



RichViewActions

RichViewActions © trichview.com

Table of Contents

Part I RichViewActions	16
1..Overview	16
Using RichViewActions	16
Style templates	20
History	21
2..Components	33
TCustomRVFontComboBox	33
Properties	34
TCustomRVFontComboBox.ActionFont	35
TCustomRVFontComboBox.Editor	35
TCustomRVFontListBox	35
Properties	36
TCustomRVFontListBox.ActionFont	37
TCustomRVFontListBox.AllowFocusEditor	37
TCustomRVFontListBox.Editor	37
TRulerItemSelector	37
Properties	38
TCustomRulerItemSelector.Ruler	39
TRVAControlPanel	39
Properties	41
TRVAControlPanel.ActionsEnabled	42
TRVAControlPanel.AddColorNameToHints	42
TRVAControlPanel.AutoDeleteUnusedStyles	42
TRVAControlPanel.ColorDialog	42
TRVAControlPanel.ColorDialogInterface	43
TRVAControlPanel.DefaultColor	43
TRVAControlPanel.DefaultControl	43
TRVAControlPanel.DefaultCustomFilterIndex	43
TRVAControlPanel.DefaultDocParameters	44
TRVAControlPanel.DefaultExt	44
TRVAControlPanel.DefaultFileFormat	44
TRVAControlPanel.DefaultFileName	45
TRVAControlPanel.DefaultMargin	45
TRVAControlPanel.DialogFontName	45
TRVAControlPanel.DialogFontSize	46
TRVAControlPanel.DialogIcons	46
TRVAControlPanel.DialogPosition	47
TRVAControlPanel.DialogZoomPercent	48
TRVAControlPanel.DownloadInterface	48
TRVAControlPanel.FirstPageNumber	48
TRVAControlPanel.Header, Footer	49
TRVAControlPanel.HelpType	49
TRVAControlPanel.HTMLComponent (deprecated)	53
TRVAControlPanel.Language	53
TRVAControlPanel.MetafileCompatibility	53
TRVAControlPanel.PixelBorders	53
TRVAControlPanel.RVFFilter	54
TRVAControlPanel.RVFLocalizable	54
TRVAControlPanel.RVFormatTitle	54
TRVAControlPanel.RVPrint	55

TRVAControlPanel.RVStylesFilter, RVStylesExt.....	55
TRVAControlPanel.SearchScope.....	55
TRVAControlPanel.ShowSoftPageBreaks	56
TRVAControlPanel.SpellInterface.....	56
TRVAControlPanel.TableGridStyle.....	56
TRVAControlPanel.UnitsDisplay	56
TRVAControlPanel.UnitsProgram.....	57
TRVAControlPanel.UseDefaultHelpFile.....	57
TRVAControlPanel.UseHelpFiles	58
TRVAControlPanel.UserInterface.....	58
TRVAControlPanel.UseTextCodePageDialog	59
TRVAControlPanel.UseXPThemes	59
TRVAControlPanel.XMLComponent.....	60
TRVAControlPanel.XMLFilter.....	60
TRVAControlPanel.XMLLocalizable.....	60
Methods	60
TRVAControlPanel.Activate.....	61
TRVAControlPanel.GetRealDialogFontName.....	61
TRVAControlPanel.InitImportPictures, DoneImportPictures, DoImportPicture.....	61
Events	62
TRVAControlPanel.OnAddStyle.....	62
TRVAControlPanel.OnBackgroundChange.....	63
TRVAControlPanel.OnChoosePicture.....	63
TRVAControlPanel.OnCreateForm.....	64
TRVAControlPanel.OnCustomFileOperation.....	64
TRVAControlPanel.OnDownload.....	66
TRVAControlPanel.OnGetActionControlCoords	66
TRVAControlPanel.OnGetHeaderFooterCode.....	67
TRVAControlPanel.OnMarginsChanged.....	67
TRVAControlPanel.OnStyleNeeded.....	68
TRVAControlPanel.OnViewChanged.....	68
TRVAPopupMenu	68
Properties	70
TRVAPopupMenu.ActionList.....	71
TRVAPopupMenu.MaxSuggestionsCount.....	71
Events	71
TRVAPopupMenu.OnLiveSpellAdd.....	71
TRVAPopupMenu.OnLiveSpellGetSuggestions.....	72
TRVAPopupMenu.OnLiveSpellIgnoreAll.....	72
TRVAPopupMenu.OnLiveSpellWordReplace.....	73
TRVFontCharsetComboBox	73
Properties	74
TRVFontCharsetComboBox.AddDefaultCharset.....	75
TRVFontCharsetComboBox.DefaultCharsetCaption	75
TRVFontCharsetComboBox.FontName.....	75
TRVFontComboBox	75
Properties	76
TRVFontComboBox.AutoCharset.....	77
TRVFontComboBox.DropDownWidth.....	77
TRVFontComboBox.ItemHeight.....	77
TRVFontComboBox.Preview.....	78
TRVFontComboBox.SymbolPreviewString.....	78
TRVFontListBox	78
Properties	79
TRVFontListBox.AutoCharset.....	80
TRVFontListBox.ItemHeight.....	80
TRVFontListBox.Preview	80
TRVFontListBox.SymbolPreviewString.....	80

TRVFontSizeComboBox	81
Properties	81
TRVFontSizeComboBox.FontName	82
TRVRuler	82
Properties	83
TCustomRuler.BiDiModeRuler	84
TCustomRuler.Flat	85
TCustomRuler.IndentColor, MarginColor, RulerColor, TickColor, RulerColorPageEnd	85
TCustomRuler.IndentSaturation, MarginSaturation, RulerSaturation	86
TCustomRuler.RulerType	86
TCustomRuler.ScreenRes	86
TCustomRuler.SkinType	87
TCustomRuler.UnitsDisplay, UnitsProgram	87
TCustomRuler.UnitsPixelsPerInch	87
TCustomRuler.Zoom	88
TRVRuler.RichViewEdit	88
Methods	88
TRVRuler.UpdateRulerMargins	89
TRVStyleTemplateComboBox, TRVStyleTemplateListBox	89
Properties	90
TRVStyleTemplateComboBox.*ImageIndex	91
TRVStyleTemplateComboBox.ControlPanel	91
TRVStyleTemplateComboBox.Editor	91
TRVStyleTemplateComboBox.Images	92
TRVStyleTemplateComboBox.ShowAllStyles	92
TRVStyleTemplateComboBox.ShowClearFormat	92
TRVStyleTemplateComboBox.SmartHeadings	92
Methods	93
TRVStyleTemplateComboBox.Localize	93
Events	93
TRVStyleTemplateComboBox.OnClickAllStyles	94
3.Actions	94
File	98
TrvActionCustomFileIO	98
Properties	99
TrvActionCustomFileIO.CustomFilter	100
TrvActionCustomIO	100
Properties	101
TrvActionCustomIO.AutoUpdateFileName	101
TrvActionCustomIO.DialogTitle	102
TrvActionCustomIO.FileName	102
TrvActionCustomIO.InitialDir	102
TrvActionCustomNew	103
Events	103
TrvActionCustomNew.OnNew	103
Properties	104
TrvActionCustomNew.ActionSave	104
TrvActionNew	105
Properties	106
TrvActionNew.StyleTemplates, ApplyStyleTemplates	106
Methods	107
TrvActionNew.Reset	107
TrvActionOpen	108
Properties	109
TrvActionOpen.ActionNew	110
TrvActionOpen.CustomFilter	110
TrvActionOpen.DialogTitle	110

TrvActionOpen.Filter	110
TrvActionOpen.InitialDir	111
TrvActionOpen.LastFilterIndex	111
Methods	111
TrvActionOpen.LoadFile	111
Events	112
TrvActionOpen.OnOpenFile	112
TrvActionExport	112
Properties	113
TrvActionExport.CreateDirectoryForHTMLImages	115
TrvActionExport.Filter	115
Methods	115
TrvActionExport.ExportToFile	116
TrvActionPageSetup	116
Properties	117
TrvActionPageSetup.MarginsUnits	117
TrvActionPageSetup.UnitsPixelsPerInch	118
Events	118
TrvActionPageSetup.OnChange	118
TrvActionPrint	118
Properties	119
TrvActionPrint.Title	120
TrvActionPrintPreview	120
Properties	121
TrvActionPrintPreview.ActionPageSetup	121
TrvActionPrintPreview.ActionPrint	121
TrvActionPrintPreview.Maximized	122
Events	122
TrvActionPrintPreview.OnGetPreviewFormClass	122
TrvActionQuickPrint	122
Properties	123
TrvActionQuickPrint.Title	124
TrvActionSave	124
Properties	125
TrvActionSave.ActionSaveAs	125
TrvActionSave.CreateDirectoryForHTMLImages	126
TrvActionSave.DisableWhenUnmodified	126
TrvActionSave.Documents	126
TrvActionSave.LostFormatWarning	127
TrvActionSave.SuppressNextErrorMessage	127
Methods	127
TrvActionSave.CanCloseDoc	127
TrvActionSave.FindDoc	128
Events	128
TrvActionSave.OnDocumentFileChange	128
TrvActionSave.OnSaving, OnSave	128
TrvActionSaveAs	129
Properties	130
TrvActionSaveAs.ActionSave	131
TrvActionSaveAs.Filter	131
TrvCustomPrintAction	131
Edit	132
TrvActionCharCase	132
TrvActionCopy	133
TrvActionCut	133
TrvActionFind	134
Properties	135
TrvActionFind.ActionReplace	135

TrvActionFind.FindText.....	136
Methods	136
TrvActionFind.CloseDialog.....	136
TrvActionFindNext.....	136
Properties	137
TrvActionFindNext.ActionFind.....	137
TrvActionPaste.....	137
TrvActionPasteAsText.....	138
TrvActionPasteSpecial.....	139
Properties	140
TrvActionPasteSpecial.StoreFileNameInItemName.....	140
Methods	141
TrvActionPasteSpecial.AddFormat.....	141
Events	141
TrvActionPasteSpecial.OnCanPaste.....	141
TrvActionPasteSpecial.OnCustomPaste.....	142
TrvActionPasteSpecial.OnShowing.....	142
TrvActionRedo.....	142
TrvActionReplace.....	143
Properties	143
TrvActionReplace.ActionFind.....	144
TrvActionReplace.FindText, ReplaceText.....	144
TrvActionReplace.ShowReplaceAllSummary.....	145
Methods	145
TrvActionReplace.CloseDialog.....	145
Events	145
TrvActionReplace.OnReplaceAllEnd.....	145
TrvActionReplace.OnReplaceAllStart.....	145
TrvActionReplace.OnReplacing.....	146
TrvActionSelectAll	146
TrvActionUndo.....	146
TrvCustomEditAction	147
Font	148
TrvActionFontAllCaps.....	148
TrvActionFontBackColor	149
TrvActionFontBold.....	149
TrvActionFontColor.....	150
TrvActionFontCustomColor	151
TrvActionFontEx.....	151
Properties	152
TrvActionFontEx.AutoApplySymbolCharset.....	153
TrvActionFontEx.BackColor.....	154
TrvActionFontEx.CharScale.....	154
TrvActionFontEx.CharSpacing.....	154
TrvActionFontEx.FontSizeDouble.....	155
TrvActionFontEx.FontStyleEx.....	155
TrvActionFontEx.PreviewInList.....	155
TrvActionFontEx.SubSuperScriptType.....	155
TrvActionFontEx.UnderlineColor	156
TrvActionFontEx.UnderlineType.....	156
TrvActionFontEx.ValidProperties.....	156
TrvActionFontEx.VShift.....	157
Events	157
TrvActionFontEx.OnShowingDialog.....	157
TrvActionFontGrow	157
Properties	158
TrvActionFontGrow.MaxSize.....	159
TrvActionFontGrowOnePoint.....	159

Properties	159
TrvActionFontGrowOnePoint.MaxSize.....	160
TrvActionFontItalic	160
TrvActionFontOverline.....	161
TrvActionFonts	161
Properties	162
TrvActionFonts.Font.....	163
TrvActionFonts.UserInterface.....	163
TrvActionFontShrink.....	163
Properties	164
TrvActionFontShrink.MinSize.....	165
TrvActionFontShrinkGrow.....	165
Properties	165
TrvActionFontShrinkGrow.Percent.....	166
TrvActionFontShrinkOnePoint.....	166
Properties	167
TrvActionFontShrinkOnePoint.MinSize.....	167
TrvActionFontStrikeout.....	168
TrvActionFontStyle.....	168
TrvActionFontStyleEx.....	169
TrvActionFontUnderline.....	170
TrvActionSSScript.....	170
TrvActionSubscript.....	171
TrvActionSuperscript.....	172
TrvActionTextBiDi.....	172
TrvActionTextLTR.....	173
TrvActionTextRTL.....	174
TrvActionTextStyles.....	175
Paragraph	175
TrvActionAlignCenter.....	176
TrvActionAlignCustomJustify.....	177
Properties	177
TrvActionAlignCustomJustify.LastLineAlignment, UseLastLineAlignment.....	178
TrvActionAlignDistribute.....	178
TrvActionAlignJustify.....	179
TrvActionAlignLeft.....	180
TrvActionAlignment.....	180
TrvActionAlignRight.....	181
TrvActionClearBoth.....	182
TrvActionClearLeft.....	182
TrvActionClearNone.....	183
TrvActionClearRight.....	183
TrvActionClearTextFlow.....	184
TrvActionCustomParaListSwitcher.....	184
Properties	185
TrvActionCustomParaListSwitcher.IndentStep.....	186
TrvActionCustomParaListSwitcher.ListLevels.....	186
TrvActionIndent.....	186
Properties	187
TrvActionIndent.IndentStep.....	188
TrvActionIndentDec.....	188
TrvActionIndentInc.....	189
Properties	189
TrvActionIndentInc.IndentMax.....	190
TrvActionLineSpacing.....	190
TrvActionLineSpacing100.....	191
TrvActionLineSpacing150.....	191
TrvActionLineSpacing200.....	192

TrvActionParaBiDi.....	193
TrvActionParaBorder.....	193
Properties	194
TrvActionParaBorder.Background.....	195
TrvActionParaBorder.Border.....	195
TrvActionParaBorder.UserInterface.....	196
TrvActionParaBorder.ValidProperties.....	196
Events	196
TrvActionParaBorder.OnShowingDialog.....	196
TrvActionParaBullets.....	197
TrvActionParaColor.....	197
TrvActionParaColorAndPadding.....	198
Properties	199
TrvActionParaColorAndPadding.Padding.....	199
TrvActionParaColorAndPadding.UsePadding.....	200
TrvActionParaCustomColor.....	200
TrvActionParagraph.....	200
Properties	201
TrvActionParagraph.Alignment.....	202
TrvActionParagraph.DeleteAllTabs.....	202
TrvActionParagraph.FirstIndent.....	203
TrvActionParagraph.KeepLinesTogether.....	203
TrvActionParagraph.KeepWithNext.....	203
TrvActionParagraph.LastLineAlignment.....	204
TrvActionParagraph.LeftIndent.....	204
TrvActionParagraph.LineSpacing.....	204
TrvActionParagraph.LineSpacingType.....	205
TrvActionParagraph.OutlineLevel.....	205
TrvActionParagraph.RightIndent.....	205
TrvActionParagraph.SpaceAfter.....	205
TrvActionParagraph.SpaceBefore.....	206
TrvActionParagraph.Tabs.....	206
TrvActionParagraph.TabsToDelete.....	206
TrvActionParagraph.UserInterface.....	207
TrvActionParagraph.ValidProperties.....	207
Events	207
TrvActionParagraph.OnShowingDialog.....	207
TrvActionParaList.....	208
Properties	208
TrvActionParaList.IndentStep.....	209
TrvActionParaList.UpdateAllActionsOnForm.....	209
TrvActionParaList.ActionParaNumbering.....	210
TrvActionParaList.ActionParaBullets.....	210
TrvActionParaLTR.....	210
TrvActionParaNumbering.....	211
TrvActionParaRTL.....	212
TrvActionParaStyles.....	212
TrvActionWordWrap.....	213
Styles	214
TrvActionStyleTemplates.....	214
Properties	215
TrvActionStyleTemplates.Images.....	216
TrvActionStyleTemplates.StandardStyleTemplates.....	216
TrvActionStyleTemplates.*StyleImageIndex.....	217
TrvActionAddStyleTemplate.....	217
Properties	218
TrvActionAddStyleTemplate.ActionStyleTemplates.....	218
TrvActionStyleInspector.....	218

Properties	219
TrvActionStyleInspector.Images	220
TrvActionStyleInspector.FontImageIndex, ParaImageIndex	220
Methods	220
TrvActionStyleInspector.UpdateInfo	220
Events	220
TrvActionCustomInfoWindow.OnShowing	221
TrvActionClearFormat	221
TrvActionClearTextFormat	221
Insert	222
TrvActionBookmarks	223
Properties	223
TrvActionBookmarks.AllowedCharacters	224
TrvActionBookmarks.ScrollToCenter	224
TrvActionInsertCaption	225
Properties	226
TrvActionInsertCaption.ExcludeLabel	226
TrvActionInsertCaption.InsertAbove	227
TrvActionInsertCustomPageNumber	227
Properties	227
TrvActionInsertCustomPageNumber.NumberType	228
TrvActionInsertEquation	228
Properties	229
TrvActionInsertEquation.UserInterface	230
TrvActionInsertEquation.Expression	230
TrvActionInsertFile	230
Properties	231
TrvActionInsertFile.Filter	232
Methods	232
TrvActionInsertFile.InsertFile	232
TrvActionInsertHLine	233
Properties	233
TrvActionInsertHLine.Color	234
TrvActionInsertHLine.Style	234
TrvActionInsertHLine.Width	234
TrvActionInsertHyperlink	234
Properties	236
TrvActionInsertHyperlink.ActionStyleTemplates	237
TrvActionInsertHyperlink.AutoAddHyperlinkStyleTemplate	237
TrvActionInsertHyperlink.BackColor	237
TrvActionInsertHyperlink.Color	238
TrvActionInsertHyperlink.Cursor	238
TrvActionInsertHyperlink.HoverBackColor	238
TrvActionInsertHyperlink.HoverColor	239
TrvActionInsertHyperlink.HoverEffects	239
TrvActionInsertHyperlink.HoverUnderlineColor	239
TrvActionInsertHyperlink.ScrollToCenter	240
TrvActionInsertHyperlink.SetToPictures	240
TrvActionInsertHyperlink.SpaceFiller	240
TrvActionInsertHyperlink.Style	240
TrvActionInsertHyperlink.StyleTemplateName	241
TrvActionInsertHyperlink.UnderlineColor	241
TrvActionInsertHyperlink.ValidProperties	241
Methods	242
TrvActionInsertHyperlink.DetectURL	242
TrvActionInsertHyperlink.EncodeTarget	243
TrvActionInsertHyperlink.GetHyperlinkStyleNo	243
TrvActionInsertHyperlink.GetNormalStyleNo	244

TrvActionInsertHyperlink.GoToLink.....	244
TrvActionInsertHyperlink.SetTags	245
TrvActionInsertHyperlink.TerminateHyperlink.....	245
Events	246
TrvActionInsertHyperlink.OnApplyHyperlinkToItem.....	246
TrvActionInsertHyperlink.OnGetHyperlinkTargetFromItem.....	246
TrvActionInsertHyperlink.OnHyperlinkForm.....	246
TrvActionInsertNumber.....	247
TrvActionInsertNumSequence.....	248
Properties	248
TrvActionInsertNumSequence.NumberType.....	249
TrvActionInsertNumSequence.SeqName.....	249
TrvActionInsertNumSequence.UseDefaults	249
TrvActionInsertPageBreak.....	250
TrvActionInsertPageCount.....	250
TrvActionInsertPageNumber	251
TrvActionInsertPicture.....	251
Properties	252
TrvActionInsertPicture.BackgroundColor.....	253
TrvActionInsertPicture.BorderWidth, BorderColor.....	253
TrvActionInsertPicture.DefaultExt.....	254
TrvActionInsertPicture.Filter.....	254
TrvActionInsertPicture.MaxImageSize.....	254
TrvActionInsertPicture.OuterHSpacing, OuterVSpacing	255
TrvActionInsertPicture.Spacing	255
TrvActionInsertPicture.StoreFileNameInItemName.....	255
TrvActionInsertPicture.VAlign.....	255
Events	256
TrvActionInsertPicture.OnInserting.....	256
TrvActionInsertSymbol.....	256
Properties	257
TrvActionInsertSymbol.CharCode.....	257
TrvActionInsertSymbol.FontName.....	258
TrvActionInsertSymbol.Protection	258
TrvActionInsertSymbol.UseCurrentFont.....	258
TrvActionInsertText.....	258
Events	259
TrvActionInsertText.OnInsertText.....	259
Notes and text boxes	259
TrvActionCustomInsertSidenote.....	260
Properties	260
TrvActionCustomInsertSidenote.BoxPosition	261
TrvActionCustomInsertSidenote.BoxProperties.....	262
TrvActionCustomInsertSidenote.StyleTemplateName.....	262
TrvActionEditNote.....	262
Properties	264
TrvActionEditNote.JumpToNextNote.....	264
Events	265
TrvActionEditNote.OnStartEditNote.....	265
TrvActionInsertEndnote.....	265
TrvActionInsertFootnote.....	266
TrvActionInsertNote.....	267
Properties	267
TrvActionInsertNote.ActionEditNote.....	268
TrvActionInsertNote.Font.....	268
TrvActionInsertSidenote.....	269
Properties	270
TrvActionInsertSidenote.RefStyleTemplateName.....	271

TrvActionInsertTextBox	271
Tables	272
TrvActionInsertTable.....	273
Properties	274
TrvActionInsertTable.BackgroundImage.....	275
TrvActionInsertTable.BackgroundProperties	275
TrvActionInsertTable.BestWidth.....	276
TrvActionInsertTable.BorderColor.....	276
TrvActionInsertTable.BorderHSpacing.....	276
TrvActionInsertTable.BorderLightColor.....	276
TrvActionInsertTable.BorderStyle.....	277
TrvActionInsertTable.BorderVSpacing.....	277
TrvActionInsertTable.BorderWidth	277
TrvActionInsertTable.CellBorderColor.....	277
TrvActionInsertTable.CellBorderLightColor.....	278
TrvActionInsertTable.CellBorderStyle.....	278
TrvActionInsertTable.CellBorderWidth.....	278
TrvActionInsertTable.CellHPadding,CellVPadding.....	278
TrvActionInsertTable.CellHSpacing.....	279
TrvActionInsertTable.CellVSpacing.....	279
TrvActionInsertTable.ColBandSize, RowBandSize.....	279
TrvActionInsertTable.ColCount.....	280
TrvActionInsertTable.Color.....	280
TrvActionInsertTable.HeadingRowCount.....	280
TrvActionInsertTable.HOutermostRule.....	280
TrvActionInsertTable.HRuleColor	280
TrvActionInsertTable.HRuleWidth.....	281
TrvActionInsertTable.ItemText.....	281
TrvActionInsertTable.RowCount.....	281
TrvActionInsertTable.RowsKeepTogether.....	281
TrvActionInsertTable.RowsVAlign.....	282
TrvActionInsertTable.TableOptions	282
TrvActionInsertTable.TablePrintOptions	282
TrvActionInsertTable.VisibleBorders.....	282
TrvActionInsertTable.VOutermostRule.....	282
TrvActionInsertTable.VRuleColor.....	283
TrvActionInsertTable.VRuleWidth	283
TrvActionInsertTable's row and column colors.....	283
Methods	283
TrvActionInsertTable.ShowTableSizeDialog.....	284
Events	284
TrvActionInsertTable.OnCloseTableSizeDialog.....	284
TrvActionInsertTable.OnInserting.....	284
TrvActionTableCell.....	284
TrvActionTableCellAllBorders.....	285
TrvActionTableCellBorder.....	286
TrvActionTableCellBottomBorder.....	286
TrvActionTableCellLeftBorder	287
TrvActionTableCellNoBorders.....	288
TrvActionTableCellRightBorder	289
TrvActionTableCellRotation.....	289
TrvActionTableCellRotationNone.....	290
TrvActionTableCellRotation90.....	291
TrvActionTableCellRotation180	291
TrvActionTableCellRotation270	292
TrvActionTableCellTopBorder.....	293
TrvActionTableCellVAlign.....	293
TrvActionTableCellVAlignBottom.....	294

TrvActionTableCellVAlignDefault.....	295
TrvActionTableCellVAlignMiddle.....	295
TrvActionTableCellVAlignTop.....	296
TrvActionTableDeleteCols.....	297
TrvActionTableDeleteRows.....	297
TrvActionTableDeleteTable.....	298
TrvActionTableGrid.....	298
TrvActionTableInsertColLeft.....	299
TrvActionTableInsertColRight.....	299
TrvActionTableInsertRowsAbove.....	300
TrvActionTableInsertRowsBelow.....	301
Properties.....	301
TrvActionTableInsertRowsBelow.AllowMultiple.....	302
TrvActionTableMergeCells.....	302
TrvActionTableMultiCellAttributes.....	303
TrvActionTableProperties.....	303
Properties.....	304
TrvActionTableProperties.ActionInsertTable, UpdateAllInsertTableActions.....	305
TrvActionTableProperties.BackgroundGraphicFilter.....	305
TrvActionTableProperties.DefaultChecked, DefaultPersistent.....	305
TrvActionTableRCBase.....	306
TrvActionTableSelectCell.....	307
TrvActionTableSelectCols.....	307
TrvActionTableSelectRows.....	308
TrvActionTableSelectTable.....	308
TrvActionTableSort.....	309
TrvActionTableSplit.....	310
TrvActionTableSplitCells.....	310
TrvActionTableToText.....	311
Spelling check and thesaurus.....	312
TrvActionSpellingCheck.....	312
TrvActionThesaurus.....	312
Other actions.....	313
TrvAction.....	313
Properties.....	314
TrvAction.Control.....	314
TrvActionBackground.....	315
Properties.....	316
TrvActionBackground.CanChangeMargins.....	316
TrvActionBackground.ImageFileName.....	317
Events.....	317
TrvActionBackground.OnChange.....	317
TrvActionColor.....	317
TrvActionUserDefinedColor.....	318
Properties.....	318
TrvActionUserDefinedColor.UseOpacity.....	319
Events.....	319
TrvActionUserDefinedColor.OnColorSelected.....	319
TrvActionUserDefinedColor.OnGetColor.....	320
TrvActionCustomColor.....	320
Properties.....	320
TrvActionCustomColor.CallerControl.....	321
TrvActionCustomColor.Color.....	321
TrvActionCustomColor.Opacity.....	322
TrvActionCustomColor.UserInterface.....	322
Events.....	323
TrvActionCustomColor.OnShowColorPicker.....	323
TrvActionCustomColor.OnHideColorPicker.....	323

TrvActionEvent.....	323
Events	324
TrvActionEvent.OnExecute.....	324
TrvActionEvent.OnUpdate.....	324
TrvActionFillColor.....	324
Properties	325
TrvActionFillColor.AllowApplyingTo.....	326
TrvActionHide.....	326
TrvActionItemProperties.....	326
Properties	327
TrvActionItemProperties.ActionInsertHLine.....	328
TrvActionItemProperties.ActionInsertTable, UpdateAllInsertTableActions.....	328
TrvActionItemProperties.BackgroundGraphicFilter.....	329
TrvActionItemProperties.DefaultChecked, DefaultPersistent.....	329
TrvActionItemProperties.GraphicFilter	330
TrvActionItemProperties.StoreImageFileName.....	330
Events	330
TrvActionItemProperties.OnCanApply.....	330
TrvActionItemProperties.OnCustomItemPropertiesDialog.....	331
TrvActionRemoveHyperlinks	331
Properties	332
TrvActionRemoveHyperlinks.ActionInsertHyperlink.....	332
TrvActionRemovePageBreak.....	332
TrvActionShowSpecialCharacters.....	333
Properties	334
TrvActionShowSpecialCharacters.ShowCheckpoints.....	334
TrvActionVAlign.....	335
TrvCustomAction	335
Properties	336
TrvCustomAction.Caption.....	336
TrvCustomAction.ControlPanel	337
TrvCustomAction.Disabled.....	337
TrvCustomAction.Hint.....	337
Methods	337
TrvCustomAction.GetControlPanel	338
TrvCustomEditorAction	338
Properties	338
TrvCustomEditorAction.Form.....	339
TrvCustomEditorAction.SubDocEditor	339
Events	339
TrvCustomEditorAction.OnFormCreate.....	340
TrvCustomEditorAction.OnShowing, OnHide.....	340
4. Localization of RichViewActions	342
RVALocalize unit	343
Procedures and functions	344
Functions for progress messages.....	345
RVA_EnumLanguages procedure.....	346
RVA_FillLanguageList procedure.....	346
RVA_GetCharset function.....	347
RVA_GetHelpFile function.....	347
RVA_GetLanguageName function	348
RVA_GetS function	348
RVA_SetHelpFile procedure.....	349
RVA_SwitchLanguage procedure.....	349
RVA_TranslateUnits procedure.....	349
Types	350
TRVALanguageName	350

TRVLocString, TRVLocStrings, TRVLocStringList	350
RichViewActions unit	351
Procedures and functions	351
RVA_ChooseLanguage function	351
RVA_GetColorName function	352
RVA_LocalizeForm procedure	352
Other units	353
GetAddictSpellLanguage, GetAddictThesLanguage functions	353
RVLocalizeRuler procedure	353
RVGetHelpKeyword and RVSetHelpKeyword	354
5..Interfaces for third-party components	354
Download interfaces	355
TRVACIDownloadInterface	355
TRVACustomDownloadInterface	356
TRVAIndyDownloadInterface	356
Spelling check interfaces	357
TRVSpellInterface	358
TRVAddictSpellInterface	358
TRVAASpellInterface	359
TRVACustomSpellInterface	359
TRVADXSpellInterface	360
TRVAHunSpellInterface	361
TRVAPolarSpellInterface	361
Color dialog interfaces	362
6..Third-party and additional tools	362
Components	363
Spelling checkers	364
Addict 3 and 4	364
ASpell	365
ExpressSpellChecker	366
HunSpell	367
Polar SpellChecker ActiveX	368
Downloaders	369
CleverComponents	369
Indy	370
File readers and writers	370
RichViewXML	370
RvHtmlImporter (deprecated)	370
RvHtmlViewImporter (deprecated)	371
User interface	372
SpTBXLib	372
TNT Controls	372
TBX	373
Toolbar2000	373
Toolbar Images	373
TRichView Icons	374
GlyFX toolbar images	376
Glyfz toolbar images	377
Fugue Icons toolbar images	377
FamFamFam Silk Icons toolbar images	378
7..Procedures and Functions	378
RichViewActions unit	379
GoToCheckpoint function	379
PasteHTML procedure (deprecated)	380
RVA_ChooseStyle function	380
RVA_EditorControlFunctionDef function	380
RVA_ConvertTo* procedures	380

RvHtmlHelp function.....	381
RVARibbonUtils unit	381
FillActionClientContents procedure.....	381
RemoveEllipsis, RemoveEllipsisFromRibbon procedures.....	381
LocalizeRibbonPage procedure.....	382
SetRulerColors procedure.....	382
Other units	382
LoadCSV function.....	382
MarkSubString functions.....	383
NormalizeRichView procedure.....	384
RVChangeCharCase, RVGetCharCase procedures	385
8..Types	385
TRulerUnits	386
TRVAEditEvent, TRVAEditEvent2	386
TRVAHFInfo	386
TRVASpellString, TRVASpellStrings, TRVASpellStringList	387
TrvFileExportFilter, TrvFileSaveFilter	388
TRVFileFormatComponent	389
TrvFileImportFilter, TrvFileOpenFilter	389
TRVShowFormEvent	390
9..Global variables	390
MainRVAControlPanel	390
RVA_EditForceDefControl	391
RVA_EditorControlFunction	391
ShowUntranslatedControls	392
10..How to	392
How to change RVF and XML format name	392
 Index	 393

1 RichViewActions

RichViewActions is a set of actions and components for Delphi, C++Builder and Lazarus (for Windows and Linux) allowing you to create a user interface for the TRichView and ScaleRichView editors. Actions can be assigned to command buttons, menu items, toolbar buttons. You can use the standard Delphi/Lazarus components or your favorite third-party components if they support actions.

The actions require no programming, just add a new action in the action manager, assign it to a component, and it will do all the work automatically.

RichViewActions homepage: www.trichview.com/resources/actions/.

Minimal supported version of Delphi: 5

Minimal supported version of C++Builder: 6.

This help file includes the descriptions of all actions, main components, and utility functions. It does not describe visual controls installed with the package (the most of them are self-explanatory).

See also:

- Overview ⁽¹⁶⁾
- What's new? ⁽²¹⁾

1.1 Overview

- Using RichViewActions ⁽¹⁶⁾
- Style Templates ⁽²⁰⁾
- What's new ⁽²¹⁾

1.1.1 Using RichViewActions

Overview

All actions have Control ⁽³¹⁴⁾ property of TCustomRVControl type. If this property is not assigned, actions work with the focused control (or GetControlPanel ⁽³³⁸⁾.DefaultControl ⁽⁴³⁾ component).

All actions have Disabled ⁽³³⁷⁾ property (Boolean). If it is *True*, all actions are disabled. If it is *False* (default), actions update their Enabled property automatically.

TRVAControlPanel ⁽³⁹⁾ component has a set of properties affecting all (or many) actions.

TRVAPopupMenu ⁽⁶⁸⁾ is a popup menu for TCustomRichViewEdit that maintains items itself. Set its ActionList ⁽⁷¹⁾ property to the action list containing RichViewActions. Assign this component to the editor's PopupMenu. Do not add items in this menu manually – they all will be cleared (you can add them on each call of OnPopup, though). The menu supports live spelling, see below.

Preparing for the actions

Right-click RichViewEdit in Delphi/C++Builder/Lazarus, choose "Settings" in the context menu, select "Allow adding styles dynamically", check options for saving and loading background and layout.

Include *rvoCtrlJumps* in RichViewEdit.EditorOptions.

How to use

Put an ActionList component on the form. Link it with some image list. Double click the ActionList. Click "New Standard Actions" on the component editor toolbar. Select the action and click OK. If the linked image-list is 16x16, an action image will be added automatically. However, we recommend to use high-quality images ⁽³⁷⁴⁾ instead of default images.

Alternatively, you can use existing datamodules:

- **dmActions.pas** contains an ActionList and an ImageList, with simple 16-color images;
- **dmActionImages1.pas** contains ImageLists with high-quality toolbar images, set #1, both normal and disabled versions (for Delphi/C++Builder 2009 and newer);
- **dmActionImages2.pas** the same, set #2;
- **dmactionimageslaz1.pas** the same, set #1 (for Lazarus)
- **dmactionimageslaz2.pas** the same, set #2 (for Lazarus)

For newer versions of Delphi and Lazarus, there are datamodules containing multi-resolution ImageLists:

- **dmActionsVirtualImageLists.pas** contains a virtual ImageList (for Delphi/C++Builder 10.3+), the actual images are in dmActionsImageCollection1.pas (set #1) and dmActionsImageCollection2.pas (set #2)
- **dmactionsimagesmultireslaz1.pas** the same, set #1 (for Lazarus 2+)
- **dmactionsimagesmultireslaz2.pas** the same, set #2 (for Lazarus 2+)

If you will not modify these units, you can include them in your project directly. But it's highly recommended to create a copy of them (under new names) and use the copies instead. Important: if you access actions in OnCreate event of some form, this form must be created after the datamodule (the form creation order may be specified in the project options).

Localization

Several languages are available for translations of user interface. Some operations must be performed even if you use only one language. See Localization of RichViewActions ⁽³⁴²⁾.

Ruler

RichViewActions include TRuler component by Pieter Zijlstra and TRVRuler ⁽⁸²⁾ component inherited from it.

Create a TRVRuler ⁽⁸²⁾ component, place on the form above the editor, assign RVRuler.RichViewEdit ⁽⁸⁸⁾ to this editor.

In TRVAControlPanel.OnMarginsChanged ⁽⁶⁷⁾ call RVRuler.UpdateRulerMargins ⁽⁸⁹⁾.

The ruler supports units ⁽⁸⁷⁾: inches, centimeters, millimeters, picas, pixels and points.

If you want to remove support for tab stops, set ruler's MaxTabs property to 0.

Actions requiring a special attention

It's recommended to assign Control ⁽³¹⁴⁾ property for the following actions:

- TrvActionStyleInspector ⁽²¹⁸⁾;

- TrvActionEditNote⁽²⁶²⁾.

Otherwise, it will be assigned automatically when the action is executed for the first time.

Some additional code may be required to implement UI for a note editor of TrvActionEditNote⁽²⁶²⁾.

Files

File-related actions (TrvActionNew⁽¹⁰³⁾, TrvActionOpen⁽¹⁰⁸⁾, TrvActionSave⁽¹²⁴⁾, TrvActionSaveAs⁽¹²⁹⁾) are linked together. The most of them cannot work without others. These actions keep track of all opened files and can be used even for multiple RichViewEdits (for example, in MDI application) without any additional effort from the programmer. These actions cannot be used for TDBRichViewEdit.

TrvActionExport⁽¹¹²⁾, TrvActionInsertFile⁽²³⁰⁾ are independent – they are not linked with other actions, they just allow to export/insert a file.

Printing

TRVAControlPanel⁽³⁹⁾ has RVPrint⁽⁵⁵⁾: TRVPrint property. All printing actions (TrvActionPrintPreview⁽¹²⁰⁾, TrvActionQuickPrint⁽¹²²⁾, TrvActionPrint⁽¹¹⁸⁾) use it.

It is recommended to assign it to some TRVPrint component.

If this property is not assigned, these actions search TRVPrint component on the same form as the target editor. If it is not found, they create a temporal TRVPrint component.

Besides, these actions use another property of TRVAControlPanel⁽³⁹⁾ – ShowSoftPageBreaks⁽⁵⁶⁾: Boolean. If True (default), soft page breaks are shown when possible (after print preview or printing – until the next change in the document).

TrvActionPageSetup⁽¹¹⁶⁾ is a special printing action. It does not require RichViewEdit, but requires RVPrint⁽⁵⁵⁾ assigned to TRVAControlPanel⁽³⁹⁾.

Color

All the main coloring actions (TrvActionFontColor⁽¹⁵⁰⁾, TrvActionFontBackColor⁽¹⁴⁹⁾, TrvActionParaColor⁽¹⁹⁷⁾, TrvActionColor⁽³¹⁷⁾) can work in three modes, depending on UserInterface⁽³²²⁾ property:

- *rvacAdvanced* (default) – on execution, a special non-modal color dialog pops up; it pops up at the position calculated by the coordinates of control that caused this action to execute (ActionComponent – button, for example) or by mouse coordinates;
- *rvacColorDialog* – on execution, a color is chosen with standard color dialog;
- *rvacNone* – no user interface, action just applies Color property.

This property gives you an ability to implement a color combo (you need two actions – one for applying the last chosen color (UserInterface⁽³²²⁾ = *rvacNone*) and one for displaying a color dialog). Or you can implement your own interface for choosing colors.

All coloring actions have OnShowColorPicker⁽³²³⁾ and OnHideColorPicker⁽³²³⁾ events helping to implement a custom color combo.

Formatting

TrvActionFontGrow⁽¹⁵⁷⁾ and TrvActionFontShrink⁽¹⁶³⁾ change the size of selected font by the given number of percents (Percent⁽¹⁶⁶⁾ property). If you change its value, do not forget to update hints for the actions.

TrvActionFontGrow⁽¹⁵⁷⁾, TrvActionFontGrowOnePoint⁽¹⁵⁹⁾ have MaxSize property, TrvActionFontShrink⁽¹⁶³⁾ TrvActionFontShrinkOnePoint⁽¹⁶⁶⁾ have MinSize property.

TrvActionParagraph⁽²⁰⁰⁾ can work in two modes, depending on UserInterface⁽²⁰⁷⁾: Boolean property:

- *True* (default) – the action gets values of its properties from RichViewEdit, displays a dialog for modifying them, then apply them to the selection;
- *False* – the action applies its properties to the selection, without displaying a dialog.

TrvActionFontEx⁽¹⁵¹⁾ is similar to TrvActionParagraph⁽²⁰⁰⁾, it can also work in two modes.

Live spelling check

TRVAPopupMenu supports live spelling.

You have the following options:

- assign a spelling checker interface component⁽³⁵⁷⁾ to TRVAControlPanel⁽³⁹⁾.SpellInterface⁽⁵⁶⁾ property, and all commands (suggestions, "add to dictionary" and "ignore all", etc.) will be added in the menu automatically;
- use the menu events: OnLiveSpellGetSuggestions⁽⁷²⁾, OnLiveSpellIgnoreAll⁽⁷²⁾, OnLiveSpellAdd⁽⁷¹⁾; if your spellchecker can store a user choice to generate better suggestions in future, you can also process OnLiveSpellWordReplace⁽⁷³⁾.

RichViewActions do not process OnSpellingCheck event, you still need to do it yourself, see the commented code in Unit3.pas.

Bidirectional text

TrvActionParaLTR⁽²¹⁰⁾, TrvActionParaRTL⁽²¹²⁾ – set default bidi mode for the selected paragraphs.

TrvActionTextLTR⁽¹⁷³⁾, TrvActionTextRTL⁽¹⁷⁴⁾ – set bidi mode for the selected text.

Before using these actions set RichViewEdit.BiDiMode either to *rvbdLeftToRight* or *rvbdRightToLeft* (do not use for BiDiMode=*rvbdUnspecified*!)

Providing help files

RichViewActions do not include help files for users. However, if you create help files for RichViewActions dialogs, RichViewActions can use them.

To enable using help files, assign RVAControlPanel.UseHelpFiles⁽⁵⁸⁾ = *True*.

Each UI language can have its own help file, you can specify in in RVA_SetHelpFile⁽³⁴⁹⁾ procedure.

If these help files are not defined, RichViewActions may use the application default help file (Application.HelpFile), if RVAControlPanel.UseDefaultHelpFile⁽⁵⁷⁾ = *True*.

You can use either HelpContext or HelpKeyword of RichViewActions forms, see RVAControlPanel.HelpType⁽⁴⁹⁾.

The list of HelpContexts and HelpKeywords is in the topic about RVAControlPanel.HelpType⁽⁴⁹⁾. The default HelpKeywords are in English. If you want to modify them, use RVSetHelpKeyword procedure.

1.1.2 Style templates

Several actions work differently depending on the value of UseStyleTemplates property of the target editor.

TrvActionNew⁽¹⁰³⁾, TrvActionOpen⁽¹⁰⁸⁾

If Editor.UseStyleTemplates=*True*, TrvActionNew resets the Editor.Style's TextStyles and ParaStyles according to StyleTemplates⁽¹⁰⁶⁾, and clears its ListStyles.

If Editor.UseStyleTemplates=*False*, TrvActionNew does not reset styles; it may only delete unused styles (if GetControlPanel⁽³³⁸⁾.AutoDeleteUnusedStyles⁽⁴²⁾=*True*). If you want to reset styles, use OnNew⁽¹⁰³⁾ event.

TrvActionOpen uses its linked⁽¹¹⁰⁾ TrvActionNew to perform the same operations before loading.

TrvActionPasteSpecial⁽¹³⁹⁾

If Editor.UseStyleTemplates=*True*, the action displays additional options for pasting RTF and RVF documents:

- "apply styles of the target document"
- "keep styles and appearance"
- "keep appearance, ignore styles".

The action temporarily changes Editor.StyleTemplateInsertMode according to the chosen option.

TrvActionInsertHyperlink⁽²³⁴⁾

If Editor.UseStyleTemplates=*True*, the action applies StyleTemplateName⁽²⁴¹⁾ to hyperlinks (or clears a style template when it converts hyperlinks to a plain text). The dialog window allows choosing a style template to apply to a hyperlink.

If Editor.UseStyleTemplates=*False*, the action applies its properties listed in ValidProperties⁽²⁴¹⁾ (or applies properties of Editor.Style.TextStyles[0] when it converts hyperlinks to a plain text). The dialog window allows choosing colors and effects to apply to a hyperlink.

TrvActionRemoveHyperlinks⁽³³¹⁾ is affected as well.

TrvActionClearFormat⁽²²¹⁾, TrvActionClearTextFormat⁽²²¹⁾

If Editor.UseStyleTemplates=*True*, the actions clear formatting according to style templates (applying "Normal" to paragraphs, clearing styles from normal text, applying "Hyperlink" to hyperlinks, resetting additional attributes).

If Editor.UseStyleTemplates=*False*, the actions apply Editor.Style.TextStyles[0] and Editor.Style.ParaStyles[0].


TrvActionStyleTemplates⁽²¹⁴⁾, TrvActionAddStyleTemplate⁽²¹⁷⁾

These actions are enabled only if `Editor.UseStyleTemplates=True`.

TrvActionInsertFootnote⁽²⁶⁶⁾, TrvActionInsertEndnote⁽²⁶⁵⁾, TrvActionInsertSidenote⁽²⁶⁹⁾

If `Editor.UseStyleTemplates=True`, `TrvActionInsertFootnote`⁽²⁶⁶⁾ inserts footnote characters using "footnote reference" style template, and formats footnote text using "footnote text" style template. `TrvActionInsertEndnote`⁽²⁶⁵⁾ uses "endnote reference" and "endnote text" style templates, `TrvActionInsertSidenote`⁽²⁶⁹⁾ uses `RefStyleTemplateName`⁽²⁷¹⁾ ("Sidenote Reference" by default) and `StyleTemplateName`⁽²⁶²⁾ ("Sidenote Text" by default) style templates.

"footnote reference", "footnote text", "endnote reference", "endnote text" are included in `TrvActionNew.StyleTemplates`⁽¹⁰⁶⁾ by default. "Sidenote Reference" and "Sidenote Text" are not included.

 **ScaleRichView:** `TsrvActionInsertFootnote`, `TsrvActionInsertEndnote`, `TsrvActionInsertSidenote` use the same style templates.

TrvActionInsertCaption⁽²²⁵⁾

`TrvActionInsertCaption` uses "caption" style template. It is included in `TrvActionNew.StyleTemplates`⁽¹⁰⁶⁾ by default.

1.1.3 History

▼ Changes in version 15 (after TRichView 24)

Compatibility issues

- `TrvActionInsertTable`⁽²⁷³⁾. `BackgroundStyle` property is removed and replaced by `BackgroundProperties`⁽²⁷⁵⁾.

Icons in dialogs

Ability to use Google Icons, new property: `TRVAControlPanel.DialogIcons`⁽⁴⁶⁾.

Background

- `TRVActionBackground`⁽³¹⁵⁾ and `TrvActionItemProperties`⁽³²⁶⁾ use new dialog windows for changing properties of background of documents, tables, and cells.

▼ Changes in version 14 (between TRichView 23 and 24)

Compatibility issues

- Changes in packages for Lazarus: `rvrichviewactionslaz_dsgn.lpk` is removed. Install `rvrichviewactionslaz.lpk` instead (this package is "runtime + designtime")

- TrvActionParaBullets use Unicode characters in bullets in list templates (instead of characters of "Symbol" and "Wingdings" fonts)


RAD Studio 13 Florence

Delphi and C++Builder 13 are supported.

Lazarus for Linux

RichViewActions can be used in Lazarus for Linux.


New component

 TRVFontListBox (analog of TRVFontComboBox⁽⁷⁵⁾). Internally, it is used in the font dialog⁽¹⁵¹⁾ for Lazarus for Linux.

Changes

- dialogs are redesigned to look better both for Windows and Linux
- widths of all office radio buttons are calculated optimally (depending on the image and text widths), if they are arranged in a single row.
- HTML-style bullet and numbering dialog is obsolete and removed.
- TRVAControlPanel⁽³⁹⁾.DialogFontName⁽⁴⁵⁾ now contains a list of font names in order of preference. This property is used only in the Windows version of RichViewActions. The default value is changed.
- new property TRVAControlPanel⁽³⁹⁾.DialogFontNameLin⁽⁴⁵⁾ is used in the Linux version of RichViewActions.
- new value for TRVAControlPanel⁽³⁹⁾.DialogPosition⁽⁴⁷⁾: *rvafpEditorFormCenter*.
- TrvActionPasteSpecial⁽¹³⁹⁾ supports pasting text as Markdown
- TRVActionBackground⁽³¹⁵⁾ changed BackgroundPicture, not BackgroundBitmap.

▼ Changes in version 13 (between TRichView 22 and 23)

 New TRVSpellInterface⁽³⁵⁸⁾ component allows using TRVSpellChecker (included in TRichView) Support of "Windows 64-bit (Modern)" platform in C++Builder 12+.

▼ Changes in version 12 (between TRichView 21 and 22)

Delphi and C++Builder 12 Athens are supported.

TrvActionInsertPicture⁽²⁵¹⁾ allows inserting multiple images.

Support of decimal tab alignment (in TrvActionParagraph⁽²⁰⁰⁾, TRVRuler⁽⁸²⁾ and TRVRulerItemSelector⁽³⁷⁾).

Option to turn on/off smooth image resizing in TrvActionItemProperties⁽³²⁶⁾.

New properties of TRVAControlPanel⁽³⁹⁾:

- TRVAControlPanel⁽³⁹⁾.DialogPosition⁽⁴⁷⁾ – position of dialog windows
- TRVAControlPanel⁽³⁹⁾.DialogZoomPercent⁽⁴⁸⁾ allows scaling all dialog windows

New action: TrvActionUserDefinedColor⁽³¹⁸⁾.

New events: TrvActionFontEx⁽¹⁵¹⁾.OnShowingDialog⁽¹⁵⁷⁾, TrvActionParagraph⁽²⁰⁰⁾.OnShowingDialog⁽²⁰⁷⁾, TrvActionParaBorder⁽¹⁹³⁾.OnShowingDialog⁽¹⁹⁶⁾. They allow initializing dialogs with predefined values, instead of using attributes of the selected fragment.

New event: TrvActionSave⁽¹²⁴⁾.OnSave⁽¹²⁸⁾, it occurs on successful saving.

▼ Changes in version 11 (between TRichView 20 and 21)

Compatibility issues

ImagePrefix, FileTitle, SaveOptions properties of TrvActionSave⁽¹²⁴⁾ and TrvActionExport⁽¹¹²⁾ are removed. Use HTMLSaveProperties and DocParameters.Title properties of the target editor.

TRVAControlPanel.HTMLComponent⁽⁵³⁾ is not removed but deprecated. It's highly recommended to remove rvHtmlImporter⁽³⁷⁰⁾ and rvHtmlViewImporter⁽³⁷¹⁾ components from your application, because HTML loading methods of TRichView provide better results.

TRVAControlPanel.InitImportPictures, DonelImportPictures, DoImportPicture are removed. They are replaced by InitImportPicturesAndFiles and DonelImportPicturesAndFiles⁽⁶¹⁾.

HTML

Since this version RichViewActions use TRichView methods for loading HTML, and new methods for saving HTML.

TRVAControlPanel.HTMLComponent⁽⁵³⁾ property still can be used, but it is deprecated, because TRichView methods provide much better results than imported components. We plan to remove this property in future updates of RichViewActions.

Some properties for HTML export are removed from actions (see the compatibility issues above).

A component assigned to TRVAControlPanel.DownloadInterface⁽⁴⁸⁾ can be used to download not only external pictures, but also external CSS files.

Other changes

New properties controlling a state of the "Default" checkbox in property dialogs:

- TrvActionItemProperties.DefaultChecked, DefaultPersistent⁽³²⁹⁾
- TrvActionTableProperties.DefaultChecked, DefaultPersistent⁽³⁰⁵⁾

You can implement your own way for choosing pictures using the new TRVAControlPanel⁽³⁹⁾.OnChoosePicture⁽⁶³⁾ event.

Table properties dialog allows choosing cell height mode: "exactly" or "at least".

▼ Changes in version 10 (between TRichView 19 and 20)

RAD Studio 11 Alexandria

RichViewActions can be used in Delphi and C++Builder 11 Alexandria.

Help files

RVA_SetHelpFile⁽³⁴⁹⁾ allows defining the help file that will be used in RichViewActions dialogs for the specified UI language.

TRVAControlPanel⁽³⁹⁾.UseDefaultHelpFile⁽⁵⁷⁾ allows using Application.HelpFile.

TRVAControlPanel⁽³⁹⁾.HelpType⁽⁴⁹⁾ allows choosing between using HelpContext and HelpKeyword properties.

RVSetHelpKeyword⁽³⁵⁴⁾ allows changing HelpKeyword for the given HelpContext.

Markdown

Markdown is supported not only as export format, but also as import format (TrvActionInsertFile⁽²³⁰⁾) and document format (TrvActionOpen⁽¹⁰⁸⁾ and TrvActionSaveAs⁽¹²⁹⁾)

▼ Changes in version 9 (between TRichView 18 and 19)

RAD Studio 10.4 Sydney

RichViewActions can be used in Delphi and C++Builder 10.4 Sydney. Per-control VCL styling is supported; in dialogs, previews use the style of the target editor.

Localization

New UI translation: Slovenian

Files: DocX and Markdown

TrvActionOpen⁽¹⁰⁸⁾ and TrvActionInsertFile⁽²³⁰⁾ support DocX files.

TrvActionExport⁽¹¹²⁾ support **Markdown** files.

Find and replace

New properties:

- TrvActionFind⁽¹³⁴⁾.FindText⁽¹³⁶⁾
- TrvActionReplace⁽¹⁴³⁾.FindText, ReplaceText⁽¹⁴⁴⁾

The actions do not display "not found" dialog after the user answered "No" to "continue search from the beginning/end?" anymore.

Other

Smooth scrolling to checkpoint (in TrvActionBookmarks⁽²²³⁾ and TrvActionInsertHyperlink⁽²³⁴⁾.GoToLink⁽²⁴⁴⁾).

▼ Changes in version 8 (between TRichView 17 and 18)

Compatibility issues:

- RVA_ConvertToPixels and RVA_ConvertToTwips⁽³⁸⁰⁾ procedures have a new ARVStyle parameter.
- The following properties of TrvActionInsertSymbol⁽²⁵⁶⁾ are removed: AlwaysInsertUnicode, DisplayUnicodeBlocks, SymbolType
- String parameters are changed to TRVUnicodeString in the events:
 - TRVAControlPanel.OnCustomFileOperation⁽⁶⁴⁾, OnDownload⁽⁶⁶⁾, OnGetHeaderFooterCode⁽⁶⁷⁾.

- TRVAPopupMenu.OnLiveSpellAdd⁽⁷¹⁾, OnLiveSpellGetSuggestions⁽⁷²⁾, OnLiveSpellIgnoreAll⁽⁷²⁾, OnLiveSpellWordReplace⁽⁷³⁾
- TrvActionOpen.OnOpenFile⁽¹¹²⁾
- TrvActionSave.OnSaving⁽¹²⁸⁾, OnDocumentFileChange⁽¹²⁸⁾
- TrvActionReplace.OnReplacing⁽¹⁴⁶⁾
- TrvActionInsertHyperlink.OnApplyHyperlinkToItem⁽²⁴⁶⁾, OnGetHyperlinkTargetFromItem⁽²⁴⁶⁾, OnHyperlinkForm⁽²⁴⁶⁾
- TrvActionInsertPicture.OnInserting⁽²⁵⁶⁾
- TrvActionInsertText.OnInsertText⁽²⁵⁹⁾
- Support for KSDev ThemeEngine is discontinued
- RVAFormat function is moved from RichViewActions to RVAFuncs unit (you can simply use Format instead).

RAD Studio 10.3 Rio

RichViewActions can be used in Delphi and C++Builder 10.3 Rio.

Lazarus

RichViewActions can be used in Lazarus (for Windows 32-bit and 64-bit).

High-DPI display modes

All controls and dialogs in RichViewActions support high-dpi display modes.

"Per monitor" and "per monitor v2" modes are supported, if they are supported by the application (Delphi 10.1+ is required for "per monitor", Delphi 10.3+ for "per monitor v2").

Virtual image lists

New data modules are added for RAD Studio 10.3. They contain 16x16, 32x32, and selected 64x64 toolbar images in TImageCollection and TVirtualImageList components. Pascal and C++ versions of these data modules are available. Details are explained in the topic about TRichView icons⁽³⁷⁴⁾.

New demo projects are included:

- DelphiUnicode\ActionTest_MultiRes\
- CBuilderUnicode\ActionTest_MultiRes\

These projects use virtual image lists and support "per monitor v2".

Changes in "Insert Symbol" dialog

Since this version "Insert Symbol" allows inserting only Unicode characters. Now it supports all UTF-32 characters.

- TrvActionInsertSymbol⁽²⁵⁶⁾ has new properties: CharCode⁽²⁵⁷⁾, FontName⁽²⁵⁸⁾, Protection⁽²⁵⁸⁾.

Other changes

- new TrvActionEditNote⁽²⁶²⁾.OnStartEditNote⁽²⁶⁵⁾ event
- new TrvActionInsertHyperlink⁽²³⁴⁾.ScrollToCenter⁽²⁴⁰⁾ property
- new TRVFontComboBox⁽⁷⁵⁾.AutoCharset⁽⁷⁷⁾ property

- ruler is updated to support high DPI screen modes

▼ Changes in version 7 (between TRichView 16 and 17)

Compatibility issues:

- Ruler.pas is renamed to RVRulerBase.pas
- Color property is removed from Rules.Tabs[]
- new parameters in functions from MarkSearch⁽³⁸³⁾ unit
- No more compiler \$defines for using Addict⁽³⁶⁴⁾, you cannot add RVAddictSpell3 and RVAddictThesaurus3 properties to TRVAControlPanel⁽³⁹⁾. Use SpellInterface⁽⁵⁶⁾ property instead. TrvActionAddictSpell3, TrvActionAddictThesaurus3 are removed, they are superseded by TrvActionSpellingCheck⁽³¹²⁾ and TrvActionThesaurus⁽³¹²⁾. RVA_Addict3AutoCorrect function is removed, use TRVAAddictSpellInterface⁽³⁵⁸⁾'s auto-correct methods
- No more compiler \$defines for using TIdHttp (Indy⁽³⁷⁰⁾) and TCIdHttp (CleverComponents⁽³⁶⁹⁾), you cannot add IdHttp and CIdHttp properties to TRVAControlPanel⁽³⁹⁾. Use DownloadInterface⁽⁴⁸⁾ property instead.
- No more compiler \$defines for using RichViewXML⁽³⁷⁰⁾, RvHtmlImporter⁽³⁷⁰⁾ and RvHtmlViewImporter⁽³⁷¹⁾, you cannot add RVXML, RVHTMLImporter, RVHTMLViewImporter properties to TRVAControlPanel. Use XMLComponent⁽⁶⁰⁾ and HTMLComponent⁽⁵³⁾ properties instead.
- The compiler \$define for using TNT Controls⁽³⁷²⁾ is moved from RichViewActions.inc to RV_Defs.inc.
- TrvActionShowSpecialCharacters⁽³³³⁾ shows/hides checkpoints by default.
- The following properties are removed: TrvActionPasteSpecial⁽¹³⁹⁾.StoreFileName, TrvActionInsertPicture⁽²⁵¹⁾.StoreFileName, TrvActionTableProperties⁽³⁰³⁾.StoreImageFileName, TrvActionItemProperties⁽³²⁶⁾.StoreImageFileName. Instead of these properties, the actions check *rvoAssignImageFileNames* in the Options property of the target editor.

New "interface" components

Interface components⁽³⁵⁴⁾ provide an intermediate layer between RichViewActions and third-party components, so RichViewActions can use third-party components without compiler \$defines.


New properties are added to TRVAControlPanel⁽³⁹⁾: ColorDialogInterface⁽⁴³⁾, DownloadInterface⁽⁴⁸⁾, SpellInterface⁽⁵⁶⁾.

Interface components for using TdxColorDialog, Indy⁽³⁷⁰⁾, CleverComponents⁽³⁶⁹⁾, Addict⁽³⁶⁴⁾, ASpell⁽³⁶⁵⁾, ExpressSpellChecker⁽³⁶⁶⁾, HunSpell⁽³⁶⁷⁾, Polar SpellChecker⁽³⁶⁸⁾ are added.

File loading and saving components

Previously, programmers needed to add \$defines in RichViewActions.inc to allow using RichViewXML⁽³⁷⁰⁾, RvHtmlImporter⁽³⁷⁰⁾ and RvHtmlViewImporter⁽³⁷¹⁾ in RichViewActions. Now, you can use XMLComponent⁽⁶⁰⁾ and HTMLComponent⁽⁵³⁾ properties of TRVAControlPanel⁽³⁹⁾.


Checkpoints (Bookmarks)

New action  TrvActionBookmarks⁽²²³⁾ allows adding and managing checkpoints.

In TrvActionInsertHyperlink⁽²³⁴⁾, the editor of the link target is now a combo-box containing a list of checkpoint names.

TrvActionShowSpecialCharacters⁽³³³⁾ can show/hide checkpoints.

Mathematical formulas (equations)

New action:  TrvActionInsertEquation⁽²²⁸⁾. TrvActionItemProperties⁽³²⁶⁾ can edit properties of equation items.

Font preview

TRVFontComboBox⁽⁷⁵⁾ can display preview of fonts. It has new properties: Preview⁽⁷⁸⁾, DropDownWidth⁽⁷⁷⁾, SymbolPreviewString⁽⁷⁸⁾.

The dialog of TrvActionFontEx⁽¹⁵¹⁾ can display a preview of font names in the list as well. It has a new property: PreviewInList⁽¹⁵⁵⁾.

Changes related to tables

- TrvActionInsertTable⁽²⁷³⁾ has new properties to assign to table (HeadingRowColor and other colors⁽²⁸³⁾, ColBandSize, RowBandSize⁽²⁷⁹⁾) and to table rows (RowsVAlign⁽²⁸²⁾, RowsKeepTogether⁽²⁸¹⁾).
- The table properties dialog (displayed by TrvActionItemProperties⁽³²⁶⁾ and TrvActionTableProperties⁽³⁰³⁾) allows defining colors of rows and columns.
- new properties TrvActionItemProperties.UpdateAllInsertTableActions⁽³²⁸⁾ and TrvActionTableProperties.UpdateAllInsertTableActions⁽³⁰⁵⁾ allows applying default properties to all "insert table" actions on the same form/datamodule.
- The "Default" check box in the table properties affects not only the page "Table", but also pages "Rows" and "Columns".

Changes related to printing

- TrvActionInsertPageBreak⁽²⁵⁰⁾ and TrvActionRemovePageBreak⁽³³²⁾, when called from a table cell, add and remove page breaks before the current table row.
- PageNumberType⁽²²⁸⁾ property is added to TrvActionInsertPageCount⁽²⁵⁰⁾. For inserted "page count" fields, a number type can be changed by TrvActionItemProperties⁽³²⁶⁾.
- New TRVAControlPanel⁽³⁹⁾.OnGetHeaderFooterCode⁽⁶⁷⁾ event.

Other changes

- OnInserting⁽²⁵⁶⁾ event is added to TrvActionInsertPicture⁽²⁵¹⁾.
- new property TRuler.TickColor for drawing ticks and tab stops.
- New untranslated features can be hidden using ShowUntranslatedControls⁽³⁹²⁾ variable.
- MarkSearch⁽³⁸³⁾ functions are improved

- Packages were separated into runtime and designtime packages. RVARibbonUtils.pas (unit providing TRibbon support) is moved to a separate runtime package.

▼ Changes in version 6 (between TRichView 15 and 16)

Compatibility issues:

TrvActionItemProperties⁽³²⁶⁾ and TrvActionTableProperties⁽³⁰³⁾ update linked TrvActionInsertTable⁽²⁷³⁾ and TrvActionInsertHLine⁽²³³⁾ actions when the user checks "Default" checkbox. Previously, they were always updated when assigned explicitly.

Installation and directory structure


Starting from this update, RichViewActions are installed automatically in Delphi and C++Builder IDE together with TRichView.


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-files are moved to "Source\Include" folder, demo projects are moved to "Demos" folder.

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

New actions

 TrvActionAlignDistribute⁽¹⁷⁸⁾ aligns the selected paragraph to the both left and right sides by adding space between all characters.

 TrvActionInsertPageCount⁽²⁵⁰⁾ inserts a page count field.

Changes related to paragraph alignments

TrvActionAlignJustify⁽¹⁷⁹⁾ (as well as TrvActionAlignDistribute⁽¹⁷⁸⁾) has new properties: LastLineAlignment and UseLastLineAlignment⁽¹⁷⁸⁾.

TrvActionParagraph⁽²⁰⁰⁾ has a new property: LastLineAlignment⁽²⁰⁴⁾.

New properties:

- TRVAControlPanel⁽³⁹⁾.MetafileCompatibility⁽⁵³⁾ affects printing of TSRichViewEdit components;
- TrvActionSave⁽¹²⁴⁾.DisableWhenUnmodified⁽¹²⁶⁾ allows to disable the action for unmodified documents.

TrvActionRemovePageBreak⁽³³²⁾ removes a page break from the current paragraph (not the current item as before).

TrvActionItemProperties⁽³²⁶⁾ and TrvActionTableProperties⁽³⁰³⁾ update linked TrvActionInsertTable⁽²⁷³⁾ and TrvActionInsertHLine⁽²³³⁾ actions when the user checks "Default" checkbox.









▼ Changes in version 5 (between TRichView 14 and 15)

Compatibility issues:

- TrvActionFontEx⁽¹⁵¹⁾.Font⁽¹⁶³⁾.Size is no longer used. Existing code must be changed to use TrvActionFontEx⁽¹⁵¹⁾.FontSizeDouble⁽¹⁵⁵⁾ instead.

- 'Francais' cannot be used as a language name; use 'French' or 'Français'.

New actions

-  TrvActionInsertTextBox⁽²⁷¹⁾ inserts a floating box
-  TrvActionInsertFootnote⁽²⁶⁶⁾ inserts a footnote
-  TrvActionInsertEndnote⁽²⁶⁵⁾ inserts an endnote
-  TrvActionInsertSidenote⁽²⁶⁹⁾ inserts a note in a floating box
-  TrvActionEditNote⁽²⁶²⁾ displays a window for editing a note or a floating box
-  TrvActionInsertNumber⁽²⁴⁷⁾ inserts a "numbered sequence" item
-  TrvActionInsertCaption⁽²²⁵⁾ inserts a caption for an image or a table
-  TrvActionInsertPageNumber⁽²⁵¹⁾ inserts a "page number" field

Changes

TrvActionExport⁽¹¹²⁾ can export DocX files (without office converters). Make sure that *ffeDocX* is included in Filter⁽¹¹⁵⁾.

Changes made by TrvActionColor⁽³¹⁷⁾, TrvActionBackground⁽³¹⁵⁾ can be undone.

New property: TrvActionFontEx⁽¹⁵¹⁾.FontSizeDouble⁽¹⁵⁵⁾.

TRVFontSizeComboBox⁽⁸¹⁾ supports fractional font sizes.

Changes in TrvActionItemProperties⁽³²⁶⁾:

- properties for sidenotes and text boxes
- properties for "numbered sequence" items
- properties for "page number" item
- redesigned picture properties pages
- redesigned table properties pages
- changes can be applied to ActionInsertTable⁽³²⁸⁾ and ActionInsertHLine⁽³²⁸⁾ (in addition to the current item)

Changes in TrvActionNew⁽¹⁰⁵⁾:

- Reset⁽¹⁰⁷⁾ is made public.
- new item is included in StyleTemplates⁽¹⁰⁶⁾: 'caption'.

New TrvActionInsertPicture⁽²⁵¹⁾ properties to assign to inserted pictures: BackgroundColor⁽²⁵³⁾, BorderWidth, BorderColor⁽²⁵³⁾, OuterHSpacing, OuterVSpacing⁽²⁵⁵⁾.

New TrvActionInsertTable⁽²⁷³⁾ properties to assign to inserted tables: CellHPadding and CellVPadding⁽²⁷⁸⁾ (they replace CellPadding), VisibleBorders⁽²⁸²⁾, BackgroundStyle, BackgroundPicture⁽²⁷⁵⁾.

TRVAControlPanel⁽³⁹⁾.DialogFontName⁽⁴⁵⁾'s default value is changed to 'Tahoma'.

New RemoveEllipsisFromRibbon⁽³⁸¹⁾ procedure replaces RemoveEllipsis.

Languages have two names specified: a name in English and a native name.

▼ Changes added between TRichView v13 and v14

Compatibility issues:






TRVAControlPanel⁽³⁹⁾.TableGridStyle property is removed.

Redesigned control panel

TRVAControlPanel⁽³⁹⁾ is redesigned. It does not assign global variables any more. You can have more than one control panel in a single application. New properties of TRVAControlPanel: Header, Footer⁽⁴⁹⁾.

New actions

Styles:⁽²⁰⁾

-  TrvActionStyleTemplates⁽²¹⁴⁾
-  TrvActionAddStyleTemplate⁽²¹⁷⁾
-  TrvActionClearFormat⁽²²¹⁾
-  TrvActionClearTextFormat⁽²²¹⁾
-  TrvActionStyleInspector⁽²¹⁸⁾

Cell rotation:










-  TrvActionTableCellRotationNone⁽²⁹⁰⁾
-  TrvActionTableCellRotation90⁽²⁹¹⁾
-  TrvActionTableCellRotation180⁽²⁹¹⁾
-  TrvActionTableCellRotation270⁽²⁹²⁾

Table operations:




-  TrvActionTableSplit⁽³¹⁰⁾
-  TrvActionTableSort⁽³⁰⁹⁾
-  TrvActionTableToText⁽³¹¹⁾

New and improved components

New components for applying style templates⁽²⁰⁾:

-  TRVStyleTemplateComboBox⁽⁸⁹⁾
-  TRVStyleTemplateListBox⁽⁸⁹⁾

Improved components (can be linked to an editor, do not require additional code any more):

-  TRVFontComboBox⁽⁷⁵⁾
-  TRVFontSizeComboBox⁽⁸¹⁾
-  TRVFontCharsetComboBox⁽⁷³⁾

ScaleRichView support

You can assign not only TRichViewEdit, but also TSRichViewEdit to TRVAControlPanel.DefaultControl⁽⁴³⁾, or Control⁽³¹⁴⁾ properties of actions.

Simplification: since this version, when using ScaleRichView, you do not need to assign RVA_GetRichViewEditFromPopupComponent and RVA_GetRichViewEdit variables, you do not need to call SRichViewEdit.SetRVMargins in TRVAControlPanel.OnMarginsChanged⁽⁶⁷⁾.

Visual changes

Delphi XE2+'s visual styles are supported in components and dialogs. The function RVA_ChooseStyle⁽³⁸⁰⁾ allows choosing and applying one of available styles.

Inverted (a light text on a dark background) color schemes are supported.

Text files

The file actions allow choosing a code page for text files (including UTF-8). This feature can be disabled by assigning `TRVAControlPanel.UseTextCodePageDialog`⁽⁵⁹⁾ = `False`.

New features, properties and events

A new event `TRVAControlPanel.OnGetActionControlCoords`⁽⁶⁶⁾ allows to specify coordinates for color-picker windows when actions are linked to non-visual components (for example, when using `ExpressBars` by Developer Express).

`TrvActionNew`⁽¹⁰⁵⁾ resets styles according to `StyleTemplates`⁽¹⁰⁶⁾, if style templates are used. `TrvActionOpen`⁽¹⁰⁸⁾ can use a linked `TrvActionNew`⁽¹¹⁰⁾ to reset document before loading.

`TrvActionPasteSpecial`⁽¹³⁹⁾:

- allows pasting URLs;
- respects `RichViewEdit.AcceptPasteFormats` property;
- if style templates⁽²⁰⁾ are used, displays style modes for pasting RTF and RVF.

`TrvActionFontEx`⁽¹⁵¹⁾ can change a text background color (new `BackColor`⁽¹⁵⁴⁾ property).

`TrvActionInsertHyperlink`⁽²³⁴⁾ can apply `HoverUnderlineColor` and `HoverEffects` properties (if style templates are not used). If style templates are used, it applies the chosen `StyleTemplateName`⁽²⁴¹⁾.

`TrvActionTableGrid`⁽²⁹⁸⁾ shows/hides grid only in the target editor.

In the print preview form (`TrvActionPrintPreview`⁽¹²⁰⁾), a mouse wheel changes pages, **Ctrl** + mouse wheel zooms in/out

New optional parameters in `NormalizeRichView`⁽³⁸⁴⁾.

Localization

New localization functions: `RVA_GetProgressMessage`, `RVA_GetPrintingMessage`⁽³⁴⁵⁾.

New languages:

- Portuguese (European)
- Catalan
- Hindi (in Delphi 4-2007, it is available only if TNT Controls⁽³⁷²⁾ are used);
- Thai

▼ Changes added between TRichView v12 and v13

New units of measurement

All dialogs display values measured according to new `TRVAControlPanel`⁽³⁹⁾.`UnitsDisplay`⁽⁵⁶⁾ and `TRVAControlPanel.PixelBorders`⁽⁵³⁾ properties.

Properties of several actions are measured in `TRVAControlPanel`⁽³⁹⁾.`UnitsProgram`⁽⁵⁷⁾.

New images

A new set of toolbar images⁽³⁷⁴⁾ is available, created specially for RichViewActions. This set of images is used in this manual.

New component images are used in the Delphi/C++Builder's Component Palette for the components included in RichViewActions.

New properties of TRVAControlPanel

TRVAControlPanel.DefaultDocParameters⁽⁴⁴⁾ define a page layout for new documents (especially useful for ScaleRichView)


New TRVAControlPanel⁽³⁹⁾.DialogFontSize⁽⁴⁶⁾ property.

Changes in menus

New TRVAPopupActionBar⁽⁶⁸⁾ component (for Delphi 2006 or newer).

The type of ActionList⁽⁷¹⁾ property for menus⁽⁶⁸⁾ was changed from TActionList to TCustomActionList, to allow using TActionManager.

Changes in actions

New action:  TrvActionHide⁽³²⁶⁾. TrvActionShowSpecialCharacters⁽³³³⁾ shows/hides a hidden text as well as special characters.

New properties:

- TrvActionFontEx⁽¹⁵¹⁾.AutoApplySymbolCharset⁽¹⁵³⁾
- TrvActionInsertHLine⁽²³³⁾.Style⁽²³⁴⁾ defines the style for inserted horizontal lines
- TrvActionBackground⁽³¹⁵⁾.CanChangeMargins⁽³¹⁶⁾ allows/disallows changing margins
- TrvActionParaBullets⁽¹⁹⁷⁾'s and TrvActionParaNumbering⁽²¹¹⁾'s IndentStep⁽¹⁸⁶⁾ defines default indents for list levels.

