character.

Attributes of inserted items depend on the target editor's *UseStyleTemplates*<sup>(69)</sup> property.

If style templates are used, the footnote and the footnote reference characters are formatted using "footnote reference" style template. Otherwise, or if this style template does not exist, they are formatted as superscript text. In the both cases, *rvprDoNotAutoSwitch* is included in its *Protection*.

If style templates are used, footnote paragraph and text are formatted using "footnote text" style template. Otherwise, or if this style template does not exist, footnote text is formatted using *Font*<sup>(322)</sup> property.

**Note:** "footnote reference" and "footnote text" style templates are added by default to *TrvActionNew.StyleTemplates*, so they are included in new documents.

Use this action instead of *TrvActionInsertFootnote*.

## 2.6.5 TsrvActionInsertNote

*TsrvActionInsertNote* is the base class for actions for inserting footnotes, endnotes, sidenotes and text box items.

**Unit** SRVActions;

### Syntax

```
TsrvActionInsertNote = class (TsrvAction(284))
```

### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>

## Description

This action is not used directly.

The following actions are inherited from it:

- TsrvActionInsertFootnote <sup>(320)</sup>
- TsrvActionInsertEndnote <sup>(319)</sup>
- TsrvActionInsertSidenote <sup>(322)</sup>
- TsrvActionInsertTextBox <sup>(324)</sup>

### 2.6.5.1 Properties

#### In TsrvActionInsertNote

■ Font <sup>(322)</sup>

#### Inherited from TsrvAction <sup>(284)</sup>

■ Control <sup>(285)</sup>

#### Inherited from TrvCustomAction

■ Disabled

#### 2.6.5.1.1 TsrvActionInsertNote.Font

Default font for text in a note's document.

##### type

```
TSRVFont = class (TFont)
```

**property** Font: TSRVFont;

This value is used if UseStyleTemplates <sup>(69)</sup> = *False* for the target editor, or the required style template does not exist in Editor.RichViewEdit <sup>(60)</sup>.Style.StyleTemplates. Otherwise, note text is formatted according to a style template.

Style template names:

- "footnote text" for footnotes,
- "endnote text" for endnotes,
- StyleTemplateName for sidenotes and text boxes (default values: "Sidenote Text" for sidenotes, "Normal" for text boxes)

##### Default value:

'Arial', 10

### 2.6.6 TsrvActionInsertSidenote

TsrvActionInsertSidenote inserts a sidenote (a note in a floating box) in the caret position.

**Unit** SRVActions;

##### Syntax

```
TsrvActionInsertSidenote = class (TsrvActionCustomInsertSidenote (317))
```

##### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction* <sup>(284)</sup>  
*TsrvActionInsertNote* <sup>(321)</sup>  
*TsrvActionCustomInsertSidenote* <sup>(317)</sup>

## Description

If *TSRichViewEdit.RichViewEdit* <sup>(60)</sup>, *RVHeader* <sup>(64)</sup> or *RVFooter* <sup>(62)</sup> is active, this action inserts a sidenote (TRVSidenoteItemInfo item).

If *TSRichViewEdit.RVNote* <sup>(64)</sup> is active (editing a sidenote.Document), this action inserts a reference to the parent sidenote (TRVNoteReferenceItemInfo item).

Otherwise, this action is disabled.

When inserting a sidenote, this action adds in its Document the following items:

- reference to this sidenote;
- space character.

Attributes of inserted items depend on the target editor's *UseStyleTemplates* <sup>(69)</sup> property.

If style templates are used, the sidenote and the sidenote reference characters are formatted using *RefStyleTemplateName* <sup>(324)</sup> style template. Otherwise, or if this style template does not exist, they are formatted as superscript text. In the both cases, *rvprDoNotAutoSwitch* is included in its Protection.

If style templates are used, sidenote paragraph and text are formatted using *StyleTemplateName* <sup>(318)</sup> style template. Otherwise, or if this style template does not exist, sidenote text is formatted using *Font* <sup>(322)</sup> property.

The following properties are assigned to the inserted sidenote:

- *BoxProperties* <sup>(318)</sup> (default: width: 20% of page width, height: auto)
- *BoxPosition* <sup>(318)</sup> (default: horizontal - right aligned to the main text area, vertical - aligned to the top of the paragraph)

Use this action instead of *TrvActionInsertSidenote*.

### 2.6.6.1 Properties

#### In *TsrvActionInsertSidenote*

- *RefStyleTemplateName* <sup>(324)</sup>

#### Inherited from *TsrvActionCustomInsertSidenote* <sup>(317)</sup>

- *BoxPosition* <sup>(318)</sup>
- *BoxProperties* <sup>(318)</sup>

■ StyleTemplateName<sup>(318)</sup>

## Inherited from TsrvActionInsertNote<sup>(321)</sup>

■ Font<sup>(322)</sup>

## Inherited from TsrvAction<sup>(284)</sup>

■ Control<sup>(285)</sup>

## Inherited from TrvCustomAction

■ Disabled

### 2.6.6.1.1 TsrvActionInsertSidenote.RefStyleTemplateName

Specifies the name of a style template that will be applied to the sidenote character and to the reference of to the parent sidenote (inserted in the sidenote.Document)

**property** RefStyleTemplateName: TRVStyleTemplateName;

This property is used only if the target editor's UseStyleTemplates<sup>(69)</sup>=True. Otherwise, these characters are formatted as a superscript.

#### Default value

'Sidenote Reference' (note: this style template is standard for RichViewActions, but it is not standard for Microsoft Word)

## 2.6.7 TsrvActionInsertTextBox

TsrvActionInsertTextBox inserts a text box item in the caret position.

**Unit** SRVActions;

#### Syntax

TsrvActionInsertTextBox = **class** (TsrvActionCustomInsertSidenote<sup>(317)</sup>)

#### ▼ Hierarchy

TObject  
 TPersistent  
 TComponent  
 TBasicAction  
 TContainedAction  
 TCustomAction  
 TAction  
 TrvCustomAction  
 TsrvAction<sup>(284)</sup>  
 TsrvActionInsertNote<sup>(321)</sup>  
 TsrvActionCustomInsertSidenote<sup>(317)</sup>

## Description

If TSRichViewEdit.RichViewEdit<sup>(60)</sup>, RVHeader<sup>(64)</sup> or RVFooter<sup>(62)</sup> is active, this action inserts a text box item (TRVTextBoxItemInfo item).



Otherwise, this action is disabled.

When inserting a text box, this action adds a single empty text item in its Document .

Attributes of this text item depend on the target editor's UseStyleTemplates<sup>(69)</sup> property.

If style templates are used, this text item's paragraph and text are formatted using StyleTemplateName<sup>(318)</sup> style template. Otherwise, or if this style template does not exist, this text is formatted using Font<sup>(322)</sup> property.

The following properties are assigned to the inserted sidenote:

- BoxProperties<sup>(318)</sup> (default: width: 20% of page width, height: auto)
- BoxPosition<sup>(318)</sup> (default: horizontal - centered in the main text area, vertical - 20 pixels below the top of the line)

Use this action instead of TrvActionInsertSidenote.

## 2.6.8 TsrvActionReturnToNote

TsrvActionReturnToNote ends editing a footnote/endnote/sidenote/text box.

**Unit** SRVActions;

### Syntax

```
TsrvActionReturnToNote = class (TsrvAction(284))
```

#### ▼ Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TBasicAction*  
*TContainedAction*  
*TCustomAction*  
*TAction*  
*TrvCustomAction*  
*TsrvAction*<sup>(284)</sup>

### Description

This action is enabled only if a note or a text box is being edited in TSRichViewEdit control.

The action activates the main document editor (RichViewEdit<sup>(60)</sup>) and moves the caret to the note.


## 3 SRVControls

SRVControls is a set of controls designed for insertion in TSRichViewEdit<sup>(40)</sup>.


### Controls:


 TSrvButton<sup>(343)</sup> – button;

 TSrvCheckBox<sup>(345)</sup> – check box;

 TSrvComboBox<sup>(346)</sup> – combo box; hierarchical; with images, custom colors and fonts; with hints


(suggestions);

 TSRVEdit<sup>(349)</sup> – plain-text one-line editor with hints (suggestions);


 TSRVGroupBox<sup>(352)</sup> – panel with a title;

 TSRVImagesScroll<sup>(354)</sup> – gallery of images;

 TSRVLabel<sup>(358)</sup> – text label;

 TSRVListBox<sup>(359)</sup> – scrollable list of items; hierarchical; with images, check boxes, custom colors and fonts;

 TSRVMemo<sup>(362)</sup> – plain-text multi-line editor;

 TSRVPaintBox<sup>(364)</sup> – controls for custom drawing;

 TSRVPanel<sup>(365)</sup> – panel;

 TSVRRadioButton<sup>(367)</sup> – radio button.


**See also:**


 TSVRScrollBar<sup>(243)</sup>;


 TSVRTabSet<sup>(249)</sup>.


All the controls above can be skinned using TSVRSkinManager<sup>(228)</sup>.

### Data-aware controls:


 TSRVDBCheckBox<sup>(369)</sup> – data-aware check box;

 TSRVDBComboBox<sup>(346)</sup> – data-aware combo box; hierarchical; with images, custom colors and fonts; with hints (suggestions);

 TSRVDBEdit<sup>(371)</sup> – data-aware plain-text one-line editor with hints (suggestions);

 TSRVDBListBox<sup>(372)</sup> – data-aware scrollable list of items; hierarchical; with images, check boxes, custom colors and fonts;

 TSRVDBMemo<sup>(373)</sup> – data-aware plain-text multi-line editor;

 TSRVDBText<sup>(374)</sup> – data-aware text label.

### How to use

SRVControls may work both on a form (like any other Delphi controls) or inserted in TSVRichViewEdit<sup>(40)</sup> / TDBSRichViewEdit<sup>(169)</sup>.

While you can insert any control in TSVRichViewEdit, SRVControls work better in this mode:

- they provide high-quality scaling when displaying in the editor;
- they provide high-quality printing;
- they are scaled properly if Screen.PixelsPerInch <> 96 (but only in Delphi 10.1 and newer).

If you create SRVControls at run-time in code, it's highly recommended (instead of a direct assigning of its Parent) to use RVSetControlParent function (defined in RVFuncs unit). This recommendation is in effect both to controls that will be added to a form, and to controls that will be inserted in the editor (in the latter case, RVSetControlParent should be called before calling InsertItem or AddItem methods):

```
c := TSVRCheckBox(345).Create(nil);
RVSetControlParent(c, SRichViewEdit1.ActiveEditor(47));
c.Caption := 'Default';
SRichViewEdit1.ActiveEditor(47).InsertControl(' ', c, rvvaMiddle);
```

## GDI+ notes

SRVControls can use either standard Windows drawing functions (GDI), or advanced GDI+ drawing functions. GDI+ functions can be used if their SRVControlStyle = *srvcSimple*<sup>(276)</sup>.

GDI+ functions provide high-quality anti-aliased drawing, and SRVControls looks much better in this mode.

For Delphi: GDI+ is used automatically in Delphi XE2 or newer. It cannot be used on older versions of Delphi.

For Lazarus: GDI+ can be used optionally. To use it:

1. Download **GDI+ Library for Delphi and Lazarus** and compile lazgdiplus.lpk. Alternatively, you can install it using Lazarus Online package manager (menu "Package | Online Package Manager")
2. Compile <TRichView Dir>\TRichView\Source\rvgdipluslaz.lpk (Note: it is included in <TRichView Dir>\TRichViewLazarus\_Optional.lpg project group)
3. Include a reference to rvgdipluslaz package in your project (Menu "Project | Project Inspector" to open Project Inspector window. Right click "Required Packages", select menu "Add...". Select RVGDIPlusLaz)
4. Include the unit RVGDIPlusGrInLaz in "uses" of the main form unit.

Note: there is an alternative GDI+ support package for Lazarus (rvpgdipluslaz.lpk, containing RVPGDIPlusGrInLaz unit). It uses GDI+ library by Progdigy. It is not available now at its original location, but **can be downloaded from our website**. Still, I recommend using the package above.

## 3.1 Ancestor classes

### Parent Classes for SRVControls:

- TCustomSRVEdit<sup>(327)</sup>
- TCustomSRVGraphicControl<sup>(329)</sup>
- TCustomSRVPanel<sup>(330)</sup>
- TSVRCanvasControl<sup>(331)</sup>
- TSVRCustomControl<sup>(335)</sup>
- TSVRGraphicControl<sup>(339)</sup>
- TSVRWinControl<sup>(340)</sup>
- TSVRCustomComboBox<sup>(332)</sup>
- TSVRCustomListBox<sup>(336)</sup>
- TSVREditControl<sup>(339)</sup>

### 3.1.1 TCustomSRVEdit

TSVREditControl is the base class for plain-text edit controls.

**Unit** SRVCustomEdit;

## Syntax

```
TCustomSRVEdit = class(TSRVEditControl339);
```

## Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*


*TWinControl*

*TSRVWinControl*<sup>340</sup>

*TSRVEditControl*<sup>339</sup>

## Properties and events

### Public properties

| Property   | Type             | Default value        | Meaning   |
|--|------------------|----------------------|---|
| <b>Alignment</b>   | TAlignment       | <i>taLeftJustify</i> | Horizontal text alignment (left/right/center). RTL BiDiMode inverts left and right alignments.                                    |
|  <b>CanUndo</b> | Boolean          |                      | Read-only. Indicates whether the edit control contains changes that can be backed out.  |
| <b>Modified</b>  | Boolean          |                      | Indicates whether the user edited the text of the edit control.   |
| <b>ReadOnly</b>  | Boolean          | <i>False</i>         | Determines whether the user can change the text of the edit control.  |
| <b>SelText</b>   | TRVUnicodeString |                      | Specifies the selected portion of the edit control's text.  |
| <b>Transparent</b>   | Boolean          | <i>False</i>         | Indicates whether the background is transparent. This property works only for editors inserted in TSVRichViewEdit <sup>40</sup> . |

See also properties inherited from TSRVWinControl<sup>340</sup>.

## Methods

| Method                 | Description  |
|------------------------|--|
| <b>Clear</b>           | Deletes all text from the edit control.  |
| <b>ClearSelection</b>  | Removes the selected text from the edit control.   |
| <b>ClearUndo</b>       | Clears the undo buffer so that no changes to the text can be backed out.                           |
| <b>CopyToClipboard</b> | Copies the selected text in the edit control to the Clipboard in CF_UNICODETEXT format.            |
| <b>CutToClipboard</b>  | Copies the selected text to the Clipboard in CF_UNICODETEXT format and then deletes the selection. |

| Method                    | Description  |
|---------------------------|--|
| <b>PasteFromClipboard</b> | Pastes the contents of the Clipboard into edit control, replacing the current selection. |
| <b>Undo</b>               | Backs out all changes in the undo buffer.  |

See also methods inherited from [TSRVWinControl](#) <sup>340</sup>.

## Inherited components



TSRVEdit <sup>349</sup>



TSRVMemo <sup>362</sup>

### 3.1.2 TCustomSRVGraphicControl

TSRVGraphicControl is the base class for lightweight SRVControls.

**Unit** SRVControl;

#### Syntax

```
TCustomSRVGraphicControl = class (TGraphicControl);
```

#### Hierarchy

*TObject*

*TPersistent*


*TComponent*


*TControl*

*TGraphicControl*

#### Properties and events

**Public properties:**




| Property   | Type    | Default value | Meaning   |
|--|---------|---------------|---|
| <b>AutoDarkMode</b>  | Boolean | <i>True</i>   | If <i>True</i> , and the control is inserted in a dark-mode editor (TSRichViewEdit or TRichView), luminance of all colors is inverted. Otherwise, <b>DarkMode</b> property is used. |
| <b>DarkMode</b>  | Boolean | <i>False</i>  | If <i>True</i> , luminance of all colors are inverted. This property is used only if the control is not inserted in an editor, or if <b>AutoDarkMode</b> = <i>False</i> .           |
| <b>DrawOnPrint</b>   | Boolean | <i>True</i>   | if <i>True</i> , this control will be printed   |
|  <b>IsMouseDown</b> | Boolean |               | Read-only. Returns <i>True</i> if the user pressed the left mouse button above the control.   |

| Property   | Type    | Default value | Meaning  |
|--|---------|---------------|--|
|  <b>MouseIn</b> | Boolean |               | Read-only. Returns <i>True</i> , if the mouse pointer is above the control |

#### Public events:

- **OnMouseEnter**, **OnMouseLeave** (TNotifyEvent) occur when the mouse pointer enter/leaves the control.


#### Protected properties:

| Property   | Type                             | Default value        | Meaning   |
|--|----------------------------------|----------------------|---|
|  <b>Alignment</b>       | TAlignment                       | <i>taLeftJustify</i> | Text alignment. RTL BiDiMode inverts left and right alignments  |
|  <b>SkinManager</b>     | TSRVSkinManager <sup>(228)</sup> |                      | A link to a skin manager component.   |
|  <b>SkinSchemeIndex</b> | Integer                          | 0                    | Index in a scheme in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> (a scheme depends on the control type) |

Additionally, for Delphi versions prior to 2009, this control introduces Unicode Caption (of TRVUnicodeString) type to use instead of ANSI string Caption (so Caption is Unicode in all versions of Delphi)

### Inherited components

 **TSRVLabel**<sup>(358)</sup>

 **TSRVPaintBox**<sup>(364)</sup>

### 3.1.3 TCustomSRVPanel

**TCustomSRVPanel** is the base class for **TSRVPanel**<sup>(365)</sup> and **TSRVGroupBox**<sup>(362)</sup>.

**Unit** SRVPanel;

#### Syntax

```
TCustomSRVPanel = class (TSRVCustomControl)
```

#### Hierarchy

TObject

TPersistent

TComponent

TControl



TWinControl

TCustomControl



TSRVCustomControl<sup>(335)</sup>

## Properties and events

### Protected properties:

| Property   | Type             | Default value | Meaning         |
|--|------------------|---------------|-----------------|
|  <b>BorderWidth</b>   | Integer          | 1             | Border width.   |
|  <b>CaptionOffset</b> | TRVPixel96Length | 10            | Caption offset. |

Protected properties used only if **SRVControlStyle** = *srvcClassic*:

| Property   | Type    | Default value | Meaning   |
|--|---------|---------------|---|
|  <b>BorderColor</b>   | TColor  | \$00B8D8D7    | Border color  |
|  <b>CornersOffset</b> | Integer | 10            | Value in range 1..30. Defines how much border corners are rounded |

### Inherited components



TSRVGroupBox <sup>352</sup>



TSRVPanel <sup>365</sup>

### 3.1.4 TSRVCanvasControl

TSRVCanvasControl is the base class for lightweight SRVControls.

**Unit** SRVControl;

#### Syntax

```
TSRVCanvasControl = class (TGraphicControl);
```

### Hierarchy

TObject

TPersistent

TComponent

TControl

TGraphicControl

TCustomSRVGraphicControl <sup>329</sup>

### Properties and events

#### Protected events:

- **OnPaint** (TSRVCtrlPaintPage) occurs when the control must be painted.

#### type

```
TSRVCtrlPaintPage = procedure (Sender: TObject; R: TRect; Canvas: TCanvas) of
```

See also properties and events inherited from TCustomSRVGraphicControl <sup>329</sup>.

### Inherited components



TSRVPaintBox <sup>364</sup>

### 3.1.5 TSRVCustomComboBox

TSRVCustomComboBox is the base class for combo box controls.

**Unit** SRVComboBox;

#### Syntax

```
TSRVCustomComboBox = class (TSRVEditControl339);
```

#### Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl*<sup>340</sup>


*TSRVEditControl*<sup>339</sup>

#### Properties and events

##### Public properties

| Property             | Type             | Default value      | Meaning  |
|----------------------|------------------|--------------------|--|
| <b>AutoComplete</b>  | Boolean          | <i>True</i>        | Specifies whether the combo box automatically completes words that the user types by selecting the first item that begins with the currently typed string. |
| <b>AutoDropDown</b>  | Boolean          | <i>False</i>       | Specifies whether the drop-down list drops down automatically in response to user keystrokes.  |
| <b>AutoSize</b>      | Boolean          | <i>True</i>        | Determines whether the height of the edit region automatically resizes to accommodate the text height  |
| <b>CharCase</b>      | TSRVEditCharCase | <i>srvecNormal</i> | Determines the case of the text within the edit region:<br><i>srvecNormal</i> , <i>srvecUpperCase</i> , <i>srvecLowerCase</i>                              |
| <b>Color</b>         | TColor           | <i>clWindow</i>    | Background color of the editing region.  |
| <b>DropDownCount</b> | Integer          | 5                  | Specifies the maximum number of items displayed in the drop-down list.   |
| <b>DropDownHints</b> | Integer          | 10                 | Defines how many hints (suggestions) are visible in a list.  |
| <b>HideSelection</b> | Boolean          | <i>True</i>        | Determines whether the visual indication of the selected text remains  |



| Property   | Type                                   | Default value     | Meaning  |
|--|--|-------------------|--|
|  |  |                   | when focus shifts to another control.  |
| <b>HintsItemSkinSchemeIndex</b>  | Integer                                |                   | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> collection, for items in the list of hints (suggestions)                    |
| <b>HintsScrollBarSkinSchemeIndex</b>   | Integer                                |                   | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .VerticalScrollBarSchemes <sup>(233)</sup> collection, for scrollbars in the list of hints (suggestions) |
| <b>HintsSkinSchemeIndex</b>  | Integer                                |                   | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> collection, for the list of hints (suggestions)                             |
| <b>ImageList</b>   | TCustomImageList                       |                   | Specifies the images that appear next to items in the list box.  |
| <b>ItemHeight</b>  | Integer                                | 20                | Specifies the height in pixels of an item in an owner-draw list. When the screen DPI is changed, this value is recalculated.   |
| <b>ItemIndex</b>   | Integer                                |                   | Specifies the ordinal number of the selected item.   |
| <b>ItemWidth</b>   | TRVPixel96Length                       | 0                 | If set to a positive value, defines a width of a drop down list.   |
| <b>Items</b>   | TSRVBoxItemCollection <sup>(375)</sup> |                   | Items, a collection of TSRVBoxItem <sup>(375)</sup> .  |
| <b>ListMarginHeight</b>  | TRVPixel96Length                       | 2                 | Top and bottom margins in each item.   |
| <b>ListMarginWidth</b>   | TRVPixel96Length                       | 3                 | Left and right margins of each item.   |
| <b>MaxLength</b>   | Integer                                | 0                 | Specifies the maximum number of characters the user can type into the edit portion (0 for unlimited)   |
| <b>Modified</b>  | Boolean                                |                   | Indicates whether the user edited the text of the edit control.  |
| <b>ReadOnly</b>  | Boolean                                | <i>False</i>      | Determines whether the user can change the text of the edit region.  |
|  <b>ScrollBar</b> | TSRVScrollBar <sup>(243)</sup>         |                   | Read-only. Returns a scroll bar object.  |
| <b>Sorted</b>  | Boolean                                | <i>False</i>      | Specifies whether the items in a list box are arranged alphabetically.   |
| <b>Style</b>   | TComboBoxStyle                         | <i>csDropDown</i> | Determines the display style of the combo box.   |

| Property                           | Type    | Default value | Meaning   |
|------------------------------------|---------|---------------|---|
| <b>Text</b>                        | String  |               | Text in the editing region.   |
| <b>UseItemFontNames</b>            | Boolean | <i>False</i>  | Specifies whether the <b>FontName</b> properties of items are used to display their <b>Captions</b> .   |
| <b>UseSelectedImagesInHotState</b> | Boolean | <i>False</i>  | Specifies whether <code>Items[].SelectedImageIndex</code> is used instead of <code>Items[].ImageIndex</code> for highlighted items (below the mouse pointer). |

Public properties working only if **SRVControlStyle** = *srvcClassic*:

| Property                       | Type   | Default value          | Meaning                                 |
|--------------------------------|--------|------------------------|---|
| <b>SelectedItemBackColor</b>   | TColor | <i>clHighlight</i>     | Background color for the selected item. |
| <b>SelectedItemBorderColor</b> | TColor | <i>clSilver</i>        | Border color for the selected item.     |
| <b>SelectedItemTextColor</b>   | TColor | <i>clHighlightText</i> | Text color for the selected item.       |

### Public events

**OnChange** occurs when the user changes the text displayed in the edit region.

**OnDropDown** occurs when the user opens the drop-down list.

**OnMeasureItem** occurs when the application needs to redisplay an item in a variable height owner-draw list box.

**OnDrawItem** occurs when an item in an owner-draw list box needs to be redisplayed.

**OnEditHints** occurs when a list of hints (suggestions) is about to be displayed. See the section about "Hints (Suggestions)" for `TSRVEdit`<sup>349</sup>.

See also properties and events inherited from `TSRVEditControl`<sup>339</sup>.

### Methods

**BeginUpdate/EndUpdate** locks/unlocks redrawing.

**IndexOf** returns the index of item by its Caption.

**ItemHasParent** returns *True* if the given item has a parent item.

**GetItemParent** returns the index of the parent item for the given item.

**AppendItem** adds an item with the specified properties to the end of the item list.

**InsertItem** inserts an item with the specified properties at the position Index.

**DeleteItem** deletes the specified item.

## Inherited components

 TSRVComboBox <sup>346</sup>

### 3.1.6 TSRVCustomControl

TSRVCustomControl is the base class for several SRVControls.

**Unit** SRVControl;

#### Syntax

```
TSRVCustomControl = class (TCustomControl);
```

#### Hierarchy

TObject

TPersistent

TComponent

TControl


TWinControl

TCustomControl

#### Properties and events

**Public properties:**





| Property             | Type       | Default value        | Meaning  |
|----------------------|------------|----------------------|--|
| <b>Alignment</b>     | TAlignment | <i>taLeftJustify</i> |  |
| <b>AutoDarkMode</b>  | Boolean    | <i>True</i>          | If <i>True</i> , and the control is inserted in a dark-mode editor (TSRichViewEdit or TRichView), luminance of all colors is inverted. Otherwise, <b>DarkMode</b> property is used. Supported only if <b>SRVControlStyle</b> = <i>srvcSimple</i> . |
| ▶ <b>CaretPos</b>    | TPoint     |                      | Read-only. Returns the caret position in the control. If the control does not have a caret, returns (-1, -1).  |
| ▶ <b>CaretHeight</b> | Integer    |                      | Read-only. Returns the caret height in the control. If the control does not have a caret, returns -1.  |
| <b>DrawOnPrint</b>   | Boolean    | <i>True</i>          | if <i>True</i> , this control will be printed  |
| <b>DarkMode</b>      | Boolean    | <i>False</i>         | If <i>True</i> , luminance of all colors are inverted. This property is used only if the control is not inserted in an editor, or if <b>AutoDarkMode</b> = <i>False</i> . Supported only if <b>SRVControlStyle</b> = <i>srvcSimple</i> .           |

| Property   | Type                              | Default value | Meaning  |
|--|-----------------------------------|---------------|--|
|  <b>MouseIn</b> | Boolean                           |               | Read-only. Returns <i>True</i> , if the mouse pointer is above the control |
| <b>SRVControlStyle</b>   | TSRVControlStyle <sup>(276)</sup> | srvcSimple    | Defines visual appearance  |

#### Public events:







- **OnMouseEnter, OnMouseLeave** (TNotifyEvent) occur when the mouse pointer enter/leaves the control.

#### Protected properties:

| Property   | Type                             | Default value | Meaning  |
|--|----------------------------------|---------------|--|
|  <b>ButtonSkinSchemeIndex</b> | Integer                          | 0             | Additional index in a scheme in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> (a scheme depends on the control type) |
|  <b>SkinManager</b>           | TSRVSkinManager <sup>(228)</sup> |               | A link to a skin manager component.  |
|  <b>SkinSchemeIndex</b>     | Integer                          | 0             | Index in a scheme in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> (a scheme depends on the control type)            |
|  <b>SkinFontIndex</b>       | Integer                          | 0             | Index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .Fonts <sup>(232)</sup>                                       |

Additionally, for Delphi versions prior to 2009, this control introduces Unicode Caption (of TRVUnicodeString) type to use instead of ANSI string Caption (so Caption is Unicode in all versions of Delphi)

### Inherited components

-  TSRVButton<sup>(343)</sup>
-  TSRVCheckBox<sup>(345)</sup>
-  TSRVGroupBox<sup>(352)</sup>
-  TSRVPanel<sup>(365)</sup>
-  TSRVRadioButton<sup>(367)</sup>
-  TSRVScrollBar<sup>(243)</sup>

### 3.1.7 TSRVCustomListBox

TSRVCustomListBox is the base class for list box controls.

**Unit** SRVListBox;

#### Syntax

```
TSRVCustomListBox = class (TSRVWinControl340) ;
```

## Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl*<sup>340</sup>

## Properties and events

### Public properties

| Property                  | Type                                     | Default value                  | Meaning  |
|---------------------------|--|--------------------------------|--|
| <b>HideSelection</b>      | Boolean                                  | <i>True</i>                    | Determines whether the visual indication of the selected item remains when focus shifts to another control.  |
| <b>HotItemBackColor</b>   |  | <i>clBackgroun</i><br><i>d</i> | Background color for the highlighted item (below the mouse pointer).   |
| <b>HotItemBorderColor</b> |  | <i>clSilver</i>                | Border color for the highlighted item.   |
| <b>HotItemTextColor</b>   |  | <i>clWhite</i>                 | Text color for the highlighted item.   |
| <b>ImageList</b>          | TCustomImageList                         |                                | Specifies the images that appear next to items in the list box.  |
| <b>ItemHeight</b>         | Integer                                  |                                | Ignored. Item height is calculated automatically.  |
| <b>ItemIndex</b>          | Integer                                  |                                | Specifies the ordinal number of the selected item in the list box's item list. If the value of the <b>MultiSelect</b> property is <i>True</i> the user can select more than one item in the list box. In this case, the <b>ItemIndex</b> value is the index of the selected item that has focus. |
| <b>Items</b>              | TSRVListBoxItemCollection <sup>375</sup> |                                | Items, a collection of TSRVListBoxItem <sup>375</sup> .  |
| <b>ListMarginHeight</b>   | TRVPixel96Length                         | 2                              | Top and bottom margins in each item.   |
| <b>ListMarginWidth</b>    | TRVPixel96Length                         | 3                              | Left and right margins of each item.   |
| <b>MultiSelect</b>        | Boolean                                  | <i>False</i>                   | Determines whether the user can select more than one element at a time.  |
| ▶ <b>ScrollBar</b>        | TSRVScrollBar <sup>243</sup>             |                                | Read-only. Returns a scroll bar object.  |
| <b>ShowCheckBoxes</b>     | Boolean                                  | <i>False</i>                   | Shows/hides check boxes in each item.  |

| Property                            | Type    | Default value | Meaning   |
|-------------------------------------|---------|---------------|---|
| <b>Sorted</b>                       | Boolean | <i>False</i>  | Specifies whether the items in a list box are arranged alphabetically.  |
| <b>UseItemFontNames</b>             | Boolean | <i>False</i>  | Specifies whether the <b>FontName</b> properties of items are used to display their <b>Captions</b> .   |
| <b>UseSelectedImagesInHot State</b> | Boolean | <i>False</i>  | Specifies whether <code>Items[].SelectedImageIndex</code> is used instead of <code>Items[].ImageIndex</code> for highlighted items (below the mouse pointer). |

Public properties working only if **SRVControlStyle** = *srvcClassic*:

| Property                       | Type   | Default value          | Meaning                                 |
|--------------------------------|--------|------------------------|---|
| <b>SelectedItemBackColor</b>   | TColor | <i>clHighlight</i>     | Background color for the selected item. |
| <b>SelectedItemBorderColor</b> | TColor | <i>clSilver</i>        | Border color for the selected item.     |
| <b>SelectedItemTextColor</b>   | TColor | <i>clHighlightText</i> | Text color for the selected item.       |

### Public events

**OnMeasureItem** occurs when the application needs to redisplay an item in a variable height owner-draw list box.

**OnDrawItem** occurs when an item in an owner-draw list box needs to be redisplayed.

See also properties and events inherited from `TSRVWinControl` <sup>340</sup>.

### Methods

**BeginUpdate/EndUpdate** locks/unlocks redrawing.

**IndexOf** returns the index of item by its Caption.

**ItemHasParent** returns *True* if the given item has a parent item.

**GetItemParent** returns the index of the parent item for the given item.

**AppendItem** adds an item with the specified properties to the end of the item list.

**InsertItem** inserts an item with the specified properties at the position Index.

**DeleteItem** deletes the specified item.

---

## Inherited components

---

 [TSRVListBox](#) <sup>(359)</sup>

### 3.1.8 TSRVEditControl

TSRVEditControl is the base class for plain-text controls.

**Unit** SRVControl;

#### Syntax

```
TSRVEditControl = class (TSRVWinControl (340));
```

#### Hierarchy

---

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl* <sup>(340)</sup>

#### Properties and events

---

##### Public properties

| Property         | Type    | Default value | Meaning  |
|------------------|---------|---------------|--|
| <b>SelLength</b> | Integer |               | Specifies the number of selected characters.                               |
| <b>SelStart</b>  | Integer |               | Specifies the position of the first selected character in the text, from 0 |

See also properties inherited from *TSRVWinControl* <sup>(340)</sup>.

#### Methods

---

**SelectAll** selects all text in the edit control.

## Inherited components

---

 [TSRVEdit](#) <sup>(349)</sup>

 [TSRVMemo](#) <sup>(362)</sup>

 [TSRVComboBox](#) <sup>(346)</sup>

### 3.1.9 TSRVGraphicControl

TSRVGraphicControl is the base class for lightweight SRVControls.

**Unit** SRVControl;

#### Syntax

```
TSRVGraphicControl = class (TCustomSRVGraphicControl329) ;
```

## Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TGraphicControl*

*TCustomSRVGraphicControl*<sup>329</sup>

## Properties and events

### Public properties:

| Property        | Type    | Default value | Meaning   |
|-----------------|---------|---------------|---|
| <b>AutoSize</b> | Boolean | <i>True</i>   | Determines whether the control automatically resizes itself to accommodate its content. |

See also properties and events inherited from *TCustomSRVGraphicControl*<sup>329</sup>.

## Inherited components

**Abc** *TSRVLabel*<sup>358</sup>

### 3.1.10 TSRVWinControl

TSRVWinControl is the base class for all SRVControls that are wrappers for Microsoft Windows screen objects.

**Unit** SRVControl;

### Syntax

```
TSRVWinControl = class (TWinControl) ;
```

## Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

## Properties and events

### Public properties:

| Property            | Type    | Default value | Meaning   |
|---------------------|---------|---------------|---|
| <b>AutoDarkMode</b> | Boolean | <i>True</i>   | If <i>True</i> , and the control is inserted in a dark-mode editor (TSRichViewEdit or TRichView), |



| Property                   | Type    | Default value | Meaning  |
|----------------------------|---------|---------------|--|
|                            |         |               | luminance of all colors is inverted. Otherwise, <b>DarkMode</b> property is used. Supported only if <b>SRVControlStyle</b> = <i>svcsSimple</i> .   |
| ▶ <b>CaretPos</b>          | TPoint  |               | Read-only. Returns the caret position in the control. If the control does not have a caret, returns (-1, -1).  |
| ▶ <b>CaretHeight</b>       | Integer |               | Read-only. Returns the caret height in the control. If the control does not have a caret, returns -1.  |
| <b>DarkMode</b>            | Boolean | <i>False</i>  | If <i>True</i> , luminance of all colors are inverted. This property is used only if the control is not inserted in an editor, or if <b>AutoDarkMode</b> = <i>False</i> . Supported only if <b>SRVControlStyle</b> = <i>svcsSimple</i> . |
| <b>DrawOnPrint</b>         | Boolean | <i>True</i>   | if <i>True</i> , this control will be printed  |
| <b>DisabledBorderWidth</b> | Integer | 1             | Border width in disabled state   |
| <b>EnabledBorderWidth</b>  | Integer | 1             | Border width in normal state   |
| <b>FocusedBorderWidth</b>  | Integer | 1             | Border width when the control is focused   |
| ▶ <b>MouseIn</b>           | Boolean |               | Read-only. Returns True, if the mouse pointer is above the control   |
| <b>MouseInBorderWidth</b>  | Integer | 1             | Border width when the control is below the mouse pointer   |

Public properties working only if **SRVControlStyle** = *svcsClassic*

| Property                   | Type   | Default value   | Meaning  |
|----------------------------|--------|-----------------|--|
| <b>DisabledBorderColor</b> | TColor | <i>\$B8D8D7</i> | Border color in a disabled state                         |
| <b>EnabledBorderColor</b>  | TColor | <i>\$BF9F7F</i> | Border color in a normal state                           |
| <b>FocusedBorderColor</b>  | TColor | <i>clBlue</i>   | Border color when the control is focused                 |
| <b>MouseInBorderColor</b>  | TColor | <i>clRed</i>    | Border color when the control is below the mouse pointer |

#### Public events:

- **OnMouseEnter, OnMouseLeave** (TNotifyEvent) occur when the mouse pointer enter/leaves the control.

#### Protected properties:

| Property                          | Type                              | Default value      | Meaning   |
|-----------------------------------|-----------------------------------|--------------------|---|
| ■ <b>SkinManager</b>              | TSRVSkinManager <sup>(228)</sup>  |                    | A link to a skin manager component.   |
| ■ <b>SkinSchemeIndex</b>          | Integer                           | 0                  | An index in a scheme in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> (a scheme depends on the control type)                                    |
| ■ <b>ScrollBarSkinSchemeIndex</b> | Integer                           | 0                  | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .VerticalScrollBarSchemes <sup>(233)</sup> collection, for controls having scrollbars |
| ■ <b>ItemSkinSchemeIndex</b>      | Integer                           | 0                  | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> collection, for controls having items                    |
| ■ <b>EditSkinSchemeIndex</b>      | Integer                           | 0                  | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> collection, for controls having an embedded text editor  |
| ■ <b>ButtonSkinSchemeIndex</b>    | Integer                           | 0                  | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> collection, for controls having an embedded button       |
| ■ <b>SRVControlStyle</b>          | TSRVControlStyle <sup>(276)</sup> | <i>srvcsSimple</i> | Defines a visual appearance   |

## Inherited classes


- TSRVEditControl<sup>(339)</sup>
- TSRVCustomListBox<sup>(336)</sup>


## 3.2 Main Controls


### Controls:

 TSRVButton<sup>(343)</sup> – button;

 TSRVCheckBox<sup>(345)</sup> – check box;


 TSRVComboBox<sup>(346)</sup> – combo box; hierarchical; with images, custom colors and fonts; with hints (suggestions);

 TSRVEdit<sup>(349)</sup> – plain-text one-line editor with hints (suggestions);

 TSRVGroupBox<sup>(352)</sup> – panel with a title;

 TSRVImagesScroll<sup>(354)</sup> – gallery of images;

 TSRVLabel<sup>(358)</sup> – text label;

 TSRVListBox<sup>(359)</sup> – scrollable list of items; hierarchical; with images, check boxes, custom colors and fonts;

 TSRVMemo<sup>(362)</sup> – plain-text multi-line editor;

 **TSRVPaintBox** <sup>(364)</sup> – controls for custom drawing;

 **TSRVPanel** <sup>(365)</sup> – panel;

 **TSRVRadioButton** <sup>(367)</sup> – radio button.

**See also:**

 **TSRVScrollBar** <sup>(243)</sup>;

 **TSRVTabSet** <sup>(249)</sup>.

All the controls above can be skinned using **TSRVSkinManager** <sup>(228)</sup>.

### 3.2.1 TSRVButton

**TSRVLabel** is a control displaying text.

**Unit** SRVButton;

**Syntax**

```
TSRVButton = class (TSRVCustomControl (335))
```

#### Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TCustomControl*



*TSRVCustomControl* <sup>(335)</sup>

#### Properties and events

This component publishes the following properties inherited from **TSRVCustomControl** <sup>(335)</sup>:

- **DrawOnPrint**;
- **SkinManager**;
- **SkinSchemeIndex**;
- **SRVControlStyle**;
- properties inherited from **TCustomControl**.

**New published properties:**

| Property   | Type     | Default value | Meaning   |
|--|----------|---------------|---|
|  <b>Default</b> | Boolean  | <i>False</i>  | Determines whether the button's <b>OnClick</b> event handler executes when the <b>Enter</b> key is pressed.                       |
|  <b>Glyph</b>   | TPicture | (empty)       | Specifies the picture that appears on the button. The picture is displayed to the left/right side of text, depending on BiDiMode. |

| Property               | Type             | Default value | Meaning   |
|------------------------|------------------|---------------|---|
| ■ <b>ModalResult</b>   | TModalResult     | <i>mrNone</i> | Determines whether and how the button closes its (modal) parent form.                     |
| ■ <b>OffsetOnClick</b> | TRVPixel96Length | 0             | Specifies how far <b>Glyph</b> and <b>Caption</b> are shifted when the button is pressed. |

The following published properties work only if **SRVControlStyle** = *srvcsClassic*:

| Property                  | Type             | Default value | Meaning  |
|---------------------------|------------------|---------------|--|
| ■ <b>MouseBorderWidth</b> | TRVPixel96Length | 2             | Width of highlight frame under the mouse or when the button is focused |
| ■ <b>ShowFocusRect</b>    | Boolean          | <i>True</i>   | Shows/hides a dotted rectangle drawn on a focused button               |

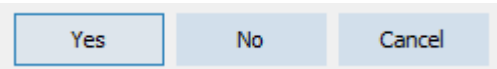
#### New published event:

- **OnClick**, occurs when the user clicks the control.

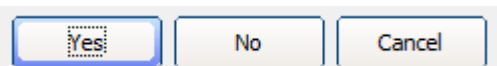
### Painting

The default button appearance (if skins are not assigned) depends on **SRVControlStyle**:  
 TSRVControlStyle<sup>(276)</sup> property:

*srvcsSimple* (if RVControlsPainter.Theme = *rvctPaleBlue*):

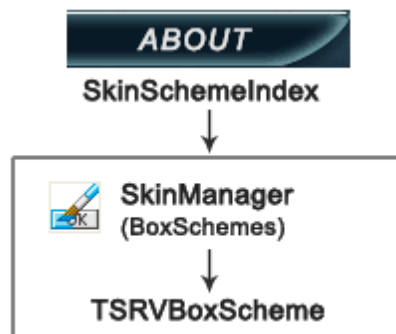


*srvcsClassic*:



If **SkinManager** is assigned, the button is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**SkinSchemeIndex**].

#### Example:



You can also use the skin font: **SkinManager.CurrentSkin**<sup>(230)</sup>.Fonts<sup>(232)</sup> [**SkinFontIndex**].

### 3.2.2 TSRVCheckBox

TSRVCheckBox represents a check box that can be on (checked) or off (unchecked)

**Unit** SRVCheckBox;

#### Syntax

```
TSRVCheckBox = class (TSRVCustomControl335)
```

#### Hierarchy




*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TSRVCustomControl*<sup>335</sup>

#### Properties and events

This component publishes the following properties inherited from *TSRVCustomControl*<sup>335</sup>:

- **Alignment**;
- **ButtonSkinSchemeIndex**;
- **DrawOnPrint**;
- **SkinManager, SkinSchemeIndex, SkinFontIndex**;
- **SRVControlStyle**;
- properties inherited from *TCustomControl*.

**New published properties:**

| Property   | Type              | Default value | Meaning   |
|--|-------------------|---------------|---|
|  <b>AllowGrayed</b> | Boolean           | <i>False</i>  | Determines whether a check box can be in a “grayed” state. Not supported in a skinned mode.   |
|  <b>Checked</b>     | Boolean           | <i>False</i>  | Specifies whether the check box is checked. If the <b>AllowGrayed</b> property is <i>True</i> , you may find it more useful to use the <b>State</b> property. |
|  <b>State</b>       | TSRVCheckBoxState | cbUnchecked   | Indicates whether the check box is selected ( <i>cbChecked</i> ), deselected ( <i>cbUnchecked</i> ), or grayed ( <i>cbGrayed</i> )                            |

#### Painting

The default check-box appearance (if skins are not assigned) depends on **SRVControlStyle**:  
*TSRVControlStyle*<sup>276</sup> property:

*srvcSimple* (if *RVControlsPainter.Theme = rvctPaleBlue*):

- ☒ Apple  
☒ Banana  
☐ Cherry

*srvcClassic:*

- ☒ Apple
- ☒ Banana
- ☐ Cherry

Positions of a box and **Caption** depend on **Alignment**:

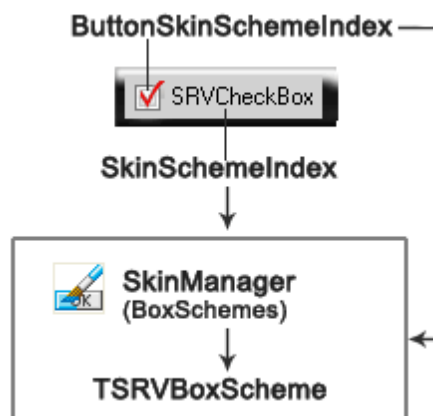
- *taLeftJustify*: the box and the caption are aligned to the left; the box is to the left of the caption;
- *taRightJustify*: the box and the caption are aligned to the right; the box is to the right of the caption;
- *taCenter*: the box and the caption are in the middle; the box is to the left/right of the caption depending in BiDiMode.

RTL BiDiMode inverts left and right **Alignment**.

Note: **Alignment** is different from the standard TCheckBox.Alignment. In TCheckBox, there are only left and right alignments, and they define Caption position relative to a box. In TSRVCheckBox, there are three alignments, and they define alignment of a box and Caption in the control (so the meanings are almost opposite).

If **SkinManager** is assigned, the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.  
**.BoxSchemes**<sup>(232)</sup>[**SkinSchemeIndex**], and the box is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.  
**.BoxSchemes**<sup>(232)</sup>[**ButtonSkinSchemeIndex**]. In this mode, a grayed state is not supported.

**Example:**



### 3.2.3 TSRVComboBox

TSRVComboBox combines an edit box with a scrollable list. Items can have captions and images. A list can be hierarchical. Additionally, editing suggestions (hints) may be displayed.

**Unit** SRVComboBox;

**Syntax**

```
TSRVComboBox = class (TSRVCustomComboBox(332))
```

**Hierarchy**

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl* <sup>(340)</sup>

*TSRVEditControl* <sup>(339)</sup>

*TSRVCustomComboBox* <sup>(332)</sup>

## Properties and events

This component publishes public properties inherited from *TSRVCustomComboBox* <sup>(332)</sup>.

## Hints (Suggestions)

A list of suggestions can be displayed on editing.

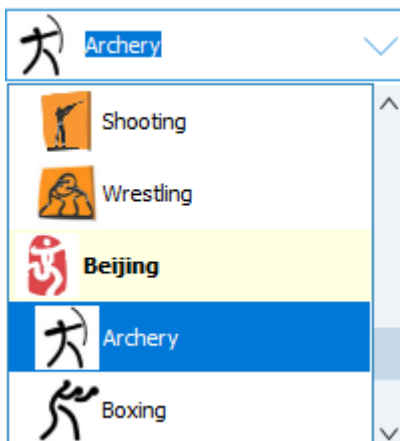
Suggestions are requested in the event **OnEditHints**. In this event, you can fill a list of suggestions (**EditHints**) for the **Text**, and specify the **Index** of item to highlight initially.

A height of this list is defined in **DropDownHints** property.

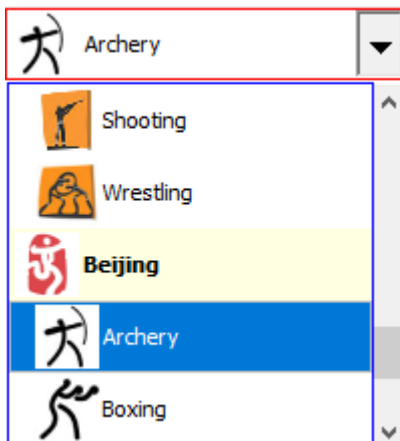
## Painting

The default combo box appearance (if skins are not assigned) depends on **SRVControlStyle**: *TSRVControlStyle* <sup>(276)</sup> property:

*srvcSimple* (if *RVControlsPainter.Theme = rvctPaleBlue*):



*srvcClassic* (the scroll bar appearance depends on Windows, this screenshot is for Windows 10)::



The border around the box has width defined in **DisabledBorderWidth**, **EnabledBorderWidth**, **MouseInBorderWidth**, **FocusedBorderWidth** properties. When auto-sizing the control, it is assumed that the border width is equal to **EnabledBorderWidth**.

A border color depends on the combo box state (normal, disabled, under the mouse pointer, focused) and **SRVControlStyle**: `TSRVControlStyle`<sup>(276)</sup> property. If it is equal to *srvcsSimple*, colors are defined by `RVControlsPainter` (see the `TRichView` manual). If it is equal to *srvcsClassic*, colors are defined in **EnabledBorderColor**, **DisabledBorderColor**, **MouseInBorderColor**, **FocusedBorderColor** properties. The same is for colors used for drawing selected items (**SelectedItemBackColor**, **SelectedItemBorderColor**, **SelectedItemTextColor**).

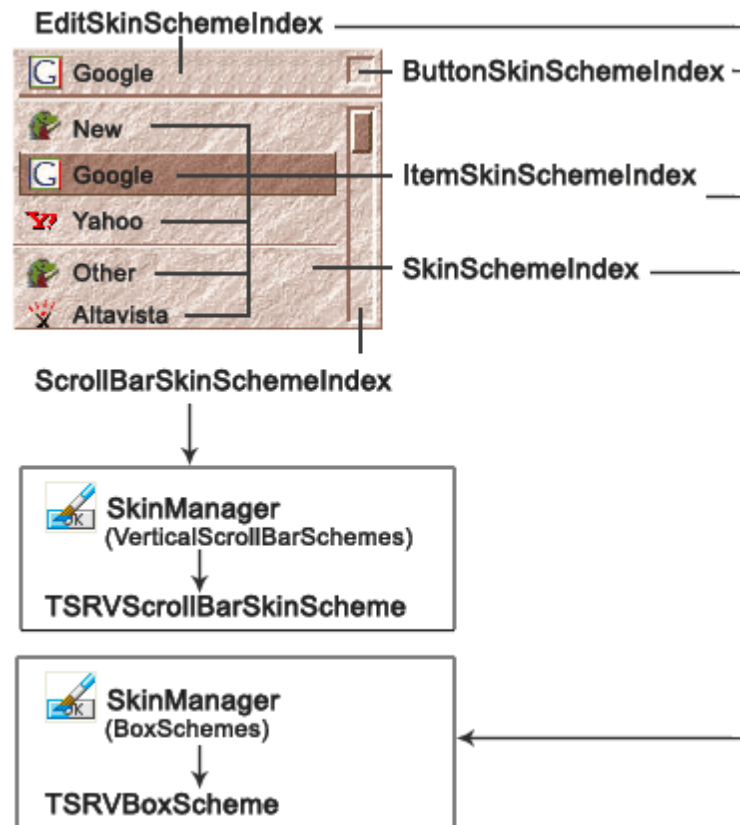
Item colors are defined in the items properties<sup>(375)</sup>.

If **SkinManager** is assigned:

- an editing region is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**EditSkinSchemeIndex**];
- a drop down list is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**SkinSchemeIndex**];
- each item in a drop down list is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**ItemSkinSchemeIndex**];
- a scrollbar of a drop down list is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup> [**ScrollBarSkinSchemeIndex**].
- a list of hints (suggestions) is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**HintsSkinSchemeIndex**];
- each item in a list of hints is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [**HintsItemSkinSchemeIndex**];
- a scrollbar of a list hints is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup> [**HintsScrollBarSkinSchemeIndex**].

**Example:**





### 3.2.4 TSRVEdit

TSRVEdit is a single-line plain-text edit control, with an optional popup list displaying hints (suggestions).

TSRVEdit edits Unicode text in all versions of Delphi.