TrvActionParaList⁽²⁰⁸⁾.IndentStep⁽²⁰⁹⁾ now affects indents of list style templates in non-HTML dialog too.








Other changes

New RVARibbonUtils⁽³⁸¹⁾ unit.

Ability to use CleverComponents⁽³⁶⁹⁾ for downloading images.

▼ Changes added between TRichView v11 and v12

New actions:

-  TrvActionPasteAsText⁽¹³⁸⁾
-  TrvActionVAlign⁽³³⁵⁾
-  TrvActionClearLeft⁽¹⁸²⁾
-  TrvActionClearRight⁽¹⁸³⁾
-  TrvActionClearBoth⁽¹⁸²⁾
-  TrvActionClearNone⁽¹⁸³⁾
-  TrvActionRemoveHyperlinks⁽³³¹⁾

New methods, properties and events

- TrvActionInsertFile.InsertFile⁽²³²⁾ method
- TrvActionInsertPicture.Spacing⁽²⁵⁵⁾ property
- TrvActionParagraph.OutlineLevel⁽²⁰⁵⁾ property (paragraph dialog has a new combobox for specifying a paragraph outline level)

- TrvActionItemProperties.OnCanApply⁽³³⁰⁾ event

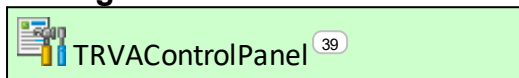
New features:

- New types of vertical alignment for pictures in TrvActionItemProperties⁽³²⁶⁾.
- Properties dialog for breaks (TrvActionItemProperties⁽³²⁶⁾) has a new combobox for specifying a break style (line/rectangle/3d/dotted/dashed).
- Pasting HTML from the Clipboard using rvHtmlViewImporter⁽³⁷¹⁾.
- New datamodules with alternative toolbar images (GlyFX⁽³⁷⁶⁾ and Fugue Icons⁽³⁷⁷⁾)

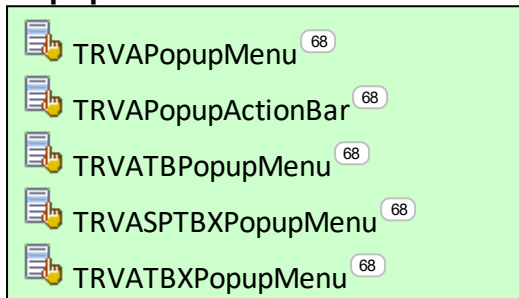
1.2 Components

Components

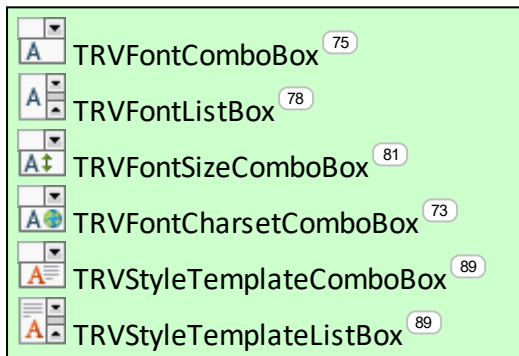
Settings



Popup Menus



Combo- and List-Boxes



1.2.1 TCustomRVFontComboBox

TCustomRVFontComboBox is the base class for font combo-boxes.

Unit RVFontCombos;

Syntax, if TNT Controls⁽³⁷²⁾ are not used:

```
TCustomRVFontComboBox = class (TComboBox)
```

Syntax, if TNT Controls⁽³⁷²⁾ are used:

```
TCustomRVFontComboBox = class (TTntComboBox)
```

Hierarchy

(Hierarchy of TCustomRVFontComboBox, if TNT Controls⁽³⁷²⁾ are not used)*TObject**TPersistent**TComponent**TControl**TWinControl**TCustomComboBox**TComboBox*

Description

This component is not used directly.

The following components are inherited from TCustomRVFontComboBox:

- TRVFontComboBox⁽⁷⁵⁾
- TRVFontSizeComboBox⁽⁸¹⁾
- TRVFontCharsetComboBox⁽⁷³⁾

1.2.1.1 Properties

In TCustomRVFontComboBox

- ActionFont⁽³⁵⁾
- Editor⁽³⁵⁾

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu

- ShowHint
- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.1.1.1 TCustomRVFontComboBox.ActionFont

Links this component to TrvActionFontEx⁽¹⁵¹⁾ action.

property ActionFont: TrvActionFontEx⁽¹⁵¹⁾;

This action is required to apply the selection to Editor⁽³⁵⁾.

1.2.1.1.2 TCustomRVFontComboBox.Editor

Links this combo-box with an editor.

property Editor: TCustomRVControl;

Controls of the following types can be assigned to this property:

- TRichViewEdit
- TDBRichViewEdit
- TSRichViewEdit (ScaleRichView)
- TDBSRichViewEdit (ScaleRichView)

When the combo-box is linked with **Editor**:

- the combo-box is updated automatically according to changes in **Editor**
- the combo-box applies the user choice to the selected fragment in **Editor** (ActionFont⁽³⁵⁾ must be defined as well)

1.2.2 TCustomRVFontListBox

TCustomRVFontListBox is the base class for font list-boxes.

Unit RVFontCombos;

Syntax, if TNT Controls⁽³⁷²⁾ are not used:

TCustomRVFontListBox = **class** (TListBox)

Syntax, if TNT Controls⁽³⁷²⁾ are used:

TCustomRVFontListBox = **class** (TTntListBox)

Hierarchy

(Hierarchy of TCustomRVFontListBox, if TNT Controls⁽³⁷²⁾ are not used)

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomListBox

TListBox

Description

This component is not used directly.

The following components are inherited from TCustomRVFontListBox:

- TRVFontListBox⁷⁸

1.2.2.1 Properties

In TCustomRVFontComboBox

- ActionFont³⁷
- AllowFocusEditor³⁷
- Editor³⁷

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.2.1.1 TCustomRVFontListBox.ActionFont

Links this component to TrvActionFontEx⁽¹⁵¹⁾ action.

property ActionFont: TrvActionFontEx⁽¹⁵¹⁾;

This action is required to apply the selection to Editor⁽³⁷⁾.

1.2.2.1.2 TCustomRVFontListBox.AllowFocusEditor

Allows/disallows moving the input focus to Editor⁽³⁷⁾ when this list-box is clicked.

property AllowFocusEditor: Boolean;

Default value

False

1.2.2.1.3 TCustomRVFontListBox.Editor

Links this list-box with an editor.

property Editor: TCustomRVControl;

Controls of the following types can be assigned to this property:

- TRichViewEdit
- TDBRichViewEdit
- TSRichViewEdit (ScaleRichView)
- TDBSRichViewEdit (ScaleRichView)

When the list-box is linked with **Editor**:

- the list-box is updated automatically according to changes in **Editor**
- the list-box applies the user choice to the selected fragment in **Editor** (ActionFont⁽³⁷⁾ must be defined as well)

1.2.3 TRulerItemSelector

TRVRulerItemSelector defines the current tabulation type in TRVRuler⁽⁸²⁾ component.

Unit RVRuler;

TRVRulerItemSelector = **class** (TRulerItemSelector)

Hierarchy

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomControl
TCustomPanel
TCustomRulerItemSelector
TRulerItemSelector

Description

This component allows switching a type of current tab stops in Ruler³⁹: left-, right-, center-, decimal-aligned tab stops.

Tab stops of this type are added when the user clicks on the ruler.

Acknowledgement

Thanks to Pieter Zijlstra who created this component.

1.2.3.1 Properties

Derived from TCustomRulerSelector

- AvailableItems
- BevelHighLightColor
- BevelShadowColor
- Ruler³⁹

Derived from TCustomPanel

- Align
- Anchors
- BevelEdges
- BevelInner
- BevelKind
- BevelOuter
- BevelWidth
- BorderWidth
- BorderStyle
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- Enabled
- FullRepaint
- Locked
- Padding
- ParentBackground
- ParentColor
- ParentCtl3D
- ParentShowHint
- PopupMenu
- ShowHint
- TabOrder
- TabStop
- Visible

1.2.3.1.1 TCustomRulerItemSelector.Ruler

Specifies the ruler.

property Ruler: TCustomRuler⁽⁸²⁾;

This component changes the default type of tab stop for the ruler specified in this property.

1.2.4 TRVAControlPanel

TRVAControlPanel is the component allowing to change settings for RichViewActions

Unit RichViewActions;

Syntax

TRVAControlPanel = **class** (TComponent)

Hierarchy

TObject

TPersistent

TComponent

Description

This component contains properties affecting many RichViewActions.

Main control panel

If the application does not create TRVAControlPanel explicitly, a default control panel is used (accessible as MainRVAControlPanel⁽³⁹⁰⁾). If the application contains a single TRVControlPanel, it becomes the main control panel automatically (assigned to MainRVAControlPanel). Another control panel can be made the main one if you call its Activate⁽⁶¹⁾ method.

Linking

All actions and many components in RichViewActions have links to a control panel, including:

- TrvCustomAction⁽³³⁵⁾.ControlPanel⁽³³⁷⁾
- TRVStyleTemplateCombo/ListBox⁽⁸⁹⁾.ControlPanel⁽⁹¹⁾

If this link is not assigned (*nil*), the action/component uses the main control panel (see above).

Properties

The component has the following group of properties:

- links to components providing additional functionality:
 - ColorDialog⁽⁴²⁾ and ColorDialogInterface⁽⁴³⁾
 - DownloadInterface⁽⁴⁸⁾
 - SpellInterface⁽⁵⁶⁾
 - RVPrint⁽⁵⁵⁾
 - XMLComponent⁽⁶⁰⁾
- properties for customizing user interface:
 - Language⁽⁵³⁾
 - AddColorNameToHints⁽⁴²⁾

- `UIInterface` ⁵⁸
- `UseXPThemes` ⁵⁹
- properties for dialog windows:
 - `DialogFontName` ⁴⁵ and `DialogFontNameLin` ⁴⁵
 - `DialogFontSize` ⁴⁶
 - `DialogPosition` ⁴⁷
 - `DialogZoomPercent` ⁴⁸
 - `DialogIcons` ⁴⁶
- properties related to end-user help files
 - `UseHelpFiles` ⁵⁸
 - `UseDefaultHelpFile` ⁵⁷
 - `HelpType` ⁴⁹
- properties of new documents:
 - `DefaultColor` ⁴³
 - `DefaultCustomFilterIndex` ⁴³
 - `DefaultExt` ⁴⁴
 - `DefaultFileFormat` ⁴⁴
 - `DefaultFileName` ⁴⁵
 - `DefaultMargin` ⁴⁵
 - `DefaultDocParameters` ⁴⁴
- properties controlling all actions:
 - `DefaultControl` ⁴³
 - `ActionsEnabled` ⁴²
- properties for changing names of file formats:
 - `RVFFilter` ⁵⁴
 - `RVFLocalizable` ⁵⁴
 - `RVFormatTitle` ⁵⁴
 - `RVStylesFilter` ⁵⁵
 - `RVStylesExt` ⁵⁵
 - `XMLFilter` ⁶⁰
 - `XMLLocalizable` ⁶⁰
- other properties:
 - `AutoDeleteUnusedStyles` ⁴²
 - `FirstPageNumber` ⁴⁸
 - `SearchScope` ⁵⁵
 - `ShowSoftPageBreaks` ⁵⁶
 - `UseTextCodePageDialog` ⁵⁹
 - `MetafileCompatibility` ⁵³

Events

If you use `TRVRuler` ⁸², process `OnMarginsChanged` ⁶⁷.

If you want to implement opening, saving, exporting and inserting files in additional formats, use `OnCustomFileOperation` ⁶⁴.

If you want to implement your own dialog for opening pictures, use `OnChoosePicture` ⁶³ event.

1.2.4.1 Properties

In TRVAControlPanel

- ActionsEnabled⁴²
- AddColorNameToHints⁴²
- AutoDeleteUnusedStyles⁴²
- ColorDialog⁴²
- ColorDialogInterface⁴³
- DefaultColor⁴³
- DefaultControl⁴³
- DefaultCustomFilterIndex⁴³
- DefaultDocParameters⁴⁴
- DefaultExt⁴⁴
- DefaultFileFormat⁴⁴
- DefaultFileName⁴⁵
- DefaultMargin⁴⁵
- DialogFontName⁴⁵
- DialogFontNameLin⁴⁵
- DialogFontSize⁴⁶
- DialogIcons⁴⁶
- DialogPosition⁴⁷
- DialogZoomPercent⁴⁸
- DownloadInterface⁴⁸
- FirstPageNumber⁴⁸
- HelpType⁴⁹ (D6+)
- Language⁵³
- MetafileCompatibility⁵³
- PixelBorders⁵³
- RVFFilter⁵⁴
- RVFLocalizable⁵⁴
- RVFormatTitle⁵⁴
- HTMLComponent⁵³ (deprecated)
- RVPrint⁵⁵
- RVStylesExt⁵⁵
- RVStylesFilter⁵⁵
- SearchScope⁵⁵
- ShowSoftPageBreaks⁵⁶
- SpellInterface⁵⁶
- UnitsDisplay⁵⁶
- UnitsProgram⁵⁷
- UseDefaultHelpFile⁵⁷
- UseHelpFiles⁵⁸
- UserInterface⁵⁸
- UseTextCodePageDialog⁵⁹
- UseXPThemes⁵⁹
- XMLComponent⁶⁰
- XMLFilter⁶⁰

■ XMLLocalizable⁽⁶⁰⁾

1.2.4.1.1 TRVAControlPanel.ActionsEnabled

Allows to disable all actions.

property ActionsEnabled: Boolean;

Set to *False* to disable all actions inherited from TrvAction⁽³¹³⁾, linked to this control panel.

Default value:

True

1.2.4.1.2 TRVAControlPanel.AddColorNameToHints

Allowing adding color names to hints of actions inherited from TrvActionCustomColor⁽³²⁰⁾.

property AddColorNameToHints: Boolean;

A color name can be added to the action's hint, if its UserInterface⁽³²²⁾=*rvacNone* (i.e. the action applies a specific color without displaying a dialog).

Default value:

True

1.2.4.1.3 TRVAControlPanel.AutoDeleteUnusedStyles

Instructs to delete unused styles from documents when it's possible.

property AutoDeleteUnusedStyles: Boolean;

If *True*, all unused text, paragraph and list styles are deleted after executing TrvActionNew⁽¹⁰³⁾ and TrvActionOpen⁽¹⁰⁸⁾.

It's highly recommended to leave it equal to *True*, otherwise a number of unused styles will be constantly increased (as well as sizes of documents).

If style templates are used⁽²⁰⁾, this property is ignored: instead of deleting unused styles, "New" and "Open" actions resets styles according to StyleTemplates⁽¹⁰⁶⁾.

Default value:

True

1.2.4.1.4 TRVAControlPanel.ColorDialog

Specifies a color dialog for using in RichViewActions.

property ColorDialog: TColorDialog;

This is a link to TColorDialog component. If it is not defined, actions create and use temporal color dialogs.

If the link is defined, the user can define a set of custom colors and use them in all actions. Otherwise, this link is optional.

This property can be overridden by ColorDialogInterface⁽⁴³⁾ property.

1.2.4.1.5 TRVAControlPanel.ColorDialogInterface

Specifies a custom color dialog for using in RichViewActions.

property ColorDialogInterface: TRVACustomColorDialogInterface⁽³⁶²⁾;

This property allows using a custom (third-party) color dialog instead of the standard TColorDialog.

ColorDialogInterface has a higher priority than ColorDialog⁽⁴²⁾: if both **ColorDialogInterface** and ColorDialog⁽⁴²⁾ are specified, **ColorDialogInterface** is used, ColorDialog⁽⁴²⁾ is ignored.

1.2.4.1.6 TRVAControlPanel.DefaultColor

Specifies the default background color.

property DefaultColor: TColor;

This property is assigned to TCustomRichViewEdit.Color property when executing TrvActionNew⁽¹⁰³⁾ or TrvActionOpen⁽¹⁰⁸⁾ (before loading).

See also:

- DefaultMargin⁽⁴⁵⁾
- DefaultDocParameters⁽⁴⁴⁾
- TrvActionNew⁽¹⁰⁵⁾.StyleTemplates⁽¹⁰⁶⁾

1.2.4.1.7 TRVAControlPanel.DefaultControl

Specifies the editor component for using in RichViewActions.

property DefaultControl: TCustomRVControl;

If this link is defined, all actions inherited from TrvAction⁽³¹³⁾ (and linked with this control panel) work with this editor. This property can be overridden by Control⁽³¹⁴⁾ property of the specific action.

Controls of the following types can be assigned to this property:

- TRichViewEdit
- TDBRichViewEdit
- TSRichViewEdit (ScaleRichView)
- TDBSRichViewEdit (ScaleRichView)

If this link (and action.Control⁽³¹⁴⁾ property) is not assigned, the action works with the editor component having the input focus, or, if an editor component is not focused, with the first found editor component.

1.2.4.1.8 TRVAControlPanel.DefaultCustomFilterIndex

Specifies the default index of a custom format for new documents.

property DefaultCustomFilterIndex: Integer;

This property is used if DefaultFileFormat⁽⁴⁴⁾=*ffeCustom*. This is an index of custom format (from 1); custom formats are listed in TrvActionSaveAs.CustomFilter⁽¹⁰⁰⁾ property.

This is a temporal format assigned to new documents after executing TrvActionNew⁽¹⁰³⁾. This format will be offered as a default format on the first execution of TrvActionSave⁽¹²⁴⁾ or TrvActionSaveAs⁽¹²⁹⁾ for new documents.

OnCustomFileOperation⁽⁶⁴⁾ occurs when files of this format need to be saved.

Default value:

1

1.2.4.1.9 TRVAControlPanel.DefaultDocParameters

Specifies default page parameters for new documents (page size, margins, etc.)


property DefaultDocParameters: TRVDocParameters;

If *rvfoSaveDocProperties* is included in RVFOptions properties of TRichViewEdit control, this property is assigned to its DocParameters property in the following cases:

- when TrvActionNew⁽¹⁰³⁾ is executed;
- before TrvActionOpen⁽¹⁰⁸⁾ loads a file.

ZoomPercent and ZoomMode sub-properties are not assigned.

Units sub-property is not assigned too. All sizes are converted to RichViewEdit.DocParameters.Units.

 **ScaleRichView note:** this property is especially important if RichViewActions are used with TSRichViewEdit control (ScaleRichView). DefaultDocParameters define a page layout for new documents, and DefaultMargin⁽⁴⁵⁾ property is ignored.

See also:

- DefaultColor⁽⁴³⁾
- TrvActionNew⁽¹⁰⁵⁾.StyleTemplates⁽¹⁰⁶⁾

1.2.4.1.10 TRVAControlPanel.DefaultExt

Specifies a default file extension for dialogs in TrvActionOpen⁽¹⁰⁸⁾, TrvActionSaveAs⁽¹²⁹⁾, and TrvActionExport⁽¹¹²⁾ actions.

property DefaultExt: TRVALocString⁽³⁵⁰⁾;

This value is assigned to DefaultExt property of TOpenDialog and TSaveDialog components used in these actions.

Default value:

'rvf'

1.2.4.1.11 TRVAControlPanel.DefaultFileFormat

Specifies the default format for new documents.

property DefaultFileFormat: TrvFileSaveFilter⁽³⁸⁸⁾;

If it equals to *ffeCustom*, DefaultCustomFilterIndex⁽⁴³⁾ property defines a subformat.

This is a temporal format assigned to new documents after executing TrvActionNew⁽¹⁰³⁾. This format will be offered as a default format on the first execution of TrvActionSave⁽¹²⁴⁾ or TrvActionSaveAs⁽¹²⁹⁾ for new documents.

Default value:*ffeRVF*

See also:

- DefaultExt⁽⁴⁴⁾

1.2.4.1.12 TRVAControlPanel.DefaultFileName

Specifies a default file name for new documents.

property DefaultFileName: TRVLocString⁽³⁵⁰⁾;

This is a temporal file name assigned to new documents after executing TrvActionNew⁽¹⁰³⁾. This file name will be offered as a default file name on the first execution of TrvActionSave⁽¹²⁴⁾ or TrvActionSaveAs⁽¹²⁹⁾ for new documents.

Default value:

'Untitled.rvf'

See also:

- DefaultExt⁽⁴⁴⁾


1.2.4.1.13 TRVAControlPanel.DefaultMargin

Specifies default margins for new documents.

property DefaultMargin: TRVPixelLength;

This property is assigned to LeftMargin, TopMargin, RightMargin, and BottomMargin properties of TCustomRichViewEdit component in the following cases:

- when TrvActionNew⁽¹⁰³⁾ is executed;
- before TrvActionOpen⁽¹⁰⁸⁾ loads a file.

 **ScaleRichView note:** for TSRichViewEdit, this property is ignored, see DefaultDocParameters⁽⁴⁴⁾

.

Default value:

5

See also:

- DefaultColor⁽⁴³⁾
- DefaultDocParameters⁽⁴⁴⁾
- TrvActionNew⁽¹⁰⁵⁾.StyleTemplates⁽¹⁰⁶⁾

1.2.4.1.14 TRVAControlPanel.DialogFontName

The properties define the default font name for all dialog windows displayed by RichViewActions.

property DialogFontName: String;

property DialogFontNameLin: String;

DialogFontName is used on Windows. **DialogFontNameLin** is used on Linux.

Each property contains a semicolon-delimited list of font names, in order of preference. The first font existed in the system is used.

'default' is a special font name, valid only for Lazarus. It means that the actual font name is determined by the application.

The actual font name used by RichViewActions is returned by `GetRealDialogFontName`⁶¹ method.

Default value:

- `DialogFontName`: 'Segoe UI;Tahoma;Microsoft Sans Serif;MS Sans Serif;Arial'
- `DialogFontNameLin`: 'default'

See also:

- `DialogFontSize`⁴⁶

1.2.4.1.15 TRVAControlPanel.DialogFontSize

Defines the default font size for all dialog windows displayed by RichViewActions.

property `DialogFontSize`: Integer;

If want to use a large font size, you should increase sizes of dialogs, see `DialogZoomPercent`⁴⁸ property.

Default value:

8

See also:

- `DialogFontName`⁴⁵

1.2.4.1.16 TRVAControlPanel.DialogIcons

Defines the image set used in dialogs.

type

```
TRVAIconSet = (rvaicnClassic, rvaicnGoogle);
```

property `DialogIcons`: TRVAIconSet;

Value	Meaning
<i>rvaicnClassic</i>	Standard images of RichViewActions
<i>rvaicnGoogle</i>	Images based on Google Material Symbols and Icons (for RAD Studio 10.3 and newer)

Standard images ---

Availability:

All supported versions of Delphi, C++Builder, and Lazarus.

License

No special license; integral part of RichViewActions.

Support of High-DPI display modes

Yes (partial). The original images were created for 96 dpi (zoom 100%). If the screen's DPI is higher, the images are scaled accordingly, but only by an integer number (2x, 3x, etc.). The quality of the enlarged images is lower than the original.

Support for color themes

Partial. For dark color themes, image colors are inverted.

Images based on Google Material Symbols and Icons

Availability

Delphi and C++Builder 10.3 and newer.

License:

▼ Apache License, Version 2.0

Google Material Symbols and Icons are licensed under the Apache License, Version 2.0 (the "License");

You may not use them except in compliance with the License.

You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and limitations under the License.

The following changes there made:

- New images were created based on Google Material Symbols and Icons, using elements present in the original set of images.
- Small images were edited to align objects pixel-perfectly without blurring.

Support of High-DPI display modes

Yes. The set includes versions of images in various sizes, from which the most suitable ones are selected and scaled if necessary.

Support for color themes

Yes. The color of images changes to the color of text (of Windows theme or VCL theme)

Default value:

rvaicnClassic

1.2.4.1.17 TRVAControlPanel.DialogPosition

Defines the position of dialog windows displayed by RichViewActions.

type

```
TRVAFormPosition =  
    (rvafpMainFormCenter, rvafpScreenCenter,  
     rvafpEditorFormCenter);
```

property DialogPosition: TRVAFormPosition;

Value	Meaning
<i>rvafpMainFormCenter</i>	Dialog windows are positioned in the center of the application's main form
<i>rvafpScreenCenter</i>	Dialog windows are positioned in the center of the screen
<i>rvafpEditorFormCenter</i>	Dialog windows are positioned in the center of the application's target editor's form

Default value:

rvafpMainFormCenter

1.2.4.1.18 TRVAControlPanel.DialogZoomPercent

Allows scaling all dialog windows displayed by RichViewActions.

property `DialogZoomPercent: Integer;`

The value of this property defines sizes of RichViewActions' dialogs, %.

If you assign a large value to DialogFontSize⁽⁴⁶⁾, consider increasing sizes of dialogs using this property.

Default value:

100

1.2.4.1.19 TRVAControlPanel.DownloadInterface

Allows using a third-party components to download images and external files (such as CSS files) from HTTP and HTTPS links.

property `DownloadInterface: TRVACustomDownloadInterface(356);`

The actions may need to download images when loading/inserting RTF, DocX or HTML files, if these files contain external images or CSS files.

You can assign one of the following components to this property:

- TRVAIndyDownloadInterface⁽³⁵⁶⁾ (to use Indy⁽³⁷⁰⁾)
- TRVACIDownloadInterface⁽³⁵⁵⁾ (to use CleverComponents⁽³⁶⁹⁾)

See also

- OnDownload⁽⁶⁶⁾ event
- InitImportPicturesAndFiles, DoneImportPicturesAndFiles⁽⁶¹⁾ methods

1.2.4.1.20 TRVAControlPanel.FirstPageNumber

Defines the page number for the first page.

property `FirstPageNumber: Integer;`

If page numbers are printed, this value specifies the number printed on the first page.

Default value:

1

See also properties:

- Header, Footer⁽⁴⁹⁾

1.2.4.1.21 TRVAControlPanel.Header, Footer

The properties define the page headers and footers.


```
property Header: TRVAHFInfo(386);
property Footer: TRVAHFInfo(386);
```

You can change properties of a header or a footer by assigning subproperties.

TrvActionPageSetup⁽¹¹⁶⁾ changes these properties as well.

The following actions print headers and footers:

- TrvActionPrint⁽¹¹⁸⁾
- TrvActionQuickPrint⁽¹²²⁾
- TrvActionPrintPreview⁽¹²⁰⁾

 **ScaleRichView note:** these properties are ignored when printing TSRichViewEdit or TDBSRichViewEdit controls.

Default value:

Header: defaults for TRVAHFInfo, but Text := '- &p -';

Footer: defaults for TRVAHFInfo

See also properties:

- FirstPageNumber⁽⁴⁸⁾

1.2.4.1.22 TRVAControlPanel.HelpType

Specifies whether the RichViewActions forms' context-sensitive Help topics are identified by a context ID or by a keyword.

```
property HelpType: THelpType;
```

This property affects all RichViewActions forms.

The list of values of HelpContext and HelpKeyword properties are shown below.

Values of HelpContext are hard-coded. Values of HelpKeyword can be modified by RVSetHelpKeyword⁽³⁵⁴⁾.

Core RichViewActions

HelpContext	HelpKeyword	Dialog is displayed by
90000	Document background dialog	TrvActionBackground ⁽³¹⁵⁾
90100	Fill color dialog	TrvActionFillColor ⁽³²⁴⁾
90200	Font dialog	TrvActionFontEx ⁽¹⁵¹⁾
90300	Paragraph padding dialog	TrvActionParaBorder ⁽¹⁹³⁾ TrvActionFillColor ⁽³²⁴⁾

90301	Paragraph text margins dialog	TrvActionParaBorder ⁽¹⁹³⁾
90302	Document padding dialog	TrvActionBackground ⁽³¹⁵⁾
90400	Hyperlink dialog	TrvActionInsertHyperlink ⁽²³⁴⁾
90450	Hyperlink attributes dialog	TrvActionInsertHyperlink ⁽²³⁴⁾ (if StyleTemplates are not used)
90460	Active hyperlink attributes dialog	TrvActionStyleTemplates ⁽²¹⁴⁾ TrvActionAddStyleTemplate ⁽²¹⁷⁾ (when editing properties of hypertext style)
90500	Symbol insertion dialog	TrvActionInsertSymbol ⁽²⁵⁶⁾
90600	Table insertion dialog	TrvActionInsertTable ⁽²⁷³⁾
90700	Row count dialog	TrvActionTableInsertRowsBelow ⁽³⁰¹⁾
90800	Picture properties dialog	TrvActionItemProperties ⁽³²⁶⁾
90900	Horizontal line properties dialog	TrvActionItemProperties ⁽³²⁶⁾
91000	Table properties dialog	TrvActionTableProperties ⁽³⁰³⁾ TrvActionItemProperties ⁽³²⁶⁾
90910	Sidenote properties dialog	TrvActionItemProperties ⁽³²⁶⁾
90920	Number properties dialog	TrvActionItemProperties ⁽³²⁶⁾
90930	Page number and page count properties dialog	TrvActionItemProperties ⁽³²⁶⁾
90940	Equation dialog	TrvActionInsertEquation ⁽²²⁸⁾ TrvActionItemProperties ⁽³²⁶⁾
91100	Bullets and numbering dialog	TrvActionParaList ⁽²⁰⁸⁾
91200	HTML bullets and numbering dialog	TrvActionParaList ⁽²⁰⁸⁾ (if RVAControlPanel.UserInterface ⁽⁵⁸⁾ = <i>rvaiHTML</i>)
91300	Paste special dialog	TrvActionPasteSpecial ⁽¹³⁹⁾
91301	Language dialog	RVA_ChooseLanguage ⁽³⁵¹⁾ function

91400	Page setup dialog	TrvActionPageSetup ⁽¹¹⁶⁾
91500	Paragraph border and background dialog	TrvActionParaBorder ⁽¹⁹³⁾
91600	Bullets and numbering customization dialog	TrvActionParaList ⁽²⁰⁸⁾
91700	Paragraph dialog	TrvActionParagraph ⁽²⁰⁰⁾
91800	Print preview dialog	TrvActionPrintPreview ⁽¹²⁰⁾
91900	Cell spacing dialog	TrvActionTableProperties ⁽³⁰³⁾ TrvActionItemProperties ⁽³²⁶⁾
92000	Cell splitting dialog	TrvActionTableSplitCells ⁽³¹⁰⁾
92100	Table background dialog	TrvActionTableProperties ⁽³⁰³⁾ TrvActionItemProperties ⁽³²⁶⁾
92200	Table border dialog	TrvActionTableProperties ⁽³⁰³⁾ TrvActionItemProperties ⁽³²⁶⁾
92300	Text and paragraph styles dialog	TrvActionStyleTemplates ⁽²¹⁴⁾ TrvActionAddStyleTemplate ⁽²¹⁷⁾
92400	Text and paragraph styles import dialog	TrvActionStyleTemplates ⁽²¹⁴⁾ TrvActionAddStyleTemplate ⁽²¹⁷⁾ (when importing styles from a file)
92500	Number insertion dialog	TrvActionInsertNumber ⁽²⁴⁷⁾
92600	Visible sides dialog	TrvActionTableProperties ⁽³⁰³⁾ TrvActionItemProperties ⁽³²⁶⁾
92700	Caption insertion dialog	TrvActionInsertCaption ⁽²²⁵⁾
92800	Object alignment dialog	TrvActionVAlign ⁽³³⁵⁾
92900	Table sort dialog	TrvActionTableSort ⁽³⁰⁹⁾
93000	Bookmark dialog	TrvActionBookmarks ⁽²²³⁾
93100	Visual styles dialog	RVA_ChooseStyle ⁽³⁸⁰⁾ function
93110	File encoding dialog	File saving and loading actions (when choosing encoding of ANSI text files)

93120	Standard text and paragraph styles dialog	TrvActionStyleTemplates ⁽²¹⁴⁾ TrvActionAddStyleTemplate ⁽²¹⁷⁾ (when adding a standard style)
93130	Table delimiters dialog	TrvActionTableToText ⁽³¹¹⁾

ScaleRichView

HelpContext	HelpKeyword	Dialog is displayed by
91400	Page setup dialog	TsrvActionPageSetup

ScaleRichView introduces an alternative action for displaying a page setup dialog: TsrvActionPageSetup. When using TSRichViewEdit, TsrvActionPageSetup must be used instead of TrvActionPageSetup⁽¹¹⁶⁾. It's assumed that only one of these actions are used in an application, so these dialogs have the same HelpContext and HelpKeyword.

ReportWorkshop

HelpContext	HelpKeyword	Dialog is displayed by
95000	Report table cell dialog	TrvrActionCellProperties
95100	Background diagram dialog	TrvrActionCellProperties
95200	Color scale dialog	TrvrActionCellProperties
95300	Row generation rules dialog	TrvrActionRowGenerationRules
95400	Cross tabulation dialog	TrvrActionCrossTab
95500	Report properties dialog	TrvrActionDocProperties
95600	Background diagram type dialog	TrvrActionCellProperties
95700	Shape properties dialog	TrvActionItemProperties ⁽³²⁶⁾
95800	Report table insertion dialog	TrvrActionInsertTable
95900	Report Wizard	TrvrActionReportWizard
96000	Master-detail link dialog	TrvrActionReportWizard
96100	Chart dialog	TrvrActionInsertChart

Default value:

htContext

See also:

- UseHelpFiles⁽⁵⁸⁾

- UseDefaultHelpFile⁽⁵⁷⁾

1.2.4.1.23 TRVAControlPanel.HTMLComponent (deprecated)

Deprecated. Do not assign this property: TRichView HTML methods provide better results without it.

Defines a link to a component allowing to import and paste HTML files.

property HTMLComponent: TRVHTMLComponent⁽³⁸⁹⁾;

You can assign either TRvHtmlImporter⁽³⁷⁰⁾ or TrvHtmlViewImporter⁽³⁷¹⁾ component to this property.

If this link is defined, HTML format can be used in TrvActionInsertFile⁽²³⁰⁾, TrvActionPaste⁽¹³⁷⁾, and TrvActionPasteSpecial⁽¹³⁹⁾ actions.

1.2.4.1.24 TRVAControlPanel.Language

Defines the language for user interface.

property Language: TRVLanguageName⁽³⁵⁰⁾;

This property returns a language name in English.

You can assign either English or native language name to this property.

See Localization of RichViewActions⁽³⁴²⁾.

1.2.4.1.25 TRVAControlPanel.MetafileCompatibility

Specifies whether the printing output is compatible with metafiles.

property MetafileCompatibility: Boolean;

This property is ignored when printing TRichViewEdit, because TrvActionPrint⁽¹¹⁸⁾, TrvActionQuickPrint⁽¹²²⁾, and TrvActionPrintPreview⁽¹²⁰⁾ use RVPrint⁽⁵⁵⁾. MetafileCompatibility property instead.

This property is used when printing ScaleRichView (see TsrvActionPrint and TsrvActionQuickPrint actions in the ScaleRichView help file).

Default value:

False

1.2.4.1.26 TRVAControlPanel.PixelBorders

Specifies pixels as the units of measurement for widths of lines in RichViewActions dialogs.

property PixelBorders: Boolean;

If this property is *True*, widths of lines in RichViewActions dialogs are measured in pixels, even if UnitsDisplay⁽⁵⁶⁾ $\langle \rightarrow \text{rvuPixels} \rangle$.

This property is used by the following actions:

- TrvActionParaBorder⁽¹⁹³⁾ for border widths, including an internal width;
- TrvActionTableProperties⁽³⁰³⁾ for widths of a table and cells borders;
- TrvActionItemProperties⁽³²⁶⁾ for widths of a table and cells borders; for width of horizontal lines.

Default value:*False***1.2.4.1.27 TRVAControlPanel.RVFFilter**

Defines the file filter for RVF format.

property RVFFilter: TRVALocString⁽³⁵⁰⁾;

This value is used only if RVFLocalizable⁽⁵⁴⁾ = *False*. This string will appear in file filters in dialogs used in TrvActionOpen⁽¹⁰⁸⁾, TrvActionSaveAs⁽¹²⁹⁾, TrvActionExport⁽¹¹²⁾, TrvActionInsertFile⁽²³⁰⁾, TrvActionStyleTemplates⁽²¹⁴⁾ (style import) actions.

If RVFLocalizable⁽⁵⁴⁾ = *True*, this property is ignored (RVF file filter name is assigned automatically when a new language is applied).

Default value:

'RichView Files (*.rvf) | *.rvf'

See also properties:

- RVStylesFilter⁽⁵⁵⁾
- DefaultExt⁽⁴⁴⁾

1.2.4.1.28 TRVAControlPanel.RVFLocalizable

Specifies whether UI text related to RVF format is maintained automatically by the localization procedures.

property RVFLocalizable: Boolean;

If *True*, all text related to RVF format is maintained automatically (i.e. localized text is used when UI language is changed).

If *False*, RVFFilter⁽⁵⁴⁾, RVFormatTitle⁽⁵⁴⁾, RVStylesFilter, RVStylesExt⁽⁵⁵⁾ properties are used.

Default value:*True***See also:**

- XMLLocalizable⁽⁶⁰⁾
- DefaultExt⁽⁴⁴⁾

1.2.4.1.29 TRVAControlPanel.RVFormatTitle

Specifies how RVF (RichView Format) appears in the "Paste Special" dialog.

property RVFormatTitle: TRVALocString⁽³⁵⁰⁾;

This value is used only if RVFLocalizable⁽⁵⁴⁾ = *False*. This string will appear in the dialog used in TrvActionPasteSpecial⁽¹³⁹⁾ action and in RVA_GetProgressMessage⁽³⁴⁵⁾ function.

If RVFLocalizable⁽⁵⁴⁾ = *True*, this property is ignored (RVF title is assigned automatically when a new language is applied).

Default value:

'RichView Format'

1.2.4.1.30 TRVAControlPanel.RVPrint

Contains a link to TRVPrint component.

property RVPrint: TRVPrint;

This link (if defined) is used by TrvActionPrint⁽¹¹⁸⁾, TrvActionQuickPrint⁽¹²²⁾, TrvActionPrintPreview⁽¹²⁰⁾, TrvActionPageSetup⁽¹¹⁶⁾ actions.

For TrvActionPageSetup⁽¹¹⁶⁾, this link *must* be defined, otherwise the action will be disabled (because this action reads/writes margins from/to properties of this component).

1.2.4.1.31 TRVAControlPanel.RVStylesFilter, RVStylesExt

RVStylesFilter defines the file filter for files containing style templates.

RVStylesExt defines the file extension for files containing style templates.

property RVStylesFilter: TRVALocString⁽³⁵⁰⁾;

property RVStylesExt: TRVALocString⁽³⁵⁰⁾;

These values are used only if RVFLocalizable⁽⁵⁴⁾=False. They will appear in file filters in the style import and export dialogs used in TrvActionStyleTemplates⁽²¹⁴⁾, TrvActionAddStyleTemplate⁽²¹⁷⁾ actions.

If RVFLocalizable⁽⁵⁴⁾=True, these properties are ignored (a file filter and extension are assigned automatically when a new language is applied).

Default values:

- RVStylesFilter: 'RichView Styles (*.rvst)|*.rvst'
- sRVStylesExtension: 'rvst'

See also properties:

- RVFFilter⁽⁵⁴⁾

1.2.4.1.32 TRVAControlPanel.SearchScope

Defines a search scope for TrvActionFind⁽¹³⁴⁾, TrvActionFindNext⁽¹³⁶⁾, and TrvActionReplace⁽¹⁴³⁾ actions.

type

```
TRVASearchScope = (rvssFromCursor, rvssAskUser, rvssGlobal);
```

property SearchScope: TRVASearchScope;

Value	Meaning
<i>rvssFromCursor</i>	Searching from the caret position to the end/beginning of the document.
<i>rvssAskUser</i>	Searching from the caret position to the end/beginning of the document, then asking the user whether to continue from the beginning/end, then (if "yes") searching from the beginning/end.
<i>rvssGlobal</i>	Searching from the caret position to the end/beginning of the document, then searching from the beginning/end.

Default value:*rvssAskUser***1.2.4.1.33 TRVAControlPanel.ShowSoftPageBreaks**

Specifies whether page breaks should be displayed in TCustomRichViewEdit component when possible.

property ShowSoftPageBreaks: Boolean;

Soft page breaks can be displayed after the document is formatted for printing (after executing TrvActionPrint⁽¹¹⁸⁾, TrvActionQuickPrint⁽¹²²⁾, or TrvActionPrintPreview⁽¹²⁰⁾ actions) until it is changed.

Default value:*True***1.2.4.1.34 TRVAControlPanel.SpellInterface**

Allows using a third-party components to correct spelling errors and to find synonyms.

property SpellInterface: TRVACustomSpellInterface⁽³⁵⁹⁾;

You can assign one of the following components to this property:

- TRVAAddictSpellInterface⁽³⁵⁸⁾ (to use Addict⁽³⁶⁴⁾)
- TRVAASpellInterface⁽³⁵⁹⁾ (to use ASpell⁽³⁶⁵⁾)
- TRVADXSpellInterface⁽³⁶⁰⁾ (to use ExpressSpellChecker⁽³⁶⁶⁾)
- TRVAHunSpellInterface⁽³⁶¹⁾ (to use HunSpell⁽³⁶⁷⁾)
- TRVAPolarSpellInterface⁽³⁶¹⁾ (to use Polar SpellChecker⁽³⁶⁸⁾)

1.2.4.1.35 TRVAControlPanel.TableGridStyle

Defines the pen style for drawing grid lines in table.

property TableGridStyle: TPenStyle;

This property is used in TrvActionTableGrid⁽²⁹⁸⁾ action.

Default value:*psDot***1.2.4.1.36 TRVAControlPanel.UnitsDisplay**

Specifies units of measurement for using in RichViewActions dialogs.

property UnitsDisplay: TRVUnits;

These units are used for entering lengths in dialog windows. For pixels, only integer values are allowed. Values measured in other units can be fractional.

Small lengths, for example paragraph spacing, are measured either in pixels (if **UnitsDisplay**=*rvuPixels*) or in points (if **UnitsDisplay** has another value).

Line widths can be measured in pixels even if **UnitsDisplay**<>*rvuPixels*, if PixelBorders⁽⁵³⁾=*True*.

Units of rulers must be assigned separately, see TRVRuler⁽⁸²⁾.UnitsDisplay⁽⁸⁷⁾.

Default value:*rvuPixels*

1.2.4.1.37 TRVAControlPanel.UnitsProgram

Specifies units of measurement for properties of TRVStyleLength type of some actions.

property UnitsProgram: TRVStyleUnits;

These units are used for properties of the type TRVStyleLength of the following actions:

- TrvActionNew⁽¹⁰⁵⁾,
- TrvActionIndentInc⁽¹⁸⁹⁾,
- TrvActionIndentDec⁽¹⁸⁸⁾,
- TrvActionParaList⁽²⁰⁸⁾,
- TrvActionParaBullets⁽¹⁹⁷⁾,
- TrvActionParaNumbering⁽²¹¹⁾,
- TrvActionStyleTemplates⁽²¹⁴⁾,
- TrvActionInsertPicture⁽²⁵¹⁾,
- TrvActionInsertHLine⁽²³³⁾,
- TrvActionInsertTable⁽²⁷³⁾.

Not all actions' properties of the type TRVStyleLength are measured in these units. Generally, if the action takes values from the editor, modifies them and applies back to the editor, properties of this action are measured in Editor.Style.Units (where Editor is the target TRichViewEdit control). However, if the action applies some predefined values to the editor, these values are measured using this property.

At design time, values of all action's properties measured in these units are displayed with units ('px' or 'tw') in the Object Inspector (in Delphi 5 or newer). Values of properties measured in Editor.Style.Units are not displayed with units.

Note: assigning a new value to this property does not convert values of actions' properties. To convert properties to new units of measurement, RVA_ConvertToPixels/RVA_ConvertToTwips⁽³⁸⁰⁾ must be called before changing value of this property.

Default value:

rvstuPixels

1.2.4.1.38 TRVAControlPanel.UseDefaultHelpFile

Specifies whether Application.HelpFile can be used.

property UseDefaultHelpFile: Boolean;

This property is ignored if UseHelpFiles⁽⁵⁸⁾ = *False*.

Application.HelpFile is used only if other help file is not defined⁽³⁴⁹⁾ for the current UI Language⁽⁵³⁾.

Default value:

False

See also:

- HelpType⁽⁴⁹⁾

1.2.4.1.39 TRVAControlPanel.UseHelpFiles

Specifies whether help files should be used if available.

property UseHelpFiles: Boolean;

Help files can be:

- provided for RichViewActions; each language ⁽⁵³⁾ may have its own help file (see RVA_SetHelpFile ⁽³⁴⁹⁾ procedure).
- if UseDefaultHelpFile ⁽⁵⁷⁾ = *True*, and the current language does not have a help file specified, RichViewActions uses Application.HelpFile.

If a help file is available, additional "Help" button is shown in all RichViewActions dialogs.

Default value:

True

See also:

- HelpType ⁽⁴⁹⁾
- UseDefaultHelpFile ⁽⁵⁷⁾

1.2.4.1.40 TRVAControlPanel.UserInterface

Allows hiding UI options incompatible with some formats.

type

```
TRVAUserInterface = (rvaiFull, rvaiHTML, rvaiRTF, rvaiText);
```

property UserInterface: TRVAUserInterface;

Value	Meaning
<i>rvaiFull</i>	All user interface controls are available. Optimal for RVF documents.
<i>rvaiHTML</i>	All user interface controls providing options incompatible with HTML are hidden. The most noticeable changes: <ul style="list-style-type: none"> • an alternative HTML-style dialog in TrvActionParaList ⁽²⁰⁸⁾ action; • a hidden "Tabs" page in the TrvActionParagraph ⁽²⁰⁰⁾ action's dialog.
<i>rvaiRTF</i>	All user interface controls providing options incompatible with RTF are hidden. The most noticeable changes: <ul style="list-style-type: none"> • TrvActionColor ⁽³¹⁷⁾, TrvActionVAlign ⁽³³⁵⁾, TrvActionBackground ⁽³¹⁵⁾, TrvActionTableCellRotation180 ⁽²⁹¹⁾ are disabled; • in the table properties dialog, table background is disabled, some options for table borders are not available; • in the image properties dialog, image alignment is not available; • in the paragraph borders dialog, some options are disabled

<i>rvauitext</i>	All user interface controls providing options incompatible with a plain text are hidden or disabled.
------------------	--

Default value:*rvauitFull*

1.2.4.1.41 TRVAControlPanel.UseTextCodePageDialog

Specifies whether the file-related actions can display a dialog for choosing a text file codepage.

property `UseTextCodePageDialog: Boolean;`

This property affects the following actions:

- TrvActionOpen ⁽¹⁰⁸⁾
- TrvActionExport ⁽¹¹²⁾
- TrvActionSave ⁽¹²⁴⁾
- TrvActionInsertFile ⁽²³⁰⁾

It affects loading and saving text files in the editor containing Unicode text.

If *False*, non-Unicode text files are saved using the system-default code page; a code page for insertion is taken from the Charset of the current text.

If *True*, a dialog for choosing a code page is displayed (the saving and exporting actions display this dialog only if the document is completely in Unicode). In addition to ANSI code pages, it allows choosing UTF-8 and UTF-16.

Default value:*True*

1.2.4.1.42 TRVAControlPanel.UseXPThemes

Specifies whether controls on RichViewActions dialog should use Windows themes (visual styles) if available.

property `UseXPThemes: Boolean;`

This property affects the following controls placed on dialogs:

- TRichView
- TRVPrintPreview
- TRVOfficeRadioButton
- TRVOfficeRadioGroup
- TRVColorCombo
- TRVColorGrid
- color picker window used in TrvActionCustomColor ⁽³²⁰⁾

Other controls use themes if it is supported by Delphi (and a manifest is included).

Default value:*True*

1.2.4.1.43 TRVAControlPanel.XMLComponent

Defines a link to a component allowing to load and save XML files.

property XMLComponent: TRVXMLComponent⁽³⁸⁹⁾;

You can assign TRichViewXML⁽³⁷⁰⁾ component to this property.

If this link is defined, XML format can be used in TrvActionOpen⁽¹⁰⁸⁾, TrvActionSaveAs⁽¹²⁹⁾, TrvActionExport⁽¹¹²⁾ and TrvActionInsertFile⁽²³⁰⁾ actions.

See also:

- XMLFilter⁽⁶⁰⁾
- XMLLocalizable⁽⁶⁰⁾

1.2.4.1.44 TRVAControlPanel.XMLFilter

Defines the file filter for XML format.

property XMLFilter: TRVALocString⁽³⁵⁰⁾;

This property is used if XMLComponent⁽⁶⁰⁾ is assigned.

This value is used only if XMLLocalizable⁽⁶⁰⁾=*False*. This string will appear in file filters in dialogs used in TrvActionOpen⁽¹⁰⁸⁾, TrvActionSaveAs⁽¹²⁹⁾, TrvActionExport⁽¹¹²⁾, TrvActionInsertFile⁽²³⁰⁾ actions.

If XMLLocalizable⁽⁶⁰⁾=*True*, this property is ignored (XML file filter name is assigned automatically when a new language is applied).

Default value:

'XML Files (*.xml)|*.xml'

1.2.4.1.45 TRVAControlPanel.XMLLocalizable

Specifies whether UI text related to XML format is maintained automatically by the localization procedures.

property RVFLocalizable: Boolean;

This property is used if XMLComponent⁽⁶⁰⁾ is assigned.

If *True*, all text related to XML format is maintained automatically (i.e. localized text is used when UI language is changed).

If *False*, XMLFilter⁽⁶⁰⁾ property is used.

Default value:

True

See also:

- RVFLocalizable⁽⁵⁴⁾

1.2.4.2 Methods

In TRVAControlPanel

Activate⁽⁶¹⁾

DoneImportPicturesAndFiles⁽⁶¹⁾

InitImportPicturesAndFiles⁽⁶¹⁾

1.2.4.2.1 TRVAControlPanel.Activate

Makes this control panel the main control panel in the application.

procedure Activate;

This method is useful if you have more than one control panel in an application. The first created control panel becomes the main control panel automatically.

The main control panel is used for actions and components that do not have a link to the specific control panel.

The main control panel is returned in MainRVAControlPanel⁽³⁹⁰⁾ variable.

See also:

- TrvCustomAction.ControlPanel⁽³³⁷⁾
- TRVStyleTemplateCombo/ListBox.ControlPanel⁽⁹¹⁾

1.2.4.2.2 TRVAControlPanel.GetRealDialogFontName

Returns the font name used in all dialog windows of RichViewActions.

function GetRealDialogFontName: TFontName;

The function uses DialogFontName/DialogFontNameLin⁽⁴⁵⁾ properties.

1.2.4.2.3 TRVAControlPanel.InitImportPictures, DoneImportPictures, DoImportPicture

Allows using DownloadInterface⁽⁴⁸⁾ component for a loading operation that were not initialized by RichViewActions.

```
procedure InitImportPicturesAndFiles(Sender: TrvAction(313);
  Editor: TCustomRichViewEdit);
procedure DoneImportPicturesAndFiles(
  Editor: TCustomRichViewEdit);
```

Normally, when loading is initiated by an action, DownloadInterface⁽⁴⁸⁾ is used automatically. The following actions initializes loading:

- TrvActionOpen⁽¹⁰⁸⁾
- TrvActionInsertFile⁽²³⁰⁾
- TrvActionPaste⁽¹³⁷⁾
- TrvActionPasteSpecial⁽¹³⁹⁾

However, when loading is not initiated by an action, DownloadInterface⁽⁴⁸⁾ is not used. To solve this problem, call **InitImportPicturesAndFiles** before loading, and **DoneImportPicturesAndFiles** after.

Parameters:

Sender will be used as a parameter of OnDownload⁽⁶⁶⁾ event

Editor is a target editor.

Example:

This code allows downloading external pictures referred by files that are inserted in the editor as a result of drag-and-drop operation.

```
procedure TMyForm.MyRichViewEditBeforeOleDrop(Sender: TObject);
begin
    MyRVAControlPanel.InitImportPicturesAndFiles(nil,
        (Sender as TCustomRichViewEdit));
end;
procedure TForm3.RichViewEdit1AfterOleDrop(Sender: TObject);
begin
    MyRVAControlPanel.DoneImportPicturesAndFiles(
        (Sender as TCustomRichViewEdit));
end;
```

1.2.4.3 Events

In TRVAControlPanel

- OnAddStyle⁶²
- OnBackgroundChange⁶³
- OnChoosePicture⁶³
- OnCreateForm⁶⁴
- OnCustomFileOperation⁶⁴
- OnDownload⁶⁶
- OnGetActionControlCoords⁶⁶
- OnGetHeaderFooterCode⁶⁷
- OnMarginsChanged⁶⁷
- OnStyleNeeded⁶⁸
- OnViewChanged⁶⁸

1.2.4.3.1 TRVAControlPanel.OnAddStyle

Occurs after some action added a style as a result of an editing operation.

type

```
TRVAddStyleEvent = procedure (Sender: TrvAction313;
    StyleInfo: TCustomRVInfo) of object;
```

property OnAddStyle: TRVAddStyleEvent;

Actions may add text, paragraph, and list styles as a result of editing operations.

Depending on the style type, **StyleInfo** can be of TFontInfo, TParalInfo, or TRVListInfo class.

If you need to perform additional operations on new styles, use this event.

See also:

- OnStyleNeeded⁶⁸
- TCustomRichView.OnAddStyle

1.2.4.3.2 TRVAControlPanel.OnBackgroundChange

Occurs after TrvActionBackground⁽³¹⁵⁾ or TrvActionColor⁽³¹⁷⁾ actions change background properties.

property OnBackgroundChange: TRVAEditEvent⁽³⁸⁶⁾;

1.2.4.3.3 TRVAControlPanel.OnChoosePicture

Using this event you can provide your own user interface for opening a picture.

type

```
TRVChoosePictureEvent = procedure (Sender: TObject;
  Editor: TCustomRichViewEdit;
  OwnerObject: TObject; const DialogTitle: TRVALocString(350);
  out FileName: TRVUnicodeString;
  out Graphic: TGraphic; var DoDefault: Boolean) of object;

property OnChoosePicture: TRVChoosePictureEvent;
```

This event is called in the following cases:

Picture for...	Called by the action	OwnerObject parameter
New picture item	TrvActionInsertPicture ⁽²⁵¹⁾	<i>nil</i>
Existing picture item	TrvActionItemProperties ⁽³²⁶⁾	Picture item (TrvGraphicItemInfo or inherited)
Document background	TrvActionBackground ⁽³¹⁵⁾	Editor (inherited from TCustomRichViewEdit)
Table background	TrvActionItemProperties ⁽³²⁶⁾ TrvActionTableProperties ⁽³⁰³⁾	Table (TRVTableItemInfo or inherited)
Table cell background	TrvActionItemProperties ⁽³²⁶⁾ TrvActionTableProperties ⁽³⁰³⁾	Table (TRVTableItemInfo or inherited)
List marker picture	TrvActionParaList ⁽²⁰⁸⁾	List level (TRVListLevel)

Input parameters:

Sender – the action that called the event

Editor – the target editor

OwnerObject depends on the action, see the table above.

DialogTitle – a suggested dialog title (if it is an empty string, a default dialog title must be used)

DoDefault = *True*

Input parameters:

Graphic – chosen picture; if the user canceled the selection, it must be *nil*.

FileName – file name (or another picture identifier). Depending on settings, it may be stored in the item.

DoDefault:

- if *True*, the action must perform the default way of picture choosing (using TOpenPicture dialog); in this case, **Graphic** must be *nil*;
- if *False*, and **Graphic** is not *nil*, the action must assign **Graphic** to the item. If **Graphic** is *nil*, the action should do nothing.

1.2.4.3.4 TRVAControlPanel.OnCreateForm

Occurs before any RichViewActions' dialog window is displayed.

type

```
TRVCreateFormEvent = procedure (Form: TForm) of object;
```

property OnCreateForm: TRVCreateFormEvent;

The event allows setting properties to the form before it is displayed.

1.2.4.3.5 TRVAControlPanel.OnCustomFileOperation

Occurs when the application needs to save or load a file in a custom format.

type

```
TRVAFileOperation = (rvafoOpen, rvafoSave, rvafoExport, rvafoInsert);
TRVCustomFileOperationEvent = procedure (Sender: TrvAction313;
  Edit: TCustomRichViewEdit; const FileName: TRVUnicodeString;
  Operation: TRVAFileOperation; var SaveFormat: TrvFileSaveFilter388;
  var CustomFilterIndex: Integer; var Success: Boolean) of object;
```

property OnCustomFileOperation: TRVCustomFileOperationEvent;

This event occurs when the following actions are executed:

- TrvActionOpen¹⁰⁸ (**Operation**=*rvafoOpen*)
- TrvActionSave¹²⁴ (**Operation**=*rvafoSave*)
- TrvActionExport¹¹² (**Operation**=*rvafoExport*)
- TrvActionInsertFile²³⁰ (**Operation**=*rvafoInsertFile*)

FileName is a file name (full path) for opening/saving.

CustomFilterIndex is an index of custom format (in the CustomFilter property of the corresponding action), from 1.

For an opening operation, you can define a saving format in **SaveFormat** parameter (this format will be used for subsequent saving of document loaded from this file); you can also change **CustomFilterIndex** (on input, it corresponds to TrvActionOpen.CustomFilter¹¹⁰; on output, it must correspond to TrvActionSaveAs.CustomFilter¹⁰⁰, if **SaveFormat**=*ffeCustom*). For all other operations, **SaveFormat** and output value of **CustomFilterIndex** are ignored.

Set **Success** to *True* if you loaded/saved file successfully.

See also:

- TrvActionSave.SuppressNextErrorMessage¹²⁷

You can use LoadCSV⁽³⁸²⁾ function to implement an insertion of CSV files.

Example

Let you implemented saving and loading TRichView in your format. Let its extension is 'myf'.

Let you implemented the functions:

```
function SaveToMyFormat(Edit: TCustomRichViewEdit;
  const FileName: TRVUnicodeString): Boolean;
function LoadFromMyFormat(Edit: TCustomRichViewEdit;
  const FileName: TRVUnicodeString): Boolean;
```

These function must return *True* on success.

First, assign CustomFilter property of TrvActionOpen⁽¹⁰⁸⁾ and TrvActionSaveAs⁽¹²⁹⁾. You can do it at design time in the Object Inspector, or in code:

```
rvActionsResource.rvActionOpen1.CustomFilter(110) :=
  'My Files (*.myf)|*.myf';
rvActionsResource.rvActionSaveAs1.CustomFilter(100) :=
  'My Files (*.myf)|*.myf';
```

Make sure that *ffiCustom* is in TrvActionOpen.Filter⁽¹¹⁰⁾, and *ffeCustom* is in TrvActionSaveAs.Filter⁽¹³¹⁾.

Next, process RVAContolPanel.OnCustomFileOperation event:

```
procedure TForm3.RVAControlPanel1CustomFileOperation(Sender: TrvAction(313);
  Edit: TCustomRichViewEdit; const FileName: TRVUnicodeString;
  Operation: TRVAFileOperation; var SaveFormat: TrvFileSaveFilter(388);
  var CustomFilterIndex: Integer; var Success: Boolean);
begin
  case Operation of
    rvafoOpen:
      begin
        try
          // This example assumes that there is only one custom open
          // format, so it does not check CustomFilterIndex
          // (it must be 1)
          Success := LoadFromMyFormat(Edit, FileName);
        end;
      rvafoSave:
        begin
          // This example assumes that there is only one custom save
          // format, so it does not check CustomFilterIndex
          // (it must be 1)
          Success := SaveToMyFormat(Edit, FileName);
        end;
      end;
  end;
end;
```

1.2.4.3.6 TRVAControlPanel.OnDownload

Occurs when DownloadInterface⁽⁴⁸⁾ downloads an image.

type

```
TRVDownloadEvent = procedure (Sender: TrvAction;  
    const Source: TRVUnicodeString) of object;
```

property OnDownload: TRVDownloadEvent;

For each image, this event occurs twice:

- before downloading, **Source** is the image file name (HTTP location)
- after downloading, **Source** is an empty string.

See also:

- how to display a localized text message about downloading⁽³⁴⁵⁾

1.2.4.3.7 TRVAControlPanel.OnGetActionControlCoords

The event requests coordinates of the control which called the given action.

type

```
TRVGetActionControlCoordsEvent = procedure (Sender: TrvAction;  
    var R: TRect) of object;
```

property OnGetActionControlCoords: TRVGetActionControlCoordsEvent;

This event may occur when one of the following actions needs to position a color-picker window relative to the control which called this action:

- TrvActionFontColor⁽¹⁵⁰⁾
- TrvActionFontBackColor⁽¹⁴⁹⁾
- TrvActionParaColor⁽¹⁹⁷⁾
- TrvActionColor⁽³¹⁷⁾

(if UserInterface⁽³²²⁾=rvacAdvanced)

This event occurs only when the component linked to the action is not a TControl (if a control executes the action, coordinates are calculated automatically).

Input parameters:

Sender – the action that is being executed.

R = Rect(0,0,0,0).

Output parameter:

R – screen coordinates of the control (if non-empty).

If this action is not processed, or if empty **R** is returned, a color-picker window will be displayed at the mouse pointer position.

Example:

This example shows how to work with TdxBarManager by [Developer Express](#):

```
procedure TMyForm.RVAControlPanel1GetActionControlCoords (  
    Sender: TrvAction; var R: TRect);  
begin  
    if (Sender.ActionComponent<>nil) and
```

```

    (Sender.ActionComponent is TdxBarItem) then
    with TdxBarItem(Sender.ActionComponent).ClickItemLink do begin
        R := BarControl.GetItemRect (Control);
        R.TopLeft := BarControl.ClientToScreen (R.TopLeft);
        R.BottomRight := BarControl.ClientToScreen (R.BottomRight);
    end;
end;

```

1.2.4.3.8 TRVAControlPanel.OnGetHeaderFooterCode

This event allows requesting a value for the code in the header/footer string.

type

```

TRVHeaderFooterCodeEvent = procedure (Sender: TObject;
    Edit: TCustomRichViewEdit; const Code: TRVUnicodeString;
    var Value: TRVUnicodeString) of object;

```

property OnGetHeaderFooterCode: TRVHeaderFooterCodeEvent;

The string is stored in Header⁽⁴⁹⁾.Text⁽³⁸⁶⁾ and Footer⁽⁴⁹⁾.Text⁽³⁸⁶⁾. It can be edited in the dialog displayed by TrvActionPageSetup⁽¹¹⁶⁾.

Input parameters:

Sender – TRVPrint component

Editor – the editor for printing.

Code – the one-letter code (English character). The following codes are reserved: 'P', 'p', 'd', 't'.

Value is empty on input.

Output parameter:

Value – text to insert in the place of this code.

1.2.4.3.9 TRVAControlPanel.OnMarginsChanged

Occurs after some action changes TCustomRichViewEdit control's margins (LeftMargin, TopMargin, RightMargin, and BottomMargin properties).

property OnMarginsChanged: TRVAEditEvent⁽³⁸⁶⁾;

Margins can be changed by TrvActionOpen⁽¹⁰⁸⁾ and TrvActionBackground⁽³¹⁵⁾ actions.

If you use TRVRuler⁽⁸²⁾, you must update it in this event:

```

procedure TForm1.RVAControlPanel1MarginsChanged(Sender: TrvAction(313);
    Edit: TCustomRichViewEdit);
begin
    RVRuler1.UpdateRulerMargins(89);
end;

```

1.2.4.3.10 TRVAControlPanel.OnStyleNeeded

Occurs when some action needs a style for an editing operation.

type

```
TRVStyleNeededEvent = procedure (Sender: TrvAction313;  
  RVStyle: TRVStyle; StyleInfo: TCustomRVInfo;  
  var StyleNo: Integer) of object;
```

property OnStyleNeeded: TRVStyleNeededEvent;

Actions may need text, paragraph and list styles.

StyleInfo is a sample of the required style. **Sender** needs a style with properties like in **StyleInfo**.

Depending on the style type, **StyleInfo** can be of TFontInfo, TParaInfo, or TRVListInfo class.

You need to return **StyleNo**, the index of style in the proper collection (TextStyles, ParaStyles, ListStyles) of **RVStyle** component. If you return -1, the action will perform a default processing (reusing existing styles if possible, adding new styles otherwise).

This event is rarely used. You can use it, for example, to disallow adding new styles (always return an index of an existing style the most similar to **StyleInfo**). Or you can prevent some style attributes from appearing in document as a result of editing operation (by changing **StyleInfo** properties and returning -1 in **StyleNo**).

1.2.4.3.11 TRVAControlPanel.OnViewChanged

Occurs when some action changes view properties of the target editor.

property OnViewChanged: TRVAEditEvent2³⁸⁶;

This event is called by TrvActionNew¹⁰⁵ and TrvActionOpen¹⁰⁸ actions. In ScaleRichView, this event is also called by the actions that change view layouts and zooming in the target editor.

Usually, it makes sense to process this event only in ScaleRichView, if TSVRichViewEdit is used as a target editor. In this event you can update user interface components that display the current zooming percent.

1.2.5 TRVAPopupMenu

TRVAPopupMenu encapsulates the properties, methods, and events of a pop-up menu for TCustomRichViewEdit.

Unit RichViewActions, RVAPopupActionBar;

Syntax, if TNT Controls³⁷² are not used:

```
TRVAPopupMenu = class (TPopupMenu, IRVAPopupMenu)
```

Syntax, if TNT Controls³⁷² are used:

```
TRVAPopupMenu = class (TTntPopupMenu, IRVAPopupMenu)
```

Syntax:

```
TRVATBPopupMenu = class (TTBPopupMenu, IRVAPopupMenu)
```

```
TRVATBXPopupMenu = class (TTBXPopupMenu, IRVAPopupMenu)
```

```
TRVASPTBXPopupMenu = class (TSPTBXPopupMenu, IRVAPopupMenu)
```

```
TRVAPopupActionBar = class (TPopupActionBar, IRVAPopupMenu)
```


Hierarchy

(Hierarchy of TRVAPopupMenu, if TNT Controls⁽³⁷²⁾ are not used)

TObject

TPersistent

TComponent

TMenu

TPopupMenu

Description

Menus

There are several similar components representing pop-up menu:

- TRVAPopupMenu is inherited from the standard TPopupMenu, or, if TNT Controls⁽³⁷²⁾ are used, from TTntPopupMenu.
- TRVAPopupActionBar is inherited from TPopupMenuActionBar, available for Delphi 2006 or newer, defined in RVAPopupActionBar unit.
- TRVATBPopupMenu is available if Toolbar2000⁽³⁷³⁾ is used.
- TRVATBXPopupMenu is available if TBX⁽³⁷³⁾ is used.
- TRVASPTBXPopupMenu is available if SpTBXLib⁽³⁷²⁾ is used.

Standard actions

This menu is automatically filled with RichViewActions from ActionList⁽⁷¹⁾.

The following actions are always added (if available):

- TrvActionCut⁽¹³³⁾
- TrvActionCopy⁽¹³³⁾
- TrvActionPaste⁽¹³⁷⁾
- TrvActionFontEx⁽¹⁵¹⁾
- TrvActionParagraph⁽²⁰⁰⁾
- TrvActionParaList⁽²⁰⁸⁾

The following action is added if an image or a *break* is at the position of the caret:

- TrvActionItemProperties⁽³²⁶⁾

The following actions are added if there is a table with multicell selection:

- TrvActionTableInsertRowsAbove⁽³⁰⁰⁾
- TrvActionTableInsertRowsBelow⁽³⁰¹⁾
- TrvActionTableInsertColLeft⁽²⁹⁹⁾
- TrvActionTableInsertColRight⁽²⁹⁹⁾

The following actions are added if the table row(s)/column(s) are selected completely:

- TrvActionTableDeleteRows⁽²⁹⁷⁾
- TrvActionTableDeleteCols⁽²⁹⁷⁾

The following action is added if there is a table without edited cell:

- TrvActionFillColor⁽³²⁴⁾

The following action is added if there is a table with selection, and the selected cells can be merged:

- TrvActionTableMergeCells ⁽³⁰²⁾

Otherwise, the following action is added:

- TrvActionTableSplitCells ⁽³¹⁰⁾

The following action is added if the current item is a table:

- TrvActionTableProperties ⁽³⁰³⁾

Live spelling

This menu supports live spelling. If Addict 3 or 4 ⁽³⁶⁴⁾ is used, the live spelling commands appear automatically (suggestions for the current misspelled word, "Add to Dictionary" and "Ignore" commands). For other spellcheckers, you need to process the events:

- OnLiveSpellAdd ⁽⁷¹⁾
- OnLiveSpellGetSuggestions ⁽⁷²⁾
- OnLiveSpellIgnoreAll ⁽⁷²⁾

Adding custom commands

This menu is recreated each time before it is displayed, before OnPopup event occur. All menu items are deleted and new items are added.

If you need adding another commands (or delete some existing commands), do it in OnPopup event.

1.2.5.1 Properties

In TRVAPopupMenu

- ActionList ⁽⁷¹⁾
- MaxSuggestionsCount ⁽⁷¹⁾

Derived from TPopupMenu

- Alignment
- AutoHotkeys
- AutoLineReduction
- AutoPopup
- BiDiMode
- HelpContext
- Images
- MenuAnimation
- OwnerDraw
- ParentBiDiMode
- PopupComponent
- PopupPoint
- TrackButton

1.2.5.1.1 TRVAPopupMenu.ActionList

Defines a link to TActionList or TActionManager component.

property ActionList: TCustomActionList;

The menu can be filled with actions only if this link is defined.

1.2.5.1.2 TRVAPopupMenu.MaxSuggestionsCount

Specifies the maximal possible count of suggestions displayed for changing the current misspelled word.

property MaxSuggestionsCount: Integer;

If this property is set to positive value, it limits the count of suggestions displayed in the menu.

Other suggestions returned by Addict 3⁽³⁶⁴⁾ or OnLiveSpellGetSuggestions⁽⁷²⁾ event are ignored.

Default value:

0 (all suggestions are displayed)

1.2.5.2 Events

In TRVAPopupMenu

- OnLiveSpellAdd⁽⁷¹⁾
- OnLiveSpellGetSuggestions⁽⁷²⁾
- OnLiveSpellIgnoreAll⁽⁷²⁾
- OnLiveSpellWordReplace⁽⁷³⁾

Derived from TPopupMenu

OnChange
OnPopup

1.2.5.2.1 TRVAPopupMenu.OnLiveSpellAdd

Occurs when the user clicks "Add to Dictionary" command in the menu.

type

```
TRVALiveSpellAddEvent = procedure (Sender: TRVAPopupMenu(68);
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer) of object;
TRVA2LiveSpellAddEvent = procedure (Sender: TObject;
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer) of object;
```

For TRVAPopupMenu:

property OnLiveSpellAdd: TRVALiveSpellAddEvent;

For TRVATBPopupMenu, TRVATBXPopupMenu, TRVASPTBXPopupMenu:

property OnLiveSpellAdd: TRVA2LiveSpellAddEvent;

This event is used for live spelling check.

Word is a word for adding to the custom dictionary.

StyleNo is an index of text style for this word in **Edit.Style.TextStyles**.

If SpellInterface⁽⁵⁶⁾ is used, this command is processed automatically.

Otherwise, use this event for adding word to the user dictionary (this command appear in the menu only if this event is assigned).

1.2.5.2.2 TRVAPopupMenu.OnLiveSpellGetSuggestions

Occurs when the menu needs to add items containing suggestions for the current misspelled word.

type

```
TRVALiveSpellGetSuggestionsEvent = procedure (Sender: TRVAPopupMenu(68);
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer; Suggestions: TRVASpellStrings(387)) of object;
TRVA2LiveSpellGetSuggestionsEvent = procedure (Sender: TObject;
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer; Suggestions: TRVASpellStrings(387)) of object;
```

For TRVAPopupMenu:

property OnLiveSpellGetSuggestions: TRVALiveSpellGetSuggestionsEvent

For TRVATBPopupMenu, TRVATBXPopupMenu, TRVASPTBXPopupMenu:

property OnLiveSpellGetSuggestions: TRVA2LiveSpellGetSuggestionsEvent

This event is used for live spelling check.

Word is a misspelled word.

StyleNo is an index of text style for this word in **Edit.Style.TextStyles**.

Add suggestions to **Suggestions**.

If SpellInterface⁽⁵⁶⁾ is used, this command is processed automatically.

Otherwise, use this event for generating suggestions.

See also:

- MaxSuggestionsCount⁽⁷¹⁾

1.2.5.2.3 TRVAPopupMenu.OnLiveSpellIgnoreAll

Occurs when the user clicks "Ignore All" command in the menu.

type

```
TRVALiveSpellIgnoreAllEvent = procedure (Sender: TRVAPopupMenu(68);
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer) of object;
TRVA2LiveSpellIgnoreAllEvent = procedure (Sender: TObject;
  Edit: TCustomRichViewEdit; const Word: TRVUnicodeString;
  StyleNo: Integer) of object;
```

For TRVAPopupMenu:

property OnLiveSpellIgnoreAll: TRVALiveSpellIgnoreAllEvent;

For TRVATBPopupMenu, TRVATBXPopupMenu, TRVASPTBXPopupMenu:

property OnLiveSpellIgnoreAll: TRVA2LiveSpellIgnoreAllEvent;

This event is used for live spelling check.

Word is a word for adding to the ignore list.

StyleNo is an index of text style for this word in **Edit.Style.TextStyles**.

If SpellInterface⁽⁵⁶⁾ is used, this command is processed automatically.

Otherwise, use this event for adding word to the ignore list (this command appear in the menu only if this event is assigned).

1.2.5.2.4 TRVAPopupMenu.OnLiveSpellWordReplace

Occurs before **Word** is changed to **Correction** in **Edit**.

type

```
TRVALiveSpellReplaceEvent = procedure (Sender: TRVAPopupMenu;
    Edit: TCustomRichViewEdit; const Word, Correction: TRVUnicodeString;
    StyleNo: Integer) of object;
TRVA2LiveSpellReplaceEvent = procedure (Sender: TObject;
    Edit: TCustomRichViewEdit; const Word, Correction: TRVUnicodeString;
    StyleNo: Integer) of object;
```

For TRVAPopupMenu:

property OnLiveSpellWordReplace: TRVALiveSpellReplaceEvent;

For TRVATBPopupMenu, TRVATBXPopupMenu, TRVASPTBXPopupMenu:

property OnLiveSpellWordReplace: TRVA2LiveSpellReplaceEvent;

This event is used for live spelling check. It occurs when the user clicks on one of suggestions for the misspelled word in the menu.

StyleNo is an index of text style for this word in **Edit.Style.TextStyles**.

Processing this event is not necessary. It may be used if a spellchecker can use a history of replacements to generate better suggestions in future.

1.2.6 TRVFontCharsetComboBox

TRVFontCharsetComboBox is a combo-box for choosing a font character set.

Unit RVFontCombos;

```
TRVFontCharsetComboBox = class (TCustomRVFontComboBox(33))
```

Hierarchy

(Hierarchy of TRVFontCharsetComboBox, if TNT Controls⁽³⁷²⁾ are not used)

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomComboBox

TComboBox

TCustomRVFontComboBox⁽³³⁾

Description

This combo-box allows changing a font Charset for a selected text in Editor⁽³⁵⁾.

This component works automatically: place it on a form, assign Editor⁽³⁵⁾ and ActionFont⁽³⁵⁾ properties; no additional code is required.

Unicode note: for Unicode text, Charset does not affect appearance (except for SYMBOL_CHARSET).

In addition to character sets available for the current font, the combo-box may contain DEFAULT_CHARSET item, if AddDefaultCharset⁽⁷⁵⁾=*True*.

See also:

- TRVFontComboBox⁽⁷⁵⁾
- TRVFontSizeComboBox⁽⁸¹⁾

1.2.6.1 Properties

In TRVFontCharsetComboBox

- AddDefaultCharset⁽⁷⁵⁾
- DefaultCharsetCaption⁽⁷⁵⁾
FontName⁽⁷⁵⁾

Derived from TCustomRVFontComboBox⁽³³⁾

- ActionFont⁽³⁵⁾
- Editor⁽³⁵⁾

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint

- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.6.1.1 TRVFontCharsetComboBox.AddDefaultCharset

Specifies whether the combo-box must contain DEFAULT_CHARSET item.

property AddDefaultCharset: Boolean;

The caption for this item is defined in DefaultCharsetCaption⁽⁷⁵⁾ property.

Default value:

False

1.2.6.1.2 TRVFontCharsetComboBox.DefaultCharsetCaption

Specifies the caption for DEFAULT_CHARSET item.

property DefaultCharsetCaption: TRVLocString⁽³⁵⁰⁾;

This item is added if AddDefaultCharset⁽⁷⁵⁾=*True*.

The recommended value for this property: RVA_GetS⁽³⁴⁸⁾(rvam_font_DefaultCharset).

Default value:

'(Default)'

1.2.6.1.3 TRVFontCharsetComboBox.FontName

Contains a font name.

property FontName: TRVLocString⁽³⁵⁰⁾;

This property may be used when the combo-box is not linked with Editor⁽³⁵⁾. If linked, the combo-box items are maintained automatically.

When this property is assigned, the combo-box displays character sets available for this font.

1.2.7 TRVFontComboBox

TRVFontComboBox is a combo-box for choosing a font name.

Unit RVFontCombos;

TRVFontComboBox = **class** (TCustomRVFontComboBox⁽³³⁾)

Hierarchy

(Hierarchy of TRVFontComboBox, if TNT Controls⁽³⁷²⁾ are not used)

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomComboBox

TComboBox

TCustomRVFontComboBox ⁽³³⁾

Description

This combo-box allows changing a font name for a selected text in Editor ⁽³⁵⁾.

This component works automatically: place it on a form, assign Editor ⁽³⁵⁾ and ActionFont ⁽³⁵⁾ properties; no additional code is required.

The component displays preview of fonts in items. To disable this feature, assign Preview ⁽⁷⁸⁾ = *False*.

See also:

- TRVFontListBox ⁽⁷⁸⁾
- TRVFontSizeComboBox ⁽⁸¹⁾
- TRVFontCharsetComboBox ⁽⁷³⁾

1.2.7.1 Properties

In TRVFontComboBox

- AutoCharset ⁽⁷⁷⁾
- DropDownWidth ⁽⁷⁷⁾
- ItemHeight ⁽⁷⁷⁾
- Preview ⁽⁷⁸⁾
- SymbolPreviewString ⁽⁷⁸⁾

Derived from TCustomRVFontComboBox ⁽³³⁾

- ActionFont ⁽³⁵⁾
- Editor ⁽³⁵⁾

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength

- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.7.1.1 TRVFontComboBox.AutoCharset

Defines whether the combobox must change a text character set.

property AutoCharset: Boolean;

If *True*, the component changes not only font name, but also font' Charset. When applying a symbol font, the combo box applies SYMBOL_CHARSET. Otherwise, it applies DEFAULT_CHARSET.

Default value:

True

See also:

- TrvActionFontEx⁽¹⁵¹⁾.AutoApplySymbolCharset⁽¹⁵³⁾

1.2.7.1.2 TRVFontComboBox.DropDownWidth

Gets or sets the width of the of the drop-down portion of a combo box.

property DropDownWidth: TRVPixel96Length;

This property is used if Preview⁽⁷⁸⁾ = *True*.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

Default value:

300

1.2.7.1.3 TRVFontComboBox.ItemHeight

Gets or sets the height of items in the drop-down portion of a combo box.

property ItemHeight: TRVPixel96Length;

This property is used if Preview⁽⁷⁸⁾ = *True*.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

Default value:

22

1.2.7.1.4 TRVFontComboBox.Preview

Specifies whether the combo-box draws items using corresponding fonts.

property Preview: Boolean;

Default value:

True

See also:

- DropDownWidth⁽⁷⁷⁾
- ItemHeight⁽⁷⁷⁾
- SymbolPreviewString⁽⁷⁸⁾

1.2.7.1.5 TRVFontComboBox.SymbolPreviewString

Specifies a string to display after the font name for symbol fonts.

property SymbolPreviewString: TRVALocString⁽³⁵⁰⁾;

This property is used if Preview⁽⁷⁸⁾ = *True*.

For fonts of SYMBOL_CHARSET, their font name is drawn using Font.Name, and a preview is drawn next to the font name. This property is used to draw this preview.

Default value:

'Abc'

1.2.8 TRVFontListBox

TRVFontListBox is a list-box for choosing a font name.

Unit RVFontCombos;

TRVFontListBox = **class** (TCustomRVFontListBox⁽³⁵⁾)

Hierarchy

(Hierarchy of TRVFontListBox, if TNT Controls⁽³⁷²⁾ are not used)

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomListBox

TListBox

TCustomRVFontListBox⁽³⁵⁾

Description

This list-box allows changing a font name for a selected text in Editor⁽³⁷⁾.

This component works automatically: place it on a form, assign Editor⁽³⁷⁾ and ActionFont⁽³⁷⁾ properties; no additional code is required.

The component displays preview of fonts in items. To disable this feature, assign Preview⁽⁸⁰⁾ = *False*.

See also:

- TRVFontComboBox ⁷⁵

1.2.8.1 Properties

In TRVFontComboBox

- AutoCharset ⁸⁰
- ItemHeight ⁸⁰
- Preview ⁸⁰
- SymbolPreviewString ⁸⁰

Derived from TCustomRVFontListBox ³⁵

- ActionFont ³⁷
- Editor ³⁷

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.8.1.1 TRVFontListBox.AutoCharset

Defines whether the listbox must change a text character set.

property AutoCharset: Boolean;

If *True*, the component changes not only font name, but also font' Charset. When applying a symbol font, the combo box applies SYMBOL_CHARSET. Otherwise, it applies DEFAULT_CHARSET.

Default value:

True

See also:

- TrvActionFontEx⁽¹⁵¹⁾.AutoApplySymbolCharset⁽¹⁵³⁾

1.2.8.1.2 TRVFontListBox.ItemHeight

Gets or sets the height of items of the list box.

property ItemHeight: TRVPixel96Length;

This property is used if Preview⁽⁸⁰⁾ = *True*.

This value is measured in pixels at 96 pixels per inch; the actual size scaled according to the current screen pixel density.

Default value:

22

1.2.8.1.3 TRVFontListBox.Preview

Specifies whether the list-box draws items using corresponding fonts.

property Preview: Boolean;

Default value:

True

See also:

- ItemHeight⁽⁸⁰⁾
- SymbolPreviewString⁽⁸⁰⁾

1.2.8.1.4 TRVFontListBox.SymbolPreviewString

Specifies a string to display after the font name for symbol fonts.

property SymbolPreviewString: TRVLocString⁽³⁵⁰⁾;

This property is used if Preview⁽⁸⁰⁾ = *True*.

For fonts of SYMBOL_CHARSET, their font name is drawn using Font.Name, and a preview is drawn next to the font name. This property is used to draw this preview.

Default value:

'Abc'

1.2.9 TRVFontSizeComboBox

TRVFontSizeComboBox is a combo-box for choosing a font size (with the precision up to 0.5 point).

Unit RVFontCombos;

```
TRVFontSizeComboBox = class (TCustomRVFontComboBox33)
```

Hierarchy

(Hierarchy of TRVFontSizeComboBox, if TNT Controls³⁷² are not used)

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomComboBox

TComboBox

*TCustomRVFontComboBox*³³

Description

This combo-box allows changing a font size for a selected text in Editor³⁵.

This component works automatically: place it on a form, assign Editor³⁵ and ActionFont³⁵ properties; no additional code is required.

See also:

- TRVFontComboBox⁷⁵
- TRVFontCharsetComboBox⁷³

1.2.9.1 Properties

In TRVFontSizeComboBox

FontName⁸²

Derived from TCustomRVFontComboBox³³

- ActionFont³⁵
- Editor³⁵

Derived from TComboBox

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled

- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- MaxLength
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Sorted
- Style
- TabOrder
- TabStop
- Text
- Visible

1.2.9.1.1 TRVFontSizeComboBox.FontName

Contains a font name.

property FontName: TFontName;

This property may be used when the combo-box is not linked with Editor³⁵. If linked, the combo-box items are maintained automatically.

When this property is assigned, the combo-box displays font sizes available for this font. For True Type and Open Type fonts, it displays a predefined set of font sizes (8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 36, 48, 72).

1.2.10 TRVRuler

TRVRuler is a ruler that works with TRichViewEdit component.

Unit RVRuler;

TRVRuler = **class** (TRuler)

Hierarchy

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomControl
TCustomRuler
TRuler

Description

This component has many properties customizing its appearance and behavior. In this manual, we describe only the most important of them.

To associate the ruler with the editor, assign `TRichViewEdit` or `TDBRichViewEdit` control to `RichViewEdit`⁽⁸⁸⁾ property. The ruler can change the following parts in the editor:

- `LeftMargin` and `RightMargin`;
- indents of paragraphs;
- levels of paragraph lists;
- widths of table columns;
- tab stops in paragraphs (see `TRVRulerItemSelector`⁽³⁷⁾ component).

If `rvoClientTextWidth` is not included in `RichViewEdit`⁽⁸⁸⁾.Options, the ruler assigns `RichViewEdit`⁽⁸⁸⁾.`MinTextWidth` and `MaxTextWidth` according to its `PageWidth` property.

To set the ruler margins and page sizes according to `RichViewEdit`⁽⁸⁸⁾ settings, call `UpdateRulerMargins`⁽⁸⁹⁾. If you use `RichViewActions`, it's enough to call it in `TRVAControlPanel`⁽³⁹⁾.`OnMarginsChanged`⁽⁶⁷⁾ event.

To change the ruler's hints according to `TRVAControlPanel`⁽³⁹⁾.`Language`⁽⁵³⁾, call `RVALocalizeRuler`⁽³⁵³⁾.

Acknowledgement

Thanks to Pieter Zijlstra, who created the first version of `TRVRuler`.

Thanks to Rivarez, who implemented skin modes⁽⁸⁷⁾ for the rulers.

1.2.10.1 Properties

In `TRVRuler`

- `LineStyle`
- `RichViewEdit`⁽⁸⁸⁾
- `RVRulerOptions`

Derived from `TCustomRuler`

- `BiDiModeRuler`⁽⁸⁴⁾
- `BottomMargin`
- `DefaultTabWidth`
- `FirstIndent`
- `Flat`⁽⁸⁵⁾
- `IndentColor`⁽⁸⁵⁾
- `IndentSaturation`⁽⁸⁶⁾
- `IndentSettings`
- `Inset`
- `LeftIndent`
- `LeftMargin`
- `MarginColor`⁽⁸⁵⁾
- `MarginSaturation`⁽⁸⁶⁾

- MarginSettings
- MaxTabs
- Options
- TickColor ⁸⁵
- SkinType ⁸⁷
- RulerSaturation ⁸⁶
- RightIndent
- RightMargin
- RulerColor ⁸⁵
- RulerColorPageEnd ⁸⁵
- RulerTexts
- RulerType ⁸⁶
- ScreenRes ⁸⁶
- Tabs
- TabSettings
- TopMargin
- UnitsDisplay ⁸⁷
- UnitsPixelsPerInch ⁸⁷
- UnitsProgram ⁸⁷
- Zoom ⁸⁸

Derived from TCustomControl ---

- Align
- Anchors
- BiDiMode
- Color
- Constraints
- DragCursor
- DragKind
- DragMode
- Enabled
- Font
- Hint
- ParentBackground
- ParentBiDiMode
- ParentColor
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Visible

1.2.10.1.1 TCustomRuler.BiDiModeRuler

Allows forcing right-to-left or left-to-right ruler mode.

type

```
TBiDiModeRuler = (bmUseBiDiMode, bmLeftToRight, bmRightToLeft);
```

property BiDiModeRuler: TBiDiModeRuler;

Value	Meaning
<i>bmUseBiDiMode</i>	Left-to-right or right-to-left, depending in BiDiMode property
<i>bmLeftToRight</i>	Left-to-right
<i>bmRightToLeft</i>	Right-to-left

Default value*bmUseBiDiMode***1.2.10.1.2 TCustomRuler.Flat**

Switches between flat and 3d ruler appearance.

property Flat: Boolean;

This property works only if SkinType⁽⁸⁷⁾ = *stNoSkin*.

If you assign *False* to **Flat**, SkinType⁽⁸⁷⁾ is changed to *stNoSkin*.

Default value*False***1.2.10.1.3 TCustomRuler.IndentColor, MarginColor, RulerColor, TickColor, RulerColorPageEnd**

Colors of the ruler's visual elements.

property RulerColor: TColor;

property IndentColor: TColor;

property MarginColor: TColor;

property TickColor: TColor;

property RulerColorPageEnd: TColor;

RulerColor is used to paint an internal ruler area.

IndentColor is used to paint handles that change paragraph indents.

MarginColor is used to paint left and right margin areas (or top and bottom margin areas for a vertical ruler).

TickColor is used to draw ticks and tab stops.

RulerColorPageEnd is used to paint the area to the right of the editor width (or below the editor height for a vertical ruler); only if SkinType⁽⁸⁷⁾ = *stNoSkin*.

Color is used to paint outside areas.

Default values:

- RulerColor: *clWindow*;
- IndentColor: *clBtnFace*;
- MarginColor: *clInfoBk*;
- TickColor: *clWindowText*;
- RulerColorPageEnd: *clBtnFace*;

See also:

- IndentSaturation, MarginSaturation, RulerSaturation⁽⁸⁶⁾

1.2.10.1.4 TCustomRuler.IndentSaturation, MarginSaturation, RulerSaturation

Saturation values of the ruler's visual elements in skin modes.

```
property RulerSaturation: Integer;
property IndentSaturation: Integer;
property MarginSaturation: Integer;
```

These properties are applied if SkinType⁽⁸⁷⁾ <> stNoSkin.

RulerSaturation and RulerColor⁽⁸⁵⁾ are applied to an internal ruler area.

IndentSaturation and IndentColor⁽⁸⁵⁾ are applied to handles that are used to change paragraph indents.

MarginSaturation and MarginColor⁽⁸⁵⁾ are applied to left and right margin areas (or top and bottom margin areas for a vertical ruler).

Default value

250

1.2.10.1.5 TCustomRuler.RulerType

Switches between horizontal and vertical ruler.

```
type
  TRulerType = (rtHorizontal, rtVertical);
property RulerType: TRulerType;
```

For TRichView, only a horizontal ruler makes sense.

For ScaleRichView, both horizontal and vertical rulers are useful.

Default value

rtHorizontal

1.2.10.1.6 TCustomRuler.ScreenRes

The number of screen pixels that make up a logical inch.

```
property ScreenRes: Integer;
```

By default, this property is equal to the screen DPI (or DPI of a monitor that contains a form containing this ruler, if the application supports it (in Delphi 10.1+)).

Moreover, the component resets this property to the screen DPI (as described above) after loading from a form and on resizing. So, if you want to change value of this property, you need to reassign it in these cases.

However, it's highly **not recommended to change value of this property**. Allow the ruler to maintain it automatically. Change Zoom⁽⁸⁸⁾ instead.

If you changed a value of the global RichViewPixelsPerInch variable (not recommended!), you need to update this property accordingly:

```
Ruler1.ScreenRes := RichViewPixelsPerInch;
```

This value affects not only sizes that this ruler measures, but also sizes of its visual elements (such as tab icons and thumbs), they are scaled from UnitsPixelsPerInch⁽⁸⁷⁾ to **ScreenRes**.




1.2.10.1.7 TCustomRuler.SkinType

Changes the ruler appearance.

type

```
TSkinType = (stNoSkin, stSkin1, stSkin2);
```

property SkinType: TSkinType;

Value	Appearance
<i>stNoSkin</i>	
<i>stSkin1</i>	
<i>stSkin2</i>	

Default value

stNoSkin

See also

- RulerSaturation, IndentSaturation, MarginSaturation⁽⁸⁶⁾
- Flat⁽⁸⁵⁾

1.2.10.1.8 TCustomRuler.UnitsDisplay, UnitsProgram

Measure units.

property UnitsDisplay: TRulerUnits⁽³⁸⁶⁾;

property UnitsProgram: TRulerUnits⁽³⁸⁶⁾;

UnitsDisplay defines units that the ruler displays. It's recommended to assign the value corresponding to TRVAControlPanel⁽³⁹⁾.UnitsDisplay⁽⁵⁶⁾.

UnitsProgram is used internally in the ruler (for properties of TRulerFloat type), it rarely need to be changed.

Default value

ruCentimeters

1.2.10.1.9 TCustomRuler.UnitsPixelsPerInch

The number of pixels that make up a logical inch if ScreenRes⁽⁸⁶⁾ = 96 and Zoom⁽⁸⁸⁾ = 100%.

property UnitsPixelsPerInch: Integer;

This value must be equal to Style.UnitsPixelsPerInch of TRichViewEdit that works with this event.

It's highly **not recommended to change value of this property** (as well as RichViewEdit.Style.PixelsPerInch). Change Zoom⁽⁸⁸⁾ instead.

This value affects not only sizes that this ruler measures, but also sizes of its visual elements (such as tab icons and thumbs), they are scaled from **UnitsPixelsPerInch** to ScreenRes⁽⁸⁶⁾.

Default value

96

1.2.10.1.10 TCustomRuler.Zoom

This property must be equal to zooming in the editor that works with this ruler, percent.

type

```
TZoomRange = 1 .. 10000; // 1% - 10000%
```

property Zoom: TZoomRange;

By default, if you did not change any “pixels per inch” properties of TRichViewEdit and this ruler, then a document is not scaled, and **Zoom** = 100 (meaning 100%).

However, if you changed TRichViewEdit.DocumentPixelsPerInch, you must change **Zoom** accordingly:

```
RVRuler1.Zoom :=  
Round(100*RichViewEdit1.GetRealDocumentPixelsPerInch /  
RVRuler1.ScreenRes(86));
```

An alternative solution would be assigning RVRuler1.ScreenRes⁽⁸⁶⁾ = RichViewEdit1.GetRealDocumentPixelsPerInch. However, it is not recommended, because:

- it scales all the ruler visual elements too;
- ScreenRes is reset when the ruler is loaded from DFM and when it is resized.

Default value

100

1.2.10.1.11 TRVRuler.RichViewEdit

Specifies the editor for this ruler.

property RichViewEdit: TCustomRichViewEdit;

The ruler works differently depending on RichViewEdit.Options.

If *rvoClientTextWidth* is included in **RichViewEdit.Options**, the ruler assumes that a document width is equal to **RichViewEdit.ClientWidth**.

If *rvoClientTextWidth* is excluded from **RichViewEdit.Options**, when the user resizes a margin, the ruler assigns RichViewEdit.MinTextWidth and MaxTextWidth according to its PageWidth property.

1.2.10.2 Methods

In TRVRuler

UpdateRulerMargins⁽⁸⁹⁾

1.2.10.2.1 TRVRuler.UpdateRulerMargins

Change the ruler margins according to RichViewEdit⁽⁸⁸⁾'s margins.

procedure UpdateRulerMargins;

If RulerType⁽⁸⁶⁾ = *rtHorizontal*, this procedure sets the ruler's LeftMargin, RightMargin, PageWidth according to RichViewEdit⁽⁸⁸⁾.

If RulerType⁽⁸⁶⁾ = *rtVertical*, this procedure sets the ruler's TopMargin, BottomMargin, PageHeight according to RichViewEdit⁽⁸⁸⁾.

If you use TRVRuler without RichViewActions, call this method:

- when the ruler is associated with the editor;
- any time when you call RichViewEdit⁽⁸⁸⁾.Format or Reformat.

If you use TRVRuler with RichViewActions, call this method in TRVAControlPanel⁽³⁹⁾.OnMarginsChanged⁽⁶⁷⁾ event.

If RichViewEdit⁽⁸⁸⁾ is TDBRichViewEdit, also call this method in DataSet.AfterScroll event

1.2.11 TRVStyleTemplateComboBox, TRVStyleTemplateListBox

TRVStyleTemplateComboBox is a combo-box for choosing a style template.

TRVStyleTemplateListBox is a list-box for choosing a style template.

Unit RVStyleFuncs;

Syntax, if TNT Controls⁽³⁷²⁾ are not used:

TRVStyleTemplateComboBox = **class** (TCustomComboBox)

TRVStyleTemplateListBox = **class** (TCustomListBox)

Syntax, if TNT Controls⁽³⁷²⁾ are used:

TRVStyleTemplateComboBox = **class** (TTntCustomComboBox)

TRVStyleTemplateListBox = **class** (TTntCustomListBox)

Hierarchy

(Hierarchy of TRVStyleTemplateComboBox/TRVStyleTemplateListBox, if TNT Controls⁽³⁷²⁾ are not used:)

TObject

TPersistent

TComponent

TControl

TWinControl

TCustomComboBox

TCustomComboBox/TCustomListBox

Description

These component allow applying style templates to Editor⁽⁹¹⁾. The components require Editor⁽⁹¹⁾.UseStyleTemplates=*True*, otherwise their Items are empty.

The components work automatically: place them on a form, assign Editor⁽³⁵⁾ property; no additional code is required, unless you want to process "All Styles" command in OnClickAllStyles⁽⁹⁴⁾ event.

By default, the components show only style templates in use, "quick access" style templates, and heading⁽⁹²⁾ style templates. You can display all style templates by assigning ShowAllStyles⁽⁹²⁾=True.

If ShowClearFormat⁽⁹²⁾=True, the components display "Clear Format" command.

You can display icons in items by assigning Images⁽⁹²⁾ and image indices⁽⁹¹⁾.

When the application language is changed, call Localize⁽⁹³⁾.

1.2.11.1 Properties

In TRVStyleTemplateCombo/ListBox ---

- AllStylesImageIndex⁽⁹¹⁾
- ClearFormatImageIndex⁽⁹¹⁾
- ControlPanel⁽⁹¹⁾
- Editor⁽⁹¹⁾
- Images⁽⁹²⁾
- ParaStyleImageIndex⁽⁹¹⁾
- ParaTextStyleImageIndex⁽⁹¹⁾
- ShowAllStyles⁽⁹²⁾
- ShowClearFormat⁽⁹²⁾
- SmartHeadings⁽⁹²⁾
- TextStyleImageIndex⁽⁹¹⁾

Derived from TCustomCombo/ListBox ---

- Anchors
- BiDiMode
- Color
- Constraints
- Ctl3D
- DragCursor
- DragKind
- DragMode
- DropDownCount
- Enabled
- Font
- ImeMode
- ImeName
- ItemHeight
- Items
- ParentBiDiMode
- ParentColor
- ParentCtl3D
- ParentFont
- ParentShowHint
- PopupMenu
- ShowHint
- Sorted

- TabOrder
- TabStop
- Visible

1.2.11.1.1 TRVStyleTemplateComboBox.*ImageIndex

The properties contain indexes in Images⁽⁹²⁾.

```
property AllStylesImageIndex: Integer;
property ClearFormatImageIndex: Integer;
property TextStyleImageIndex: Integer;
property ParaStyleImageIndex: Integer;
property ParaTextStyleImageIndex: Integer;
```

ClearFormatImageIndex – index of an image for "Clear Format" item, shown if ShowClearFormat⁽⁹²⁾=*True*.

AllStylesImageIndex – index of an image for "All Styles" item, shown if ShowAllStyles⁽⁹²⁾=*False*, and not all styles are displayed.

TextStyleImageIndex – index of an image for style templates having Kind=*rvstkText*

ParaStyleImageIndex – index of an image for style templates having Kind=*rvstkPara*

ParaTextStyleImageIndex – index of an image for style templates having Kind=*rvstkParaText*

Default value:

-1

1.2.11.1.2 TRVStyleTemplateComboBox.ControlPanel

Links this component with a control panel.

```
property ControlPanel: TComponent;
```

You can assign a TRVAControlPanel⁽³⁹⁾ component to this property. If this property is not assigned (*nil*), the component uses MainRVAControlPanel⁽³⁹⁰⁾.

ControlPanel is used to display localized captions according to its Language⁽⁵³⁾.

1.2.11.1.3 TRVStyleTemplateComboBox.Editor

Links this combo-box with an editor.

```
property Editor: TCustomRVControl;
```

Controls of the following types can be assigned to this property:

- TRichViewEdit
- TDBRichViewEdit
- TSRichViewEdit (ScaleRichView)
- TDBSRichViewEdit (ScaleRichView)

When the combo-box is linked with **Editor**:

- the combo-box is updated automatically according to changes in **Editor**
- the combo-box applies the user choice to the selected fragment in **Editor**

1.2.11.1.4 TRVStyleTemplateComboBox.Images

A link to TImageList component containing images for items.

property Images: TCustomImageList;

See also:

- AllStylesImageIndex, ClearFormatImageIndex, ParaStyleImageIndex, ParaTextStyleImageIndex, TextStyleImageIndex⁽⁹¹⁾

1.2.11.1.5 TRVStyleTemplateComboBox.ShowAllStyles

Specifies whether the component displays all available style templates.

property ShowAllStyles: Boolean;

If *True*, the component displays all style templates from Editor⁽⁹¹⁾.Style.StyleTemplates.

If *False*, the component displays:

- style templates in use (i.e. style templates referred by Editor⁽⁹¹⁾.Style.TextStyles and ParaStyles);
- style templates having QuickAccess=*True*
- "All styles" item (if not all style templates are displayed)

If SmartHeadings⁽⁹²⁾=*True*, the component uses a special procedure to calculate a visibility of "heading 1".."heading 9" style templates.

When "All Styles" item is clicked:

- if OnClickAllStyles⁽⁹⁴⁾ is assigned, it is called
- otherwise, the combo-/list-box temporarily displays all style templates.

Default value:

False

See also:

- AllStylesImageIndex⁽⁹¹⁾

1.2.11.1.6 TRVStyleTemplateComboBox.ShowClearFormat

Shows/hides "Clear Format" item.

property ShowClearFormat: Boolean;

If *True*, "Clear Format" is added as the first item.

Default value:

True

See also:

- ClearFormatImageIndex⁽⁹¹⁾

1.2.11.1.7 TRVStyleTemplateComboBox.SmartHeadings

Specifies whether a special procedure should be used to calculate a visibility of heading style templates.

property SmartHeadings: Boolean;

If ShowAllStyles⁽⁹²⁾=*False* and **SmartHeadings**=*True*, the component uses the following procedure to display heading style templates ("heading 1".."heading 9"):

- if headings are not used, or only "heading 1" is used, the component displays headings 1, 2, and 3
- otherwise, if headings up to level N are used (inclusive), the component displays headings from 1 to N+1

Default value:

True

1.2.11.2 Methods

In TRVStyleTemplateCombo/ListBox

- Localize⁹³

Derived from TCustomCombo/ListBox

...

1.2.11.2.1 TRVStyleTemplateComboBox.Localize

Rebuilds the content of the component.

procedure Localize;

Call this method when Language⁵³ of ControlPanel⁹¹ is changed.

1.2.11.3 Events

In TRVStyleTemplateCombo/ListBox

- OnClickAllStyles⁹⁴

Derived from TCustomCombo/ListBox

- OnChange
- OnClick
- OnContextPopup
- OnDbClick
- OnDragDrop
- OnDragOver
- OnDropDown (combo-box only)
- OnEndDock
- OnEndDrag
- OnEnter
- OnExit
- OnKeyDown
- OnKeyPress
- OnKeyUp
- OnStartDock
- OnStartDrag

1.2.11.3.1 TRVStyleTemplateComboBox.OnClickAllStyles

Occurs when "All Styles" item is clicked.

property OnClickAllStyles: TNotifyEvent;

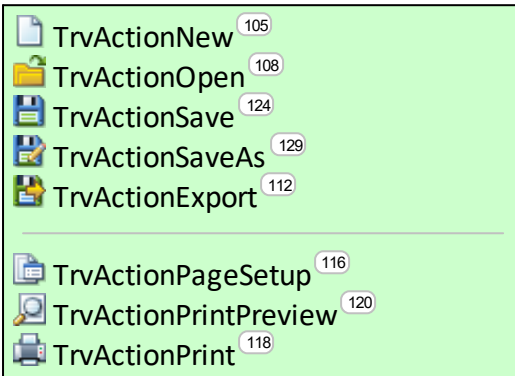
See ShowAllStyles⁽⁹²⁾ for details.

1.3 Actions

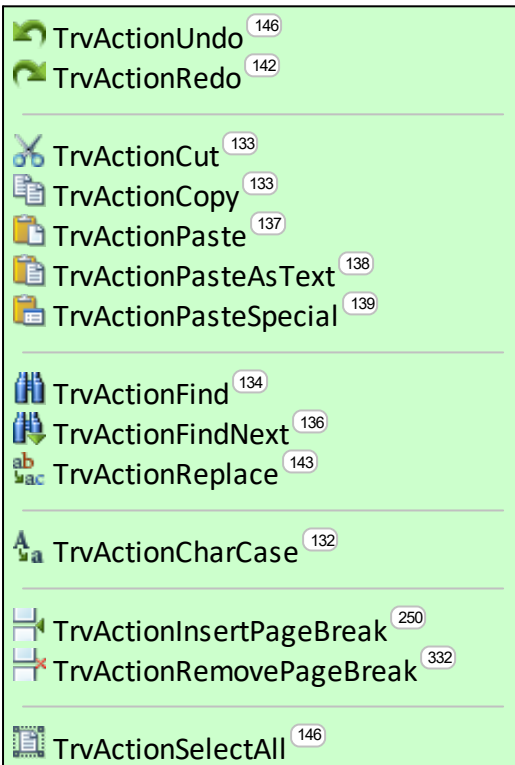
Actions

The actions in the order as they appear in the menu of the ActionTest demo:

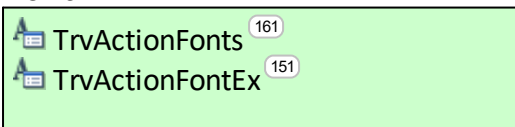
File



Edit





Font






B TrvActionFontBold ¹⁴⁹
I TrvActionFontItalic ¹⁶⁰
U TrvActionFontUnderline ¹⁷⁰
~~S~~ TrvActionFontStrikeout ¹⁶⁸

AB TrvActionFontAllCaps ¹⁴⁸
^⓪ TrvActionFontOverline ¹⁶¹



_f TrvActionSubscript ¹⁷¹
^f TrvActionSuperscript ¹⁷²






 TrvActionFontShrink ¹⁶³
 TrvActionFontGrow ¹⁵⁷




 TrvActionFontShrinkOnePoint ¹⁶⁶
 TrvActionFontGrowOnePoint ¹⁵⁹


 TrvActionFontColor ¹⁵⁰
 TrvActionFontBackColor ¹⁴⁹



Paragraph




 TrvActionParagraph ²⁰⁰
 TrvActionParaBorder ¹⁹³



 TrvActionAlignLeft ¹⁸⁰
 TrvActionAlignCenter ¹⁷⁶
 TrvActionAlignRight ¹⁸¹
 TrvActionAlignJustify ¹⁷⁹
 TrvActionAlignDistribute ¹⁷⁸



 TrvActionParaList ²⁰⁸
 TrvActionParaBullets ¹⁹⁷
 TrvActionParaNumbering ²¹¹


 TrvActionWordWrap ²¹³

 TrvActionIndentInc ¹⁸⁹
 TrvActionIndentDec ¹⁸⁸






 TrvActionLineSpacing100 ¹⁹¹
 TrvActionLineSpacing150 ¹⁹¹
 TrvActionLineSpacing200 ¹⁹²



 TrvActionClearLeft ¹⁸²
 TrvActionClearRight ¹⁸³


-  TrvActionClearBoth ¹⁸²
-  TrvActionClearNone ¹⁸³



-  TrvActionParaColor ¹⁹⁷




Format

-  TrvActionStyleTemplates ²¹⁴
-  TrvActionAddStyleTemplate ²¹⁷
-  TrvActionClearFormat ²²¹
-  TrvActionClearTextFormat ²²¹
-  TrvActionStyleInspector ²¹⁸












-  TrvActionBackground ³¹⁵
-  TrvActionColor ³¹⁷





-  TrvActionFillColor ³²⁴

-  TrvActionHide ³²⁶
-  TrvActionRemoveHyperlinks ³³¹




-  TrvActionItemProperties ³²⁶
-  TrvActionInsertCaption ²²⁵
-  TrvActionVAlign ³³⁵































Insert

-  TrvActionInsertFile ²³⁰
-  TrvActionInsertPicture ²⁵¹
-  TrvActionInsertHLine ²³³
-  TrvActionInsertHyperlink ²³⁴
-  TrvActionBookmarks ²²³
-  TrvActionInsertSymbol ²⁵⁶
-  TrvActionInsertEquation ²²⁸ *
-  TrvActionInsertNumber ²⁴⁷
-  TrvActionInsertTextBox ²⁷¹
-  TrvActionInsertPageNumber ²⁵¹
-  TrvActionInsertPageCount ²⁵⁰



-  TrvActionInsertFootnote ²⁶⁶
-  TrvActionInsertEndnote ²⁶⁵
-  TrvActionInsertSidenote ²⁶⁹
-  TrvActionEditNote ²⁶²

Table

-  TrvActionInsertTable ²⁷³
-  TrvActionTableInsertColLeft ²⁹⁹
-  TrvActionTableInsertColRight ²⁹⁹

	TrvActionTableInsertRowsAbove	300
	TrvActionTableInsertRowsBelow	301
<hr/>		
	TrvActionTableDeleteCols	297
	TrvActionTableDeleteRows	297
	TrvActionTableDeleteTable	298
	TrvActionTableToText	311
<hr/>		
	TrvActionTableSelectTable	308
	TrvActionTableSelectCols	307
	TrvActionTableSelectRows	308
	TrvActionTableSelectCell	307
<hr/>		
	TrvActionTableCellVAlignTop	296
	TrvActionTableCellVAlignMiddle	295
	TrvActionTableCellVAlignBottom	294
	TrvActionTableCellVAlignDefault	295
<hr/>		
	TrvActionTableCellRotationNone	290
	TrvActionTableCellRotation90	291
	TrvActionTableCellRotation180	291
	TrvActionTableCellRotation270	292
<hr/>		
	TrvActionTableCellLeftBorder	287
	TrvActionTableCellTopBorder	293
	TrvActionTableCellRightBorder	289
	TrvActionTableCellBottomBorder	286
	TrvActionTableCellAllBorders	285
	TrvActionTableCellNoBorders	288
<hr/>		
	TrvActionTableSplit	310
	TrvActionTableSplitCells	310
	TrvActionTableMergeCells	302
<hr/>		
	TrvActionTableGrid	298
<hr/>		
	TrvActionTableSort	309
	TrvActionTableProperties	303

Toolbar only

	TrvActionQuickPrint	122
	TrvActionShowSpecialCharacters	333

Not used in ActionTest

TrvActionTextLTR	173
TrvActionTextRTL	174
<hr/>	
TrvActionParaLTR	210
TrvActionParaRTL	212
<hr/>	
TrvActionSpellingCheck	312
TrvActionThesaurus	312

Customized

TrvActionEvent	323
TrvActionInsertText	258
TrvActionUserDefinedColor	

* the action is not included in RichViewActions directly, it is included in a separate package.

1.3.1 File

File Actions

This group of actions creates a new document, saves, opens, prints it.

TrvActionNew	105
TrvActionOpen	108
TrvActionSave	124
TrvActionSaveAs	129
TrvActionExport	112
<hr/>	
TrvActionPageSetup	116
TrvActionPrintPreview	120
TrvActionPrint	118

1.3.1.1 TrvActionCustomFileIO

TrvActionCustomFileIO is a base class for "Save As", "Export" and "Insert File" actions.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionCustomFileIO = class (TrvActionCustomIO100)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionCustomIO* ⁽¹⁰⁰⁾**Description**

This action is not used directly.

Descendants:

- *TrvActionSaveAs* ⁽¹²⁹⁾
- *TrvActionExport* ⁽¹¹²⁾
- *TrvActionInsertFile* ⁽²³⁰⁾

1.3.1.1.1 Properties**In *TrvActionCustomFileIO***

■ *CustomFilter* ⁽¹⁰⁰⁾

Derived from *TrvActionCustomIO* ⁽¹⁰⁰⁾

■ *AutoUpdateFileName* ⁽¹⁰¹⁾

■ *DialogTitle* ⁽¹⁰²⁾

■ *FileName* ⁽¹⁰²⁾

■ *InitialDir* ⁽¹⁰²⁾

Derived from *TrvAction* ⁽³¹³⁾

■ *Control* ⁽³¹⁴⁾

Derived from *TrvCustomAction* ⁽³³⁵⁾

■ *Caption* ⁽³³⁶⁾

■ *ControlPanel* ⁽³³⁷⁾

■ *Disabled* ⁽³³⁷⁾

■ *Hint* ⁽³³⁷⁾

Derived from *TAction*

■ *AutoCheck*

■ *Caption*

■ *Checked*
 DisableIfNoHandler

■ *Enabled*

■ *GroupIndex*

■ *HelpContext*

■ *HelpKeyword*

■ *HelpType*

■ *Hint*

■ *ImageIndex*

■ *Name*

■ *SecondaryShortCuts*

■ *ShortCut*

■ Visible

1.3.1.1.1 TrvActionCustomFileIO.CustomFilter

Specifies the file filter for custom file formats.

property CustomFilter: TRVALocString⁽³⁵⁰⁾;

This property allows opening, saving and inserting files in custom formats (in addition to the standard formats provided by RichViewActions).

Value of this property is appended to the Filter property of the file opening/saving dialog, if the action supports custom formats.

Default value:

" (empty string)

See also:

- TRVActionSaveAs.Filter⁽¹³¹⁾
- TRVActionExport.Filter⁽¹¹⁵⁾
- TRVActionInsertFile.Filter⁽²³²⁾
- TrvActionOpen.CustomFilter⁽¹¹⁰⁾
- TRVAControlPanel.OnCustomFileOperation⁽⁶⁴⁾

1.3.1.2 TrvActionCustomIO

TrvActionCustomIO is a base class for the actions displaying dialogs for opening or saving files.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionCustomIO = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action is not used directly.

Direct descendants:

- TrvActionCustomFileIO⁽⁹⁸⁾
- TrvActionInsertPicture⁽²⁵¹⁾

1.3.1.2.1 Properties

In TrvActionCustomIO

- AutoUpdateFileName⁽¹⁰¹⁾
- DialogTitle⁽¹⁰²⁾
- FileName⁽¹⁰²⁾
- InitialDir⁽¹⁰²⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.2.1.1 TrvActionCustomIO.AutoUpdateFileName

Specifies whether FileName⁽¹⁰²⁾ is automatically updated by the action.

property AutoUpdateFileName: Boolean;

If this property is *True*, FileName⁽¹⁰²⁾ property is updated when the action is successfully executed (the user chose a new file from dialog).

This property is used and published in the following actions:

- TrvActionExport⁽¹¹²⁾
- TrvActionInsertFile⁽²³⁰⁾

Default value:

True

1.3.1.2.1.2 TrvActionCustomIO.DialogTitle

Defines a custom title (caption) for the file dialog.

property DialogTitle: TRVLocString⁽³⁵⁰⁾;

If this property is empty, default (localized) dialog titles are used.

Default value:

" (empty string)

See also:

- TrvActionOpen.DialogTitle⁽¹¹⁰⁾

1.3.1.2.1.3 TrvActionCustomIO.FileName

Specifies the default file name.

property FileName: TRVLocString⁽³⁵⁰⁾;

This property is assigned to FileName property of the file open/save dialog before showing it.

This property is used and public in the following actions:

- TrvActionExport⁽¹¹²⁾
- TrvActionInsertFile⁽²³⁰⁾
- TrvActionInsertPicture⁽²⁵¹⁾

Default value:

" (empty string)

See also properties:

- InitialDir⁽¹⁰²⁾
- AutoUpdateFileName⁽¹⁰¹⁾

1.3.1.2.1.4 TrvActionCustomIO.InitialDir

Specifies the initial directory for the file dialog.

property InitialDir: TRVLocString⁽³⁵⁰⁾;

This property is used only if FileName⁽¹⁰²⁾ is empty.

It is assigned to InitialDir property of the file open/save dialog before showing it.

Default value:

" (empty string)

1.3.1.3 TrvActionCustomNew

TrvActionCustomNew is a base class for "New" and "Open" actions.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCustomNew = class (TrvAction (313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action is not used directly.

Descendants:

- TrvActionNew ⁽¹⁰⁵⁾
- TrvActionOpen ⁽¹⁰⁸⁾

1.3.1.3.1 Events

In TrvActionCustomNew

■ OnNew ⁽¹⁰³⁾

1.3.1.3.1.1 TrvActionCustomNew.OnNew

Occurs when the action is executed.

property OnNew: TNotifyEvent;

This event occurs after the document is cleared.

This event is called by TrvActionNew ⁽¹⁰³⁾.

TrvActionOpen ⁽¹⁰⁸⁾ has this event as well, it may be called before the loading occurs. However, it calls this event only if it is not linked (either explicitly ⁽¹¹⁰⁾ or implicitly) with TrvActionNew. If linked, TrvActionOpen calls the event of TrvActionNew instead.

If style templates are not used ⁽²⁰⁾, this is a good place to define default values for collections of styles. For example:

procedure TForm1.rvActionNew1New(Sender: TObject);

begin

RVStyle1.ParaStyles.Clear;

RVStyle1.ParaStyles.Add; // one default paragraph style

RVStyle1.ListStyles.Clear; // no default list styles

```
RVStyle1.TextStyles.Clear;
with RVStyle1.TextStyles.Add do begin // one default text style
    FontName := 'Times New Roman';
    Size      := 12;
end;
end;
```

If style templates are used, styles are reset automatically.

1.3.1.3.2 Properties

In TrvActionCustomNew

■ ActionSave ⁽¹⁰⁴⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

■ AutoCheck

■ Caption

■ Checked

DisableIfNoHandler

■ Enabled

■ GroupIndex

■ HelpContext

■ HelpKeyword

■ HelpType

■ Hint

■ ImageIndex

■ Name

■ SecondaryShortCuts

■ ShortCut

■ Visible

1.3.1.3.2.1 TrvActionCustomNew.ActionSave

Links this action to TrvActionSave ⁽¹²⁴⁾ action.

property ActionSave: TrvActionSave ⁽¹²⁴⁾;

If this link is not defined, the first found TrvActionSave ⁽¹²⁴⁾ action on the same form/datamodule is used. If no TrvActionSave ⁽¹²⁴⁾ action found, an exception occurs on executing.

1.3.1.4 TrvActionNew

TrvActionNew is the action for "File | New" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionNew = class (TrvActionCustomNew(103))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomNew⁽¹⁰³⁾

Description

If you use this action, you must also use TrvActionSave⁽¹²⁴⁾ and TrvActionSaveAs⁽¹²⁹⁾ actions.

This action must be linked to TrvActionSave⁽¹²⁴⁾ action, because TrvActionSave⁽¹²⁴⁾ contains a list of all documents opened in TCustomRichViewEdit controls. This link may be defined explicitly (via ActionSave⁽¹⁰⁴⁾ property) or implicitly (this action finds and uses TrvActionSave⁽¹²⁴⁾ action placed on the same form/datamodule).

This action performs the following tasks, if style templates are not used⁽²⁰⁾:

1. If the document was modified, asks for saving and saves it (using TrvActionSave⁽¹²⁴⁾).
2. Clears the document (calls TCustomRichViewEdit.Clear); clears the background image; if *rvfoSaveDocProperties* is included in TCustomRichViewEdit.RVFOptions, clears its DocProperties and resets DocParameters properties.
3. Applies the default values specified in the properties of GetControlPanel⁽³³⁸⁾: DefaultColor⁽⁴³⁾, DefaultMargin⁽⁴⁵⁾, DefaultDocParameters⁽⁴⁴⁾.
4. Calls OnNew⁽¹⁰³⁾ event.
5. Calls OnMarginsChanged⁽⁶⁷⁾ event of GetControlPanel⁽³³⁸⁾ component.
6. Formats the document (calls TCustomRichViewEdit.Format).
7. If GetControlPanel⁽³³⁸⁾.AutoDeleteUnusedStyles⁽⁴²⁾ = *True*, deletes unused styles (calls TCustomRichViewEdit.DeleteUnusedStyles(True, True, True)).
8. Stores the following values for this TCustomRichViewEdit using the properties of GetControlPanel⁽³³⁸⁾: temporal file name = DefaultFileName⁽⁴⁵⁾, file format = DefaultFileFormat⁽⁴⁴⁾, custom filter index (used if file format is *ffeCustom*) = DefaultCustomFilterIndex⁽⁴³⁾. These values are stored in the list of documents maintained by TrvActionSave⁽¹²⁴⁾.
9. Calls OnDocumentFileChange⁽¹²⁸⁾ event of TrvActionSave⁽¹²⁴⁾ action.

If style templates are used⁽²⁰⁾:

The sequence of operations is the same, but the step #7 is different. The action resets text and paragraph styles according to StyleTemplates⁽¹⁰⁶⁾.

1.3.1.4.1 Properties

In TrvActionNew

- ApplyStyleTemplates ⁽¹⁰⁶⁾
- StyleTemplates ⁽¹⁰⁶⁾

Derived from TrvActionCustomNew ⁽¹⁰³⁾

- ActionSave ⁽¹⁰⁴⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.4.1.1 TrvActionNew.StyleTemplates, ApplyStyleTemplates

A collection of style templates for assigning to new documents.

property StyleTemplates: TRVStyleTemplateCollection;

property ApplyStyleTemplates: Boolean;


These properties are used:

- when TrvActionNew is executed
- when TrvActionOpen ⁽¹⁰⁸⁾ (linked to this action) clears a target editor before loading a file.

Values in **StyleTemplates** are measured in GetControlPanel ⁽³³⁸⁾.UnitsProgram ⁽⁵⁷⁾.

If style templates are used ⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=*True*), and **ApplyStyleTemplates** = *True*, the action:

1. deletes all items in TextStyles, ParaStyles, ListStyles;
2. adds one paragraph style; if StyleTemplates contain "Normal" style template, it is applied to this paragraph style;
3. adds one text style; if StyleTemplates contain "Normal" style template, this text style is formatted accordingly.

 **ScaleRichView note:** for TScaleRichViewEdit, this the action performs similar operations with its header and footer. However, instead of "Normal", the action uses "header" style template for RVHeader, and "footer" style template for RVFooter properties.

Default value:

By default, the property contains the following style templates: "Normal", "heading 1", "heading 2", "heading 3", "Hyperlink", "header", "footer", "footnote reference", "endnote reference", "footnote text", "endnote text", "caption".

"header" and "footer" are not used by TRichViewEdit (used only in ScaleRichView), so you can delete these items.

See also properties of GetControlPanel ⁽³³⁸⁾:

- DefaultColor ⁽⁴³⁾
- DefaultMargin ⁽⁴⁵⁾
- DefaultDocParameters ⁽⁴⁴⁾

See also:

- TrvActionStyleTemplates ⁽²¹⁴⁾.StandardStyleTemplates ⁽²¹⁶⁾

1.3.1.4.2 Methods

In TrvActionNew

Reset ⁽¹⁰⁷⁾

Inherited from TrvCustomAction ⁽³³⁵⁾

GetControlPanel ⁽³³⁸⁾

1.3.1.4.2.1 TrvActionNew.Reset

Resets properties of rve.

```
procedure Reset(rve: TCustomRichViewEdit);
```

This method performs almost the same operation as this action when it is executed: it assigns default values to properties of **rve**; if style templates are used, it applies StyleTemplates ⁽¹⁰⁶⁾.

This method:

- does not clear **rve**;
- does not delete unused styles (if style templates are not used, otherwise all old styles are deleted when StyleTemplates ⁽¹⁰⁶⁾ are applied);
- (the most important) does not associate any file with **rve**.

Normally, you do not need to call this method, because all necessary operations are performed when the action is executed.

However, this method is useful if you work with `TDBRichViewEdit` or `TDBSRichViewEdit`. For db-controls, you cannot use file-based actions (`TrvActionNew`⁽¹⁰⁵⁾, `TrvActionSave`⁽¹²⁴⁾, `TrvActionSaveAs`⁽¹²⁹⁾, `TrvActionOpen`⁽¹⁰⁸⁾) directly. However, you can place `TrvActionNew` of a form, clear its `Shortcut` (to prevent execution on `Ctrl+N`), and call `rvActionNew1.Reset(DBRichViewEdit1)` in `DBRichViewEdit1.OnNewDocument` event.

1.3.1.5 TrvActionOpen

`TrvActionOpen` is the action for "File | Open..." command.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionOpen = class (TrvActionCustomNew(103))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionCustomNew(103)

```

Description

The action displays a file opening dialog and loads the chosen file in `TCustomRichViewEdit` component. After opening, this editor is associated with the chosen file name and format, so subsequent executions of `TrvActionSave`⁽¹²⁴⁾ action save content of this `TCustomRichViewEdit` component in this file in this format.

The following formats are supported:

- RichView Format (RVF)
- Microsoft Word Document (DocX)
- Rich Text Format (RTF)
- Text files (TXT, Unicode and ANSI)
- **Markdown** (MARKDOWN, MDOWN, MKDN, MD, MKD, MDWN, MDTXT, MDTEXT)
- HTML
- RichViewXML Format (XML, only if `RichViewXML`⁽³⁷⁰⁾ is used)
- custom formats, supported in `GetControlPanel`⁽³³⁸⁾.`OnCustomFileOperation`⁽⁶⁴⁾ event.

If you use this action, you must also use `TrvActionSave`⁽¹²⁴⁾ and `TrvActionSaveAs`⁽¹²⁹⁾ actions.

This action must be linked to `TrvActionSave`⁽¹²⁴⁾ action, because `TrvActionSave`⁽¹²⁴⁾ contains a list of all documents opened in `TCustomRichViewEdit` controls. This link may be defined explicitly (via `ActionSave`⁽¹⁰⁴⁾ property) or implicitly (this action finds and uses `TrvActionSave`⁽¹²⁴⁾ action placed on the same form/datamodule).

This action may be linked to TrvActionNew⁽¹⁰⁵⁾ action. This link may be defined explicitly (via ActionNew⁽¹¹⁰⁾ property) or implicitly (this action finds and uses TrvActionNew⁽¹⁰⁵⁾ action placed on the same form/datamodule). If linked, TrvActionNew⁽¹⁰⁵⁾ is used to reset editor before loading (it is especially important, if style templates are used⁽²⁰⁾).

This action performs the following tasks:

1. If the document was modified, asks for saving and saves it (using TrvActionSave⁽¹²⁴⁾).
2. Shows a file opening dialog allowing the user to choose the file for opening.
3. If the user chose a file, calls LoadFile⁽¹¹¹⁾ method. The file format is determined not by the file extension but by the filter index chosen in the file opening dialog.

1.3.1.5.1 Properties

In TrvActionOpen

- ActionNew⁽¹¹⁰⁾
- CustomFilter⁽¹¹⁰⁾
- DialogTitle⁽¹¹⁰⁾
- Filter⁽¹¹⁰⁾
- InitialDir⁽¹¹¹⁾
- LastFilterIndex⁽¹¹¹⁾

Derived from TrvActionNew⁽¹⁰³⁾

- ActionSave⁽¹⁰⁴⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts

- ShortCut
- Visible

1.3.1.5.1.1 TrvActionOpen.ActionNew

Links this action to TrvActionNew⁽¹⁰³⁾ action.

property ActionNew: TrvActionNew⁽¹⁰³⁾;

If this link is not defined, the first found TrvActionNew⁽¹⁰³⁾ action on the same form/datamodule is used.

1.3.1.5.1.2 TrvActionOpen.CustomFilter

Specifies the file filter for custom file formats.

property CustomFilter: TRVALocString⁽³⁵⁰⁾;

This property allows opening files in custom formats (in addition to the standard formats listed in the Filter⁽¹¹⁰⁾ property of this action).

Value of this property is appended to the Filter property of the file opening dialog, if the Filter⁽¹¹⁰⁾ property includes *ffiCustom*.

Default value:

" (empty string)

See also:

- GetControlPanel⁽³³⁸⁾.OnCustomFileOperation⁽⁶⁴⁾
- TrvActionCustomFileIO.CustomFilter⁽¹⁰⁰⁾

1.3.1.5.1.3 TrvActionOpen.DialogTitle

Defines a custom title (caption) for the file opening dialog.

property DialogTitle: TRVALocString⁽³⁵⁰⁾;

If this property is empty, default (localized) dialog title is used.

Default value:

" (empty string)

See also:

- TrvActionCustomIO.DialogTitle⁽¹⁰²⁾

1.3.1.5.1.4 TrvActionOpen.Filter

Defines a set of file formats appearing in the file opening dialog.

property Filter: TrvFileOpenFilterSet⁽³⁸⁹⁾;

If *ffiCustom* is included, formats listed in the CustomFilter⁽¹¹⁰⁾ property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel⁽³³⁸⁾.RVFilter⁽⁵⁴⁾.

Default value:

all formats

1.3.1.5.1.5 TrvActionOpen.InitialDir

Specifies the initial directory for the file dialog.

property InitialDir: TRVALocString⁽³⁵⁰⁾;

Value of this property is assigned to InitialDir property of the file opening dialog before showing it.

Default value:

" (empty string)

1.3.1.5.1.6 TrvActionOpen.LastFilterIndex

Defines file format selected in the file opening dialog by default.

property LastFilterIndex: Integer;

Value of this property is assigned to FilterIndex property of the file opening dialog before showing it.

Default value:

1

1.3.1.5.2 Methods

In TrvActionOpen

LoadFile⁽¹¹¹⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.1.5.2.1 TrvActionOpen.LoadFile

Loads the specified file in **rve**.

```
procedure LoadFile(rve: TCustomRichViewEdit;
  const FileName: TRVUnicodeString;
  FileFormat: TrvFileOpenFilter(389); CustomFilterIndex: Integer=0);
```

Parameters:

rve – editor where to load file.

FileName – name of the file to load.

FileFormat – format of this file.

CustomFilterIndex used only if **FileFormat** is *ffiCustom*.

If **FileFormat** = *ffiCustom*, **CustomFilterIndex** identifies a custom file format. Custom formats are numbered from 1 in the order they are listed in the CustomFilter⁽¹¹⁰⁾ property.

This procedure performs the following tasks:

1. Clears the document (calls TCustomRichViewEdit.Clear); clears the background image; if *rvfoSaveDocProperties* is included in TCustomRichViewEdit.RVFOptions, clears its DocProperties and resets DocParameters properties*

2. Applies the default values specified in the properties of GetControlPanel⁽³³⁸⁾: DefaultColor⁽⁴³⁾, DefaultMargin*⁽⁴⁵⁾
 3. Calls OnNew⁽¹⁰³⁾ event*
 4. If FileFormat = *ffiTextANSI*, and the linked⁽³³⁷⁾ ControlPanel's UseTextCodePageDialog⁽⁵⁹⁾ = *True*, shows a dialog for choosing the text code page.
 5. Loads the file in **rve**.
 6. Associates **rve** with this file and file format (in the list of documents maintained by TrvActionSave⁽¹²⁴⁾).
 7. Calls OnDocumentFileChange⁽¹²⁸⁾ event of the linked (implicitly or explicitly⁽¹⁰⁴⁾) TrvActionSave⁽¹²⁴⁾ action.
 8. Calls OnMarginsChanged⁽⁶⁷⁾ event of GetControlPanel⁽³³⁸⁾ component.
 9. Formats the document (calls TCustomRichViewEdit.Format).
 10. If GetControlPanel⁽³³⁸⁾.AutoDeleteUnusedStyles⁽⁴²⁾ = *True*, deletes unused styles (calls TCustomRichViewEdit.DeleteUnusedStyles(True, True, True)).
 11. Calls OnOpenFile⁽¹¹²⁾ event.
- Subsequent executions of TrvActionSave⁽¹²⁴⁾ will save **rve** content in this file in this format.

* If TrvActionNew⁽¹⁰⁵⁾ is linked (either implicitly or explicitly⁽¹¹⁰⁾), its functions for document clearing and resetting are used; they do the same, but also apply TrvActionNew.StyleTemplates⁽¹⁰⁶⁾.

1.3.1.5.3 Events

In TrvActionOpen

■ OnOpenFile⁽¹¹²⁾

Derived from TrvActionNew⁽¹⁰³⁾

■ OnNew⁽¹⁰³⁾

1.3.1.5.3.1 TrvActionOpen.OnOpenFile

Occurs when the action opened a file.

type

```
TRVOpenFileEvent = procedure (Sender: TObject;
    Editor: TCustomRichViewEdit;
    const FileName: TRVUnicodeString;
    FileFormat: TrvFileOpenFilter(389);
    CustomFilterIndex: Integer) of object;
```

property OnOpenFile: TRVOpenFileEvent;

Parameters are the same as in LoadFile⁽¹¹¹⁾ method.

1.3.1.6 TrvActionExport

TrvActionExport is the action for "File | Export..." command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionExport = class (TrvActionCustomFileIO98)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³
*TrvActionCustomIO*¹⁰⁰
*TrvActionCustomFileIO*⁹⁸

Description

This action displays a file saving dialog and saves the content of *TRichViewEdit* component in the chosen file.

In addition to formats supported by *TrvActionSaveAs*¹²⁹ action, this action supports:

- simplified HTML (without CSS)
- formats supported by Microsoft Office file export converters.

After the execution, this *TCustomRichViewEdit* component is still associated with the file name and format assigned by the last execution of *TrvActionNew*¹⁰³, *TrvActionOpen*¹⁰⁸ or *TrvActionSaveAs*¹²⁹ action. *TrvActionExport* does not change this association.

This action performs the following tasks:

1. Shows a file saving dialog allowing the user to choose the file name and format for exporting.
2. If the user chose a file, calls *ExportToFile*¹¹⁶ method. The file format is determined not by the file extension but by the filter index chosen in the file saving dialog.

1.3.1.6.1 Properties

In *TrvActionExport*

- *CreateDirectoryForHTMLImages*¹¹⁵
- *Filter*¹¹⁵

Derived from *TrvActionCustomFileIO*⁹⁸

- *CustomFilter*¹⁰⁰

Derived from *TrvActionCustomIO*¹⁰⁰

- *AutoUpdateFileName*¹⁰¹
- *DialogTitle*¹⁰²
- *FileName*¹⁰²
- *InitialDir*¹⁰²

Derived from *TrvAction*³¹³

- *Control*³¹⁴

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible CustomFilter ¹⁰⁰

Derived from TrvActionCustomIO ¹⁰⁰

- AutoUpdateFileName ¹⁰¹
- DialogTitle ¹⁰²
- FileName ¹⁰²
- InitialDir ¹⁰²

Derived from TrvAction ³¹³

- Control ³¹⁴

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword

- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.6.1.1 TrvActionExport.CreateDirectoryForHTMLImages

Specifies whether a subdirectory for saving images is created when exporting HTML or Markdown file.

property `CreateDirectoryForHTMLImages: Boolean;`

This property is used when exporting to HTML and Markdown formats (*ffeHTMLCSS*, *ffeHTML*, *ffeMarkdown*⁽³⁸⁸⁾).

If *True*, images for HTML/Markdown files are saved in a subdirectory having the same name as HTML/Markdown file but '.files' extension.

If *False*, all images are saved in the same directory as the HTML/Markdown file.

This option is used for images with undefined (empty) file names, or when *rvhtmlsioUseItemImageFileNames* is not included in `HTMLSaveProperties.ImageOptions` of the target editor.

Default value:

True

See also:

- `TrvActionSave.CreateDirectoryForHTMLImages`⁽¹²⁶⁾

1.3.1.6.1.2 TrvActionExport.Filter

Defines a set of file formats appearing in the file saving dialog.

property `Filter: TrvFileExportFilterSet`⁽³⁸⁸⁾;

If *ffeCustom* is included, formats listed in the `CustomFilter`⁽¹⁰⁰⁾ property are used.

Name and extension of RichView Format (RVF) may be customized, see `GetControlPanel`⁽³³⁸⁾.
`.RVFilter`⁽⁵⁴⁾.

Default value:

all formats

1.3.1.6.2 Methods

In TrvActionExport

`ExportToFile`⁽¹¹⁶⁾

Inherited from TrvCustomAction⁽³³⁵⁾

`GetControlPanel`⁽³³⁸⁾

1.3.1.6.2.1 TrvActionExport.ExportToFile

Saves **Edit** in the specified file in the specified format.

```
function ExportToFile(Edit: TCustomRichViewEdit;
  const FileName: TRVUnicodeString;
  ExportFormat: TrvFileExportFilter388;
  ConverterOrCustomIndex: Integer;
  rvc: TRVOfficeConverter): Boolean;
```

Parameters:

Edit – editor for saving.

FileName – file name for saving.

FileFormat – file format for saving.

CustomFilterIndex used only if **FileFormat** is *ffeCustom* or *ffeOfficeConverters*.

If **FileFormat** = *ffeCustom*, **CustomFilterIndex** identifies a custom file format. Custom formats are numbered from 1 in the order they are listed in the CustomFilter¹⁰⁰ property.

If **FileFormat** = *ffeOfficeConverters*, **CustomFilterIndex** identifies a format supported by Microsoft Office file export converters. Formats are numbered from 0. A list of these formats and their order are different on different computers. This is the index in **rvc.ExportConverters** collection.

If saving is unsuccessful, an error message is displayed.

Return value:

True if the saving was successful.

1.3.1.7 TrvActionPageSetup

TrvActionPageSetup is the action for "File | Page Setup" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionPageSetup = class (TrvCustomAction335)
```

Hierarchy

```
TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
```

Description

This action displays a page setup dialog.

This action does not work with the specific editor. It defines global page properties for the application.

This action is enabled in `GetControlPanel338.RVPrint55` is assigned, and there is at least one printer is installed in the system (`Printer.Printers.Count>0`).

The settings are stored:

- margins are stored in `GetControlPanel338.RVPrint55` properties;
- headers and footers are stored in `GetControlPanel338.Header and Footer49` properties;
- other properties (paper size, orientation, source) are stored in the application's printing settings (in the object returned by `Printer` function).

1.3.1.7.1 Properties

In TrvActionPageSetup

- `MarginsUnits117`
- `UnitsPixelsPerInch118`

Derived from TrvCustomAction³³⁵

- `Caption336`
- `ControlPanel337`
- `Disabled337`
- `Hint337`

Derived from TAction

- `AutoCheck`
- `Caption`
- `Checked`
- `DisableIfNoHandler`
- `Enabled`
- `GroupIndex`
- `HelpContext`
- `HelpKeyword`
- `HelpType`
- `Hint`
- `ImageIndex`
- `Name`
- `SecondaryShortCuts`
- `ShortCut`
- `Visible`

1.3.1.7.1.1 TrvActionPageSetup.MarginsUnits

Defines measuring units used for margins in the page setup dialog .

type

```
TrvPaperMarginsUnits = (rvpmuMillimeters, rvpmuInches);
```

property `MarginsUnits`: `TrvPaperMarginsUnits`;

Default value:

rvpmuMillimeters

1.3.1.7.1.2 TrvActionPageSetup.UnitsPixelsPerInch

Specify a count of pixels in one logical inch.

property UnitsPixelsPerInch: Integer;

This value is used when converting RVPrint⁽⁵⁵⁾.Margins to MarginUnits⁽¹¹⁷⁾ when displaying a page setup dialog, and back to RVPrint⁽⁵⁵⁾.Units when applying changes.

This property must be equal to Style.UnitsPixelsPerInch property of target editors.

Default value:

96

1.3.1.7.2 Events

In TrvActionPageSetup

■ OnChange⁽¹¹⁸⁾

1.3.1.7.2.1 TrvActionPageSetup.OnChange

Occurs after the changes made in the page setup dialog are applied.

property OnChange: TNotifyEvent;

1.3.1.8 TrvActionPrint

TrvActionPrint is the action for "File | Print..." command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionPrint = **class** (TrvCustomPrintAction⁽¹³¹⁾)

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction
 TContainedAction
 TCustomAction
 TAction
 TrvCustomAction⁽³³⁵⁾
 TrvAction⁽³¹³⁾
 TrvCustomPrintAction⁽¹³¹⁾

Description

This action displays a printing dialog (TPrintDialog) and prints the document from TCustomRichViewEdit component.

This action uses settings defined in TrvActionPageSetup⁽¹¹⁶⁾. If GetControlPanel⁽³³⁸⁾.RVPrint⁽⁵⁵⁾ is not assigned, the action uses the first TRVPrint component found on the form owning the

TCustomRichViewEdit component. If not found, it creates a temporal TRVPrint with default property settings.

The action can print the whole document, the range of pages, or the selected fragment, depending on the user's choice.

This action is enabled only if at least one printer is installed in the system (Printer.Printers.Count>0).

See also:

- TrvActionQuickPrint¹²²
- TrvActionPrintPreview¹²⁰

1.3.1.8.1 Properties

In TrvActionPrint

- Title¹²⁰

Derived from TrvAction³¹³

- Control³¹⁴

Derived from TrvCustomAction³³⁵

- Caption³³⁶
- ControlPanel³³⁷
- Disabled³³⁷
- Hint³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.8.1.1 TrvActionPrint.Title

Determines the text that is listed in the Print Manager.

property Title: TRVUnicodeString;

See also:

TrvActionQuickPrint.Title⁽¹²⁴⁾

1.3.1.9 TrvActionPrintPreview

TrvActionPrintPreview is the action for "File | Print Preview" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionPrintPreview = **class** (TrvCustomPrintAction⁽¹³¹⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvCustomPrintAction⁽¹³¹⁾

Description

This action displays a printing preview window. This window has "Page Setup" and "Print" buttons where the user can print the document. For the "Page Setup" button, TrvActionPageSetup⁽¹¹⁶⁾ action is used. If it is not linked (via ActionPageSetup⁽¹²¹⁾ property), a temporal TrvActionPageSetup⁽¹¹⁶⁾ action is used. For the "Print" button, TrvActionPrint⁽¹¹⁸⁾ action is used. If it is not linked (via ActionPrint⁽¹²¹⁾ property), the first TrvActionPrint⁽¹¹⁸⁾ action on the same form/datamodule is used; if not found, printing in this action is not possible.

This action uses settings defined in TrvActionPageSetup⁽¹¹⁶⁾. If GetControlPanel⁽³³⁸⁾.RVPrint⁽⁵⁵⁾ is not assigned, the action uses the first TRVPrint component found on the form owning the TCustomRichViewEdit component. If not found, it creates a temporal TRVPrint with default property settings.

This action is enabled only if at least one printer is installed in the system (Printer.Printers.Count>0).

See also:

- TrvActionQuickPrint⁽¹²²⁾
- TrvActionPrint⁽¹¹⁸⁾

1.3.1.9.1 Properties

In TrvActionPrintPreview

- ActionPageSetup⁽¹²¹⁾
- ActionPrint⁽¹²¹⁾
- Maximized⁽¹²²⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.9.1.1 TrvActionPrintPreview.ActionPageSetup

A link to TrvActionPageSetup⁽¹¹⁶⁾ action.

property ActionPageSetup: TrvActionPageSetup⁽¹¹⁶⁾;

This link is used for the "Page Setup" button on the print preview dialog.

If this link is not assigned, a temporal action is used.

1.3.1.9.1.2 TrvActionPrintPreview.ActionPrint

A link to TrvActionPrint⁽¹¹⁸⁾ action.

property ActionPrint: TrvActionPrint⁽¹¹⁸⁾;

This link is used for the "Print" button on the print preview dialog.

If this link is not assigned, the first found TrvActionPrint⁽¹¹⁸⁾ action on the same form is used. If not found, printing is not possible.

1.3.1.9.1.3 TrvActionPrintPreview.Maximized

Defines the initial state of the print preview window.

property `Maximized: Boolean;`

Assign *True* to this property to display the print preview window maximized.

Default value:

False

1.3.1.9.2 Events

In TrvActionPrintPreview

■ OnGetPreviewFormClass ⁽¹²²⁾

1.3.1.9.2.1 TrvActionPrintPreview.OnGetPreviewFormClass

Allows defining your own window for a print preview.

type

```
TRVGetFormClassEvent = procedure (Sender: TObject;  
    var FormClass: TFormClass) of object;
```

property `OnGetPreviewFormClass: TRVGetFormClassEvent;`

You can return your own form class for a print preview window. This class must be inherited from `TfrmRVPreview`, otherwise an exception occurs.

1.3.1.10 TrvActionQuickPrint

`TrvActionQuickPrint` is the action for the "quick print" command (usually assigned to a toolbar button).

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

```
TrvActionQuickPrint = class (TrvCustomPrintAction (131))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction
```

TAction

TrvCustomAction ⁽³³⁵⁾

TrvAction ⁽³¹³⁾

TrvCustomPrintAction ⁽¹³¹⁾

Description

This action prints the document from TCustomRichViewEdit component without displaying a printing dialog.

This action uses settings defined in TrvActionPageSetup ⁽¹¹⁶⁾ action. If GetControlPanel ⁽³³⁸⁾.RVPrint ⁽⁵⁵⁾ is not assigned, the action uses the first TRVPrint component found on the form owning the TCustomRichViewEdit component. If not found, it creates a temporal TRVPrint with default property settings.

This action is enabled only if at least one printer is installed in the system (Printer.Printers.Count>0).

See also:

- TrvActionPrint ⁽¹¹⁸⁾
- TrvActionPrintPreview ⁽¹²⁰⁾

1.3.1.10.1 Properties

In TrvActionQuickPrint

■ Title ⁽¹²⁴⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

■ AutoCheck

■ Caption

■ Checked

DisableIfNoHandler

■ Enabled

■ GroupIndex

■ HelpContext

■ HelpKeyword

■ HelpType

■ Hint

■ ImageIndex

■ Name

- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.10.1.1 TrvActionQuickPrint.Title

Determines the text that is listed in the Print Manager.

property Title: TRVUnicodeString;

See also:

TrvActionPrint.Title⁽¹²⁰⁾

1.3.1.11 TrvActionSave

TrvActionSave is the action for "File | Save" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionSave = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action saves the document from TCustomRichViewEdit to a file. In addition, this action maintains a list of all documents (TCustomRichViewEdits associated with their file names and formats).

If the document has a temporal name (assigned after executing TrvActionNew⁽¹⁰³⁾ action), this action executes TrvActionSaveAs⁽¹²⁹⁾ action. So, if you use this action, you must also use TrvActionSaveAs⁽¹²⁹⁾ action. The link between the actions may be defined explicitly (via ActionSaveAs⁽¹²⁵⁾ property) or implicitly (this action finds and uses TrvActionSaveAs⁽¹²⁹⁾ action placed on the same form/datamodule).

If the document has a permanent name and format (assigned after executing TrvActionOpen⁽¹⁰⁸⁾ or TrvActionSaveAs⁽¹²⁹⁾ action), the action just saves the file.

If the file cannot be saved, an error message is displayed.

CanCloseDoc⁽¹²⁷⁾ method should be used in Form.OnCloseQuery event.

1.3.1.11.1 Properties

In TrvActionSave

- ActionSaveAs ⁽¹²⁵⁾
- CreateDirectoryForHTMLImages ⁽¹²⁶⁾
- DisableWhenUnmodified ⁽¹²⁶⁾
- ▶ Documents ⁽¹²⁶⁾
- LostFormatWarning ⁽¹²⁷⁾
- SuppressNextErrorMessage ⁽¹²⁷⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.11.1.1 TrvActionSave.ActionSaveAs

Links this action to TrvActionSaveAs ⁽¹²⁹⁾ action.

```
property ActionSaveAs: TrvActionSaveAs (129);
```

TrvActionSaveAs ⁽¹²⁹⁾ action is used when the document has a temporal name (after the execution of TrvActionNew ⁽¹⁰³⁾ action). In this case, instead of saving, the action executes TrvActionSaveAs ⁽¹²⁹⁾ action.

If this link is not defined, the first found TrvActionSaveAs ⁽¹²⁹⁾ action on the same form/datamodule is used. If no TrvActionSaveAs ⁽¹²⁹⁾ action found, an exception occurs on executing.

1.3.1.11.1.2 TrvActionSave.CreateDirectoryForHTMLImages

Specifies whether a subdirectory for saving images is created when saving HTML file.

property `CreateDirectoryForHTMLImages: Boolean;`

This property is used when saving to HTML format.

If *True*, images for HTML files are saved in subdirectory having the same name as HTML file but '.files' extension.

If *False*, all images are saved in the same directory as the HTML file.

This option is used for images with undefined (empty) file names, or when *rvhtmlsioUseItemImageFileNames* is not included in `HTMLSaveProperties.ImageOptions` of the target editor.

Default value:

True

See also:

`TrvActionExport.CreateDirectoryForHTMLImages` ⁽¹¹⁵⁾

1.3.1.11.1.3 TrvActionSave.DisableWhenUnmodified

Allows disabling the action if the editor is unmodified.

property `DisableWhenUnmodified: Boolean;`

If *False*, the action is always enabled.