**Unit** SRVEdit;

#### Syntax

```
TSRVEdit = class (TCustomSRVEdit327)
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TSRVWinControl*<sup>340</sup>  
*TSRVEditControl*<sup>339</sup>  
*TCustomSRVEdit*<sup>327</sup>

## Properties and events

This component publishes the following new properties and properties inherited from TCustomSRVEdit<sup>(327)</sup>:

| Property                 | Type                              | Default value        | Meaning   |
|--------------------------|-----------------------------------|----------------------|---|
| ■ <b>Alignment</b>       | TAlignment                        | <i>taLeftJustify</i> | Horizontal text alignment (left/right/center). RTL BiDiMode inverts left and right alignments.  |
| ■ <b>AutoSelect</b>      | Boolean                           | <i>True</i>          | Determines whether all the text in the edit control is automatically selected when the control gets focus.  |
| ■ <b>AutoShowHints</b>   | Boolean                           | <i>True</i>          | Determines whether hints (suggestions) are shown when the control is focused and when its text is edited. They can also be displayed by <b>DisplayHints</b> method. |
| ■ <b>AutoSize</b>        | Boolean                           | <i>True</i>          | Determines whether the height of the edit control automatically resizes to accommodate the text height.   |
| ■ <b>CharCase</b>        | TSRVEditCharCase                  | <i>srvecNormal</i>   | Determines the case of the text within the edit control:<br><i>srvecNormal, srvecUpperCase, srvecLowerCase</i>  |
| ■ <b>DropDownHints</b>   | Integer                           | 10                   | Defines how many hints (suggestions) are visible in a list.   |
| ■ <b>MaxLength</b>       | Integer                           | 0                    | Specifies the maximum number of characters the user can enter into the edit control (0 for unlimited)   |
| ■ <b>NumbersOnly</b>     | Boolean                           | <i>False</i>         | Allows only numbers to be typed into the text edit.   |
| ■ <b>PasswordChar</b>    | Char                              | #0                   | Indicates the character, if any, to display in place of the actual characters typed in the control (#0 to disable this feature)                                     |
| ■ <b>ReadOnly</b>        | Boolean                           | <i>False</i>         | Determines whether the user can change the text of the edit control   |
| ■ <b>SRVControlStyle</b> | TSRVControlStyle <sup>(276)</sup> | <i>srvecsSimple</i>  | Defines a visual appearance (mainly affects border colors)  |
| ■ <b>Text</b>            | TRVUnicodeString                  |                      | Contains a text displayed by the editor.  |

- **DrawOnPrint;**
- **SkinManager;**
- **EditSkinSchemeIndex, SkinSchemeIndex** (used in the hints window), **ScrollBarSkinSchemeIndex** (used in the hints window), **ItemSkinSchemeIndex** (used in the hints window);
- **SkinFontIndex;**

- **EnabledBorderColor, DisabledBorderColor, MouseInBorderColor, FocusedBorderColor** (all colors are used only if **SRVControlStyle** = *rvsrvcsClassic*)
- **DisabledBorderWidth, EnabledBorderWidth, MouseInBorderWidth, FocusedBorderWidth;**
- properties inherited from **TWinControl**.

#### New published events:

- **OnChange** occurs when the text for the edit control may have changed.
- **OnEditHints**: **TSRVEditHintsEvent** occurs when a list of hints (suggestions) is about to be displayed.
- **OnCloseHints**: **TFormHintsClose** occurs when a list of hints (suggestions) is closed.

#### type

```
TSRVEditHintsEvent = procedure (Sender: TSRVEditControl339;
  Text: TRVUnicodeString; EditHints: TRVStringList;
  var ItemIndex : Integer) of object;
TFormHintsClose = procedure (Sender: TObject;
  FormWidth, FormHeight: Integer;
  var NewText: TRVUnicodeString) of object;
```

where **TRVStringList** is a string list that contains Unicode string in all versions of Delphi (for Delphi 2009+, it contains **UnicodeString** items; for older versions of Delphi, it contains UTF-8 strings).

## Methods

**DisplayHints** displays a list of hints (suggestions).

**BeginHintsUpdate, EndHintsUpdate** blocks/unblocks hints (suggestions) showing.

See also the methods inherited from **TCustomSRVEdit**<sup>327</sup>.

## Hints (Suggestions)

A list of suggestions can be displayed on editing or by the method **ShowHints**. This feature can be temporary blocked using **BeginHintsUpdate/EndHintsUpdate**.

Suggestions are requested in the event **OnEditHints**. In this event, you can fill a list of suggestions (**EditHints**) for the **Text**, and specify the **Index** of item to highlight initially.

A height of this list is defined in **DropDownHints** property.

When a list of suggestions is closed, **OnCloseHints** event occurs. The parameter **NewText** contains the chosen suggestion (it is empty if canceled).

## Painting

### If skins are not defined:

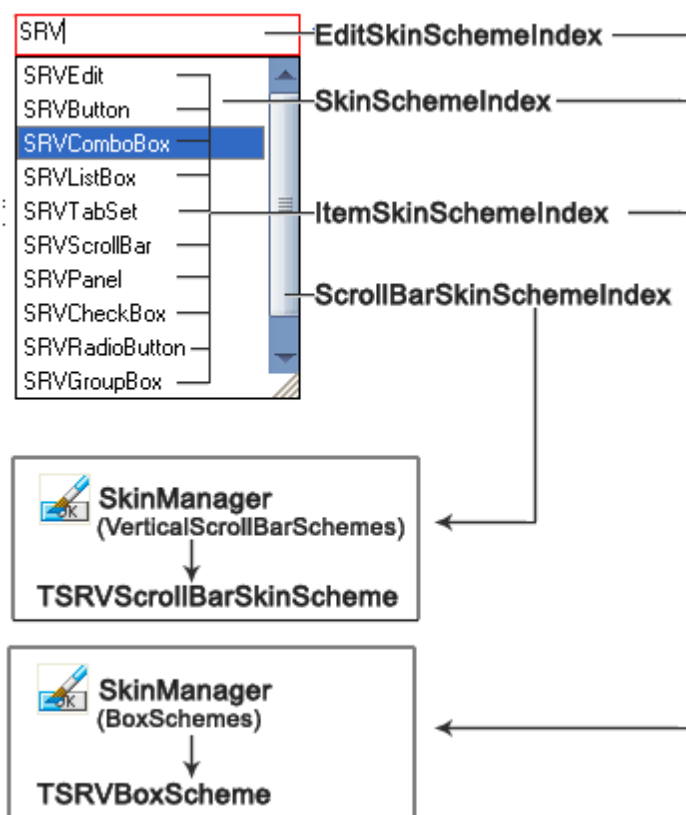
The border around the editor has width defined in **DisabledBorderWidth, EnabledBorderWidth, MouseInBorderWidth, FocusedBorderWidth** properties. When auto-sizing the editor, it is assumed that the border width is equal to **EnabledBorderWidth**.

A border color depends on the editors state (normal, disabled, under the mouse pointer, focused) and **SRVControlStyle**: **TSRVControlStyle**<sup>276</sup> property. If it is equal to *srvcsSimple*, colors are defined by **RVControlsPainter** (see the **TRichView** manual). If it is equal to *srvcsClassic*, colors are defined in **EnabledBorderColor, DisabledBorderColor, MouseInBorderColor, FocusedBorderColor** properties.

If **SkinManager** is assigned:

- the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup>[**EditSkinSchemeIndex**];
- a list of hints (suggestions) is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup>[**SkinSchemeIndex**];
- each item in this list is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup>[**ItemSkinSchemeIndex**];
- a scrollbar of this list is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup>[**ScrollBarSkinSchemeIndex**].
- font is defined in **SkinManager.CurrentSkin**<sup>(230)</sup>.Fonts<sup>(232)</sup>[**SkinFontIndex**].

Example:



### 3.2.5 TSRVGroupBox

TSRVGroupBox component is used to group related controls. This component is similar to TSRVPanel<sup>(365)</sup> (the difference is only in the caption position).

**Unit** SRVGroupBox;

**Syntax**

```
TSRVGroupBox = class (TCustomSRVPanel(330))
```

## Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TWinControl  
 TCustomControl  
 TSRVCustomControl<sup>(335)</sup>  
 TCustomSRVPanel<sup>(330)</sup>

## Properties and events

This component publishes the following new properties and properties inherited from TCustomSRVPanel<sup>(330)</sup>:

| Property                 | Type                              | Default value        | Meaning  |
|--------------------------|-----------------------------------|----------------------|--|
| ■ <b>Alignment</b>       | TAlignment                        | <i>taLeftJustify</i> | Horizontal text alignment (left/right/center). RTL BiDiMode inverts left and right alignments.                                 |
| ■ <b>BorderWidth</b>     | Integer                           | 1                    | Border width.  |
| ■ <b>CaptionOffset</b>   | TRVPixel96Length                  | 10                   | Caption offset (Caption cannot be displayed closer to the left and right side of the control than specified in this property). |
| ■ <b>SRVControlStyle</b> | TSRVControlStyle <sup>(276)</sup> | <i>srvcSimple</i>    | Defines visual appearance.   |

- **DrawOnPrint**;
- **SkinManager**;
- **SkinSchemeIndex**;
- **Color** (default value *clWhite*);
- properties inherited from TCustomControl.

Published properties used only if **SRVControlStyle** = *srvcClassic*:

| Property               | Type    | Default value     | Meaning   |
|------------------------|---------|-------------------|---|
| ■ <b>BorderColor</b>   | TColor  | <i>\$00B8D8D7</i> | Border color  |
| ■ <b>CornersOffset</b> | Integer | 10                | Value in range 1..30. Defines how much border corners are rounded |

## Painting

**Caption** is aligned according to **Alignment**. **CaptionOffset** shifts the text (to the right, if **Alignment**=*taLeftJustify*, or to the left, if **Alignment**=*taRightJustify*). Caption is placed at the top. RTL **BiDiMode** inverts **Alignment**.

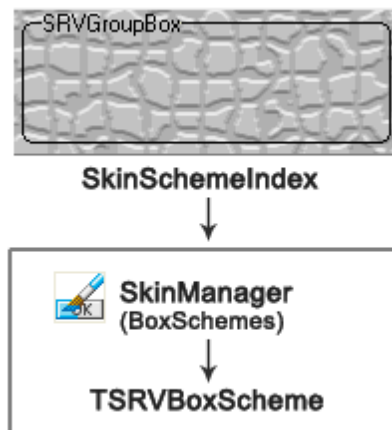
Background is drawn using **Color** property.

The Border has width defined in **BorderWidth** property (even in a skin mode). If **SRVControlStyle** = *srvcsSimple*, the border color is defined by RVControlsPainter (see the TRichView manual). If it is equal to *srvcsClassic*, the border color is defined in **BorderColor** property.

If **SRVControlStyle** = *srvcsClassic*, the border can be rounded according to **CornersOffset** property.

If **SkinManager** is assigned, the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.  
**.BoxSchemes**<sup>(232)</sup> [**SkinSchemeIndex**].

**Example:**



### 3.2.6 TSRVImagesScroll

The **TSRVImagesScroll** component is used to display images in a scrollable row (or column)

**Unit** **SRVImagesScroll**;

**Syntax**

```
TSRVImagesScroll = class (TSRVCustomControl(335))
```

**Hierarchy**

**TObject**  
**TPersistent**  
**TComponent**  
**TControl**  
**TWinControl**  
**TCustomControl**  
**TSRVCustomControl**<sup>(335)</sup>

**Properties**

This component publishes the following properties inherited from **TSRVCustomControl**<sup>(335)</sup>:

- **SkinManager**;
- **SkinSchemeIndex**;
- **SRVControlStyle**;
- properties inherited from **TCustomControl**.

**New published properties:**

| Property                 | Type                                       | Default value          | Meaning  |
|--------------------------|--|------------------------|--|
| ■ <b>CanMoveImages</b>   | Boolean                                    | <i>True</i>            | Determines whether items can be rearranged using the mouse.  |
| ■ <b>CloseButton</b>     | TSRVCloseButton <sup>(262)</sup>           |                        | Properties for closing button in each item.  |
| ■ <b>ImageList</b>       | TCustomImageList                           |                        | A link to an image list to display in items  |
| ■ <b>Indent</b>          | TRVPixel96Length                           | 10                     | Indent before the first item.  |
| ■ <b>ItemBorderStyle</b> | TPenStyle                                  | <i>psSolid</i>         | Item border pen style.   |
| ■ <b>ItemHeight</b>      | TRVPixel96Length                           | 40                     | Item height  |
| ■ <b>ItemIndex</b>       | Integer                                    |                        | Index of the selected item (in <b>Items</b> )  |
| ■ <b>Items</b>           | TSRVImageScrollCollection <sup>(375)</sup> |                        | Items, a collection of TSRVImageScrollItem <sup>(379)</sup>  |
| ■ <b>ItemWidth</b>       | TRVPixel96Length                           | 80                     | Item width   |
| ■ <b>Kind</b>            | TSRVImageScrollKind                        | <i>srvbkHorizontal</i> | Determines how images are placed ( <i>srvbkHorizontal</i> – in a row, <i>srvbkVertical</i> – in a column).     |
| ■ <b>RestrictMove</b>    | Boolean                                    | <i>False</i>           | Determines whether the user can move items outside visible area of the control (if <b>CanMoveImages=True</b> ) |
| ■ <b>Spacings</b>        | TRVPixel96Length                           | 10                     | Spacing between items.   |

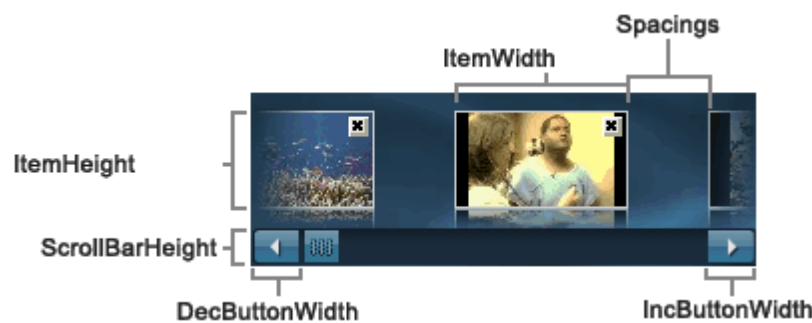
**New published properties-colors (for non-skin mode):**

| Property                   | Type   | Default value            | Meaning                             |
|----------------------------|--------|--------------------------|-------------------------------------|
| ■ <b>ItemBorderColor</b>   | TColor | <i>clBtnShadow</i>       | Item border color.                  |
| ■ <b>ItemColor</b>         |        | <i>clBtnFace</i>         | Item color in normal mode.          |
| ■ <b>ItemDownColor</b>     |        | <i>clInactiveCaption</i> | Item color when pressed.            |
| ■ <b>ItemHotColor</b>      |        | <i>clBtnShadow</i>       | Item color below the mouse pointer. |
| ■ <b>ItemSelectedColor</b> |        | <i>clHighlight</i>       | Item color when selected.           |

**New published properties related to scroll bar:**

| Property                | Type             | Default value | Meaning   |
|-------------------------|------------------|---------------|---|
| ■ <b>DecButtonWidth</b> | TRVPixel96Length | 0             | Width of up/left button in a skin mode. 0 for system default.               |
| ■ <b>IncButtonWidth</b> | TRVPixel96Length | 0             | Width of down/right button in a skin mode. 0 for system default.            |
| ■ <b>LargeChange</b>    | Integer          | 10            | Determines how much Position changes when the user clicks the scroll bar on |

| Property                  | Type                        | Default value   | Meaning  |
|---------------------------|-----------------------------|-----------------|--|
|                           |                             |                 | either side of the thumb tab.  |
| ■ <b>ScrollerPosition</b> | <i>TSRVScrollerPosition</i> | <i>srvpsEnd</i> | Scroll bar position ( <i>srvpsBegin</i> – top/left, <i>srvpsEnd</i> – bottom/right)                          |
| ■ <b>ScrollingDelay</b>   | Integer                     | 100             | Defines a scrolling speed (interval between scrolling events)  |
| ■ <b>SmallChange</b>      | Integer                     | 1               | Determines how much the scrollbar Position changes when the user clicks the arrow buttons on the scroll bar. |



#### Published properties related to skins:

| Property                          | Type                                    | Default value | Meaning  |
|-----------------------------------|---|---------------|--|
| ■ <b>SkinManager</b>              | <i>TSRVSkinManager</i> <sup>(228)</sup> |               | Inherited. A link to a skin manager component.   |
| ■ <b>SkinSchemeIndex</b>          | Integer                                 | 0             | Inherited. Index in <b>SkinManager</b> .CurrentSkin <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> , defines the appearance of the whole control.   |
| ■ <b>ScrollBarSkinSchemeIndex</b> | Integer                                 | 0             | Index in <b>SkinManager</b> .CurrentSkin <sup>(230)</sup> .HorizontalScrollBarSchemes <sup>(232)</sup> (or VerticalScrollBarSchemes <sup>(233)</sup> in a vertical mode), defines the appearance of scrollbar. |
| ■ <b>ItemSkinSchemeIndex</b>      | Integer                                 | 0             | Index in <b>SkinManager</b> .CurrentSkin <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> , defines the appearance of each item.  |
| ■ <b>ButtonSkinSchemeIndex</b>    | Integer                                 | 0             | Index in <b>SkinManager</b> .CurrentSkin <sup>(230)</sup> .BoxSchemes <sup>(232)</sup> , defines the appearance of closing buttons.  |

#### Events

type



```

TSRVCloseImageScrollEvent = procedure(Sender: TSRVImagesScroll;
  ItemIndex : Integer; var CanClose : Boolean) of object;
TSRVImageScrollMoveEvent = procedure(Sender: TSRVImagesScroll;
  NewImageIndex, OldImageIndex : Integer) of object;
TSRVDrawImageScrollEvent = procedure(Sender: TSRVImagesScroll;
  Canvas : TCanvas; ARect, PaintRect : TRect; State : TSRVDrawStates;
  var DoDefault : Boolean) of object;
TSRVDrawImageScrollItemEvent = procedure(Sender: TSRVImagesScroll;
  Canvas : TCanvas; ARect, PaintRect : TRect; ItemIndex : Integer;
  State : TSRVDrawStates; var DoDefault : Boolean) of object;

```

**OnChange:** TNotifyEvent occurs after a new item is selected.

**OnItemMove:** TSRVImageScrollMoveEvent occurs when two items exchange their places as a result of item moving. Similar to OnMoveTab<sup>(261)</sup> for TSRVTabSet<sup>(249)</sup>.

**OnCloseItem :** TSRVCloseImageScrollEvent occurs when the user attempts to close an item.

**OnDrawBorder, OnDrawBackground, OnDrawItem, OnDrawCloseButton:**

**TSRVDrawImageScrollEvent** allow custom drawing for the corresponding objects. Canvas – a canvas where to draw. ARect – a rectangle defining the position of the object to draw. APaintRect – a rectangle that needs to be redrawn; you can use this parameter to optimize drawing. State – a set describing the current object state. Set DoDefault to *False* to prevent the default drawing of this object.

## Painting

The default appearance (if skins are not assigned) depends on **SRVControlStyle**:

TSRVControlStyle<sup>(276)</sup> property:

*srvcsSimple* (if RVControlsPainter.Theme = *rvctPaleBlue*):



*srvcsClassic* (the scroll bar appearance depends on Windows, this screenshot is for Windows 10):

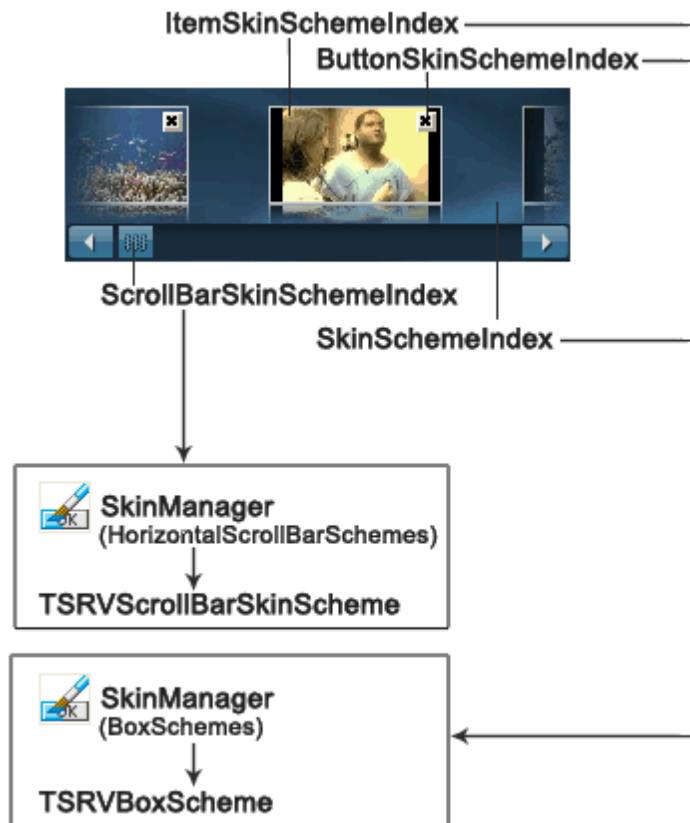


If **SRVControlStyle** = *srvcsSimple*, all colors are defined in RVControlsPainter (see the TRichView manual for details).

If **SRVControlStyle** = *srvcsClassic*, the control uses **Color**, **ItemBorderColor**, **ItemColor**, **ItemDownColor**, **ItemHotColor**, **ItemSelectedColor** properties.

If **SkinManager** is assigned, the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup>[**SkinSchemeIndex**], the scroll bar is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.HorizontalScrollBarSchemes<sup>(232)</sup>[**ScrollBarSkinSchemeIndex**] (or VerticalScrollBarSchemes<sup>(233)</sup>, if a scroll bar is vertical).

**Example:**



### 3.2.7 TSRVLabel

TSRVLabel is a control displaying text.

**Unit** SRVLabel;

#### Syntax

```
TSRVLabel = class(TSRVGraphicControl339);
```

#### Hierarchy




*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TGraphicControl*  
*TCustomSRVGraphicControl*<sup>329</sup>  
*TSRVGraphicControl*<sup>339</sup>

#### Properties and events

This component publishes the following properties inherited from *TSRVGraphicControl*<sup>339</sup>:

- Alignment
- AutoSize
- DrawOnPrint;

- SkinManager;
- SkinSchemeIndex;
- properties inherited from TGraphicControl.
- 

| Property   | Type             | Default value | Meaning                                   |
|--|------------------|---------------|---|
|  <b>BackgroundColor</b> | TColor           | \$881C10      | Text shadow color                         |
|  <b>ForegroundColor</b> | TColor           | \$FFFFFF      | Text color (overrides <b>Font.Color</b> ) |
|  <b>Offset</b>          | TRVPixel96Length | 1             | Shadow offset (0 disables shadow)         |

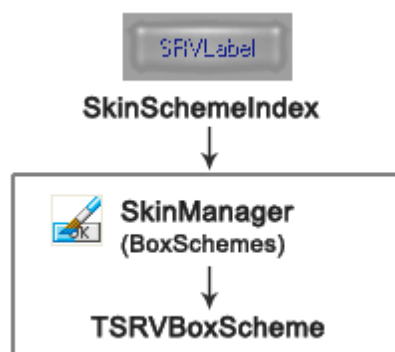
## Painting

This component displays text specified in **Caption**. This component is transparent.

**ForegroundColor** is a text color, **BackgroundColor** is a text shadow color. A shadow is shifted by **Offset**. The caption is aligned according to **Alignment** (RTL **BiDiMode** inverts alignments). The control size is set automatically, if **AutoSize=True**.

If **SkinManager** is assigned, a border is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.**BoxSchemes**<sup>(232)</sup> [**SkinSchemeIndex**].

Example:



### 3.2.8 TSRVListBox

Use TSRVListBox to display a scrollable list of items. Items can have captions, images, check boxes. A list can be hierarchical.

**Unit** SRVListBox;

#### Syntax

```
TSRVListBox = class (TSRVCustomListBox(336))
```

## Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl* <sup>(340)</sup>

*TSRVCustomListBox* <sup>(336)</sup>

## Properties and events

---

This component publishes public properties inherited from *TSRVCustomListBox* <sup>(336)</sup>.

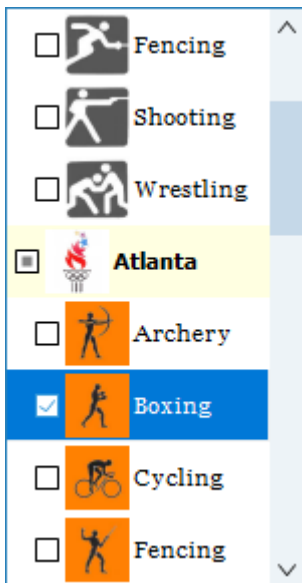
## Painting

---

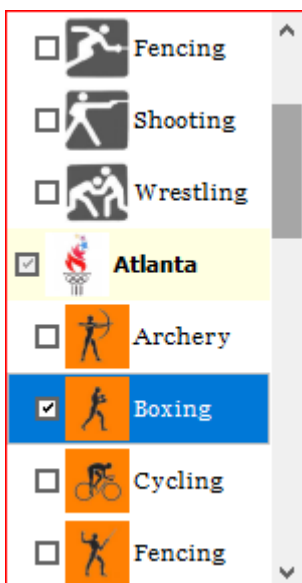
The default list box appearance (if skins are not assigned) depends on **SRVControlStyle**:

*TSRVControlStyle* <sup>(276)</sup> property:

*srvcSimple* (if *RVControlsPainter.Theme = rvctPaleBlue*):



*srvcClassic* (the scroll bar appearance depends on Windows, this screenshot is for Windows 10)::



The border around the list box has width defined in **DisabledBorderWidth**, **EnabledBorderWidth**, **MouseInBorderWidth**, **FocusedBorderWidth** properties. When auto-sizing the control, it is assumed that the border width is equal to **EnabledBorderWidth**.

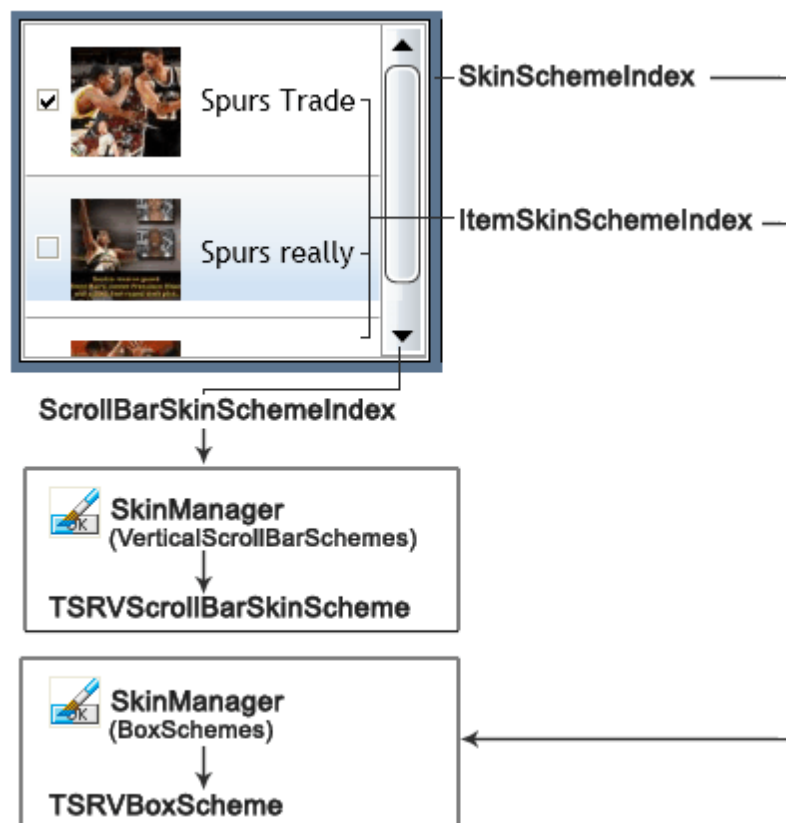
A border color depends on the list box state (normal, disabled, under the mouse pointer, focused) and **SRVControlStyle**: **TSRVControlStyle**<sup>(276)</sup> property. If it is equal to *srvcsSimple*, colors are defined by **RVControlsPainter** (see the **TRichView** manual). If it is equal to *srvcsClassic*, colors are defined in **EnabledBorderColor**, **DisabledBorderColor**, **MouseInBorderColor**, **FocusedBorderColor** properties. The same is for colors used for drawing selected items (**SelectedItemBackColor**, **SelectedItemBorderColor**, **SelectedItemTextColor**).

Item colors are defined in the items properties<sup>(375)</sup>.

If **SkinManager** is assigned:

- the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.**BoxSchemes**<sup>(232)</sup> [**SkinSchemeIndex**];
- each item is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.**BoxSchemes**<sup>(232)</sup> [**ItemSkinSchemeIndex**];
- a scrollbar is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.**VerticalScrollBarSchemes**<sup>(233)</sup> [**ScrollBarSkinSchemeIndex**].

Example:



### 3.2.9 TSRVMemo

TSRVMemo is a multiline plain-text edit control.

TSRVMemo edits Unicode text in all versions of Delphi.

**Unit** SRVMemo;

#### Syntax

```
TSRVMemo = class (TCustomSRVEdit(327))
```

#### Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TSRVWinControl*<sup>(340)</sup>

*TSRVEditControl*<sup>(339)</sup>

*TCustomSRVEdit*<sup>(327)</sup>

#### Properties and events

This component publishes the following new properties and properties inherited from TCustomSRVEdit<sup>(327)</sup>:

| Property                       | Type             | Default value        | Meaning   |
|--------------------------------|------------------|----------------------|---|
| ■ <b>Alignment</b>             | TAlignment       | <i>taLeftJustify</i> | Horizontal text alignment (left/right/center). RTL BiDiMode inverts left and right alignments.  |
| ■ <b>CharCase</b>              | TSRVEditCharCase | <i>srvecNormal</i>   | Determines the case of the text within the edit control:<br><i>srvecNormal</i> , <i>srvecUpperCase</i> , <i>srvecLowerCase</i>  |
| ■ <b>HScrollBarSchemeIndex</b> | Integer          | 0                    | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .HorizontalScrollBarSchemes <sup>(232)</sup> collection, allows applying skins to a horizontal scrollbar                      |
| ■ <b>Lines</b>                 | TSRVMemoStrings  |                      | Contains the individual lines of text in the memo control. <b>Lines</b> contain Unicode strings for all versions of Delphi. WordWrap affects how <b>Text</b> is split into <b>Lines</b> . |
| ■ <b>MaxLength</b>             | Integer          | 0                    | Specifies the maximum number of characters the user can enter into the edit control (0 for unlimited)   |
| ■ <b>ReadOnly</b>              | Boolean          | <i>False</i>         | Determines whether the user can change the text of the edit control.  |

| Property                       | Type                              | Default value      | Meaning  |
|--------------------------------|-----------------------------------|--------------------|--|
| ■ <b>ScrollBars</b>            | TScrollStyle                      | <i>ssNone</i>      | Determines whether the memo control has scroll bars.   |
| ■ <b>SRVControlStyle</b>       | TSRVControlStyle <sup>(276)</sup> | <i>srvcsSimple</i> | Defines a visual appearance (affects borders and scroll bars).   |
| ■ <b>Text</b>                  |                                   |                    | Contains a text displayed by the editor (the same text as accessible by <b>Lines</b> ).  |
| ■ <b>VScrollBarSchemeIndex</b> | Integer                           | 0                  | An index in <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .VerticalScrollBarSchemes <sup>(233)</sup> collection, allows applying skins to a vertical scrollbar |
| ■ <b>WantReturns</b>           | Integer                           | <i>True</i>        | Determines whether the user can insert return characters into the text.  |
| ■ <b>WantTabs</b>              | Boolean                           | <i>False</i>       | Determines whether the user can insert tab characters into the text.   |
| ■ <b>WordWrap</b>              | Boolean                           | <i>True</i>        | Determines whether the edit control automatically breaks lines so that the text wraps to fit the control's window.   |

- **DrawOnPrint**;
- **SkinManager**;
- **EditSkinSchemeIndex**;
- **SkinFontIndex**;
- **EnabledBorderColor**, **DisabledBorderColor**, **MouseInBorderColor**, **FocusedBorderColor** (all colors are used only if **SRVControlStyle** = *rvsrvcsClassic*)
- **DisabledBorderWidth**, **EnabledBorderWidth**, **MouseInBorderWidth**, **FocusedBorderWidth**;
- properties inherited from **TWinControl**.

## Painting

### If skins are not defined:

The border around the editor has width defined in **DisabledBorderWidth**, **EnabledBorderWidth**, **MouseInBorderWidth**, **FocusedBorderWidth** properties. When auto-sizing the editor, it is assumed that the border width is equal to **EnabledBorderWidth**.

A border color depends on the editors state (normal, disabled, under the mouse pointer, focused) and **SRVControlStyle**: **TSRVControlStyle**<sup>(276)</sup> property. If it is equal to *srvcsSimple*, colors are defined by **RVControlsPainter** (see the **TRichView** manual). If it is equal to *srvcsClassic*, colors are defined in **EnabledBorderColor**, **DisabledBorderColor**, **MouseInBorderColor**, **FocusedBorderColor** properties.

If **SkinManager** is assigned:

- the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.BoxSchemes<sup>(232)</sup>[**EditSkinSchemeIndex**];
- the horizontal scroll bar is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.HorizontalScrollBarSchemes<sup>(232)</sup>[**HScrollBarSkinSchemeIndex**];

- the vertical scroll bar is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.VerticalScrollBarSchemes<sup>(233)</sup> [VScrollBarSkinSchemeIndex];
- font is defined in **SkinManager**.CurrentSkin<sup>(230)</sup>.Fonts<sup>(232)</sup> [SkinFontIndex].

### 3.2.10 TSRVPaintBox

TSRVPaintBox provides a canvas that applications can use for rendering an image.

**Unit** SRVPaintBox;

#### Syntax

```
TSRVPaintBox = class (TSRVCanvasControl(331));
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TGraphicControl*  
*TCustomSRVGraphicControl*<sup>(329)</sup>  
*TSRVCanvasControl*<sup>(331)</sup>

#### Properties and events

This component publishes the following properties and events inherited from TSRVCanvasControl<sup>(331)</sup>:

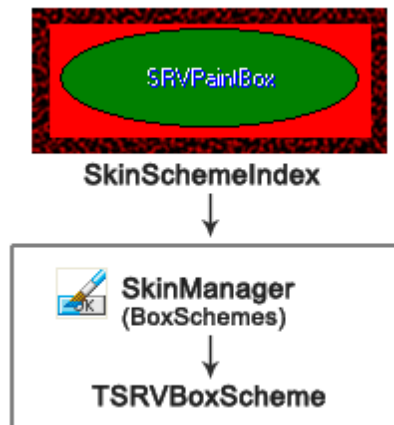
- **SkinManager**;
- **SkinSchemeIndex**;
- **DrawOnPrint**;
- **OnPaint**;
- **OnMouseEnter**, **OnMouseLeave**;
- properties inherited from TGraphicControl.

#### Painting

- 1) **Caption** is drawn at the position **Alignment** using Font (RTL **BiDiMode** inverts right and left alignments).
- 2) **OnPaint** event occurs (see TSRVCanvasControl<sup>(331)</sup>). Create an **OnPaint** event handler to draw the image of the paint box.
- 3) If **SkinManager** is assigned, a border is drawn using **SkinManager**.CurrentSkin<sup>(230)</sup>.BoxSchemes<sup>(232)</sup> [SkinSchemeIndex].

**Example** (Caption is empty; the text, the ellipse, the red rectangle are drawn in OnPaint):





### 3.2.11 TSRVPanel

TSRVPanel implements a generic panel control. You can also use panels to group controls together. This component is similar to TSRVGroupBox<sup>(352)</sup> (the difference is only in the caption position).

**Unit** SRVPanel;

#### Syntax

```
TSRVPanel = class (TCustomSRVPanel(330))
```

#### Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TWinControl  
 TCustomControl  
 TSRVCustomControl<sup>(335)</sup>  
 TCustomSRVPanel<sup>(330)</sup>

#### Properties and events

This component publishes the following new properties and properties inherited from TCustomSRVPanel<sup>(330)</sup>:

| Property               | Type             | Default value | Meaning  |
|------------------------|------------------|---------------|--|
| ■ <b>Alignment</b>     | TAlignment       | taCenter      | Horizontal text alignment (left/right/center). RTL BiDiMode inverts left and right alignments.                                 |
| ■ <b>BorderWidth</b>   | Integer          | 1             | Border width.  |
| ■ <b>CaptionOffset</b> | TRVPixel96Length | 10            | Caption offset (Caption cannot be displayed closer to the left and right side of the control than specified in this property). |

| Property                 | Type                              | Default value     | Meaning                    |
|--------------------------|-----------------------------------|-------------------|----------------------------|
| ■ <b>SRVControlStyle</b> | TSRVControlStyle <sup>(276)</sup> | <i>srvcSimple</i> | Defines visual appearance. |

- **DrawOnPrint;**
- **SkinManager;**
- **SkinSchemeIndex;**
- **Color** (default value *c/White*);
- properties inherited from TCustomControl.

Published properties used only if **SRVControlStyle** = *srvcClassic*:

| Property               | Type    | Default value     | Meaning   |
|------------------------|---------|-------------------|---|
| ■ <b>BorderColor</b>   | TColor  | <i>\$00B8D8D7</i> | Border color  |
| ■ <b>CornersOffset</b> | Integer | 10                | Value in range 1..30. Defines how much border corners are rounded |

## Painting

**Caption** is aligned horizontally according to **Alignment**. **CaptionOffset** shifts the text (to the right, if **Alignment**=*taLeftJustify*, or to the left, if **Alignment**=*taRightJustify*). Caption is placed vertically in the middle. RTL **BiDiMode** inverts **Alignment**.

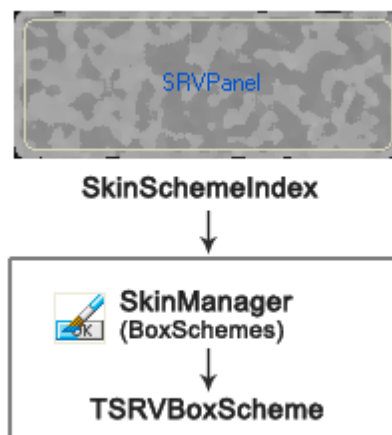
Background is drawn using **Color** property.

The Border has width defined in **BorderWidth** property (even in a skin mode). If **SRVControlStyle** = *srvcSimple*, the border color is defined by RVControlsPainter (see the TRichView manual). If it is equal to *srvcClassic*, the border color is defined in **BorderColor** property.

If **SRVControlStyle** = *srvcClassic*, the border can be rounded according to **CornersOffset** property.

If **SkinManager** is assigned, the control is drawn using **SkinManager.CurrentSkin**<sup>(230)</sup>.  
**BoxSchemes**<sup>(232)</sup> [**SkinSchemeIndex**].

**Example:**



### 3.2.12 TSRVRadioButton

Radio buttons present a set of mutually exclusive options to the user; that is, only one radio button in a set can be selected at a time.

**Unit** SRVRadioButton;

#### Syntax

```
TSRVRadioButton = class (TSRVCustomControl335)
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TCustomControl*  
*TSRVCustomControl*<sup>335</sup>

#### Properties and events

This component publishes the following properties inherited from *TSRVCustomControl*<sup>335</sup>:

- Alignment;
- ButtonSkinSchemeIndex;
- DrawOnPrint;
- SkinManager, SkinSchemeIndex, SkinFontIndex;
- SRVControlStyle;
- properties inherited from *TCustomControl*.

#### New published properties:

| Property   | Type    | Default value | Meaning  |
|--|---------|---------------|--|
| <input checked="" type="checkbox"/> <b>Checked</b> | Boolean | <i>False</i>  | Determines whether the option represented by the radio button is selected. |

#### Painting

#### Painting

The default radio-button appearance (if skins are not assigned) depends on **SRVControlStyle**: *TSRVControlStyle*<sup>276</sup> property:

*srvcsSimple* (if *RVControlsPainter.Theme = rvctPaleBlue*):

- ☒ Red  
☐ Green  
☐ Blue

*srvcsClassic*:



Positions of a circle and **Caption** depend on **Alignment**:

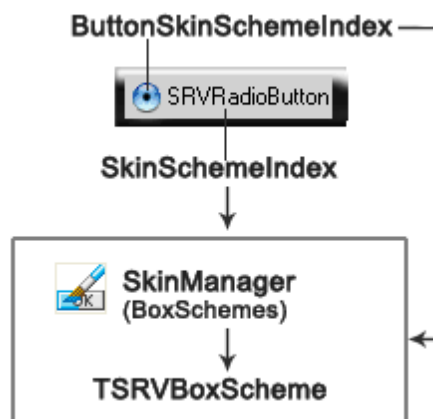
- *taLeftJustify*: the circle and the caption are aligned to the left; the circle is to the left of the caption;
- *taRightJustify*: the circle and the caption are aligned to the right; the circle is to the right of the caption;
- *taCenter*: the circle and the caption are in the middle; the circle is to the left/right of the caption depending in BiDiMode.

RTL BiDiMode inverts left and right **Alignment**.

Note: **Alignment** is different from the standard `TRadioButton.Alignment`. In `TRadioButton`, there are only left and right alignments, and they define Caption position relative to a circle. In `TSRVRadioButton`, there are three alignments, and they define alignment of a circle and Caption in the control (so the meanings are almost opposite).

If **SkinManager** is assigned, the control is drawn using `SkinManager.CurrentSkin`<sup>(230)</sup>.  
`.BoxSchemes`<sup>(232)</sup> [`SkinSchemeIndex`], and the circle is drawn using `SkinManager.CurrentSkin`<sup>(230)</sup>.  
`.BoxSchemes`<sup>(232)</sup> [`ButtonSkinSchemeIndex`].

Example:



### 3.3 Data-Aware Controls

Data-aware controls:



`TSRVDBCheckBox`<sup>(369)</sup> – data-aware check box;



`TSRVDBComboBox`<sup>(346)</sup> – data-aware combo box; hierarchical; with images, custom colors and fonts; with hints (suggestions);



`TSRVDBEdit`<sup>(371)</sup> – data-aware plain-text one-line editor with hints (suggestions);



`TSRVDBListBox`<sup>(372)</sup> – data-aware scrollable list of items; hierarchical; with images, check boxes, custom colors and fonts;



TSRVDBMemo<sup>(373)</sup> – data-aware plain-text multi-line editor;



TSRVDBText<sup>(374)</sup> – data-aware text label.

### 3.3.1 TSRVDBCheckBox

**TSRVDBCheckBox** is a data-aware control that allows the user to select or deselect a single value.

**Unit** SRVDBCheckBox;

#### Syntax

```
TSRVDBCheckBox = class (TSRVCheckbox(345))
```

#### Hierarchy

*TObject*

*TPersistent*

*TComponent*

*TControl*

*TWinControl*

*TCustomControl*

*TSRVCustomControl*<sup>(335)</sup>

*TSRVCheckbox*<sup>(345)</sup>

#### Description

Use **TSRVDBCheckBox** to insert a data-aware check box in an editor. A check box presents an option to the user; the user can check it to select the option or uncheck it to deselect the option. A database check box (**TSRVDBCheckBox**) is much like an ordinary check box (**TSRVCheckbox**), except that it is aware of the data in a particular field of a dataset. Because check boxes can represent only two values (checked and unchecked), database check boxes are most appropriate for boolean fields. They can, however, be used to group the values of any field into two sets.

For applications that don't require the data-aware capabilities of **TSRVDBCheckBox**, use **TSRVCheckbox**<sup>(345)</sup> instead to conserve system resources.

#### Properties and events

##### New properties:

| Property            | Type        | Default value    | Meaning   |
|---------------------|-------------|------------------|---|
| ■ <b>DataField</b>  | String      | " (empty string) | Identifies the field from which the control displays data.            |
| ■ <b>DataSource</b> | TDataSource | nil              | Links the control to a dataset.                                       |
| <b>Field</b>        | TField      |                  | Returns the TField object whose current value the control represents. |
| ■ <b>ReadOnly</b>   | Boolean     | False            | Determines whether the user can change the value of the field.        |

| Property                | Type   | Default value | Meaning  |
|-------------------------|--------|---------------|--|
| ■ <b>ValueChecked</b>   | String | 'True'        | Specifies the field value that corresponds to the checked state of the check box. <b>ValueChecked</b> can represent more than one value in a semicolon-delimited list.     |
| ■ <b>ValueUnchecked</b> | String | 'False'       | Specifies the field value that corresponds to the unchecked state of the check box. <b>ValueUnchecked</b> can represent more than one value in a semicolon-delimited list. |

### 3.3.2 TSRVDBComboBox

**TSRVDBComboBox** represents a data-aware combo box control.

**Unit** SRVDBComboBox;

#### Syntax

```
TSRVDBComboBox = class (TSRVComboBox346)
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TComponent*  
*TControl*  
*TWinControl*  
*TSRVWinControl*<sup>340</sup>  
*TSRVEditControl*<sup>339</sup>  
*TSRVCustomComboBox*<sup>332</sup>  
*TSRVComboBox*<sup>346</sup>

#### Description

Use **TSRVDBComboBox** to allow users to change the value of a field on the current record in a dataset either by selecting an item from a list or by typing in the edit box part of the control. The selected item or entered text becomes the new value of the field if the database combo box's **ReadOnly** property is *False*. The combo box can be customized to enable or disable typing in the edit region of the control, to display the list as a drop down or as a permanently displayed list, to sort the items in the list, and so on.

Unlike the standard **TDBComboBox**, **TSRVDBComboBox** can be used in **TDBCtrlGrid** even if its **Style** = *csSimple*.

#### Properties and events

**New properties:**

| Property            | Type        | Default value    | Meaning   |
|---------------------|-------------|------------------|---|
| ■ <b>DataField</b>  | String      | " (empty string) | Identifies the field from which the control displays data.            |
| ■ <b>DataSource</b> | TDataSource | <i>nil</i>       | Links the control to a dataset.                                       |
| <b>Field</b>        | TField      |                  | Returns the TField object whose current value the control represents. |
| ■ <b>ReadOnly</b>   | Boolean     | <i>False</i>     | Determines whether the user can change the value of the field.        |

### 3.3.3 TSRVDBEdit

**TSRVDBEdit** represents a single-line edit control that can display and edit a field in a dataset.

**Unit** SRVDBEdit;

#### Syntax

```
TSRVEdit = class (TSRVEdit349)
```

#### Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TWinControl  
 TSRVWinControl<sup>340</sup>  
 TSRVEditControl<sup>339</sup>  
 TCustomSRVEdit<sup>327</sup>  
 TSRVEdit<sup>349</sup>

#### Description

Use TSRVDBEdit to enable users to edit a database field. TSRVDBEdit uses the Text property to represent the contents of the field.

TSRVDBEdit permits only a single line of text. If the field may contain lengthy data that would require multiple lines, consider using a TSRVDBMemo<sup>373</sup> object.

If the application does not require the data-aware capabilities of TSRVDBEdit, use an edit control (TSRVEdit<sup>349</sup>) instead, to conserve system resources.

#### Properties and events

**New properties:**

| Property            | Type        | Default value             | Meaning   |
|---------------------|-------------|---------------------------|---|
| ■ <b>DataField</b>  | String      | " ( <i>empty string</i> ) | Identifies the field from which the control displays data.            |
| ■ <b>DataSource</b> | TDataSource | <i>nil</i>                | Links the control to a dataset.                                       |
| <b>Field</b>        | TField      |                           | Returns the TField object whose current value the control represents. |
| ■ <b>ReadOnly</b>   | Boolean     | <i>False</i>              | Determines whether the user can change the value of the field.        |

### 3.3.4 TSRVDBListBox

TSRVDBListBox represents a data-aware list box that allows users to change field values by selecting an item from a list.

**Unit** SRVDBListBox;

#### Syntax

```
TSRVListBox = class(TSRVListBox359)
```

#### Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TWinControl  
 TSRVWinControl<sup>340</sup>  
 TSRVCustomListBox<sup>336</sup>  
 TSRVListBox<sup>359</sup>

#### Description

Use TSRVDBListBox to add a list box to a document that permits users to change the value of a field on the current record to one of a fixed set of choices.

If the application doesn't require the data-aware capabilities of TSRVDBListBox, use a list box (TSRVListBox<sup>359</sup>) instead to conserve system resources.

#### Properties and events

**New properties:**

| Property            | Type        | Default value             | Meaning  |
|---------------------|-------------|---------------------------|--|
| ■ <b>DataField</b>  | String      | " ( <i>empty string</i> ) | Identifies the field from which the control displays data. |
| ■ <b>DataSource</b> | TDataSource | <i>nil</i>                | Links the control to a dataset.                            |



| Property          | Type    | Default value | Meaning   |
|-------------------|---------|---------------|---|
| <b>Field</b>      | TField  |               | Returns the TField object whose current value the control represents. |
| ■ <b>ReadOnly</b> | Boolean | <i>False</i>  | Determines whether the user can change the value of the field.        |

### 3.3.5 TSRVDBMemo

**TSRVDBMemo** represents a multiline edit control that can display and edit a field in a dataset.

**Unit** SRVDBMemo;

#### Syntax

```
TSRVMemo = class (TSRVMemo362)
```

#### Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TWinControl  
 TSRVWinControl<sup>340</sup>  
 TSRVEditControl<sup>339</sup>  
 TCustomSRVEdit<sup>327</sup>  
 TSRVMemo<sup>362</sup>

#### Description

Use TSRVDBMemo to let users edit a field that may contain lengthy textual data or to simply display the contents of such a field. TSRVDBMemo uses the Text property to represent the contents of the field.

TSRVDBMemo permits multiple lines of text. Thus, TSRVDBMemo is appropriate for long alphanumeric fields or text binary large objects (BLOBs). For short alphanumeric fields, consider using a TSRVDBEdit<sup>371</sup> component instead.

If the application doesn't require the data-aware capabilities of TSRVDBMemo, use a memo control (TSRVMemo<sup>362</sup>) instead, to conserve system resources.

#### Properties and events

**New properties:**

| Property           | Type   | Default value           | Meaning  |
|--------------------|--------|-------------------------|--|
| ■ <b>DataField</b> | String | <i>" (empty string)</i> | Identifies the field from which the control displays data. |

| Property            | Type        | Default value | Meaning   |
|---------------------|-------------|---------------|---|
| ■ <b>DataSource</b> | TDataSource | <i>nil</i>    | Links the control to a dataset.                                       |
| <b>Field</b>        | TField      |               | Returns the TField object whose current value the control represents. |
| ■ <b>ReadOnly</b>   | Boolean     | <i>False</i>  | Determines whether the user can change the value of the field.        |

### 3.3.6 TSRVDBText

**TSRVDBText** represents a data-aware control that displays the value of a field in a document.

**Unit** SRVDBLabel;

#### Syntax

```
TSRVLabel = class (TSRVLabel358) ;
```

#### Hierarchy

TObject  
 TPersistent  
 TComponent  
 TControl  
 TGraphicControl  
 TCustomSRVGraphicControl<sup>329</sup>  
 TSRVGraphicControl<sup>339</sup>  
 TSRVLabel<sup>358</sup>

#### Description

Use TSRVDBText to display the contents of a field in the current record of a dataset on a form. Field values displayed by database text controls cannot be modified by the user using the text control. To allow the user to edit the field value, use TSRVDBEdit<sup>371</sup> or TSRVDBMemo<sup>373</sup> instead.

If the application does not require the data-aware capabilities of TSRVDBText, use the label component (TSRVLabel<sup>358</sup>) instead to conserve system resources.

#### Properties and events

##### New properties:

| Property            | Type        | Default value           | Meaning   |
|---------------------|-------------|-------------------------|---|
| ■ <b>DataField</b>  | String      | <i>" (empty string)</i> | Identifies the field from which the control displays data.            |
| ■ <b>DataSource</b> | TDataSource | <i>nil</i>              | Links the control to a dataset.                                       |
| <b>Field</b>        | TField      |                         | Returns the TField object whose current value the control represents. |

## 3.4 Classes of properties

- `TSRVComboBoxItemCollection`<sup>(375)</sup> – collection of `TSRVComboBoxItem`<sup>(375)</sup> items (used in `TSRVComboBox`<sup>(346)</sup>)
- `TSRVListBoxItemCollection`<sup>(375)</sup> – collection of `TSRVListBoxItem`<sup>(375)</sup> items (used in `TSRVListBox`<sup>(359)</sup>)
- `TSRVImageScrollCollection`<sup>(375)</sup> – collection of `TSRVImageScrollItem`<sup>(379)</sup> items (used in `TSRVImageScroll`<sup>(354)</sup>)

### 3.4.1 TSRVComboBoxItemCollection, TSRVListBoxItemCollection

`TSRVComboBoxItemCollection` is a type of collection of items of `TSRVComboBox`<sup>(346)</sup>. Item type is `TSRVComboBoxItem`<sup>(375)</sup>.

`TSRVListBoxItemCollection` is a type of collection of items of `TSRVListBox`<sup>(359)</sup>. Item type is `TSRVListBoxItem`<sup>(375)</sup>.