If *True*, the action is enabled only when a document in the target editor is changed.

Default value

False

1.3.1.11.1.4 TrvActionSave.Documents

A read-only list of opened documents.

property `Documents: TRVList;`

This is a list of `TrvaDocumentInfo` objects.

The main properties of `TrvaDocumentInfo` are:

- `rve`: `TCustomRichViewEdit` – editor;
- `FileName`: `TRVUnicodeString` – file name (full path) for the document in `rve`.
- `FileFormat`: `TrvFileSaveFilter` ⁽³⁸⁸⁾ – format for saving this file.
- `CustomFilterIndex`: Integer, used only if `FileFormat=ffeCustom`. Custom formats are numbered from 1 in the order they are listed in the `CustomFilter` ⁽¹⁰⁰⁾ property of the linked `TrvActionSaveAs` ⁽¹²⁹⁾ action.
- `Defined` – *False* if `FileName` and `FileFormat` contain temporal values assigned for new document.

This list is maintained automatically.

See also methods:

- `FindDoc` ⁽¹²⁸⁾

1.3.1.11.1.5 TrvActionSave.LostFormatWarning

Lists "lossy" file formats.

property LostFormatWarning: TrvFileSaveFilterSet³⁸⁸;

When saving to file in one of these formats, a warning/confirmation is displayed:

"<FileName> may contain features that are not compatible with the chosen saving format. Do you want to save the document in this format?". This message is displayed only when saving the document in this format for the first time.

Default value:

all formats except for RVF (RichView Format)

1.3.1.11.1.6 TrvActionSave.SuppressNextErrorMessage

Turns off the default error message when saving to the custom format (*ffeCustom*³⁸⁸).

property SuppressNextErrorMessage: Boolean;

This property is set to *False* before calling GetControlPanel³³⁸.OnCustomFileOperation⁶⁴.

If you assign *True* to this property in this event, an error message will not be displayed even if *False* is returned in the Success parameter of this event.

1.3.1.11.2 Methods

In TrvActionSave

CanCloseDoc¹²⁷

FindDoc¹²⁸

Inherited from TrvCustomAction³³⁵

GetControlPanel³³⁸

1.3.1.11.2.1 TrvActionSave.CanCloseDoc

Performs necessary operations when closing the form containing **rve**.

function CanCloseDoc(rve: TCustomRichViewEdit): Boolean;

If document in **rve** is not modified, the method returns *True*.

Otherwise, the method asks the user: *"Save changes to <FileName>?"* (Yes/No/Cancel). If "Yes", the method saves and returns *True*. If "No", the method does not save and returns *True*. If "Cancel", the method returns *False*.

This method should be used in OnCloseQuery method of the form containing **rve**.

Example:

```
procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
    CanClose := rvActionSave1.CanCloseDoc(RichViewEdit1);
end;
```

1.3.1.11.2.2 TrvActionSave.FindDoc

Returns the index of item in Documents⁽¹²⁶⁾ for the given editor **rve**. Returns -1 if not found.

```
function FindDoc(rve: TCustomRichViewEdit): Integer;
```

1.3.1.11.3 Events

In TrvActionSave

- OnDocumentFileChange⁽¹²⁸⁾
- OnSave⁽¹²⁸⁾
- OnSaving⁽¹²⁸⁾

1.3.1.11.3.1 TrvActionSave.OnDocumentFileChange

Occurs when file for document in **Editor** is changed.

type

```
TRVFileChangeEvent = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit; const FileName: TRVUnicodeString;  
    FileFormat: TrvFileSaveFilter(388); IsNew: Boolean) of object;
```

```
property OnDocumentFileChange: TRVFileChangeEvent;
```

This event occurs in the following cases:

- TrvActionNew⁽¹⁰³⁾ action is executed;
- TrvActionOpen⁽¹⁰⁸⁾ action is executed, or TrvActionOpen.LoadFile⁽¹¹¹⁾ is called;
- TrvActionSaveAs⁽¹²⁹⁾ action is executed.

FileName is a file name (full path) assigned to the document in **Editor**.

FileFormat is a file format assigned to the document in **Editor**.

IsNew is *True* after the execution of TrvActionNew⁽¹⁰³⁾ action.

If **FileFormat**=*ffeCustom*, this event does not allow to identify the specific custom format. You can get it using Documents⁽¹²⁶⁾ property.

1.3.1.11.3.2 TrvActionSave.OnSaving, OnSave

The events occur before and after saving to a file.

type

```
TRVSaveFileEvent = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit; const FileName: TRVUnicodeString;  
    FileFormat: TrvFileSaveFilter(388);  
    CustomFilterIndex: Integer) of object;
```

```
property OnSaving: TRVSaveFileEvent;
```

```
property OnSave: TRVSaveFileEvent;
```

OnSaving occurs before saving to a file. If saving shows a dialog window (for example, for choosing a text file code page), this event occurs before showing these dialogs.

OnSave occurs after successful saving. It does not occur if saving was canceled (for example, in a code page dialog), or if an error occurs when saving to a file.

Parameters:

Editor – editor for saving.

FileName – file name (full path) to save.

FileFormat – format to save.

CustomFilterIndex used only if **FileFormat** is *ffeCustom*.

If **FileFormat** = *ffeCustom*, **CustomFilterIndex** identifies a custom file format. Custom formats are numbered from 1 in the order they are listed in the CustomFilter⁽¹⁰⁰⁾ property of the linked TrvActionSaveAs⁽¹²⁹⁾ action.

1.3.1.12 TrvActionSaveAs

TrvActionSaveAs is the action for "File | Save As..." command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionSaveAs = class (TrvActionCustomFileIO(98))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionCustomIO(100)
TrvActionCustomFileIO(98)

```

Description

This action displays a file saving dialog and saves the content of TRichViewEdit component in the chosen file. The editor remains associated with this file name and file format, so subsequent executions of TrvActionSave⁽¹²⁴⁾ action save its content in this file in this format.

The following formats are supported:

- RichView Format (RVF)
- Rich Text Format (RTF)
- Microsoft Word Document (DocX)
- Text files (TXT, Unicode and ANSI)
- **Markdown** (MARKDOWN, MDOWN, MKDN, MD, MKD, MDWN, MDTXT, MDTEXT)
- HTML
- RichViewXML Format (XML, only if RichViewXML⁽³⁷⁰⁾ is used)
- custom formats, supported in GetControlPanel⁽³³⁸⁾.OnCustomFileOperation⁽⁶⁴⁾ event.

If you use this action, you must also use TrvActionSave⁽¹²⁴⁾ action.

This action must be linked to TrvActionSave⁽¹²⁴⁾ action, because TrvActionSave⁽¹²⁴⁾ contains a list of all documents opened in TCustomRichViewEdit controls. Besides, this action uses TrvActionSave⁽¹²⁴⁾ to save the file when file name and format are chosen. This link may be defined explicitly (via ActionSave⁽¹³¹⁾ property) or implicitly (this action finds and uses TrvActionSave⁽¹²⁴⁾ action placed on the same form/datamodule).

This action performs the following tasks:

1. Shows a file saving dialog allowing the user to choose the file name and format for saving.
2. If the chosen format is in LostFormatWarning⁽¹²⁷⁾ of TrvActionSave⁽¹²⁴⁾, displays a warning about lossy saving.
3. Saves the file, associates TCustomRichViewEdit control with this file name and file format.
4. Calls OnDocumentFileChange⁽¹²⁸⁾ event of TrvActionSave⁽¹²⁴⁾ action.

1.3.1.12.1 Properties

In TrvActionSaveAs

- ActionSave⁽¹³¹⁾
- Filter⁽¹³¹⁾

Derived from TrvActionCustomFileIO⁽⁹⁸⁾

- CustomFilter⁽¹⁰⁰⁾

Derived from TrvActionCustomIO⁽¹⁰⁰⁾

- AutoUpdateFileName⁽¹⁰¹⁾
- DialogTitle⁽¹⁰²⁾
- FileName⁽¹⁰²⁾
- InitialDir⁽¹⁰²⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType

- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.1.12.1.1 TrvActionSaveAs.ActionSave

Links this action to TrvActionSave⁽¹²⁴⁾ action.

property ActionSave: TrvActionSave⁽¹²⁴⁾;

If this link is not defined, the first found TrvActionSave⁽¹²⁴⁾ action on the same form/datamodule is used. If no TrvActionSave⁽¹²⁴⁾ action found, an exception occurs on executing.

1.3.1.12.1.2 TrvActionSaveAs.Filter

Defines a set of file formats appearing in the file saving dialog.

property Filter: TrvFileSaveFilterSet⁽³⁸⁸⁾;

If *ffeCustom* is included, formats listed in the CustomFilter⁽¹⁰⁰⁾ property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel⁽³³⁸⁾.RVFFilter⁽⁵⁴⁾.

Default value:

all formats

1.3.1.13 TrvCustomPrintAction

TrvCustomPrintAction is a base class for the printing actions.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvCustomPrintActions = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action is not used directly, this is the base class for the following actions:

- TrvActionPrintPreview⁽¹²⁰⁾

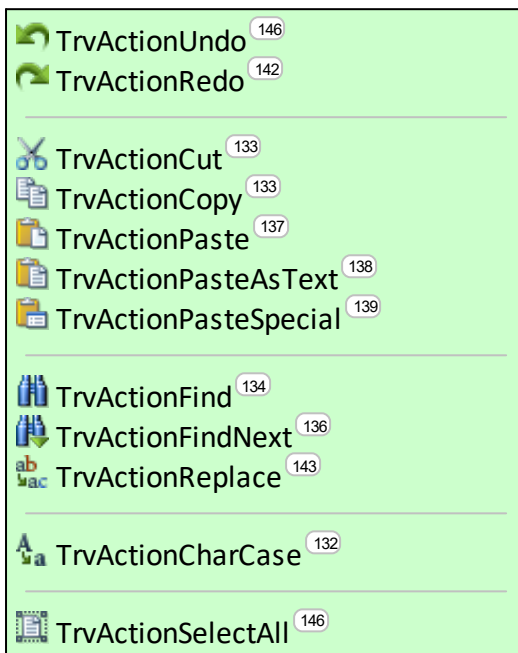
- TrvActionQuickPrint ⁽¹²²⁾
- TrvActionPrint ⁽¹¹⁸⁾

All these actions are enabled only if at least one printer installed in the system (Printer.Printers.Count>0).

1.3.2 Edit

Edit Actions

This group of actions includes undo/redo commands, Clipboard commands, find and replace commands, changing character case, page break commands.



1.3.2.1 TrvActionCharCase

TrvActionCharCase is the action for "Character Case" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCharCase = class (TrvAction (313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

The action changes character case in the selected fragment: "all lower case" → "all upper case" → "first letter in each word upper case" → "all lower case" → ...

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

Limitation: this action cannot be applied to multicell selection in table.

See also:

- RVChangeCharCase, RVGetCharCase ⁽³⁸⁵⁾ procedures.
- TrvActionFontAllCaps ⁽¹⁴⁸⁾

1.3.2.2 TrvActionCopy

TrvActionCopy is the action for "Edit | Copy" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCopy = class (TrvCustomEditAction (147))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvCustomEditAction (147)

```

Description

This action copies the selection to the Clipboard (calls TCustomRichViewEdit.CopyDef).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction ⁽¹⁴⁷⁾ topic.

1.3.2.3 TrvActionCut

TrvActionCut is the action for "Edit | Cut" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCut = class (TrvCustomEditAction (147))
```

Hierarchy

```

TObject
TPersistent

```

TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvCustomEditAction ⁽¹⁴⁷⁾

Description

This action cuts the selection to the Clipboard (calls *TCustomRichViewEdit.CutDef*).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action can work with several types of editors, see the comments in the *TrvCustomEditAction* ⁽¹⁴⁷⁾ topic.

1.3.2.4 TrvActionFind

TrvActionFind is the action for "Edit | Find" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionFind = class (TrvAction (313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action displays a *TFindDialog* for searching a substring in the *TCustomRichViewEdit* component. If at the moment of execution a single line of text is selected, this text is copied to the find dialog. If at the moment of execution a replace dialog is shown, it is closed.

The search starts from the caret position to the end or to the beginning of the document. If the text is found, it is selected. If not found, the action's behavior depends on *GetControlPanel* ⁽³³⁸⁾ *.SearchScope* ⁽⁵⁵⁾ property:

- *rvssFromCursor*, the search stops;
- *rvssAskUser*, the action asks user whether to continue from the beginning/end;
- *rvssGlobal*, the search continues from the beginning/end.

If nothing found, a message is displayed.

In Delphi 2009+, or if TNT Controls ⁽³⁷²⁾ are used, the action searches for Unicode string. Otherwise, it searches for ANSI string.

See also:

TrvActionFindNext ⁽¹³⁶⁾
TrvActionReplace ⁽¹⁴³⁾

1.3.2.4.1 Properties

In TrvActionFind

- ActionReplace ⁽¹³⁵⁾
- FindText ⁽¹³⁶⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.2.4.1.1 TrvActionFind.ActionReplace

Links this action to TrvActionReplace ⁽¹⁴³⁾ action.

property ActionReplace: TrvActionReplace ⁽¹⁴³⁾;

If this link is not defined, the first TrvActionReplace ⁽¹⁴³⁾ action found in the same form/datamodule is used.

When this action is executed, it calls TrvActionReplace.CloseDialog ⁽¹⁴⁵⁾.

1.3.2.4.1.2 TrvActionFind.FindText

A string to initialize the find dialog.

property FindText: TRVUnicodeString;

If value of this property is not empty, it is assigned to the find dialog's FindText when a search is started. Otherwise, the dialog is initialized with text selected in the target editor.

Default value

'' (empty string)

1.3.2.4.2 Methods

In TrvActionFind

CloseDialog⁽¹³⁶⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.2.4.2.1 TrvActionFind.CloseDialog

Closes the find dialog (if it is shown).

procedure CloseDialog;

1.3.2.5 TrvActionFindNext

TrvActionFindNext is the action for "Edit | Find Next" action.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionFindNext = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action continues the search started by TrvActionFind⁽¹³⁴⁾ action. The link between the actions may be defined explicitly (via ActionFind⁽¹³⁷⁾ property) or implicitly (this action finds and uses TrvActionFind⁽¹³⁴⁾ action placed on the same form/datamodule).

1.3.2.5.1 Properties

In TrvActionFindNext

- ActionFind⁽¹³⁷⁾

Derived from TrvAction⁽³¹³⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.2.5.1.1 TrvActionFindNext.ActionFind

Links this action to TrvActionFind⁽¹³⁴⁾ action.

```
property ActionFind: TrvActionFind(134);
```

If this link is not defined, the first TrvActionFind⁽¹³⁴⁾ action found in the same form/datamodule is used. If it cannot be found, an exception occurs.

When executed, TrvActionFindNext action continues the search started by TrvActionFind⁽¹³⁴⁾ action. If search was not started, the action executes TrvActionFind⁽¹³⁴⁾ action instead.

1.3.2.6 TrvActionPaste

TrvActionPaste is the action for "Edit | Paste" command.

```
Unit RichViewActions(94);
```

Syntax

```
TrvActionPaste = class(TrvCustomEditAction(147))
```

Hierarchy

TObject

TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ³³⁵
TrvAction ³¹³
TrvCustomEditAction ¹⁴⁷

Description

This action pastes data from the Clipboard (calls `TCustomRichViewEdit.Paste`).

This action does not introduce any new properties in addition to properties ³¹⁴ of `TrvAction`.

This action can work with several types of editors, see the comments in the `TrvCustomEditAction` ¹⁴⁷ topic.

1.3.2.7 TrvActionPasteAsText

`TrvActionPasteAsText` is the action for "Edit | Paste as Text" command.

Unit `RichViewActions` ⁹⁴;

Syntax

```
TrvActionPasteAsText = class (TrvCustomEditAction 147)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ³³⁵
TrvAction ³¹³
TrvCustomEditAction ¹⁴⁷

Description

This action pastes text from the Clipboard. If the current text in the editor is Unicode, it pastes Unicode text.

This action does not introduce any new properties in addition to properties ³¹⁴ of `TrvAction`.

This action can work with several types of editors, see the comments in the `TrvCustomEditAction` ¹⁴⁷ topic.

1.3.2.8 TrvActionPasteSpecial

TrvActionPasteSpecial is the action for "Edit | Paste Special" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionPasteSpecial = class (TrvActionPaste(137))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvCustomEditAction(147)
TrvActionPaste(137)

```

Description

This action displays a dialog containing a list of data formats available in the Clipboard. When the user chooses a format, data are pasted into TCustomRichViewEdit/TSRichViewEdit component in this format.

Supported formats:

- ANSI text [Windows]
- Unicode text
- Unicode text as Markdown
- Windows bitmap
- Windows metafile
- URL ("UniformResourceLocator") [Windows]
- RTF
- RVF
- HTML
- Graphic files [Windows]
- custom formats (see OnShowing⁽¹⁴²⁾ and OnCustomPaste⁽¹⁴²⁾ events).

The action displays only formats listed in in TCustomRichViewEdit.AcceptPasteFormats. **Note:** URL format is not included in AcceptPasteFormats by default.

If GetControlPanel⁽³³⁸⁾.RVFLocalizable⁽⁵⁴⁾ = True, GetControlPanel⁽³³⁸⁾.RVFFormatTitle⁽⁵⁴⁾ is used in the list for RVF Format. You can change it to something like "My Application's Format". But in this case you need to provide a localization of this property yourself.

When pasting graphic files and images from URL, the action assigns "file name" property to the inserted images, if *rvoAssignImageFileNames* is included in the Options property of the target editor (you can modify the assigned string in OnAssignImageFileName event of the target editor). If StoreFileNameInItemName⁽¹⁴⁰⁾ = True, it also assigns the item text.

If style templates are used (`RichViewEdit.UseStyleTemplates=True`), the action displays options for insertion of RTF and RVF.

Unlike all other actions inherited from `TrvCustomEditAction`⁽¹⁴⁷⁾, this action cannot work with editor types other than `TCustomRichViewEdit` and `TSRichViewEdit`.

1.3.2.8.1 Properties

In `TrvActionPasteSpecial`

■ `StoreFileNameInItemName`⁽¹⁴⁰⁾

Derived from `TrvAction`⁽³¹³⁾

■ `Control`⁽³¹⁴⁾

Derived from `TrvCustomAction`⁽³³⁵⁾

■ `Caption`⁽³³⁶⁾

■ `ControlPanel`⁽³³⁷⁾

■ `Disabled`⁽³³⁷⁾

■ `Hint`⁽³³⁷⁾

Derived from `TAction`

■ `AutoCheck`

■ `Caption`

■ `Checked`

`DisableIfNoHandler`

■ `Enabled`

■ `GroupIndex`

■ `HelpContext`

■ `HelpKeyword`

■ `HelpType`

■ `Hint`

■ `ImageIndex`

■ `Name`

■ `SecondaryShortCuts`

■ `ShortCut`

■ `Visible`

1.3.2.8.1.1 `TrvActionPasteSpecial.StoreFileNameInItemName`

This property is used when pasting graphic files (CF_HDROP).

property `StoreFileNameInItemName: Boolean;`

If this property is *True*, a path to the graphic file is stored in the document in the item name (item text).

It's not recommended to use this option. The recommended place for storing image file names is *rvesplImageFileName* extra item property. It is assigned if *rvoAssignImageFileNames* is included in the Options property of the target editor.

Paths cannot be stored for bitmaps and metafiles pasted in CF_BITMAP and CF_METAFILEPICT formats.

Default value

False

See also:

- TrvActionInsertPicture.StoreFileNameInItemName⁽²⁵⁵⁾
- TrvActionItemProperties⁽³²⁶⁾ (does not support storing paths in item names)
- RTF reading (does not support storing paths in item names)

1.3.2.8.2 Methods

In TrvActionPasteSpecial

AddFormat⁽¹⁴¹⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.2.8.2.1 TrvActionPasteSpecial.AddFormat

Adds a new format in the paste-special dialog.

```
procedure AddFormat(const FormatName: TRVLocString(350); Format: Word);
```

This method must be called only from OnShowing⁽¹⁴²⁾ event.

Parameters:

FormatName – format name, how it appears in the dialog.

Format – Clipboard format. This parameter can be a registered format or any of the standard Clipboard formats.

Data in the format specified in this method must be available in the Clipboard. If the user will choose this format, OnCustomPaste⁽¹⁴²⁾ event will occur.

1.3.2.8.3 Events

In TrvActionPasteSpecial

■ OnCanPaste⁽¹⁴¹⁾

■ OnCustomPaste⁽¹⁴²⁾

■ OnShowing⁽¹⁴²⁾

1.3.2.8.3.1 TrvActionPasteSpecial.OnCanPaste

This event is used to update Enabled property of the action.

type

```
TRVACanPasteEvent = procedure (Sender: TrvActionPasteSpecial;  
    var CanPaste: Boolean) of object;
```

property OnCanPaste: TRVACanPasteEvent;

On input, **CanPaste** parameter is the value returned by `TCustomRichViewEdit.CanPaste`.

If you implement pasting in additional formats, assign *True* to **CanPaste** if data in these formats are available in the Clipboard.

See also events:

- OnShowing⁽¹⁴²⁾
- OnCustomPaste⁽¹⁴²⁾

1.3.2.8.3.2 TrvActionPasteSpecial.OnCustomPaste

Occurs when the user chose a Clipboard format added with `AddFormat`⁽¹⁴¹⁾ method.

type

```
TRVACustomPasteEvent = procedure (Sender: TrvActionPasteSpecial;  
    Editor: TCustomRichViewEdit; Format: Word) of object;
```

property OnCustomPaste: TRVACustomPasteEvent;

Paste data in the format **Format** into the editor **Editor**.

1.3.2.8.3.3 TrvActionPasteSpecial.OnShowing

Occurs before the paste-special dialog is shown.

property OnShowing: TNotifyEvent;

In this event, use `AddFormat`⁽¹⁴¹⁾ method to add additional formats in the dialog.

1.3.2.9 TrvActionRedo

TrvActionRedo is the action for "Edit | Redo" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionRedo = class (TrvAction(313))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction  
TAction  
TrvCustomAction(335)  
TrvAction(313)
```

Description

This action redoes the last undone editing operation (calls `TCustomRichViewEdit.Redo`).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.2.10 TrvActionReplace

TrvActionReplace is the action for "Edit | Replace" command

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionReplace = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action displays a TReplaceDialog for searching and replacing a substring in the TCustomRichViewEdit component. If at the moment of execution a single line of text is selected, this text is copied to the replace dialog (as a text to find). If at the moment of execution a find dialog is shown, it is closed.

The search starts from the caret position to the end or to the beginning of the document. If the text is found, it is replaced. If not found, the action's behavior depends on GetControlPanel⁽³³⁸⁾.SearchScope⁽⁵⁵⁾ property:

- *rvssFromCursor*, the search stops;
- *rvssAskUser*, the action asks user whether to continue from the beginning/end;
- *rvssGlobal*, the search continues from the beginning/end.

If nothing found, a message is displayed.

In Delphi 2009+, or if TNT Controls⁽³⁷²⁾ are used, the action searches for Unicode string and replaces it with Unicode string. Otherwise, it works with ANSI strings.

See also:

TrvActionFindNext⁽¹³⁶⁾

TrvActionFind⁽¹³⁴⁾

1.3.2.10.1 Properties

In TrvActionReplace

- ActionFind⁽¹⁴⁴⁾
- FindText⁽¹⁴⁴⁾
- ReplaceText⁽¹⁴⁴⁾
- ShowReplaceAllSummary⁽¹⁴⁵⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.2.10.1.1 TrvActionReplace.ActionFind

Links this action to TrvActionFind ¹³⁴ action.

property ActionFind: TrvActionFind ¹³⁴;

If this link is not defined, the first TrvActionFind ¹³⁴ action found in the same form/datamodule is used.

When this action is executed, it calls TrvActionFind.CloseDialog ¹³⁶.

1.3.2.10.1.2 TrvActionReplace.FindText, ReplaceText

Strings to initialize the replace dialog.

property FindText: TRVUnicodeString;

property ReplaceText: TRVUnicodeString;

If value of **FindText** is not empty, it is assigned to the replace dialog's FindText when a search is started. Otherwise, the dialog is initialized with text selected in the target editor.

If value of **ReplaceText** is not empty, it is assigned to the replace dialog's ReplaceText when a search is started. Otherwise, the dialog is initialized an empty string.

Default values

" (empty string)

1.3.2.10.1.3 TrvActionReplace.ShowReplaceAllSummary

Specifies whether the action should display a summary message after completing a "Replace All" command.

property ShowReplaceAllSummary: Boolean;

If *True*, a message "*N strings replaced.*" is shown.

Default value:

True

1.3.2.10.2 Methods

In TrvActionReplace

CloseDialog⁽¹⁴⁵⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.2.10.2.1 TrvActionReplace.CloseDialog

Closes the replace dialog (if it is shown).

procedure CloseDialog;

1.3.2.10.3 Events

In TrvActionReplace

■ OnReplaceAllEnd⁽¹⁴⁵⁾

■ OnReplaceAllStart⁽¹⁴⁵⁾

■ OnReplacing⁽¹⁴⁶⁾

1.3.2.10.3.1 TrvActionReplace.OnReplaceAllEnd

Occurs when "replace all" operation finishes.

property OnReplaceAllEnd: TRVAEditEvent⁽³⁸⁶⁾;

If GetControlPanel⁽³³⁸⁾.SearchScope⁽⁵⁵⁾ property is equal to *rvssAskUser* or *rvssGlobal*, this event may occur twice: when the end/beginning of the document is reached, and when all replacements are completed.

The event occurs before showing a replace-all summary message.

1.3.2.10.3.2 TrvActionReplace.OnReplaceAllStart

Occurs when "replace all" operation starts.

property OnReplaceAllStart: TRVAEditEvent⁽³⁸⁶⁾;

If GetControlPanel⁽³³⁸⁾.SearchScope⁽⁵⁵⁾ property is equal to *rvssAskUser* or *rvssGlobal*, this event may occur twice: when the search starts, and when the search continues from the beginning/end of the document.

1.3.2.10.3.3 TrvActionReplace.OnReplacing

Occurs before the action replaces one occurrence of the found text with **NewText**.

type

```
TRVAReplacingEvent = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit;  
    const NewText: TRVUnicodeString) of object;
```

property OnReplacing: TRVAReplacingEvent;

When this event occurs, the found text is selected in **Editor**.

1.3.2.11 TrvActionSelectAll

TrvActionSelectAll is the action for "Edit | Select All" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionSelectAll = class (TrvAction(313))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction  
TAction  
TrvCustomAction(335)  
TrvAction(313)
```

Description

This action selects the whole document (calls TCustomRichViewEdit.SelectAll).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.2.12 TrvActionUndo

TrvActionUndo is the action for "Edit | Undo" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionUndo = class (TrvCustomEditAction(147))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction
```

*TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvCustomEditAction* ⁽¹⁴⁷⁾

Description

This action undoes the last editing operation (calls TCustomRichViewEdit.Undo).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction ⁽¹⁴⁷⁾ topic.

1.3.2.13 TrvCustomEditAction

TrvCustomEditAction is a base class for several editing actions.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvCustomEditAction = class (TrvAction (313))
```

Hierarchy

*TObject**TPersistent**TComponent**TBasicAction**TContainedAction**TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionUndo ⁽¹⁴⁶⁾
- TrvActionCut ⁽¹³³⁾
- TrvActionCopy ⁽¹³³⁾
- TrvActionPaste ⁽¹³⁷⁾

All these actions can work not only with TCustomRichViewEdit components, but also with another editors.

By default, the list of these editors includes:

- TCustomEdit (TEdit, TMemo, TRichEdit and others);
- TComboBox (having *csDropDown* or *csSimple* style).

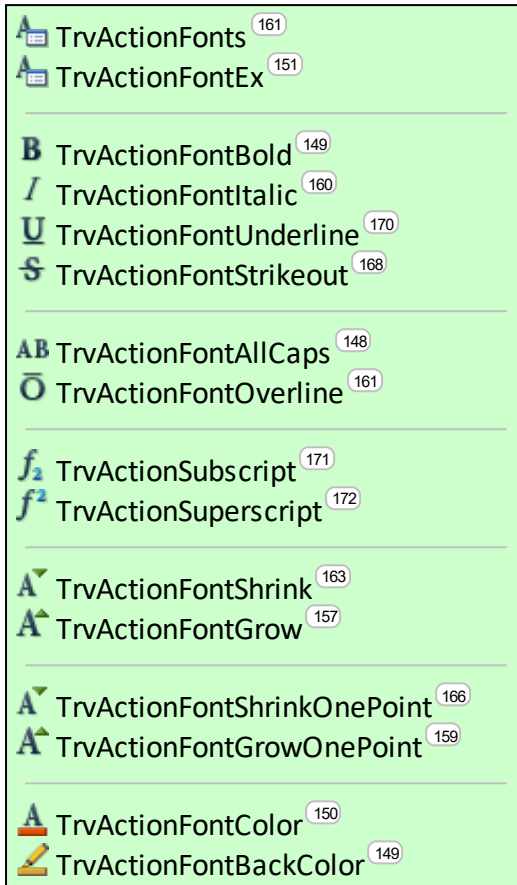
You can implement working with additional editor types by assigning your own function to RVA_EditorControlFunction ⁽³⁹¹⁾ variable.

See also RVA_EditForceDefControl ⁽³⁹¹⁾ global variable.

1.3.3 Font

Font Actions

This group of actions includes commands for changing font of selected text.



1.3.3.1 TrvActionFontAllCaps

TrvActionFontAllCaps is the action for "All Capitals" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFontAllCaps = class (TrvActionFontStyleEx(169))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionTextStyles(175)
```


TrvActionFontStyleEx ⁽¹⁶⁹⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action converts the selected fragment to upper case characters (toggle). It adds/removes *rvfsAllCaps* in *StyleEx* property for styles of all selected text items, without changing the actual text.

This is a checkbox-like action, its *Checked* property is updated depending on the state of selection (or the current text style).

See also:

- *TrvActionCharCase* ⁽¹³²⁾

1.3.3.2 TrvActionFontBackColor

TrvActionFontBackColor is the action for changing background color of the selected text.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionFontBackColor = class (TrvActionFontCustomColor (151))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionCustomColor (320)
TrvActionFontCustomColor (151)

```

Description

This action does not introduce any new properties and events in addition to properties and events of *TrvActionCustomColor* ⁽³²⁰⁾.

On execution, this action changes *BackColor* property of styles of the selected text (*RVStyle.TextStyles[].BackColor*).

See also the list of color changing actions in the topic about *TrvActionCustomColor* ⁽³²⁰⁾.

1.3.3.3 TrvActionFontBold

TrvActionFontBold is the action for "Bold" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionFontBold = class (TrvActionFontStyle (168))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTextStyles ⁽¹⁷⁵⁾
TrvActionFontStyle ⁽¹⁶⁸⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action makes the selected text bold (or not bold if already bold). It adds/removes *fsBold* in *Style* property for styles of all selected text items.

This is a checkbox-like action, its *Checked* property is updated depending on the state of selection (or the current text style).

1.3.3.4 TrvActionFontColor

TrvActionFontColor is the action for changing color of the selected text.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionFontColor = class (TrvActionFontCustomColor (151))
```

Hierarchy

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionCustomColor ⁽³²⁰⁾
TrvActionFontCustomColor ⁽¹⁵¹⁾

Description

This action does not introduce any new properties and events in addition to properties and events of *TrvActionCustomColor* ⁽³²⁰⁾. However, it changes the default value of *Color* ⁽³²¹⁾ property to *clWindowText*.

On execution, this action changes Color property of styles of the selected text (RVStyle.TextStyles[].Color).

See also the list of color changing actions in the topic about TrvActionCustomColor⁽³²⁰⁾.

1.3.3.5 TrvActionFontCustomColor

TrvActionFontCustomColor is a base class for the actions changing color properties of the selected text.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFontCustomColor = class (TrvActionCustomColor(320))
```

Hierarchy

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomColor⁽³²⁰⁾

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor⁽³²⁰⁾.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontColor⁽¹⁵⁰⁾
- TrvActionFontBackColor⁽¹⁴⁹⁾

1.3.3.6 TrvActionFontEx

TrvActionFontEx is the action for "Font" command. It uses an advanced font dialog.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFontEx = class (TrvActionFonts(161))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction

*TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionTextStyles* ⁽¹⁷⁵⁾*TrvActionFonts* ⁽¹⁶¹⁾

Description

If `UserInterface` ⁽¹⁶³⁾ = *False*, the action applies its properties to the selected text. The list of properties to apply is specified in `ValidProperties` ⁽¹⁵⁶⁾.

If `UserInterface` ⁽¹⁶³⁾ = *True*, the action uses a dialog window. The action performs the following steps:

1. assigns font attributes from the selected text to the action's properties
2. calls `OnShowingDialog` ⁽¹⁵⁷⁾ event
3. displays a dialog window
4. applies changes made in the dialog window to the action's properties
5. applies the action's properties to the selected text.

For example, if you want to apply *{"Arial", 12, not bold, dotted underline}* to the selected text in `RichViewEdit1`:

```
with rvActionFontEx1 do
begin
  UserInterface (163) := False;
  ValidProperties (156) := [rvfimFontName, rvfimSize, rvfimBold,
    rvfimUnderline, rvfimUnderlineType];
  Font (163).Name := FontName;
  FontSizeDouble (155) := 12*2;
  Font (163).Style := [fsUnderline];
  UnderlineType (156) := rvutDotted;
  ExecuteTarget(RichViewEdit1);
  UserInterface (163) := True;
end;
```

See also:

- `TrvActionFonts` ⁽¹⁶¹⁾

1.3.3.6.1 Properties

In `TrvActionFontEx`

- `AutoApplySymbolCharset` ⁽¹⁵³⁾
- `BackColor` ⁽¹⁵⁴⁾
- `CharScale` ⁽¹⁵⁴⁾
- `CharSpacing` ⁽¹⁵⁴⁾
- `FontStyleEx` ⁽¹⁵⁵⁾
- `PreviewInList` ⁽¹⁵⁵⁾
- `SubSuperScriptType` ⁽¹⁵⁵⁾
- `UnderlineColor` ⁽¹⁵⁶⁾
- `UnderlineType` ⁽¹⁵⁶⁾

ValidProperties⁽¹⁵⁶⁾
VShift⁽¹⁵⁷⁾

Derived from TrvActionFonts⁽¹⁶¹⁾

Font⁽¹⁶³⁾
■ UserInterface⁽¹⁶³⁾

Derived from TrvAction⁽³¹³⁾

■ Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

■ Caption⁽³³⁶⁾
■ ControlPanel⁽³³⁷⁾
■ Disabled⁽³³⁷⁾
■ Hint⁽³³⁷⁾

Derived from TAction

■ AutoCheck
■ Caption
■ Checked
 DisableIfNoHandler
■ Enabled
■ GroupIndex
■ HelpContext
■ HelpKeyword
■ HelpType
■ Hint
■ ImageIndex
■ Name
■ SecondaryShortCuts
■ ShortCut
■ Visible

1.3.3.6.1.1 TrvActionFontEx.AutoApplySymbolCharset

Allows to correct charset for symbol fonts.

property AutoApplySymbolCharset: Boolean;

This property works if UserInterface⁽¹⁶³⁾=False.

If *True*, then when the action applies a font name without charset (i.e. *rvfimFontName* in ValidProperties⁽¹⁵⁶⁾, *rvfimCharset* not in ValidProperties⁽¹⁵⁶⁾), if this is a symbol font (i.e. if it supports only SYMBOL_CHARSET), the action applies SYMBOL_CHARSET to the selection; if this is not a symbol font, it applies DEFAULT_CHARSET.

Default value:

True

See also:

- TRVFontComboBox⁽⁷⁵⁾.AutoCharset⁽⁷⁷⁾

1.3.3.6.1.2 TrvActionFontEx.BackgroundColor

Specifies the background color for applying to the selected text.

property BackgroundColor: TColor;

If UserInterface⁽¹⁶³⁾=False and *rvfimBackgroundColor* in ValidProperties⁽¹⁵⁶⁾, the action applies this property to the selected text.

If UserInterface⁽¹⁶³⁾=True, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, *rvfimBackgroundColor* is included in ValidProperties⁽¹⁵⁶⁾, and the action applies this property to the selected text.

This property is applied to the text style's (RVStyle.TextStyles[]) property of the same name.

1.3.3.6.1.3 TrvActionFontEx.CharScale

Specifies the character scale ratio for applying to the selected text.

property CharScale: Integer;

If UserInterface⁽¹⁶³⁾=False and *rvfimCharScale* in ValidProperties⁽¹⁵⁶⁾, the action applies this property to the selected text.

If UserInterface⁽¹⁶³⁾=True, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, *rvfimCharScale* is included in ValidProperties⁽¹⁵⁶⁾, and the action applies this property to the selected text.

This property is applied to the text style's (RVStyle.TextStyles[]) property of the same name.

1.3.3.6.1.4 TrvActionFontEx.CharSpacing

Specifies the character spacing for applying to the selected text.

property CharSpacing: TRVStyleLength;

If UserInterface⁽¹⁶³⁾=False and *rvfimCharSpacing* in ValidProperties⁽¹⁵⁶⁾, the action applies this property to the selected text.

If UserInterface⁽¹⁶³⁾=True, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, *rvfimCharSpacing* is included in ValidProperties⁽¹⁵⁶⁾, and the action applies this property to the selected text.

This property is applied to the text style's (RVStyle.TextStyles[]) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in RVStyle.Units).

1.3.3.6.1.5 TrvActionFontEx.FontSizeDouble

Specifies the font size (in half-points) for applying to the selected text.

property `FontSizeDouble: Integer;`

If `UserInterface(163)=False` and `rvfimSize` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163)=True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimSize` is included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) property `SizeDouble`.

When you assign this property, `Font(163).Size` is updated.

1.3.3.6.1.6 TrvActionFontEx.FontStyleEx

Specifies the additional text styles (overline and/or all-capitals) for applying to the selected text.

property `FontStyleEx: TRVFontStyles;`

If `UserInterface(163)=False` and `rvfimOverline` or `rvfimAllCaps` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163)=True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimOverline` and/or `rvfimAllCaps` are included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) `StyleEx` property.

1.3.3.6.1.7 TrvActionFontEx.PreviewInList

Specifies whether the font dialog displays a font preview in the list of font names.

property `PreviewInList: Boolean;`

Default value

True

1.3.3.6.1.8 TrvActionFontEx.SubSuperScriptType

Specifies the sub/superscript mode for applying to the selected text.

property `SubSuperScriptType: TRVSubSuperScriptType;`

If `UserInterface(163)=False` and `rvfimSubSuperScriptType` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163)=True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimSubSuperScriptType` is included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) property of the same name.

1.3.3.6.1.9 TrvActionFontEx.UnderlineColor

Specifies the underline color for applying to the selected text.

property UnderlineColor: TColor;

If `UserInterface(163) = False` and `rvfimUnderlineColor` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163) = True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimUnderlineColor` is included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) property of the same name.

1.3.3.6.1.10 TrvActionFontEx.UnderlineType

Specifies the underline type for applying to the selected text.

property UnderlineType: TRVUnderlineType;

If `UserInterface(163) = False` and `rvfimUnderlineType` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163) = True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimUnderlineType` is included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) property of the same name.

1.3.3.6.1.11 TrvActionFontEx.ValidProperties

Defines a set of properties for applying to the selected text.

type

```
TRVFontInfoMainProperty = (rvfimFontName, rvfimSize, rvfimCharset,
    rvfimBold, rvfimItalic, rvfimUnderline, rvfimStrikeout,
    rvfimOverline, rvfimAllCaps, rvfimVShift, rvfimColor, rvfimBackColor,
    rvfimCharScale, rvfimCharSpacing, rvfimSubSuperScriptType,
    rvfimUnderlineType, rvfimUnderlineColor);
TRVFontInfoMainProperties = set of TRVFontInfoMainProperty;
```

property ValidProperties: TRVFontInfoMainProperties;

If `UserInterface(163) = False`, assign a subset of properties (for applying) to `ValidProperties`.

If `UserInterface(163) = True`, this property is maintained automatically. After displaying a font dialog, this property contains a set of properties corresponding to non-blank fields in the dialog.

For properties-sets (`Font(163).Style` and `FontStyleEx(155)`) each value in the set is represented by one value in this property. For example:

- if `rvfimOverline` in `ValidProperties`, `rvfsOverline` in `FontStyleEx(155)`, the selected text will be overlined;
- if `rvfimOverline` in `ValidProperties`, `rvfsOverline` not in `FontStyleEx(155)`, overlines will be removed from the selected text;
- if `rvfimOverline` not in `ValidProperties`, overlines in the selected text will not be affected.

1.3.3.6.1.12 TrvActionFontEx.VShift

Specifies the vertical shift for applying to the selected text.

property VShift: Integer;

If `UserInterface(163)=False` and `rvfimVShift` in `ValidProperties(156)`, the action applies this property to the selected text.

If `UserInterface(163)=True`, the action displays a font dialog (initialized with font properties of the selected text); if the user changed value of this property in the dialog, `rvfimVShift` is included in `ValidProperties(156)`, and the action applies this property to the selected text.

This property is applied to the text style's (`RVStyle.TextStyles[]`) property of the same name.

1.3.3.6.2 Events

In TrvActionFontEx

■ OnShowingDialog⁽¹⁵⁷⁾

1.3.3.6.2.1 TrvActionFontEx.OnShowingDialog

Occurs before showing a dialog window.

property OnShowingDialog: TNotifyEvent;

This event occurs if `UserInterface(163)=True`.

It is called when properties of the action are already assigned from attributes of the selected text of the target editor, but the dialog is not initialized yet.

In this event, you can modify properties of the action, so modified properties will be applied to the dialog.

You can use this event to initialize a dialog with predefined values, instead of taking them from a selected text.

1.3.3.7 TrvActionFontGrow

TrvActionFontGrow is the action for "Grow Font" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionFontGrow = **class** (TrvActionFontShrinkGrow⁽¹⁶⁵⁾)

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

```

TrvCustomAction ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionTextStyles* ⁽¹⁷⁵⁾*TrvActionFontShrinkGrow* ⁽¹⁶⁵⁾

Description

The action increases font size of the selected text by Percent ⁽¹⁶⁶⁾ (until it reaches MaxSize ⁽¹⁵⁹⁾).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow ⁽¹⁶⁵⁾.

1.3.3.7.1 Properties

In TrvActionFontGrow

■ MaxSize ⁽¹⁵⁹⁾

Derived from TrvActionFontShrinkGrow ⁽¹⁶⁵⁾

■ Percent ⁽¹⁶⁶⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

■ AutoCheck

■ Caption

■ Checked

 DisableIfNoHandler

■ Enabled

■ GroupIndex

■ HelpContext

■ HelpKeyword

■ HelpType

■ Hint

■ ImageIndex

■ Name

■ SecondaryShortCuts

■ ShortCut

■ Visible

1.3.3.7.1.1 TrvActionFontGrow.MaxSize

Defines maximal font size.

property MaxSize: Integer;

The action does not increase font size above this value.

Default value:

100

1.3.3.8 TrvActionFontGrowOnePoint

TrvActionFontGrowOnePoint is the action for "Grow Font by One Point" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

TrvActionFontGrowOnePoint = **class** (TrvActionTextStyles ⁽¹⁷⁵⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTextStyles ⁽¹⁷⁵⁾

Description

The action increases font size of the selected text by 1 (until it reaches MaxSize ⁽¹⁶⁰⁾).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow ⁽¹⁶⁵⁾.

1.3.3.8.1 Properties

In TrvActionFontGrowOnePoint

■ MaxSize ⁽¹⁶⁰⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.3.8.1.1 TrvActionFontGrowOnePoint.MaxSize

Defines maximal font size.

property MaxSize: Integer;

The action does not increase font size above this value.

Default value:

100

1.3.3.9 TrvActionFontItalic

TrvActionFontItalic is the action for "Italic" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionFontItalic = **class** (TrvActionFontStyle⁽¹⁶⁸⁾)

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction
 TContainedAction
 TCustomAction
 TAction
 TrvCustomAction⁽³³⁵⁾
 TrvAction⁽³¹³⁾
 TrvActionTextStyles⁽¹⁷⁵⁾
 TrvActionFontStyle⁽¹⁶⁸⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action makes the selected text italic (or not italic if already italic). It adds/removes *fsItalic* in Style property for styles of all selected text items.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

1.3.3.10 TrvActionFontOverline

TrvActionFontOverline is the action for "Overline" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionFontOverline = class (TrvActionFontStyleEx (169))
```

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction
 TContainedAction
 TCustomAction
 TAction
 TrvCustomAction ⁽³³⁵⁾
 TrvAction ⁽³¹³⁾
 TrvActionTextStyles ⁽¹⁷⁵⁾
 TrvActionFontStyleEx ⁽¹⁶⁹⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action overlines the selected text (or removes overline, if already overlined). It adds/removes *rvfsOverline* in StyleEx property for styles of all selected text items.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

1.3.3.11 TrvActionFonts

TrvActionFonts is the action for "Font" command. It uses a standard font dialog (TFontDialog).

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionFonts = class (TrvActionTextStyles (175))
```

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction

*TContainedAction**TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionTextStyles* ⁽¹⁷⁵⁾

Description

If `UserInterface` ⁽¹⁶³⁾ = *False*, the action applies `Font` ⁽¹⁶³⁾ to the selected text.

If `UserInterface` ⁽¹⁶³⁾ = *True*, the action displays a `TFontDialog` (initialized with font properties of the current text style), assigns the chosen font to `Font` ⁽¹⁶³⁾, and then applies `Font` ⁽¹⁶³⁾ to the selected text.

It's recommended to use `TrvActionFontEx` ⁽¹⁵¹⁾ instead of this action because of two reasons:

- `TrvActionFonts` applies all font properties, it's not possible to leave some fields blank in `TFontDialog`. In `TrvActionFontEx` ⁽¹⁵¹⁾, if you open a dialog and press OK without changing anything, nothing will be changed in the selected text; this is not so for this action;
- `TrvActionFontEx` ⁽¹⁵¹⁾ allows defining much more text properties.

1.3.3.11.1 Properties

In `TrvActionFonts`

`Font` ⁽¹⁶³⁾■ `UserInterface` ⁽¹⁶³⁾

Derived from `TrvAction` ⁽³¹³⁾

■ `Control` ⁽³¹⁴⁾

Derived from `TrvCustomAction` ⁽³³⁵⁾

■ `Caption` ⁽³³⁶⁾■ `ControlPanel` ⁽³³⁷⁾■ `Disabled` ⁽³³⁷⁾■ `Hint` ⁽³³⁷⁾

Derived from `TAction`

■ `AutoCheck`■ `Caption`■ `Checked``DisableIfNoHandler`■ `Enabled`■ `GroupIndex`■ `HelpContext`■ `HelpKeyword`■ `HelpType`■ `Hint`■ `ImageIndex`■ `Name`■ `SecondaryShortCuts`

- ShortCut
- Visible

1.3.3.11.1.1 TrvActionFonts.Font

Specifies the font for applying to the selected text.

type

```
TRVAFont = class(TFont);
```

property Font: TRVAFont;

If `UserInterface(163)=False`, the action applies this property to the selected text. For `TrvActionFontEx(151)`, only properties specified in `ValidProperties(156)` are applied (a subset of *rvfimFontName*, *rvfimSize*, *rvfimCharset*, *rvfimBold*, *rvfimItalic*, *rvfimUnderline*, *rvfimStrikeout*, *rvfimColor*).

If `UserInterface(163)=True`, the action displays a font dialog, assigns the chosen font to this property, and then applies this property to the selected text.

This property is applied to the following properties of text style (`RVStyle.TextStyles[]`):

- FontName
- Size (in `TrvActionFontEx(151)`, it is overridden by `FontSizeDouble(155)`; when you assign Size, the action's `FontSizeDouble(155)` is changed accordingly)
- Charset
- Style
- Color

1.3.3.11.1.2 TrvActionFonts.UserInterface

Allows working with or without the user interaction.

property UserInterface: Boolean;

If `UserInterface=False`, the action applies its properties to the selected text.

If `UserInterface=True`, the action displays a font dialog, assigns changes to the action's properties, and then applies these properties to the selected text.

Default value:

True

1.3.3.12 TrvActionFontShrink

`TrvActionFontShrink` is the action for "Shrink Font" command.

Unit `RichViewActions(94)`;

Syntax

```
TrvActionFontShrink = class(TrvActionFontShrinkGrow(165))
```

Hierarchy

```
TObject
TPersistent
TComponent
TBasicAction
```

*TContainedAction**TCustomAction**TAction**TrvCustomAction* ³³⁵*TrvAction* ³¹³*TrvActionTextStyles* ¹⁷⁵*TrvActionFontShrinkGrow* ¹⁶⁵**Description**

The action decreases font size of the selected text by Percent ¹⁶⁶ (until it reaches MinSize ¹⁶⁵).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow ¹⁶⁵.

1.3.3.12.1 Properties**In TrvActionFontShrink**

- MinSize ¹⁶⁵

Derived from TrvActionFontShrinkGrow ¹⁶⁵

- Percent ¹⁶⁶

Derived from TrvAction ³¹³

- Control ³¹⁴

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.3.12.1 TrvActionFontShrink.MinSize

Defines minimal font size.

property MinSize: Integer;

The action does not decrease font size below this value.

Default value:

1

1.3.3.13 TrvActionFontShrinkGrow

TrvActionFontShrinkGrow is a base class for the actions changing font size of the selected text by the specified percent.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

TrvActionFontShrinkGrow = **class** (TrvActionTextStyles ⁽¹⁷⁵⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTextStyles ⁽¹⁷⁵⁾

Description

The action changes Size property of text styles for the selected text items (RVStyle.TextStyles[].Size).

This action is not used directly. The following actions are inherited from it:

- TrvActionFontShrink ⁽¹⁶³⁾
- TrvActionFontGrow ⁽¹⁵⁷⁾

See also:

- TrvActionFontShrinkOnePoint ⁽¹⁶⁶⁾
- TrvActionFontGrowOnePoint ⁽¹⁵⁹⁾

1.3.3.13.1 Properties

In TrvActionFontShrinkGrow

■ Percent ⁽¹⁶⁶⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.3.13.1.1 TrvActionFontShrinkGrow.Percent

Defines the percent for increasing or decreasing font size.

property Percent: Integer;

Default value:

10

1.3.3.14 TrvActionFontShrinkOnePoint

TrvActionFontShrinkOnePoint is the action for "Shrink Font by One Point" command.

Unit RichViewActions ⁹⁴;

Syntax

TrvActionFontShrinkOnePoint = **class** (TrvActionTextStyles ¹⁷⁵)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ³³⁵

TrvAction ⁽³¹³⁾*TrvActionTextStyles* ⁽¹⁷⁵⁾**Description**

The action decreases font size of the selected text by 1 (until it reaches *MinSize* ⁽¹⁶⁷⁾).

See the list of font size changing actions in the topic about *TrvActionFontShrinkGrow* ⁽¹⁶⁵⁾.

1.3.3.14.1 Properties**In TrvActionFontShrinkOnePoint**

- *MinSize* ⁽¹⁶⁷⁾

Derived from TrvAction ⁽³¹³⁾

- *Control* ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- *Caption* ⁽³³⁶⁾
- *ControlPanel* ⁽³³⁷⁾
- *Disabled* ⁽³³⁷⁾
- *Hint* ⁽³³⁷⁾

Derived from TAction

- *AutoCheck*
- *Caption*
- *Checked*
- *DisableIfNoHandler*
- *Enabled*
- *GroupIndex*
- *HelpContext*
- *HelpKeyword*
- *HelpType*
- *Hint*
- *ImageIndex*
- *Name*
- *SecondaryShortCuts*
- *ShortCut*
- *Visible*

1.3.3.14.1.1 TrvActionFontShrinkOnePoint.MinSize

Defines minimal font size.

property *MinSize*: Integer;

The action does not decrease font size below this value.

Default value:

1

1.3.3.15 TrvActionFontStrikeout

TrvActionFontStrikeout is the action for "Strikeout" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFontStrikeout = class (TrvActionFontStyle(168))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTextStyles⁽¹⁷⁵⁾
TrvActionFontStyle⁽¹⁶⁸⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action strikes through the selected text (toggle). It adds/removes *fsStrikeOut* in Style property for styles of all selected text items.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

1.3.3.16 TrvActionFontStyle

TrvActionFontStyle is a base class for the actions changing Style property of text styles for the selected text (RVStyle.TextStyles[].Style)

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFontStyle = class (TrvActionTextStyles(175))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTextStyles⁽¹⁷⁵⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontBold ⁽¹⁴⁹⁾
- TrvActionFontItalic ⁽¹⁶⁰⁾
- TrvActionFontUnderline ⁽¹⁷⁰⁾
- TrvActionFontStrikeout ⁽¹⁶⁸⁾

These actions change one option in the Style property of styles of text in the selected fragment of TCustomRichViewEdit (for example, *fsBold*)

These actions are checked (Checked=*True*) in the following cases:

- if several table cells are selected, all text items in these cells have the required option included in Style (for example, all text is bold);
- otherwise, if the current text style (TCustomRichViewEdit.CurTextStyleNo) has this option included in Style (for example, it is bold),

1.3.3.17 TrvActionFontStyleEx

TrvActionFontStyle is a base class for the actions changing StyleEx property of text styles for the selected text (RVStyle.TextStyles[].StyleEx)

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionFontStyleEx = class (TrvActionTextStyles (175))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionTextStyles (175)
```

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontAllCaps ⁽¹⁴⁸⁾
- TrvActionFontOverline ⁽¹⁶¹⁾

These actions change one option in the `StyleEx` property of styles of text in the selected fragment of `TCustomRichViewEdit` (for example, *rvfsOverline*)

These actions are checked (`Checked=True`) in the following cases:

- if several table cells are selected, all text items in these cells have the required option included in `StyleEx` (for example, all text is overlined);
- otherwise, if the current text style (`TCustomRichViewEdit.CurTextStyleNo`) has this option included in `StyleEx` (for example, it is overlined),

1.3.3.18 TrvActionFontUnderline

`TrvActionFontUnderline` is the action for "Underline" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionFontUnderline = class(TrvActionFontStyle168)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313
TrvActionTextStyles175
TrvActionFontStyle168

```

Description

This action does not introduce any new properties in addition to properties³¹⁴ of `TrvAction`.

This action underlines the selected text (or removes underline, if already underlined). It adds/removes *fsUnderline* in `Style` property for styles of all selected text items. This action changes `UnderlineType` property of text styles to *rvutNormal*. This action does not change underline color.

This is a checkbox-like action, its `Checked` property is updated depending on the state of selection (or the current text style).

1.3.3.19 TrvActionSSScript

`TrvActionSSScript` is a base class for the subscript and the superscript actions.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionTextStyles = class (TrvActionTextStyles175)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³
*TrvActionTextStyles*¹⁷⁵

Description

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

This action changes SubSuperScriptType property for styles of the selected text (RVStyle.TextStyles[].SubSuperScriptType).

This action is not used directly. The following actions are inherited from it:

- TrvActionSubscript¹⁷¹
- TrvActionSuperscript¹⁷²

1.3.3.20 TrvActionSubscript

TrvActionSubscript is the action for "Subscript" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionSubscript = class (TrvActionSSScript170)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³
*TrvActionTextStyles*¹⁷⁵
*TrvActionSSScript*¹⁷⁰

Description

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

The action makes the selected text subscript (or normal, if already subscript).

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

1.3.3.21 TrvActionSuperscript

TrvActionSuperscript is the action for "Superscript" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionSuperscript = class (TrvActionSSScript(170))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTextStyles⁽¹⁷⁵⁾
TrvActionSSScript⁽¹⁷⁰⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

The action makes the selected text superscript (or normal, if already superscript).

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

1.3.3.22 TrvActionTextBiDi

TrvActionTextBiDi is a base class for the actions changing the default text flow direction in the selected text.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTextBiDi = class (TrvActionTextStyles(175))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

TrvActionTextStyles ⁽¹⁷⁵⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action changes *BiDiMode* property for styles of the selected text (*RVStyle.TextStyles[]BiDiMode*).

This action is not used directly. The following actions are inherited from it:

- *TrvActionTextLTR* ⁽¹⁷³⁾
- *TrvActionTextRTL* ⁽¹⁷⁴⁾

If you use these actions, it's recommended to set *TCustomRichViewEdit.BiDiMode* to *rvbdLeftToRight* or *rvbdRightToLeft*.

Text flow direction specified in text styles has higher priority than text flow direction specified in paragraph styles.

See also:

- *TrvActionParaBiDi* ⁽¹⁹³⁾
- *TrvActionParaLTR* ⁽²¹⁰⁾
- *TrvActionParaRTL* ⁽²¹²⁾

1.3.3.23 TrvActionTextLTR

TrvActionTextLTR is the action for "Left to Right Text" command.

Unit *RichViewActions* ⁽⁹⁴⁾ ;

Syntax

```
TrvActionTextLTR = class (TrvActionTextBiDi (172))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTextStyles ⁽¹⁷⁵⁾
TrvActionTextBiDi ⁽¹⁷²⁾

Description

This action changes the default text direction in the selected text to left-to-right. If you use this action, it's recommended to set *TCustomRichViewEdit.BiDiMode* to *rvbdLeftToRight* or *rvbdRightToLeft*.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action changes BiDiMode property for styles of the selected text (RVStyle.TextStyles[].BiDiMode) to *rvbdLeftToRight*.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

See also:

- TrvActionTextRTL ⁽¹⁷⁴⁾
- TrvActionParaLTR ⁽²¹⁰⁾
- TrvActionParaRTL ⁽²¹²⁾

1.3.3.24 TrvActionTextRTL

TrvActionTextRTL is the action for "Right to Left Text" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTextRTL = class (TrvActionTextBiDi (172))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionTextStyles (175)
TrvActionTextBiDi (172)
```

Description

This action changes the default text direction in the selected text to right-to-left. If you use this action, it's recommended to set TCustomRichViewEdit.BiDiMode to *rvbdLeftToRight* or *rvbdRightToLeft*.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action changes BiDiMode property for styles of the selected text (RVStyle.TextStyles[].BiDiMode) to *rvbdRightToLeft*.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current text style).

See also:

- TrvActionTextLTR ⁽¹⁷³⁾
- TrvActionParaLTR ⁽²¹⁰⁾
- TrvActionParaRTL ⁽²¹²⁾

1.3.3.25 TrvActionTextStyles

TrvActionTextStyles is a base class for:

- the actions applying changes to text style of the selected fragment of TCustomRichViewEdit component,
- TrvActionInsertHyperlink⁽²³⁴⁾.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTextStyles = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

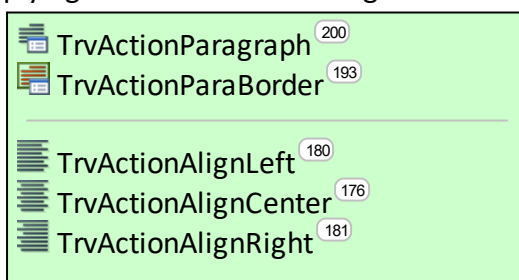
This action is not used directly. This action has the following direct descendants:

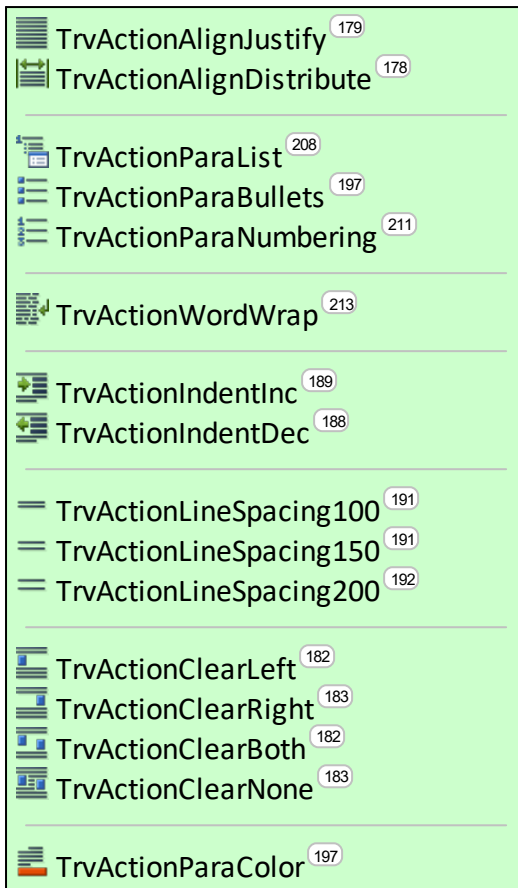
- TrvActionFonts⁽¹⁶¹⁾
- TrvActionFontStyle⁽¹⁶⁸⁾
- TrvActionFontStyleEx⁽¹⁶⁹⁾
- TrvActionSSScript⁽¹⁷⁰⁾
- TrvActionTextBiDi⁽¹⁷²⁾
- TrvActionFontShrinkGrow⁽¹⁶⁵⁾
- TrvActionFontShrinkOnePoint⁽¹⁶⁶⁾
- TrvActionFontGrowOnePoint⁽¹⁵⁹⁾
- TrvActionInsertHyperlink⁽²³⁴⁾

1.3.4 Paragraph

Paragraph Actions

This group of actions includes commands for changing attributes of selected paragraphs, and for applying bullets and numbering.





1.3.4.1 TrvActionAlignCenter

TrvActionAlignCenter is the action for "Align Center" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionAlignCenter = class (TrvActionAlignment (180))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionAlignment ⁽¹⁸⁰⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action aligns the selected paragraphs to the center.

This action changes Alignment property for styles of the selected paragraphs (RVStyle.ParaStyles[].Alignment) to *rvaCenter*.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style.

1.3.4.2 TrvActionAlignCustomJustify

TrvActionAlignCustomJustify is a base class for the actions changing alignment of selected paragraphs to the both left and right sides.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionAlignCustomJustify = class (TrvActionAlignment (180))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionAlignment ⁽¹⁸⁰⁾

Description

Properties:

LastLineAlignment ⁽¹⁷⁸⁾ is applied to the selected paragraphs, if UseLastLineAlignment ⁽¹⁷⁸⁾ = *True*.

This action is not used directly. The following actions are inherited from it:

- TrvActionAlignLeft ⁽¹⁸⁰⁾
- TrvActionAlignRight ⁽¹⁸¹⁾
- TrvActionAlignCenter ⁽¹⁷⁶⁾
- TrvActionAlignJustify ⁽¹⁷⁹⁾

1.3.4.2.1 Properties

In TrvActionAlignCustomJustify

- LastLineAlignment ⁽¹⁷⁸⁾
- UseLastLineAlignment ⁽¹⁷⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.2.1.1 TrvActionAlignCustomJustify.LastLineAlignment, UseLastLineAlignment

Defines the alignment applied to the last line of paragraphs.

```
property LastLineAlignment: TRVLastLineAlignment;
property UseLastLineAlignment: Boolean;
```

If **UseLastLineAlignment**=*True*, **LastLineAlignment** is applied to the selected paragraphs.

Default values:

- **UseLastLineAlignment**:*True*
- **LastLineAlignment**:
 - in TrvActionAlignJustify ¹⁷⁹: *rvllaDefault*
 - in TrvActionAlignDistribute: *rvllaJustify*

1.3.4.3 TrvActionAlignDistribute

TrvActionAlignDistribute is the action for "Distribute" command.

Unit RichViewActions ⁹⁴;

Syntax

```
TrvActionAlignDistribute = class (TrvActionAlignCustomJustify 177)
```

Hierarchy

TObject
TPersistent

TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionAlignment ⁽¹⁸⁰⁾
TrvActionAlignCustomJustify ⁽¹⁷⁷⁾

Description

This action does not introduce any new properties in addition to properties ⁽¹⁷⁷⁾ of *TrvActionAlignCustomJustify*.

This action aligns the selected paragraphs both to the right and to the left sides by adding space between all characters.

This action changes Alignment property for styles of the selected paragraphs (*RVStyle.ParaStyles[]*.Alignment) to *rvaDistribute*. Additionally, it may change LastLineAlignment property to LastLineAlignment ⁽¹⁷⁸⁾.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style.

1.3.4.4 TrvActionAlignJustify

TrvActionAlignJustify is the action for "Justify" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionAlignJustify = class (TrvActionAlignCustomJustify (177))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionAlignment ⁽¹⁸⁰⁾
TrvActionAlignCustomJustify ⁽¹⁷⁷⁾

Description

This action does not introduce any new properties in addition to properties ⁽¹⁷⁷⁾ of *TrvActionAlignCustomJustify*.

This action aligns the selected paragraphs both to the right and to the left sides by adding space between words.

This action changes Alignment property for styles of the selected paragraphs (RVStyle.ParaStyles[].Alignment) to *rvaJustify*. Additionally, it may change LastLineAlignment property to LastLineAlignment⁽¹⁷⁸⁾.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style.

1.3.4.5 TrvActionAlignLeft

TrvActionAlignLeft is the action for "Align Left" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionAlignLeft = class (TrvActionAlignment(180))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionParaStyles⁽²¹²⁾
TrvActionAlignment⁽¹⁸⁰⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action aligns the selected paragraphs to the left.

This action changes Alignment property for styles of the selected paragraphs (RVStyle.ParaStyles[].Alignment) to *rvaLeft*.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style.

1.3.4.6 TrvActionAlignment

TrvActionAlignment is a base class for the actions changing alignment of the selected paragraphs.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionAlignment = class (TrvActionParaStyles(212))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action is not used directly. The following actions are inherited from it:

- *TrvActionAlignLeft* ⁽¹⁸⁰⁾
- *TrvActionAlignRight* ⁽¹⁸¹⁾
- *TrvActionAlignCenter* ⁽¹⁷⁶⁾
- *TrvActionAlignJustify* ⁽¹⁷⁹⁾
- *TrvActionAlignDistribute* ⁽¹⁷⁸⁾

1.3.4.7 TrvActionAlignRight

TrvActionAlignRight is the action for "Align Right" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionAlignRight = class (TrvActionAlignment (180))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionAlignment ⁽¹⁸⁰⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action aligns the selected paragraphs to the right side.

This action changes Alignment property for styles of the selected paragraphs (*RVStyle.ParaStyles[]*.Alignment) to *rvaRight*.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style.

1.3.4.8 TrvActionClearBoth

TrvActionClearBoth is the action for "Clear Text Flow at Both Sides"

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionClearBoth = class (TrvActionClearTextFlow(184))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionClearTextFlow⁽¹⁸⁴⁾

Description

This action does not allow the selected paragraphs flowing around left- and right-aligned pictures. The selected paragraphs will be placed below side-aligned pictures.

This actions calls RichViewEdit.ClearTextFlow(*True*, *True*).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.4.9 TrvActionClearLeft

TrvActionClearLeft is the action for "Clear Text Flow at Left Side"

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionClearLeft = class (TrvActionClearTextFlow(184))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionClearTextFlow⁽¹⁸⁴⁾

Description

This action does not allow the selected paragraphs flowing around left-aligned pictures. The selected paragraphs will be placed below left-aligned pictures. Text flow around right-aligned pictures is allowed.

This actions calls `RichViewEdit.ClearTextFlow(True, False)`.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of `TrvAction`.

1.3.4.10 TrvActionClearNone

`TrvActionClearNone` is the action for "Normal Text Flow"

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

```
TrvActionClearNone = class (TrvActionClearTextFlow (184))
```

Hierarchy

```

    TObject
    TPersistent
    TComponent
    TBasicAction
    TContainedAction
    TCustomAction
    TAction
    TrvCustomAction (335)
    TrvAction (313)
    TrvActionClearTextFlow (184)

```

Description

This action allows the selected paragraphs flowing around left- and right-aligned pictures.

This actions calls `RichViewEdit.ClearTextFlow(False, False)`.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of `TrvAction`.

1.3.4.11 TrvActionClearRight

`TrvActionClearRight` is the action for "Clear Text Flow at Right Side"

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

```
TrvActionClearRight = class (TrvActionClearTextFlow (184))
```

Hierarchy

```

    TObject
    TPersistent
    TComponent
    TBasicAction
    TContainedAction

```

*TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionClearTextFlow* ⁽¹⁸⁴⁾

Description

This action does not allow the selected paragraphs flowing around right-aligned pictures. The selected paragraphs will be placed below right-aligned pictures. Text flow around left-aligned pictures is allowed.

This actions calls `RichViewEdit.ClearTextFlow(False, True)`.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of `TrvAction`.

1.3.4.12 TrvActionClearTextFlow

`TrvActionClearTextFlow` is a base class for the actions clearing a text flow around left- and right-aligned pictures.

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

```
TrvActionClearTextFlow = class (TrvAction (313))
```

Hierarchy

*TObject**TPersistent**TComponent**TBasicAction**TContainedAction**TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of `TrvAction`.

This action is not used directly. The following actions are inherited from it:

- `TrvActionClearLeft` ⁽¹⁸²⁾
- `TrvActionClearRight` ⁽¹⁸³⁾
- `TrvActionClearBoth` ⁽¹⁸²⁾
- `TrvActionClearNone` ⁽¹⁸³⁾

1.3.4.13 TrvActionCustomParaListSwitcher

`TrvActionCustomParaListSwitcher` is a base class for the actions toggling paragraphs' bullets or numbering on/off.

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

```
TrvActionCustomParaListSwitcher = class(TrvAction313)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³

Description

This action is checked when the current paragraph has a list marker with properties identical to the properties of ListLevels¹⁸⁶. In this case, applying the action removes bullets and numbering in all selected paragraphs. Otherwise, applying this action applies a list style with levels specified in ListLevels¹⁸⁶ to all selected paragraphs. The 0th list level is applied to paragraphs that did not have list markers; for other paragraphs, level indices are not changed.

This action is not used directly. The following actions are inherited from it:

- TrvActionParaBullets¹⁹⁷
- TrvActionParaNumbering²¹¹

1.3.4.13.1 Properties

In TrvActionCustomParaListSwitcher

- IndentStep¹⁸⁶
- ListLevels¹⁸⁶

Derived from TrvAction³¹³

- Control³¹⁴

Derived from TrvCustomAction³³⁵

- Caption³³⁶
- ControlPanel³³⁷
- Disabled³³⁷
- Hint³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex

- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.13.1.1 TrvActionCustomParaListSwitcher.IndentStep

Defines the step for increasing LeftIndent and MarkerIndent of ListLevels ⁽¹⁸⁶⁾.

property IndentStep: TRVStyleLength;

This value is measured in GetControlPanel ⁽³³⁸⁾.UnitsProgram ⁽⁵⁷⁾.

Assigning value to this property resets ListLevels ⁽¹⁸⁶⁾ to default value.

Default value:

24

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

See also:

- TrvActionIndent ⁽¹⁸⁶⁾.IndentStep ⁽¹⁸⁸⁾
- TrvActionParaList ⁽²⁰⁸⁾.IndentStep ⁽²⁰⁹⁾

1.3.4.13.1.2 TrvActionCustomParaListSwitcher.ListLevels

Specifies list levels for applying to the selected paragraphs.

property ListLevels: TRVListLevelCollection;

Subproperties of this property are measured in GetControlPanel ⁽³³⁸⁾.UnitsProgram ⁽⁵⁷⁾.

Default value for TrvActionParaBullets ⁽¹⁹⁷⁾:

9 levels, all with ListType=*rvlstBullet*, FirstIndent=0, MarkerIndent is increased by IndentStep ⁽¹⁸⁶⁾*2 starting from 0, LeftIndent is increased by IndentStep ⁽¹⁸⁶⁾*2 starting from IndentStep ⁽¹⁸⁶⁾, Font.Size=12, disk/circle/square cycled 3 times.

Default value for TrvActionParaNumbering ⁽²¹¹⁾:

9 levels, all with ListType=*rvlstDecimal*, FirstIndent=0, MarkerIndent is increased by IndentStep ⁽¹⁸⁶⁾*2 starting from 0, LeftIndent is increased by IndentStep ⁽¹⁸⁶⁾*2 starting from IndentStep ⁽¹⁸⁶⁾, Font.Size=10, format sting displays "N."

1.3.4.14 TrvActionIndent

TrvActionIndent is a base class for the actions changing indents in the selected paragraphs.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

TrvActionIndent = **class** (TrvActionParaStyles ⁽²¹²⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾

Description

The actions inherited from this class perform the following operations:

- change LeftIndent property of styles for the selected paragraphs (RVStyle.ParaStyles[].LeftIndent); for paragraphs styles having BiDiMode=*rvbdRightToLeft*, RightIndent is changed instead;
- change list levels for all paragraphs with bullets&numbering (by calling TCustomRichViewEdit.ChangeListLevels).

The actions do not change indents of list styles for bulleted or numbered paragraphs, they promote/demote their levels instead.

To do: to implement changing of LeftIndent property of list style, if bullets or numbering has only one level.

This action is not used directly. The following actions are inherited from it:

- TrvActionIndentDec ⁽¹⁸⁸⁾
- TrvActionIndentInc ⁽¹⁸⁹⁾

1.3.4.14.1 Properties

In TrvActionIndent

- IndentStep ⁽¹⁸⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler

- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.14.1 TrvActionIndent.IndentStep

Defines the step for paragraph indenting or unindenting.

property IndentStep: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value:

24

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

See also:

- TrvActionCustomParaListSwitcher⁽¹⁸⁴⁾.IndentStep⁽¹⁸⁶⁾
- TrvActionParaList⁽²⁰⁸⁾.IndentStep⁽²⁰⁹⁾

1.3.4.15 TrvActionIndentDec

TrvActionIndentDec is the action for "Decrease Indent" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionIndentDec = **class** (TrvActionIndent⁽¹⁸⁶⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionParaStyles⁽²¹²⁾
TrvActionIndent⁽¹⁸⁶⁾

Description

This action does not introduce any new properties in addition to properties ⁽¹⁸⁷⁾ of TrvActionIndent.

The action does not allow changing the paragraph indent to a negative value. If TCustomRichViewEdit.LeftMargin+LeftIndent+FirstIndent becomes negative, the action changes FirstIndent to -TCustomRichViewEdit.LeftMargin-LeftIndent (for right-to-left paragraphs, these formulas use RightIndent instead).

1.3.4.16 TrvActionIndentInc

TrvActionIndentInc is the action for "Increase Indent" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionIndentInc = class (TrvActionIndent (186))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾
TrvActionIndent ⁽¹⁸⁶⁾

Description

The action does not allow changing the paragraph indent above the MaxIndent ⁽¹⁹⁰⁾.

1.3.4.16.1 Properties

In TrvActionIndentInc

■ IndentMax ⁽¹⁹⁰⁾

Derived from TrvActionIndent ⁽¹⁸⁶⁾

■ IndentStep ⁽¹⁸⁸⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾
 ■ ControlPanel ⁽³³⁷⁾
 ■ Disabled ⁽³³⁷⁾
 ■ Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.16.1.1 TrvActionIndentInc.IndentMax

Defines the maximal possible value for the indent.

property IndentMax: TRVStyleLength;

The action does not increase the paragraph's indent above this value.

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value:

200

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.4.17 TrvActionLineSpacing

TrvActionLineSpacing is a base class for the actions changing line spacing of the selected paragraphs.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionLineSpacing = **class** (TrvActionParaStyles⁽²¹²⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾

TrvAction ⁽³¹³⁾*TrvActionParaStyles* ⁽²¹²⁾**Description**

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action is not used directly. The following actions are inherited from it:

- *TrvActionLineSpacing100* ⁽¹⁹¹⁾
- *TrvActionLineSpacing150* ⁽¹⁹¹⁾
- *TrvActionLineSpacing200* ⁽¹⁹²⁾

These actions change the following properties of styles of the selected paragraphs (*RVStyle.ParaStyles[]*):

- *LineSpacingType* := *rvlsPercent*
- *LineSpacing*

1.3.4.18 TrvActionLineSpacing100

TrvActionLineSpacing100 is the action for "Single Line Spacing" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionLineSpacing100 = class (TrvActionLineSpacing (190))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionParaStyles (212)
TrvActionLineSpacing (190)

```

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This actions changes the following properties of styles of the selected paragraphs (*RVStyle.ParaStyles[]*):

- *LineSpacingType* := *rvlsPercent*
- *LineSpacing* := 100

1.3.4.19 TrvActionLineSpacing150

TrvActionLineSpacing150 is the action for "1.5 Line Spacing" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionLineSpacing150 = class (TrvActionLineSpacing190)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313
TrvActionParaStyles212
TrvActionLineSpacing190

```

Description

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

This actions changes the following properties of styles of the selected paragraphs (RVStyle.ParaStyles[]):

- LineSpacingType := *rvlsPercent*
- LineSpacing := 150

1.3.4.20 TrvActionLineSpacing200

TrvActionLineSpacing200 is the action for "Double Line Spacing" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionLineSpacing200 = class (TrvActionLineSpacing190)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313
TrvActionParaStyles212
TrvActionLineSpacing190

```

Description

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

This actions changes the following properties of styles of the selected paragraphs (RVStyle.ParaStyles[]):

- LineSpacingType := *rvIsPercent*
- LineSpacing := 200

1.3.4.21 TrvActionParaBiDi

TrvActionParaBiDi is a base class for the actions changing the default text flow direction in the selected paragraphs.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionParaBiDi = class (TrvActionParaStyles (212))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action changes BiDiMode property for styles of the selected paragraphs (RVStyle.ParaStyles[].BiDiMode).