**Unit** `SRVTypeItems`;

#### Syntax

```
TSRVComboBoxItemCollection = class (TCollection);
TSRVListBoxItemCollection = class (TSRVComboBoxItemCollection);
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TCollection*

### 3.4.2 TSRVImageScrollCollection

`TSRVImageScrollCollection` is a type of collection of items of `TSRVImageScroll`<sup>(354)</sup>. Item type is `TSRVImageScrollItem`<sup>(379)</sup>.

**Unit** `SRVImageScroll`;

#### Syntax

```
TSRVImageScrollCollection = class (TCollection);
```

#### Hierarchy

*TObject*  
*TPersistent*  
*TCollection*

### 3.4.3 TSRVComboBoxItem, TSRVListBoxItem

`TSRVComboBoxItem` is a type of item for collection of items<sup>(375)</sup> of `TSRVComboBox`<sup>(346)</sup>.

`TSRVListBoxItem` is a type of item for collection of items<sup>(375)</sup> of `TSRVListBox`<sup>(359)</sup>.

**Unit** `SRVTypeItems`;

#### Syntax

```
TSRVComboBoxItem = class (TCollectionItem);
```

```
TSRVLBoxItem = class (TSRVBoxItem) ;
```

## Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*

## Properties of TSRVBoxItem

### Published properties

| Property            | Type             | Default value         | Meaning  |
|---------------------|------------------|-----------------------|--|
| ■ <b>AlignImage</b> | TLRAlignment     | <i>tlrLeftJustify</i> | Specifies the position of image relative to <b>Caption</b> .<br>RTL <b>BiDiMode</b> inverts value of this property.  |
| ■ <b>Alignment</b>  | TAlignment       | <i>taLeftJustify</i>  | Specifies the horizontal alignment of content inside the item.<br>RTL <b>BiDiMode</b> inverts value of this property.  |
| ■ <b>BackColor</b>  | TColor           | <i>clWindow</i>       | Background color.  |
| ■ <b>Caption</b>    | TRVUnicodeString |                       | Specifies the text value of the item. Can be hidden, see <b>Options</b> .  |
| ■ <b>Color</b>      | TColor           | <i>clWindowText</i>   | Text color for <b>Caption</b> .  |
| ■ <b>Enabled</b>    | Boolean          | <i>True</i>           | Controls whether the item can be selected.   |
| ■ <b>Expanded</b>   | Boolean          | <i>True</i>           | Specifies whether the item is expanded (children are visible)  |
| ■ <b>FontName</b>   | TFontName        | 'Tahoma'              | Font name for displaying <b>Caption</b> . Used only if <b>UseItemFontNames</b> = <i>True</i> for the parent control.   |
| ■ <b>Hint</b>       | String           |                       | Reserved   |
| ■ <b>ImageIndex</b> | Integer          | -1                    | Identifies the image associated with this item. <b>ImageIndex</b> is an index into the <b>ImageList</b> property of the parent control. When the item is selected, the item displays the image specified by the <b>SelectedImageIndex</b> property instead, if that property is assigned.<br>Can be hidden, see <b>Options</b> . |
| ■ <b>Indent</b>     | TRVPixel96Length | 0                     | Defines both the indent from the left side to the item content, and the level of the item in the items hierarchy. Items with greater <b>Indent</b> are children of the preceding item with smaller <b>Indent</b> .   |

| Property                    | Type            | Default value   | Meaning  |
|-----------------------------|-----------------|---|--|
| ■ <b>Options</b>            | TSRVItemOptions | [ <i>srvioShowCheckBox</i> ,<br><i>srvioShowImage</i> ,<br><i>srvioShowText</i> ,<br><i>srvioCanSelect</i> ,<br><i>srvioCanHaveChildren</i> ] | Item options (see below)   |
| ■ <b>SkinSchemeIndex</b>    | Integer         | -1  | Index in the <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> .<br><b>BoxSchemes</b> <sup>(232)</sup> (SkinManager is a property of the parent control). Defines a visual appearance of the item. If equals to -1, <b>ItemSkinSchemeIndex</b> property of the parent control is used.   |
| ■ <b>SelectedImageIndex</b> | Integer         | -1  | Identifies the image associated with this item when it is selected.<br><b>SelectedImageIndex</b> is an index into the <b>ImageList</b> property of the parent control. If it equals to -1, <b>ImageIndex</b> property is used instead.<br>Can be hidden, see <b>Options</b> .<br>Can be also used for highlighted (under the mouse pointer) items, if see <b>UseSelectedImagesInHotState</b> of <b>TSRVListBox</b> <sup>(359)</sup> and <b>TSRVComboBox</b> <sup>(346)</sup> . |
| ■ <b>Style</b>              | TFontStyles     | []  | Font styles to display <b>Caption</b> .  |
| ■ <b>Tag</b>                | Integer         | 0   | Stores an integer value as part of the item.   |

**type**

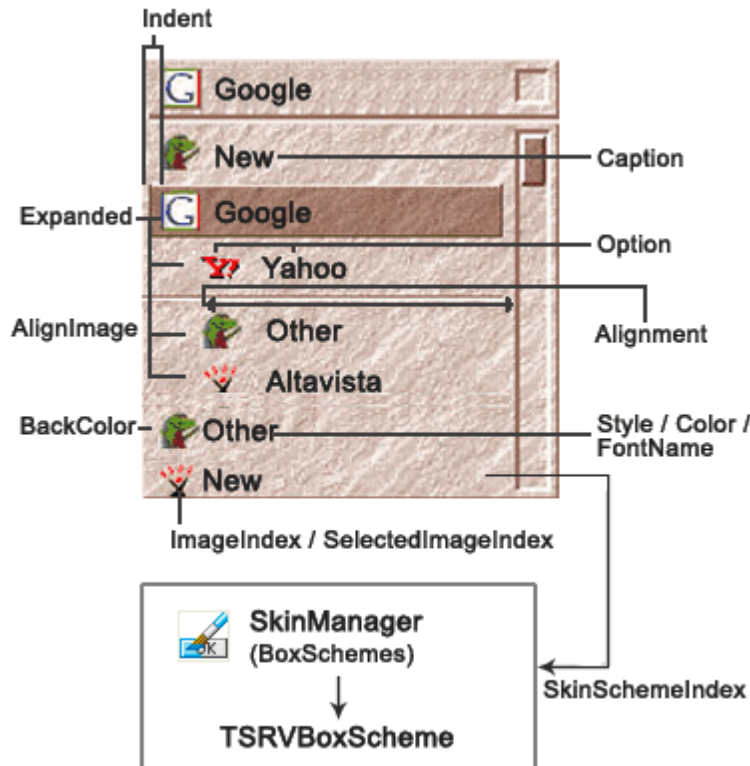
```

TLRAlignment = (tlraLeftJustify, tlraRightJustify);
TSRVItemOption = (srvioShowCheckBox, srvioShowImage, srvioShowText,
    srvioCanSelect, srvioCanHaveChildren);
TSRVItemOptions = set of TSRVItemOption;

```

**Options** can include the following values:

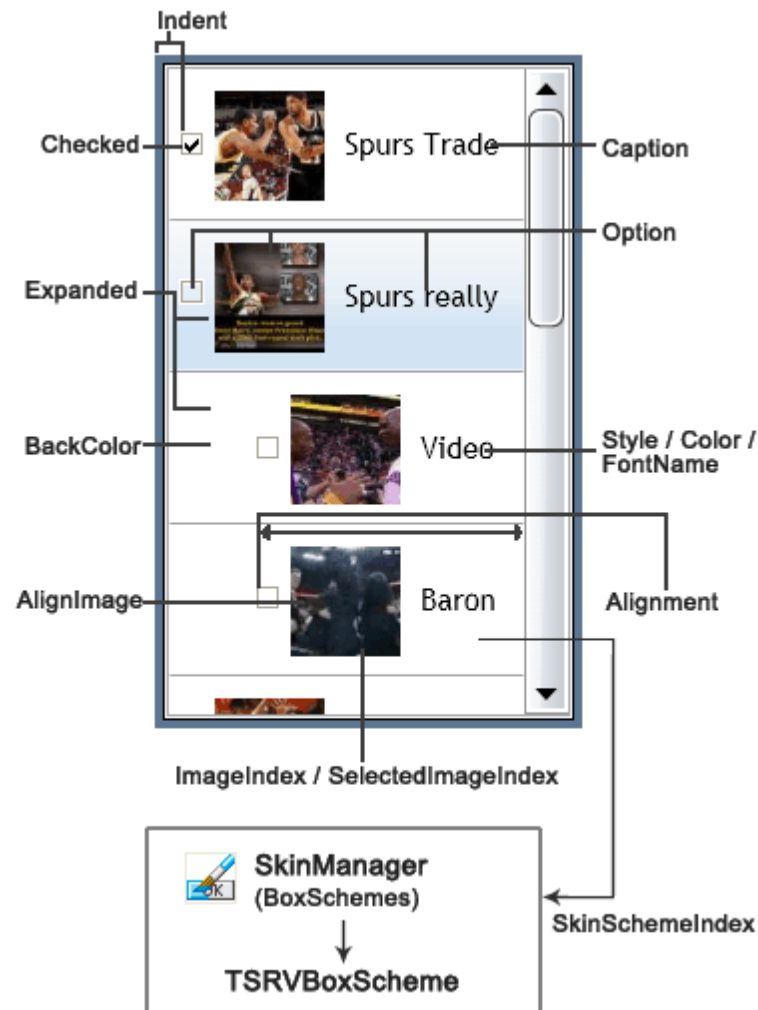
- *srvioShowCheckBox*: if excluded, a check box for this item is not shown (check boxes are shown only if **ShowCheckBoxes** property of the parent control is *True*); this option is ignored for combo boxes, it is used only in list boxes;
- *srvioShowImage*: shows the image specified in **ImageIndex** and **SelectedImageIndex**;
- *srvioShowText*: shows **Caption**;
- *srvioCanSelect*: allows this item to be selected; similar to **Enabled**, but does not provide a visual indication;
- *srvioCanHaveChildren*: allows this item to have children; if excluded, subsequent items are not treated as its children, even if they have greater **Indents**;



## Properties of TSRVLBoxItem

In addition to the properties inherited from `TSRVBoxItem`, `TSRVLBoxItem` has the following published properties:

| Property               | Type                        | Default value         | Meaning  |
|------------------------|-----------------------------|-----------------------|--|
| ■ <b>AlignCheckBox</b> | <code>TLRAlignment</code>   | <i>tlrLeftJustify</i> | Specifies the position of checkbox relative to <b>Caption</b> and image.<br>RTL <b>BiDiMode</b> inverts value of this property.    |
| ■ <b>Checked</b>       | Boolean                     | <i>False</i>          | Specifies whether the check box is checked. See also <b>State</b> property.  |
| ■ <b>State</b>         | <code>TCheckBoxState</code> | <i>cbUnchecked</i>    | Indicates whether the check box is selected ( <i>cbChecked</i> ), deselected ( <i>cbUnchecked</i> ), or grayed ( <i>cbGrayed</i> ) |



### 3.4.4 TSRVImageScrollItem

TSRVImageScrollItem is a type of item for collection of items <sup>(375)</sup> of TSRVImagesScroll <sup>(354)</sup>.

**Unit** SRVImagesScroll;

#### Syntax

```
TSRVImageScrollItem = class(TCollectionItem);
```

#### Hierarchy

*TObject*

*TPersistent*

*TCollectionItem*

#### Properties

#### Published properties

| Property                    | Type             | Default value | Meaning   |
|-----------------------------|------------------|---------------|---|
| ■ <b>Enabled</b>            | Boolean          | <i>True</i>   | Determines whether the item can be selected.  |
| ■ <b>Hint</b>               | String           |               | Reserved  |
| ■ <b>ImageIndex</b>         | Integer          | -1            | Identifies the image associated with this item. <b>ImageIndex</b> is an index into the <b>ImageList</b> property of the parent control. When the item is selected, the item displays the image specified by the <b>SelectedImageIndex</b> property instead, if that property is assigned. |
| ■ <b>Link</b>               | TRVUnicodeString |               | Reserved  |
| ■ <b>Name</b>               | TRVUnicodeString |               | Item name, not displayed  |
| ■ <b>SkinSchemeIndex</b>    | Integer          | -1            | Index in the <b>SkinManager.CurrentSkin</b> <sup>(230)</sup> . <b>BoxSchemes</b> <sup>(232)</sup> ( <b>SkinManager</b> is a property of the parent control). Defines a visual appearance of the item. If equals to -1, <b>ItemSkinSchemeIndex</b> of the parent control is used.          |
| ■ <b>SelectedImageIndex</b> | Integer          | -1            | Identifies the image associated with this item when it is selected. <b>SelectedImageIndex</b> is an index into the <b>ImageList</b> property of the parent control. If it equals to -1, <b>ImageIndex</b> property is used instead.   |
| ■ <b>ShowHint</b>           | Boolean          | <i>True</i>   | Reserved  |
| ■ <b>Tag</b>                | LongInt          | 0             | Stores an integer value as part of the item.  |
| ■ <b>Visible</b>            | Boolean          | <i>True</i>   | Determines whether the item is visible  |



# Index

## - A -

AcceptDragDropFormats 47  
     TSRichViewEdit 47  
 AcceptPasteFormats 47  
     TSRichViewEdit 47  
 ActionSave 286  
     TsrvCustomActionThatNeedsDocProps 286  
 Active 152  
     TSRVReadModeProperty 152  
 ActiveEditor 47  
     TSRichViewEdit 47  
 Align 149, 150  
     TSRVPropertyTBH 149  
     TSRVPropertyTBV 150  
 AlignButton 128, 191  
     TSRVCustomPropertyTB 128  
     TSRVToolBar 191  
 AlignPageH 144  
     TSRVPagePositionProperty 144  
 AlignPageV 145  
     TSRVPagePositionProperty 145  
 AlignText 176, 191, 198  
     TSRVPageScroll 176  
     TSRVToolBar 191  
     TSRVToolWindow 198  
 AllowAllUp 273  
     TSRVToolButton 273  
 AllowCloseLastTab 263  
     TSRVCloseButton 263  
 AnimationMode 48  
     TSRichViewEdit 48  
 AutoDarkMode 253  
     TSRVTabSet 253  
 AutoDisplay 173  
     TDBSRichViewEdit 173  
 AutoScrolled 177  
     TSRVPageScroll 177  
 AutoSize 192  
     TSRVToolBar 192  
 AutoTabWidth 252  
     TSRVTabSet 252  
 AutoUpdate 177, 207  
     TSRVPageScroll 177  
     TSRVPrint 207

AutoVScrollBarPosition 156  
     TSRVViewProperty 156  
 AutoWidth 135  
     TSRVPageProperty 135

## - B -

BackgroundProperty 48  
     TSRichViewEdit 48  
 BackgroundTextColor 177  
     TSRVPageScroll 177  
 BeginComponentsUpdate 72  
     TSRichViewEdit 72  
 BeginUpdate 183, 218  
     TSRVPageScroll 183  
     TSRVPrint 218  
 BiDiMode 48  
     TSRichViewEdit 48  
 BorderColor 225  
     TSRVPreview 225  
 BorderPen 135  
     TSRVPageProperty 135  
 BorderWidth 128, 192, 199, 225  
     TSRVCustomPropertyTB 128  
     TSRVPreview 225  
     TSRVToolBar 192  
     TSRVToolWindow 199  
 BottomMargin 136  
     TSRVPageProperty 136  
 BottomMargin100Pix 48  
     TSRichViewEdit 48  
 BottomMarginPix 48  
     TSRichViewEdit 48  
 BoxPosition 318  
     TsrvActionCustomInsertSidenote 318  
 BoxProperties 318  
     TsrvActionCustomInsertSidenote 318  
 BoxSchemes 232  
     TSRVSkin 232  
 ButtonCol 192, 199  
     TSRVToolBar 192  
     TSRVToolWindow 199  
 ButtonHeight 128, 192, 199  
     TSRVCustomPropertyTB 128  
     TSRVToolBar 192  
     TSRVToolWindow 199  
 Buttons 129, 192, 199  
     TSRVCustomPropertyTB 129  
     TSRVToolBar 192  
     TSRVToolWindow 199

ButtonWidth 129, 193, 199  
 TSRVCustomPropertyTB 129  
 TSRVToolBar 193  
 TSRVToolWindow 199

## - C -

CachedPagesCount 178  
 TSRVPageScroll 178

CacheScrollLimit 178  
 TSRVPageScroll 178

CalculateAllPagePositions 72  
 TSRichViewEdit 72

CalculatePageCount 72  
 TSRichViewEdit 72

CalculatePagePosition 73  
 TSRichViewEdit 73

CanMoveTabs 253  
 TSRVTabSet 253

CanScroll 49

CanUpdate 49  
 TSRichViewEdit 49

CanUpdateMargin 49  
 TSRichViewEdit 49

CanUpdatePosition 49  
 TSRichViewEdit 49

Caption 273  
 TSRVToolButton 273

CaretBlinkTime 49  
 TSRichViewEdit 49

CaretPen 136  
 TSRVPageProperty 136

CaretPos 49  
 TSRichViewEdit 49

CaretVisible 50  
 TSRichViewEdit 50

Center 207  
 TSRVPrint 207

ChangeDelay 178  
 TSRVPageScroll 178

CheckMargin 218  
 TSRVPrint 218

CheckMargins 188  
 TScIRVRuler 188

Clear 73  
 TSRichViewEdit 73

ClipMargins 208  
 TSRVPrint 208

CloseButton 253  
 TSRVTabSet 253

Color 129, 193, 200, 303, 306, 309  
 TsrvActionLayoutPrint 303  
 TsrvActionLayoutRead 306  
 TsrvActionLayoutSideToSide 309  
 TSRVCustomPropertyTB 129  
 TSRVToolBar 193  
 TSRVToolWindow 200

ColorBegin 124  
 TSRVBackgroundProperty 124

ColorDisable 129, 193, 200  
 TSRVCustomPropertyTB 129  
 TSRVToolBar 193  
 TSRVToolWindow 200

ColorDown 130, 193, 200  
 TSRVCustomPropertyTB 130  
 TSRVToolBar 193  
 TSRVToolWindow 200

ColorEnd 124  
 TSRVBackgroundProperty 124

ColorMiddle 124  
 TSRVBackgroundProperty 124

ColorSelect 130, 194, 201  
 TSRVCustomPropertyTB 130  
 TSRVToolBar 194  
 TSRVToolWindow 201

Columns 178  
 TSRVPageScroll 178

Control 285  
 TsrvAction 285

ConvertPageFormatToPageSize 143  
 TSRVPageProperty 143

ConvertPageSizeToPageFormat 143  
 TSRVPageProperty 143

ConvertRVtoSRV 73  
 TSRichViewEdit 73

ConvertSRVtoRV 74  
 TSRichViewEdit 74

ConvertToDifferentUnits 75  
 TSRichViewEdit 75

ConvertToEMU 75  
 TSRichViewEdit 75

ConvertToPixels 75  
 TSRichViewEdit 75

ConvertToTwips 75  
 TSRichViewEdit 75

Copy 75  
 TSRichViewEdit 75

CPEventKind 50  
 TSRichViewEdit 50

Create 75

Create 75  
     TSRichViewEdit 75  
 CurControl 50  
     TSRichViewEdit 50  
 CurrentNote 50  
     TSRichViewEdit 50  
 CurrentNoteParentEditor 50  
     TSRichViewEdit 50  
 CurrentPage 51  
     TSRichViewEdit 51  
 CurrentSkin 230  
     TSRVSkinManager 230  
 Cut 75  
     TSRichViewEdit 75

## - D -

DarkMode 157, 253  
     TSRVTabSet 253  
     TSRVViewProperty 157  
 DarkModeUI 157  
     TSRVViewProperty 157  
 DataField 173  
     TDBSRichViewEdit 173  
 DataSource 173  
     TDBSRichViewEdit 173  
 DecButtonWidth 245  
     TSRVScrollBar 245  
 DeletePage 76  
     TSRichViewEdit 76  
 Destroy 76  
     TSRichViewEdit 76  
 DistanceFromText 132  
     TSRVLineNumberProperty 132  
 Document 51  
     TSRichViewEdit 51  
 Document parts in TSRichViewEdit 37  
 Down 273  
     TSRVToolButton 273  
 DrawMetafile 76  
     TSRichViewEdit 76  
 DrawPage 77  
     TSRichViewEdit 77

## - E -

Elements 232  
     TSRVSkin 232  
 Enabled 265, 273  
     TSRVTabItem 265

TSRVToolButton 273  
 EndComponentsUpdate 77  
     TSRichViewEdit 77  
 EndUpdate 183, 219  
     TSRVPageScroll 183  
     TSRVPrint 219  
 Execute 203  
     TSRVToolWindow 203  
 ExternalRV 174  
     TDBSRichViewEdit 174  
 ExternalRVStyle 51  
     TSRichViewEdit 51  
 ExternalRVStyleFooter 51  
     TSRichViewEdit 51  
 ExternalRVStyleHeader 51  
     TSRichViewEdit 51

## - F -

FacingPages 136  
     TSRVPageProperty 136  
 FieldFormat 174  
     TDBSRichViewEdit 174  
 FillType 125  
     TSRVBackgroundProperty 125  
 FindNextCheckpoint 78  
     TSRichViewEdit 78  
 FindNextHeading 78  
     TSRichViewEdit 78  
 FindNextHyperlink 79  
     TSRichViewEdit 79  
 FindNextItem 79  
     TSRichViewEdit 79  
 FindPriorCheckpoint 80  
     TSRichViewEdit 80  
 FindPriorHeading 80  
     TSRichViewEdit 80  
 FindPriorHyperlink 81  
     TSRichViewEdit 81  
 FindPriorItem 81  
     TSRichViewEdit 81  
 First 184, 227  
     TSRVPageScroll 184  
     TSRVPreview 227  
 FirstCurPage 82  
     TSRichViewEdit 82  
 FirstPageNo 52  
     TSRichViewEdit 52  
 FirstTabIndex 253  
     TSRVTabSet 253

FixMarginsMode 52  
     TSRichViewEdit 52  
 Flat 245  
     TSRVScrollBar 245  
 FntTab 241  
     TSRVTabSetSkinScheme 241  
 FocusBlinkDelay 245  
     TSRVScrollBar 245  
 Font 132, 322  
     TsrvActionInsertNote 322  
     TSRVLineNumberProperty 132  
 Fonts 232  
     TSRVSkin 232  
 FooterPen 157  
     TSRVViewProperty 157  
 FooterTitleColor 157  
     TSRVViewProperty 157  
 FooterTitleFont 158  
     TSRVViewProperty 158  
 FooterVisible 136, 303, 309  
     TsrvActionLayoutPrint 303  
     TsrvActionLayoutSideToSide 309  
     TSRVPageProperty 136  
 FooterY 137  
     TSRVPageProperty 137  
 Format 82  
     TSRichViewEdit 82  
 FrameBorderColor 225  
     TSRVPreview 225  
 FrameBorderWidth 225  
     TSRVPreview 225  
 FrameCountX 208  
     TSRVPrint 208  
 FrameCountY 204, 209  
     TSRVPrint 209  
 FrameHeightPix 209  
     TSRVPrint 209  
 FrameWidthPix 210  
     TSRVPrint 210  
 FreePosPage 158  
     TSRVViewProperty 158

## - G -

GetCaretPosInUnits 83  
     TSRichViewEdit 83  
 GetCurrentLineCol 83  
     TSRichViewEdit 83  
 GetFirstItemVisibleRV 83  
     TSRichViewEdit 83

GetFirstVisiblePage 84  
     TSRichViewEdit 84  
 GetItemAt 84  
     TSRichViewEdit 84  
 GetItemBounds 84  
     TSRichViewEdit 84  
 GetItemBounds100 84  
     TSRichViewEdit 84  
 GetItemCoords100 85  
     TSRichViewEdit 85  
 GetItemPages 86  
     TSRichViewEdit 86  
 GetLastItemVisibleRV 86  
     TSRichViewEdit 86  
 GetLastVisiblePage 87  
     TSRichViewEdit 87  
 GetLeftMargin100Pix 54  
     TSRichViewEdit 54  
 GetLeftMarginPix 54  
     TSRichViewEdit 54  
 GetPageAt 87, 184  
     TSRichViewEdit 87  
     TSRVPageScroll 184  
 GetPageClientRect 87  
     TSRichViewEdit 87  
 GetPageLastItemNo 87  
     TSRichViewEdit 87  
 GetPageNo 88  
     TSRichViewEdit 88  
 GetPageProperties 188  
     TScIRVRuler 188  
 GetPageStartItemNo 88  
     TSRichViewEdit 88  
 GetPageStartTableItem 88  
     TSRichViewEdit 88  
 GetPageStartTableRow 89  
     TSRichViewEdit 89  
 GetRightMargin100Pix 60  
     TSRichViewEdit 60  
 GetRightMarginPix 60  
     TSRichViewEdit 60  
 GetTableconItem 89  
     TSRichViewEdit 89  
 GetTableconRVData 89  
     TSRichViewEdit 89  
 GlobalPageBackgroundColor 125  
     TSRVBackgroundProperty 125  
 GroupIndex 274  
     TSRVToolButton 274

# - H -

- HeaderFooterShade 158
  - TSRVViewProperty 158
- HeaderPen 159
  - TSRVViewProperty 159
- HeaderTitleColor 159
  - TSRVViewProperty 159
- HeaderTitleFont 159
  - TSRVViewProperty 159
- HeaderVisible 137, 303, 309
  - TsrvActionLayoutPrint 303
  - TsrvActionLayoutSideToSide 309
  - TSRVPageProperty 137
- HeaderY 137
  - TSRVPageProperty 137
- Height 263
  - TSRVCloseButton 263
- HiddenDistances 145
  - TSRVPagePositionProperty 145
- HidePageBackgroundImage 152, 306
  - TsrvActionLayoutRead 306
  - TSRVReadModeProperty 152
- Hint 273
  - TSRVToolButton 273
- HintCancel 194, 201
  - TSRVToolBar 194
  - TSRVToolWindow 201
- HintFont 160, 194, 201
  - TSRVToolBar 194
  - TSRVToolWindow 201
  - TSRVViewProperty 160
- HintHeight 194, 201
  - TSRVToolBar 194
  - TSRVToolWindow 201
- HintPrefixText 160
  - TSRVViewProperty 160
- HMaxScrollPos 52
  - TSRichViewEdit 52
- HorizontalAlign 263
  - TSRVCloseButton 263
- HorizontalScrollBarSchemes 232
  - TSRVSkin 232
- HorizontalTabSetSchemes 232
  - TSRVSkin 232
- HPadding 146, 301, 304, 306, 309
  - TsrvActionLayoutDraft 301
  - TsrvActionLayoutPrint 304
  - TsrvActionLayoutRead 306
- TsrvActionLayoutSideToSide 309
- TSRVPagePositionProperty 146
- HScrollBar 53
  - TSRichViewEdit 53
- HScrollBarSchemeIndex 53
  - TSRichViewEdit 53
- HScrollPos 53
  - TSRichViewEdit 53
- HTMLReadProperties 53
  - TSRichViewEdit 53
- HTMLSaveProperties 53
  - TSRichViewEdit 53

# - I -

- IconStyle 160
  - TSRVViewProperty 160
- IgnoreEscape 174
  - TDBSRichViewEdit 174
- ImageIndex 265
  - TSRVTabItem 265
- ImageIndexDisable 274
  - TSRVToolButton 274
- ImageIndexDown 274
  - TSRVToolButton 274
- ImageIndexMove 274
  - TSRVToolButton 274
- ImageIndexNormal 275
  - TSRVToolButton 275
- ImageList 130, 195, 202, 253
  - TSRVCustomPropertyTB 130
  - TSRVTabSet 253
  - TSRVToolBar 195
  - TSRVToolWindow 202
- ImgAddTabButton 241
  - TSRVTabSetSkinScheme 241
- ImgBackgroundBegin 241
  - TSRVTabSetSkinScheme 241
- ImgBackgroundCenter 241
  - TSRVTabSetSkinScheme 241
- ImgBackgroundEnd 241
  - TSRVTabSetSkinScheme 241
- ImgBorderBackground 241
  - TSRVTabSetSkinScheme 241
- ImgBorderBegin 241
  - TSRVTabSetSkinScheme 241
- ImgBorderEnd 241
  - TSRVTabSetSkinScheme 241
- ImgCloseButton 241
  - TSRVTabSetSkinScheme 241

ImgDecButton 241  
     TSRVTabSetSkinScheme 241  
 ImgIncButton 241  
     TSRVTabSetSkinScheme 241  
 ImgTabBackground 241  
     TSRVTabSetSkinScheme 241  
 ImgTabBegin 241  
     TSRVTabSetSkinScheme 241  
 ImgTabEnd 241  
     TSRVTabSetSkinScheme 241  
 IncButtonWidth 246  
     TSRVScrollBar 246  
 Indent 130, 195, 254  
     TSRVCustomPropertyTB 130  
     TSRVTabSet 254  
     TSRVToolBar 195  
 IndentX 263  
     TSRVCloseButton 263  
 IndentY 263  
     TSRVCloseButton 263  
 InsertPageBreak 89  
     TSRichViewEdit 89  
 IntegerCount 210  
     TSRVPrint 210  
 InZoomRect 90  
     TSRichViewEdit 90

## - K -

Kind 246, 254  
     TSRVScrollBar 246  
     TSRVTabSet 254

## - L -

LargeChange 247  
     TSRVScrollBar 247  
 Last 184, 227  
     TSRVPageScroll 184  
     TSRVPreview 227  
 LastCurPage 90  
     TSRichViewEdit 90  
 LastPageNo 54  
     TSRichViewEdit 54  
 LastTabIndex 254  
     TSRVTabSet 254  
 LeftMargin 138  
     TSRVPageProperty 138  
 LeftMargin100Pix 54  
     TSRichViewEdit 54

LeftMarginPix 54  
     TSRichViewEdit 54  
 LineNumberProperty 54  
     TSRichViewEdit 54  
 LoadRVF 90  
     TSRichViewEdit 90  
 LoadRVFFromStream 90  
     TSRichViewEdit 90

## - M -

MainDocumentShade 160  
     TSRVViewProperty 160  
 MainPen 161  
     TSRVViewProperty 161  
 MainTitleColor 161  
     TSRVViewProperty 161  
 MainTitleFont 161  
     TSRVViewProperty 161  
 MarginHorizontal 179  
     TSRVPageScroll 179  
 Margins 254  
     TSRVTabSet 254  
 MarginsPen 162  
     TSRVViewProperty 162  
 MarginsRectVisible 162  
     TSRVViewProperty 162  
 MarginVertical 179  
     TSRVPageScroll 179  
 MarkdownProperties 54  
     TSRichViewEdit 54  
 MaxPageColCount 147, 152  
     TSRVPagePositionProperty 147  
     TSRVReadModeProperty 152  
 MaxPrintedItemNo 55  
     TSRichViewEdit 55  
 MaxPrintedOffsInItem 55  
     TSRichViewEdit 55  
 MaxTabSize 255  
     TSRVTabSet 255  
 MenuHButtons 55  
     TSRichViewEdit 55  
 MenuHorizontal 55  
     TSRichViewEdit 55  
 MenuVButtons 56  
     TSRichViewEdit 56  
 MenuVertical 56  
     TSRichViewEdit 56  
 MetafileCompatibility 211  
     TSRVPrint 211

MinPageColCount 152  
     TSRVReadModeProperty 152  
 MinPrintedItemNo 56  
     TSRichViewEdit 56  
 MinPrintedOffsInItem 56  
     TSRichViewEdit 56  
 MinTabSize 255  
     TSRVTabSet 255  
 MirrorMargins 138  
     TSRVPageProperty 138  
 MirrorText 255  
     TSRVTabSet 255  
 MouseWheelZoom 162  
     TSRVViewProperty 162  
 MouseWheelZoomMax 281  
 MouseWheelZoomMin 281  
 MouseWheelZoomSpeed 281

## - N -

Name 233, 241  
     TSRVSkin 233  
     TSRVTabSetSkinScheme 241  
 Next 184, 227  
     TSRVPageScroll 184  
     TSRVPreview 227  
 NextCurCheckpoint 90  
     TSRichViewEdit 90  
 NextCurHeading 91  
     TSRichViewEdit 91  
 NextCurHyperlink 91  
     TSRichViewEdit 91  
 NextCurItem 91  
     TSRichViewEdit 91  
 NextCurPage 91  
     TSRichViewEdit 91  
 NoMetafiles 211  
     TSRVPrint 211  
 NoPageSpacing 309  
     TsrvActionLayoutSideToSide 309  
 Notes and text boxes in TSRichViewEdit 38

## - O -

OffsetX 57  
     TSRichViewEdit 57  
 OffsetY 57  
     TSRichViewEdit 57  
 OffXPix 211  
     TSRVPrint 211

OffYPix 212  
     TSRVPrint 212  
 OnAfterMarginChanged 189  
     TSclRVRuler 189  
 OnAfterOleDrop 104  
     TSRichViewEdit 104  
 OnAfterPaint 228  
     TSRVPreview 228  
 OnAssignImageFileName 103  
     TSRichViewEdit 103  
 OnBeforeOleDrop 104  
     TSRichViewEdit 104  
 OnBeforePaint 228  
     TSRVPreview 228  
 OnCaretGetOut 104  
     TSRichViewEdit 104  
 OnCaretMove 104  
     TSRichViewEdit 104  
 OnCaretMoving 104  
     TSRichViewEdit 104  
 OnChange 104, 260, 293  
     TSRichViewEdit 104  
     TsrvActionPageSetup 293  
     TSRVTabSet 260  
 OnChangeActiveEditor 104  
     TSRichViewEdit 104  
 OnChangeCurrentControl 105  
     TSRichViewEdit 105  
 OnChangeViewModeAfter 105  
     TSRichViewEdit 105  
 OnChangeViewModeBefore 105  
     TSRichViewEdit 105  
 OnChanging 105  
     TSRichViewEdit 105  
 OnCheckControl 105  
     TSRichViewEdit 105  
 OnCheckpointVisible 106  
     TSRichViewEdit 106  
 OnCheckStickingItems 106  
     TSRichViewEdit 106  
 OnClickButton 196, 203  
     TSRVToolBar 196  
     TSRVToolWindow 203  
 OnClickPage 106  
     TSRichViewEdit 106  
 OnCloseTab 260  
     TSRVTabSet 260  
 OnContextPopup 106  
     TSRichViewEdit 106  
 OnControlAction 106

|                        |          |                        |          |
|------------------------|----------|------------------------|----------|
| OnControlAction        | 106      | TsrvActionCustomLayout | 298      |
| TSRichViewEdit         | 106      | TsrvActionThumbnails   | 312      |
| OnCopy                 | 106      | OnFullRedraw           | 108      |
| TSRichViewEdit         | 106      | TSRichViewEdit         | 108      |
| OnCurParaStyleChanged  | 107      | OnGetPagePos           | 108      |
| TSRichViewEdit         | 107      | TSRichViewEdit         | 108      |
| OnCurrentPageChange    | 107      | OnHMenuClickButton     | 109      |
| TSRichViewEdit         | 107      | TSRichViewEdit         | 109      |
| OnCurTextStyleChanged  | 107      | OnHMenuEnterButton     | 109      |
| TSRichViewEdit         | 107      | TSRichViewEdit         | 109      |
| OnDrawArea             | 249      | OnHScrolled            | 109      |
| TSRVScrollBar          | 249      | TSRichViewEdit         | 109      |
| OnDrawBackground       | 260      | OnHTMLSaveImage        | 110      |
| TSRVTabSet             | 260      | TSRichViewEdit         | 110      |
| OnDrawBorder           | 249, 260 | OnImportPicture        | 110      |
| TSRVScrollBar          | 249      | TSRichViewEdit         | 110      |
| TSRVTabSet             | 260      | OnItemAction           | 110      |
| OnDrawButtonBackground | 196      | TSRichViewEdit         | 110      |
| TSRVToolBar            | 196      | OnItemHint             | 110      |
| OnDrawCloseButton      | 260      | TSRichViewEdit         | 110      |
| TSRVTabSet             | 260      | OnItemResize           | 110      |
| OnDrawDecButton        | 249, 260 | TSRichViewEdit         | 110      |
| TSRVScrollBar          | 249      | OnItemTextEdit         | 110      |
| TSRVTabSet             | 260      | TSRichViewEdit         | 110      |
| OnDrawHyperlink        | 107      | OnJump                 | 111      |
| TSRichViewEdit         | 107      | TSRichViewEdit         | 111      |
| OnDrawIncButton        | 249, 260 | OnLeave                | 197, 204 |
| TSRVScrollBar          | 249      | TSRVToolBar            | 197      |
| TSRVTabSet             | 260      | TSRVToolWindow         | 204      |
| OnDrawPageBorder       | 186      | OnLoadCustomFormat     | 111      |
| TSRVPageScroll         | 186      | TSRichViewEdit         | 111      |
| OnDrawPageNumber       | 186      | OnLoadDocument         | 111      |
| TSRVPageScroll         | 186      | TSRichViewEdit         | 111      |
| OnDrawSelectedPage     | 187      | OnMarginsChanged       | 111      |
| TSRVPageScroll         | 187      | TSRichViewEdit         | 111      |
| OnDrawTab              | 260      | OnMessageControl       | 112      |
| TSRVTabSet             | 260      | TSRichViewEdit         | 112      |
| OnDrawThumbTab         | 249      | OnMoveTab              | 261      |
| TSRVScrollBar          | 249      | TSRVTabSet             | 261      |
| OnDropFile             | 108      | OnNewDocument          | 113      |
| TSRichViewEdit         | 108      | TSRichViewEdit         | 113      |
| OnDropFiles            | 108      | OnOleDragEnter         | 113      |
| TSRichViewEdit         | 108      | TSRichViewEdit         | 113      |
| OnEnter                | 196, 203 | OnOleDragLeave         | 113      |
| TSRVToolBar            | 196      | TSRichViewEdit         | 113      |
| TSRVToolWindow         | 203      | OnOleDragOver          | 113      |
| OnEnterButton          | 197, 203 | TSRichViewEdit         | 113      |
| TSRVToolBar            | 197      | OnOleDrop              | 113      |
| TSRVToolWindow         | 203      | TSRichViewEdit         | 113      |
| OnExecuted             | 298, 312 | OnPageCountChanged     | 114      |



- OnPageCountChanged 114
  - TSRichViewEdit 114
- OnPageFormatChanged 114
  - TSRichViewEdit 114
- OnPageScrolled 114
  - TSRichViewEdit 114
- OnPaint 114
  - TSRichViewEdit 114
- OnPaintComponent 114
  - TSRichViewEdit 114
- OnPaintPage 115
  - TSRichViewEdit 115
- OnParaStyleConversion 116
  - TSRichViewEdit 116
- OnPaste 116
  - TSRichViewEdit 116
- OnPrinting 116
  - TSRichViewEdit 116
- OnProgress 117
  - TSRichViewEdit 117
- OnReadField 117
  - TSRichViewEdit 117
- OnReadHyperlink 117
  - TSRichViewEdit 117
- OnReadMergeField 117
  - TSRichViewEdit 117
- OnResize 117
  - TSRichViewEdit 117
- OnResizeDoc 117
  - TSRichViewEdit 117
- OnResizing 117
  - TSRichViewEdit 117
- OnRVDbIClick 117
  - TSRichViewEdit 117
- OnRVFControlNeeded 118
  - TSRichViewEdit 118
- OnRVFImageListNeeded 118
  - TSRichViewEdit 118
- OnRVFPictureNeeded 118
  - TSRichViewEdit 118
- OnRVMouseDown 118
  - TSRichViewEdit 118
- OnRVMouseMove 118
  - TSRichViewEdit 118
- OnRVMouseUp 118
  - TSRichViewEdit 118
- OnRVRightClick 119
  - TSRichViewEdit 119
- OnSaveComponentToFile 119
  - TSRichViewEdit 119
- OnSaveCustomFormat 119
  - TDBSRichViewEdit 119
- OnSaveDocXExtra 119
  - TSRichViewEdit 119
- OnSaveHTMLExtra 120
  - TSRichViewEdit 120
- OnSaveImage2 120
  - TSRichViewEdit 120
- OnSaveItemToFile 120
  - TSRichViewEdit 120
- OnSaveRTFExtra 120
  - TSRichViewEdit 120
- OnSelect 120
  - TSRichViewEdit 120
- OnSelectEvent 204
  - TSRVToolWindow 204
- OnSendingToPrinter 222
  - TSRVPrint 222
- OnSmartPopupClick 120
  - TSRichViewEdit 120
- OnSpellingCheck 121
  - TSRichViewEdit 121
- OnStyleConversion 121
  - TSRichViewEdit 121
- OnTableIconClick 121
  - TSRichViewEdit 121
- OnTextFound 121
  - TSRichViewEdit 121
- OnUpdateData 222
  - TSRVPrint 222
- OnURLNeeded 121
  - TSRichViewEdit 121
- OnVMenuClickButton 121
  - TSRichViewEdit 121
- OnVMenuEnterButton 122
  - TSRichViewEdit 122
- OnVScrolled 122
  - TSRichViewEdit 122
- OnWriteHyperlink 122
  - TSRichViewEdit 122
- OnWriteObjectProperties 122
  - TSRichViewEdit 122
- OnZoomChanged 122
  - TSRichViewEdit 122
- OnZoomViewChanged 123
  - TSRichViewEdit 123
- OppositeTabPosition 256
  - TSRVTabSet 256
- OptimalOrientation 219
  - TSRVPrint 219

Orientation 138, 212  
 TSRVPageProperty 138  
 TSRVPrint 212  
 OutZoomRect 92  
 TSRichViewEdit 92

## - P -

PageBorderColor 179, 226  
 TSRVPageScroll 179  
 TSRVPreview 226  
 PageBorderStyle 226  
 TSRVPreview 226  
 PageBorderWidth 226  
 TSRVPreview 226  
 PageBreakHeight 179  
 TSRVPageScroll 179  
 PageBreakWidth 180  
 TSRVPageScroll 180  
 PageColCount 213  
 TSRVPrint 213  
 PageColor 153, 306  
 TsrvActionLayoutRead 306  
 TSRVReadModeProperty 153  
 PageCount 57  
 TSRichViewEdit 57  
 PageFlipEffect 153, 306, 310  
 TsrvActionLayoutRead 306  
 TsrvActionLayoutSideToSide 310  
 TSRVReadModeProperty 153  
 PageFormat 139, 213, 290  
 TsrvActionPageFormat 290  
 TSRVPageProperty 139  
 TSRVPrint 213  