This action is not used directly. The following actions are inherited from it:

- TrvActionParaLTR ⁽²¹⁰⁾
- TrvActionParaRTL ⁽²¹²⁾

If you use these actions, it's recommended to set TCustomRichViewEdit.BiDiMode to *rvbdLeftToRight* or *rvbdRightToLeft*.

Text flow direction specified in text styles has higher priority than text flow direction specified in paragraph styles.

See also:

- TrvActionTextBiDi ⁽¹⁷²⁾
- TrvActionTextLTR ⁽¹⁷³⁾
- TrvActionTextRTL ⁽¹⁷⁴⁾

1.3.4.22 TrvActionParaBorder

TrvActionParaBorder is the action for "Paragraph Border and Background" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionParaBorder = class (TrvActionParaStyles212)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³
*TrvActionParaStyles*²¹²

Description

If `UserInterface`¹⁹⁶ = *False*, the action applies its properties to the selected paragraphs. The list of properties to apply is specified in `ValidProperties`¹⁹⁶.

If `UserInterface`¹⁹⁶ = *True*, the action uses a dialog window. The action performs the following steps:

1. assigns font attributes from the selected paragraphs to the action's properties
2. calls `OnShowingDialog`¹⁹⁶ event
3. displays a dialog window
4. applies changes made in the dialog window to the action's properties
5. applies the action's properties to the selected paragraphs.

This action allows changing attributes of paragraph border and background. For changing other paragraph attributes, see `TrvActionParagraph`²⁰⁰.

1.3.4.22.1 Properties

In TrvActionParaBorder

Background¹⁹⁵
 Border¹⁹⁵
 ■ `UserInterface`¹⁹⁶
 ValidProperties¹⁹⁶

Derived from TrvAction³¹³

■ `Control`³¹⁴

Derived from TrvCustomAction³³⁵

■ `Caption`³³⁶
 ■ `ControlPanel`³³⁷
 ■ `Disabled`³³⁷
 ■ `Hint`³³⁷

Derived from TAction

■ `AutoCheck`

- Caption
- Checked
 - DisableViewNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.22.1.1 TrvActionParaBorder.Background

Specifies background properties for applying to the selected paragraphs.

property Background: TRVBackgroundRect;

If `UserInterface(196)=False` and `rvpibBackground_***` in ValidProperties⁽¹⁹⁶⁾, the action applies selected subproperties of this property to the selected paragraphs.

If `UserInterface(196)=True`, the action displays a dialog (initialized with properties of the selected paragraphs); if the user changed values of subproperties of this property in the dialog, corresponding `rvpibBackground_***` values are included in ValidProperties⁽¹⁹⁶⁾, and the action applies them to the selected paragraphs.

This property is applied to the paragraph style's (RVStyle.ParaStyles[]) property of the same name.

Subproperties of this property are measured in the same units (pixels or twips) as properties of the target editor (i.e. in RVStyle.Units).

1.3.4.22.1.2 TrvActionParaBorder.Border

Specifies border properties for applying to the selected paragraphs.

property Border: TRVBorder;

If `UserInterface(196)=False` and `rvpibBorder_***` in ValidProperties⁽¹⁹⁶⁾, the action applies selected subproperties of this property to the selected paragraphs.

If `UserInterface(196)=True`, the action displays a dialog (initialized with properties of the selected paragraphs); if the user changed values of subproperties of this property in the dialog, corresponding `rvpibBorder_***` values are included in ValidProperties⁽¹⁹⁶⁾, and the action applies them to the selected paragraphs.

This property is applied to the paragraph style's (RVStyle.ParaStyles[]) property of the same name.

Subproperties of this property are measured in the same units (pixels or twips) as properties of the target editor (i.e. in RVStyle.Units).

1.3.4.22.1.3 TrvActionParaBorder.UserInterface

Allows working with or without user interaction.

property UserInterface: Boolean;

If UserInterface=*False*, the action applies its properties to the selected paragraphs.

If UserInterface=*True*, the action displays a dialog, assigns changes to the action's properties, and then applies these properties to the selected paragraphs.

Default value:

True

1.3.4.22.1.4 TrvActionParaBorder.ValidProperties

Defines a set of properties for applying to the selected paragraphs.

type

```
TRVParaInfoBorderProperty = (rvpibBackground_Color,
    rvpibBackground_BO_Left, rvpibBackground_BO_Top,
    rvpibBackground_BO_Right, rvpibBackground_BO_Bottom,
    rvpibBorder_Color, rvpibBorder_Style, rvpibBorder_Width,
    rvpibBorder_InternalWidth,
    rvpibBorder_BO_Left, rvpibBorder_BO_Top,
    rvpibBorder_BO_Right, rvpibBorder_BO_Bottom,
    rvpibBorder_Vis_Left, rvpibBorder_Vis_Top,
    rvpibBorder_Vis_Right, rvpibBorder_Vis_Bottom);
TRVParaInfoBorderProperties = set of TRVParaInfoBorderProperty;
```

property ValidProperties: TRVParaInfoBorderProperty;

If UserInterface⁽¹⁹⁶⁾=*False*, assign a subset of properties (for applying) to ValidProperties .

If UserInterface⁽¹⁹⁶⁾=*True*, this property is maintained automatically. After displaying a dialog, this property contains a set of properties corresponding to non-blank fields in the dialog.

1.3.4.22.2 Events

In TrvActionParaBorder

■ OnShowingDialog⁽¹⁹⁶⁾

1.3.4.22.2.1 TrvActionParaBorder.OnShowingDialog

Occurs before showing a dialog window.

property OnShowingDialog: TNotifyEvent;

This event occurs if UserInterface⁽¹⁹⁶⁾=*True*.

It is called when properties of the action are already assigned from attributes of the selected paragraphs of the target editor, but the dialog is not initialized yet.

In this event, you can modify properties of the action, so modified properties will be applied to the dialog.

You can use this event to initialize a dialog with predefined values, instead of taking them from selected paragraphs.

1.3.4.23 TrvActionParaBullets

TrvActionParaBullets is the action for "Bullets" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionParaBullets = class (TrvActionCustomParaListSwitcher(184))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomParaListSwitcher⁽¹⁸⁴⁾

Description

This action applies or removes bullets to the selected paragraphs.

This action does not introduce any new properties in addition to properties⁽¹⁸⁵⁾ of TrvActionCustomParaListSwitcher.

ListLevels⁽¹⁸⁶⁾ must not contains numbered levels.

When applying list, the action tries to reuse existing list styles with identical properties, if they exist.

1.3.4.24 TrvActionParaColor

TrvActionParaColor is the action for changing background color of the selected paragraphs.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionParaColor = class (TrvActionParaCustomColor(200))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction

*TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionCustomColor* ⁽³²⁰⁾*TrvActionParaCustomColor* ⁽²⁰⁰⁾

Description

This action does not introduce any new properties and events in addition to properties and events of *TrvActionCustomColor* ⁽³²⁰⁾. However, it published inherited *Opacity* ⁽³²⁰⁾ property.

On execution, this action changes *Background.Color* and *Opacity* properties of styles of the selected paragraphs (*RVStyle.ParaStyles[]*.*Background.Color* and *.Opacity*).

See also the list of color changing actions in the topic about *TrvActionCustomColor* ⁽³²⁰⁾.

1.3.4.25 TrvActionParaColorAndPadding

TrvActionParaColorAndPadding is the action for changing background color and padding of the selected paragraphs.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionParaColorAndPadding = class (TrvActionParaColor (197))
```

Hierarchy

*TObject**TPersistent**TComponent**TBasicAction**TContainedAction**TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionCustomColor* ⁽³²⁰⁾*TrvActionParaCustomColor* ⁽²⁰⁰⁾*TrvActionParaColor* ⁽¹⁹⁷⁾

Description

This action is reserved. It does not have *Caption* and *Hint* assigned by the localization procedure. If you use this action, assign its *Caption* and *Hint* yourself.

This action does not introduce any new events in addition to events of *TrvActionCustomColor* ⁽³²⁰⁾.

On execution, this action changes *Background.Color* and *Background.BorderOffsets* properties of styles of the selected paragraphs (*RVStyle.ParaStyles[]*.*Background.Color* and *RVStyle.ParaStyles[]*.*Background.BorderOffsets*). The specific (left/top/right/bottom) padding value specified in *Padding* ⁽¹⁹⁹⁾ property is assigned only if the corresponding property of *UsePadding* ⁽²⁰⁰⁾ is *True*. If the padding exceeds the paragraph's indents/spacing (*LeftIndent/SpaceBefore/RightIndent/SpaceAfter*), indents/spacing is increased.

Values of `Padding`⁽¹⁹⁹⁾ and `UsePadding`⁽²⁰⁰⁾ properties must be assigned by the programmer.

1.3.4.25.1 Properties

In `TrvActionParaColorAndPadding`

`Padding`⁽¹⁹⁹⁾

`UsePadding`⁽²⁰⁰⁾

Derived from `TrvActionCustomColor`⁽³²⁰⁾

■ `CallerControl`⁽³²¹⁾

■ `Color`⁽³²¹⁾

■ `UserInterface`⁽³²²⁾

Derived from `TrvAction`⁽³¹³⁾

■ `Control`⁽³¹⁴⁾

Derived from `TrvCustomAction`⁽³³⁵⁾

■ `Caption`⁽³³⁶⁾

■ `Disabled`⁽³³⁷⁾

■ `Hint`⁽³³⁷⁾

Derived from `TAction`

■ `AutoCheck`

■ `Caption`

■ `Checked`
`DisableIfNoHandler`

■ `Enabled`

■ `GroupIndex`

■ `HelpContext`

■ `HelpKeyword`

■ `HelpType`

■ `Hint`

■ `ImageIndex`

■ `Name`

■ `SecondaryShortCuts`

■ `ShortCut`

■ `Visible`

1.3.4.25.1.1 `TrvActionParaColorAndPadding.Padding`

Specifies the padding for applying to the selected paragraphs.

property `Padding`: `TRVRect`;

This property is applied to `Background.BorderOffsets` property of paragraph styles (`RVStyle.ParStyles[].Background.BorderOffsets`).

If `Padding.Left` exceeds `LeftIndent` property of the paragraph style, `LeftIndent` is increased.

If `Padding.Right` exceeds `RightIndent` property of the paragraph style, `RightIndent` is increased.

If `Padding.Top` exceeds `SpaceBefore` property of the paragraph style, `SpaceBefore` is increased.

If `Padding.Bottom` exceeds `SpaceAfter` property of the paragraph style, `SpaceAfter` is increased.

`UsePadding`⁽²⁰⁰⁾ property specifies which sides of padding are applied.

1.3.4.25.1.2 TrvActionParaColorAndPadding.UsePadding

Specifies sides of paragraphs for applying `Padding`⁽¹⁹⁹⁾.

property `UsePadding`: `TRVBooleanRect`;

1.3.4.26 TrvActionParaCustomColor

`TrvActionParaCustomColor` is a base class for the actions changing color properties of the selected paragraphs.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

`TrvActionParaCustomColor` = **class** (`TrvActionCustomColor`⁽³²⁰⁾)

Hierarchy

`TObject`
`TPersistent`
`TComponent`
`TBasicAction`
`TContainedAction`
`TCustomAction`
`TAction`
`TrvCustomAction`⁽³³⁵⁾
`TrvAction`⁽³¹³⁾
`TrvActionCustomColor`⁽³²⁰⁾
`TrvActionParaCustomColor`⁽²⁰⁰⁾

Description

This action does not introduce any new properties and events in addition to properties and events of `TrvActionCustomColor`⁽³²⁰⁾.

This action is not used directly. The following actions are inherited from it:

- `TrvActionParaColor`⁽¹⁹⁷⁾
- `TrvActionParaColorAndPadding`⁽¹⁹⁸⁾

1.3.4.27 TrvActionParagraph

`TrvActionParagraph` is the action for "Paragraph" command.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

`TrvActionParagraph` = **class** (`TrvActionParaStyles`⁽²¹²⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionParaStyles ⁽²¹²⁾

Description

If `UserInterface` ⁽²⁰⁷⁾ = `False`, the action applies its properties to the selected paragraphs. The list of properties to apply is specified in `ValidProperties` ⁽²⁰⁷⁾.

If `UserInterface` ⁽²⁰⁷⁾ = `True`, , the action uses a dialog window. The action performs the following steps:

1. assigns font attributes from the selected paragraphs to the action's properties
2. calls `OnShowingDialog` ⁽²⁰⁷⁾ event
3. displays a dialog window
4. applies changes made in the dialog window to the action's properties
5. applies the action's properties to the selected paragraphs.

This action allows changing almost all paragraph attributes, except for border and background (see `TrvActionParaBorder` ⁽¹⁹³⁾).

1.3.4.27.1 Properties

In `TrvActionParagraph`

`Alignment` ⁽²⁰²⁾
`DeleteAllTabs` ⁽²⁰²⁾
`FirstIndent` ⁽²⁰³⁾
`KeepLinesTogether` ⁽²⁰³⁾
`KeepWithNext` ⁽²⁰³⁾
`LastLineAlignment` ⁽²⁰⁴⁾
`LeftIndent` ⁽²⁰⁴⁾
`LineSpacing` ⁽²⁰⁴⁾
`LineSpacingType` ⁽²⁰⁵⁾
`OutlineLevel` ⁽²⁰⁵⁾
`RightIndent` ⁽²⁰⁵⁾
`SpaceAfter` ⁽²⁰⁵⁾
`SpaceBefore` ⁽²⁰⁶⁾
`Tabs` ⁽²⁰⁶⁾
`TabsToDelete` ⁽²⁰⁶⁾
■ `UserInterface` ⁽²⁰⁷⁾
`ValidProperties` ⁽²⁰⁷⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.27.1.1 TrvActionParagraph.Alignment

Specifies the alignment for applying to the selected paragraphs.

property Alignment: TRVAlignment;

If UserInterface ⁽²⁰⁷⁾ = *False* and *rvpimAlignment* in ValidProperties ⁽²⁰⁷⁾, the action applies this property to the selected paragraphs.

If UserInterface ⁽²⁰⁷⁾ = *True*, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, *rvpimAlignment* is included in ValidProperties ⁽²⁰⁷⁾, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (RVStyle.ParaStyles[]) property of the same name.

1.3.4.27.1.2 TrvActionParagraph.DeleteAllTabs

Instructs to delete all tab-stop positions from styles of the selected paragraphs.

property DeleteAllTabs: Boolean;

If *True*, all existing tab-stop positions are deleted from styles of the selected paragraphs.

If *False*, only tab-stops specified in TabsToDelete ⁽²⁰⁶⁾ are deleted.

After that, tab-stops from Tabs⁽²⁰⁶⁾ are added.

All these properties are applied only if ValidProperties⁽²⁰⁷⁾ contains *rvpimTabs*.

If UserInterface⁽²⁰⁷⁾=*True*, you should assign these properties yourself.

If UserInterface⁽²⁰⁷⁾=*False*, these properties are defined by the user in a dialog.

1.3.4.27.1.3 TrvActionParagraph.FirstIndent

Specifies the first line indent for applying to the selected paragraphs.

property FirstIndent: TRVStyleLength;

If UserInterface⁽²⁰⁷⁾=*False* and *rvpimFirstIndent* in ValidProperties⁽²⁰⁷⁾, the action applies this property to the selected paragraphs.

If UserInterface⁽²⁰⁷⁾=*True*, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, *rvpimFirstIndent* is included in ValidProperties⁽²⁰⁷⁾, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (RVStyle.ParaStyles[]) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in RVStyle.Units).

1.3.4.27.1.4 TrvActionParagraph.KeepLinesTogether

Specifies the keep-lines-together option for applying to the selected paragraphs.

property KeepLinesTogether: Boolean;

If UserInterface⁽²⁰⁷⁾=*False* and *rvpimKeepLinesTogether* in ValidProperties⁽²⁰⁷⁾, the action applies this property to the selected paragraphs.

If UserInterface⁽²⁰⁷⁾=*True*, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, *rvpimKeepLinesTogether* is included in ValidProperties⁽²⁰⁷⁾, and the action applies this property to the selected paragraphs.

This option adds/removes *rvpaoKeepLinesTogether* in the Options property of styles of the selected paragraphs (RVStyle.ParaStyles[]).Options).

1.3.4.27.1.5 TrvActionParagraph.KeepWithNext

Specifies the keep-with-next option for applying to the selected paragraphs.

property KeepWithNext: Boolean;

If UserInterface⁽²⁰⁷⁾=*False* and *rvpimKeepWithNext* in ValidProperties⁽²⁰⁷⁾, the action applies this property to the selected paragraphs.

If UserInterface⁽²⁰⁷⁾=*True*, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, *rvpimKeepWithNext* is included in ValidProperties⁽²⁰⁷⁾, and the action applies this property to the selected paragraphs.

This option adds/removes *rvpaoKeepWithNext* in the Options property of styles of the selected paragraphs (RVStyle.ParaStyles[]).Options).

1.3.4.27.1.6 TrvActionParagraph.LastLineAlignment

Specifies the alignment for applying to the last line of selected justified and distributed paragraphs.

property LastLineAlignment: TRVLastLineAlignment;

If `UserInterface(207) = False` and `rvpimLastLineAlignment` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207) = True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimLastLineAlignment` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

1.3.4.27.1.7 TrvActionParagraph.LeftIndent

Specifies the left indent for applying to the selected paragraphs.

property LeftIndent: TRVStyleLength;

If `UserInterface(207) = False` and `rvpimLeftIndent` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207) = True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimLeftIndent` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.8 TrvActionParagraph.LineSpacing

Specifies the line spacing value for applying to the selected paragraphs.

property LineSpacing: TRVLineSpacingValue;

If `UserInterface(207) = False` and `rvpimLineSpacing` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207) = True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimLineSpacing` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

Meaning of this property depends on the value of `LineSpacingType(205)` property. These two properties are applied together.

If `LineSpacingType(205) <> rvsPercent`, this value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.9 TrvActionParagraph.LineSpacingType

Specifies the line spacing type for applying to the selected paragraphs.

property LineSpacingType: TRVLineSpacingType;

If `UserInterface(207)=False` and `rvpimLineSpacing` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207)=True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimLineSpacing` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

This property is applied together with `LineSpacing(204)` property.

1.3.4.27.1.10 TrvActionParagraph.OutlineLevel

Specifies the line spacing type for applying to the selected paragraphs.

property OutlineLevel: Integer;

If `UserInterface(207)=False` and `rvpimOutlineLevel` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207)=True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimOutlineLevel` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

1.3.4.27.1.11 TrvActionParagraph.RightIndent

Specifies the right indent for applying to the selected paragraphs.

property RightIndent: TRVStyleLength;

If `UserInterface(207)=False` and `rvpimRightIndent` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207)=True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimRightIndent` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.12 TrvActionParagraph.SpaceAfter

Specifies spacing for applying to the bottom side of the selected paragraphs.

property SpaceAfter: TRVStyleLength;

If `UserInterface(207)=False` and `rvpimSpaceAfter` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207) = True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimSpaceAfter` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.13 TrvActionParagraph.SpaceBefore

Specifies spacing for applying to the top side of the selected paragraphs.

property `SpaceBefore`: `TRVStyleLength`;

If `UserInterface(207) = False` and `rvpimSpaceBefore` in `ValidProperties(207)`, the action applies this property to the selected paragraphs.

If `UserInterface(207) = True`, the action displays a paragraph dialog (initialized with properties of the selected paragraphs); if the user changed value of this property in the dialog, `rvpimSpaceBefore` is included in `ValidProperties(207)`, and the action applies this property to the selected paragraphs.

This property is applied to the paragraph style's (`RVStyle.ParaStyles[]`) property of the same name.

This value is measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.14 TrvActionParagraph.Tabs

Contains a list of tab-stops for adding to styles of the selected paragraphs.

property `Tabs`: `TRVTabInfos`;

Before applying this property, some (`TabsToDelete(206)`) or all (`DeleteAllTabs(202)`) tab-stops are deleted.

All these properties are applied only if `ValidProperties(207)` contains `rvpimTabs`.

If `UserInterface(207) = True`, you should assign these properties yourself.

If `UserInterface(207) = False`, these properties are defined by the user in a dialog.

Position properties of tabs are measured in the same units (pixels or twips) as properties of the target editor (i.e. in `RVStyle.Units`).

1.3.4.27.1.15 TrvActionParagraph.TabsToDelete

Contains a list of tab stop positions for deleting from styles of the selected paragraphs.

property `TabsToDelete`: `TRVIntegerList`;

If `DeleteAllTabs(202) = True`, all tab-stops are deleted instead.

After that, tab-stops from `Tabs(206)` are added.

All these properties are applied only if `ValidProperties(207)` contains `rvpimTabs`.

If `UserInterface(207) = True`, you should assign these properties yourself.

If `UserInterface(207) = False`, these properties are defined by the user in a dialog.

1.3.4.27.1.16 TrvActionParagraph.UserInterface

Allows working with or without user interaction.

property UserInterface: Boolean;

If UserInterface=False, the action applies its properties to the selected paragraphs.

If UserInterface=True, the action displays a paragraph dialog, assigns changes to the action's properties, and then applies these properties to the selected paragraphs.

Default value:

True

1.3.4.27.1.17 TrvActionParagraph.ValidProperties

Defines a set of properties for applying to the selected paragraphs.

type

```
TRVParaInfoMainProperty = (rvpimFirstIndent, rvpimLeftIndent,
    rvpimRightIndent, rvpimSpaceBefore, rvpimSpaceAfter,
    rvpimAlignment, rvpimLastLineAlignment,
    rvpimOutlineLevel, rvpimLineSpacing,
    rvpimKeepLinesTogether, rvpimKeepWithNext, rvpimTabs);
TRVParaInfoMainProperties = set of TRVParaInfoMainProperty;
```

property ValidProperties: TRVParaInfoMainProperties;

If UserInterface⁽²⁰⁷⁾=False, assign a subset of properties (for applying) to ValidProperties .

If UserInterface⁽²⁰⁷⁾=True, this property is maintained automatically. After displaying a paragraph dialog, this property contains a set of properties corresponding to non-blank fields in the dialog.

1.3.4.27.2 Events

In TrvActionParagraph

■ OnShowingDialog⁽²⁰⁷⁾

1.3.4.27.2.1 TrvActionParagraph.OnShowingDialog

Occurs before showing a dialog window.

property OnShowingDialog: TNotifyEvent;

This event occurs if UserInterface⁽²⁰⁷⁾=True.

It is called when properties of the action are already assigned from attributes of the selected paragraphs of the target editor, but the dialog is not initialized yet.

In this event, you can modify properties of the action, so modified properties will be applied to the dialog.

You can use this event to initialize a dialog with predefined values, instead of taking them from selected paragraphs.

1.3.4.28 TrvActionParaList

TrvActionParaList is the action for "Bullets and Numbering" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionParaList = class (TrvActionParaStyles(212))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionParaStyles⁽²¹²⁾

Description

The action displays a dialog window where the user can choose from the predefined list templates (may be after customizing them). If the caret is placed at the paragraph having bullets or numbering, the current list style takes the place of one of these templates, so the user can customize it. All templates have 9 list levels (maximal possible count that can be saved in RTF and DocX). For numbered styles, the user can choose to continue or to reset numbering, or to create a new list.

When applying bullets (without numbered levels), the action tries to reuse existing list styles with the same properties, if they exist. The action may apply properties to TrvActionParaBullets⁽¹⁹⁷⁾ actions.

When applying numbering (if at least one level is numbered), the action uses list styles around the selection (if they exist and have identical properties with the chosen properties). If it's not possible, the action tries to reuse unused list styles (if they exist and have identical properties with the chosen properties). Otherwise, a new list is created. If all levels are numbered, the action may apply properties to TrvActionParaNumbering⁽²¹¹⁾ actions.

1.3.4.28.1 Properties

In TrvActionParaList

ActionParaBullets⁽²¹⁰⁾
 ActionParaNumbering⁽²¹⁰⁾
 ■ IndentStep⁽²⁰⁹⁾
 UpdateAllActionsOnForm⁽²⁰⁹⁾

Derived from TrvAction⁽³¹³⁾

■ Control⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.4.28.1.1 TrvActionParaList.IndentStep

Defines the step for increasing LeftIndent and MarkerIndent of list levels.

property IndentStep: TRVStyleLength;

This value is measured in GetControlPanel ⁽³³⁸⁾.UnitsProgram ⁽⁵⁷⁾.

This property is used to adjust indents in the predefined set of lists (indents are adjusted when this action is executed for the first time).

Default value:

24

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

See also:

- TrvActionCustomParaListSwitcher ⁽¹⁸⁴⁾.IndentStep ⁽¹⁸⁶⁾
- TrvActionIndent ⁽¹⁸⁶⁾.IndentStep ⁽¹⁸⁸⁾

1.3.4.28.1.2 TrvActionParaList.UpdateAllActionsOnForm

Allows updating all TrvActionParaBullets ⁽¹⁹⁷⁾ and TrvActionParaNumbering ⁽²¹¹⁾ actions on the same form as this action.

property UpdateAllActionsOnForm: Boolean;

If this property is *True*:

- when the user applies a list without numbered levels, the action assigns these levels to ListLevels⁽¹⁸⁶⁾ of all TrvActionParaBullets⁽¹⁹⁷⁾ actions on the same form/datamodule.
- when the user applies a list with all numbered levels, the action assigns these levels to ListLevels⁽¹⁸⁶⁾ of all TrvActionParaNumbering⁽²¹¹⁾ actions on the same form/datamodule.

Default value:

True

See also properties:

- ActionParaNumbering⁽²¹⁰⁾
- ActionParaBullets⁽²¹⁰⁾

1.3.4.28.1.3 TrvActionParaList.ActionParaNumbering

Defines a link to the "Numbering" action.

property ActionParaNumbering: TrvActionParaNumbering⁽²¹¹⁾;

When the user applies a list with all numbered levels, the action assigns these levels to ListLevels⁽¹⁸⁶⁾ properties to the action specified in this link. If UpdateAllActionsOnForm⁽²⁰⁹⁾=*True*, these levels are assigned to ListLevels⁽¹⁸⁶⁾ of all TrvActionParaNumbering⁽²¹¹⁾ actions on the same form/datamodule.

1.3.4.28.1.4 TrvActionParaList.ActionParaBullets

Defines a link to the "Bullets" action.

property ActionParaBullets: TrvActionParaBullets⁽¹⁹⁷⁾;

When the user applies a list without numbered levels, the action assigns these levels to ListLevels⁽¹⁸⁶⁾ properties to the action specified in this link. If UpdateAllActionsOnForm⁽²⁰⁹⁾=*True*, these levels are assigned to ListLevels⁽¹⁸⁶⁾ of all TrvActionParaBullets⁽¹⁹⁷⁾ actions on the same form/datamodule.

1.3.4.29 TrvActionParaLTR

TrvActionParaLTR is the action for "Left to Right" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionParaLTR = **class** (TrvActionParaBiDi⁽¹⁹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionParaStyles⁽¹⁷⁵⁾

TrvActionParaBiDi⁽¹⁹³⁾

Description

This action changes the default text direction in the selected paragraphs to left-to-right. If you use this action, it's recommended to set `TCustomRichViewEdit.BiDiMode` to *rvbdLeftToRight* or *rvbdRightToLeft*.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of `TrvAction`.

This action changes `BiDiMode` property for styles of the selected paragraphs (`RVStyle.ParaStyles[].BiDiMode`) to *rvbdLeftToRight*.

This is a checkbox-like action, its `Checked` property is updated depending on the state of selection (or the current paragraph style).

See also:

- `TrvActionTextLTR`⁽¹⁷³⁾
- `TrvActionTextRTL`⁽¹⁷⁴⁾
- `TrvActionParaRTL`⁽²¹²⁾

1.3.4.30 TrvActionParaNumbering

`TrvActionParaNumbering` is the action for "Numbering" command.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionParaNumbering = class (TrvActionCustomParaListSwitcher(184))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionCustomParaListSwitcher(184)

```

Description

This action applies or removes numbering to the selected paragraphs.

This action does not introduce any new properties in addition to properties⁽¹⁸⁵⁾ of `TrvActionCustomParaListSwitcher`.

`ListLevels`⁽¹⁸⁶⁾ should contain numbered levels.

When applying lists, the action does the following:

1. if the paragraph before the selection, the selection or the paragraph after the selection contains a list style with the same properties as `ListLevels`⁽¹⁸⁶⁾, applies this list style;

2. otherwise, if RVStyle.ListStyles has unused list style with the same properties as ListLevels⁽¹⁸⁶⁾, applies this list style;
3. otherwise, creates a new list style, assigns ListLevels⁽¹⁸⁶⁾ to it, applies this list style.

See also:

- TrvActionParaBullets⁽¹⁹⁷⁾
- TrvActionParaList⁽²⁰⁸⁾

1.3.4.31 TrvActionParaRTL

TrvActionParaRTL is the action for "Right to Left" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionParaRTL = class (TrvActionParaBiDi(193))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionParaStyles⁽¹⁷⁵⁾
TrvActionParaBiDi⁽¹⁹³⁾

Description

This action changes the default text direction in the selected paragraphs to right-to-left. If you use this action, it's recommended to set TCustomRichViewEdit.BiDiMode to *rvbdLeftToRight* or *rvbdRightToLeft*.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action changes BiDiMode property for styles of the selected paragraphs (RVStyle.ParaStyles[].BiDiMode) to *rvbdRightToLeft*.

This is a checkbox-like action, its Checked property is updated depending on the state of selection (or the current paragraph style).

See also:

- TrvActionTextLTR⁽¹⁷³⁾
- TrvActionTextRTL⁽¹⁷⁴⁾
- TrvActionParaLTR⁽²¹⁰⁾

1.3.4.32 TrvActionParaStyles

TrvActionParaStyles is a base class for the actions applying changes to the selected paragraphs.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionParaStyles = class (TrvAction313)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³

Description

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionParagraph²⁰⁰
- TrvActionAlignment¹⁸⁰
- TrvActionParaBiDi¹⁹³
- TrvActionIndent¹⁸⁶
- TrvActionWordWrap²¹³
- TrvActionLineSpacing¹⁹⁰
- TrvActionParaBorder¹⁹³

1.3.4.33 TrvActionWordWrap

TrvActionWordWrap is the action for "Word Wrap" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionWordWrap = class (TrvActionParaStyles212)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³
*TrvActionParaStyles*¹⁷⁵

Description

This action turns word wrapping on/off for the selected paragraphs. It excludes/includes *rvpaoNoWrap* from the Options property of styles of the selected paragraphs (RVStyle.ParaStyles[.Options).

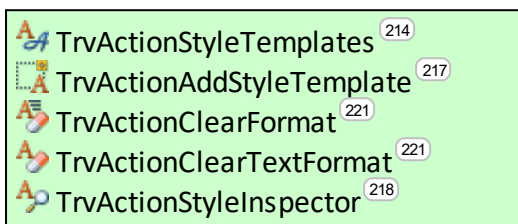
This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

This is a checkbox-like action, its Checked property is updated depending on the current paragraph style (or selection).

1.3.5 Styles

Styles Actions

This group of actions includes commands for managing and applying named styles of text and paragraphs.



1.3.5.1 TrvActionStyleTemplates

TrvActionStyleTemplates is the action for "Format | Styles" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionStyleTemplates = class (TrvAction313)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313

```

Description

This action is enabled only if Editor.UseStyleTemplates=True.

The action shows a dialog window allowing to manage style templates: to add, to delete, to modify properties, to import (from the special format, DocX, RTF, HTML, and RVF files), to export (to the special format).

When shown initially, the dialog window displays a style template of the current text in the editor.

You can define initial properties of standard style templates using `StandardStyleTemplates`⁽²¹⁶⁾ property.

You can customize the appearance of the dialog window using `Images`⁽²¹⁶⁾, `TextStyleImageIndex`, `ParaStyleImageIndex`, `ParaTextStyleImageIndex`⁽²¹⁷⁾ properties.

By default, the style import and export dialogs use localized names of RVF files (*.rvf) and of files containing style templates (*.rvst). If you assign `GetControlPanel`⁽³³⁸⁾.RVFLocalizable⁽⁵⁴⁾, you can change them in the control panel properties: `RVFFilter`⁽⁵⁴⁾, `RVStylesFilter`, `RVStylesExt`⁽⁵⁵⁾.

This action is used by other actions:

- `TrvActionAddStyleTemplate`⁽²¹⁷⁾ (to open a dialog window)
- `TrvActionInsertHyperlink`⁽²³⁴⁾ (to get default properties of "Hyperlink" style template)

1.3.5.1.1 Properties

In TrvActionStyleTemplates

- `Images`⁽²¹⁶⁾
- `ParaStyleImageIndex`⁽²¹⁷⁾
- `ParaTextStyleImageIndex`⁽²¹⁷⁾
- `StandardStyleTemplates`⁽²¹⁶⁾
- `TextStyleImageIndex`⁽²¹⁷⁾

Derived from TrvAction⁽³¹³⁾

- `Control`⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- `Caption`⁽³³⁶⁾
- `ControlPanel`⁽³³⁷⁾
- `Disabled`⁽³³⁷⁾
- `Hint`⁽³³⁷⁾

Derived from TAction

- `AutoCheck`
- `Caption`
- `Checked`
- `DisableIfNoHandler`
- `Enabled`
- `GroupIndex`
- `HelpContext`
- `HelpKeyword`
- `HelpType`
- `Hint`
- `ImageIndex`
- `Name`
- `SecondaryShortCuts`
- `ShortCut`
- `Visible`

1.3.5.1.1.1 TrvActionStyleTemplates.Images

Images for icons in the styles dialog window.

property Images: TCustomImageList;

If **Images** is not assigned:

- the tree-view containing a list of styles uses standard 16-color icons;
- a description of styles is displayed without icons.

If **Images** is assigned:

- if at least one of TextStyleImageIndex, ParaStyleImageIndex, ParaTextStyleImageIndex⁽²¹⁷⁾ >=0, **Images** and these image indices are used in the tree-view; ImageIndex of this action is used for the root tree node;
- the component searches for the instances of TrvActionFontEx⁽¹⁵¹⁾, TrvActionParagraph⁽²⁰⁰⁾, TrvActionParaBorder⁽¹⁹³⁾, TrvActionInsertHyperlink⁽²³⁴⁾ actions on the same form; if found, **Images** and the image indices of these actions are used in a style description.

Images are used in the style import dialog window as well.

Generally, you should assign the same TImageList component as used for TActionList/TActionManager containing this action.

1.3.5.1.1.2 TrvActionStyleTemplates.StandardStyleTemplates

Contains a list of style templates used as samples of standard style templates.

property StandardStyleTemplates: TRVStyleTemplateCollection;

When an user adds a standard style template in the dialog, the action searches in **StandardStyleTemplates** for a style template having this name. If found, the action uses properties of the **StandardStyleTemplates**' item as initial values for the added style template. If not found, the action uses hard-coded values as initial properties.

The following names are used for standard styles: 'Normal', 'Normal Indent', 'No Spacing', 'heading 1'...'heading 9', 'List Paragraph', 'Hyperlink', 'Title', 'Subtitle', 'Emphasis', 'Subtle Emphasis', 'Intense Emphasis', 'Strong', 'Quote', 'Intense Quote', 'Subtle Reference', 'Intense Reference', 'Block Text', 'HTML Variable', 'HTML Code', 'HTML Acronym', 'HTML Definition', 'HTML Keyboard', 'HTML Sample', 'HTML Typewriter', 'HTML Preformatted', 'HTML Cite', 'header', 'footer', 'page number', 'endnote reference', 'footnote reference', 'endnote text', 'footnote text'. Names are case sensitive. These names are not shown to users, their localized versions are shown instead.

Values in this collection are measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property may also be used by TrvActionInsertHyperlink, when it adds "Hyperlink" style template.

See also:

TrvActionNew⁽¹⁰⁵⁾.StyleTemplates⁽¹⁰⁶⁾

TrvActionInsertHyperlink⁽²³⁴⁾.AutoAddHyperlinkStyleTemplate⁽²³⁷⁾.

1.3.5.1.1.3 TrvActionStyleTemplates.*StyleImageIndex

The properties define image indices for icons in the tree-view containing a list of styles.

property TextStyleImageIndex: Integer;

property ParaStyleImageIndex: Integer;

property ParaTextStyleImageIndex: Integer;

If Images⁽²¹⁶⁾ is assigned, and at least one of these properties ≥ 0 , the tree-view in the style management dialog window uses these images:

- **TextStyleImageIndex** for style templates having Kind=*rvstkText* (and a tree node used as a root of for such style templates);
- **ParaStyleImageIndex** for style templates having Kind=*rvstkPara*;
- **ParaTextStyleImageIndex** for style templates having Kind=*rvstkParaText*.

Additionally, ImageIndex of this action is used for the root tree node.

Default value:

-1

1.3.5.2 TrvActionAddStyleTemplate

TrvActionAddStyleTemplate is the action for "Format | Add Style..." command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionAddStyleTemplate = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action is enabled only if Editor.UseStyleTemplates=*True*.

The action shows a dialog window allowing to manage style templates. When shown initially, the dialog window has a new style template added; its properties are based on the text and paragraph attributes at the position of the caret.

This action can work only if it is linked to TrvActionStyleTemplates⁽²¹⁴⁾ action. It can be linked using ActionStyleTemplates⁽²¹⁸⁾ property; if this property is not assigned, the action searches for TrvActionStyleTemplates⁽²¹⁴⁾ action on the same form.

1.3.5.2.1 Properties

In TrvActionAddStyleTemplate

- ActionStyleTemplates ⁽²¹⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.5.2.1.1 TrvActionAddStyleTemplate.ActionStyleTemplates

Links this action to TrvActionStyleTemplates ⁽²¹⁴⁾ action.

property ActionStyleTemplates: TrvActionStyleTemplates ⁽²¹⁴⁾;

If this property is not assigned, the action searches for TrvActionStyleTemplates ⁽²¹⁴⁾ action on the same form. If not found, the action does nothing.

TrvActionStyleTemplates ⁽²¹⁴⁾ defines properties for the style management dialog window.

1.3.5.3 TrvActionStyleInspector

TrvActionStyleInspector is the action for "Format | Style Inspector" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCustomInfoWindow = class (TrvAction (313));
TrvActionStyleInspector = class (TrvActionCustomInfoWindow);
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action shows/hides a non-modal window containing information about text and paragraph styles and attributes at the position of the caret.

Assign Control ⁽³¹⁴⁾ property of this action, and the info window will be updated automatically according to changes in Control ⁽³¹⁴⁾. When the action is executed for some editor, the action automatically assign this editor to Control ⁽³¹⁴⁾.

You can customize the appearance of the info window using Images ⁽²²⁰⁾, FontImageIndex and ParaImageIndex ⁽²²⁰⁾ properties.

1.3.5.3.1 Properties

In TrvActionStyleInspector

- FontImageIndex ⁽²²⁰⁾
- Images ⁽²²⁰⁾
- ParaImageIndex ⁽²²⁰⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType

- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.5.3.1.1 TrvActionStyleInspector.Images

Images for icons used in a description of text and paragraph attributes.

property Images: TCustomImageList;

This image list must contain two images: text and paragraph icons. Their indices should be specified in FontImageIndex and ParaImageIndex⁽²²⁰⁾ properties.

1.3.5.3.1.2 TrvActionStyleInspector.FontImageIndex, ParaImageIndex

Indices in Images⁽²²⁰⁾.

property FontImageIndex: Integer;

property ParaImageIndex: Integer;

FontImageIndex specifies the index of "font" icon.

ParaImageIndex specifies the index of "paragraph" icon.

Default value:

-1

1.3.5.3.2 Methods

In TrvActionStyleInspector

UpdateInfo⁽²²⁰⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.5.3.2.1 TrvActionStyleInspector.UpdateInfo

Refresh a content of the info window.

procedure UpdateInfo(OnlyIfVisible: Boolean = True);

Normally, the action updates the window when necessary.

Call this method when GetControlPanel⁽³³⁸⁾.Language⁽⁵³⁾ is changed.

1.3.5.3.3 Events

Inherited from TrvActionCustomInfoWindow

■ OnShowing⁽²²¹⁾

1.3.5.3.3.1 TrvActionCustomInfoWindow.OnShowing

Occurs before showing the info window.

property OnShowing: TRVShowFormEvent⁽³⁹⁰⁾;

Example (embedding the info window in the main window TfrmMain, at the right side):

```
procedure TfrmMain.rvActionStyleInspector1Showing(Sender: TrvAction(313);
    Form: TForm);
begin
    Form.Align := alRight;
    Form.Parent := Self;
end;
```

1.3.5.4 TrvActionClearFormat

TrvActionClearFormat is the action for "Format | Clear Format" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionClearFormat = class (TrvAction(313))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
```

Description

This action resets text and/or paragraph attributes in the selected fragment by calling Editor.ApplyStyleTemplate(-1).

The results depend on the selection. The action clears the paragraph format in the following cases:

- the selection is empty
- the selection includes the whole paragraph completely
- the selection includes several paragraphs.

If the selection is not empty, the action clears its text format.

This action can be used even if Editor.UseStyleTemplates=*False*. In this case, it simply applies the 0th text and the 0th paragraph styles.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.5.5 TrvActionClearTextFormat

TrvActionClearFormat is the action for "Format | Clear Text Format" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionClearTextFormat = class (TrvAction313)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³

Description

This action resets text attributes in the selected fragment by calling `Editor.ApplyTextStyleTemplate(-1, True)`.

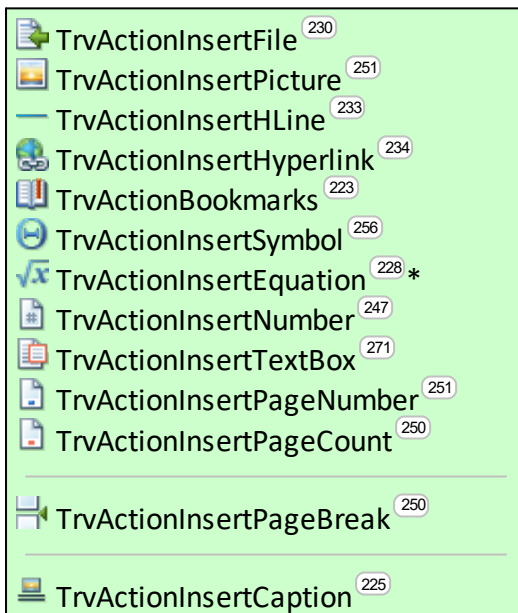
This action can be used even if `Editor.UseStyleTemplates=False`. In this case, it simply applies the 0th text style.

This action does not introduce any new properties in addition to properties³¹⁴ of `TrvAction`.

1.3.6 Insert

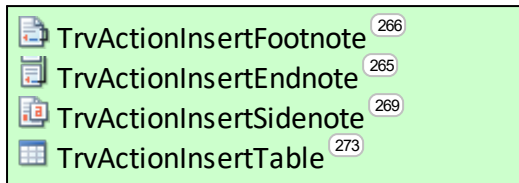
Insertion Actions

This group of actions includes commands to insert objects in the caret position, bookmark insertion and management.



* the action is not included in RichViewActions directly, it is included in a separate package.

See also



1.3.6.1 TrvActionBookmarks

TrvActionBookmarks is the action for "Insert Bookmark..." command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionBookmarks = class (TrvAction (313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action displays a dialog window allowing to:

- insert a new checkpoint (bookmark) at the current position in the target editor
- delete existing checkpoints
- move the caret to an existing checkpoint

This action allows in bookmark names:

- alphanumeric characters;
- ' _ ';
- characters included in AllowedCharacters ⁽²²⁴⁾.

If you use this action with Report Workshop, assign '{}' to AllowedCharacters ⁽²²⁴⁾.

1.3.6.1.1 Properties

In TrvActionBookmarks

- AllowedCharacters ⁽²²⁴⁾
- ScrollToCenter ⁽²²⁴⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.1.1.1 TrvActionBookmarks.AllowedCharacters

Specifies additional characters allowed in names of bookmarks.

property AllowedCharacters: TRVUnicodeString;

The action allows alphanumeric characters and underscores in bookmark names. To allow additional characters, include them in this property.

For example, if you use this action with Report Workshop, assign '{}' to allow field codes in bookmark names.

Initial value:

'' (empty string)

1.3.6.1.1.2 TrvActionBookmarks.ScrollToCenter

Specifies how the editor window is scrolled to show a checkpoint.

property ScrollToCenter: Boolean;

This property affects the behavior of "Go to" button in the bookmark dialog.

If *True*, the action scrolls the editor window to show the bookmarked item in the middle.

If *False*, the action scrolls the editor window to show the bookmarked item at the top.

Default value:

True

See also:

- `TrvActionInsertHyperlink(234).ScrollToCenter(224)`

1.3.6.2 TrvActionInsertCaption

`TrvActionInsertCaption` is the action for "Insert Object Caption..." command.

Unit `RichViewActions(94)`;

Syntax

```
TrvActionInsertCaption = class (TrvActionInsertNumSequence(248))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionInsertNumSequence(248)

```

Description

By default, this action allows inserting captions for pictures, "hot pictures" and tables. You can allow captions for additional items using `OnCanApply` event.

This action displays a dialog window allowing to enter caption for the object at the caret position.

The caption consists of the following parts:

- label
- "numbered sequence" item (name of its sequence=label)
- ' ' string
- used-defined text

Example: *Figure 1. Population dynamics of green zebras*

If `UseStyleTemplates=True` for the target editor, this caption is formatted using 'caption' style template. Otherwise, the current text style is used.

The following caption properties are supported:

- label (equal to a sequence name);
- numbering type (decimal, lower Roman, etc.);
- exclude label or not?
- caption text
- insert above or below?

If `UseDefaults(249)=True`, the dialog is initialized with a label 'Figure' (localized string), a decimal numbering type, `ExcludeLabel(226)` and `InsertAbove(227)` properties.

When the user clicks the "Insert" button, a caption is inserted above or below the selected object, `UseDefaults(249)` is reset to `False`, and chosen values are stored in `SeqName(249)`, `NumberType(249)`,

ExcludeLabel⁽²²⁶⁾ and InsertAbove⁽²²⁷⁾ properties (they are used to initialize a dialog when it will be displayed for the next time).

1.3.6.2.1 Properties

In TrvActionInsertCaption

ExcludeLabel⁽²²⁶⁾
InsertAbove⁽²²⁷⁾

Derived from TrvActionInsertNumSequence⁽²⁴⁸⁾

NumberType⁽²⁴⁹⁾
SeqName⁽²⁴⁹⁾
UseDefaults⁽²⁴⁹⁾

Derived from TrvAction⁽³¹³⁾

■ Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

■ Caption⁽³³⁶⁾
■ ControlPanel⁽³³⁷⁾
■ Disabled⁽³³⁷⁾
■ Hint⁽³³⁷⁾

Derived from TAction

■ AutoCheck
■ Caption
■ Checked
 DisableIfNoHandler
■ Enabled
■ GroupIndex
■ HelpContext
■ HelpKeyword
■ HelpType
■ Hint
■ ImageIndex
■ Name
■ SecondaryShortCuts
■ ShortCut
■ Visible

1.3.6.2.1.1 TrvActionInsertCaption.ExcludeLabel

Specifies whether a label is included in a caption text.

property ExcludeLabel: Boolean;

This property is used to initialize a dialog. When a dialog is applied, this property is updated automatically.

Example of caption if `ExcludeLabel=False`:

Table 1. Aging Statistics

Example of caption if `ExcludeLabel=True`:

1. Aging Statistics

Initial value:

False

1.3.6.2.1.2 TrvActionInsertCaption.InsertAbove

Specifies where a caption is inserted: above or below the object.

property `InsertAbove: Boolean;`

This property is used to initialize a dialog. When a dialog is applied, this property is updated automatically.

Initial value:

False

1.3.6.3 TrvActionInsertCustomPageNumber

`TrvActionInsertCustomPageNumber` is a base class for actions inserting page numbers and page counts.

Unit `RichViewActions` ⁽⁹⁴⁾;

Syntax

`TrvActionInsertCustomPageNumber = class (TrvAction` ⁽³¹³⁾ `)`

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action is not used directly. It is a parent for the following actions:

- `TrvActionInsertPageNumber` ⁽²⁵¹⁾
- `TrvActionInsertPageCount` ⁽²⁵⁰⁾

1.3.6.3.1 Properties

In `TrvActionInsertCustomPageNumber`

`NumberType` ⁽²²⁸⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

■ AutoCheck

■ Caption

■ Checked

DisableIfNoHandler

■ Enabled

■ GroupIndex

■ HelpContext

■ HelpKeyword

■ HelpType

■ Hint

■ ImageIndex

■ Name

■ SecondaryShortCuts

■ ShortCut

■ Visible

1.3.6.3.1.1 TrvActionInsertCustomPageNumber.NumberType

A type of numbering for the inserted item (decimal, lower Roman, etc.)

property NumberType: TRVPageNumberType;

Default value

rvpntDecimal

1.3.6.4 TrvActionInsertEquation

TrvActionInsertEquation is the action for "Insert Equation..." command.

Unit RVMathActions;

Syntax

TrvActionInsertEquation = **class** (TrvAction ⁽³¹³⁾)

Hierarchy

TObject

TPersistent

TComponent

TBasicAction

TContainedAction

TCustomAction

TAction

TrvCustomAction ⁽³³⁵⁾

TrvAction ⁽³¹³⁾

Description

This action is not included in RichViewActions directly, it is included in a separate package.

This item uses Adit Math Engine by Denis Sletkov (Adit Software): ***aditsoftware.com***

Units of Adit Math Engine are included in TRichView installation (in Math folder). They are covered by **MPL 2.0** with the following addition restrictions:

Adit Math Engine cannot be used in any E-learning/Assessment/Testing/Math software (Freeware or Shareware) or outside TRichView engine without our [Adit Software] written permission.

This action inserts a mathematical equation (TRVMathItemInfo object) in the caret position. The user can edit equation before insertion, if *UserInterface* ⁽²³⁰⁾ = *True*.

The dialog window allows editing not only property of the new equation, but default properties (font name, size and color) of all equations in the current document.

1.3.6.4.1 Properties

In TrvActionInsertEquation

- Expression ⁽²³⁰⁾
- UserInterface ⁽²³⁰⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType

- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.4.1.1 TrvActionInsertEquation.UserInterface

Specifies whether the action shows a dialog window for editing equation before the insertion.

property `UserInterface: Boolean;`

If **UserInterface** = *False*, the action inserts a new equation, assigning Expression⁽²³⁰⁾ to its text.

If **UserInterface** = *False*, the action displays a dialog for editing properties of the new equation, initializing it with Expression⁽²³⁰⁾.

Initial value:

True

1.3.6.4.1.2 TrvActionInsertEquation.Expression

Mathematical expression to insert.

property `Expression: TRVUnicodeString;`

This is a mathematical expression in LaTeX-like format.

Initial value:

'a+b=c'

See also

- UserInterface⁽²³⁰⁾

1.3.6.5 TrvActionInsertFile

TrvActionInsertFile is the action for "Insert | File" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

`TrvActionInsertFile = class (TrvActionCustomFileIO(98))`

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomIO⁽¹⁰⁰⁾

TrvActionCustomFileIO ⁽⁹⁸⁾

Description

This action displays a file opening dialog and inserts the chosen file at the caret position.

In addition to formats supported by *TrvActionOpen* ⁽¹⁰⁸⁾ action, this action supports formats supported by Microsoft Office file import converters.

1.3.6.5.1 Properties

In *TrvActionInsertFile*

■ Filter ⁽²³²⁾

Derived from *TrvActionCustomFileIO* ⁽⁹⁸⁾

■ CustomFilter ⁽¹⁰⁰⁾

Derived from *TrvActionCustomIO* ⁽¹⁰⁰⁾

■ AutoUpdateFileName ⁽¹⁰¹⁾

■ DialogTitle ⁽¹⁰²⁾

 FileName ⁽¹⁰²⁾

■ InitialDir ⁽¹⁰²⁾

Derived from *TrvAction* ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from *TrvCustomAction* ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from *TAction*

■ AutoCheck

■ Caption

■ Checked

 DisableIfNoHandler

■ Enabled

■ GroupIndex

■ HelpContext

■ HelpKeyword

■ HelpType

■ Hint

■ ImageIndex

■ Name

■ SecondaryShortCuts

■ ShortCut

■ Visible

1.3.6.5.1.1 TrvActionInsertFile.Filter

Defines a set of file formats appearing in the file opening dialog.

property Filter: TrvFileImportFilterSet⁽³⁸⁹⁾;

If *ffiCustom* is included, formats listed in the CustomFilter⁽¹⁰⁰⁾ property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel⁽³³⁸⁾.RVFFilter⁽⁵⁴⁾.

Default value:

all formats

1.3.6.5.2 Methods

In TrvActionInsertFile

InsertFile⁽²³²⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.6.5.2.1 TrvActionInsertFile.InsertFile

Inserts the specified file.

```
function InsertFile(rve: TCustomRichViewEdit;
  const FileName: TRVUnicodeString;
  FileFormat: TrvFileImportFilter(389);
  ConverterOrCustomIndex: Integer=0;
  rvc: TRVOfficeConverter=nil;
  Silent: Boolean = False): Boolean;
```

Parameters:

rve – editor where to insert file.

FileName – name of the file to insert.

FileFormat – format of this file.

ConverterCustomFilterIndex used only if **FileFormat** is *ffiCustom* or *ffiOfficeConverters*.

If **FileFormat** = *ffiCustom*, **ConverterCustomFilterIndex** identifies a custom file format. Custom formats are numbered from 1 in the order they are listed in the CustomFilter⁽¹⁰⁰⁾ property.

If **FileFormat** = *ffiOfficeConverters*, **ConverterCustomFilterIndex** identifies an import format, as an index in **rvc.ImportConverters**.

rvc is used if **FileFormat** = *ffiOfficeConverters*.

Silent – if **False**, the action displays an error message if the insertion was not successful.

Return value:

True if the insertion was successful.

If the insertion failed, this method displays an error message.

This method does not change values of the action's properties.

1.3.6.6 TrvActionInsertHLine

TrvActionInsertHLine is the action for "Insert | Horizontal Line" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertHLine = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action inserts a horizontal line ("break" item type) in the caret position in TCustomRichViewEdit component.

The inserted line has the specified Color⁽²³⁴⁾ and Width⁽²³⁴⁾, and Style⁽²³⁴⁾.

1.3.6.6.1 Properties

In TrvActionInsertHLine

- Color⁽²³⁴⁾
- Style⁽²³⁴⁾
- Width⁽²³⁴⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption

- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.6.1.1 TrvActionInsertHLine.Color

Specifies the color for horizontal line.

property Color: TColor;

Default value:

clWindowText

1.3.6.6.1.2 TrvActionInsertHLine.Style

Specifies the style for horizontal line.

property Style: TRVBreakStyle;

Default value:

rvbsLine

1.3.6.6.1.3 TrvActionInsertHLine.Width

Specifies the width for horizontal line.

property Width: TRVStyleLenth;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value:

1

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.6.7 TrvActionInsertHyperlink

TrvActionInsertHyperlink is the action for "Insert | Hyperlink" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionInsertHyperlink = **class** (TrvActionTextStyles⁽¹⁷⁵⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTextStyles ⁽¹⁷⁵⁾

Description

This action can perform the following tasks:

- inserting a new text hyperlink, with the specified target and text;
- converting the selected text and (optionally) pictures to hyperlinks (with the specified target);
- removing hyperlink from the selected fragment.

When this action is executed, it displays a dialog for defining hyperlink attributes. But before the dialog is shown, the action adjusts the selection in *TCustomRichViewEdit*: if the selection includes a hyperlink partially (or if there is no selection but the caret is inside a link), this hyperlink is selected completely. If the selection is still empty, the action will insert a new hyperlink. Otherwise it will convert the selected fragment to hyperlinks (or will change targets for existing hyperlinks in the selection). If the user will enter an empty target, all links will be removed instead.

By default, the action stores targets of hyperlinks in items' tags. You can override this behavior using *OnApplyHyperlinkToItem* ⁽²⁴⁶⁾ and *OnGetHyperlinkTargetFromItem* ⁽²⁴⁶⁾ events.

Using style templates

Visual appearance of a hyperlink depends on whether style templates are used ⁽²⁰⁾.

If style templates are not used: the action allows defining text and background colors for hyperlinks (including "hover" colors). The chosen colors will be applied to all new links. When converting a text to a hypertext, properties listed in *ValidProperties* ⁽²⁴¹⁾ are applied to its style (and the original style is assigned to its *NextStyleNo* property).

When converting a hypertext to a text, the same set of properties are changed. But instead of properties of this action, properties of *RVStyle.TextStyles[0]* are applied (and -1 is assigned to *NextStyleNo* property).

The dialog window has a button invoking a new dialog for changing the link color.

If style templates are used: the action applies *StyleTemplateName* ⁽²⁴¹⁾ to hyperlinks. When converting a hypertext to a text, the action clears a link to style template.

The dialog window has a combo-box allowing to choose a style template. Only style templates having *Kind=rvstkText* are listed.

Methods

In addition to the operations performed on the execution, the action has several useful methods:

- *GetHyperlinkStyleNo* ⁽²⁴³⁾ returns the index of hypertext style for the given text style;
- *GetNormalStyleNo* ⁽²⁴⁴⁾ returns the index of non-hypertext style for the given text style;

- DetectURL⁽²⁴²⁾ analyses text to the left of the caret; if it is URL, converts it to hyperlink;
- TerminateHyperlink⁽²⁴⁵⁾ converts the current text style to non-hypertext, if the caret is at the end of hyperlink;
- GoToLink⁽²⁴⁴⁾ opens the hyperlink target in the external browser; for local links, it moves the caret to the target *checkpoint*.

1.3.6.7.1 Properties

In TrvActionInsertHyperlink ---

- ActionStyleTemplates⁽²³⁷⁾
- AutoAddHyperlinkStyleTemplate⁽²³⁷⁾
- BackColor⁽²³⁷⁾
- Color⁽²³⁸⁾
- Cursor⁽²³⁸⁾
- HoverBackColor⁽²³⁸⁾
- HoverColor⁽²³⁹⁾
- HoverEffects⁽²³⁹⁾
- HoverUnderlineColor⁽²³⁹⁾
- ScrollToCenter⁽²⁴⁰⁾
- SetToPictures⁽²⁴⁰⁾
- SpaceFiller⁽²⁴⁰⁾
- Style⁽²⁴⁰⁾
- StyleTemplateName⁽²⁴¹⁾
- UnderlineColor⁽²⁴¹⁾
- ValidProperties⁽²⁴¹⁾

Derived from TrvAction⁽³¹³⁾ ---

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾ ---

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction ---

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType

- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.7.1.1 TrvActionInsertHyperlink.ActionStyleTemplates

Links this action to TrvActionStyleTemplates⁽²¹⁴⁾ action.

property ActionStyleTemplates: TrvActionStyleTemplates⁽²¹⁴⁾;

This property is used only if style templates are used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=True).

If this link is not defined, the first found TrvActionStyleTemplates⁽²¹⁴⁾ action on the same form/datamodule is used.

See StyleTemplateName⁽²⁴¹⁾ for details.

1.3.6.7.1.2 TrvActionInsertHyperlink.AutoAddHyperlinkStyleTemplate

Defines whether the action may add "Hyperlink" style template.

property AutoAddHyperlinkStyleTemplate: Boolean;

This property is used only if style templates are used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=True).

See StyleTemplateName⁽²⁴¹⁾ for details.

Default value:

True

1.3.6.7.1.3 TrvActionInsertHyperlink.BackColor

Defines the background color for hyperlinks.

property BackColor: TColor;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=False). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if *rvhlBackColor* is included in ValidProperties⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

c/None

1.3.6.7.1.4 TrvActionInsertHyperlink.Color

Defines the text color for hyperlinks.

property Color: TColor;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=False). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if *rvhlColor* is included in ValidProperties⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clBlue

1.3.6.7.1.5 TrvActionInsertHyperlink.Cursor

Defines the hypertext cursor for the links.

property Cursor: TCursor;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=False). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if *rvhlCursor* is included in ValidProperties⁽²⁴¹⁾.

Default value:

crJump

1.3.6.7.1.6 TrvActionInsertHyperlink HoverBackColor

Defines the background color for hyperlinks under the mouse pointer.

property HoverBackColor: TColor;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=False). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if *rvhlHoverBackColor* is included in ValidProperties⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clNone (transparent)

1.3.6.7.1.7 TrvActionInsertHyperlink HoverColor

Defines the text color for hyperlinks under the mouse pointer.

property `HoverColor: TColor;`

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's `UseStyleTemplates=False`). If style templates are used, the action applies `StyleTemplateName`⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if `rvhlHoverColor` is included in `ValidProperties`⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

`c/Blue`

1.3.6.7.1.8 TrvActionInsertHyperlink HoverEffects

Defines the effects (such as underlining) for hyperlinks under the mouse pointer.

property `HoverEffects: TRVHoverEffects;`

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's `UseStyleTemplates=False`). If style templates are used, the action applies `StyleTemplateName`⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if `rvhlHoverEffects` is included in `ValidProperties`⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

`[]`

1.3.6.7.1.9 TrvActionInsertHyperlink HoverUnderlineColor

Defines the underline color for hyperlinks under the mouse pointer.

property `HoverUnderlineColor: TColor;`

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's `UseStyleTemplates=False`). If style templates are used, the action applies `StyleTemplateName`⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if `rvhlHoverUnderlineColor` is included in `ValidProperties`⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

`c/None` (instructs using `UnderlineColor`⁽²⁴¹⁾)

1.3.6.7.1.10 TrvActionInsertHyperlink.ScrollToCenter

Specifies how the editor window is scrolled to show a checkpoint.

property ScrollToCenter: Boolean;

This property affect the behavior of GoToLink⁽²⁴⁴⁾ method.

If *True*, the action scrolls the editor window to show the bookmarked item in the middle.

If *False*, the action scrolls the editor window to show the bookmarked item at the top.

Default value:

True

See also:

- TrvActionBookmarks⁽²²³⁾.ScrollToCenter⁽²²⁴⁾

1.3.6.7.1.11 TrvActionInsertHyperlink.SetToPictures

Specifies whether this action should convert pictures to hot-pictures and vice versa.

property SetToPictures: Boolean;

If this property is *False*, the action does nothing with the selected pictures.

Default value:

True

1.3.6.7.1.12 TrvActionInsertHyperlink.SpaceFiller

Specifies the string used for replacing space characters in hyperlink targets (by EncodeTarget⁽²⁴³⁾).

property SpaceFiller: TRVUnicodeString;

Default value:

' ' (space character, so nothing is replaced)

1.3.6.7.1.13 TrvActionInsertHyperlink.Style

Defines the font style for hyperlinks.

property Style: TFontStyles;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if *rvhlBold*, *rvhlItalic*, *rvhlUnderline*, or *rvhlStrikeout* are included in ValidProperties⁽²⁴¹⁾.

For example:

- if *rvhlBold* is included in ValidProperties⁽²⁴¹⁾ and *fsBold* is included in Style, hyperlinks will be bold;
- if *rvhlBold* is included in ValidProperties⁽²⁴¹⁾ and *fsBold* is not included in Style, hyperlinks will not be bold;
- if *rvhlBold* is not included in ValidProperties⁽²⁴¹⁾ (default), boldness is not changed.

Default value:

[*fsUnderline*]

1.3.6.7.1.14 TrvActionInsertHyperlink.StyleTemplateName

Defines the name of a style template for applying to hyperlinks.

property StyleTemplateName: TRVUnicodeString;

This property is used only if style templates are used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=True).

This property may be changed by an user in the dialog window.

This style template must be a text style template (Kind=rvskText), otherwise it will not be listed in a dialog window.

To get a style template for a hyperlink, the action performs the following steps:

1. The action searches in the target editor's Style.StyleTemplates for the style template named **StyleTemplateName**. If found, it is used. Otherwise,
2. The action searches for "Hyperlink" style template. If found, it is used. Otherwise,
3. If AutoAddHyperlinkStyleTemplate⁽²³⁷⁾=True, the action adds "Hyperlink" style template to the target editor's Style.StyleTemplates. Properties of this new style templates may be taken from the linked⁽²³⁷⁾ TrvActionStyleTemplates⁽²¹⁴⁾ action, if it has "Hyperlink" in its StandardStyleTemplates⁽²¹⁶⁾; otherwise, the action uses hard-coded default values.

Default value:

'Hyperlink'

1.3.6.7.1.15 TrvActionInsertHyperlink.UnderlineColor

Defines the underline color for hyperlinks.

property UnderlineColor: TColor;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's UseStyleTemplates=False). If style templates are used, the action applies StyleTemplateName⁽²⁴¹⁾ instead.

This property is used when converting text to hypertext, if rvhlHoverUnderlineColor is included in ValidProperties⁽²⁴¹⁾.

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

c/None (instructs using Color⁽²³⁸⁾)

1.3.6.7.1.16 TrvActionInsertHyperlink.ValidProperties

Defines properties distinguishing hyperlinks from normal text, if style templates are not used.

type

```
TRVHyperlinkProperty = (rvhlBold, rvhlItalic, rvhlUnderline,
    rvhlUnderlineColor, rvhlStrikeout, rvhlColor, rvhlBackColor,
    rvhlHoverColor, rvhlHoverBackColor, rvhlHoverUnderlineColor,
    rvhlHoverUnderline, rvhlCursor, rvhlJump);
TRVHyperlinkProperties = set of TRVHyperlinkProperty;
```

property ValidProperties: TRVHyperlinkProperties;

This property is used only if style templates are not used⁽²⁰⁾ (i.e. the target editor's `UseStyleTemplates=False`). If style templates are used, the action applies `StyleTemplateName`⁽²⁴¹⁾ instead.

When this action converts text to hypertext, properties of this action listed in this property are applied:

- `BackColor`⁽²³⁷⁾
- `Color`⁽²³⁸⁾
- `HoverBackColor`⁽²³⁸⁾
- `HoverColor`⁽²³⁹⁾
- `HoverEffects`⁽²³⁹⁾
- `HoverUnderlineColor`⁽²³⁹⁾
- `UnderlineColor`⁽²⁴¹⁾
- `Style`⁽²⁴⁰⁾

When this action converts text to non-hypertext, properties of `RVStyle.TextStyles[0]` listed in this property are applied.

Default value:

`[rvhlUnderline, rvhlColor, rvhlBackColor, rvhlHoverColor, rvhlHoverBackColor, rvhlCursor, rvhlJump]`
(hyperlink with the standard hyperlink cursor, blue and underlined)

1.3.6.7.2 Methods

In TrvActionInsertHyperlink

`DetectURL`⁽²⁴²⁾
`EncodeTarget`⁽²⁴³⁾
`GetHyperlinkStyleNo`⁽²⁴³⁾
`GetNormalStyleNo`⁽²⁴⁴⁾
`GoToLink`⁽²⁴⁴⁾
`SetTags`⁽²⁴⁵⁾
`TerminateHyperlink`⁽²⁴⁵⁾

Inherited from TrvCustomAction⁽³³⁵⁾

`GetControlPanel`⁽³³⁸⁾

1.3.6.7.2.1 TrvActionInsertHyperlink.DetectURL

Helps to detect URLs on typing in **rve**.

```
procedure DetectURL(rve: TCustomRichViewEdit);
```

This action checks the text to the left of the caret position, and if its a URL, converts it to a hyperlink.

The function does nothing of the text is already a hyperlink, or if the selection in **rve** is not empty.

The method uses `GetHyperlinkStyleNo`⁽²⁴³⁾ for this conversion.

Example [VCL]:

```
procedure TForm1.RichViewEdit1KeyPress(Sender: TObject; var Key: Char);  
begin
```

```

if Key in [#9, ' ', ',', ';'] then begin
    rvActionInsertHyperlink1.DetectURL(RichViewEdit1);
    rvActionInsertHyperlink1.TerminateHyperlink(245)(RichViewEdit1);
end;
end;

```

You should call this method:

- in OnKeyPress event (in OnUTF8KeyPress for Lazarus) when the user types tab, space, comma or semicolon characters
- in OnKeyDown event when the user pressed ENTER

1.3.6.7.2.2 TrvActionInsertHyperlink.EncodeTarget

Encodes hypertext target before assigning it to the item's tag.

```
function EncodeTarget(const Target: TRVUnicodeString): TRVUnicodeString;
```

The function returns the processed **Target** parameter.

It removes the line break characters (#13 and #10) and replaces the space characters with SpaceFiller⁽²⁴⁰⁾.

Normally, the returned value is the same as **Target** (because the action does not allow entering multiline targets, and SpaceFiller⁽²⁴⁰⁾=' ').

1.3.6.7.2.3 TrvActionInsertHyperlink.GetHyperlinkStyleNo

Returns the index of hypertext style for the given style.

```
function GetHyperlinkStyleNo(rve: TCustomRichViewEdit;
    StyleNo: Integer=-1): Integer;
```

If **StyleNo**=-1, the action returns the index of hypertext style for **rve.CurTextStyleNo**, otherwise for **StyleNo**.

The appearance of the returned style depends on **rve.UseStyleTemplates**.

If style templates are not used⁽²⁰⁾, the method returns the index of style (in **rve.Style.TextStyles**) having all properties of the specified style, except for the properties defined in ValidProperties⁽²⁴¹⁾. For these properties, values specified in properties of this action are used.

If style templates are used⁽²⁰⁾, the method returns the index of style (in **rve.Style.TextStyles**) having all properties of the specified style, by with the style template StyleTemplateName⁽²⁴¹⁾ applied.

If a style with the required properties already exists in **rve.Style.TextStyles**, its index is returned. Otherwise, a new style is created and its index is returned.

This method is used in DetectURL⁽²⁴²⁾ method, or when inserting/applying hyperlink. You can use it in other cases when you need a hypertext style index.

For example, in TCustomRichViewEdit.OnReadHyperlink event:

```

procedure TForm1.RichViewEdit1ReadHyperlink(Sender: TCustomRichView;
    const Target, Extras: TRVUnicodeString; DocFormat: TRVLoadFormat;
    var StyleNo: Integer; var ItemTag: TRVTag;
    var ItemName: TRVUnicodeString);
begin
    if DocFormat=rvlfURL then

```

```

StyleNo := rvActionInsertHyperlink1.GetHyperlinkStyleNo(
    RichViewEdit1);
ItemTag := rvActionInsertHyperlink1.EncodeTarget(243)(Target);
end;

```

See also methods:

- GetNormalStyleNo⁽²⁴⁴⁾

1.3.6.7.2.4 TrvActionInsertHyperlink.GetNormalStyleNo

Returns the index of non-hypertext style for the given style.

```

function GetNormalStyleNo(rve: TCustomRichViewEdit;
    StyleNo: Integer=-1): Integer;

```

If **StyleNo**=-1, the action returns the index of non-hypertext style for **rve.CurTextStyleNo**, otherwise for **StyleNo**.

The appearance of the returned style depends on **rve.UseStyleTemplates**.

If style templates are not used⁽²⁰⁾, the method returns the index of style (in **rve.Style.TextStyles**) having all properties of the specified style, except for the properties defined in **ValidProperties**⁽²⁴¹⁾. For these properties, values of properties of **rve.Style.TextStyles[0]** are used.

If style templates are used⁽²⁰⁾, the method returns the index of style (in **rve.Style.TextStyles**) having all properties of the specified style, by with link to a style template cleared.

If a style with the required properties already exists in **rve.Style.TextStyles**, its index is returned. Otherwise, a new style is created and its index is returned.