 PageFormats1 292  
 TsrvActionPageSetup 292  
 PageFormats2 292  
 TsrvActionPageSetup 292  
 PageFormats3 292  
 TsrvActionPageSetup 292  
 PageFormats4 292  
 TsrvActionPageSetup 292  
 PageFormats5 292  
 TsrvActionPageSetup 292  
 PageFormats6 292  
 TsrvActionPageSetup 292  
 PageHeight 139, 162, 180, 290  
 TsrvActionPageFormat 290  
 TSRVPageProperty 139  
 TSRVPageScroll 180  
 TSRVViewProperty 162  
 PageHeight100Pix 57  
 TSRichViewEdit 57  
 PageHeightPix 57, 213  
 TSRichViewEdit 57  
 TSRVPrint 213  
 PageHSpacing 147  
 TSRVPagePositionProperty 147  
 PageNo 226  
 TSRVPreview 226  
 PageNoFirst 139  
 TSRVPageProperty 139  
 PageNoFont 139, 180  
 TSRVPageProperty 139  
 TSRVPageScroll 180  
 PageNoFromNumber 140  
 TSRVPageProperty 140  
 PageNoHAlign 140  
 TSRVPageProperty 140  
 PageNoMouseDown 58  
 TSRichViewEdit 58  
 PageNoMouseMove 58  
 TSRichViewEdit 58  
 PageNoVAlign 140  
 TSRVPageProperty 140  
 PageNoVisible 140, 181  
 TSRVPageProperty 140  
 TSRVPageScroll 181  
 PagePosProperty 58  
 TSRichViewEdit 58  
 PageProperty 58  
 TSRichViewEdit 58  
 PageProportions 181  
 TSRVPageScroll 181  
 PageRowCount 214  
 TSRVPrint 214  
 PageScroll 312  
 TsrvActionThumbnails 312  
 PageSize 246  
 TSRVScrollBar 246  
 PageViewMode 141  
 TSRVPageProperty 141  
 PageVSpacing 147  
 TSRVPagePositionProperty 147  
 PageWidth 141, 162, 181, 290  
 TsrvActionPageFormat 290  
 TSRVPageProperty 141  
 TSRVPageScroll 181  
 TSRVViewProperty 162  
 PageWidth100Pix 58

PageWidth100Pix 58  
    TSRichViewEdit 58  
PageWidthPix 58, 214  
    TSRichViewEdit 58  
    TSRVPrint 214  
PaperCodeToPageFormat 219  
    TSRVPrint 219  
Paste 92  
    TSRichViewEdit 92  
PenFrame 130, 195, 202  
    TSRVCustomPropertyTB 130  
    TSRVToolBar 195  
    TSRVToolWindow 202  
PercentMiddle 126  
    TSRVBackgroundProperty 126  
Picture 126  
    TSRVBackgroundProperty 126  
PicturePosition 126  
    TSRVBackgroundProperty 126  
Position 246  
    TSRVScrollBar 246  
Prev 185, 227  
    TSRVPageScroll 185  
    TSRVPreview 227  
Print 219  
    TSRVPrint 219  
PrintableAreaPen 141  
    TSRVPageProperty 141  
PrintAll 92  
    TSRichViewEdit 92  
PrintCurrent 93  
    TSRichViewEdit 93  
PrintFrame 220  
    TSRVPrint 220  
PrintFrames 220  
    TSRVPrint 220  
PrintFramesEx 221  
    TSRVPrint 221  
PrintMode 214  
    TSRVPrint 214  
PrintPages 221  
    TSRVPrint 221  
PrintRange 93  
    TSRichViewEdit 93  
PriorCurCheckpoint 94  
    TSRichViewEdit 94  
PriorCurHeading 94  
    TSRichViewEdit 94  
PriorCurHyperlink 94  
    TSRichViewEdit 94

PriorCurlItem 95  
    TSRichViewEdit 95  
PriorCurPage 95  
    TSRichViewEdit 95  
ProcessControls 95  
    TSRichViewEdit 95

## - R -

RangeSearch 58  
    TSRichViewEdit 58  
RangeSearchFirstY 59  
    TSRichViewEdit 59  
RangeSearchLastY 59  
    TSRichViewEdit 59  
ReadModeProperty 59  
    TSRichViewEdit 59  
ReadOnly 59  
    TSRichViewEdit 59  
RealHeight 215  
    TSRVPrint 215  
RealWidth 216  
    TSRVPrint 216  
Reformat 82  
    TSRichViewEdit 82  
RefStyleTemplateName 324  
    TsrvActionInsertSidenote 324  
RestoreCanvasZoom 100  
    TSRichViewEdit 100  
ReturnToNote 95  
    TSRichViewEdit 95  
RichViewEdit 60, 174  
    TDBSRichViewEdit 174  
    TSRichViewEdit 60  
RightMargin 142  
    TSRVPageProperty 142  
RightMargin100Pix 60  
    TSRichViewEdit 60  
RightMarginPix 60  
    TSRichViewEdit 60  
RTFOptions 60  
    TSRichViewEdit 60  
RTFReadProperties 61  
    TSRichViewEdit 61  
RVBackgroundBitmap 61  
    TSRichViewEdit 61  
RVBackgroundPicture 61  
    TSRichViewEdit 61  
RVBackgroundProperties 61  
    TSRichViewEdit 61

RVBackgroundStyle 61  
     TSRichViewEdit 61  
 RVColor 62  
     TSRichViewEdit 62  
 RVEditorOptions 62  
     TSRichViewEdit 62  
 RVFooter 62  
     TSRichViewEdit 62  
 RVFOptions 63  
     TSRichViewEdit 63  
 RVFParaStylesReadMode 63  
     TSRichViewEdit 63  
 RVFTextStylesReadMode 63  
     TSRichViewEdit 63  
 RVHeader 64  
     TSRichViewEdit 64  
 RVNote 64  
     TSRichViewEdit 64  
 RVOptions 65  
     TSRichViewEdit 65

## - S -

ScaleImage 216  
     TSRVPrint 216  
 ScaleImagesForDPI 131, 195, 202  
     TSRVCustomPropertyTB 131  
     TSRVToolBar 195  
     TSRVToolWindow 202  
 ScaleRichView  
     actions 282  
     Document parts 37  
     New after v11.0 20  
     New after v12.0 20  
     New in v10.0 22  
     New in v11.0 21  
     New in v2.0 34  
     New in v3.0 33  
     New in v4.0 30  
     New in v5.0 28  
     New in v6.0 27  
     New in v7.0 26  
     New in v8.0 25  
     New in v9.0 23  
     Notes and text boxes 38  
 ScaleRichView overview 18, 37  
 ScrollBarControlStyle 65  
     TSRichViewEdit 65  
 ScrollButtonWidth 256  
     TSRVTabSet 256  
 ScrollCaretToCenter 96  
     TSRichViewEdit 96  
 Scrolled 189  
     TScIRVRuler 189  
 ScrolledPage 65  
     TSRichViewEdit 65  
 ScrollerMenu 256  
     TSRVTabSet 256  
 ScrollerPosition 256  
     TSRVTabSet 256  
 ScrollingDelay 247, 257  
     TSRVScrollBar 247  
     TSRVTabSet 257  
 ScrollToCaret 96  
     TSRichViewEdit 96  
 ScrollToItem 96  
     TSRichViewEdit 96  
 ScrollToPage 185  
     TSRVPageScroll 185  
 ScrollToTab 259  
     TSRVTabSet 259  
 ScrollType 181  
     TSRVPageScroll 181  
 SelectAll 97  
     TSRichViewEdit 97  
 SelectColor 182  
     TSRVPageScroll 182  
 SetFloatPropertyEd 97  
     TSRichViewEdit 97  
 SetIntPropertyEd 97  
     TSRichViewEdit 97  
 SetMargin 97  
     TSRichViewEdit 97  
 SetMarginMM 97  
     TSRichViewEdit 97  
 SetMarginUnit 98  
     TSRichViewEdit 98  
 SetPageProperties 189  
     TScIRVRuler 189  
 SetRVMargins 98  
     TSRichViewEdit 98  
 ShadowColor 182  
     TSRVPageScroll 182  
 ShadowWidth 182  
     TSRVPageScroll 182  
 ShowHint 275  
     TSRVToolButton 275  
 ShowScrollHint 163  
     TSRVViewProperty 163  
 SkinFontIndex 265

- SkinFontIndex 265
    - TSRVTabItem 265
  - SkinIndex 230
    - TSRVSkinManager 230
  - SkinManager 66, 247, 257
    - TSRichViewEdit 66
    - TSRVScrollBar 247
    - TSRVTabSet 257
  - Skins 230
    - TSRVSkinManager 230
  - SkinSchemeIndex 247, 257, 266
    - TSRVScrollBar 247
    - TSRVTabItem 266
    - TSRVTabSet 257
  - SmallChange 247
    - TSRVScrollBar 247
  - SmartPopupProperties 66
    - TSRichViewEdit 66
  - SmartPopupVisible 66
    - TSRichViewEdit 66
  - SmoothScroll 182
    - TSRVPageScroll 182
  - SmoothThumbnails 183
    - TSRVPageScroll 183
  - Spacer 131, 196, 202
    - TSRVCustomPropertyTB 131
    - TSRVToolBar 196
    - TSRVToolWindow 202
  - Spacings 257
    - TSRVTabSet 257
  - SRichViewEdit 183, 188, 217
    - TScIRVRuler 188
    - TSRVPageScroll 183
    - TSRVPrint 217
  - SRVA\_GetPC 283
  - SRVA\_GetS 283
  - SRVA\_GetUnitsName 283
  - SRVA\_LocalizeSRichViewEdit 283
  - SRVA\_LocalizeToolWindow 283
  - SRVControlStyle 248, 258
    - TSRVScrollBar 248
    - TSRVTabSet 258
  - SRVPrint 227
    - TSRVPreview 227
  - SRVToolBar 131
    - TSRVCustomPropertyTB 131
  - StartEditing 99
    - TSRichViewEdit 99
  - StartEditNote 99
    - TSRichViewEdit 99
  - StartFrom 132
    - TSRVLineNumberProperty 132
  - Step 132
    - TSRVLineNumberProperty 132
  - StyleTemplateInsertMode 66
    - TSRichViewEdit 66
  - StyleTemplateName 318
    - TsrvActionCustomInsertSidenote 318
  - Subdocuments 66
    - TSRichViewEdit 66
- ## - T -
- TabHeight 258
    - TSRVTabSet 258
  - TabIndex 258
    - TSRVTabSet 258
  - TableIconDelay 163
    - TSRVViewProperty 163
  - TableIconPopupMenu 163
    - TSRVViewProperty 163
  - TablesAsLines 133
    - TSRVLineNumberProperty 133
  - TabNavigation 68
    - TSRichViewEdit 68
  - Tabs 259
    - TSRVTabSet 259
  - Tag 266, 275
    - TSRVTabItem 266
    - TSRVToolButton 275
  - TCustomSRVEdit 327
  - TCustomSRVGraphicControl 329
  - TCustomSRVPanel 330
  - TDBSRichViewEdit 169
  - TDBSRichViewEdit.AutoDisplay 173
  - TDBSRichViewEdit.DataField 173
  - TDBSRichViewEdit.DataSource 173
  - TDBSRichViewEdit.ExternalRV 174
  - TDBSRichViewEdit.FieldFormat 174
  - TDBSRichViewEdit.IgnoreEscape 174
  - TDBSRichViewEdit.RichViewEdit 174
  - Text 266
    - TSRVTabItem 266
  - TextAlignH 259
    - TSRVTabSet 259
  - TextAlignV 259
    - TSRVTabSet 259
  - TextEngine 68
    - TSRichViewEdit 68
  - Texts 163

- Texts 163
  - TSRVViewProperty 163
- TFormHintsClose 349
- Ticks 133
  - TSRVLineNumberProperty 133
- Title 294, 295
  - TsrvActionPrint 294
  - TsrvActionQuickPrint 295
- TitlePage 142
  - TSRVPageProperty 142
- TLRAlignment 375
- TopMargin 142
  - TSRVPageProperty 142
- TopMargin100Pix 68
  - TSRichViewEdit 68
- TopMarginPix 68
  - TSRichViewEdit 68
- TotalFrameHPix 217
  - TSRVPrint 217
- TotalFrameWPix 217
  - TSRVPrint 217
- TPSAlignText 176
- TPSDrawPageBorderEvent 186
- TPSDrawPageNumberEvent 186
- TPSDrawSelectedPageEvent 187
- TPSScrollType 181
- TransparentColor 126
  - TSRVBackgroundProperty 126
- TScLRVRuler 187
  - TScLRVRuler.CheckMargins 188
  - TScLRVRuler.GetPageProperties 188
  - TScLRVRuler.OnAfterMarginChanged 189
  - TScLRVRuler.Scrolled 189
  - TScLRVRuler.SetPageProperties 189
  - TScLRVRuler.SRichViewEdit 188
- TSRichViewEdit 40
  - classes of properties 123
- TSRichViewEdit.AcceptDragDropFormats 47
- TSRichViewEdit.AcceptPasteFormats 47
- TSRichViewEdit.ActiveEditor 47
- TSRichViewEdit.AnimationMode 48
- TSRichViewEdit.BackgroundProperty 48
- TSRichViewEdit.BeginComponentsUpdate 72
- TSRichViewEdit.BiDiMode 48
- TSRichViewEdit.BottomMargin100Pix 48
- TSRichViewEdit.BottomMarginPix 48
- TSRichViewEdit.CalculateAllPagePositions 72
- TSRichViewEdit.CalculatePageCount 72
- TSRichViewEdit.CalculatePagePosition 73
- TSRichViewEdit.CanScroll 49
- TSRichViewEdit.CanUpdate 49
- TSRichViewEdit.CanUpdateMargin 49
- TSRichViewEdit.CanUpdatePosition 49
- TSRichViewEdit.CaretBlinkTime 49
- TSRichViewEdit.CaretPos 49
- TSRichViewEdit.CaretVisible 50
- TSRichViewEdit.Clear 73
- TSRichViewEdit.ConvertRVtoSRV 73
- TSRichViewEdit.ConvertSRVtoRV 74
- TSRichViewEdit.ConvertToDifferentUnits 75
- TSRichViewEdit.ConvertToEMU 75
- TSRichViewEdit.ConvertToPixels 75
- TSRichViewEdit.ConvertToTwips 75
- TSRichViewEdit.Copy 75
- TSRichViewEdit.CPEventKind 50
- TSRichViewEdit.Create 75
- TSRichViewEdit.CurControl 50
- TSRichViewEdit.CurrentNote 50
- TSRichViewEdit.CurrentNoteParentEditor 50
- TSRichViewEdit.CurrentPage 51
- TSRichViewEdit.Cut 75
- TSRichViewEdit.DeletePage 76
- TSRichViewEdit.Destroy 76
- TSRichViewEdit.Document 51
- TSRichViewEdit.DrawMetafile 76
- TSRichViewEdit.DrawPage 77
- TSRichViewEdit.EndComponentsUpdate 77
- TSRichViewEdit.ExternalRVStyle 51
- TSRichViewEdit.ExternalRVStyleFooter 51
- TSRichViewEdit.ExternalRVStyleHeader 51
- TSRichViewEdit.FindNextCheckpoint 78
- TSRichViewEdit.FindNextHeading 78
- TSRichViewEdit.FindNextHyperlink 79
- TSRichViewEdit.FindNextItem 79
- TSRichViewEdit.FindPriorCheckpoint 80
- TSRichViewEdit.FindPriorHeading 80
- TSRichViewEdit.FindPriorHyperlink 81
- TSRichViewEdit.FindPriorItem 81
- TSRichViewEdit.FirstCurPage 82
- TSRichViewEdit.FirstPageNo 52
- TSRichViewEdit.FixMarginsMode 52
- TSRichViewEdit.Format 82
- TSRichViewEdit.GetCaretPosInUnits 83
- TSRichViewEdit.GetCurrentLineCol 83
- TSRichViewEdit.GetFirstItemVisibleRV 83
- TSRichViewEdit.GetFirstVisiblePage 84
- TSRichViewEdit.GetItemAt 84
- TSRichViewEdit.GetItemBounds 84

- 
- TSRichViewEdit.GetItemBounds100 84
  - TSRichViewEdit.GetItemCoords100 85
  - TSRichViewEdit.GetItemPages 86
  - TSRichViewEdit.GetLastItemVisibleRV 86
  - TSRichViewEdit.GetLastVisiblePage 87
  - TSRichViewEdit.GetLeftMargin100Pix 54
  - TSRichViewEdit.GetLeftMarginPix 54
  - TSRichViewEdit.GetPageAt 87
  - TSRichViewEdit.GetPageClientRect 87
  - TSRichViewEdit.GetPageLastItemNo 87
  - TSRichViewEdit.GetPageNo 88
  - TSRichViewEdit.GetPageStartItemNo 88
  - TSRichViewEdit.GetPageStartTableItem 88
  - TSRichViewEdit.GetPageStartTableRow 89
  - TSRichViewEdit.GetRightMargin100Pix 60
  - TSRichViewEdit.GetRightMarginPix 60
  - TSRichViewEdit.GetTableIconItem 89
  - TSRichViewEdit.GetTableIconRVData 89
  - TSRichViewEdit.HMaxScrollPos 52
  - TSRichViewEdit.HScrollBar 53
  - TSRichViewEdit.HScrollBarSchemeIndex 53
  - TSRichViewEdit.HScrollPos 53
  - TSRichViewEdit.HTMLReadProperties 53
  - TSRichViewEdit.HTMLSaveProperties 53
  - TSRichViewEdit.InsertPageBreak 89
  - TSRichViewEdit.InZoomRect 90
  - TSRichViewEdit.LastCurPage 90
  - TSRichViewEdit.LastPageNo 54
  - TSRichViewEdit.LeftMargin100Pix 54
  - TSRichViewEdit.LeftMarginPix 54
  - TSRichViewEdit.LineNumberProperty 54
  - TSRichViewEdit.LoadRVF 90
  - TSRichViewEdit.LoadRVFFromStream 90
  - TSRichViewEdit.MarkdownProperties 54
  - TSRichViewEdit.MaxPrintedItemNo 55
  - TSRichViewEdit.MaxPrintedOffsInItem 55
  - TSRichViewEdit.MenuHButtons 55
  - TSRichViewEdit.MenuHorizontal 55
  - TSRichViewEdit.MenuVButtons 56
  - TSRichViewEdit.MenuVertical 56
  - TSRichViewEdit.MinPrintedItemNo 56
  - TSRichViewEdit.MinPrintedOffsInItem 56
  - TSRichViewEdit.NextCurCheckpoint 90
  - TSRichViewEdit.NextCurHeading 91
  - TSRichViewEdit.NextCurHyperlink 91
  - TSRichViewEdit.NextCurItem 91
  - TSRichViewEdit.NextCurPage 91
  - TSRichViewEdit.OffsetX 57
  - TSRichViewEdit.OffsetY 57
  - TSRichViewEdit.OnAfterOleDrop 104
  - TSRichViewEdit.OnAssignImageFileName 103
  - TSRichViewEdit.OnBeforeOleDrop 104
  - TSRichViewEdit.OnCaretGetOut 104
  - TSRichViewEdit.OnCaretMove 104
  - TSRichViewEdit.OnCaretMoving 104
  - TSRichViewEdit.OnChange 104
  - TSRichViewEdit.OnChangeActiveEditor 104
  - TSRichViewEdit.OnChangeCurrentControl 105
  - TSRichViewEdit.OnChangeViewModeAfter 105
  - TSRichViewEdit.OnChangeViewModeBefore 105
  - TSRichViewEdit.OnChanging 105
  - TSRichViewEdit.OnCheckControl 105
  - TSRichViewEdit.OnCheckpointVisible 106
  - TSRichViewEdit.OnCheckStickingItems 106
  - TSRichViewEdit.OnClickPage 106
  - TSRichViewEdit.OnContextPopup 106
  - TSRichViewEdit.OnControlAction 106
  - TSRichViewEdit.OnCopy 106
  - TSRichViewEdit.OnCurParaStyleChanged 107
  - TSRichViewEdit.OnCurrentPageChange 107
  - TSRichViewEdit.OnCurTextStyleChanged 107
  - TSRichViewEdit.OnDrawHyperlink 107
  - TSRichViewEdit.OnDropFile 108
  - TSRichViewEdit.OnDropFiles 108
  - TSRichViewEdit.OnFullRedraw 108
  - TSRichViewEdit.OnGetPagePos 108
  - TSRichViewEdit.OnHMenuClickButton 109
  - TSRichViewEdit.OnHMenuEnterButton 109
  - TSRichViewEdit.OnHScrolled 109
  - TSRichViewEdit.OnHTMLSaveImage 110
  - TSRichViewEdit.OnImportPicture 110
  - TSRichViewEdit.OnItemAction 110
  - TSRichViewEdit.OnItemHint 110
  - TSRichViewEdit.OnItemResize 110
  - TSRichViewEdit.OnItemTextEdit 110
  - TSRichViewEdit.OnJump 111
  - TSRichViewEdit.OnLoadCustomFormat 111
  - TSRichViewEdit.OnLoadDocument 111
  - TSRichViewEdit.OnMarginsChanged 111
  - TSRichViewEdit.OnMessageControl 112
  - TSRichViewEdit.OnNewDocument 113
  - TSRichViewEdit.OnOleDragEnter 113
  - TSRichViewEdit.OnOleDragLeave 113
  - TSRichViewEdit.OnOleDragOver 113
  - TSRichViewEdit.OnOleDrop 113
  - TSRichViewEdit.OnPageCountChanged 114
  - TSRichViewEdit.OnPageFormatChanged 114
  - TSRichViewEdit.OnPageScrolled 114

|  |     |                                       |     |
|--|-----|---------------------------------------|-----|
| TSRichViewEdit.OnPaint                 | 114 | TSRichViewEdit.PageWidth100Pix        | 58  |
| TSRichViewEdit.OnPaintComponent        | 114 | TSRichViewEdit.PageWidthPix           | 58  |
| TSRichViewEdit.OnPaintPage             | 115 | TSRichViewEdit.Paste                  | 92  |
| TSRichViewEdit.OnParaStyleConversion   | 116 | TSRichViewEdit.PrintAll               | 92  |
| TSRichViewEdit.OnPaste                 | 116 | TSRichViewEdit.PrintCurrent           | 93  |
| TSRichViewEdit.OnProgress              | 117 | TSRichViewEdit.Printing               | 116 |
| TSRichViewEdit.OnReadField             | 117 | TSRichViewEdit.PrintRange             | 93  |
| TSRichViewEdit.OnReadHyperlink         | 117 | TSRichViewEdit.PriorCurCheckpoint     | 94  |
| TSRichViewEdit.OnReadMergeField        | 117 | TSRichViewEdit.PriorCurHeading        | 94  |
| TSRichViewEdit.OnResize                | 117 | TSRichViewEdit.PriorCurHyperlink      | 94  |
| TSRichViewEdit.OnResizeDoc             | 117 | TSRichViewEdit.PriorCurlItem          | 95  |
| TSRichViewEdit.OnResizing              | 117 | TSRichViewEdit.PriorCurPage           | 95  |
| TSRichViewEdit.OnRVDbClick             | 117 | TSRichViewEdit.ProcessControls        | 95  |
| TSRichViewEdit.OnRVFControlNeeded      | 118 | TSRichViewEdit.RangeSearch            | 58  |
| TSRichViewEdit.OnRVFImageListNeeded    | 118 | TSRichViewEdit.RangeSearchFirstY      | 59  |
| TSRichViewEdit.OnRVFPictureNeeded      | 118 | TSRichViewEdit.RangeSearchLastY       | 59  |
| TSRichViewEdit.OnRVMouseDown           | 118 | TSRichViewEdit.ReadModeProperty       | 59  |
| TSRichViewEdit.OnRVMouseMove           | 118 | TSRichViewEdit.ReadOnly               | 59  |
| TSRichViewEdit.OnRVMouseUp             | 118 | TSRichViewEdit.Reformat               | 82  |
| TSRichViewEdit.OnRVRightClick          | 119 | TSRichViewEdit.RestoreCanvasZoom      | 100 |
| TSRichViewEdit.OnSaveComponentToFile   | 119 | TSRichViewEdit.ReturnToNote           | 95  |
| TSRichViewEdit.OnSaveCustomFormat      | 119 | TSRichViewEdit.RichViewEdit           | 60  |
| TSRichViewEdit.OnSaveDocXExtra         | 119 | TSRichViewEdit.RightMargin100Pix      | 60  |
| TSRichViewEdit.OnSaveHTMLExtra         | 120 | TSRichViewEdit.RightMarginPix         | 60  |
| TSRichViewEdit.OnSaveImage2            | 120 | TSRichViewEdit.RTFOptions             | 60  |
| TSRichViewEdit.OnSaveItemToFile        | 120 | TSRichViewEdit.RTFReadProperties      | 61  |
| TSRichViewEdit.OnSaveRTFExtra          | 120 | TSRichViewEdit.RVBackgroundBitmap     | 61  |
| TSRichViewEdit.OnSelect                | 120 | TSRichViewEdit.RVBackgroundPicture    | 61  |
| TSRichViewEdit.OnSmartPopupClick       | 120 | TSRichViewEdit.RVBackgroundProperties | 61  |
| TSRichViewEdit.OnSpellingCheck         | 121 | TSRichViewEdit.RVBackgroundStyle      | 61  |
| TSRichViewEdit.OnStyleConversion       | 121 | TSRichViewEdit.RVColor                | 62  |
| TSRichViewEdit.OnTableIconClick        | 121 | TSRichViewEdit.RVEditorOptions        | 62  |
| TSRichViewEdit.OnTextFound             | 121 | TSRichViewEdit.RVFooter               | 62  |
| TSRichViewEdit.OnURLNeeded             | 121 | TSRichViewEdit.RVFOptions             | 63  |
| TSRichViewEdit.OnVMenuClickButton      | 121 | TSRichViewEdit.RVFPaStylesReadMode    | 63  |
| TSRichViewEdit.OnVMenuEnterButton      | 122 | TSRichViewEdit.RVFTextStylesReadMode  | 63  |
| TSRichViewEdit.OnVScrolled             | 122 | TSRichViewEdit.RVHeader               | 64  |
| TSRichViewEdit.OnWriteHyperlink        | 122 | TSRichViewEdit.RVNote                 | 64  |
| TSRichViewEdit.OnWriteObjectProperties | 122 | TSRichViewEdit.RVOptions              | 65  |
| TSRichViewEdit.OnZoomChanged           | 122 | TSRichViewEdit.ScrollBarControlStyle  | 65  |
| TSRichViewEdit.OnZoomViewChanged       | 123 | TSRichViewEdit.ScrollCaretToCenter    | 96  |
| TSRichViewEdit.OutZoomRect             | 92  | TSRichViewEdit.ScrolledPage           | 65  |
| TSRichViewEdit.PageCount               | 57  | TSRichViewEdit.ScrollToCaret          | 96  |
| TSRichViewEdit.PageHeight100Pix        | 57  | TSRichViewEdit.ScrollToItem           | 96  |
| TSRichViewEdit.PageHeightPix           | 57  | TSRichViewEdit.SelectAll              | 97  |
| TSRichViewEdit.PageNoMouseDown         | 58  | TSRichViewEdit.SetFloatPropertyEd     | 97  |
| TSRichViewEdit.PageNoMouseMove         | 58  | TSRichViewEdit.SetIntPropertyEd       | 97  |
| TSRichViewEdit.PagePosProperty         | 58  | TSRichViewEdit.SetMargin              | 97  |
| TSRichViewEdit.PageProperty            | 58  | TSRichViewEdit.SetMarginMM            | 97  |



- 
- TSRichViewEdit.SetMarginUnit 98
  - TSRichViewEdit.SetRVMargins 98
  - TSRichViewEdit.SkinManager 66
  - TSRichViewEdit.SmartPopupProperties 66
  - TSRichViewEdit.SmartPopupVisible 66
  - TSRichViewEdit.StartEditing 99
  - TSRichViewEdit.StartEditNote 99
  - TSRichViewEdit.StyleTemplateInsertMode 66
  - TSRichViewEdit.Subdocuments 66
  - TSRichViewEdit.TabNavigation 68
  - TSRichViewEdit.TextEngine 68
  - TSRichViewEdit.TopMargin100Pix 68
  - TSRichViewEdit.TopMarginPix 68
  - TSRichViewEdit.UnitsPerInchH 100
  - TSRichViewEdit.UnitsPerInchV 100
  - TSRichViewEdit.UnitsProgram 68
  - TSRichViewEdit.UpdateBuffer 100
  - TSRichViewEdit.UpdateBufferAt 100
  - TSRichViewEdit.UseDrawHyperlinksEvent 69
  - TSRichViewEdit.UseStyleTemplates 69
  - TSRichViewEdit.Version 69
  - TSRichViewEdit.ViewProperty 69
  - TSRichViewEdit.VMaxScrollPos 69
  - TSRichViewEdit.VScrollBar 69
  - TSRichViewEdit.VScrollBarSchemeIndex 70
  - TSRichViewEdit.VScrollPos 70
  - TSRichViewEdit.ZoomCanvas 100
  - TSRichViewEdit.ZoomRectIn 101
  - TSRichViewEdit.ZoomRectOut 101
  - TSRichViewEdit.ZoomToFullPages 101
  - TsrvAction 284
  - TsrvAction.Control 285
  - TsrvActionCustomInsertSidenote 317
  - TsrvActionCustomInsertSidenote.BoxPosition 318
  - TsrvActionCustomInsertSidenote.BoxProperties 318
  - TsrvActionCustomInsertSidenote.StyleTemplateName 318
  - TsrvActionCustomLayout 296
  - TsrvActionCustomLayout.OnExecuted 298
  - TsrvActionCustomOrientation 286
  - TsrvActionCustomZoom 299
  - TsrvActionEditFooter 315
  - TsrvActionEditHeader 315
  - TsrvActionEditMain 316
  - TsrvActionEditNote 319
  - TsrvActionInsertEndnote 319
  - TsrvActionInsertFootnote 320
  - TsrvActionInsertNote 321
  - TsrvActionInsertNote.Font 322
  - TsrvActionInsertSidenote 322
  - TsrvActionInsertSidenote.RefStyleTemplateName 324
  - TsrvActionInsertTextBox 324
  - TsrvActionLayoutDraft 299
  - TsrvActionLayoutDraft.HPadding 301
  - TsrvActionLayoutPrint 301
  - TsrvActionLayoutPrint.Color 303
  - TsrvActionLayoutPrint.FooterVisible 303
  - TsrvActionLayoutPrint.HeaderVisible 303
  - TsrvActionLayoutPrint.HPadding 304
  - TsrvActionLayoutRead 304
  - TsrvActionLayoutRead.Color 306
  - TsrvActionLayoutRead.HidePageBackgroundImage 306
  - TsrvActionLayoutRead.HPadding 306
  - TsrvActionLayoutRead.PageColor 306
  - TsrvActionLayoutRead.PageFlipEffect 306
  - TsrvActionLayoutSideToSide 307
  - TsrvActionLayoutSideToSide.Color 309
  - TsrvActionLayoutSideToSide.FooterVisible 309
  - TsrvActionLayoutSideToSide.HeaderVisible 309
  - TsrvActionLayoutSideToSide.HPadding 309
  - TsrvActionLayoutSideToSide.NoPageSpacing 309
  - TsrvActionLayoutSideToSide.PageFlipEffect 310
  - TsrvActionLayoutWeb 310
  - TsrvActionLineNumbers 287
  - TsrvActionOrientationLandscape 287
  - TsrvActionOrientationPortrait 288
  - TsrvActionPageFormat 289
  - TsrvActionPageFormat.PageFormat 290
  - TsrvActionPageFormat.PageHeight 290
  - TsrvActionPageFormat.PageWidth 290
  - TsrvActionPageFormat.Units 290
  - TsrvActionPageSetup 291
  - TsrvActionPageSetup.OnChange 293
  - TsrvActionPageSetup.PageFormats1 292
  - TsrvActionPageSetup.PageFormats2 292
  - TsrvActionPageSetup.PageFormats3 292
  - TsrvActionPageSetup.PageFormats4 292
  - TsrvActionPageSetup.PageFormats5 292
  - TsrvActionPageSetup.PageFormats6 292
  - TsrvActionPreview 293
  - TsrvActionPrint 293
  - TsrvActionPrint.Title 294
  - TsrvActionQuickPrint 294
  - TsrvActionQuickPrint.Title 295
  - TsrvActionReturnToNote 325
  - TsrvActionThumbnails 311
  - TsrvActionThumbnails.OnExecuted 312
  - TsrvActionThumbnails.PageScroll 312

|  |     |  |          |
|--|-----|--|----------|
| TsrvActionZoom                                   | 313 | TSRVControlStyle                             | 276      |
| TsrvActionZoom.ZoomPercent                       | 313 | TSRVCtrlPaintPage                            | 331      |
| TsrvActionZoomFullPage                           | 314 | TsrvCustomActionThatNeedsDocProps            | 285      |
| TsrvActionZoomPageWidth                          | 314 | TsrvCustomActionThatNeedsDocProps.ActionSave | 286      |
| TSRVAlignPageH                                   | 144 | TSRVCustomComboBox                           | 332      |
| TSRVAlignPageV                                   | 145 | TSRVCustomControl                            | 335      |