This method is used in **TerminateHyperlink**⁽²⁴⁵⁾ method, or when user entered an empty target in the dialog.

See also methods:

- GetHyperlinkStyleNo⁽²⁴³⁾

1.3.6.7.2.5 TrvActionInsertHyperlink.GoToLink

These methods execute the given hyperlink.

```

procedure GoToLink(rve: TCustomRichViewEdit; id: Integer);
procedure GoToLink(rve: TCustomRichViewEdit;
    const Target: TRVUnicodeString);

```

The first version of the method should be called in **rve.OnJump** event:

```

procedure TForm3.RichViewEdit1Jump(Sender: TObject; id: Integer);
begin
    rvActionInsertHyperlink1.GoToLink(RichViewEdit1, id);
end;

```

The second version of this method can be called from **rve.OnJump** or from any place of code.

If the link's target (stored in its tag) is started from '#', the action moves the caret to the *checkpoint* of the same name (but without '#'). If ScrollToCenter⁽²⁴⁰⁾ = True, the bookmarked item is centered in the editor.

Otherwise, the action opens the link's target in external application (using ShellExecute function).

On error (no such checkpoint, or ShellExecute returned an error code), an error message is displayed.

See also:

- GoToCheckpoint⁽³⁷⁹⁾ function

1.3.6.7.2.6 TrvActionInsertHyperlink.SetTags

Applies the action to the selection.

```
procedure SetTags(rve: TCustomRichViewEdit;  
  const Target: TRVUnicodeString);
```

Normally, calling this method is not necessary. It is called automatically when the action applies **Target** to the selection.

If **Target** is not empty, the action converts the selection to hyperlink(s).

If **Target** is empty, the action removes hyperlinks from the selection.

1.3.6.7.2.7 TrvActionInsertHyperlink.TerminateHyperlink

Helps to terminate hyperlinks when typing in **rve**.

```
procedure TerminateHyperlink(rve: TCustomRichViewEdit);
```

If the selection is empty, the current text item is a hyperlink, and the caret is at the end of this item, this function assigns the index non-hypertext style to the current text style (**rve.CurTextStyleNo**). GetNormalStyleNo⁽²⁴⁴⁾ is used.

Example [VCL]:

```
procedure TForm1.RichViewEdit1KeyPress(Sender: TObject; var Key: Char);  
begin  
  if Key in [#9, ' ', ',', ';'] then begin  
    rvActionInsertHyperlink1.DetectURL(242)(RichViewEdit1);  
    rvActionInsertHyperlink1.TerminateHyperlink(RichViewEdit1);  
  end;  
end;
```

When the user types something at the end of a hyperlink, new characters are added to the hyperlink's text. But when he/she types a space character (or tab, or comma, or Enter, or semicolon), the current text style becomes a non-hypertext, so this and subsequent characters will not be added to the hyperlink's text.

You should call this method:

- in OnKeyPress event (in OnUTF8KeyPress for Lazarus) when the user types tab, space, comma or semicolon characters
- in OnKeyDown event when the user pressed ENTER.

1.3.6.7.3 Events

In TrvActionInsertHyperlink

- OnApplyHyperlinkToItem⁽²⁴⁶⁾
- OnGetHyperlinkTargetFromItem⁽²⁴⁶⁾
- OnHyperlinkForm⁽²⁴⁶⁾

1.3.6.7.3.1 TrvActionInsertHyperlink.OnApplyHyperlinkToItem

Occurs when converting item of unknown type to/from hyperlink

type

```
TRVApplyHyperlinkToItemEvent = procedure (Sender: TObject;
  Editor: TCustomRichViewEdit; RVData: TCustomRVData; ItemNo: Integer;
  const Target: TRVUnicodeString) of object;
```

property OnApplyHyperlinkToItem: TRVApplyHyperlinkToItemEvent;

For text and pictures, the action applies hypertext (or removes hypertext) automatically.

For other types of items, this event occurs.

The item is defined by **Editor** and **ItemNo** (it is the **ItemNo**-th item in **Editor**, where **Editor** can be a main editor or cell inplace editor). **RVData** is **Editor.RVData**.

When converting to hypertext, **Target** is not empty.

When converting from hypertext, **Target** is empty.

In this event, you can perform an editing operation on this item.

1.3.6.7.3.2 TrvActionInsertHyperlink.OnGetHyperlinkTargetFromItem

Occurs when the action needs to get a hyperlink target from item of unknown type.

type

```
TRVGetHyperlinkTargetFromItem = procedure (Sender: TObject;
  Editor: TCustomRichViewEdit; RVData: TCustomRVData; ItemNo: Integer;
  var Target: TRVUnicodeString) of object;
```

property OnGetHyperlinkTargetFromItem: TRVGetHyperlinkTargetFromItem;

For text and pictures, the action gets targets automatically. For items of other types, this event occurs.

If you return empty **Target**, the action will treat this item as non-hypertext.

The item is defined by **Editor** and **ItemNo** (it is the **ItemNo**-th item in **Editor**, where **Editor** can be a main editor or cell inplace editor). **RVData** is **Editor.RVData**.

1.3.6.7.3.3 TrvActionInsertHyperlink.OnHyperlinkForm

Allows displaying your own hypertext dialog.

type

```
TRVHyperlinkFormEvent = procedure (Sender: TObject; InsertNew: Boolean;
  var Text, Target: TRVUnicodeString; var Proceed: Boolean) of object;
```

property OnHyperlinkForm: TRVHyperlinkFormEvent;

Use this event if you want to replace the default dialog used in this action.

Input parameters:

If **InsertNew**=*True*, this event is called for inserting a new hyperlink.

Text – the selected text, valid only if **InsertNew**=*False*.

Target – hypertext target of selected text, use only if **InsertNew**=*False*. It is not empty only in case when the selection contains only hyperlinks with equal targets.

Proceed = *True*.

Output parameters:

Text – text for new hyperlink, used only if **InsertNew**=*True*.

Target – hyperlink target. You can return empty string only if **InsertNew**=*False* (to remove hyperlinks from the selection)

Proceed must be *True* if the user pressed "OK", and *False* if the user pressed "Cancel".

1.3.6.8 TrvActionInsertNumber

TrvActionInsertNumber is the action for "Insert Number..." command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionInsertNumber = **class** (TrvActionInsertNumSequence⁽²⁴⁸⁾)

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction
 TContainedAction
 TCustomAction
 TAction
 TrvCustomAction⁽³³⁵⁾
 TrvAction⁽³¹³⁾
 TrvActionInsertNumSequence⁽²⁴⁸⁾

Description

This action displays a dialog window allowing to enter properties of an item to insert.

The following properties are supported:

- sequence name (identifier);
- numbering type (decimal, lower Roman, etc.);
- continue numbering / start from the specified value.

If UseDefaults⁽²⁴⁹⁾=*True*, the dialog is initialized with a sequence name 'Numbering' (localized string) and a decimal numbering type.

When the user clicks the "Insert" button, a "numbered sequence" item is inserted in the caret position, `UseDefaults`⁽²⁴⁹⁾ is reset to *False*, and chosen values are stored in `SeqName`⁽²⁴⁹⁾ and `NumberType`⁽²⁴⁹⁾ properties (they are used to initialize a dialog when it will be displayed for the next time).

1.3.6.9 TrvActionInsertNumSequence

`TrvActionInsertNumSequence` is a base class for the actions inserting a "numbered sequence" item.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionInsertNumSequence = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action is not used directly.

The following actions are inherited from it:

- `TrvActionInsertNumber`
- `TrvActionInsertCaption`

1.3.6.9.1 Properties

In `TrvActionInsertNumSequence`

`NumberType`⁽²⁴⁹⁾
`SeqName`⁽²⁴⁹⁾
`UseDefaults`⁽²⁴⁹⁾

Derived from `TrvAction`⁽³¹³⁾

■ `Control`⁽³¹⁴⁾

Derived from `TrvCustomAction`⁽³³⁵⁾

■ `Caption`⁽³³⁶⁾
 ■ `ControlPanel`⁽³³⁷⁾
 ■ `Disabled`⁽³³⁷⁾
 ■ `Hint`⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.9.1.1 TrvActionInsertNumSequence.NumberType

A type of numbering for the inserted item (decimal, lower Roman, etc.)

property `NumberType: TRVSeqType;`

If `UseDefaults`⁽²⁴⁹⁾ = *False*, this property is used to initialize a numbering type field in a dialog.

This property is updated automatically to the value entered in this field, when this dialog is applied.

1.3.6.9.1.2 TrvActionInsertNumSequence.SeqName

A name (identifier) of the numbered sequence. The inserted item will continue numbering started by items having the same sequence name.

property `SeqName: TRVUnicodeString;`

If `UseDefaults`⁽²⁴⁹⁾ = *False*, this property is used to initialize a sequence name field in a dialog.

This property is updated automatically to the value entered in this field, when this dialog is applied.

The following sequence names are not allowed (the "Insert" button is disabled if they are entered):

- empty strings
- strings started from '@'

1.3.6.9.1.3 TrvActionInsertNumSequence.UseDefaults

Specifies whether values of `SeqName`⁽²⁴⁹⁾ and `NumberType`⁽²⁴⁹⁾ properties are used.

property `UseDefaults: Boolean`

When this property is *True*, the following values are used to initialize a dialog:

- sequence name = 'Numbering' (localized) for `TrvActionInsertNumber` or 'Figure' (localized) for `TrvActionInsertCaption`;
- numbering type = decimal.

When this property is *False*, values of `SeqName`⁽²⁴⁹⁾ and `NumberType`⁽²⁴⁹⁾ properties are used instead.

After an insertion, this property is reset to *False* automatically.

Initial value:

True

1.3.6.10 TrvActionInsertPageBreak

TrvActionInsertPageBreak is the action for "Insert Page Break" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertPageBreak = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action inserts a page break in the caret position (or, when called from a table cell, before the current row).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionRemovePageBreak⁽³³²⁾

1.3.6.11 TrvActionInsertPageCount

TrvActionInsertPageCount is the action for "Insert | Page Count" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertPageCount = class (TrvActionInsertCustomPageNumber(227))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾

TrvAction ⁽³¹³⁾*TrvActionInsertCustomPageNumber* ⁽²²⁷⁾**Description**

This action inserts "page count" field in the position of the caret. The number is formatted according to *NumberType* ⁽²²⁸⁾ property.

See also

- *TrvActionInsertPageNumber* ⁽²⁵¹⁾

1.3.6.12 TrvActionInsertPageNumber

TrvActionInsertPageNumber is the action for "Insert | Page Number" command.

Unit *RichViewActions* ⁽⁹⁴⁾ ;

Syntax

```
TrvActionInsertPageNumber = class (TrvActionInsertCustomPageNumber (227))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionInsertCustomPageNumber (227)

```

Description

This action inserts "page number" field in the position of the caret. The number is formatted according to *NumberType* ⁽²²⁸⁾ property.

See also

- *TrvActionInsertPageCount* ⁽²⁵⁰⁾

1.3.6.13 TrvActionInsertPicture

TrvActionInsertPicture is the action for "Insert | Picture" command.

Unit *RichViewActions* ⁽⁹⁴⁾ ;

Syntax

```
TrvActionInsertPicture = class (TrvActionCustomIO (100))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction

```

TContainedAction

TCustomAction

TAction

TrvCustomAction ⁽³³⁵⁾

TrvAction ⁽³¹³⁾

TrvActionCustomIO ⁽¹⁰⁰⁾

Description

This action inserts pictures from files at the caret position. Unlike all other file opening actions, this action detects file format by the file extension, not by the file filter index.

The action calls `GetControlPanel` ⁽³³⁸⁾.`OnChoosePicture` ⁽⁶³⁾ event (if assigned). If the event provides an image, the action inserts it.

Otherwise, the action displays `TOpenPictureDialog` to choose pictures. The action allows users to select more than one file in the dialog. Then it inserts chosen files one by one.

The following additional properties can be applied to inserted pictures:

- `VAlign` ⁽²⁵⁵⁾ – the image alignment relative to text;
- `Spacing` ⁽²⁵⁵⁾ – padding (spacing between the image itself and its border);
- `OuterHSpacing`, `OuterVSpacing` ⁽²⁵⁵⁾ – horizontal and vertical spacing around the image border;
- `BorderWidth`, `BorderColor` ⁽²⁵³⁾ – border width and color;
- `BackgroundColor` ⁽²⁵³⁾ – background color;
- `MaxImageSize` ⁽²⁵⁴⁾ – maximal allowed size (the image will be stretched if it exceeds this size);
- `StoreFileNameInItemName` ⁽²⁵⁵⁾ allows storing file name with the image (It's not recommended to use this option. The recommended place for storing image file names is *rvesplImageFileName* extra item property. It is assigned if *rvoAssignImageFileNames* is included in the Options property of the target editor).

You can modify a picture before insertion in `OnInserting` ⁽²⁵⁶⁾ event.

1.3.6.13.1 Properties

In `TrvActionInsertPicture`

- `BackgroundColor` ⁽²⁵³⁾
- `BorderColor` ⁽²⁵³⁾
- `BorderWidth` ⁽²⁵³⁾
- `DefaultExt` ⁽²⁵⁴⁾
- `Filter` ⁽²⁵⁴⁾
- `MaxImageSize` ⁽²⁵⁴⁾
- `OuterHSpacing` ⁽²⁵⁵⁾
- `OuterVSpacing` ⁽²⁵⁵⁾
- `Spacing` ⁽²⁵⁵⁾
- `StoreFileNameInItemName` ⁽²⁵⁵⁾

- VAlign ⁽²⁵⁵⁾

Derived from TrvActionCustomIO ⁽¹⁰⁰⁾

- AutoUpdateFileName ⁽¹⁰¹⁾
- DialogTitle ⁽¹⁰²⁾
- FileName ⁽¹⁰²⁾
- InitialDir ⁽¹⁰²⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.13.1.1 TrvActionInsertPicture.BackgroundColor

The property specify a background color for inserted images.

property BackgroundColor: TColor;

Default value

clNone (no background color)

1.3.6.13.1.2 TrvActionInsertPicture.BorderWidth, BorderColor

The property specify a border width and color for inserted images.

property BorderWidth: TRVStyleLength;

property BorderColor: TColor;

Assign a positive value to BorderWidth to add border to inserted pictures.

BorderWidth is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value

- BorderWidth: 0
- BorderColor: clBlack

1.3.6.13.1.3 TrvActionInsertPicture.DefaultExt

Specifies the default file extension.

property DefaultExt: TRVALocString;

DefaultExt specifies a file extension that is appended automatically to the selected file name, unless the selected file name already includes a registered extension. If the user selects a file name with an extension that is unregistered, DefaultExt is appended to the unregistered extension.

This property is assigned to DefaultExt property of the file opening dialog (TOpenPictureDialog).

Default value:

'bmp'

1.3.6.13.1.4 TrvActionInsertPicture.Filter

Determines the file masks (filters) available in the file opening dialog.

property Filter: TRVALocString;

If this property is not empty, it is assigned to the Filter property of the file opening dialog (TOpenPictureDialog). Otherwise, a filter containing all registered graphic formats is created.

Default value:

" (empty string)

See also:

- TrvActionItemProperties.GraphicFilter⁽³³⁰⁾
- TrvActionItemProperties.BackgroundGraphicFilter⁽³²⁹⁾
- TrvActionTableProperties.BackgroundGraphicFilter⁽³⁰⁵⁾

1.3.6.13.1.5 TrvActionInsertPicture.MaxImageSize

Defines the maximal image size for inserting without scaling.

property MaxImageSize: TRVStyleLength;

A positive value limits sizes of inserted images. If width or height of the inserted picture exceeds this value, the image is scaled down proportionally. The actual image is not changed, it's just inserted scaled (its *rveplImageHeight* and *rveplImageWidth* properties are assigned, see the help file on TRVExtralItemProperty).

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value:

0 (no scaling)

1.3.6.13.1.6 TrvActionInsertPicture.OuterHSpacing, OuterVSpacing

Specifies outer spacing for inserted images (spacing around the image border)

property OuterHSpacing: TRVStyleLength;

property OuterVSpacing: TRVStyleLength;

OuterHSpacing defines a horizontal spacing, OuterVSpacing defines a vertical spacing,

Assign a positive values to these property to add spacing around inserted pictures.

These values are measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value

0

1.3.6.13.1.7 TrvActionInsertPicture.Spacing

Specifies padding for inserted images (spacing between the image and its border)

property Spacing: TRVStyleLength;

Assign a positive value to this property to add padding for inserted pictures.

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

Default value

0

1.3.6.13.1.8 TrvActionInsertPicture.StoreFileNameInItemName

Specifies whether file names (full paths) are stored in item names of inserted pictures.

property StoreFileNameInItemName: Boolean;

If this property is *True*, path to the graphic file is stored in the document in the item name (item text).

It's not recommended to use this option. The recommended place for storing image file names is *rvsplImageFileName* extra item property. It is assigned if *rvoAssignImageFileNames* is included in the Options property of the target editor.

Default value

False

See also:

- TrvActionPasteSpecial.StoreFileNameInItemName⁽¹⁴⁰⁾
- TrvActionItemProperties⁽³²⁶⁾ (does not support storing paths in item names)
- RTF reading (does not support storing paths in item names)

1.3.6.13.1.9 TrvActionInsertPicture.VAlign

Specifies the vertical alignment for inserted pictures.

property VAlign: TRVVAlign;

Default value:

rvvaBaseline

1.3.6.13.2 Events

In TrvActionInsertPicture

■ OnInserting⁽²⁵⁶⁾

1.3.6.13.2.1 TrvActionInsertPicture.OnInserting

Occurs before inserting image in the target editor.

type

```
TRVInsertPictureEvent = procedure (Sender: TObject;
  Editor: TCustomRichViewEdit; var Graphic: TGraphic;
  const FileName: TRVUnicodeString)
of object;
```

property OnInserting: TRVInsertPictureEvent;

Parameters

Graphic – picture that will be inserted in **Editor**.

FileName – name of file, from where **Graphic** was read.

You can:

- modify the content of **Graphic**, or
- free **Graphic** and assign a new graphic object to it, or
- free **Graphic** and assign *nil* to it (to prevent insertion)

1.3.6.14 TrvActionInsertSymbol

TrvActionInsertSymbol is the action for "Insert | Symbol" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertSymbol = class (TrvAction(313))
```

Hierarchy

```
TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
```

Description

This action inserts a character in the caret position of the target editor.

1.3.6.14.1 Properties

In TrvActionInsertSymbol

- CharCode ⁽²⁵⁷⁾
- FontName ⁽²⁵⁸⁾
- Protection ⁽²⁵⁸⁾
- UseCurrentFont ⁽²⁵⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.6.14.1.1 TrvActionInsertSymbol.CharCode

Last inserted character (UTF-32)

property CharCode: Cardinal;

When the dialog will be shown next time, it will show this character. When the user inserts a character, it is assigned to this property.

Default value:

\$F0B7 (bullet of "Symbol" font)

1.3.6.14.1.2 TrvActionInsertSymbol.FontName

Last used font name.

property FontName: TRVALocString⁽³⁵⁰⁾;

When the dialog will be shown next time, it will show this font name. When the user inserts a character, its font is assigned to this property.

This property is used only if UseCurrentFont⁽²⁵⁸⁾ = *False*.

Default value:

'Symbol'

1.3.6.14.1.3 TrvActionInsertSymbol.Protection

Defines protection for inserted characters;

property Protection: TRVProtectOptions;

Protection options specified in this property is added to the protection options of text styles of inserted characters.

Default value:

[rvprDoNotAutoSwitch]

1.3.6.14.1.4 TrvActionInsertSymbol.UseCurrentFont

Defines how the initial value for a character's font is chosen.

property UseCurrentFont: Boolean;

If *False*: "Symbol" font is used when the dialog is displayed for the first time. The font of the last chosen character is used for subsequent calls.

If *True*: the font of the current text style is used.

If you assign *True* to this property, it makes sense to assign a different initial value to CharCode⁽²⁵⁷⁾, because the default value is for "Symbol" font.

Default value:

False

1.3.6.15 TrvActionInsertText

TrvActionInsertText is the action for text insertion.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionInsertText = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject

TPersistent

TComponent

TBasicAction

TContainedAction

*TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾

Description

This action calls `OnInsertText` ⁽²⁵⁹⁾ event and inserts the returned text in the caret position. This action does not have predefined Caption and Hint. Assign these properties yourself.

For example, this action may be used to insert the current date or time.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of `TrvAction`.

See also:

- `TrvActionEvent` ⁽³²³⁾

1.3.6.15.1 Events

In `TrvActionInsertText`

■ `OnInsertText` ⁽²⁵⁹⁾

1.3.6.15.1.1 `TrvActionInsertText.OnInsertText`

Requests text for insertion.

type

```
TRVInsertTextEvent = procedure (Sender: TObject;
  Editor: TCustomRichViewEdit;
  var Text: TRVUnicodeString) of object;
```

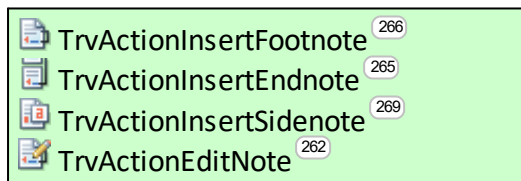
property `OnInsertText`: `TRVInsertTextEvent`;


The string assigned to **Text** will be inserted in **Editor**.

1.3.7 Notes and text boxes

Actions Related to Notes and Text Boxes

This group of actions includes commands to insert notes and text boxes, and to edit them (in a separate editor).



 **ScaleRichView note:** it's not recommended to use these actions for `TSRichViewEdit`. `TSRichViewEdit` does not need a separate editor to edit notes and text boxes. `ScaleRichView` includes its own set of actions for notes and text boxes.

1.3.7.1 TrvActionCustomInsertSidenote

TrvActionCustomInsertSidenote is a base class for the actions inserting sidenotes and text box items.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionCustomInsertSidenote = class (TrvActionInsertNote (267))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionInsertNote ⁽²⁶⁷⁾

Description

This action introduces properties that are assigned to properties of an inserted sidenote (or a text box item):

- BoxProperties ⁽²⁶²⁾
- BoxPosition ⁽²⁶¹⁾

Text in the note's Document is formatted either according to StyleTemplateName ⁽²⁶²⁾ (if style templates are used) or according to Font ⁽²⁶⁸⁾ (otherwise).

This class is not used directly. The following actions are inherited from TrvActionCustomInsertSidenote:

- TrvActionInsertSidenote ⁽²⁶⁹⁾
- TrvActionInsertTextBox ⁽²⁷¹⁾

1.3.7.1.1 Properties

In TrvActionCustomInsertSidenote

- BoxPosition ⁽²⁶¹⁾
- BoxProperties ⁽²⁶²⁾
- StyleTemplateName ⁽²⁶²⁾

Derived from TRVActionInsertNote ⁽²⁶⁷⁾

- ActionEditNote ⁽²⁶⁸⁾
- Font ⁽²⁶⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.7.1.1.1 TrvActionCustomInsertSidenote.BoxPosition

Specifies position properties for the floating box.

type

```
TRVABoxPosition = class (TRVBoxPosition);
```

property BoxPosition: TRVABoxPosition;

This property is assigned to BoxPosition property of the inserted sidenote / text box item.

Default values are different in inherited actions.

TrvActionInsertSidenote ²⁶⁹:

- HorizontalAnchor=*rvhanMainTextArea*;
- HorizontalPositionKind=*rvfpkAlignment*;
- HorizontalAlignment=*rvfphalRight*;
- VerticalAnchor=*rvvanParagraph*;
- VerticalPositionKind=*rvfpkAbsPosition*;
- VerticalOffset=0.

TrvActionInsertTextBox ²⁷¹:

- HorizontalAnchor=*rvhanMainTextArea*;
- HorizontalPositionKind=*rvfpkAlignment*;
- HorizontalAlignment=*rvfphalCenter*;
- VerticalAnchor=*rvvanLine*;
- VerticalPositionKind=*rvfpkAbsPosition*;
- VerticalOffset=20.

1.3.7.1.1.2 TrvActionCustomInsertSidenote.BoxProperties

Specifies size, background, border, vertical alignment properties for the floating box.

type

```
TRVABoxProperties = class (TRVBoxProperties);
```

property BoxProperties: TRVABoxProperties;

This property is assigned to BoxProperties property of the inserted sidenote / text box item.

Default values

- WidthType=rvbwtRelPage;
- Width=20000;

1.3.7.1.1.3 TrvActionCustomInsertSidenote.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=*True*. Otherwise, the 0-th paragraph style and Font⁽²⁶⁸⁾ are used.

Default values are different in inherited actions:

TrvActionInsertSidenote⁽²⁶⁹⁾:

'Sidenote Text' (note: this style template is standard for RichViewActions, but it is not standard for Microsoft Word)

TrvActionInsertTextBox⁽²⁷¹⁾:

'Normal'

1.3.7.2 TrvActionEditNote

TrvActionEditNote allows editing a footnote, an endnote, a sidenote, or a text box in a non-modal dialog window.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertNote = class (TrvCustomEditorAction(338))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvCustomEditorAction(338)
```

Description



ScaleRichView note: it's not recommended to use this action for TSRichViewEdit. Use TrsvActionEditNote instead.

This action can be executed directly, or it is called by TrvActionInsertFootnote⁽²⁶⁶⁾, TrvActionInsertEndnote⁽²⁶⁵⁾, TrvActionInsertSidenote⁽²⁶⁹⁾, TrvActionInsertTextBox⁽²⁷¹⁾.

It's recommended to assign Control⁽³¹⁴⁾ property for this action; otherwise, this property is assigned automatically when the action is executed for the first time.

When executed, the action shows a non-modal dialog window containing TRichViewEdit for editing subdocuments of items of the following types:

- footnote;
- endnote;
- sidenote;
- text box item.

If there is no item of these types in the caret position of the target editor, the action can move the caret to the next such item, if JumpToNextNote⁽²⁶⁴⁾=True. If there are no such items, a note editor is shown disabled.

If a note editor window is already visible when this action is executed, the action activates its window.

After the execution, the window is still open and the action reacts on the caret movement in the target editor (more exactly, in Control⁽³¹⁴⁾): the window always shows a document for the current note. The user can edit the note document. It is saved to the note (as an undoable operation) automatically in the following cases:

- a caret movement in Control⁽³¹⁴⁾;
- saving Control⁽³¹⁴⁾ to a file or a stream;
- printing or displaying a print preview of Control⁽³¹⁴⁾;
- showing or hiding the note editor window;
- assigning a different value to Control⁽³¹⁴⁾.

If you do not want to save recent changes, undo them in the note editor (there is no "Cancel Changes" command).

Editing in the note editor

By default, a note editor is shown in a floating window, without any additional controls in this window. If you attempt to use toolbar buttons located on another form, you will see the problem: clicking the button activates another form, and the command will be executed for an editor in this form.

So you have two main options to implement UI for a note editor:

- use OnFormCreate⁽³⁴⁰⁾ event to insert additional control in the note editor window (such as menu or toolbar)
- insert the note window in the main form, and switch RVAControlPanel.DefaultControl⁽⁴³⁾ when the note editor acquires and releases the input focus (see the example in the topic about OnShowing and OnHide⁽³⁴⁰⁾)

The latter is implemented in the ActionTest demos.

1.3.7.2.1 Properties

In TrvActionEditNote

- JumpToNextNote ⁽²⁶⁴⁾

Derived from TRVCustomEditorAction ⁽³³⁸⁾

- ▶ Form ⁽³³⁹⁾
- ▶ SubDocEditor ⁽³³⁹⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.7.2.1.1 TrvActionEditNote.JumpToNextNote

Allows moving the caret to the next note in the target editor, when this action is executed.

property JumpToNextNote: Boolean;

The action always shows a note editor window on execution.

If **JumpToNextNote**=*True*, and the current item in the target editor is not a note when the action is executed, the action finds the next note in the target editor, moves the caret to it, and then shows a note editor window.

Default value:

True

1.3.7.2.2 Events

In TrvActionEditNote

- OnStartEditNote ⁽²⁶⁵⁾

Inherited from TRVCustomEditorAction ⁽³³⁸⁾

- OnFormCreate ⁽³⁴⁰⁾
- OnHide ⁽³⁴⁰⁾
- OnShowing ⁽³⁴⁰⁾

1.3.7.2.2.1 TrvActionEditNote.OnStartEditNote

Occurs when the action starts editing a new note.

property OnStartEditNote: TNotifyEvent;

This event occurs:

- when the action starts editing a note
- when the action ends editing a note

You can store a reference to the form containing the note editor in OnShowing ⁽³⁴⁰⁾ event, and use this form's caption in this event.

1.3.7.3 TrvActionInsertEndnote

TrvActionInsertEndnote inserts a new endnote in the document.

Unit RichViewActions ⁽⁹⁴⁾;


Syntax

TrvActionInsertEndnote = **class** (TrvActionInsertNote ⁽²⁶⁷⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionInsertNote ⁽²⁶⁷⁾

Description

 **ScaleRichView note:** it's not recommended to use this action for TScaleRichViewEdit. Use TsrvActionInsertEndnote instead.

TrvActionInsertEndnote inserts a new endnote in the document, then executes ActionEditNote ⁽²⁶⁸⁾ to open an editor for this note.

If `TrvActionEditNote`⁽²⁶²⁾'s editor is active when this action is executed, this action inserts a reference to the parent endnote (`TRVNoteReferenceItemInfo` item).

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` 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`⁽²⁶⁸⁾ property.

Note: "endnote reference" and "endnote text" style templates are added by default to `TrvActionNew.StyleTemplates`, so they are included in new documents.

1.3.7.4 TrvActionInsertFootnote

`TrvActionInsertFootnote` inserts a new footnote in the document.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax


```
TrvActionInsertFootnote = class(TrvActionInsertNote(267))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionInsertNote(267)
```

Description

 **ScaleRichView note:** it's not recommended to use this action for `TSRichViewEdit`. Use `TsrvActionInsertFootnote` instead.

`TrvActionInsertFootnote` inserts a new footnote in the document, then executes `ActionEditNote`⁽²⁶⁸⁾ to open an editor for this note.

If `TrvActionEditNote`⁽²⁶²⁾'s editor is active when this action is executed, this action inserts a reference to the parent footnote (`TRVNoteReferenceItemInfo` item).

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` 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`⁽²⁶⁸⁾ property.

Note: "footnote reference" and "footnote text" style templates are added by default to `TrvActionNew.StyleTemplates`, so they are included in new documents.

1.3.7.5 TrvActionInsertNote

`TrvActionInsertNote` is a base class for the actions inserting footnotes, endnotes, sidenotes and text boxes.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionInsertNote = class (TrvAction(313))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)

```

Description

This class is not used directly.

The following actions are inherited from `TrvActionInsertNote`:

- `TrvActionInsertFootnote`⁽²⁶⁶⁾
- `TrvActionInsertEndnote`⁽²⁶⁵⁾
- `TrvActionInsertSidenote`⁽²⁶⁹⁾
- `TrvActionInsertTextBox`⁽²⁷¹⁾

1.3.7.5.1 Properties

In `TRVActionInsertNote`

- `ActionEditNote`⁽²⁶⁸⁾
- `Font`⁽²⁶⁸⁾

Derived from TrvAction ⁽³¹³⁾

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾

■ ControlPanel ⁽³³⁷⁾

■ Disabled ⁽³³⁷⁾

■ Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.7.5.1.1 TrvActionInsertNote.ActionEditNote

Links this action to TrvActionEditNote ⁽²⁶²⁾ action.

```
property ActionEditNote: TrvActionEditNote (262);
```

The action executes ActionEditNote after inserting a note.

If this link is not defined, the first found TrvActionEditNote ⁽²⁶²⁾ action on the same form/datamodule is used. If not found, a window for editing the inserted note is not shown.

1.3.7.5.1.2 TrvActionInsertNote.Font

Default font for text in a note's document.

type

```
TRVAFont = TFont;
```

```
property Font: TRVAFont;
```

This value is used if UseStyleTemplates=*False* for the target editor, or the required style template does not exist in Style.StyleTemplates of the target editor. Otherwise, note text is formatted according to a style template.

Style template names:

- "footnote text" for footnotes,
- "endnote text" for endnotes,
- `StyleTemplateName`⁽²⁶²⁾ for sidenotes and text boxes (default values: "Sidenote Text" for sidenotes, "Normal" for text boxes)

Default value:

'Arial', 10

1.3.7.6 TrvActionInsertSidenote

`TrvActionInsertSidenote` inserts a new sidenote (a note in a floating box) in the document.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionCustomInsertSidenote = class (TrvActionCustomInsertSidenote(260))
```


Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionInsertNote(267)
TrvActionCustomInsertSidenote(260)

```

Description

 **ScaleRichView note:** it's not recommended to use this action for `TSRichViewEdit`. Use `TsrvActionInsertSidenote` instead.

`TrvActionInsertSidenote` inserts a new sidenote in the document, then executes `ActionEditNote`⁽²⁶⁸⁾ to open an editor for this note.

If `TrvActionEditNote`⁽²⁶²⁾'s editor is active when this action is executed, this action inserts a reference to the parent sidenote (`TRVNoteReferenceItemInfo` item).

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` property.

If style templates are used, the sidenote and the sidenote reference characters are formatted using `RefStyleTemplateName`⁽²⁷¹⁾ 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`⁽²⁶²⁾ style template. Otherwise, or if this style template does not exist, sidenote text is formatted using `Font`⁽²⁶⁸⁾ property.

The following properties are assigned to the inserted sidenote:

- `BoxProperties`⁽²⁶²⁾ (default: width: 20% of page width, height: auto)
- `BoxPosition`⁽²⁶¹⁾ (default: horizontal - right aligned to the main text area, vertical - aligned to the top of the paragraph)

1.3.7.6.1 Properties

In `TrvActionInsertSidenote`

- `RefStyleTemplateName`⁽²⁷¹⁾

Derived from `TrvActionCustomInsertSidenote`⁽²⁶⁰⁾

- `BoxPosition`⁽²⁶¹⁾
- `BoxProperties`⁽²⁶²⁾
- `StyleTemplateName`⁽²⁶²⁾

Derived from `TRVActionInsertNote`⁽²⁶⁷⁾

- `ActionEditNote`⁽²⁶⁸⁾
- `Font`⁽²⁶⁸⁾

Derived from `TrvAction`⁽³¹³⁾

- `Control`⁽³¹⁴⁾

Derived from `TrvCustomAction`⁽³³⁵⁾

- `Caption`⁽³³⁶⁾
- `ControlPanel`⁽³³⁷⁾
- `Disabled`⁽³³⁷⁾
- `Hint`⁽³³⁷⁾

Derived from `TAction`

- `AutoCheck`
- `Caption`
- `Checked`
- `DisableIfNoHandler`
- `Enabled`
- `GroupIndex`
- `HelpContext`
- `HelpKeyword`
- `HelpType`
- `Hint`
- `ImageIndex`
- `Name`
- `SecondaryShortCuts`
- `ShortCut`

■ Visible

1.3.7.6.1.1 TrvActionInsertSidenote.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=*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)

1.3.7.7 TrvActionInsertTextBox

TrvActionInsertTextBox inserts a new text box item in the document.

Unit RichViewActions⁽⁹⁴⁾;


Syntax

TrvActionInsertTextBox = **class** (TrvActionCustomInsertSidenote⁽²⁶⁰⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionInsertNote⁽²⁶⁷⁾
TrvActionCustomInsertSidenote⁽²⁶⁰⁾

Description

 **ScaleRichView note:** it's not recommended to use this action for TScaleRichViewEdit. Use TsvActionInsertTextBox instead.

TrvActionInsertTextBox inserts a new text box item in the document, then executes ActionEditNote⁽²⁶⁸⁾ to open an editor for its document.

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 property.

If style templates are used, this text item's paragraph and text are formatted using StyleTemplateName⁽²⁶²⁾ style template. Otherwise, or if this style template does not exist, this text is formatted using Font⁽²⁶⁸⁾ property.

The following properties are assigned to the inserted text box:




























- BoxProperties⁽²⁶²⁾ (default: width: 20% of page width, height: auto)

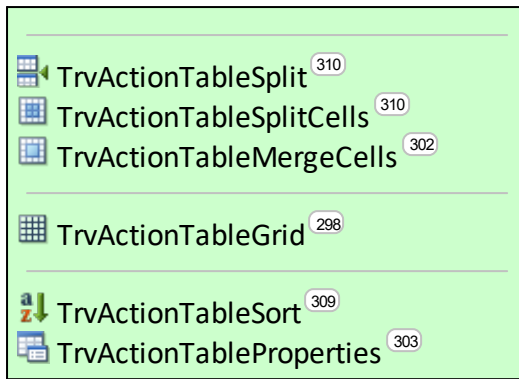
- `BoxPosition` ⁽²⁶¹⁾ (default: horizontal - centered in the main text area, vertical - 20 pixels below the top of the line)

1.3.8 Tables

Table Actions

This group of actions includes a command to insert table, and various operations on the selected table.

	<code>TrvActionInsertTable</code> ⁽²⁷³⁾
	<code>TrvActionTableInsertColLeft</code> ⁽²⁹⁹⁾
	<code>TrvActionTableInsertColRight</code> ⁽²⁹⁹⁾
<hr/>	
	<code>TrvActionTableInsertRowsAbove</code> ⁽³⁰⁰⁾
	<code>TrvActionTableInsertRowsBelow</code> ⁽³⁰¹⁾
<hr/>	
	<code>TrvActionTableDeleteCols</code> ⁽²⁹⁷⁾
	<code>TrvActionTableDeleteRows</code> ⁽²⁹⁷⁾
	<code>TrvActionTableDeleteTable</code> ⁽²⁹⁸⁾
	<code>TrvActionTableToText</code> ⁽³¹¹⁾
<hr/>	
	<code>TrvActionTableSelectTable</code> ⁽³⁰⁸⁾
	<code>TrvActionTableSelectCols</code> ⁽³⁰⁷⁾
	<code>TrvActionTableSelectRows</code> ⁽³⁰⁸⁾
	<code>TrvActionTableSelectCell</code> ⁽³⁰⁷⁾
<hr/>	
	<code>TrvActionTableCellVAlignTop</code> ⁽²⁹⁶⁾
	<code>TrvActionTableCellVAlignMiddle</code> ⁽²⁹⁵⁾
	<code>TrvActionTableCellVAlignBottom</code> ⁽²⁹⁴⁾
	<code>TrvActionTableCellVAlignDefault</code> ⁽²⁹⁵⁾
<hr/>	
	<code>TrvActionTableCellRotationNone</code> ⁽²⁹⁰⁾
	<code>TrvActionTableCellRotation90</code> ⁽²⁹¹⁾
	<code>TrvActionTableCellRotation180</code> ⁽²⁹¹⁾
	<code>TrvActionTableCellRotation270</code> ⁽²⁹²⁾
<hr/>	
	<code>TrvActionTableCellLeftBorder</code> ⁽²⁸⁷⁾
	<code>TrvActionTableCellTopBorder</code> ⁽²⁹³⁾
	<code>TrvActionTableCellRightBorder</code> ⁽²⁸⁹⁾
	<code>TrvActionTableCellBottomBorder</code> ⁽²⁸⁶⁾
	<code>TrvActionTableCellAllBorders</code> ⁽²⁸⁵⁾
	<code>TrvActionTableCellNoBorders</code> ⁽²⁸⁸⁾



1.3.8.1 TrvActionInsertTable

TrvActionInsertTable is the action for "Table | Insert Table" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionInsertTable = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

The action can insert a table in two ways:

- using a table dialog (when executed)
- using a special table size window (when ShowTableSizeDialog⁽²⁸⁴⁾ is called).

In the table dialog, the user can specify a count of rows and columns and table width modes:

- autosize (table.BestWidth=0; widths of table cells are specified in pixels (or twips) so that the table has width not exceeding the current document width)
- fit window (table.BestWidth=-100 (i.e. 100%); widths of table cells are not specified)
- custom size (table.BestWidth is defined in the dialog, in pixels (or twips) or percent; widths of table cells are not specified)

When inserting using a table size window, table.BestWidth is specified in BestWidth⁽²⁷⁶⁾. If it is zero, cells widths are defined like in "autosize" mode.

If the user checked "remember dimensions for new tables" in the dialog, RowCount⁽²⁸¹⁾, ColCount⁽²⁸⁰⁾ and BestWidth⁽²⁷⁶⁾ properties are set to the values entered in the dialog.

Before the insertion:

- the properties of this action are assigned to the table properties of the same names;

- RowsKeepTogether⁽²⁸¹⁾ and RowsVAlign⁽²⁸²⁾ are assigned to properties of all rows of the table,
- OnInserting⁽²⁸⁴⁾ event occurs.

The user can assign properties of this action by checking "Default" check box in the table properties dialog displayed by TrvActionTableProperties⁽³⁰³⁾ or TrvActionItemProperties⁽³²⁶⁾.

1.3.8.1.1 Properties

In TrvActionInsertTable

- BackgroundPicture⁽²⁷⁵⁾
- BackgroundProperties⁽²⁷⁵⁾
- BestWidth⁽²⁷⁶⁾
- BorderColor⁽²⁷⁶⁾
- BorderHSpacing⁽²⁷⁶⁾
- BorderLightColor⁽²⁷⁶⁾
- BorderStyle⁽²⁷⁷⁾
- BorderVSpacing⁽²⁷⁷⁾
- BorderWidth⁽²⁷⁷⁾
- CellBorderColor⁽²⁷⁷⁾
- CellBorderLightColor⁽²⁷⁸⁾
- CellBorderStyle⁽²⁷⁸⁾
- CellBorderWidth⁽²⁷⁸⁾
- CellHPadding⁽²⁷⁸⁾
- CellHSpacing⁽²⁷⁹⁾
- CellPadding⁽²⁷⁸⁾
- CellVPadding⁽²⁷⁸⁾
- CellVSpacing⁽²⁷⁹⁾
- ColBandSize⁽²⁷⁹⁾
- ColCount⁽²⁸⁰⁾
- Color⁽²⁸⁰⁾
- EvenColumnsColor⁽²⁸³⁾
- EvenRowsColor⁽²⁸³⁾
- FirstColumnColor⁽²⁸³⁾
- HeadingRowColor⁽²⁸³⁾
- HeadingRowCount⁽²⁸⁰⁾
- HOutermostRule⁽²⁸⁰⁾
- HRuleColor⁽²⁸⁰⁾
- HRuleWidth⁽²⁸¹⁾
- ItemText⁽²⁸¹⁾
- LastColumnColor⁽²⁸³⁾
- LastRowColor⁽²⁸³⁾
- OddColumnsColor⁽²⁸³⁾
- OddRowsColor⁽²⁸³⁾
- RowBandSize⁽²⁷⁹⁾
- RowCount⁽²⁸¹⁾
- RowsKeepTogether⁽²⁸¹⁾
- RowsVAlign⁽²⁸²⁾
- TableOptions⁽²⁸²⁾

- TablePrintOptions ⁽²⁸²⁾
- VisibleBorders ⁽²⁸²⁾
- VOutermostRule ⁽²⁸²⁾
- VRuleColor ⁽²⁸³⁾
- VRuleWidth ⁽²⁸³⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.8.1.1.1 TrvActionInsertTable.BackgroundImage

Defines the table background image.

```
property BackgroundPicture: TPicture;
```

BackgroundPicture.Graphic is assigned to the table's BackgroundImage

1.3.8.1.1.2 TrvActionInsertTable.BackgroundProperties

Defines the table background properties.

```
property BackgroundProperties: TRVBackgroundProperties;
```

This property is assigned to the table property of the same name.

1.3.8.1.1.3 TrvActionInsertTable.BestWidth

Defines the table width.

property BestWidth: TRVHTMLLength;

This value is used:

- as an initial value for initializing a table dialog (on the action execution),
- directly (when ShowTableSizeDialog⁽²⁸⁴⁾ is called).

If the user chose to remember dimensions for new tables, value from the dialog is assigned to this property.

This property (or the value from the table dialog) is assigned to table.BestWidth property. 0 means a default width, negative values mean percent, positive values mean GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

If it is zero (i.e. a default table width), the action sets BestWidth properties of all table cells to positive values so that the table width does not exceed the current document width. Otherwise, BestWidth properties of all cells are 0.

Default value:

0

1.3.8.1.1.4 TrvActionInsertTable.BorderColor

Defines the table border color (or color of dark sides of 3D table border)

property BorderColor: TColor;

This property is assigned to the table property of the same name.

Default value:

clWindowText

1.3.8.1.1.5 TrvActionInsertTable.BorderHSpacing

Defines the spacing between the left table border and the cells, and between the right table border and the cells.

property BorderHSpacing: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

Default value:

2

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.6 TrvActionInsertTable.BorderLightColor

Defines the color of light sides of 3D table border

property BorderLightColor: TColor;

This property is assigned to the table property of the same name.

Default value:*clBtnHighlight***1.3.8.1.1.7 TrvActionInsertTable.BorderStyle**

Defines the style of table border.

property BorderStyle: TRVTableBorderStyle;

This property is assigned to the table property of the same name.

Default value:*rvtbColor***1.3.8.1.1.8 TrvActionInsertTable.BorderVSpacing**

Defines the spacing between the top table border and the cells, and between the bottom table border and the cells.

property BorderVSpacing: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

Default value:

2

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.9 TrvActionInsertTable.BorderWidth

Defines the width of table border.

property BorderWidth: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

0 hides the border.

Default value:

1

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.10 TrvActionInsertTable.CellBorderColor

Defines the default color of cell borders (or of dark sides of 3D cell borders).

property CellBorderColor: TColor;

This property is assigned to the table property of the same name.

Default value:

clWindowText

1.3.8.1.1.11 TrvActionInsertTable.CellBorderLightColor

Defines the default color of light sides of 3D cell borders.

property CellBorderLightColor: TColor;

This property is assigned to the table property of the same name.

Default value:

clBtnHighlight

1.3.8.1.1.12 TrvActionInsertTable.CellBorderStyle

Defines the style of cells borders.

property CellBorderStyle: TRVTableBorderStyle;

This property is assigned to the table property of the same name.

Default value:

rvtbColor

1.3.8.1.1.13 TrvActionInsertTable.CellBorderWidth

Defines the width of cells borders.

property CellBorderWidth: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

0 hides the border.

Default value:

1

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.14 TrvActionInsertTable.CellHPadding,CellVPadding

The properties define spacing between border and content in cells.

property CellHPadding: TRVStyleLength;

property CellVPadding: TRVStyleLength;

property CellPadding: TRVStyleLength;

These values are measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

CellHPadding defines a horizontal spacing, CellVPadding defines a vertical spacing.

These properties are assigned to the table properties of the same names (converted to Units of the target editor).

CellPadding is used to simplify access to these properties. Assigning value to CellPadding assigns it to CellHPadding and CellVPadding. CellPadding returns (CellHPadding+CellVPadding)/2.

Default value:

1

This default value assumes that these properties are measured in pixels. For twips, this value may be too small.

1.3.8.1.1.15 TrvActionInsertTable.CellHSpacing

Defines the horizontal spacing between cells (i.e. spacing between table columns).

property CellHSpacing: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

Default value:

2

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.16 TrvActionInsertTable.CellVSpacing

Defines the vertical spacing between cells (i.e. spacing between table rows).

property CellVSpacing: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

Default value:

2

This default value assumes that this property is measured in pixels. For twips, this value may be too small.

1.3.8.1.1.17 TrvActionInsertTable.ColBandSize, RowBandSize

The properties defining count of columns/rows in alternating column/row bands.

property ColBandSize: Integer;

property RowBandSize: Integer;

RowBandSize specify the count of rows in row bands.

ColBandSize specify the count of columns in column bands.

These properties are assigned to the table properties of the same names.

Default value:

1

See also

- HeadingRowColor and other colors ⁽²⁸³⁾

1.3.8.1.18 TrvActionInsertTable.ColCount

Defines the count of table columns.

property ColCount: Integer;

This property is used as an initial value for initializing a table dialog. If the user chose to remember dimensions for new tables, value from the dialog is assigned to this property.

This property is ignored in ShowTableSizeDialog ⁽²⁸⁴⁾ method.

Default value:

2

1.3.8.1.19 TrvActionInsertTable.Color

Defines the background table color.

property Color: TColor;

This property is assigned to the table property of the same name.

Default value:

clNone

1.3.8.1.20 TrvActionInsertTable.HeadingRowCount

Defines the count of heading rows (rows repeated on each page containing this table).

property HeadingRowCount: Integer;

This property is assigned to the table property of the same name.

Default value:

0

1.3.8.1.21 TrvActionInsertTable.HOutermostRule

Allows showing horizontal lines between the top and the bottom table borders and cells.

property HOutermostRule: Boolean;

This property is assigned to the table property of the same name.

Horizontal rules are drawn only if HRuleWidth ⁽²⁸¹⁾ > 0.

Default value:

False

1.3.8.1.22 TrvActionInsertTable.HRuleColor

Defines the color of horizontal rules.

property HRuleColor: TColor;

This property is assigned to the table property of the same name.

Horizontal rules are drawn only if HRuleWidth ⁽²⁸¹⁾ > 0.

Default value:*c/WindowText***1.3.8.1.1.23 TrvActionInsertTable.HRuleWidth**

Defines the width of horizontal rules.

property HRuleWidth: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

0 hides rules.

Default value:

0

1.3.8.1.1.24 TrvActionInsertTable.ItemText

Defines the item text (item name) for the table.

property ItemText: TRVUnicodeString;

This property is used as a parameter of InsertItem method.

Default value:

" (empty string)

1.3.8.1.1.25 TrvActionInsertTable.RowCount

Defines the count of table rows.

property RowCount: Integer;

This property is used as an initial value for initializing a table dialog. If the user chose to remember dimensions for new tables, value from the dialog is assigned to this property.

This property is ignored in ShowTableSizeDialog⁽²⁸⁴⁾ method.

Default value:

2


1.3.8.1.1.26 TrvActionInsertTable.RowsKeepTogether

Defines the "keep together" property for all rows of inserted tables.

property RowsKeepTogether: Boolean;

This property is assigned to KeepTogether properties of all rows of inserted tables.

If *True*, the component avoids breaking pages inside rows (pagination adds page breaks between rows, when possible)

 **ScaleRichView note:** TScaleRichViewEdit does not support page breaks inside table cells yet.

Default value:

False

1.3.8.1.1.27 TrvActionInsertTable.RowsVAlign

Defines the default vertical alignment of content of cells of all rows of inserted tables.

property RowsVAlign: TRVCellVAlign;

This property is assigned to VAlign properties of all rows of inserted tables.

Default value:

rvcTop

1.3.8.1.1.28 TrvActionInsertTable.TableOptions

Defines the table options.

property TableOptions: TRVTableOptions;

This property is assigned to the table property of the same name.

Default value:

[rvtoEditing, rvtoRowSizing, rvtoColSizing, rvtoRowSelect, rvtoColSelect]

1.3.8.1.1.29 TrvActionInsertTable.TablePrintOptions

Defines the table printing options.

property TablePrintOptions: TRVTablePrintOptions;

This property is assigned to the table's PrintOptions property.

Default value:

[rvtoHalftoneBorders,rvtoRowsSplit]

1.3.8.1.1.30 TrvActionInsertTable.VisibleBorders

Defines visible sides of a table border.

property VisibleBorders: TRVBooleanRect;

This property is assigned to the table property of the same name.

Default value:

(True, True, True, True)

1.3.8.1.1.31 TrvActionInsertTable.VOutermostRule

Allows showing vertical lines between the left and the right table borders and cells.

property VOutermostRule: Boolean;

This property is assigned to the table property of the same name.

Vertical rules are drawn only if VRuleWidth⁽²⁸³⁾>0.

Default value:

False

1.3.8.1.1.32 TrvActionInsertTable.VRuleColor

Defines the color of vertical rules.

property VRuleColor: TColor;

This property is assigned to the table property of the same name.

Vertical rules are drawn only if VRuleWidth⁽²⁸³⁾>0.

Default value:

c/WindowText

1.3.8.1.1.33 TrvActionInsertTable.VRuleWidth

Defines the width of vertical rules.

property VRuleWidth: TRVStyleLength;

This value is measured in GetControlPanel⁽³³⁸⁾.UnitsProgram⁽⁵⁷⁾.

This property is assigned to the table property of the same name (converted to Units of the target editor).

0 hides rules.

Default value:

0

1.3.8.1.1.34 TrvActionInsertTable's row and column colors

A set of properties defining default colors of cells in specific rows and columns.

property HeadingRowColor: TColor;

property LastRowColor: TColor;

property OddRowsColor: TColor;

property EvenRowsColor: TColor;

property FirstColumnColor: TColor;

property LastColumnColor: TColor;

property OddColumnsColor: TColor;

property EvenColumnsColor: TColor;

These properties are assigned to the table properties of the same names.

Default values

c/None

See also

- ColBandSize, RowBandSize

1.3.8.1.2 Methods

In TrvActionInsertTable

ShowTableSizeDialog⁽²⁸⁴⁾

Inherited from TrvCustomAction⁽³³⁵⁾

GetControlPanel⁽³³⁸⁾

1.3.8.1.2.1 TrvActionInsertTable.ShowTableSizeDialog

These methods display a table sizing window. When this window is closed, a table with the specified count of rows and columns is inserted in **Target**.

```
procedure ShowTableSizeDialog(Target: TCustomRichViewEdit;
  Button: TControl); overload;
procedure ShowTableSizeDialog(Target: TCustomRichViewEdit;
  const ButtonRect: TRect); overload;
```

The first version of this method displays this window relative to the position of **Button**.

In the second version of this method, this position is specified in **ButtonRect** (some third-party toolbar component do not use controls for buttons, so the first version of this method cannot be used).

If the user presses **Escape** while this window is open, the window is closed and insertion is canceled.

1.3.8.1.3 Events

In TrvActionInsertTable

- OnCloseTableSizeDialog⁽²⁸⁴⁾
- OnInserting⁽²⁸⁴⁾

1.3.8.1.3.1 TrvActionInsertTable.OnCloseTableSizeDialog

Occurs when a table sizing window (shown by ShowTableSizeDialog⁽²⁸⁴⁾ method) is closed.

```
property OnCloseTableSizeDialog: TNotifyEvent;
```

This event can be used to uncheck a button pressed before calling ShowTableSizeDialog⁽²⁸⁴⁾ method (if you have some button that must be pressed while this window is displayed).

1.3.8.1.3.2 TrvActionInsertTable.OnInserting

Occurs before **table** is inserted.

type

```
TRVInsertTableEvent = procedure (Sender: TrvActionInsertTable;
  table: TRVTableItemInfo) of object;
```

```
property OnInserting: TRVInsertTableEvent;
```

This event occurs after all the action's properties are applied to **table**. You can make additional changes in this event.

1.3.8.2 TrvActionTableCell

TrvActionTableCell is a base class for actions performing operations on table and table cells.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCell = class (TrvAction(313))
```


Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action is not used directly. This action has the following direct descendants:

- *TrvActionTableRCBase* ⁽³⁰⁶⁾
- *TrvActionTableDeleteTable* ⁽²⁹⁸⁾
- *TrvActionTableMergeCells* ⁽³⁰²⁾
- *TrvActionTableSplitCells* ⁽³¹⁰⁾
- *TrvActionTableSelectTable* ⁽³⁰⁸⁾
- *TrvActionTableSelectCell* ⁽³⁰⁷⁾
- *TrvActionTableMultiCellAttributes* ⁽³⁰³⁾
- *TrvActionTableProperties* ⁽³⁰³⁾

1.3.8.3 TrvActionTableCellAllBorders

TrvActionTableCellAllBorders is the action for "Table | Cell Borders | All Borders" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellAllBorders = class (TrvActionTableCellBorder (286))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes ⁽³⁰³⁾
TrvActionTableCellBorder ⁽²⁸⁶⁾

Description

The action makes all sides of borders of the selected table cells (or the edited cell) visible/invisible.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellLeftBorder⁽²⁸⁷⁾
- TrvActionTableCellRightBorder⁽²⁸⁹⁾
- TrvActionTableCellTopBorder⁽²⁹³⁾
- TrvActionTableCellBottomBorder⁽²⁸⁶⁾
- TrvActionTableCellNoBorders⁽²⁸⁸⁾

1.3.8.4 TrvActionTableCellBorder

TrvActionTableCellBorder is a base class for the actions changing cell borders.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellBorder = class(TrvActionTableMultiCellAttributes(303))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes⁽³⁰³⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableCellLeftBorder⁽²⁸⁷⁾
- TrvActionTableCellRightBorder⁽²⁸⁹⁾
- TrvActionTableCellTopBorder⁽²⁹³⁾
- TrvActionTableCellBottomBorder⁽²⁸⁶⁾
- TrvActionTableCellAllBorders⁽²⁸⁵⁾
- TrvActionTableCellNoBorders⁽²⁸⁸⁾

These actions change table.Cells[].VisibleBorders properties for the selected cells (or the edited cell). All these actions (except for TrvActionTableCellNoBorders⁽²⁸⁸⁾) are checkbox-like buttons: they are checked if the corresponding sides of cells borders are visible.

1.3.8.5 TrvActionTableCellBottomBorder

TrvActionTableCellBottomBorder is the action for "Table | Cell Borders | Bottom Border" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellBottomBorder = class (TrvActionTableCellBorder286)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313
TrvActionTableCell284
TrvActionTableMultiCellAttributes303
TrvActionTableCellBorder286

```

Description

The action makes the bottom borders of the selected table cells (or the edited cell) visible/invisible.

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

See also:

- TrvActionTableCellLeftBorder²⁸⁷
- TrvActionTableCellRightBorder²⁸⁹
- TrvActionTableCellTopBorder²⁹³
- TrvActionTableCellAllBorders²⁸⁵
- TrvActionTableCellNoBorders²⁸⁸

1.3.8.6 TrvActionTableCellLeftBorder

TrvActionTableCellLeftBorder is the action for "Table | Cell Borders | Left Border" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionTableCellLeftBorder = class (TrvActionTableCellBorder286)
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction335
TrvAction313
TrvActionTableCell284
TrvActionTableMultiCellAttributes303

```

TrvActionTableCellBorder ⁽²⁸⁶⁾

Description

The action makes the left borders of the selected table cells (or the edited cell) visible/invisible.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellRightBorder ⁽²⁸⁹⁾
- TrvActionTableCellTopBorder ⁽²⁹³⁾
- TrvActionTableCellBottomBorder ⁽²⁸⁶⁾
- TrvActionTableCellAllBorders ⁽²⁸⁵⁾
- TrvActionTableCellNoBorders ⁽²⁸⁸⁾

1.3.8.7 TrvActionTableCellNoBorders

TrvActionTableCellNoBorders is the action for "Table | Cell Borders | No Borders" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellNoBorders = class (TrvActionTableCellBorder (286))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes ⁽³⁰³⁾
TrvActionTableCellBorder ⁽²⁸⁶⁾

Description

The action makes all sides of border of the selected table cells (or the edited cell) invisible.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This is a redundant action, because it only provides a subset of functionality of TrvActionTableCellAllBorders ⁽²⁸⁵⁾.

See also:

- TrvActionTableCellLeftBorder ⁽²⁸⁷⁾
- TrvActionTableCellRightBorder ⁽²⁸⁹⁾
- TrvActionTableCellTopBorder ⁽²⁹³⁾
- TrvActionTableCellBottomBorder ⁽²⁸⁶⁾
- TrvActionTableCellAllBorders ⁽²⁸⁵⁾

1.3.8.8 TrvActionTableCellRightBorder

TrvActionTableCellRightBorder is the action for "Table | Cell Borders | Right Border" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRightBorder = class (TrvActionTableCellBorder (286))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionTableCell (284)
TrvActionTableMultiCellAttributes (303)
TrvActionTableCellBorder (286)

```

Description

The action makes the right borders of the selected table cells (or the edited cell) visible/invisible.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellLeftBorder ⁽²⁸⁷⁾
- TrvActionTableCellTopBorder ⁽²⁹³⁾
- TrvActionTableCellBottomBorder ⁽²⁸⁶⁾
- TrvActionTableCellAllBorders ⁽²⁸⁵⁾
- TrvActionTableCellNoBorders ⁽²⁸⁸⁾

1.3.8.9 TrvActionTableCellRotation

TrvActionTableCellRotation is a base class for the actions changing a rotation in table cells.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRotation = class (TrvActionTableMultiCellAttributes (303))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

```

TrvCustomAction ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionTableCell* ⁽²⁸⁴⁾*TrvActionTableMultiCellAttributes* ⁽³⁰³⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action is not used directly. The following actions are inherited from it:

- *TrvActionTableCellRotationNone* ⁽²⁹⁰⁾
- *TrvActionTableCellRotation90* ⁽²⁹¹⁾
- *TrvActionTableCellRotation180* ⁽²⁹¹⁾
- *TrvActionTableCellRotation270* ⁽²⁹²⁾

1.3.8.10 TrvActionTableCellRotationNone

TrvActionTableCellRotationNone is the action for "Table | No Cell Rotation" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRotationNone = class (TrvActionTableCellRotation (289))
```

Hierarchy

*TObject**TPersistent**TComponent**TBasicAction**TContainedAction**TCustomAction**TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾*TrvActionTableCell* ⁽²⁸⁴⁾*TrvActionTableMultiCellAttributes* ⁽³⁰³⁾*TrvActionTableCellRotation* ⁽²⁸⁹⁾

Description

The action resets the rotation of the selected table cells (or the edited cell) to 0°. It changes *table.Cells[].Rotation* to *rvrotNone*. If the cell was rotated vertically (90° or 270°), it also exchanges *table.Cells[].BestWidth* and *BestHeight*.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

See also:

- *TrvActionTableCellRotation90* ⁽²⁹¹⁾
- *TrvActionTableCellRotation180* ⁽²⁹¹⁾
- *TrvActionTableCellRotation270* ⁽²⁹²⁾

1.3.8.11 TrvActionTableCellRotation90

TrvActionTableCellRotation90 is the action for "Table | Rotate Cell by 90°" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRotation90 = class (TrvActionTableCellRotation (289))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes ⁽³⁰³⁾
TrvActionTableCellRotation ⁽²⁸⁹⁾

Description

The action sets the rotation of the selected table cells (or the edited cell) to 90°. It changes table.Cells[].Rotation to *rvrot90*. If the cell was rotated horizontally (0° or 180°), it also exchanges table.Cells[].BestWidth and BestHeight.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellRotationNone ⁽²⁹⁰⁾
- TrvActionTableCellRotation180 ⁽²⁹¹⁾
- TrvActionTableCellRotation270 ⁽²⁹²⁾

1.3.8.12 TrvActionTableCellRotation180

TrvActionTableCellRotation180 is the action for "Table | Rotate Cell by 180°" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRotation180 = class (TrvActionTableCellRotation (289))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

[TrvCustomAction](#)⁽³³⁵⁾
[TrvAction](#)⁽³¹³⁾
[TrvActionTableCell](#)⁽²⁸⁴⁾
[TrvActionTableMultiCellAttributes](#)⁽³⁰³⁾
[TrvActionTableCellRotation](#)⁽²⁸⁹⁾

Description

The action sets the rotation of the selected table cells (or the edited cell) to 180°. It changes `table.Cells[].Rotation` to `rvrot180`. If the cell was rotated vertically (90° or 270°), it also exchanges `table.Cells[].BestWidth` and `BestHeight`.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of `TrvAction`.

See also:

- [TrvActionTableCellRotationNone](#)⁽²⁹⁰⁾
- [TrvActionTableCellRotation90](#)⁽²⁹¹⁾
- [TrvActionTableCellRotation270](#)⁽²⁹²⁾

1.3.8.13 TrvActionTableCellRotation270

`TrvActionTableCellRotation270` is the action for "Table | Rotate Cell by 270°" command.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellRotation270 = class (TrvActionTableCellRotation(289))
```

Hierarchy

[TObject](#)
[TPersistent](#)
[TComponent](#)
[TBasicAction](#)
[TContainedAction](#)
[TCustomAction](#)
[TAction](#)
[TrvCustomAction](#)⁽³³⁵⁾
[TrvAction](#)⁽³¹³⁾
[TrvActionTableCell](#)⁽²⁸⁴⁾
[TrvActionTableMultiCellAttributes](#)⁽³⁰³⁾
[TrvActionTableCellRotation](#)⁽²⁸⁹⁾

Description

The action sets the rotation of the selected table cells (or the edited cell) to 270°. It changes `table.Cells[].Rotation` to `rvrot270`. If the cell was rotated horizontally (0° or 180°), it also exchanges `table.Cells[].BestWidth` and `BestHeight`.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of `TrvAction`.

See also:

- [TrvActionTableCellRotationNone](#)⁽²⁹⁰⁾
- [TrvActionTableCellRotation90](#)⁽²⁹¹⁾

- TrvActionTableCellRotation180⁽²⁹¹⁾

1.3.8.14 TrvActionTableCellTopBorder

TrvActionTableCellTopBorder is the action for "Table | Cell Borders | Top Border" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellTopBorder = class (TrvActionTableCellBorder(286))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)
TrvActionTableCell(284)
TrvActionTableMultiCellAttributes(303)
TrvActionTableCellBorder(286)

```

Description

The action makes the top borders of the selected table cells (or the edited cell) visible/invisible.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellLeftBorder⁽²⁸⁷⁾
- TrvActionTableCellRightBorder⁽²⁸⁹⁾
- TrvActionTableCellBottomBorder⁽²⁸⁶⁾
- TrvActionTableCellAllBorders⁽²⁸⁵⁾
- TrvActionTableCellNoBorders⁽²⁸⁸⁾

1.3.8.15 TrvActionTableCellVAlign

TrvActionTableCellVAlign is a base class for the actions changing a vertical alignment in table cells.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellVAlign = class (TrvActionTableMultiCellAttributes(303))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction

```

TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes ⁽³⁰³⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

This action is not used directly. The following actions are inherited from it:

- *TrvActionTableCellVAlignTop* ⁽²⁹⁶⁾
- *TrvActionTableCellVAlignMiddle* ⁽²⁹⁵⁾
- *TrvActionTableCellVAlignBottom* ⁽²⁹⁴⁾
- *TrvActionTableCellVAlignDefault* ⁽²⁹⁵⁾

1.3.8.16 TrvActionTableCellVAlignBottom

TrvActionTableCellVAlignBottom is the action for "Table | Align Cell to the Bottom" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellVAlignBottom = class (TrvActionTableCellVAlign (293))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes ⁽³⁰³⁾
TrvActionTableCellVAlign ⁽²⁹³⁾

Description

The action aligns contents of the selected table cells (or the edited cell) to bottom. It changes `table.Cells[].VAlign` to `rvcBottom`.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

See also:

- *TrvActionTableCellVAlignTop* ⁽²⁹⁶⁾
- *TrvActionTableCellVAlignMiddle* ⁽²⁹⁵⁾
- *TrvActionTableCellVAlignDefault* ⁽²⁹⁵⁾

1.3.8.17 TrvActionTableCellVAlignDefault

TrvActionTableCellVAlignDefault is the action for "Table | Default Cell Vertical Alignment" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellVAlignDefault = class(TrvActionTableCellVAlign(293))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾
TrvActionTableMultiCellAttributes⁽³⁰³⁾
TrvActionTableCellVAlign⁽²⁹³⁾

Description

The action sets the default alignment for the selected table cells (or the edited cell). It changes table.Cells[].VAlign to *rvcVDefault*. In this mode, cells use table.Rows[].VAlign.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableCellVAlignTop⁽²⁹⁶⁾
- TrvActionTableCellVAlignMiddle⁽²⁹⁵⁾
- TrvActionTableCellVAlignBottom⁽²⁹⁴⁾

1.3.8.18 TrvActionTableCellVAlignMiddle

TrvActionTableCellVAlignMiddle is the action for "Table | Align Cell to the Middle" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellVAlignMiddle = class(TrvActionTableCellVAlign(293))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

[TrvCustomAction](#)⁽³³⁵⁾
[TrvAction](#)⁽³¹³⁾
[TrvActionTableCell](#)⁽²⁸⁴⁾
[TrvActionTableMultiCellAttributes](#)⁽³⁰³⁾
[TrvActionTableCellVAlign](#)⁽²⁹³⁾

Description

The action aligns contents of the selected table cells (or the edited cell) to the middle. It changes `table.Cells[].VAlign` to `rvcMiddle`.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of `TrvAction`.

See also:

- [TrvActionTableCellVAlignTop](#)⁽²⁹⁶⁾
- [TrvActionTableCellVAlignBottom](#)⁽²⁹⁴⁾
- [TrvActionTableCellVAlignDefault](#)⁽²⁹⁵⁾

1.3.8.19 TrvActionTableCellVAlignTop

`TrvActionTableCellVAlignTop` is the action for "Table | Align Cell to the Top" command.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionTableCellVAlignTop = class (TrvActionTableCellVAlign(293))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
[TrvCustomAction](#)⁽³³⁵⁾
[TrvAction](#)⁽³¹³⁾
[TrvActionTableCell](#)⁽²⁸⁴⁾
[TrvActionTableMultiCellAttributes](#)⁽³⁰³⁾
[TrvActionTableCellVAlign](#)⁽²⁹³⁾

Description

The action aligns contents of the selected table cells (or the edited cell) to top. It changes `table.Cells[].VAlign` to `rvcTop`.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of `TrvAction`.

See also:

- [TrvActionTableCellVAlignMiddle](#)⁽²⁹⁵⁾
- [TrvActionTableCellVAlignBottom](#)⁽²⁹⁴⁾
- [TrvActionTableCellVAlignDefault](#)⁽²⁹⁵⁾

1.3.8.20 TrvActionTableDeleteCols

TrvActionTableDeleteCols is the action for "Table | Delete Columns" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableDeleteColumns = class (TrvActionTableRCBase(306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾
TrvActionTableRCBase⁽³⁰⁶⁾

Description

This action deletes all table columns containing selected cells (or the edited cell).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableDeleteRows⁽²⁹⁷⁾

1.3.8.21 TrvActionTableDeleteRows

TrvActionTableDeleteRows is the action for "Table | Delete Rows" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableDeleteRows = class (TrvActionTableRCBase(306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾
TrvActionTableRCBase⁽³⁰⁶⁾

Description

This action deletes all table rows containing selected cells (or the edited cell).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableDeleteCols⁽²⁹⁷⁾

1.3.8.22 TrvActionTableDeleteTable

TrvActionTableDeleteTable is the action for "Table | Delete Table" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableDeleteTable = class(TrvActionTableCell(284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾

Description

The action deletes the table at the caret position.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.8.23 TrvActionTableGrid

TrvActionTableGrid is the action for "Table | Show Grid Lines" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableGrid = class(TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

The action shows/hides grid lines in tables. Grid lines are shown in places of invisible borders (zero border width, or sides hidden by VisibleBorders properties).

The action includes/excludes rvoShowGridLines in Options of the target editor.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

1.3.8.24 TrvActionTableInsertColLeft

TrvActionTableInsertColLeft is the action for "Table | Insert Column Left" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableInsertColLeft = class (TrvActionTableRCBase (306))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction (335)
TrvAction (313)
TrvActionTableCell (284)
TrvActionTableRCBase (306)

```

Description

This action inserts one new column to the left of the selected cells (or of the edited cell).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableInsertRowsAbove ⁽³⁰⁰⁾
- TrvActionTableInsertRowsBelow ⁽³⁰¹⁾
- TrvActionTableInsertColRight ⁽²⁹⁹⁾

1.3.8.25 TrvActionTableInsertColRight

TrvActionTableInsertColRight is the action for "Table | Insert Column Right" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableInsertColRight = class (TrvActionTableRCBase (306))
```

Hierarchy

```

TObject
TPersistent

```

TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableRCBase ⁽³⁰⁶⁾

Description

This action inserts one new column to the right of the selected cells (or of the edited cell).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

See also:

- *TrvActionTableInsertRowsAbove* ⁽³⁰⁰⁾
- *TrvActionTableInsertRowsBelow* ⁽³⁰¹⁾
- *TrvActionTableInsertColLeft* ⁽²⁹⁹⁾

1.3.8.26 TrvActionTableInsertRowsAbove

TrvActionTableInsertRowsAbove is the action for "Table | Insert Row Above" command.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvActionTableInsertRowsAbove = class (TrvActionTableRCBase (306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableRCBase ⁽³⁰⁶⁾

Description

This action inserts one new row above the selected cells (or above the edited cell).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

See also:

- *TrvActionTableInsertRowsBelow* ⁽³⁰¹⁾
- *TrvActionTableInsertColLeft* ⁽²⁹⁹⁾
- *TrvActionTableInsertColRight* ⁽²⁹⁹⁾

1.3.8.27 TrvActionTableInsertRowsBelow

TrvActionTableInsertRowsBelow is the action for "Table | Insert Row Below" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableInsertRowsBelow = class (TrvActionTableRCBase(306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾
TrvActionTableRCBase⁽³⁰⁶⁾

Description

This action inserts new rows below the selected cells (or below the edited cell).

When inserting rows to the end of the table, if AllowMultiple⁽³⁰²⁾ = *True*, the action requests a number of rows to insert and inserts this count of rows.

See also:

- TrvActionTableInsertRowsAbove⁽³⁰⁰⁾
- TrvActionTableInsertColLeft⁽²⁹⁹⁾
- TrvActionTableInsertColRight⁽²⁹⁹⁾

1.3.8.27.1 Properties

In TrvActionTableInsertRowsBelow

■ AllowMultiple⁽³⁰²⁾

Derived from TrvAction⁽³¹³⁾

■ Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

■ Caption⁽³³⁶⁾

■ ControlPanel⁽³³⁷⁾

■ Disabled⁽³³⁷⁾

■ Hint⁽³³⁷⁾

Derived from TAction

■ AutoCheck

■ Caption

- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.8.27.1.1 TrvActionTableInsertRowsBelow.AllowMultiple

Allowing inserting several rows at the end of table at once.

property AllowMultiple: Boolean;

If *True*, when inserting rows to the end of the table, the number of new rows is requested from the user.

Default value:

True

1.3.8.28 TrvActionTableMergeCells

TrvActionTableMergeCells is the action for "Table | Merge Cells" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionTableMergeCells = **class** (TrvActionTableCell⁽²⁸⁴⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾

Description

The action merges the selected table cells in one.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.8.29 TrvActionTableMultiCellAttributes

TrvActionTableMultiCellAttributes is the base class for actions changing properties of table cells.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableMultiCellAttributes = class (TrvActionTableCell(284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾

Description

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionTableCellVAlign⁽²⁹³⁾
- TrvActionTableCellBorder⁽²⁸⁶⁾

1.3.8.30 TrvActionTableProperties

TrvActionTableProperties is the action for "Table | Table Properties" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionTableProperties = class (TrvActionTableCell(284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionTableCell⁽²⁸⁴⁾

Description

The action displays a dialog for editing table properties. If the table has no selection (i.e. the caret is placed to the right or to the left of the table), this dialog has one page ("Table"). If there is a multicell selection or an edited cell, the dialog has additional pages ("Rows", "Columns", "Cells").

The same dialog is shown by TrvActionItemProperties⁽³²⁶⁾ action. But TrvActionItemProperties⁽³²⁶⁾ edits properties of an item at the position of caret. For example, if a picture inside a table is selected, TrvActionItemProperties⁽³²⁶⁾ edits this picture, while TrvActionTableProperties edits this table.

The user can check "Default" check box in this dialog to assign properties of new tables.

When the action assigns background images of tables and cells, the action assigns their BackgroundImageFileName properties, if *rvoAssignImageFileNames* is included in the Options property of the target editor.

1.3.8.30.1 Properties

In TrvActionTableProperties

- ActionInsertTable⁽³⁰⁵⁾
- BackgroundGraphicFilter⁽³⁰⁵⁾
- DefaultChecked⁽³⁰⁵⁾
- DefaultPersistent⁽³⁰⁵⁾
- UpdateAllInsertTableActions⁽³⁰⁵⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut

■ Visible

1.3.8.30.1.1 TrvActionTableProperties.ActionInsertTable, UpdateAllInsertTableActions

ActionInsertTable specifies the "insert table" action for applying default properties to.

UpdateAllInsertTableActions allows applying to all actions "insert table" actions on the same form/datamodule.

```
property ActionInsertTable: TrvActionInsertTable(273);
property UpdateAllInsertTableActions: Boolean;
```

If "Default" checkbox is checked, when properties are applied, they are assigned to "insert table" action(s) as well, so they become default properties for new tables.

If **ActionInsertTable** is defined, properties are applied only to this action. If this link is not defined, they are applied to the first found TrvActionInsertTable⁽²⁷³⁾ action on the same form/datamodule (if **UpdateAllInsertTableActions=False**), or to all TrvActionInsertTable⁽²⁷³⁾ actions on the same form/datamodule (if **UpdateAllInsertTableActions=True**)

Note: Report Workshop includes an additional action for inserting a report table:

TrvrActionInsertTable, inherited from TrvActionInsertTable⁽²⁷³⁾. If you want to apply changes both to instances of TrvrActionInsertTable and TrvActionInsertTable⁽²⁷³⁾, **UpdateAllInsertTableActions** must be True.

Default value

UpdateAllInsertTableActions = *True*

See also

- DefaultChecked, DefaultPersistent⁽³⁰⁵⁾
- TRVActionItemProperties⁽³²⁶⁾.ActionInsertTable⁽³²⁸⁾

1.3.8.30.1.2 TrvActionTableProperties.BackgroundGraphicFilter

Determines the file masks (filters) available in the file opening dialog displayed to open a picture to assign to a background picture of table or selected cells.

```
property BackgroundGraphicFilter: TRVALocString(350);
```

If this property is not empty, it is assigned to the Filter property of the file opening dialog (TOpenPictureDialog). Otherwise, a filter containing all registered graphic formats is created.

Default value:

" (empty string)

See also:

- TrvActionInsertPicture.Filter⁽²⁵⁴⁾
- TrvActionItemProperties.GraphicFilter⁽³³⁰⁾
- TrvActionItemProperties.BackgroundGraphicFilter⁽³²⁹⁾

1.3.8.30.1.3 TrvActionTableProperties.DefaultChecked, DefaultPersistent

These properties define the state of "Default" checkbox in the dialog.

```
property DefaultChecked: Boolean;
property DefaultPersistent: Boolean
```

If "Default" checkbox is checked, when properties are applied, they are assigned to "insert table" action(s) as well, so they become default properties for new tables.

DefaultChecked specifies the initial value of this checkbox (when the form is shown).

if **DefaultPersistent** = *True*, when the user pressed "OK" in the dialog, **DefaultChecked** property is updated according to the checkbox state (so it will have the same state when the dialog will be shown next time).

Default value

False

See also

- ActionInsertTable, UpdateAllInsertTableActions ⁽³⁰⁵⁾
- TRVActionItemProperties ⁽³²⁶⁾.DefaultChecked, DefaultPersistent ⁽³²⁹⁾

1.3.8.31 TrvActionTableRCBase

TrvActionTableRCBase is a base class for the actions for insertion and deletion of table rows and columns.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableRCBase = class (TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableInsertRowsAbove ⁽³⁰⁰⁾
- TrvActionTableInsertRowsBelow ⁽³⁰¹⁾
- TrvActionTableInsertColLeft ⁽²⁹⁹⁾
- TrvActionTableInsertColRight ⁽²⁹⁹⁾
- TrvActionTableDeleteRows ⁽²⁹⁷⁾
- TrvActionTableDeleteCols ⁽²⁹⁷⁾
- TrvActionTableSelectCols ⁽³⁰⁷⁾
- TrvActionTableSelectRows ⁽³⁰⁸⁾

1.3.8.32 TrvActionTableSelectCell

TrvActionTableSelectCell is the action for "Table | Select Cell" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSelectCell = class (TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

The action selects the edited table cell.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableSelectCols ⁽³⁰⁷⁾
- TrvActionTableSelectRows ⁽³⁰⁸⁾
- TrvActionTableSelectTable ⁽³⁰⁸⁾

1.3.8.33 TrvActionTableSelectCols

TrvActionTableSelectCols is the action for "Table | Select Columns" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSelectCols = class (TrvActionTableRCBase (306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableRCBase ⁽³⁰⁶⁾

Description

The action selects table columns containing selected cells (or the edited cell).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableSelectRows ⁽³⁰⁸⁾
- TrvActionTableSelectCell ⁽³⁰⁷⁾
- TrvActionTableSelectTable ⁽³⁰⁸⁾

1.3.8.34 TrvActionTableSelectRows

TrvActionTableSelectRows is the action for "Table | Select Rows" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSelectRows = class(TrvActionTableRCBase (306))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾
TrvActionTableRCBase ⁽³⁰⁶⁾

Description

The action selects table rows containing selected cells (or the edited cell).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionTableSelectCols ⁽³⁰⁷⁾
- TrvActionTableSelectCell ⁽³⁰⁷⁾
- TrvActionTableSelectTable ⁽³⁰⁸⁾

1.3.8.35 TrvActionTableSelectTable

TrvActionTableSelectTable is the action for "Table | Select Table" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSelectTable = class(TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

The action selects all cells in the table at the caret position.

Note: a selection containing a table, and a selection containing all selected table cells are different selections, although they look identically.

When all cells are selected (for example, after execution of this action), the user can perform table operations (for example, inserting rows or merging cells). When the user presses **Delete** key, all cells are cleared. *TCustomRichViewEdit* can copy such selection to the Clipboard in RVF and text format, but cannot copy in RTF format yet.

When the table is selected, the user cannot perform table operations. When the user presses **Delete** key, the table is deleted. *TCustomRichViewEdit* can copy such selection to the Clipboard in all supported formats.

See also:

- *TrvActionTableSelectCols* ⁽³⁰⁷⁾
- *TrvActionTableSelectRows* ⁽³⁰⁸⁾
- *TrvActionTableSelectCell* ⁽³⁰⁷⁾

1.3.8.36 TrvActionTableSort

TrvActionTableSort is the action for "Table | Sort" command.

Unit *RichViewActions* ⁽⁹⁴⁾ ;

Syntax

```
TrvActionTableSort = class (TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

The action sorts the table rows. If the action is called when more than one row are selected, it sorts the selected rows; otherwise, it sorts all rows.

Table heading rows are never sorted. Additionally, you can exclude the top [selected] row from sorting.

The action displays a dialog window allowing to specify parameters for sorting:

- column to sort by;
- ascending/descending order;
- sort as numbers or as text; if as text, case sensitivity

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

1.3.8.37 TrvActionTableSplit

TrvActionTableSplit is the action for "Table | Split Table" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSplit = class(TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

This action splits the current table into two tables. The second table starts from the first selected row.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of TrvAction.

1.3.8.38 TrvActionTableSplitCells

TrvActionTableSplitCells is the action for "Table | Split Cells" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableSplitCells = class(TrvActionTableCell (284))
```

Hierarchy

TObject

TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

The action displays a dialog where the user can choose:

- to unmerge the selected cells (if the selection contains cells having values of ColSpan or RowSpan properties greater than 1);
- to split the selected cells into the specified number of rows and columns (optionally with preliminary merging).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

1.3.8.39 TrvActionTableToText

TrvActionTableToText is the action for "Table | Convert to Text" command.

Unit RichViewActions ⁽⁹⁴⁾;

Syntax

```
TrvActionTableToText = class (TrvActionTableCell (284))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾
TrvActionTableCell ⁽²⁸⁴⁾

Description

This action converts the current table to text (in other words, it removes the table without removing its content).

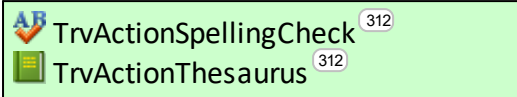
The action allows choosing a delimiter to insert between text from cells: line break, tab, semicolon, or comma. Delimiters are inserted using the current text style.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

1.3.9 Spelling check and thesaurus

Spelling Check and Thesaurus Actions

This group of actions allows using third-party spelling checkers and thesauri



1.3.9.1 TrvActionSpellingCheck

TrvActionSpellingCheck is the action for "Spell Check" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionSpellingCheck = class (TrvAction313)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
*TrvCustomAction*³³⁵
*TrvAction*³¹³

Description

This action is enabled if a spelling check interface component is assigned to ControlPanel³³⁷.SpellInterface⁵⁶.

The action starts a spelling checking process; when a misspelled word is found, a dialog is displayed to suggests a replacement.

This action does not introduce any new properties in addition to properties³¹⁴ of TrvAction.

1.3.9.2 TrvActionThesaurus

TrvActionThesaurus is the action for "Thesaurus" command.

Unit RichViewActions⁹⁴;

Syntax

```
TrvActionThesaurus = class (TrvAction313)
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action is enabled a spelling check interface component is assigned to *ControlPanel* ⁽³³⁷⁾. *SpellInterface* ⁽⁵⁶⁾, and this component supports a thesaurus.

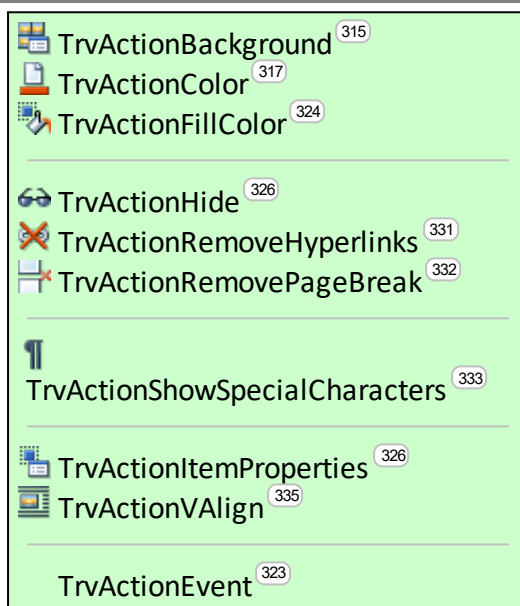
In this version, a thesaurus is provided only by *TRVAAddictSpellInterface* ⁽³⁵⁸⁾ component.

The action offers synonyms for the current word.

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

1.3.10 Other actions

Other Actions



1.3.10.1 TrvAction

TrvAction is a base class for all *RichViewActions* working with *TCustomRichViewEdit*.

Unit *RichViewActions* ⁽⁹⁴⁾;

Syntax

```
TrvAction = class (TrvCustomAction (335))
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾

Description

This action is not used directly, but all RichViewActions working with TCustomRichViewEdit are inherited from it.

1.3.10.1.1 Properties

In TrvAction

■ Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

■ Caption ⁽³³⁶⁾
 ■ ControlPanel ⁽³³⁷⁾
 ■ Disabled ⁽³³⁷⁾
 ■ Hint ⁽³³⁷⁾

Derived from TAction

■ AutoCheck
 ■ Caption
 ■ Checked
 DisableIfNoHandler
 ■ Enabled
 ■ GroupIndex
 ■ HelpContext
 ■ HelpKeyword
 ■ HelpType
 ■ Hint
 ■ ImageIndex
 ■ Name
 ■ SecondaryShortCuts
 ■ ShortCut
 ■ Visible

1.3.10.1.1.1 TrvAction.Control

Defines the editor for this action.

property Control: TCustomRVControl;

If an editor is assigned to this property, the action works with this component.

Controls of the following types can be assigned to this property:

- TRichViewEdit
- TDBRichViewEdit

- TSRichViewEdit (ScaleRichView)
- TDBSRichViewEdit (ScaleRichView)

Otherwise, if `GetControlPanel`⁽³³⁸⁾.`DefaultControl`⁽⁴³⁾ is assigned, the action works with it.

Otherwise, if an editor component is focused, the action works with it.

Otherwise, it works with the first found editor component.

Special actions

The following actions automatically assign this property, if it was not assigned before the first execution:

- TrvActionStyleInspector⁽²¹⁸⁾;
- TrvActionEditNote⁽²⁶²⁾.

They need this property assigned, because they display information for the current item in the target editor, even when the caret is moved after the execution of the action.

1.3.10.2 TrvActionBackground

TrvActionBackground is the action for "Background" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionBackground = class (TrvAction(313))
```

Hierarchy

```

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction(335)
TrvAction(313)

```

Description

This action changes background (Color, BackgroundPicture, BackgroundStyle) and "padding" (LeftMargin, TopMargin, RightMargin, BottomMargin) properties of the target editor. "Padding" can be changed only if `CanChangeMargins`⁽³¹⁶⁾ = `True`.

if `rvfoSaveBack` and/or `rvfoSaveLayout` are included in the target editor's `RVFOptions` property, the action changes background and/or margin properties as an editing operation (and these changes can be undone by the user).

The action calls the following events:

- TCustomRichViewEdit.OnChange (called by TCustomRichViewEdit.Change, see above)
- GetControlPanel⁽³³⁸⁾.OnMarginsChanged⁽⁶⁷⁾ (only if the action changed values of LeftMargin, TopMargin, RightMargin, BottomMargin properties of TCustomRichViewEdit)

- GetControlPanel⁽³³⁸⁾.OnBackgroundChange⁽⁶³⁾

1.3.10.2.1 Properties

In TrvActionBackground

- CanChangeMargins⁽³¹⁶⁾
- ▶ ImageFileName⁽³¹⁷⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.2.1.1 TrvActionBackground.CanChangeMargins

Specifies whether the action can change margin properties of the target TRichViewEdit control

property CanChangeMargins: Boolean;

If *True*, the "Padding" button is added to the action's dialog, allowing to define values for LeftMargin, RightMargin, TopMargin, BottomMargin properties of the target TRichViewEdit control.

It's recommended to hide this button when working with TScaleRichViewEdit controls from ScaleRichView.

Default value

True

1.3.10.2.1.2 TrvActionBackground.ImageFileName

Returns the file name (full path) of image file assigned to background bitmap.

property ImageFileName: TRVUnicodeString;

You can use this property to store this path somewhere.

The following events can be used for this:

- OnChange⁽³¹⁷⁾
- GetControlPanel⁽³³⁸⁾.OnBackgroundChange⁽⁶³⁾

1.3.10.2.2 Events

In TrvActionBackground

OnChange⁽³¹⁷⁾

1.3.10.2.2.1 TrvActionBackground.OnChange

Occurs after the background is changed by this action.

property OnChange: TRVAEditEvent⁽³⁸⁶⁾;

You can use either this event or GetControlPanel⁽³³⁸⁾.OnBackgroundChange⁽⁶³⁾.

1.3.10.3 TrvActionColor

TrvActionColor is the action for "Background Color" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionCustomColor = **class** (TrvActionCustomColor⁽³²⁰⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomColor⁽³²⁰⁾

Description

The action changes the target editor's Color property.

if *rvfoSaveBack* is included in the target editor's RVFOptions property, this change is made as an editing operation (and can be undone by the user)

The action calls GetControlPanel⁽³³⁸⁾.OnBackgroundChange⁽⁶³⁾ event.

This action does not introduce any new properties and events in addition to properties and events of `TrvActionCustomColor`⁽³²⁰⁾.

1.3.10.4 TrvActionUserDefinedColor

`TrvActionUserDefinedColor` allows changing a user-defined color.

Unit `RichViewActions`⁽⁹⁴⁾;

Syntax

```
TrvActionUserDefinedColor = class (TrvActionCustomColor(320))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾
TrvActionCustomColor⁽³²⁰⁾

Description

The action allows changing some color and (optionally) its opacity. You can provide the current color value in `OnGetColor`⁽³²⁰⁾ event, and apply a new color in `OnColorSelected`⁽³¹⁹⁾ event.

Assign `UseOpacity`⁽³¹⁹⁾ = `True` if you want to change a color opacity.

1.3.10.4.1 Properties

TrvActionUserDefinedColor

■ `UseOpacity`⁽³¹⁹⁾

Derived from TrvActionCustomColor⁽³²⁰⁾

■ `CallerControl`⁽³²¹⁾
 ■ `Color`⁽³²¹⁾
 ■ `Opacity`⁽³²²⁾
 ■ `UserInterface`⁽³²²⁾

Derived from TrvAction⁽³¹³⁾

■ `Control`⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

■ `Caption`⁽³³⁶⁾
 ■ `ControlPanel`⁽³³⁷⁾
 ■ `Disabled`⁽³³⁷⁾
 ■ `Hint`⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.4.1.1 TrvActionUserDefinedColor.UseOpacity

Specifies whether the action uses Opacity³²².

property UseOpacity: Boolean;

Default value:

False

1.3.10.4.2 Events

In TrvActionUserDefinedColor

- OnColorSelected³¹⁹
- OnGetColor³²⁰

Derived from TrvActionCustomColor³²⁰

- OnHideColorPicker³²³
- OnShowColorPicker³²³

1.3.10.4.2.1 TrvActionUserDefinedColor.OnColorSelected

Occurs when a color and an opacity have been changed by the user.

property OnColorSelected: TRVAEditEvent³⁸⁶;

This even occurs when the user changed values of a color and an opacity via user interface (an opacity can be changed only if UseOpacity³¹⁹ = *True*).

These values are already assigned to Color³²¹ and Opacity³²² properties.

1.3.10.4.2.2 TrvActionUserDefinedColor.OnGetColor

Occurs when the action needs to know the current values of a color and an opacity.

property OnGetColor: TRVAEditEvent⁽³⁸⁶⁾;

This even occurs when the action needs to initialize user interface controls with the proper values of a color and an opacity (an opacity is used only if UseOpacity⁽³¹⁹⁾ = *True*).

If you process this event, assign the proper values to Color⁽³²¹⁾ and Opacity⁽³²²⁾ properties.

1.3.10.5 TrvActionCustomColor

TrvActionCustomColor is a base class of the actions for color changing.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionCustomColor = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action is not used directly. The following actions are inherited from it:

- TrvActionFontCustomColor⁽¹⁵¹⁾
- TrvActionFontColor⁽¹⁵⁰⁾
- TrvActionFontBackColor⁽¹⁴⁹⁾
- TrvActionParaCustomColor⁽²⁰⁰⁾
- TrvActionParaColor⁽¹⁹⁷⁾
- TrvActionColor⁽³¹⁷⁾

1.3.10.5.1 Properties

In TrvActionCustomColor

- CallerControl⁽³²¹⁾
- Color⁽³²¹⁾
- Opacity⁽³²²⁾
- UserInterface⁽³²²⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.5.1.1 TrvActionCustomColor.CallerControl

Specifies the control (for example a toolbar button) which executed this action.

property CallerControl: TControl;

Assign this property if you use Delphi 5. For the newer versions of Delphi, the caller control is determined automatically (using ActionComponent property).

A color-picker window is positioned relative to this control. If this control is undefined, a color-picker window is displayed at the mouse pointer position.

1.3.10.5.1.2 TrvActionCustomColor.Color

Specifies the color to apply.

property Color: TColor;

See the topic about UserInterface ⁽³²²⁾ property for details.

Default value for TrvActionCustomColor:

c/None

Default value for TrvActionFontColor ⁽¹⁵⁰⁾:

c/WindowText

1.3.10.5.1.3 TrvActionCustomColor.Opacity

Specifies the color opacity to apply.

property Opacity: TRVOpacity;

See the topic about UserInterface⁽³²²⁾ property for details.

Not all actions inherited from TrvActionCustomColor use opacity.

Default value:

100000 (100% opaque)

1.3.10.5.1.4 TrvActionCustomColor.UserInterface

Defines the way how the user chooses a color.

type

```
TrvaColorInterface =  
(rvacNone, rvacColorDialog, rvacAdvanced);
```

property UserInterface: TrvaColorInterface;

Value	Meaning
<i>rvacNone</i>	Color is not chosen by the user. When the action is executed, values specified in Color ⁽³²¹⁾ and Opacity ⁽³²²⁾ * properties are applied.
<i>rvacColorDialog</i>	Color is chosen using TColorDialog (or GetControlPanel ⁽³³⁸⁾ .ColorDialogInterface ⁽⁴³⁾). Values of Color ⁽³²¹⁾ and Opacity ⁽³²²⁾ * properties before the execution is ignored (they are taken from the target editor). When the color and opacity are chosen, they are assigned to Color ⁽³²¹⁾ and Opacity ⁽³²²⁾ * properties. Note: TColorDialog cannot edit opacity. It can be edited only if GetControlPanel ⁽³³⁸⁾ .ColorDialogInterface ⁽⁴³⁾ supports it.
<i>rvacAdvanced</i>	Color is chosen using a special popup color-picker window. Values of Color ⁽³²¹⁾ and Opacity ⁽³²²⁾ * properties before the execution is ignored (they are taken from the target editor). When the color and opacity are chosen, they are assigned to Color ⁽³²¹⁾ and Opacity ⁽³²²⁾ * properties.

* only for actions that use opacity.

This action uses `GetControlPanel(338).ColorDialog(42)`, if assigned. Otherwise it creates a temporal color dialog.

In *rvacAdvanced* mode, `GetControlPanel(338).OnGetActionControlCoords(66)` may be used to position a color-picker window.

Default value:

rvacAdvanced

1.3.10.5.2 Events

In TrvActionCustomColor

- OnHideColorPicker⁽³²³⁾
- OnShowColorPicker⁽³²³⁾

1.3.10.5.2.1 TrvActionCustomColor.OnShowColorPicker

Occurs before the color-picker window is displayed.

property OnShowColorPicker: TNotifyEvent;

This event occurs if `UserInterface(322) = rvacAdvanced`.

1.3.10.5.2.2 TrvActionCustomColor.OnHideColorPicker

Occurs when the color-picker window is destroyed.

property OnHideColorPicker: TNotifyEvent;

This event occurs if `UserInterface(322) = rvacAdvanced`.

1.3.10.6 TrvActionEvent

TrvActionEvent is the action to perform custom operations.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionEvent = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
 TPersistent
 TComponent
 TBasicAction
 TContainedAction
 TCustomAction
 TAction
 TrvCustomAction⁽³³⁵⁾
 TrvAction⁽³¹³⁾

Description

This action calls `OnExecute(324)` and `OnUpdate(324)` events allowing to implement custom commands.

This action does not have predefined Caption and Hint. Assign these properties yourself.

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also:

- TrvActionInsertText⁽²⁵⁸⁾

1.3.10.6.1 Events

In TrvActionEvent

- OnExecute⁽³²⁴⁾
- OnUpdate⁽³²⁴⁾

1.3.10.6.1.1 TrvActionEvent.OnExecute

Occurs when the action is executed.

type

```
TRVAEvent = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit) of object;
```

property OnExecute: TRVAEvent;

1.3.10.6.1.2 TrvActionEvent.OnUpdate

Occurs when the application is idle or when the action list updates.

type

```
TRVAEvent = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit) of object;
```

property OnUpdate: TRVAEvent;

Update values of Enabled and Checked properties (and may be values of other action's properties) in this event.

This event is not called if Disabled⁽³³⁷⁾=True, or GetControlPanel⁽³³⁸⁾.ActionsEnabled⁽⁴²⁾=False.

1.3.10.7 TrvActionFillColor

TrvActionFillColor is the action for "Fill Color" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionFillColor = class (TrvAction(313))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction
```


*TAction**TrvCustomAction* ⁽³³⁵⁾*TrvAction* ⁽³¹³⁾**Description**

This action can change a background color of:

- text (RVStyle.TextStyles[].BackColor),
- paragraph (RVStyle.ParaStyles[].Background.Color),
- table cell (table.Cells[].Color)
- table (table.Color).

For paragraphs, it can also change padding (RVStyle.ParaStyles[].Background.BorderOffsets).

The action displays a dialog where the user can define a color and an object for application.

This is a redundant action, because these properties can be changed by other actions. But it may be convenient to have an action allowing to change all these properties.

1.3.10.7.1 Properties**In TrvActionFillColor**

- AllowApplyingTo ⁽³²⁶⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾
- Disabled ⁽³³⁷⁾
- Hint ⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.7.1.1 TrvActionFillColor.AllowApplyingTo

Contains objects that can be changed by this action.

type

```
TRVAFillColorApplyToElement = (rvafcaText, rvafcaParagraph,
    rvafcaCell, rvafcaTable);
TRVAFillColorApplyToSet = set of TRVAFillColorApplyToElement;
```

property AllowApplyingTo: TRVAFillColorApplyToSet;

The action can change a background color only for objects listed in this property.

Default value:

[rvafcaText..rvafcaTable]

1.3.10.8 TrvActionHide

TrvActionHide switches visibility of the selected fragment.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionHide = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action makes the selected text and objects hidden (or shows them if they are already hidden).

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

See also

- TrvActionShowSpecialCharacters⁽³³³⁾

1.3.10.9 TrvActionItemProperties

TrvActionItemProperties is the action for "Object Properties" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionItemProperties = class (TrvAction(313))
```

Hierarchy

TObject

TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action displays a dialog for changing properties of items of the following types:

- pictures and "hot-pictures"
- "breaks" (horizontal lines)
- tables
- sidenotes and text box items
- numbered sequences
- page numbers and page counts
- other (using OnCustomItemPropertiesDialog ⁽³³¹⁾ event)

The user can check "Default" check box in this dialog to assign properties of new tables and horizontal lines.

If *rvoAssignImageFileNames* is included in the Options property of the target editor, path to the graphic file is stored in the document:

- in the *rvesplImageFileName* item property (see the TRichView help file, TRVExtraItemStrProperty type) for tables, pictures and "hot-pictures";
- in BackgroundImageFileName cell property for table cells.

See also:

- TrvActionTableProperties ⁽³⁰³⁾

1.3.10.9.1 Properties

In TrvActionItemProperties

- ActionInsertHLine ⁽³²⁸⁾
- ActionInsertTable ⁽³²⁸⁾
- BackgroundGraphicFilter ⁽³²⁹⁾
- DefaultChecked ⁽³²⁹⁾
- DefaultPersistent ⁽³²⁹⁾
- GraphicFilter ⁽³³⁰⁾
- UpdateAllInsertTableActions ⁽³²⁸⁾

Derived from TrvAction ⁽³¹³⁾

- Control ⁽³¹⁴⁾

Derived from TrvCustomAction ⁽³³⁵⁾

- Caption ⁽³³⁶⁾
- ControlPanel ⁽³³⁷⁾

- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.9.1.1 TrvActionItemProperties.ActionInsertHLine

Specifies the "insert horizontal line" action for applying default properties to.

property ActionInsertHLine: TrvActionInsertHLine⁽²³³⁾;

If this link is not defined, the first found TrvActionInsertHLine⁽²³³⁾ action on the same form/datamodule is used.

If "Default" checkbox is checked, when properties are applied, they are assigned to "insert table" and "insert horizontal line" action as well, so they become default properties for new tables / lines.

See also:

- ActionInsertTable, UpdateAllInsertTableActions⁽³²⁸⁾
- DefaultChecked, DefaultPersistent⁽³²⁹⁾

1.3.10.9.1.2 TrvActionItemProperties.ActionInsertTable, UpdateAllInsertTableActions

ActionInsertTable specifies the "insert table" action for applying default properties to.

UpdateAllInsertTableActions allows applying to all actions "insert table" actions on the same form/datamodule.

property ActionInsertTable: TrvActionInsertTable⁽²⁷³⁾;

property UpdateAllInsertTableActions: Boolean;

If "Default" checkbox is checked, when properties are applied, they are assigned to "insert table" and "insert horizontal line" action as well, so they become default properties for new tables / lines.

If **ActionInsertTable** is defined, properties are applied only to this action. If this link is not defined, they are applied to the first found TrvActionInsertTable⁽²⁷³⁾ action on the same form/datamodule (if **UpdateAllInsertTableActions=False**), or to all TrvActionInsertTable⁽²⁷³⁾ actions on the same form/datamodule (if **UpdateAllInsertTableActions=True**).

Note: Report Workshop includes an additional action for inserting a report table:

TrvrActionInsertTable, inherited from TrvActionInsertTable⁽²⁷³⁾. If you want to apply changes both to instances of TrvrActionInsertTable and TrvActionInsertTable⁽²⁷³⁾, **UpdateAllInsertTableActions** must be True.

Default value

UpdateAllInsertTableActions = *True*

See also:

- ActionInsertHLine⁽³²⁸⁾
- DefaultChecked, DefaultPersistent⁽³²⁹⁾
- TrvActionTableProperties⁽³⁰³⁾.ActionInsertTable, UpdateAllInsertTableActions⁽³⁰⁵⁾

1.3.10.9.1.3 TrvActionItemProperties.BackgroundGraphicFilter

Determines the file masks (filters) available in the file opening dialog displayed to open a picture to assign to a background picture of table or selected cells.

property BackgroundGraphicFilter: TRVLocString⁽³⁵⁰⁾;

If this property is not empty, it is assigned to the Filter property of the file opening dialog (TOpenPictureDialog). Otherwise, a filter containing all registered graphic formats is created.

Default value:

" (empty string)

See also:

- TrvActionInsertPicture.Filter⁽²⁵⁴⁾
- GraphicFilter⁽³³⁰⁾
- TrvActionTableProperties.BackgroundGraphicFilter⁽³⁰⁵⁾

1.3.10.9.1.4 TrvActionItemProperties.DefaultChecked, DefaultPersistent

These properties define the state of "Default" checkbox in the dialog.

property DefaultChecked: Boolean;

property DefaultPersistent: Boolean

If "Default" checkbox is checked, when properties are applied, they are assigned to "insert table" and "insert horizontal line" action as well, so they become default properties for new tables / lines.

DefaultChecked specifies the initial value of this checkbox (when the form is shown).

if **DefaultPersistent** = *True*, when the user pressed "OK" in the dialog, **DefaultChecked** property is updated according to the checkbox state (so it will have the same state when the dialog will be shown next time).

Default value

False

See also

- ActionInsertTable, UpdateAllInsertTableActions⁽³²⁸⁾
- ActionInsertHLine⁽³²⁸⁾
- TrvActionTableProperties⁽³⁰³⁾.DefaultChecked, DefaultPersistent⁽³⁰⁵⁾

1.3.10.9.1.5 TrvActionItemProperties.GraphicFilter

Determines the file masks (filters) available in the file opening dialog for pictures and "hot-pictures".

property GraphicFilter: TRVALocString⁽³⁵⁰⁾;

If this property is not empty, it is assigned to the Filter property of the file opening dialog (TOpenPictureDialog). Otherwise, a filter containing all registered graphic formats is created.

Default value:

" (empty string)

See also:

- TrvActionInsertPicture.Filter⁽²⁵⁴⁾
- TrvActionItemProperties.BackgroundGraphicFilter⁽³²⁹⁾
- TrvActionTableProperties.BackgroundGraphicFilter⁽³⁰⁵⁾

1.3.10.9.1.6 TrvActionItemProperties.StoreImageFileName

Specifies whether file names (full paths) are stored for images assigned by this action.

property StoreImageFileName: Boolean;

If this property is *True*, path to the graphic file is stored in the document:

- in the *rvesplImageFileName* item property (see the RichView help file, TRVExtraltemStrProperty type) for tables, pictures and "hot-pictures";
- in BackgroundImageFileName cell property for table cells.

This is a recommended way to store path to graphic files. You can use these file names when saving HTML files (include *rvhtmlsioUseItemImageFileNames* in HTMLSaveProperties.ImageOptions of the target editor).

Default value

False

See also:

- TCustomRichView.RTFReadProperties.StoreImagesFileNames
- HTML Import (for images inserted with HTML, paths to graphic files are always stored in *rvesplImageFileName*)

1.3.10.9.2 Events

In TrvActionItemProperties

- OnCanApply⁽³³⁰⁾
- OnCustomItemPropertiesDialog⁽³³¹⁾

1.3.10.9.2.1 TrvActionItemProperties.OnCanApply

Requests whether a custom dialog can be displayed for the given item.

type

```
TRVCanApplyEvent = procedure (Sender: TObject;
    Editor: TCustomRichViewEdit;
    Item: TCustomRVItemInfo; var CanApply: Boolean) of object;
```

property OnCanApply: TRVCanApplyEvent;

This event occurs only if OnCustomItemPropertiesDialog⁽³³¹⁾ event is assigned.

A dialog will be displayed for **Item** in the **Editor** (for the selected item, if only one item is selected), if **CanApply** is set to *True*.

1.3.10.9.2.2 TrvActionItemProperties.OnCustomItemPropertiesDialog

Allows displaying a dialog for editing properties of different item types, or replacing dialogs for items of the standard types, or forbidding displaying a standard dialog for items of standard types.

type

```
TRVCustomItemPropertiesDialog = procedure (Sender: TObject;  
    Editor: TCustomRichViewEdit; var DoDefault: Boolean) of object;
```

property OnCustomItemPropertiesDialog: TRVCustomItemPropertiesDialog;

A dialog must be displayed for the current item in the **Editor** (or for the selected item, if only one item is selected).

Assign *False* to **DoDefault** if you do not want to display a standard dialog after calling this event.

See also:

- OnCanApply⁽³³⁰⁾

1.3.10.10TrvActionRemoveHyperlinks

TrvActionRemoveHyperlinks is the action for "Remove Hyperlinks" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionRemoveHyperlinks = class (TrvAction(313))
```

Hierarchy

```
TObject  
TPersistent  
TComponent  
TBasicAction  
TContainedAction  
TCustomAction  
TAction  
TrvCustomAction(335)  
TrvAction(313)
```

Description

This action converts all hyperlinks in the selected fragment to a plain text. This action does the same work as TrvActionInsertHyperlink⁽²³⁴⁾, if the user entered an empty link target.

This action is a wrapper for TrvActionInsertHyperlink⁽²³⁴⁾ action referenced in ActionInsertHyperlink property.

1.3.10.10.1 Properties

In TrvActionRemoveHyperlinks

- ActionInsertHyperlink⁽³³²⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.10.1.1 TrvActionRemoveHyperlinks.ActionInsertHyperlink

Links this action to TrvActionInsertHyperlink⁽²³⁴⁾ action.

property ActionInsertHyperlink: TrvActionInsertHyperlink⁽²³⁴⁾;

If this link is not defined, the first found TrvActionInsertHyperlink⁽²³⁴⁾ action on the same form/datamodule is used. If no TrvActionInsertHyperlink⁽²³⁴⁾ action found, this action is disabled.

1.3.10.11TrvActionRemovePageBreak

TrvActionRemovePageBreak is the action for "Remove Page Break" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvActionRemovePageBreak = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

This action removes a page break from the paragraph at the caret position (or, when called from a table cell, from the current table row).

This action does not introduce any new properties in addition to properties ⁽³¹⁴⁾ of *TrvAction*.

See also:

- *TrvActionInsertPageBreak* ⁽²⁵⁰⁾

1.3.10.12TrvActionShowSpecialCharacters

TrvActionShowSpecialCharacters is the action for "Non-printing Characters" command.

Unit RichViewActions ⁽⁹⁴⁾ ;

Syntax

```
TrvActionShowSpecialCharacters = class (TrvAction (313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction ⁽³³⁵⁾
TrvAction ⁽³¹³⁾

Description

The action shows/hides:

- marks for non-printing characters (spaces, non breaking spaces, tabs, soft hyphens, paragraphs separators, line separators) and placeholders of floating items;
- hidden items;
- (if ShowCheckpoints ⁽³³⁴⁾ = True) checkpoints.

The set of non-printing characters to display is defined in *RVVisibleSpecialCharacters* global variable (declared in *RVStyle.pas*).

The action includes/excludes *rvoShowSpecialCharacters*, *rvoShowHiddenText*, *rvoShowCheckpoints* from *TCustomRichViewEdit.Options* property and reformats *TCustomRichViewEdit*.

This action assumes that these options are included and excluded from Options together. Make sure that it is true for the initial state of editors in your application.

See also

- TrvActionHide ³²⁶

1.3.10.12.1 Properties

In TrvActionShowSpecialCharacters

- ShowCheckpoints ³³⁴

Derived from TrvAction ³¹³

- Control ³¹⁴

Derived from TrvCustomAction ³³⁵

- Caption ³³⁶
- ControlPanel ³³⁷
- Disabled ³³⁷
- Hint ³³⁷

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.12.1.1 TrvActionShowSpecialCharacters.ShowCheckpoints

Defines whether the action hides/shows checkpoints in the target editor.

property ShowCheckpoints: Boolean;

If set, the action, in addition to switching other options, includes/excludes rvoShowCheckpoints in the Options property of the target editor.

Assign *False* to this property if you want checkpoints to be always shown/hidden, or if you want to implement their showing/hiding in other place of your application.

Default value*True***1.3.10.13TrvActionVAlign**

TrvActionVAlign is the action for "Object Position" command.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

```
TrvActionVAlign = class (TrvAction(313))
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

This action displays a dialog for changing VAlign property for the items of the following types:

- pictures
- *hot-pictures*
- *bullets* (image lists)
- *hotspots* (image lists)
- controls
- label items
- numbered sequences
- custom item types inherited from TRVRectItemInfo

This action does not introduce any new properties in addition to properties⁽³¹⁴⁾ of TrvAction.

1.3.10.14TrvCustomAction

TrvCustomAction is a base class for all RichViewActions.

Unit RichViewActions⁽⁹⁴⁾;

Syntax (normal)

```
TrvCustomAction = class (TAction)
```

Syntax (if TNT Controls⁽³⁷²⁾ are used)

```
TrvCustomAction = class (TAction, ITntAction)
```

Hierarchy

TObject
TPersistent

TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

Description

This action is not used directly, but all RichViewActions are inherited from it.

Main properties:

- ControlPanel⁽³³⁷⁾ links this action to a control panel defining additional properties for the action.
- Disabled⁽³³⁷⁾ allows to disable this action.

Direct descendants:

- TrvAction⁽³¹³⁾
- TrvActionPageSetup⁽¹¹⁶⁾

1.3.10.14.1 Properties

In TrvCustomAction

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
- DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.14.1.1 TrvCustomAction.Caption

Represents the caption of client controls and menu items.

If TNT Controls⁽³⁷²⁾ are used, this property overrides the standard action's Caption property:

```
property Caption: WideString;
```

Caption holds the string that is the caption for the action when it is set. The value of Caption is propagated to client controls and menu items.

See also properties:

- Hint⁽³³⁷⁾

1.3.10.14.1.2 TrvCustomAction.ControlPanel

Links this action with a control panel.

property ControlPanel: TRVAControlPanel⁽³⁹⁾;

A control panel defines properties used for many actions.

If this property is not assigned (*nil*), the action uses MainRVAControlPanel⁽³⁹⁰⁾.

See also:

- GetControlPanel⁽³³⁸⁾.

1.3.10.14.1.3 TrvCustomAction.Disabled

Allows to disable the action.

property Disabled: Boolean;

If *False*, the action is disabled and enabled automatically.

If *True*, the action is disabled.

Default value:

False

See also properties:

- Enabled

See also:

- GetControlPanel⁽³³⁸⁾.ActionsEnabled⁽⁴²⁾

1.3.10.14.1.4 TrvCustomAction.Hint

Indicates the Help hint for client controls and menu items.

If TNT Controls⁽³⁷²⁾ are used, this property overrides the standard action's Hint property.

property Hint: WideString;

Hint holds the string that is the hint for that action when it is set. This value is propagated to client controls and menu items.

See also properties:

- Caption⁽³³⁶⁾

1.3.10.14.2 Methods

In TrvCustomAction

GetControlPanel⁽³³⁸⁾

1.3.10.14.2.1 TrvCustomAction.GetControlPanel

Returns a control panel used by this action.

function GetControlPanel: TRVAControlPanel⁽³⁹⁾;

If ControlPanel⁽³³⁷⁾ is assigned, the method returns it. Otherwise it returns MainRVAControlPanel⁽³⁹⁰⁾.

1.3.10.15TrvCustomEditorAction

TrvCustomEditorAction is a base class for action displaying a non-modal window containing TRichViewEdit component.

Unit RichViewActions⁽⁹⁴⁾;

Syntax

TrvCustomEditorAction = **class** (TrvAction⁽³¹³⁾)

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction⁽³³⁵⁾
TrvAction⁽³¹³⁾

Description

When executed, this action shows Form⁽³³⁹⁾ containing SubDocEditor⁽³³⁹⁾.

This class is not used directly. The following actions are inherited from TrvCustomEditorAction:

- TrvActionEditNote⁽²⁶²⁾

1.3.10.15.1 Properties

In TRVCustomEditorAction

- ▶ Form⁽³³⁹⁾
- ▶ SubDocEditor⁽³³⁹⁾

Derived from TrvAction⁽³¹³⁾

- Control⁽³¹⁴⁾

Derived from TrvCustomAction⁽³³⁵⁾

- Caption⁽³³⁶⁾
- ControlPanel⁽³³⁷⁾
- Disabled⁽³³⁷⁾
- Hint⁽³³⁷⁾

Derived from TAction

- AutoCheck
- Caption
- Checked
 - DisableIfNoHandler
- Enabled
- GroupIndex
- HelpContext
- HelpKeyword
- HelpType
- Hint
- ImageIndex
- Name
- SecondaryShortCuts
- ShortCut
- Visible

1.3.10.15.1.1 TrvCustomEditorAction.Form

Returns a form containing TRichViewEdit control.

property Form: TForm;

This form is created when the action is executed for the first time. This form is destroyed together with the action: on closing, it only hides itself.

See also

- OnFormCreate ⁽³⁴⁰⁾
- OnShowing and OnHide ⁽³⁴⁰⁾
- SubDocEditor ⁽³³⁹⁾

1.3.10.15.1.2 TrvCustomEditorAction.SubDocEditor

Returns a TRichViewEdit control inserted in Form ⁽³³⁹⁾.

property SubDocEditor: TRichViewEdit;

This control is created and destroyed together with Form ⁽³³⁹⁾,

1.3.10.15.2 Events

In TRVCustomEditorAction

- OnFormCreate ⁽³⁴⁰⁾
- OnHide ⁽³⁴⁰⁾
- OnShowing ⁽³⁴⁰⁾

1.3.10.15.2.1 TrvCustomEditorAction.OnFormCreate

Occurs after the Form⁽³³⁹⁾ is created.

property OnFormCreate: TRVShowFormEvent⁽³⁹⁰⁾;

See also:

- OnShowing, OnHide⁽³⁴⁰⁾ (contains example)

1.3.10.15.2.2 TrvCustomEditorAction.OnShowing, OnHide

Occurs before Form⁽³³⁹⁾ is shown, and when it is hidden.

property OnShowing: TRVShowFormEvent⁽³⁹⁰⁾;

property OnHide: TNotifyEvent;

Example

This example shows how to work with a form displayed by TrvActionEditNote⁽²⁶²⁾.

Let the main form TfrmMain have:

- RichViewEdit: TRichViewEdit – the main editor;
- RVAControlPanel1: TRVAControlPanel⁽³⁹⁾;
- RVAPopupMenu1: TRVAPopupMenu⁽⁶⁸⁾;
- rvActionEditNote1: TrvActionEditNote;
- rvActionStyleInspector1: TrvActionStyleInspector⁽²¹⁸⁾;
- cmbFont: TRVFontComboBox⁽⁷⁵⁾;
- cmbFontSize: TRVFontSizeComboBox⁽⁸¹⁾;
- cmbStyles: TRVStyleTemplateComboBox⁽⁸⁹⁾.

We want to insert Form⁽³³⁹⁾ at the bottom of TfrmMain.

Initial property settings:

- RVAControlPanel1.DefaultControl⁽⁴³⁾ := RichViewEdit1;
- rvActionStyleInspector1.Control⁽³¹⁴⁾ := RichViewEdit1;
- cmbFont.Editor⁽³⁵⁾ := RichViewEdit1;
- cmbFontSize.Editor⁽³⁵⁾ := RichViewEdit1;
- cmbStyles.Editor⁽⁹¹⁾ := RichViewEdit1

// OnFormCreate⁽³⁴⁰⁾ event

procedure TfrmMain.rvActionEditNoteFormCreate(Sender: TrvAction⁽³¹³⁾;
Form: TForm);

begin

// assigning properties of the note editor
rvActionEditNote1.SubDocEditor.OnEnter := NoteEditorEnter;
rvActionEditNote1.SubDocEditor.OnExit := NoteEditorExit;
rvActionEditNote1.SubDocEditor.PopupMenu := RVAPopupMenu1;
// moving the form to the bottom of frmMain
Form.Align := alBottom;
Form.Height := 100;
Form.Parent := Self;

end;

// OnShowing event


```
procedure TfrmMain.rvActionEditNoteShowing(Sender: TrvAction;  
    Form: TForm);  
begin  
    // leaving the current note in RichViewEdit1 highlighted  
    // after moving the input focus  
    RichViewEdit1.ForceFieldHighlight := True;  
end;
```

```
// OnHide event  
procedure TfrmMain.rvActionEditNoteHide(Sender: TObject);  
begin  
    // restoring old values of properties  
    RichViewEdit1.ForceFieldHighlight := False;  
    // focusing the main editor  
    if RichViewEdit1.CanFocus then  
        RichViewEdit1.SetFocus;  
end;
```

```
// OnEnter: occurs when the input focus is moved to the note editor  
procedure TfrmMain.NoteEditorEnter(Sender: TObject);  
begin  
    // making the note editor default  
    RVAControlPanell1.DefaultControl :=  
        rvActionsResource.rvActionEditNote1.SubDocEditor;  
    UpdateLinkedControls;  
end;
```

```
// OnExit: making the main editor default  
procedure TfrmMain.NoteEditorExit(Sender: TObject);  
begin  
    RVAControlPanell1.DefaultControl := RichViewEdit1;  
    UpdateLinkedControls;  
end;
```

```
procedure TfrmMain.UpdateLinkedControls;  
begin  
    rvActionStyleInspector1.Control := RVAControlPanell1.DefaultControl;  
    cmbFont.Editor := RVAControlPanell1.DefaultControl;  
    cmbFontSize.Editor := RVAControlPanell1.DefaultControl;  
    cmbStyles.Editor := RVAControlPanell1.DefaultControl;  
end;
```

Note: a note editor is special, it redirects many actions (for example, TrvActionNew⁽¹⁰⁵⁾, TrvActionSave⁽¹²⁴⁾, TrvActionPrint⁽¹¹⁸⁾) to the main editor.

1.4 Localization of RichViewActions

Changing UI language

RichViewActions require localization, even if you use only one UI language.

Initially, the language specified in `TRVAControlPanel.Language`⁽⁵³⁾ 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`⁽³⁵²⁾ to localize them.

For changing UI language, use `RVA_ChooseLanguage`⁽³⁵¹⁾ function. The topic about this function contains an example.

Other localization functions are declared in `RVALocalize`⁽³⁴³⁾ unit.

Hiding untranslated controls

It is possible that some new features added to RichViewActions were not translated to all languages yet.

Such controls can be hidden by `ShowUntranslatedControls`⁽³⁹²⁾ global variable.

Excluding languages

You can reduce the size of your application by excluding some languages.

To exclude languages, define one more of the following compiler defines in the options of your project:

- `RVA_NO_ARMENIAN`
- `RVA_NO_BULGARIAN`
- `RVA_NO_BYELORUSSIAN`
- `RVA_NO_CATALAN`
- `RVA_NO_CHINESE_SIMPL`
- `RVA_NO_CHINESE_TRAD`
- `RVA_NO_CZECH`
- `RVA_NO_DANISH`
- `RVA_NO_DUTCH`
- `RVA_NO_ENGLISH_US`
- `RVA_NO_FARSI`
- `RVA_NO_FINNISH`
- `RVA_NO_FRENCH`
- `RVA_NO_GERMAN`
- `RVA_NO_HINDI`
- `RVA_NO_HUNGARIAN`
- `RVA_NO_ITALIAN`
- `RVA_NO_LITHUANIAN`
- `RVA_NO_MALAY`
- `RVA_NO_NORWEGIAN`
- `RVA_NO_POLISH`
- `RVA_NO_PORTUGUESE_BR`

- RVA_NO_PORTUGUESE_PT
- RVA_NO_ROMANIAN
- RVA_NO_RUSSIAN
- RVA_NO_SLOVAK
- RVA_NO_SLOVENE
- RVA_NO_SPANISH
- RVA_NO_SWEDISH
- RVA_NO_THAI
- RVA_NO_TURKISH
- RVA_NO_UKRAINIAN

Resource language

There is one special language: "Resource language". It contains the same string as in "English (US)", but they are stored in resources. You can use this language together with an external localization tool.

By default, this language is excluded. You can enable it by removing the following line from RichViewActions.inc:

```
{#DEFINE RVA_NO_RESOURCELANGUAGE}
```

Adding a new language

If you want to add a translation to you language (for example, Greek):

1. Open RVAL_EngUS.pas, rename it to RVAL_Greek.pas. Add RVAL_Greek in "uses" of RVALocalize.pas. (If you use Delphi/C++Builder 2009, save this unit as UTF-8).
2. Translate all text in this unit (do not translate comments!)
3. Change the call of RVA_RegisterLanguage³⁴⁹ to RVA_RegisterLanguage('Greek', Ελληνικά', GREEK_CHARSET, @Messages).
4. Remove the call of RVA_SwitchLanguage³⁴⁹.
5. Send the translated file to svt@trichview.com for including in the main installation of RichViewActions. Your file will be maintained (it will be corrected when new messages will be added in the future versions of RichViewActions)

1.4.1 RVALocalize unit

This unit contains procedures and functions working with a language of user interface in RichViewActions¹⁶.

Procedures and functions declared in RVALocalize

Procedures returning names of registered UI languages:

- RVA_EnumLanguages³⁴⁶
- RVA_FillLanguageList³⁴⁶

Functions returning properties of the current language:

- RVA_GetLanguageName³⁴⁸
- RVA_GetCharset³⁴⁷
- RVA_GetHelpFile³⁴⁷

Functions for help files:

- RVA_SetHelpFile ⁽³⁴⁹⁾

Functions returning a translated string:

- RVA_GetS, RVA_GetSH, RVA_GetPC ⁽³⁴⁸⁾
- RVA_TranslateUnits ⁽³⁴⁹⁾
- RVA_GetProgressMessage, RVA_GetPrintingMessage ⁽³⁴⁵⁾

Procedure for changing the current language:

- RVA_SwitchLanguage ⁽³⁴⁹⁾

Other procedures

The following procedures and functions are declared in other units, but related to localization:

- RVA_ChooseLanguage ⁽³⁵¹⁾ (displays a dialog to choose a language name, changes the current language)
- RVA_LocalizeForm ⁽³⁵²⁾ (localizes all actions on the specified form/datamodule)
- RVA_LocalizeRuler ⁽³⁵³⁾ (localizes TRVRuler ⁽⁸²⁾ component)
- RVA_GetColorName ⁽³⁵²⁾ (returns the color name)

Types declared in RVALocalize

- TRVALanguageName ⁽³⁵⁰⁾
- TRVALocString, TRVALocStrings, TRVALocStringList ⁽³⁵⁰⁾

Additional information

See also:

- Localization of RichViewActions ⁽³⁴²⁾

1.4.1.1 Procedures and functions

Procedures and functions declared in RVALocalize

Procedures returning names of registered UI languages:

- RVA_EnumLanguages ⁽³⁴⁶⁾
- RVA_FillLanguageList ⁽³⁴⁶⁾

Functions returning properties of the current language:

- RVA_GetLanguageName ⁽³⁴⁸⁾
- RVA_GetCharset ⁽³⁴⁷⁾
- RVA_GetHelpFile ⁽³⁴⁷⁾

Functions for help files:

- RVA_SetHelpFile ⁽³⁴⁹⁾

Functions returning a translated string:

- RVA_GetS, RVA_GetSH, RVA_GetPC ⁽³⁴⁸⁾
- RVA_TranslateUnits ⁽³⁴⁹⁾
- RVA_GetProgressMessage, RVA_GetPrintingMessage ⁽³⁴⁵⁾

Procedure for changing the current language:

- RVA_SwitchLanguage⁽³⁴⁹⁾

Other procedures

The following procedures and functions are declared in other units, but related to localization:

- RVA_ChooseLanguage⁽³⁵¹⁾ (displays a dialog to choose a language name, changes the current language)
- RVA_LocalizeForm⁽³⁵²⁾ (localizes all actions on the specified form/datamodule)
- RVA_LocalizeRuler⁽³⁵³⁾ (localizes TRVRuler⁽⁸²⁾ component)
- RVA_GetColorName⁽³⁵²⁾ (returns the color name)

1.4.1.1.1 Functions for progress messages

```
function RVA_GetProgressMessage(Operation: TRVLongOperation;
  ControlPanel: TComponent=nil): TRVALocString(350);
function RVA_GetPrintingMessage(PageCompleted: Integer;
  Step: TRVPrintingStep;
  ControlPanel: TComponent=nil): TRVALocString(350);
```

If **ControlPanel** is a TRVAControlPanel⁽³⁹⁾ component, the functions use it. Otherwise, they use MainRVAControlPanel⁽³⁹⁰⁾.

RVA_GetProgressMessage returns a localized text message describing the specified long operation. It can be used in TRichViewEdit.OnProgress event.

RVA_GetPrintingMessage returns a localized text message describing the current printing progress. It can be used in TRVPrint.OnSendingToPrinter event.

The name of the current language is returned as ControlPanel.Language⁽⁵³⁾, or by RVA_GetLanguageName⁽³⁴⁸⁾ function.

The current language can be changed by assigning a new value to ControlPanel.Language⁽⁵³⁾, or by calling RVA_SwitchLanguage⁽³⁴⁹⁾ procedure.

Example:

This example assumes that Application.Hint is displayed in a status bar.

```
// TRVAControlPanel.OnDownload(66)
procedure TForm3.RVAControlPanel1Download(Sender: TrvAction;
  const Source: TRVUnicodeString);
begin
  if Source='' then
    Application.Hint := ''
  else
    Application.Hint := Format(RVA_GetS(rvam_msg_Downloading),
      [Source]);
end;
{-----}
// TRVPrint.OnSendingToPrinter
procedure TForm3.RVPrint1SendingToPrinter(Sender: TCustomRichView;
  PageCompleted: Integer; Step: TRVPrintingStep);
begin
  Application.Hint := RVA_GetPrintingMessage(PageCompleted, Step);
end;
```

```

{-----}
// Reading or writing
procedure TForm3.RichViewEdit1Progress(Sender: TCustomRichView;
Operation: TRVLongOperation; Stage: TRVProgressStage;
PercentDone: Byte);
begin
  case Stage of
    rvpstgStarting:
      begin
        ProgressBar1.Position := 0;
        ProgressBar1.Visible := True;
        Application.Hint := RVA_GetProgressMessage(Operation);
      end;
    rvpstgRunning:
      begin
        ProgressBar1.Position := PercentDone;
        ProgressBar1.Update;
      end;
    rvpstgEnding:
      begin
        Application.Hint := '';
        ProgressBar1.Visible := False;
      end;
  end;
end;

```

1.4.1.1.2 RVA_EnumLanguages procedure

This procedure allows to enumerate all UI languages available for RichViewActions.

Unit RVALocalize⁽³⁴³⁾.

```

//type
// TGetStrProc = procedure (const S: string) of object;
procedure RVA_EnumLanguages(Proc: TGetStrProc);

```

This procedure calls **Proc** one time for each language name. Only English language names are passed to this procedure.

Names of languages can be assigned to TRVAControlPanel.Language⁽⁵³⁾ or used as a parameter for RVA_SwitchLanguage⁽³⁴⁹⁾ procedure.

See also:

- RVA_FillLanguageList⁽³⁴⁶⁾

1.4.1.1.3 RVA_FillLanguageList procedure

This procedure adds all UI language names available for RichViewActions in **sl**.

Unit RVALocalize⁽³⁴³⁾.

```

procedure RVA_FillLanguageList(sl: TRVALocStrings(350);
English: Boolean = True;
Native: Boolean = False);

```

English	Native	Result
True	False	English language names
False	True	Native language names
True	True	Strings in format: "English language name - native language name" (or English language name, if they are equal)
False	False	English language names

Names of languages can be assigned to `TRVAControlPanel.Language`⁽⁵³⁾ or used as a parameter for `RVA_SwitchLanguage`⁽³⁴⁹⁾ procedure.

1.4.1.1.4 RVA_GetCharset function

This method returns the charset for the current UI language.

Unit `RVALocalize`⁽³⁴³⁾.

function `RVA_GetCharset(ControlPanel: TComponent=nil): TFontCharset;`

If **ControlPanel** is a `TRVAControlPanel`⁽³⁹⁾ component, the function uses it. Otherwise, it uses `MainRVAControlPanel`⁽³⁹⁰⁾.

The name of the current language is returned as `ControlPanel.Language`⁽⁵³⁾, or by `RVA_GetLanguageName`⁽³⁴⁸⁾ function.

The current language can be changed by assigning a new value to `ControlPanel.Language`⁽⁵³⁾, or by calling `RVA_SwitchLanguage`⁽³⁴⁹⁾ procedure.

This charset is ignored in Delphi/C++Builder 2009 or newer, because Unicode strings do not need charsets. Use this function for Delphi/C++Builder 5-2007.

`DEFAULT_CHARSET` is used for languages that do not have ANSI charset (Armenian, Hindi). For these languages, localization strings have UTF-8 encoding, and can be used in old versions of Delphi only with TNT Controls⁽³⁷²⁾.

RichViewActions apply this charset to all fonts in all dialogs displayed by RichViewActions.

You should use this property to apply to your own forms (if they support a localization) and to `Screen.HintFont` and `Screen.MenuFont`:

```
Screen.HintFont.Charset := RVA_GetCharset; // Delphi 6-2007 is required
Screen.MenuFont.Charset := RVA_GetCharset; // Delphi 6-2007 is required
```

1.4.1.1.5 RVA_GetHelpFile function

This method returns the help file name for the current UI language.

Unit `RVALocalize`⁽³⁴³⁾.

function `RVA_GetHelpFile(ControlPanel: TComponent): String;`

If **ControlPanel** is a `TRVAControlPanel`⁽³⁹⁾ component.

The name of the current language is returned as **ControlPanel.Language**⁽⁵³⁾, or by **RVA_GetLanguageName**⁽³⁴⁸⁾ function.

The current language can be changed by assigning a new value to **ControlPanel.Language**⁽⁵³⁾, or by calling **RVA_SwitchLanguage**⁽³⁴⁹⁾ procedure.

See also:

- **TRVAControlPanel.UseHelpFiles**⁽⁵⁸⁾

1.4.1.1.6 RVA_GetLanguageName function

This method returns the name of the current UI language.

Unit **RVALocalize**⁽³⁴³⁾.

```
function RVA_GetLanguageName (
    ControlPanel: TComponent=nil): TRVALanguageName(350);
```

If **ControlPanel** is a **TRVAControlPanel**⁽³⁹⁾ component, the function uses it. Otherwise, it uses **MainRVAControlPanel**⁽³⁹⁰⁾.

The same value is returned in **ControlPanel.Language**⁽⁵³⁾ property.

The current language can be changed by assigning a new value to **ControlPanel.Language**⁽⁵³⁾, or by calling **RVA_SwitchLanguage**⁽³⁴⁹⁾ procedure.

1.4.1.1.7 RVA_GetS function

This method returns a localized string.

Unit **RVALocalize**⁽³⁴³⁾.

type

```
TRVAMessageID = (rvam_Empty, rvam_menu_File, rvam_menu_Edit, ...);
function RVA_GetS(MsgID: TRVAMessageID;
    ControlPanel: TComponent=nil): TRVALocString(350);
function RVA_GetSH(MsgID: TRVAMessageID;
    ControlPanel: TComponent=nil): TRVALocString(350);
function RVA_GetPC(MsgID: TRVAMessageID;
    ControlPanel: TComponent=nil): PChar
// RVA_GetPC returns WideString for TNT Controls
```

If **ControlPanel** is a **TRVAControlPanel**⁽³⁹⁾ component, the functions use it. Otherwise, they use **MainRVAControlPanel**⁽³⁹⁰⁾.

These functions return strings corresponding to **MsgID**, translated to the current language. The name of the current language is returned as **ControlPanel.Language**⁽⁵³⁾, or by **RVA_GetLanguageName**⁽³⁴⁸⁾ function.

The current language can be changed by assigning a new value to **ControlPanel.Language**⁽⁵³⁾, or by calling **RVA_SwitchLanguage**⁽³⁴⁹⁾ procedure.

RVA_GetS just returns a string.

RVA_GetSH(MsgId) returns ' '+**RVA_GetS(MsgId)**+' ' (used for assigning to captions of radio groups and group boxes).

RVA_GetPC returns the same value as **RVA_GetS**, but not as a String but as PChar.

Note: if RichViewActions are compiled with TNT Controls⁽³⁷²⁾, these functions return WideString.

1.4.1.1.8 RVA_SetHelpFile procedure

This method defines the help file that may be used in RichViewActions dialogs.

Unit RVALocalize⁽³⁴³⁾.

```
procedure RVA_SetHelpFile(const LanguageName: TRVALanguageName(350);
  const HelpFile: String);
```

The specified **HelpFile** is used when **LanguageName** language is an active UI language.

See also:

- TRVAControlPanel.Language⁽⁵³⁾
- TRVAControlPanel.UseHelpFiles⁽⁵⁸⁾

1.4.1.1.9 RVA_SwitchLanguage procedure

Changes the current language.

```
procedure RVA_SwitchLanguage(const LanguageName: TRVALanguageName;
  ControlPanel: TComponent=nil);
```

If **ControlPanel** is a TRVAControlPanel⁽³⁹⁾ component, the function uses it. Otherwise, it uses MainRVAControlPanel⁽³⁹⁰⁾.

Assigning a new value to ControlPanel.Language⁽⁵³⁾ has the same effect.

To get a list of valid languages, use RVA_EnumLanguages⁽³⁴⁶⁾ or RVA_FillLanguageList⁽³⁴⁶⁾.

You can use either English or native language name as a parameter.

To complete a language changing, the following procedures must be called after RVA_SwitchLanguage:

- RVA_LocalizeForm⁽³⁵²⁾ for localization of Captions and Hints of all actions on the given form/datamodule;
- RVALocalizeRuler⁽³⁵³⁾ for localizations of hints in TRVRuler⁽⁸²⁾ component;
- RVA_GetCharset⁽³⁴⁷⁾ for changing Charsets of menus, hints and your forms;
- RVA_GetS⁽³⁴⁸⁾ for localizing standard menu items in your menu (such as "File", "Edit"...).

See also:

- RVA_ChooseLanguage⁽³⁵¹⁾

1.4.1.1.10 RVA_TranslateUnits procedure

Translates units of measurement to the current language.

```
procedure RVA_TranslateUnits(SL: TStrings; ControlPanel: TComponent=nil);
```

If **ControlPanel** is a TRVAControlPanel⁽³⁹⁾ component, the function uses it. Otherwise, it uses MainRVAControlPanel⁽³⁹⁰⁾.

SL must have 6 items corresponding to the following units, in the specified order:

- Inches,
- Centimeters,

- Millimeters,
- Picas,
- Pixels,
- Points.

The name of the current language is returned as `ControlPanel.Language`⁽⁵³⁾, or by `RVA_GetLanguageName`⁽³⁴⁸⁾ function.

The current language can be changed by assigning a new value to `ControlPanel.Language`⁽⁵³⁾, or by calling `RVA_SwitchLanguage`⁽³⁴⁹⁾ procedure.

1.4.1.2 Types

Types declared in `RVALocalize`

- `TRVALanguageName`⁽³⁵⁰⁾
- `TRVALocString`, `TRVALocStrings`, `TRVALocStringList`⁽³⁵⁰⁾

1.4.1.2.1 `TRVALanguageName`

A type for UI language name.

Unit `RVALocalize`⁽³⁴³⁾.

type

```
TRVALanguageName = type String;
```

See also:

- `TRVAControlPanel.Language`⁽⁵³⁾
- `RVA_GetLanguageName`⁽³⁴⁸⁾
- `RVA_SwitchLanguage`⁽³⁴⁹⁾

1.4.1.2.2 `TRVALocString`, `TRVALocStrings`, `TRVALocStringList`

unit `RVALocalize`;

If TNT controls⁽³⁷²⁾ are not used:

type

```
TRVALocString = String;
TRVALocStrings = TStrings;
TRVALocStringList = TStringList;
```

If TNT controls are used:

type

```
TRVALocString = WideString;
TRVALocStrings = TTntStrings;
TRVALocStringList = TTntStringList;
```

Strings of these types are used in user interface of `RichViewActions`.

If TNT controls are used, or in Delphi/C++Builder 2009 or newer, the strings are Unicode (UTF-16).

In Lazarus, the strings are Unicode (UTF-8).

In older versions of Delphi, the strings are ANSI strings. Charset of these strings for the active language is returned by `RVA_GetCharset`⁽³⁴⁷⁾.

1.4.2 RichViewActions unit

Procedures and functions declared in RichViewActions

- RVA_ChoseLanguage function⁽³⁵¹⁾ displays a dialog allowing to switch UI language;
- RVA_GetColorName⁽³⁵²⁾ returns a localized color name;
- RVA_LocalizeForm⁽³⁵²⁾ translates all actions on the specified form/datamodule.

1.4.2.1 Procedures and functions

Procedures and functions declared in RichViewActions

- RVA_ChoseLanguage function⁽³⁵¹⁾ displays a dialog allowing to switch UI language;
- RVA_GetColorName⁽³⁵²⁾ returns a localized color name;
- RVA_LocalizeForm⁽³⁵²⁾ translates all actions on the specified form/datamodule.

1.4.2.1.1 RVA_ChoseLanguage function

Displays a dialog for choosing UI language and applies the chosen language.

Unit RichViewActions.

function RVA_ChoseLanguage (ControlPanel: TRVAControlPanel⁽³⁹⁾ = **nil**): Boolean;

This function displays a dialog containing a list with all available UI languages.

If the user pressed "OK", this function calls RVA_SwitchLanguage⁽³⁴⁹⁾ and returns *True*.

You may need to perform additional operations on the language changing, see RVA_SwitchLanguage⁽³⁴⁹⁾ for details.

In Delphi 4-2007, this function displays language names in English.

In new versions of Delphi (or if TNT Controls⁽³⁷²⁾ are used), this function displays both English and native language names.

Example (from the ActionTest demo):

```
// button at the right side of status bar
procedure TForm3.Button1Click(Sender: TObject);
begin
    if RVA_ChoseLanguage then
        Localize;
end;

procedure TForm3.FormCreate(Sender: TObject);
begin
    ...
    Localize;
    ...
end;

procedure TForm3.Localize;
var Index: Integer;
begin
    // Fonts (for Delphi 5-2007)
```

```

Font.Charset          := RVA_GetCharset(347);
StatusBar1.Font.Charset := RVA_GetCharset(347);
// Fonts (for Delphi 6-2007)
Screen.HintFont.Charset := RVA_GetCharset(347);
Screen.MenuFont.Charset := RVA_GetCharset(347);
// Localizing all actions on rvActionsResource
RVA_LocalizeForm(352)(rvActionsResource);
// Localizing all actions on this form
RVA_LocalizeForm(352)(Self);
// Localizing ruler
RVALocalizeRuler(353)(RVRuler1);
// Localizing menus
mitFile.Caption := RVA_GetS(348)(rvam_menu_File);
mitEdit.Caption := RVA_GetS(348)(rvam_menu_Edit);
// Styles
rvActionsResource.rvActionStyleInspector1.UpdateInfo(220);
RVStyleTemplateComboBox1.Localize(93);
// Localizing combobox with units of measurement
Index := cmbUnits.ItemIndex;
RVA_TranslateUnits(349)(cmbUnits.Items);
cmbUnits.ItemIndex := Index;
end;

```

1.4.2.1.2 RVA_GetColorName function

Returns a localized color name.

Unit RichViewActions.

```

function RVA_GetColorName(Color: TColor;
  ControlPanel: TRVAControlPanel(39) = nil):TRVALocString(350);

```

If **ControlPanel** parameter is defined, its Language⁽⁵³⁾ property is used. Otherwise, the function uses MainRVAControlPanel⁽³⁹⁰⁾.Language.

For *clNone*, it returns 'Transparent' (translated to the current language).

For *clWindow*, *clBtnFace*, *clWindowText*, it returns 'Auto' (translated to the current language).

For 40 standard colors, it returns their localized names.

For other colors, it returns 'RGB(NN,NN,NN)' string.

See also:

- Localization of RichViewActions⁽³⁴²⁾.

1.4.2.1.3 RVA_LocalizeForm procedure

Applies the current language to all RichViewActions on **Form**.

Unit RichViewActions.

```

procedure RVA_LocalizeForm(Form: TComponent);

```

Form is a form or datamodule. This procedure changes Captions and Hints of all actions (inherited from TrvCustomAction⁽³³⁵⁾) owned by this form/datamodule.

The name of the current language is returned as `Action.GetControlPanel338.Language53`.

The current language can be changed by assigning a new value to `TRVAControlPanel.Language53`, or by calling `RVA_SwitchLanguage349` procedure.

See also:

- `RVALocalize unit343`.

1.4.3 Other units

Procedures and functions declared in other units

`GetAddictSpellLanguage`, `GetAddictThesLanguage353`

`RVALocalizeRuler procedure353`

`RVGetHelpKeyword` and `RVSetHelpKeyword354`

1.4.3.1 GetAddictSpellLanguage, GetAddictThesLanguage functions

The functions convert a language of RichViewActions to languages used by Addict 3³⁶⁴ components.

Unit `RVAddictLocalize`.

function `GetAddictSpellLanguage(const s: String): TSpellLanguageType;`

function `GetAddictThesLanguage(const s: String): TThesaurusLanguageType;`

Example:

```
RVAddictSpell31.UILanguage :=
  GetAddictSpellLanguage(RVA_GetLanguageName348);
RVThesaurus31.UILanguage :=
  GetAddictThesLanguage(RVA_GetLanguageName348);
```

It's not necessary to use these functions: `TrvActionSpellingCheck312` and `TrvActionThesaurus312` displays dialogs using the proper UI language.

Note: `RVAddictLocalize` unit is included in `Addict3_RichView*` packages. These packages are installed only if Addict 3/4 is already installed.

1.4.3.2 RVALocalizeRuler procedure

Localizes a `TRVRuler82` component.

Unit `RVALocRuler`.

procedure `RVALocalizeRuler(Ruler: TRVRuler82;
 ControlPanel: TRVAControlPanel39 = nil);`

The method assigns localized strings to hints displayed by **Ruler**.

If **ControlPanel** parameter is defined, its `Language53` property is used. Otherwise, the function uses `MainRVAControlPanel390.Language`.

See also:

- Localization of RichViewActions ⁽³⁴²⁾
- RVALocalize unit ⁽³⁴³⁾

1.4.3.3 RVGetHelpKeyword and RVSetHelpKeyword

Unit RVHelp.

These functions may be useful if you provide a help file and use HelpKeywords instead of HelpContexts.

```
function RVGetHelpKeyword(HelpContext: Integer): String;  
procedure RVSetHelpKeyword(HelpContext: Integer;  
    const HelpKeyword String);
```

HelpKeyword properties are used if the active TRVAControlPanel ⁽³⁹⁾'s HelpType ⁽⁴⁹⁾ = *htKeyword*.

The list of HelpContext and HelpKeyword properties used by RichViewActions is provided in the topic about HelpType ⁽⁴⁹⁾.

By default, HelpContexts are used. You can switch to HelpKeywords. Default values of HelpKeywords are in English. Since keywords are visible to the user (in case of CHM help, they are in the "Index" page of the CHM viewer), it makes sense to translate them, if your help file is not in English. You can change a HelpKeyword corresponding to the given HelpContext using **RVSetHelpKeyword**. This change happens for all UI languages, so, if you provide multiple help files for different languages, you need to change keywords after a UI language change (or use language-independent HelpContexts).

See also properties of TRVAControlPanel ⁽³⁹⁾:

- UseHelpFiles ⁽⁵⁸⁾
- UseDefaultHelpFile ⁽⁵⁷⁾

1.5 Interfaces for third-party components

RichViewActions interface components is a group of components allowing RichViewActions to use third-party components.

A third-party component is assigned to a property of an interface component, and this interface component is assigned to a property of TRVAControlPanel ⁽³⁹⁾.

This indirect linking allows using RichViewActions and third-party components together, without losing a possibility using them separately, because:

- RichViewActions units do not refer to units of third-party components
- units of third-party components do not refer to RichViewActions units
- only units of interface components refer to the both of them.

RichViewActions support the following types of interface components:



- downloaders ⁽³⁵⁵⁾
- spelling checkers ⁽³⁵⁷⁾
- color dialogs ⁽³⁶²⁾

1.5.1 Download interfaces

Download interfaces allow using third-party components to download external images and CSS files referred from RTF, DocX and HTML files.

To use a download interface component, assign it to DownloadInterface⁽⁴⁸⁾ property of TRVAControlPanel⁽³⁹⁾ component.

The following download interface components are provided:

Component	Requires
 TRVAIndyDownloadInterface ⁽³⁵⁶⁾	Indy: included in Delphi or http://www.indyproject.org
 TRVACIDownloadInterface ⁽³⁵⁵⁾	Clever Internet Suite: http://www.clevercomponents.com

See also

- RichViewActions interfaces for third-party components⁽³⁵⁴⁾

1.5.1.1 TRVACIDownloadInterface

Allows using TClHttp component (Clever Internet Suite) to download external images and CSS files when loading files.