| TSRVAlignText                                    | 271 | TSRVCustomFormatEvent                        | 111, 119 |
| TSRVBackgroundProperty                           | 123 | TSRVCustomListBox                            | 336      |
| TSRVBackgroundProperty.ColorBegin                | 124 | TsrvCustomPrintAction                        | 296      |
| TSRVBackgroundProperty.ColorEnd                  | 124 | TSRVCustomPropertyTB                         | 127      |
| TSRVBackgroundProperty.ColorMiddle               | 124 | TSRVCustomPropertyTB.AlignButton             | 128      |
| TSRVBackgroundProperty.FillType                  | 125 | TSRVCustomPropertyTB.BorderWidth             | 128      |
| TSRVBackgroundProperty.GlobalPageBackgroundColor | 125 | TSRVCustomPropertyTB.ButtonHeight            | 128      |
| TSRVBackgroundProperty.PercentMiddle             | 126 | TSRVCustomPropertyTB.Buttons                 | 129      |
| TSRVBackgroundProperty.Picture                   | 126 | TSRVCustomPropertyTB.ButtonWidth             | 129      |
| TSRVBackgroundProperty.PicturePosition           | 126 | TSRVCustomPropertyTB.Color                   | 129      |
| TSRVBackgroundProperty.TransparentColor          | 126 | TSRVCustomPropertyTB.ColorDisable            | 129      |
| TSRVBackgroundProperty.Visible                   | 127 | TSRVCustomPropertyTB.ColorDown               | 130      |
| TSRVBorderPen                                    | 135 | TSRVCustomPropertyTB.ColorSelect             | 130      |
| TSRVBoxScheme                                    | 233 | TSRVCustomPropertyTB.ImageList               | 130      |
| TSRVBoxSchemeCollection                          | 234 | TSRVCustomPropertyTB.Indent                  | 130      |
| TSRVButton                                       | 343 | TSRVCustomPropertyTB.PenFrame                | 130      |
| TSRVButtonCollection                             | 271 | TSRVCustomPropertyTB.ScaleImagesForDPI       | 131      |
| TSRVCanvasControl                                | 331 | TSRVCustomPropertyTB.Spacer                  | 131      |
| TSRVCaretPen                                     | 136 | TSRVCustomPropertyTB.SRVToolBar              | 131      |
| TSRVCBoxItem                                     | 375 | TSRVDataDrawHyperlinkEvent                   | 107      |
| TSRVCBoxItemCollection                           | 375 | TSRVDBCheckBox                               | 369      |
| TSRVChangeActiveEditorEvent                      | 104 | TSRVDBComboBox                               | 370      |
| TSRVChangeCurrentControlEvents                   | 105 | TSRVDBEdit                                   | 371      |
| TSRVChangeViewModeEvent                          | 105 | TSRVDBListBox                                | 372      |
| TSRVCheckBox                                     | 345 | TSRVDBMemo                                   | 373      |
| TSRVCheckBoxState                                | 345 | TSRVDBText                                   | 374      |
| TSRVCheckControl                                 | 105 | TSRVDDrawButtonEvent                         | 196      |
| TSRVCheckpointVisibleEvent                       | 106 | TSRVDDrawScrollBarEvent                      | 268      |
| TSRVClickButtonEvent                             | 272 | TSRVDDrawState                               | 269      |
| TSRVCloseButton                                  | 262 | TSRVDDrawStates                              | 269      |
| TSRVCloseButton.AllowCloseLastTab                | 263 | TSRVDDrawTabEvent                            | 269      |
| TSRVCloseButton.Height                           | 263 | TSRVDDrawTabSetEvent                         | 270      |
| TSRVCloseButton.HorizontalAlign                  | 263 | TSRVEdit                                     | 349      |
| TSRVCloseButton.IndentX                          | 263 | TSRVEditCharCase                             | 349      |
| TSRVCloseButton.IndentY                          | 263 | TSRVEditControl                              | 339      |
| TSRVCloseButton.VerticalAlign                    | 264 | TSRVEditHintsEvent                           | 349      |
| TSRVCloseButton.Visibility                       | 264 | TSRVFillType                                 | 125      |
| TSRVCloseButton.Visible                          | 264 | TSRVFixMarginsMode                           | 52       |
| TSRVCloseButton.Width                            | 264 | TSRVFloatProperty                            | 276      |
| TSRVCloseTabEvent                                | 260 | TSRVGetPagePosEvent                          | 108      |
| TSRVComboBox                                     | 346 | TSRVGraphicControl                           | 339      |
| TSRVComponentEvent                               | 114 | TSRVGroupBox                                 | 352      |
|  |     | TSRVHAlign                                   | 140      |

- TSRVHeaderFooterFont 158, 159, 161
- TSRVHeaderFooterPen 157, 159
- TSRVHeaderFooterType 66
- TSRVIconStyle 160
- TSRVImageScrollCollection 375
- TSRVImageScrollItem 379
- TSRVImageScrollKind 354
- TSRVImagesScroll 354
- TSRVIntProperty 277
- TSRVItemOption 375
- TSRVItemOptions 375
- TSRVLabel 358
- TSRVLBoxItem 375
- TSRVLBoxItemCollection 375
- TSRVLineNumberFont 132
- TSRVLineNumberProperty 131
- TSRVLineNumberProperty.DistanceFromText 132
- TSRVLineNumberProperty.Font 132
- TSRVLineNumberProperty.StartFrom 132
- TSRVLineNumberProperty.Step 132
- TSRVLineNumberProperty.TablesAsLines 133
- TSRVLineNumberProperty.Ticks 133
- TSRVLineNumberProperty.Visible 133
- TSRVListBox 359
- TSRVMainPen 161
- TSRVMargins 254
- TSRVMarginsChanged 111
- TSRVMarginsPen 162
- TSRVMemo 362
- TSRVMessageControl 112
- TSRVMouseEvent 277
- TSRVPageFlipEffect 153
- TSRVPageFormat 278
- TSRVPageFormatAB 292
- TSRVPageFormatANSIArch 292
- TSRVPageFormatC 292
- TSRVPageFormatCardTrad 292
- TSRVPageFormatPAOtherJISB 292
- TSRVPageFormats 292
- TSRVPageFormatsAB 292
- TSRVPageFormatsANSIArch 292
- TSRVPageFormatsC 292
- TSRVPageFormatsCardTrad 292
- TSRVPageFormatsPAOtherJISB 292
- TSRVPageFormatsUSA 292
- TSRVPageFormatUSA 292
- TSRVPageNoFont 139
- TSRVPagePositionDistance 145
- TSRVPagePositionDistances 145
- TSRVPagePositionProperty 143
- TSRVPagePositionProperty.AlignPageH 144
- TSRVPagePositionProperty.AlignPageV 145
- TSRVPagePositionProperty.HiddenDistances 145
- TSRVPagePositionProperty.HPadding 146
- TSRVPagePositionProperty.MaxPageColCount 147
- TSRVPagePositionProperty.PageHSpacing 147
- TSRVPagePositionProperty.PageVSpacing 147
- TSRVPagePositionProperty.VPadding 146
- TSRVPageProperty 133
- TSRVPageProperty.AutoWidth 135
- TSRVPageProperty.BorderPen 135
- TSRVPageProperty.BottomMargin 136
- TSRVPageProperty.CaretPen 136
- TSRVPageProperty.ConvertPageFormatToPageSize 143
- TSRVPageProperty.ConvertPageSizeToPageFormat 143
- TSRVPageProperty.FacingPages 136
- TSRVPageProperty.FooterVisible 136
- TSRVPageProperty.FooterY 137
- TSRVPageProperty.HeaderVisible 137
- TSRVPageProperty.HeaderY 137
- TSRVPageProperty.LeftMargin 138
- TSRVPageProperty.MirrorMargins 138
- TSRVPageProperty.Orientation 138
- TSRVPageProperty.PageFormat 139
- TSRVPageProperty.PageHeight 139
- TSRVPageProperty.PageNoFirst 139
- TSRVPageProperty.PageNoFont 139
- TSRVPageProperty.PageNoFromNumber 140
- TSRVPageProperty.PageNoHAlign 140
- TSRVPageProperty.PageNoVAlign 140
- TSRVPageProperty.PageNoVisible 140
- TSRVPageProperty.PageViewMode 141
- TSRVPageProperty.PageWidth 141
- TSRVPageProperty.PrintableAreaPen 141
- TSRVPageProperty.RightMargin 142
- TSRVPageProperty.TitlePage 142
- TSRVPageProperty.TopMargin 142
- TSRVPageProperty.UsePageOrders 142
- TSRVPageScroll 174
- TSRVPageScroll.AlignText 176
- TSRVPageScroll.AutoScrolled 177
- TSRVPageScroll.AutoUpdate 177
- TSRVPageScroll.BackgroundColor 177
- TSRVPageScroll.BeginUpdate 183
- TSRVPageScroll.CachedPagesCount 178
- TSRVPageScroll.CacheScrollLimit 178
- TSRVPageScroll.ChangeDelay 178
- TSRVPageScroll.Columns 178

|                                   |     |                                 |     |
|-----------------------------------|-----|---------------------------------|-----|
| TSRVPageScroll.EndUpdate          | 183 | TSRVPreview.Prev                | 227 |
| TSRVPageScroll.First              | 184 | TSRVPreview.SRVPrint            | 227 |
| TSRVPageScroll.GetPageAt          | 184 | TSRVPrint                       | 204 |
| TSRVPageScroll.Last               | 184 | TSRVPrint.AutoUpdate            | 207 |
| TSRVPageScroll.MarginHorizontal   | 179 | TSRVPrint.BeginUpdate           | 218 |
| TSRVPageScroll.MarginVertical     | 179 | TSRVPrint.Center                | 207 |
| TSRVPageScroll.Next               | 184 | TSRVPrint.CheckMargin           | 218 |
| TSRVPageScroll.OnDrawPageBorder   | 186 | TSRVPrint.ClipMargins           | 208 |
| TSRVPageScroll.OnDrawPageNumber   | 186 | TSRVPrint.EndUpdate             | 219 |
| TSRVPageScroll.OnDrawSelectedPage | 187 | TSRVPrint.FrameCountX           | 208 |
| TSRVPageScroll.PageBorderColor    | 179 | TSRVPrint.FrameCountY           | 209 |
| TSRVPageScroll.PageBreakHeight    | 179 | TSRVPrint.FrameHeightPix        | 209 |
| TSRVPageScroll.PageBreakWidth     | 180 | TSRVPrint.FrameWidthPix         | 210 |
| TSRVPageScroll.PageHeight         | 180 | TSRVPrint.IntegerCount          | 210 |
| TSRVPageScroll.PageNoFont         | 180 | TSRVPrint.MetafileCompatibility | 211 |
| TSRVPageScroll.PageNoVisible      | 181 | TSRVPrint.NoMetafiles           | 211 |
| TSRVPageScroll.PageProportions    | 181 | TSRVPrint.OffXPix               | 211 |
| TSRVPageScroll.PageWidth          | 181 | TSRVPrint.OffYPix               | 212 |
| TSRVPageScroll.Prev               | 185 | TSRVPrint.OnSendingToPrinter    | 222 |
| TSRVPageScroll.ScrollToPage       | 185 | TSRVPrint.OnUpdateData          | 222 |
| TSRVPageScroll.ScrollType         | 181 | TSRVPrint.OptimalOrientation    | 219 |
| TSRVPageScroll.SelectColor        | 182 | TSRVPrint.Orientation           | 212 |
| TSRVPageScroll.ShadowColor        | 182 | TSRVPrint.PageColCount          | 213 |
| TSRVPageScroll.ShadowWidth        | 182 | TSRVPrint.PageFormat            | 213 |
| TSRVPageScroll.SmoothScroll       | 182 | TSRVPrint.PageHeightPix         | 213 |
| TSRVPageScroll.SmoothThumbnails   | 183 | TSRVPrint.PageRowCount          | 214 |
| TSRVPageScroll.SRichViewEdit      | 183 | TSRVPrint.PageWidthPix          | 214 |
| TSRVPageScroll.UpdateCache        | 185 | TSRVPrint.PaperCodeToPageFormat | 219 |
| TSRVPaintBox                      | 364 | TSRVPrint.Print                 | 219 |
| TSRVPaintEvent                    | 114 | TSRVPrint.PrintFrame            | 220 |
| TSRVPaintPage                     | 115 | TSRVPrint.PrintFrames           | 220 |
| TSRVPanel                         | 365 | TSRVPrint.PrintFramesEx         | 221 |
| TSRVAPen                          | 141 | TSRVPrint.PrintMode             | 214 |
| TSRVPicturePosition               | 126 | TSRVPrint.PrintPages            | 221 |
| TSRVOnPaint                       | 270 | TSRVPrint.RealHeight            | 215 |
| TSRVPreview                       | 223 | TSRVPrint.RealWidth             | 216 |
| TSRVPreview.BorderColor           | 225 | TSRVPrint.ScaleImage            | 216 |
| TSRVPreview.BorderWidth           | 225 | TSRVPrint.SRichViewEdit         | 217 |
| TSRVPreview.First                 | 227 | TSRVPrint.TotalFrameHPix        | 217 |
| TSRVPreview.FrameBorderColor      | 225 | TSRVPrint.TotalFrameWPix        | 217 |
| TSRVPreview.FrameBorderWidth      | 225 | TSRVPrint.Update                | 222 |
| TSRVPreview.Last                  | 227 | TSRVPrint.UsePhysicalOffsets    | 217 |
| TSRVPreview.Next                  | 227 | TSRVPrintingEvent               | 222 |
| TSRVPreview.OnAfterPaint          | 228 | TSRVPrintingProgressEvent       | 116 |
| TSRVPreview.OnBeforePaint         | 228 | TSRVPrintMode                   | 214 |
| TSRVPreview.PageBorderColor       | 226 | TSRVPropertyTBH                 | 148 |
| TSRVPreview.PageBorderStyle       | 226 | TSRVPropertyTBH.Align           | 149 |
| TSRVPreview.PageBorderWidth       | 226 | TSRVPropertyTBH.Visible         | 149 |
| TSRVPreview.PageNo                | 226 | TSRVPropertyTBV                 | 149 |

- TSRVPropertyTBV.Align 150
- TSRVPropertyTBV.Visible 151
- TSRVRadioButton 367
- TSRVReadModeProperty 151
- TSRVReadModeProperty.Active 152
- TSRVReadModeProperty.HidePageBackgroundImage 152
- TSRVReadModeProperty.MaxPageColCount 152
- TSRVReadModeProperty.MinPageColCount 152
- TSRVReadModeProperty.PageColor 153
- TSRVReadModeProperty.PageFlipEffect 153
- TSRVRichViewEditType 99
- TSRVScrollBar 243
- TSRVScrollBar.DecButtonWidth 245
- TSRVScrollBar.Flat 245
- TSRVScrollBar.FocusBlinkDelay 245
- TSRVScrollBar.IncButtonWidth 246
- TSRVScrollBar.Kind 246
- TSRVScrollBar.LargeChange 247
- TSRVScrollBar.OnDrawArea 249
- TSRVScrollBar.OnDrawBorder 249
- TSRVScrollBar.OnDrawDecButton 249
- TSRVScrollBar.OnDrawIncButton 249
- TSRVScrollBar.OnDrawThumbTab 249
- TSRVScrollBar.PageSize 246
- TSRVScrollBar.Position 246
- TSRVScrollBar.ScrollingDelay 247
- TSRVScrollBar.SkinManager 247
- TSRVScrollBar.SkinSchemeIndex 247
- TSRVScrollBar.SmallChange 247
- TSRVScrollBar.SRVControlStyle 248
- TSRVScrollBar.UseXPTThemes 248
- TSRVScrollBarSkinScheme 238
- TSRVScrollBarSkinSchemeCollection 240
- TSRVScrollerPosition 256, 354
- TSRVSkin 231
- TSRVSkin.BoxSchemes 232
- TSRVSkin.Elements 232
- TSRVSkin.Fonts 232
- TSRVSkin.HorizontalScrollBarSchemes 232
- TSRVSkin.HorizontalTabSetSchemes 232
- TSRVSkin.Name 233
- TSRVSkin.VerticalScrollBarSchemes 233
- TSRVSkin.VerticalTabSetSchemes 233
- TSRVSkinCollection 235
- TSRVSkinElement 237
- TSRVSkinElementCollection 238
- TSRVSkinFont 235
- TSRVSkinFontCollection 236
- TSRVSkinManager 228
  - Classes of properties 230
- TSRVSkinManager.CurrentSkin 230
- TSRVSkinManager.SkinIndex 230
- TSRVSkinManager.Skins 230
- TSRVSubDocuments 66
- TSRVTabCollection 267
- TSRVTabItem 264
- TSRVTabItem.Enabled 265
- TSRVTabItem.ImageIndex 265
- TSRVTabItem.SkinFontIndex 265
- TSRVTabItem.SkinSchemeIndex 266
- TSRVTabItem.Tag 266
- TSRVTabItem.Text 266
- TSRVTabItem.Visible 266
- TSRVTabMoveEvent 261
- TSRVTabSet 249
  - classes of properties 261
- TSRVTabSet.AutoDarkMode 253
- TSRVTabSet.AutoTabWidth 252
- TSRVTabSet.CanMoveTabs 253
- TSRVTabSet.CloseButton 253
- TSRVTabSet.DarkMode 253
- TSRVTabSet.FirstTabIndex 253
- TSRVTabSet.ImageList 253
- TSRVTabSet.Indent 254
- TSRVTabSet.Kind 254
- TSRVTabSet.LastTabIndex 254
- TSRVTabSet.Margins 254
- TSRVTabSet.MaxTabSize 255
- TSRVTabSet.MinTabSize 255
- TSRVTabSet.MirrorText 255
- TSRVTabSet.OnChange 260
- TSRVTabSet.OnCloseTab 260
- TSRVTabSet.OnDrawBackground 260
- TSRVTabSet.OnDrawBorder 260
- TSRVTabSet.OnDrawCloseButton 260
- TSRVTabSet.OnDrawDecButton 260
- TSRVTabSet.OnDrawIncButton 260
- TSRVTabSet.OnDrawTab 260
- TSRVTabSet.OnMoveTab 261
- TSRVTabSet.OppositeTabPosition 256
- TSRVTabSet.ScrollButtonWidth 256
- TSRVTabSet.ScrrollerMenu 256
- TSRVTabSet.ScrrollerPosition 256
- TSRVTabSet.ScrollingDelay 257
- TSRVTabSet.ScrollToTab 259
- TSRVTabSet.SkinManager 257
- TSRVTabSet.SkinSchemeIndex 257

|  |     |                                  |     |
|--|-----|----------------------------------|-----|
| TSRVTabSet.Spacings                      | 257 | TSRVToolBar.OnEnter              | 196 |
| TSRVTabSet.SRVControlStyle               | 258 | TSRVToolBar.OnEnterButton        | 197 |
| TSRVTabSet.TabHeight                     | 258 | TSRVToolBar.OnLeave              | 197 |
| TSRVTabSet.TabIndex                      | 258 | TSRVToolBar.PenFrame             | 195 |
| TSRVTabSet.Tabs                          | 259 | TSRVToolBar.ScaleImagesForDPI    | 195 |
| TSRVTabSet.TextAlignH                    | 259 | TSRVToolBar.Spacer               | 196 |
| TSRVTabSet.TextAlignV                    | 259 | TSRVToolButton                   | 272 |
| TSRVTabSetKind                           | 254 | TSRVToolButton.AllowAllUp        | 273 |
| TSRVTabSetSkinScheme                     | 241 | TSRVToolButton.Caption           | 273 |
| TSRVTabSetSkinScheme.FntTab              | 241 | TSRVToolButton.Down              | 273 |
| TSRVTabSetSkinScheme.ImgAddTabButton     | 241 | TSRVToolButton.Enabled           | 273 |
| TSRVTabSetSkinScheme.ImgBackgroundBegin  | 241 | TSRVToolButton.GroupIndex        | 274 |
| TSRVTabSetSkinScheme.ImgBackgroundCenter | 241 | TSRVToolButton.Hint              | 273 |
| TSRVTabSetSkinScheme.ImgBackgroundEnd    | 241 | TSRVToolButton.ImageIndexDisable | 274 |
| TSRVTabSetSkinScheme.ImgBorderBackground | 241 | TSRVToolButton.ImageIndexDown    | 274 |
| TSRVTabSetSkinScheme.ImgBorderBegin      | 241 | TSRVToolButton.ImageIndexMove    | 274 |
| TSRVTabSetSkinScheme.ImgBorderEnd        | 241 | TSRVToolButton.ImageIndexNormal  | 275 |
| TSRVTabSetSkinScheme.ImgCloseButton      | 241 | TSRVToolButton.ShowHint          | 275 |
| TSRVTabSetSkinScheme.ImgDecButton        | 241 | TSRVToolButton.Tag               | 275 |
| TSRVTabSetSkinScheme.ImgIncButton        | 241 | TSRVToolButton.Visible           | 275 |
| TSRVTabSetSkinScheme.ImgTabBackground    | 241 | TSRVToolWindow                   | 197 |
| TSRVTabSetSkinScheme.ImgTabBegin         | 241 | TSRVToolWindow.AlignText         | 198 |
| TSRVTabSetSkinScheme.ImgTabEnd           | 241 | TSRVToolWindow.BorderWidth       | 199 |
| TSRVTabSetSkinScheme.Name                | 241 | TSRVToolWindow.ButtonCol         | 199 |
| TSRVTabSetSkinScheme.Collection          | 242 | TSRVToolWindow.ButtonHeight      | 199 |
| TSRVTBAlignH                             | 149 | TSRVToolWindow.Buttons           | 199 |
| TSRVTBAlignV                             | 150 | TSRVToolWindow.ButtonWidth       | 199 |
| TSRVTexts                                | 163 | TSRVToolWindow.Color             | 200 |
| TSRVTHAlign                              | 281 | TSRVToolWindow.ColorDisable      | 200 |
| TSRVToolBar                              | 189 | TSRVToolWindow.ColorDown         | 200 |
| TSRVToolBar.AlignButton                  | 191 | TSRVToolWindow.ColorSelect       | 201 |
| TSRVToolBar.AlignText                    | 191 | TSRVToolWindow.Execute           | 203 |
| TSRVToolBar.AutoSize                     | 192 | TSRVToolWindow.HintCancel        | 201 |
| TSRVToolBar.BorderWidth                  | 192 | TSRVToolWindow.HintFont          | 201 |
| TSRVToolBar.ButtonCol                    | 192 | TSRVToolWindow.HintHeight        | 201 |
| TSRVToolBar.ButtonHeight                 | 192 | TSRVToolWindow.ImageList         | 202 |
| TSRVToolBar.Buttons                      | 192 | TSRVToolWindow.OnClickButton     | 203 |
| TSRVToolBar.ButtonWidth                  | 193 | TSRVToolWindow.OnEnter           | 203 |
| TSRVToolBar.Color                        | 193 | TSRVToolWindow.OnEnterButton     | 203 |
| TSRVToolBar.ColorDisable                 | 193 | TSRVToolWindow.OnLeave           | 204 |
| TSRVToolBar.ColorDown                    | 193 | TSRVToolWindow.OnSelectEvent     | 204 |
| TSRVToolBar.ColorSelect                  | 194 | TSRVToolWindow.PenFrame          | 202 |
| TSRVToolBar.HintCancel                   | 194 | TSRVToolWindow.ScaleImagesForDPI | 202 |
| TSRVToolBar.HintFont                     | 194 | TSRVToolWindow.Spacer            | 202 |
| TSRVToolBar.HintHeight                   | 194 | TSRVTSCloseButtonVisibility      | 264 |
| TSRVToolBar.ImageList                    | 195 | TSRVTVAlign                      | 281 |
| TSRVToolBar.Indent                       | 195 | TSRVVAlign                       | 140 |
| TSRVToolBar.OnClickButton                | 196 | TSRVViewMode                     | 165 |
| TSRVToolBar.OnDrawButtonBackground       | 196 | TSRVViewProperty                 | 154 |

TSRVViewProperty.AutoVScrollBarPosition 156  
 TSRVViewProperty.DarkMode 157  
 TSRVViewProperty.DarkModeUI 157  
 TSRVViewProperty.FooterPen 157  
 TSRVViewProperty.FooterTitleColor 157  
 TSRVViewProperty.FooterTitleFont 158  
 TSRVViewProperty.FreePosPage 158  
 TSRVViewProperty.HeaderFooterShade 158  
 TSRVViewProperty.HeaderPen 159  
 TSRVViewProperty.HeaderTitleColor 159  
 TSRVViewProperty.HeaderTitleFont 159  
 TSRVViewProperty.HintFont 160  
 TSRVViewProperty.HintPrefixText 160  
 TSRVViewProperty.IconStyle 160  
 TSRVViewProperty.MainDocumentShade 160  
 TSRVViewProperty.MainPen 161  
 TSRVViewProperty.MainTitleColor 161  
 TSRVViewProperty.MainTitleFont 161  
 TSRVViewProperty.MarginsPen 162  
 TSRVViewProperty.MarginsRectVisible 162  
 TSRVViewProperty.MouseWheelZoom 162  
 TSRVViewProperty.PageHeight 162  
 TSRVViewProperty.PageWidth 162  
 TSRVViewProperty.ShowScrollHint 163  
 TSRVViewProperty.TableIconDelay 163  
 TSRVViewProperty.TableIconPopupMenu 163  
 TSRVViewProperty.Texts 163  
 TSRVViewProperty.UseTableIcons 164  
 TSRVViewProperty.UseVCLThemes 165  
 TSRVViewProperty.ViewMode 165  
 TSRVViewProperty.WheelStep 165  
 TSRVViewProperty.WorkArea 165  
 TSRVViewProperty.ZoomFont 166  
 TSRVViewProperty.ZoomMax 166  
 TSRVViewProperty.ZoomMenu 166  
 TSRVViewProperty.ZoomMin 166  
 TSRVViewProperty.ZoomMode 166  
 TSRVViewProperty.ZoomModeEdit 167  
 TSRVViewProperty.ZoomModeIN 167  
 TSRVViewProperty.ZoomModeOUT 167  
 TSRVViewProperty.ZoomPanelFont 167  
 TSRVViewProperty.ZoomPanelUsesGradient 168  
 TSRVViewProperty.ZoomPanelVisible 168  
 TSRVViewProperty.ZoomPercent 168  
 TSRVViewProperty.ZoomPercentEdit 169  
 TSRVViewProperty.ZoomPercentIN 169  
 TSRVViewProperty.ZoomPercentOUT 169  
 TSRVWinControl 340  
 TSRVWorkArea 165

TSRVZoomInfo 100

## - U -

Units 290  
     TsrvActionPageFormat 290  
 UnitsPerInchH 100  
     TSRichViewEdit 100  
 UnitsPerInchV 100  
     TSRichViewEdit 100  
 UnitsProgram 68  
     TSRichViewEdit 68  
 Update 222  
     TSRVPrint 222  
 UpdateBuffer 100  
     TSRichViewEdit 100  
 UpdateBufferAt 100  
     TSRichViewEdit 100  
 UpdateCache 185  
     TSRVPageScroll 185  
 UseDrawHyperlinksEvent 69  
     TSRichViewEdit 69  
 UsePageOrders 142  
     TSRVPageProperty 142  
 UsePhysicalOffsets 217  
     TSRVPrint 217  
 UseStyleTemplates 69  
     TSRichViewEdit 69  
 UseTableIcons 164  
     TSRVViewProperty 164  
 UseVCLThemes 165  
     TSRVViewProperty 165  
 UseXPThemes 248  
     TSRVScrollBar 248

## - V -

Version 69  
     TSRichViewEdit 69  
 VerticalAlign 264  
     TSRVCloseButton 264  
 VerticalScrollBarSchemes 233  
     TSRVSkin 233  
 VerticalTabSetSchemes 233  
     TSRVSkin 233  
 ViewMode 165  
     TSRVViewProperty 165  
 ViewProperty 69  
     TSRichViewEdit 69  
 Visibility 264

Visibility 264  
     TSRVCloseButton 264  
 Visible 127, 133, 149, 151, 264, 266, 275  
     TSRVBackgroundProperty 127  
     TSRVCloseButton 264  
     TSRVLineNumberProperty 133  
     TSRVPropertyTBH 149  
     TSRVPropertyTBV 151  
     TSRVTabItem 266  
     TSRVToolButton 275  
 VMaxScrollPos 69  
     TSRichViewEdit 69  
 VPadding 146  
     TSRVPagePositionProperty 146  
 VScrollBar 69  
     TSRichViewEdit 69  
 VScrollBarSchemeIndex 70  
     TSRichViewEdit 70  
 VScrollPos 70  
     TSRichViewEdit 70

## - W -

WheelStep 165  
     TSRVViewProperty 165  
 Width 264  
     TSRVCloseButton 264  
 WorkArea 165  
     TSRVViewProperty 165

## - Z -

ZoomCanvas 100  
     TSRichViewEdit 100  
 ZoomFont 166  
     TSRVViewProperty 166  
 ZoomMax 166  
     TSRVViewProperty 166  
 ZoomMenu 166  
     TSRVViewProperty 166  
 ZoomMin 166  
     TSRVViewProperty 166  
 ZoomMode 166  
     TSRVViewProperty 166  
 ZoomModeEdit 167  
     TSRVViewProperty 167  
 ZoomModelN 167  
     TSRVViewProperty 167  
 ZoomModeOUT 167  
     TSRVViewProperty 167

ZoomPanelFont 167  
     TSRVViewProperty 167  
 ZoomPanelUsesGradient 168  
     TSRVViewProperty 168  
 ZoomPanelVisible 168  
     TSRVViewProperty 168  
 ZoomPercent 168, 313  
     TsrvActionZoom 313  
     TSRVViewProperty 168  
 ZoomPercentEdit 169  
     TSRVViewProperty 169  
 ZoomPercentIN 169  
     TSRVViewProperty 169  
 ZoomPercentOUT 169  
     TSRVViewProperty 169  
 ZoomRectIN 101  
     TSRichViewEdit 101  
 ZoomRectOUT 101  
     TSRichViewEdit 101  
 ZoomToFullPages 101  
     TSRichViewEdit 101