Unit RVACIDownloadInterface;

```
TRVACIDownloadInterface = class (TRVACustomDownloadInterface(356));
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomDownloadInterface⁽³⁵⁶⁾

Description

To use this component, assign it to DownloadInterface⁽⁴⁸⁾ property of TRVAControlPanel⁽³⁹⁾ component.

TRVACIDownloadInterface has property: **ClHttp**. Assign TClHttp component to this property, and actions will use it to download images and CSS files. If this property is not assigned, a temporary TClHttp component is created each time when an action needs to download an image or a CSS file.

The TRichView installer installs this component automatically if Clever Internet Suite is already installed.

Clever Internet Suite can be downloaded from <http://www.clevercomponents.com>.

See also:

- RichViewActions download interfaces⁽³⁵⁵⁾

1.5.1.2 TRVACustomDownloadInterface

A base class for RichViewActions download interface components.

Unit RVADownloadInterface;

```
TRVACustomDownloadInterface = class (TComponent);
```

Hierarchy

TObject
TPersistent
TComponent

Description

This component is not used directly. Instead, use one of the following components inherited from it:

- TRVAIndyDownloadInterface ⁽³⁵⁶⁾
- TRVACIDownloadInterface ⁽³⁵⁵⁾

To use a download interface component, assign it to DownloadInterface ⁽⁴⁸⁾ property of TRVAControlPanel ⁽³⁹⁾ component.

See also:

- RichViewActions download interfaces ⁽³⁵⁵⁾

1.5.1.3 TRVAIndyDownloadInterface

Allows using TIdHttp component (Indy) to download external images and CSS files when loading files.

Unit RVAIndyDownloadInterface;

```
TRVAIndyDownloadInterface = class (TRVACustomDownloadInterface (356));
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomDownloadInterface ⁽³⁵⁶⁾

Description

To use this component, assign it to DownloadInterface ⁽⁴⁸⁾ property of TRVAControlPanel ⁽³⁹⁾ component.

TRVAIndyDownloadInterface has property: **IdHttp**. Assign TIdHttp component to this property, and actions will use it to download images and CSS files. If this property is not assigned, a temporary TIdHttp component is created each time when an action needs to download an image or a CSS file.

The TRichView installer installs this component automatically if Indy is already installed.

Indy components are included in Delphi, or can be downloaded from <http://www.indyproject.org>.

See also:

- RichViewActions download interfaces ⁽³⁵⁵⁾

1.5.2 Spelling check interfaces

Spelling check interfaces allow using third-party components to correct spelling errors in TRichViewEdit.




To use a spelling check interface component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Then you can:

- use TrvActionSpellingCheck⁽³¹²⁾ action
- use TrvActionThesaurus⁽³¹²⁾ action (if the chosen interface component supports a thesaurus)
- call its AutoCorrectInKeyDown in the editor's OnKeyDown event and AutoCorrectInKeyPress in the editor's OnKeyPress event to perform auto-correction on typing (if the chosen interface component supports an auto-correction)
- assign the inverted result of its IsWordValid method to the Misspelled parameter of the editor's OnSpellingCheck event to perform a live spelling check.

Components

All spelling check interface components implement both live spelling check (spelling check in a background thread) and a traditional spelling check using dialogs

Component	Requires	Thesaurus	Auto-correcti on	Dialog Localizati on	Dialog Type	Win32/Wi n64	Parser is implemen ted by
 TRVSpellInterface ⁽³⁵⁸⁾	—	—	Yes	Yes	Classic/ Word	Win32+Win 64	TRichView
 TRVAAddictSpellInt erface ⁽³⁵⁸⁾	Addict 3 or 4 ⁽³⁶⁴⁾	Yes	Yes	Yes ¹	Classic/ Word	Win32+Win 64	Spell checker
 TRVAASpellInterfac e ⁽³⁵⁹⁾	ASpell ⁽³⁶⁵⁾	—	—	Yes	Classic/ Word	Win32	TRichView
 TRVADXSpellInterfa ce ⁽³⁶⁰⁾	ExpressSpellC hecker ⁽³⁶⁶⁾	—	Yes	Yes ²	Classic/ Word	Win32+Win 64	TRichView
 TRVAHunSpellInterf ace ⁽³⁶¹⁾	HunSpell ⁽³⁶⁷⁾	—	—	Yes	Classic/ Word	Win32+Win 64	TRichView
 TRVAPolarSpellInte rface ⁽³⁶¹⁾	Polar SpellChecker ⁽³⁶⁸⁾	—	Yes	Yes ³	Classic	Win32	TRichView

Localization:

- 1: a spelling form is localized by the spelling checker, RichViewActions select the most appropriate language for TRVAControlPanel⁽³⁹⁾.Language⁽⁵³⁾ automatically.
- 2: a spelling form is localized by the spelling checker.
- 3: a spelling form can be localized by the spelling checker, however, ready-to-use localized text is not supplied.

See also

- RichViewActions interfaces for third-party components⁽³⁵⁴⁾

1.5.2.1 TRVSpellInterface

Allows using TRVSpellChecker (included in TRichView components) to correct spelling errors.

Unit RVAAddictSpellInterface;

```
TRVSpellChecker = class (TRVACustomSpellInterface(359));
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface⁽³⁵⁹⁾

Description

To use this component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Properties:

- **SpellChecker** – a link to TRVSpellChecker component. Assignment to this property is required, otherwise a spelling check will not work.

See also:

- RichViewActions spelling check interfaces⁽³⁵⁷⁾

1.5.2.2 TRVAAddictSpellInterface

Allows using Addict (by Addictive Software)⁽³⁶⁴⁾ to correct spelling errors and to find synonyms.

Unit RVAAddictSpellInterface;

```
TRVAAddictSpellInterface = class (TRVACustomSpellInterface(359));
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface⁽³⁵⁹⁾

Description

To use this component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Properties:

- **AddictSpell** – a link to TRVAddictSpell3 component. Assignment to this property is required, otherwise a spelling check will not work.
- **Thesaurus** – a link to TRVThesaurus3 component. Assignment to this property is required, otherwise TrvActionThesaurus⁽³¹²⁾ will be disabled.

The TRichView installer installs this component automatically if Addict is already installed, and "build control" option is changed⁽³⁶⁴⁾ for Addict packages.

Addict can be downloaded from <http://www.addictivesoftware.com>.

See also:

- RichViewActions spelling check interfaces⁽³⁵⁷⁾

1.5.2.3 TRVAASpellInterface

Allows using ASpell⁽³⁶⁵⁾ to correct spelling errors.

Unit RVAASpellInterface;

```
TRVAASpellInterface = class (TRVACustomSpellInterface(359)) ;
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface⁽³⁵⁹⁾

Description

To use this component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Properties:

- **ASpell** – a link to TRVASpell component. Assignment to this property is required, otherwise spelling check will not work.

ASpell support does not require additional components, all necessary components are installed by the TRichView installer.

However, the components works only if ASpell is installed. The installer can be found on aspell.net/win32.

See also:

- RichViewActions spelling check interfaces⁽³⁵⁷⁾

1.5.2.4 TRVACustomSpellInterface

A base class for RichViewActions spelling check interface components.

Unit RVADownloadInterface;

```
TRVACustomDownloadInterface = class (TComponent) ;
```

Hierarchy

TObject
TPersistent

TComponent

Description

This component is not used directly. Instead, use one of the following components inherited from it:

- TRVAAddictSpellInterface⁽³⁵⁸⁾ (to use Addict⁽³⁶⁴⁾)
- TRVAASpellInterface⁽³⁵⁹⁾ (to use ASpell⁽³⁶⁵⁾)
- TRVADXSpellInterface⁽³⁶⁰⁾ (to use ExpressSpellChecker⁽³⁶⁶⁾)
- TRVAHunSpellInterface⁽³⁶¹⁾ (to use HunSpell⁽³⁶⁷⁾)
- TRVAPolarSpellInterface⁽³⁶¹⁾ (to use Polar SpellChecker⁽³⁶⁸⁾)

To use a spelling check interface component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Then you can:

- use TrvActionSpellingCheck⁽³¹²⁾ action
- use TrvActionThesaurus⁽³¹²⁾ action (if the chosen interface component supports a thesaurus)
- call its AutoCorrectInKeyDown in the editor's OnKeyDown event and AutoCorrectInKeyPress in the editor's OnKeyPress (in OnUTF8KeyPress for Lazarus) event to perform auto-correction on typing (if the chosen interface component supports an auto-correction)
- assign the inverted result of its IsWordValid method to the Misspelled parameter of the editor's OnSpellingCheck event to perform a live spelling check.

Methods [VCL]

```
function AutoCorrectInKeyDown(Editor: TCustomRichViewEdit;
    Key: Word): Boolean;
function AutoCorrectInKeyPress(Editor: TCustomRichViewEdit;
    Key: Char): Boolean;
function IsWordValid(const S: TRVUnicodeString;
    rv: TCustomRichView; StyleNo: Integer): Boolean;
virtual; abstract;
```

Methods [FPC]

For Lazarus, the methods are the same, with the small difference:

```
function AutoCorrectInKeyPress(Editor: TCustomRichViewEdit;
    Key: TUTF8Char): Boolean;
```

See also:

- RichViewActions spelling check interfaces⁽³⁵⁷⁾

1.5.2.5 TRVADXSpellInterface

Allows using ExpressSpellChecker⁽³⁶⁶⁾ to correct spelling errors.

Unit RVADXSpellInterface;

```
TRVADXSpellInterface = class (TRVACustomSpellInterface(359));
```

Hierarchy

TObject

TPersistent

*TComponent**TRVACustomSpellInterface* ⁽³⁵⁹⁾

Description

To use this component, assign it to SpellInterface ⁽⁵⁶⁾ property of TRVAControlPanel ⁽³⁹⁾ component.

Properties:

- **SpellChecker** – a link to TRvDxSpellChecker component. Assignment to this property is required, otherwise a spelling check will not work.

See also:

- RichViewActions spelling check interfaces ⁽³⁵⁷⁾

1.5.2.6 TRVAHunSpellInterface

Allows using HunSpell ⁽³⁶⁷⁾ to correct spelling errors.

Unit RVAASpellInterface;

```
TRVAHunSpellInterface = class (TRVACustomSpellInterface (359));
```

Hierarchy

*TObject**TPersistent**TComponent**TRVACustomSpellInterface* ⁽³⁵⁹⁾

Description

To use this component, assign it to SpellInterface ⁽⁵⁶⁾ property of TRVAControlPanel ⁽³⁹⁾ component.

Properties:

- **HunSpell** – a link to TRVHunSpell component. Assignment to this property is required, otherwise a spelling check will not work.

HunSpell support does not require additional components, all necessary components are installed by the TRichView installer.

However, the components works only if HunSpell DLL and dictionaries are available for the application.

See also:

- RichViewActions spelling check interfaces ⁽³⁵⁷⁾

1.5.2.7 TRVAPolarSpellInterface

Allows using Polar SpellChecker ⁽³⁶⁸⁾ to correct spelling errors.

Unit RVAPolarSpellInterface;

```
TRVAPolarSpellInterface = class (TRVACustomSpellInterface (359));
```

Hierarchy

*TObject**TPersistent*

TComponent

TRVACustomSpellInterface ⁽³⁵⁹⁾

Description

To use this component, assign it to *SpellInterface* ⁽⁵⁶⁾ property of *TRVAControlPanel* ⁽³⁹⁾ component.

Properties:

- **PolarSpellChecker** – a link to *TRVPolarSpellChecker* component. Assignment to this property is required, otherwise spelling check will not work.

See also:

- RichViewActions spelling check interfaces ⁽³⁵⁷⁾


1.5.3 Color dialog interfaces

By default, RichViewActions use the standard *TColorDialog*. You can assign a specific *TColorDialog* component to *ColorDialog* ⁽⁴²⁾ property of *TRVAControlPanel* ⁽³⁹⁾, and all RichViewActions will use this component.

Alternatively, you can use third-party color dialogs via a color interface component. All color dialog interface components are inherited *TRVCustomColorDialogInterface*.

To use a color dialog interface component, assign it to *ColorDialogInterface* ⁽⁴³⁾ property of *TRVAControlPanel* ⁽³⁹⁾ component.

The following download interface components are provided:

Component	Requires	Dialog Component
 <i>TRVADXColorDialogInterface</i>	DevExpress VCL Components	<i>TdxColorDialog</i>

See also

- RichViewActions interfaces for third-party components ⁽³⁵⁴⁾

1.6 Third-party and additional tools

Components

The following additional components can be optionally used in RichViewActions.

Spell checkers:

- Addict 3 or 4 ⁽³⁶⁴⁾
- ASpell ⁽³⁶⁵⁾
- ExpressSpellChecker ⁽³⁶⁶⁾
- HunSpell ⁽³⁶⁷⁾
- Polar Spell Checker ActiveX

Downloaders:

- CleverComponents ⁽³⁶⁹⁾

- Indy⁽³⁷⁰⁾

File readers and writers:

- RichViewXML⁽³⁷⁰⁾ for opening and saving documents in XML format (freeware)

User interface:

- Toolbar 2000⁽³⁷³⁾ (free for non-commercial use), TBX⁽³⁷³⁾, SpTBXLib⁽³⁷²⁾ for providing alternative versions of TRVAPopupMenu⁽⁶⁸⁾
- TNT Controls⁽³⁷²⁾ for Unicode support in Delphi 5-2007.

Toolbar Images

By default, RichViewActions use simple 16-color 16x16 images.

In Delphi 2009+ (or if you use a thirdparty image list component supporting semitransparent images), you can use TRichView icons⁽³⁷⁴⁾, or the following alternative toolbar images:

- GlyFX⁽³⁷⁶⁾
- Glyfz⁽³⁷⁷⁾
- Fugue Icons⁽³⁷⁷⁾
- FamFamFam Silk Icons⁽³⁷⁸⁾

1.6.1 Components

Components

The following additional components can be optionally used in RichViewActions.

Spell checkers:

- TRVSpellChecker (included in TRichView)
- Addict 3 or 4⁽³⁶⁴⁾
- ASpell⁽³⁶⁵⁾
- ExpressSpellChecker⁽³⁶⁶⁾
- HunSpell⁽³⁶⁷⁾
- Polar Spell Checker ActiveX

Downloaders:

- CleverComponents⁽³⁶⁹⁾
- Indy⁽³⁷⁰⁾

File readers and writers:

- RichViewXML⁽³⁷⁰⁾ for opening and saving documents in XML format (freeware)

User interface:

- Toolbar 2000⁽³⁷³⁾ (free for non-commercial use), TBX⁽³⁷³⁾, SpTBXLib⁽³⁷²⁾ for providing alternative versions of TRVAPopupMenu⁽⁶⁸⁾
- TNT Controls⁽³⁷²⁾ for Unicode support in Delphi 5-2007.

1.6.1.1 Spelling checkers

Spelling Checking Components ---

The following additional components can be optionally used in RichViewActions:

- TRVSpellChecker (included in TRichView components)
- Addict 3 or 4 ⁽³⁶⁴⁾
- ASpell ⁽³⁶⁵⁾
- ExpressSpellChecker ⁽³⁶⁶⁾
- HunSpell ⁽³⁶⁷⁾
- Polar Spell Checker ActiveX

1.6.1.1.1 Addict 3 and 4

Addict 3 and 4 is a spelling check and thesaurus component suite. Visit www.addictive-software.com for additional information.

(discontinued)

Installing ---

1. Addict 4 packages must be installed before installing TRichView.
2. Addict 4 packages must be changed before installing this package. Open Addict 4 packages, open the package dialog. Go to the page "Description". Change "Build control" = "Explicit rebuild". Recompile the packages. Make sure that you do not have duplicated BPL and DCP files for the Addict packages in different directories (they can be in <AddictDir>\BPL and in Delphi projects directories)
3. If Addict 4 is installed, the TRichView installer installs the component integrating Addict in TRichView and RichViewActions automatically.

If you installed Addict 4 after TRichView, you can run the TRichView installer again using the shortcut in the Windows Start menu.

Using ---

Create TRVAddictSpellInterface ⁽³⁵⁸⁾ component, assign it to SpellInterface ⁽⁵⁶⁾ property of TRVAControlPanel ⁽³⁹⁾ component.

Create TRVAddictSpell3 component, assign it to AddictSpell property of this TRVAddictSpellInterface ⁽³⁵⁸⁾ component.

Create TRVThesaurus3 component, assign it to Thesaurus property of this TRVAddictSpellInterface ⁽³⁵⁸⁾ component.

Assign TrvActionSpellingCheck ⁽³¹²⁾ and TrvActionThesaurus ⁽³¹²⁾ to menu items or toolbar buttons.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface ⁽³⁵⁹⁾.

Live spelling check ---

To enable a live spelling check, assign OnSpellingCheck event of TCustomRichViewEdit component:


```
procedure TForm1.RichViewEdit1SpellingCheck(Sender: TCustomRichView;
  const AWord: String; StyleNo: Integer; var Misspelled: Boolean);
begin
  Misspelled := not RVAAddictSpellInterface1.IsWordValid(AWord,
    RichViewEdit1, StyleNo);
end;
```

If you want to start a live spelling check when the user starts editing, assign *rvlspOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile⁽¹¹²⁾ event of TrvActionOpen⁽¹⁰⁸⁾ action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject;
  Editor: TCustomRichViewEdit; const FileName: String;
  FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu⁽⁶⁸⁾ component to RichViewEdit1.PopupMenu.

1.6.1.1.2 ASpell

GNU Aspell is a Free and Open Source spell checker.

Addict 3 and 4 is a spelling check and thesaurus component suite. Visit aspell.net for additional information.

Installing

ASpell support does not require additional components, all necessary components are installed by the TRichView installer.

However, the components works only if ASpell is installed. The installer can be found on aspell.net/win32.

Using

Create TRVAASpellInterface⁽³⁵⁹⁾ component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Create TRVASpell component, assign it to ASpell property of this TRVAASpellInterface⁽³⁵⁹⁾ component.

Assign TrvActionSpellingCheck⁽³¹²⁾ to a menu item or a toolbar button.

Live spelling check

To enable a live spelling check, assign OnSpellingCheck event of TCustomRichViewEdit component:

```
procedure TForm1.RichViewEdit1SpellingCheck(Sender: TCustomRichView;
  const AWord: String; StyleNo: Integer; var Misspelled: Boolean);
begin
  Misspelled := not RVAASpellInterface1.IsWordValid(AWord,
    RichViewEdit1, StyleNo);
end;
```

If you want to start a live spelling check when the user starts editing, assign *rv/spOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile⁽¹¹²⁾ event of TrvActionOpen⁽¹⁰⁸⁾ action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject;
  Editor: TCustomRichViewEdit; const FileName: String;
  FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu⁽⁶⁸⁾ component to RichViewEdit1.PopupMenu.

1.6.1.1.3 ExpressSpellChecker

ExpressSpellChecker is a VCL spell checker component.

Visit <https://www.devexpress.com> for additional information.

Installing

If DevExpress VCL components are installed, the TRichView installer installs the component integrating ExpressSpellChecker in TRichView and RichViewActions automatically.

Using

Create TRVADXSpellInterface⁽³⁶⁰⁾ component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Create TRvDxSpellChecker component, assign it to SpellChecker property of this TRVADXSpellInterface⁽³⁶⁰⁾ component. Add the code to initialize the spell checker.

Assign TrvActionSpellingCheck⁽³¹²⁾ to a menu item or a toolbar button.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface⁽³⁵⁹⁾.

Live spelling check in TcxTRichViewEdit

In applications using DxSpellChecker, it is assumed that live spelling is performed only in the focused component.

If you use TcxTRichViewEdit component, it will be checked automatically when focused, if RvDxSpellChecker.CheckAsYouTypeOptions.Active = True.

You can assign TRVAPopupMenu⁽⁶⁸⁾ component to RichViewEdit1.PopupMenu, however, TcxTRichViewEdit can display a popup menu with live spelling options even without it.

Live spelling check in TRichViewEdit (and TsrRichViewEdit)

In applications using DxSpellChecker, it is assumed that live spelling is performed only in the focused component.

If you use TRichViewEdit component:

TRichViewEdit, process OnEnter and OnExit:

OnEnter:

```
RvDxSpellChecker1.CheckAsYouTypeOptions.Active := False;
RvDxSpellChecker1.UpdateRules;
RichViewEdit1.StartLiveSpelling;
```

OnExit:

```
RichViewEdit1.ClearLiveSpellingResults;
RvDxSpellChecker1.CheckAsYouTypeOptions.Active := True;
```

Assign OnSpellingCheck event:

```
procedure TForm1.RichViewEdit1SpellingCheck(Sender: TCustomRichView;
  const AWord: String; StyleNo: Integer; var Misspelled: Boolean);
begin
  Misspelled := not RVADxSpellInterface1.IsWordValid(AWord,
    RichViewEdit1, StyleNo);
end;
```

1.6.1.1.4 HunSpell

HunSpell is a spell checker.

Hunspell is free software, distributed under the terms of a GPL, LGPL and MPL tri-license.

Visit <http://hunspell.github.io/> for additional information.

Installing

HunSpell support does not require additional components, all necessary components are installed by the TRichView installer.

However, the components works only if HunSpell DLLs and dictionaries are available.

For example, you can download DLLs from our web site:

- **32 bit**
- **64 bit**

Dictionaries (*.aff and *.dic files) can be downloaded, for example, here:

<https://cgkit.freedesktop.org/libreoffice/dictionaries/tree/>.

Using

Create TRVAHunSpellInterface⁽³⁶¹⁾ component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Create TRVHunSpell component, assign it to HunSpell property of this TRVAHunSpellInterface⁽³⁶¹⁾ component. Assign its DictDir, DIIDir, DIIDName properties, if necessary. Load dictionaries (for example, by calling LoadAllDictionaries).

Assign TrvActionSpellingCheck⁽³¹²⁾ to a menu item or a toolbar button.

Live spelling check

To enable a live spelling check, assign OnSpellingCheck event of TCustomRichViewEdit component:

```

procedure TForm1.RichViewEdit1SpellingCheck(Sender: TCustomRichView;
  const AWord: String; StyleNo: Integer; var Misspelled: Boolean);
begin
  Misspelled := not RVAHunSpellInterface1.IsWordValid(AWord,
    RichViewEdit1, StyleNo);
end;

```

If you want to start a live spelling check when the user starts editing, assign *rv/spOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile⁽¹¹²⁾ event of TrvActionOpen⁽¹⁰⁸⁾ action:

```

procedure TForm1.RvActionOpen1OpenFile(Sender: TObject;
  Editor: TCustomRichViewEdit; const FileName: String;
  FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;

```

Assign TRVAPopupMenu⁽⁶⁸⁾ component to RichViewEdit1.PopupMenu.

1.6.1.1.5 Polar SpellChecker ActiveX

Polar SpellChecker component is a spell checker. It may be used as DLL or as ActiveX. TRichView uses it as ActiveX.

Last time Polar SpellChecker was updated in 2005.

Visit <http://www.polarsoftware.com/> for additional information.

Installing

1. Install the Polar SpellChecker Components.
2. Import it in Delphi. Menu "Component | Import Component" (or "Component | Import ActiveX Control", depending on your Delphi version) import and install "Polar SpellChecker 5.0 Component", The component must be installed in the package PolarSpellCheckerPkg*, where * depends on Delphi version:
 - Delphi 5..7: D5..D7
 - Delphi 2005..2010: D2005..D2010
 - Delphi XE..XE8: DXE..DXE8
 - Delphi 10 Seattle: D10
 - Delphi 10.1 Berlin..10.3 Rio: D10_1..D10_3
3. TRichView includes packages for installing support for Polar SpellChecker (in ThirdParty\Polar\ folder), but the installer does not install them automatically. Open the package RVPolarPkg* and compile. Open the package RVPolarPkg*_Dsgn and install.

Using

Create TRVAPolarSpellInterface⁽³⁶¹⁾ component, assign it to SpellInterface⁽⁵⁶⁾ property of TRVAControlPanel⁽³⁹⁾ component.

Create TRVPolarSpellChecker component, assign it to PolarSpellChecker property of this TRVAPolarSpellInterface⁽³⁶¹⁾ component.

Assign TrvActionSpellingCheck⁽³¹²⁾ to a menu item or a toolbar button.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface⁽³⁵⁹⁾.

Live spelling check

To enable a live spelling check, assign OnSpellingCheck event of TCustomRichViewEdit component:

```
procedure TForm1.RichViewEdit1SpellingCheck(Sender: TCustomRichViewEdit;
  const AWord: String; StyleNo: Integer; var Misspelled: Boolean);
begin
  Misspelled := not RVAPolarSpellInterface1.IsWordValid(AWord,
    RichViewEdit1, StyleNo);
end;
```

If you want to start a live spelling check when the user starts editing, assign *rvlspOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile⁽¹¹²⁾ event of TrvActionOpen⁽¹⁰⁸⁾ action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject;
  Editor: TCustomRichViewEdit; const FileName: String;
  FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu⁽⁶⁸⁾ component to RichViewEdit1.PopupMenu.

1.6.1.2 Downloaders

Downloading Components

The following additional components can be optionally used in RichViewActions:

- CleverComponents⁽³⁶⁹⁾
- Indy⁽³⁷⁰⁾

They allow downloading external images referred from imported RTF and HTML files.

1.6.1.2.1 CleverComponents

HTML, Markdown, DocX and RTF files can contain links to images located on a web server. If you want to use them, it's required to download them inside OnImportPicture event of TCustomRichViewEdit component.

RichViewActions can do it for you using CleverComponents (TCIHTTP) component.

Visit <http://www.clevercomponents.com> for additional information.

CleverComponents can be used via TRVACIDownloadInterface⁽³⁵⁵⁾ component.

1.6.1.2.2 Indy

HTML, Markdown, DocX, and RTF files can contain links to images located on web server. If you want to use them, it's required to download them inside OnImportPicture event of TCustomRichViewEdit component.

RichViewActions can do it for you using Indy (TIdHTTP) component.

Indy is included in new versions of Delphi, or it can be [downloaded](#).

Indy can be used via TRVAIndyDownloadInterface⁽³⁵⁶⁾ component.

1.6.1.3 File readers and writers

File Components

The following additional components can be optionally used in RichViewActions:

File readers and writers:

- RichViewXML⁽³⁷⁰⁾ for opening and saving documents in XML format (freeware)

1.6.1.3.1 RichViewXML

RichViewXML provides an alternative to RVF way for lossless storing of RichView documents.

If activated, RichViewActions support RichView XML files in the following actions:

- TrvActionOpen⁽¹⁰⁸⁾
- TrvActionSaveAs⁽¹²⁹⁾
- TrvActionExport⁽¹¹²⁾
- TrvActionInsertFile⁽²³⁰⁾

RichViewXML is installed by TRichView Setup.

How to use

Place TRVAControlPanel⁽³⁹⁾ component on a form. If you have more than one TRVAControlPanel component in your application, link⁽³³⁷⁾ it to actions.

Assign TRichViewXML component to TRVAControlPanel.XMLComponent⁽⁶⁰⁾.

Make sure that *ffiXML* is in Filter of TrvActionOpen⁽¹⁰⁸⁾, TrvActionInsertFile⁽²³⁰⁾, *ffeXML* in Filter of TrvActionSaveAs⁽¹²⁹⁾, TrvActionExport⁽¹¹²⁾.

You can change a file filter for XML files: see TRVAControlPanel.XMLFilter⁽⁶⁰⁾ property.

1.6.1.3.2 RvHtmlImporter (deprecated)

Deprecated. TRichView HTML methods for HTML import provide better result.

RVHTMLImporter allows importing HTML files in TCustomRichView components.

RichViewAction support RVHTMLImporter in the following actions:

- TrvActionInsertFile⁽²³⁰⁾ (RVHTMLImporter replaces the MS Office HTML import converter in the file insertion dialog)

- TrvActionPasteSpecial⁽¹³⁹⁾
 - TrvActionPaste⁽¹³⁷⁾ (pasted if not RVF or non-empty RTF available in the Clipboard)
- RVHTMLImporter is installed by TRichView Setup

How to use

Place TRVAControlPanel⁽³⁹⁾ component on a form. If you have more than one TRVAControlPanel component in your application, link⁽³³⁷⁾ it to actions.

Assign TRvHtmlImporter component to TRVAControlPanel.HTMLComponent⁽⁵³⁾.

Make sure that *ffiHTML* is in Filter⁽²³²⁾ property of TrvActionInsertFile⁽²³⁰⁾ action.

Include *rvddHTML* in AcceptPasteFormats property of the editor.

See also:

- PasteHTML⁽³⁸⁰⁾ procedure
- RvHtmlViewImporter⁽³⁷¹⁾

1.6.1.3.3 RvHtmlViewImporter (deprecated)

Deprecated. TRichView HTML methods for HTML import provide better result.

RVHTMLViewImporter allows loading HTML files in RichView using HTML parser from THTMLViewer (freeware with source).

If activated, RichViewActions support HTML files in the following actions:

- TrvActionOpen⁽¹⁰⁸⁾
- TrvActionSaveAs⁽¹²⁹⁾ (RVHTMLViewImporter is not used for saving, but HTML is added in the "Save As" action because it is added in the "Open" action)
- TrvActionInsertFile⁽²³⁰⁾
- TrvActionPasteSpecial⁽¹³⁹⁾
- TrvActionPaste⁽¹³⁷⁾ (pasted if not RVF or non-empty RTF available in the Clipboard)

RVHTMLViewImporter is installed by TRichView Setup if THTMLViewer is already installed.

THTMLViewer can be downloaded from github.com/BerndGabriel/HtmlViewer.

How to use

Place TRVAControlPanel⁽³⁹⁾ component on a form. If you have more than one TRVAControlPanel component in your application, link⁽³³⁷⁾ it to actions.

Assign TRvHtmlViewImporter component to TRVAControlPanel.HTMLComponent⁽⁵³⁾.

Assign THTMLViewer component to HTMLViewer property of TRVHTMLViewImporter (make THTMLViewer hidden).

Make sure that *ffiHTML* is in Filter⁽²³²⁾ property of TrvActionInsertFile⁽²³⁰⁾ action.

Include *rvddHTML* in AcceptPasteFormats property of the editor.

See also:

- PasteHTML⁽³⁸⁰⁾ procedure

- RvHtmlImporter⁽³⁷⁰⁾

1.6.1.4 User interface

UI Components

The following additional components can be optionally used in RichViewActions:

- Toolbar 2000⁽³⁷³⁾ (free for non-commercial use), TBX⁽³⁷³⁾, SpTBXLib⁽³⁷²⁾ for providing alternative versions of TRVAPopupMenu⁽⁶⁸⁾
- TNT Controls⁽³⁷²⁾ for Unicode support in Delphi 5-2007.

1.6.1.4.1 SpTBXLib

SpTBXLib is an expansion package for TB2K⁽³⁷³⁾ components that adds Unicode support and skinning. You can download it from www.silverpointdevelopment.com/sptbxlib/.

If activated, TRVASPTBXPopupMenu⁽⁶⁸⁾ component is available.

Installing

1. Install the SpTBXLib package.
2. Open **RichViewActions.inc**, remove the dot from the line `{$DEFINE USESPTBX}`.
3. Open the RichViewActions package. Add a link to the SpTBXLib package in "Requires" section. Install the package (or rebuild it, if already installed).

1.6.1.4.2 TNT Controls

TMS Unicode Component Pack controls allow you to develop applications that take advantage of the Unicode capabilities of Windows NT/2000/XP/2003/Vista without abandoning Delphi, C++Builder or Windows 95/98/ME. You can download them from www.tmssoftware.com.

Since Delphi/C++Builder 2009 features a built-in support of Unicode, TMS Unicode Component Pack should be used only in older versions of Delphi.

If activated, the most of standard controls on RichViewActions dialogs are replaced with TNT controls. Caption⁽³³⁶⁾ and Hints⁽³³⁷⁾ properties of RichViewActions are changed to Unicode. TRVAPopupMenu⁽⁶⁸⁾, find and replace dialogs support Unicode.

Installing

1. Open TMS Unicode Component Pack packages and set "Explicit rebuild" option. For Delphi 4-7, open the package, click "Options" in the package window, on the tab "Description" set "Build control"="Explicit rebuild". For newer version of Delphi, open the package, right click in the Project Manager and choose "Options", on the page "Description" set "Build control"="Explicit rebuild".
2. Install TMS Unicode Component Pack.
3. Open **RV_Defs.inc**, remove the dot from the line `{$DEFINE USERVTNT}`.
4. Recompile TRichView and RichViewActions. You can use "Install in IDE" shortcut in the Start menu.

See ActionTest_TNT.dpr demo.

1.6.1.4.3 TBX

SpTBXLib is an expansion package for TB2K⁽³⁷³⁾ components. You can download it from github.com/plashenkov/TBX

If activated, TRVATBXPopupMenu⁽⁶⁸⁾ component is available.

Installing

1. Install the TBX package.
2. Open **RichViewActions.inc**, remove the dot from the line `{$DEFINE USETBX}`.
3. Open the RichViewActions package. Add a link to the TBX package in "Requires" section. Install the package (or rebuild it, if already installed).

1.6.1.4.4 Toolbar2000

Toolbar2000 is a set of components designed to mimic the look and behavior of Office 2000's menus and toolbars. You can download it from www.jrsoftware.org.

It has the following extensions:

- TBX⁽³⁷³⁾
- SpTBXLib⁽³⁷²⁾

If activated, TRVATBPopupMenu⁽⁶⁸⁾ component is available.

Installing

1. Install the Toolbar2000 package.
2. Open **RichViewActions.inc**, remove the dot from the line `{$DEFINE USETB2K}`.
3. Open the RichViewActions package. Add a link to the Toolbar2000 package in "Requires" section. Install the package (or rebuild it, if already installed).

1.6.2 Toolbar Images

Toolbar Images

By default, RichViewActions use simple 16-color 16x16 images.

In Delphi 2009+ (or if you use a thirdparty image list component supporting semitransparent images), you can use TRichView icons⁽³⁷⁴⁾, or the following alternative toolbar images:

- GlyFX⁽³⁷⁶⁾
- Glyfz⁽³⁷⁷⁾
- Fugue Icons⁽³⁷⁷⁾
- FamFamFam Silk Icons⁽³⁷⁸⁾

1.6.2.1 TRichView Icons

TRichView icons

There are two sets of toolbar images created for RichViewActions. They include 16x16 and 32x32 images for all actions, 64x64 images for selected actions, some special images for ScaleRichView.

These images can be used in image lists in Delphi 2009 and newer, because older versions of Delphi do not support semitransparent images in image lists.

These images can be downloaded separately, from TRichView web site. In RichViewActions, they are included in several data modules.

The images are free for registered TRichView users. You can use these images in your applications. Any other use requires a special permission. You cannot sell these images or any graphic work created basing on them.

Data modules for Delphi 2009+, standard toolbar

Content: 16x16 images, TImageList, normal and disabled images.

- Source\dmActionsImages1.pas - image set #1
- Source\dmActionsImages2.pas - image set #2

These data modules are used in the demos:

- DelphiUnicode\ActionTest\
- CBuilderUnicode\ActionTest\

These data modules do not include icons for ScaleRichView and Report Workshop actions. These components include their own analogs of these data modules.

Data modules for Delphi 2009+, TRibbon

Content: 16x16 for all actions, 32x32 for selected actions, TImageList, normal and disabled images.

- ActionTestRibbon\RVARibbonImages1.pas - image set #1
- ActionTestRibbon\RVARibbonImages2.pas - image set #2

These data modules are used in the demos:

- DelphiUnicode\ActionTestRibbon\

These data modules do not include icons for ScaleRichView and Report Workshop actions. These components include their own analogs of these data modules.

Data modules for Delphi 10.3+

Content: 16x16 and 32x32 images for all actions, 64x64 images for selected actions, TImageCollection, normal images. Some special images for ScaleRichView are included as well in two sizes.

- Source\dmActionsImageCollection1.pas - image set #1
- Source\dmActionsImageCollection1.pas - image set #2

These images are used in the following data module:

- Source\dmActionsVirtualImageLists.pas

This data module has two TVirtualImageList components, for normal and disabled images (disabled images are auto-generated). You can switch between image sets by calling SetImageCollection(1) or SetImageCollection(2).

If you use the data module from dmActionsVirtualImageLists.pas in your project, make sure that it is created after the data modules from dmActionsImageCollection1.pas and dmActionsImageCollection2.pas.

These data modules are used in the demo:

- DelphiUnicode\ActionTest_MultiRes\

The image collections include all images, including images for ScaleRichView and Report Workshop. These components use dmActionsImageCollection1.pas and dmActionsImageCollection2.pas units, but include their own data modules with TVirtualImageList components.

Data modules for C++Builder 10.3+

There are C++ analogs of the data modules described in the previous section:

- SourceCPP\dmActionsImageCollectionCPP1.cpp
- SourceCPP\dmActionsImageCollectionCPP2.cpp
- SourceCPP\dmActionsVirtualImageListsCPP.cpp

These data modules are used in the demo:

- CBuilderUnicode\ActionTest_MultiRes\

Data modules for Lazarus 2+

Content: 16x16 and 32x32 images for all actions, 64x64 images for selected actions, TImageList, normal and disabled images.

- SourceEx\dmactionsimagesmultireslaz1.pas - image set #1
- SourceEx\dmactionsimagesmultireslaz2.pas - image set #2

These data modules are used in the demo:

- Lazarus\ActionTest\

These data modules do not include icons for ScaleRichView and Report Workshop actions. These components include their own analogs of these data modules.

Data modules for Lazarus

Content: 16x16 images, TImageList, normal and disabled images.

- SourceEx\dmactionsimageslaz1.pas - image set #1
- SourceEx\dmactionsimageslaz2.pas - image set #2

These images are included for older versions of Lazarus.

Compatibility of image lists

Indexes of images in all image lists mentioned above are the same, except for TRibbon version.

Additional information

Additional information is available on the support forum:

[*image set #1*](#)

[*image set #2*](#)

1.6.2.2 GlyFX toolbar images

GlyFX Images

(deprecated, this datamodule the not maintained anymore)

If you want to use modern looking images, consider [GlyFX](#) toolbar images.

There are 4 datamodules available.

Two datamodules based on GlyFX Windows XP icon sets:

- XP Common (free)
- XP Core II
- XP Word processing.

Two datamodules based on GlyFX Windows Vista icon sets:

- Vista Common (free)
- Vista Core II
- Vista Word processing.

The corresponding sets must be purchased in order to use these datamodules. After purchasing, request datamodules from GlyFX.

All registered users of TRichView get a 25% discount when registering all GlyFX products.

Datamodules:

- *dmActionsGlyFX.pas* uses XP-style images without alpha channel (based on bmp). Can be used in Delphi 4+
- *dmActionsGlyFXAlpha.pas* uses XP-style images with alpha channel (based on png). Can be used in Delphi 2009+
- *dmActionsGlyFXAero.pas* uses Vista-style images without alpha channel (based on bmp). Can be used in Delphi 4+
- *dmActionsGlyFXAeroAlpha.pas* uses Vista-style images with alpha channel (based on png). Can be used in Delphi 2009+

All datamodules include 6 image lists: normal, highlighted disabled; all in 16x16 and 24x24.

Additionally, projects with GlyFX and TRibbon are available for TRichViewEdit and TSRichViewEdit.

Installing

It's not necessary to install files to use these datamodules.

However, if you want to use one of these modules at design time instead of dmActions.pas:

1. Open **RichViewActions.inc** and remove the dot from one of the following lines:

```
{.$DEFINE USEGLYFX}      // to use dmActionsGlyFX      (XP-style, 1 bit transparency)
{$DEFINE USEGLYFXALPHA}  // to use dmActionsGlyFXAlpha  (XP-style, 1 byte transparency,
D2009+)
{$DEFINE USEGLYFXAERO}   // to use dmActionsGlyFXAero   (Vista-style, 1 bit transparency)
```

```
{.$DEFINE USEGLYFXAEROALPHA} // to use dmActionsGlyFXAeroAlpha (Vista-style, 1 byte transparency, D2009+)
```

2. Remove dmActions.pas from the RichViewActions package
3. Install (or recompile) the package.

Additional information

[Additional information is available on the support forum.](#)

1.6.2.3 Glyfz toolbar images

Glyfz Images

(deprecated, this datamodule the not maintained anymore)

If you want to use toolbar images looking very close to MS Word 2007/2010 images, consider [Glyfz](#) toolbar images.

We have the following demos:

- analog of Ribbon\ActionTestRibbon demo for TRichViewEdit
- analog of ActionTestTabsRibbon demo for ScaleRichView.

These projects can be used only in Delphi 2009+, because they require TRibbon component, and because images are semitransparent.

They use the following sets of Glyfz images:

- Office 2007 Ribbon Bar Icons, Core Set 1
- Office 2007 Ribbon Bar Icons, Core Set 2
- Office 2007 Ribbon Bar Icons, Word Processing Set 1
- Office 2007 Ribbon Bar Icons, Word Processing Set 2

The corresponding sets must be purchased in order to use these projects. After purchasing, request this project from Glyfz.

The projects use 16x16 and 32x32 images (32x32 images are used only for several commands)

Additional information

[Additional information is available on the support forum.](#)

1.6.2.4 Fugue Icons toolbar images

Fugue Icons

(deprecated, this datamodule the not maintained anymore)

If you want to use modern looking images, consider [Fugue Icons](#) toolbar images.

Fugue Icons – Copyright © 2009 Yusuke Kamiyamane.

The icons can be used free for any personal or commercial projects under a [Creative Commons Attribution 3.0 license](#).

If you can't or don't want to place link to the icons author website , please visit p.yusukekamiyamane.com and purchase a royalty-free license.

Download a datamodule with these icons from

<http://www.trichview.com/resources/actions/dmactionsfugue.zip>

For Delphi 2009+ only (because of semitransparent images)

The datamodule contains 2 identical sets of icons: with and without shadows.

Installing

It's not necessary to install files to use this datamodule.

However, If you want to use the datamodule at design time instead of dmActions.pas:

1. Open **RichViewActions.inc** and remove the dot from one of the following line:
{.\$DEFINE USEFUGUEICONS} // to use dmActionsFugue (1 byte transparency, D2009+)
2. Remove dmActions.pas from the RichViewActions package
3. Install (or recompile) the package.

Additional information

[Additional information is available on the support forum.](#)

1.6.2.5 FamFamFam Silk Icons toolbar images

FamFamFam Silk Icons

(deprecated, this datamodule the not maintained anymore)

A FamFamFam Icons datamodule for RichViewActions is available.

You can download it from <http://www.trichview.com/resources/actions/dmactionssilk.zip>

For Delphi 2009+ only (because of semitransparent images)

Silk Icons - Copyright © Mark James, <http://www.famfamfam.com>.

The icons are licensed under a Creative Commons Attribution 2.5 or 3.0 License, details can be found on the author's website.

Unfortunately, this set contains not too many icons for table operations.

Additional information

[Additional information is available on the support forum.](#)

1.7 Procedures and Functions

Procedures and functions declared in RichViewActions

- GoToCheckpoint⁽³⁷⁹⁾ moves the caret to the specified checkpoint;
- RVA_ChooseStyle⁽³⁸⁰⁾ allows choosing and applies Delphi VCL themes (requires Delphi XE2 or

newer);

- RVA_EditorControlFunctionDef⁽³⁸⁰⁾ allows using actions with other editor classes;
- RVA_ConvertToPixels, RVA_ConvertToTwips, RVA_ConvertToEMU. RVA_ConvertToUnits procedures⁽³⁸⁰⁾
- RvHtmlHelp function⁽³⁸¹⁾

See also

- Localization procedures and functions declared in RichViewActions unit⁽³⁵¹⁾

Procedures and functions declared in RVARibbonUtils

This unit contains procedures and functions for working with TRibbon control.

- FillActionClientContents⁽³⁸¹⁾
- RemoveEllipsis⁽³⁸¹⁾, RemoveEllipsisFromRibbon⁽³⁸¹⁾
- LocalizeRibbonPage⁽³⁸²⁾
- SetRulerColors⁽³⁸²⁾

Procedures and functions declared in other units

- LoadCSV⁽³⁸²⁾ loads CSV ("comma-separated values") files as a table;
- MarkSubString⁽³⁸³⁾ marks all occurrences of the specified string in a document;
- NormalizeRichView⁽³⁸⁴⁾ fixes possible problems in a document;
- RVChangeCharCase, RVGetCharCase⁽³⁸⁵⁾ set/get character case of the selected text.

1.7.1 RichViewActions unit

Procedures and functions declared in RichViewActions

- GoToCheckpoint⁽³⁷⁹⁾ moves the caret to the specified checkpoint;
- RVA_ChooseStyle⁽³⁸⁰⁾ allows choosing and applies Delphi VCL themes (requires Delphi XE2 or newer);
- RVA_EditorControlFunctionDef⁽³⁸⁰⁾ allows using actions with other editor classes;
- RVA_ConvertToPixels, RVA_ConvertToTwips, RVA_ConvertToEMU. RVA_ConvertToUnits procedures⁽³⁸⁰⁾
- RvHtmlHelp function⁽³⁸¹⁾

See also

- Localization procedures and functions declared in RichViewActions unit⁽³⁵¹⁾

1.7.1.1 GoToCheckpoint function

Moves the caret in **rv** to the checkpoint having the specified name. Optionally, can scroll the editor to show the marked item in the center.

Unit RichViewActions;

```
function GoToCheckpoint(rv: TCustomRichViewEdit;
  const CPName: TRVUnicodeString;
  ScrollToCenter: Boolean = False): Boolean;
```

Returns *True* if the caret was successfully moved. Returns *False* if the checkpoint is not found.

This function is used in `TrvActionInsertHyperlink.GoToLink`⁽²⁴⁴⁾ methods.

1.7.1.2 PasteHTML procedure (deprecated)

Deprecated. Use `TRichViewEdit.PasteHTML`.

Pastes HTML from the Clipboard to **rve** (if available)

Unit RichViewActions.

```
procedure PasteHTML(rve: TCustomRichViewEdit);
```

1.7.1.3 RVA_ChooseStyle function

Displays a dialog for choosing a visual style (Delphi XE2+) and applies the chosen style.

Unit RichViewActions.

```
function RVA_ChooseStyle: Boolean;
```

This function requires Delphi XE2 or newer. When called from applications compiled in older version of Delphi, it displays an error message.

1.7.1.4 RVA_EditorControlFunctionDef function

The function allows for actions inherited from `TrvCustomEditAction`⁽¹⁴⁷⁾ working with additional editor types.

Unit RichViewActions;

```
function RVA_EditorControlFunctionDef(Control: TControl;  
    Command: TRVAEditorControlCommand): Boolean;
```

This function is a default value of `RVA_EditorControlFunction`⁽³⁹¹⁾ variable.

It allows working with:

- `TCustomEdit` (`TEdit`, `TMemo`, `TRichEdit` and others);
- `TComboBox` (having `csDropDown` or `csSimple` style).

1.7.1.5 RVA_ConvertTo* procedures

The procedures convert all properties measured in `TRVAControlPanel`⁽³⁹⁾.`UnitsProgram`⁽⁵⁷⁾ for all actions on the **Form**.

Unit RichViewActions.

```
procedure RVA_ConvertToTwips(Form: TComponent; ARVStyle: TRVStyle);  
procedure RVA_ConvertToPixels(Form: TComponent; ARVStyle: TRVStyle);  
procedure RVA_ConvertToEMU(Form: TComponent; ARVStyle: TRVStyle);  
procedure RVA_ConvertToUnits(Form: TComponent; ARVStyle: TRVStyle;  
    NewUnits: TRVStyleUnits);
```

Form is a form or datamodule. This procedure changes values of properties of all actions (inherited from `TrvCustomAction`⁽³³⁵⁾) owned by this form/datamodule.

RVA_ConvertToPixels must be called before assigning *rvstuPixels* to TRVAControlPanel⁽³⁹⁾.UnitsProgram⁽⁵⁷⁾.

RVA_ConvertToTwips must be called before assigning *rvstuTwips* to TRVAControlPanel⁽³⁹⁾.UnitsProgram⁽⁵⁷⁾.

RVA_ConvertToEMU must be called before assigning *rvstuEMU* to TRVAControlPanel⁽³⁹⁾.UnitsProgram⁽⁵⁷⁾.

RVA_ConvertToUnits must be called before assigning **NewUntits** to TRVAControlPanel⁽³⁹⁾.UnitsProgram⁽⁵⁷⁾.

The property **ARVStyle**.UnitsPixelsPerInch is used when converting to/from pixels (this is a number of pixels in one logical inch).

These procedures do not change value of TRVAControlPanel⁽³⁹⁾.UnitsProgram⁽⁵⁷⁾.

If you have actions linked to different control panels on this form/datamodule, you must perform this conversion for all these control panels at the same time (first, call **RVA_ConvertTo***, next, change UnitsProgram⁽⁵⁷⁾ for all these control panels).

1.7.1.6 RvHtmlHelp function

Opens an HTML help file (*.CHM)

Unit RichViewActions.

function RvHtmlHelp (HelpFileName: TRVUnicodeString): Boolean;

HelpFileName is a name of CHM file, for example 'RichViewActions.chm'. It may include a topic name, for example 'RichViewActions.chm::/RvHtmlHelp.htm' displays this topic.

1.7.2 RVARibbonUtils unit

Procedures and functions declared in RVARibbonUtils

This unit contains procedures and functions for working with TRibbon control.

- FillActionClientContents⁽³⁸¹⁾
- RemoveEllipsis⁽³⁸¹⁾, RemoveEllipsisFromRibbon⁽³⁸¹⁾
- LocalizeRibbonPage⁽³⁸²⁾
- SetRulerColors⁽³⁸²⁾

1.7.2.1 FillActionClientContents procedure

For all action items in Manager, copies Action.Hint to CommandProperties.Content;

Unit RVARibbonUtils⁽³⁸¹⁾.

procedure FillActionClientContents (Manager: TActionManager);

1.7.2.2 RemoveEllipsis, RemoveEllipsisFromRibbon procedures

Unit RVARibbonUtils⁽³⁸¹⁾.

procedure RemoveEllipsis (Form: TComponent);

procedure RemoveEllipsisFromRibbon (Manager: TActionManager);

RemoveEllipsis removes ending '...' from captions of all actions on the form. Not recommended for using.

RemoveEllipsisFromRibbon removes ending '...' from captions of all Manager.ActionBars[].Items[].

1.7.2.3 LocalizeRibbonPage procedure

Assigns Caption and KeyTip to the ribbon **Page**.

Unit RVARibbonUtils⁽³⁸¹⁾.

procedure LocalizeRibbonPage(Page: TRibbonPage; CaptionID: TRVAMessageID);

This procedure assigns the message identified by **CaptionID** to **Page.Caption**, after removing '&' from it. If it finds '&', it assigns a hot key from this message to **Page.KeyTip**.

1.7.2.4 SetRulerColors procedure

Changes **Ruler**'s colors according to **ColorMap**.

Unit RVARibbonUtils⁽³⁸¹⁾.

procedure SetRulerColors(Ruler: TRVRuler⁽⁸²⁾;
ColorMap: TCustomRibbonColorMap);

1.7.3 Other units

Procedures and functions declared in other units

- LoadCSV⁽³⁸²⁾ loads CSV ("comma-separated values") files as a table;
- MarkSubString⁽³⁸³⁾ marks all occurrences of the specified string in a document;
- NormalizeRichView⁽³⁸⁴⁾ fixes possible problems in a document;
- RVChangeCharCase, RVGetCharCase⁽³⁸⁵⁾ set/get character case of the selected text.

1.7.3.1 LoadCSV function

The function loads CSV ("comma-separated values") files as a table. Only ANSI files can be loaded.

Unit rvcsv.

function LoadCSV(const FileName: String; rv: TCustomRichView;
Separator: TRVAnsiChar; UseQuotes: Boolean;
TextStyleNo, ParaNo: Integer): TRVTableItemInfo;

Parameters:

FileName – file to load.

rv – TCustomRichView component where you plan to insert the loaded table (it is not inserted by this function).

Separator – character used to separate values. The most common possible values:

- ',' (comma)
- ';' (semicolon)
- ' ' (space)
- #9 (tab)

UseQuotes – if *True*, double-quotes are used to allow separators inside text values (in the input file); double double-quotes are treated as double-quote character.

TextStyleNo, ParaNo – styles of text and paragraph used to add text

Return value:

the loaded table; *nil* in case of failure.

Example:

```
var table: TRVTableItemInfo;
...
table := LoadCSV(FileName, RichViewEdit1, ',', True, 0, 0);
if table<>nil then
    RichViewEdit1.InsertItem(',', table);
```

You can implement inserting CSV files using `TRVAControlPanel.OnCustomFileOperation` ⁶⁴ event.

1.7.3.2 MarkSubString functions

The functions mark all occurrences of **s** in **rv**.

Unit MarkSearch.

type

```
TRVMarkStringOption = (
    rvmsIgnoreCase, rvmsWholeWords, rvmsNormalize,
    rvmsStripDiacritic, rvmsIgnorePunctuation);
TRVMarkStringOptions = set of TRVMarkStringOption;
```

```
function MarkSubStringA(const s: TRVAnsiString;
    Color, BackColor: TColor;
    Options: TRVMarkStringOptions; rv: TCustomRichView;
    RVData: TCustomRVData = nil): Integer;
function MarkSubStringW(const s: TRVUnicodeString;
    Color, BackColor: TColor;
    Options: TRVMarkStringOptions; rv: TCustomRichView;
    RVData: TCustomRVData = nil): Integer;
```

All occurrences of **s** are marked with the specified **Color** and **BackColor**.

RVData may specify the document to process. If it is *nil*, the whole document in **rv** is processed (i.e. **RVData** = *nil* works like **RVData** = **rv.RVData**).

Options for the both functions:

Option	Meaning
<i>rvmsIgnoreCase</i>	If set, character case is ignored when comparing
<i>rvmsWholeWords</i>	If set, only occurrences as whole words are marked. "Word" is surrounded by characters listed in rv.Delimiters property.

Options for the both `MarkSubStringW`:

Option	Meaning
<i>rvmsNormalize</i>	If set, text is normalized when comparing (Compatibility Decomposition form). With this option, you can search for '1' and find '①' (and vice versa); or you can search for '卜' and find 'ㄣ'.
<i>rvmsStripDiacritic</i>	Works like <i>rvmsNormalize</i> , and also all diacritic characters are ignored. For example, you can search for 'Ä' and find 'A' (and vice versa).
<i>rvmsIgnorePunctuation</i>	If set, any sequence of punctuation characters (i.e. characters listed in rv.Delimiters property) is treated as a single space character. For example, you can search for 'Hello world.' and find 'Hello, world!!!' (and vice versa).

This operation is not an editing operation, it cannot be undone.

These functions are not used in RichViewActions. They are included in RichViewActions for a convenience.

Limitation: this function searches text in each text item separately.

Platform notes:

- *rvmsIgnoreCase* for Unicode text works only on WinNT-based systems (WinNT4, Win2000, WinXP and newer)
- *rvmsNormalize* and *rvmsStripDiacritic* work in Windows Vista and newer. For WinXP, it requires libraries from <https://www.microsoft.com/en-us/download/details.aspx?id=734>

1.7.3.3 NormalizeRichView procedure

Fixes possible problems in TCustomRichView component.

Unit RVNormalize.

```
procedure NormalizeRichViewNormalizeRichView(RVData: TCustomRVData;
  FirstItemNo: Integer=0; Recursive: Boolean=True);
```

This procedure does not update undo/redo buffer in editor, so it must be called before the first editing operations after clearing and creating/loading document.

This procedure:

- fixes incorrect "new line" flags
- fixes paragraphs consisting only of list markers without items
- fixes incorrect ParaNo assignment
- removes empty text items (from places where they should not be)
- combines adjacent text items of the same text style and tag

This procedure fixes issues 1-3 and 5-7 listed in the TRichView help topic "Valid Documents"

Example:

```
NormalizeRichView(rv.RVData);
rv.Format;
```

Optional parameters:

FirstItemNo defines the first item to check. The procedure checks items from FirstItemNo to RVData.ItemCount-1. By default, all items are checked.

If **Recursive**=*True*, the procedure checks subdocuments of RVData (table cells).

1.7.3.4 RVChangeCharCase, RVGetCharCase procedures

These procedures allow changing character case in the selected fragment of TCustomRichViewEdit component.

Unit RVCharCase;

type

```
TRVCharCase = (rvccLowerCase, rvccUpperCase, rvccTitleWord);
```

```
procedure RVChangeCharCase(rve: TCustomRichViewEdit;  
    CharCase: TRVCharCase);
```

```
procedure RVGetCharCase(rve: TCustomRichViewEdit;  
    var AllUpperCase, AllLowerCase: Boolean);
```

RVChangeCharCase changes character case in the selection of **rve** according to **CharCase** parameter.

Option	Meaning
<i>rvccLowerCase</i>	change all characters to lower case
<i>rvccUpperCase</i>	change all characters to upper case
<i>rvccTitleWord</i>	change the first character of each word to upper case, other characters to lower case.

RVGetCharCase returns *True* in **AllUpperCase** parameter if all characters in the selection are in upper case.

RVGetCharCase returns *True* in **AllLowerCase** parameter if all characters in the selection are in lower case.

See also:

- TrvActionCharCase⁽¹³²⁾

1.8 Types

Other Types Declared in RichViewActions

Classes of properties:

- TRVAHFInfo⁽³⁸⁶⁾ – properties of a header/footer drawn by RichViewActions;
- TRVFileFormatComponent⁽³⁸⁹⁾ – a component providing file saving or loading.

Strings:

- TRVASpellString, TRVASpellStrings, TRVASpellStringList⁽³⁸⁷⁾ – string types for user interface.

Files:

- TrvFileExportFilter, TrvFileSaveFilter³⁸⁸ – saving file format;
- TrvFileImportFilter, TrvFileOpenFilter³⁸⁹ – loading file format.

Rulers:

- TRulerUnits³⁸⁶

Events:

- TRVAEditEvent³⁸⁶
- TRVShowFormEvent³⁹⁰

1.8.1 TRulerUnits

Measure units used in TRVRuler⁸².

Unit RVRulerBase;

type

```
TRulerUnits = (ruInches, ruCentimeters, ruMillimeters,
  ruPicas, ruPixels, ruPoints);
```

Value	Units
<i>ruInches</i>	Inches
<i>ruCentimeters</i>	Centimeters
<i>ruMillimeters</i>	Millimeters
<i>ruPicas</i>	Picas (1 pica = 1/6 of an inch = 12 points)
<i>ruPixels</i>	Pixels (1 pixel = 1/UnitsPixelsPerInch ⁸⁷ of an inch)
<i>ruPoints</i>	Points (1 point = 1/72 of an inch)

1.8.2 TRVAEditEvent, TRVAEditEvent2

These types are used for several events.

type

```
TRVAEditEvent = procedure (Sender: TrvAction313;
  Edit: TCustomRichViewEdit) of object;
TRVAEditEvent2 = procedure (Sender: TrvCustomAction335;
  Edit: TCustomRichViewEdit) of object;
```

1.8.3 TRVAHFInfo

TRVAHFInfo is the class used for storing properties of a header and a footer in RichViewActions¹⁶.

Unit RichViewActions

Syntax

```
TRVAHFInfo = class (TPersistent);
```

Hierarchy

TObject

TPersistent

Description

This is the type of TRVAControlPanel.Header and Footer⁽⁴⁹⁾ properties.

This class has the following properties.

Text: String – text displayed in header/footer. This text may contain the following codes:

- &p – index of the current page (counted from TRVAControlPanel.FirstPageNumber⁽⁴⁸⁾);
- &P – count of pages;
- &d – current date (DateToStr)
- &t – current time (DateTimeToString(..., ShortTimeFormat, Time))
- && – '&' character.

You can implement your own one-letter codes using OnGetHeaderFooterCode⁽⁶⁷⁾ event.

Default value: '' (empty string) for footer, '- &p -' for header.

Alignment: TAlignment – horizontal alignment of text. Default value: *taCenter*.

PrintOnFirstPage: Boolean – specifies whether the header/footer is printed on the first page.

Default value: *True*.

Font: TRVAFont – font for printing text. Default value: Arial, 10 (where TRVAFont = class(TFont)).

RichViewActions print header and footer by assigning a temporal handler of TRVPrint.OnPagePrepaint event. If TRVPrint component (specified in TRVAControlPanel.RVPrint⁽⁵⁵⁾) already has this event assigned, the temporal event handler calls the old handler before drawing text.

1.8.4 TRVASpellString, TRVASpellStrings, TRVASpellStringList

Unit RVASpellInterface.

If TNT Controls are not used, these strings are Unicode in Delphi 2009 and newer, and ANSI in older versions of Delphi:

type

```
TRVASpellListString = String;
TRVASpellStrings = TStrings;
TRVASpellStringList = TStringList;
```

If TNT Controls are used⁽³⁷²⁾, these strings are always Unicode:

type

```
TRVASpellListString = WideString;
TRVASpellStrings = TTntStrings;
TRVASpellStringList = TTntStringList;
```

1.8.5 TrvFileExportFilter, TrvFileSaveFilter

Types for Filter property of "Save As" and "Export" actions.

Unit RichViewActions;

type

```
TrvFileExportFilter = (ffeRVF, ffeRTF, ffeXML, ffeTextANSI,
    ffeTextUnicode, ffeCustom, ffeDocX, ffeMarkdown, ffeHTMLCSS,
    ffeHTML, ffeOfficeConverters);
TrvFileSaveFilter = ffeRVF..ffeHTMLCSS;
TrvFileExportFilterSet = set of TrvFileExportFilter;
TrvFileSaveFilterSet = set of TrvFileSaveFilter;
```

The declaration above includes the full set of values, but the actual declaration depends on the third-party tools used in the project.

Value	File Type
<i>ffeRVF</i>	RichView Format (RVF)
<i>ffeRTF</i>	Rich Text Format (RTF)
<i>ffeXML</i>	Format of RichViewXML ⁽³⁷⁰⁾ (XML)
<i>ffeTextANSI</i>	Text files (TXT). The action asks user to select encoding (from Windows code pages, UTF-8, UTF-16)
<i>ffeTextUnicode</i>	Unicode (UTF-16) text files (TXT)
<i>ffeCustom</i>	Custom file formats defined in CustomFilter property of the corresponding action
<i>ffeHTMLCSS</i>	HTML files (HTM, HTML) saved by TCustomRichViewEdit.SaveHTMLEx method
<i>ffeHTML</i>	Simplified HTML files (HTM, HTML) saved by TCustomRichViewEdit.SaveHTML method
<i>ffeMarkdown</i>	Markdown (MARKDOWN, MDOWN, MKDN, MD, MKD, MDWN, MDTXT, MDTEXT)
<i>ffeDocX</i>	Microsoft Word 2007+ Format (DocX)
<i>ffeOfficeConverters</i>	Formats supported by Microsoft Office text export converters

See also:

- TrvActionSaveAs.Filter⁽¹³¹⁾
- TrvActionExport.Filter⁽¹¹⁵⁾

1.8.6 TRVFileFormatComponent

TRVFileFormatComponent is a base class for components allowing to load and save TRichView documents in files of various formats.

Unit RVFile;

Syntax

```
TRVFileFormatComponent = class (TComponent);
TRVHTMLComponent = class (TRVFileFormatComponent);
TRVXMLComponent = class (TRVFileFormatComponent);
```

Hierarchy

TObject
TPersistent
TComponent

Description

Although RVFile unit is included in TRichView, not in RichViewActions, we describe it here, because it is used in RichViewActions.

This class is not used directly. Instead, the following inherited components are used:

- TRichViewXML⁽³⁷⁰⁾: allows loading, saving, and inserting XML files (an alternative to RVF);

These components are not included in RichViewActions. However, they are installed by TRichView Setup (if possible). RichViewActions can use them; but if you do not use them in your application, their code is not linked to your exe-file.

See also properties of TRVAControlPanel:

- XMLComponent⁽⁶⁰⁾

1.8.7 TrvFileImportFilter, TrvFileOpenFilter

Types for Filter property of "Open" and "Insert File" actions.

Unit RichViewActions;

type

```
TrvFileImportFilter = (ffiRVF, ffiRTF, ffiXML, ffiTextANSI,
  ffiTextUnicode, ffiTextAuto, ffiCustom, ffiDocX, ffiMarkdown,
  ffiHTML, ffiOfficeConverters);
TrvFileOpenFilter = ffiRVF..ffiHTML;
TrvFileImportFilterSet = set of TrvFileImportFilter;
TrvFileOpenFilterSet = set of TrvFileOpenFilter;
```

The declaration above includes the full set of values, but the actual declaration depends on the third-party tools used in the project.

ffiXML can be used only if RichViewXML⁽³⁷⁰⁾ is used.

Value	File Type
<i>ffiRVF</i>	RichView Format (RVF)
<i>ffiRTF</i>	Rich Text Format (RTF)

<i>ffiXML</i>	Format of RichViewXML ⁽³⁷⁰⁾ (XML)
<i>ffiTextANSI</i>	Text files (TXT). The action asks user to select encoding (from Windows code pages, UTF-8, UTF-16)
<i>ffiTextUnicode</i>	Unicode (UTF-16) text files (TXT)
<i>ffiTextAuto</i>	ANSI or Unicode (UTF-16) text files (TXT), autodetected
<i>ffiCustom</i>	Custom file formats defined in CustomFilter property of the corresponding action
<i>ffiDocX</i>	Microsoft Word 2007+ Format (DocX)
<i>ffiMarkdown</i>	Markdown (MARKDOWN, MDOWN, MKDN, MD, MKD, MDWN, MDTXT, MDTEXT)
<i>ffiHTML</i>	HTML files (HTM, HTML)
<i>ffiOfficeConverters</i>	Formats supported by Microsoft Office text export converters

See also:

- TrvActionOpen.Filter⁽¹¹⁰⁾
- TrvActionInsertFile.Filter⁽²³²⁾

1.8.8 TRVShowFormEvent

This type is used in the following events:

- TrvActionCustomInfoWindow.OnShowing⁽²²¹⁾
- TrvCustomEditorAction.OnShowing⁽³⁴⁰⁾
- TrvCustomEditorAction.OnFormCreate⁽³⁴⁰⁾

type

```
TRVShowFormEvent = procedure (Sender: TrvAction(313);
    Form: TForm) of object;
```

1.9 Global variables

Global Variables Used in RichViewActions

- MainRVAControlPanel⁽³⁹⁰⁾ – the default control panel component;
- RVA_EditForceDefControl⁽³⁹¹⁾ specified priority of components when applying actions;
- RVA_EditorControlFunction⁽³⁹¹⁾ – a function implementing operations on third-party editor control;
- ShowUntranslatedControls⁽³⁹²⁾ – shows/hides untranslated UI messages.

1.9.1 MainRVAControlPanel

The main control panel for RichViewActions.

Unit RichViewActions;

```
var MainRVAControlPanel: TRVAControlPanel(39);
```

If no other control panels are created, this variable returns a default control panel object (not belonging to any form or datamodule).

If an application contains a single TRVControlPanel on a form, it becomes the main control panel automatically, and MainRVAControlPanel returns this control panel.

Another control panel can be made a main one if you call its `Activate`⁽⁶¹⁾ method.

1.9.2 RVA_EditForceDefControl

Specifies a priority of non-TCustomRichViewEdit editors for editing actions.

var

```
RVA_EditForceDefControl: Boolean = False;
```

By default, focused non-TCustomRichViewEdit controls have higher priority in editing actions⁽¹⁴⁷⁾ (such as Cut, Copy, Paste) than Control⁽³¹⁴⁾ or GetControlPanel⁽³³⁸⁾.DefaultControl⁽⁴³⁾ properties.

For example, in the ActionTest demo, if you select text in the font name combo box and press **Ctrl + X**, the selected text from this combo box will be cut to the Clipboard, the action ignores RVAControlPanel1.DefaultControl⁽⁴³⁾=RichViewEdit1.

If you want to change this priority, assign RVA_EditForceDefControl := True.

1.9.3 RVA_EditorControlFunction

This variable refers to the function allowing for the actions inherited from TrvCustomEditAction⁽¹⁴⁷⁾ to work with additional editor types.

Unit RichViewActions;

type

```
TRVAEditorControlCommand = (rvaeccIsEditorControl, rvaeccHasSelection,
    rvaeccCanPaste, rvaeccCanPasteText,
    rvaeccCanUndo, rvaeccCopy, rvaeccCut,
    rvaeccPaste, rvaeccPasteText, rvaeccUndo);
TRVAEditorControlCommandFunction =
    function (Control: TControl;
        Command: TRVAEditorControlCommand): Boolean;
```

var

```
RVA_EditorControlFunction: TRVAEditorControlCommandFunction;
```

By default, this variable is initialized with RVA_EditorControlFunctionDef⁽³⁸⁰⁾ function.

Command	The function must...
<i>rvaeccIsEditorControl</i>	... return True if Control is a supported editor control.
<i>rvaeccHasSelection</i>	... return True if the selection in Control is not empty
<i>rvaeccCanPaste</i>	... return True if Control can paste data from the Clipboard.
<i>rvaeccCanPasteText</i>	... return True if Control can paste text from the Clipboard.
<i>rvaeccCanUndo</i>	... return True if Control can undo the last editing operation

<i>rvaeccCopy</i>	... copy the selection from Control to the Clipboard
<i>rvaeccCut</i>	... cut the selection from Control to the Clipboard
<i>rvaeccPaste</i>	... paste data from the Clipboard into Control
<i>rvaeccPasteText</i>	... paste text from the Clipboard into Control
<i>rvaeccUndo</i>	... undo the last editing operation in Control

1.9.4 ShowUntranslatedControls

This variable allows hiding controls in RichViewActions dialog that were not translated yet.

Unit RichViewActions;

var

```
ShowUntranslatedControls: Boolean = True;
```

1.10 How to

How To...

- How to change RVF and XML format name⁽³⁹²⁾

1.10.1 How to change RVF and XML format name

By default, RVF (RichView Format) is the main file format of RichViewActions.

If you want to change file extension and the format name, you can do it in the following way.

Place RVAControlPanel1: TRVAControlPanel⁽³⁹⁾ component on the form. Change the following properties (in the code or in the Object Inspector):

```
RVAControlPanel1.RVFLocalizable(54) := False;
RVAControlPanel1.RVFFilter(54) := 'SuperApp Files (*.saf)|*.saf';
RVAControlPanel1.DefaultExt(44) := 'saf';
RVAControlPanel1.RVFormatTitle(54) := 'SuperApp Format';
```

Additionally, you can change names of files used by TrvActionStyleTemplates⁽²¹⁴⁾ to save and load style templates:

```
RVAControlPanel1.RVStylesFilter(55) := 'SuperApp Styles (*.sas)|*.sas';
RVAControlPanel1.RVStylesExt(55) := 'sas';
```

Now, if you want to localize these properties⁽³⁴²⁾, you need to assign them yourself every time when the application language is changed.

For XML format, use XMLLocalizable⁽⁶⁰⁾ and XMLFilter⁽⁶⁰⁾ properties.

Index

- A -

ActionEditNote 268
 TrvActionInsertNote 268
 ActionFind 137, 144
 TrvActionFindNext 137
 TrvActionReplace 144
 ActionFont 35, 37
 TCustomRVFontComboBox 35
 TCustomRVFontListBox 37
 ActionInsertHLine 328
 TrvActionItemProperties 328
 ActionInsertHyperlink 332
 TrvActionRemoveHyperlinks 332
 ActionInsertTable 305, 328
 TrvActionItemProperties 328
 TrvActionTableProperties 305
 ActionList 71
 TRVAPopupMenu 71
 TRVASPTBXPopupMenu 71
 TRVATBPopupMenu 71
 TRVATBXPopupMenu 71
 ActionNew 110
 TrvActionOpen 110
 ActionPageSetup 121
 TrvActionPrintPreview 121
 ActionParaBullets 210
 TrvActionParaList 210
 ActionParaNumbering 210
 TrvActionParaList 210
 ActionPrint 121
 TrvActionPrintPreview 121
 ActionReplace 135
 TrvActionFind 135
 Actions
 Edit 132
 File 98
 Font 148
 Notes and text boxes 259
 Other 313
 Spelling check and thesaurus 312
 Tables 272
 ActionSave 104, 131
 TrvActionCustomNew 104
 TrvActionSaveAs 131

ActionSaveAs 125
 TrvActionSave 125
 ActionsEnabled 42
 TRVAControlPanel 42
 ActionStyleTemplates 218, 237
 TrvActionAddStyleTemplate 218
 TrvActionInsertHyperlink 237
 Activate 61
 TRVAControlPanel 61
 AddColorNameToHints 42
 TRVAControlPanel 42
 AddDefaultCharset 75
 TRVFontCharsetComboBox 75
 AddFormat 141
 TrvActionPasteSpecial 141
 Addict 3 364
 Alignment 202, 204
 TrvActionParagraph 202, 204
 AllowApplyingTo 326
 TrvActionFillColor 326
 AllowedCharacters 224
 TrvActionBookmarks 224
 AllowFocusEditor 37
 TCustomRVFontListBox 37
 AllowMultiple 302
 TrvActionTableInsertRowsBelow 302
 AllStylesImageIndex 91
 TRVStyleTemplateComboBox 91
 TRVStyleTemplateListBox 91
 ASpell 365
 AutoAddHyperlinkStyleTemplate 237
 TrvActionInsertHyperlink 237
 AutoApplySymbolCharset 153
 TrvActionFontEx 153
 AutoCharset 77, 80
 TRVFontComboBox 77
 TRVFontListBox 80
 AutoDeleteUnusedStyles 42
 TRVAControlPanel 42
 AutoUpdateFileName 101
 TrvActionCustomIO 101

- B -

BackColor 154, 237
 TrvActionFontEx 154
 TrvActionInsertHyperlink 237
 Background 195
 TrvActionParaBorder 195
 BackgroundColor 253

BackgroundColor 253
 TrvActionInsertPicture 253
 BackgroundGraphicFilter 305, 329
 TrvActionItemProperties 329
 TrvActionTableProperties 305
 BackgroundPicture 275
 TrvActionInsertTable 275
 BackgroundProperties 275
 TrvActionInsertTable 275
 BestWidth 276
 TrvActionInsertTable 276
 BiDiModeRuler 84
 TCustomRuler 84
 Border 195
 TrvActionParaBorder 195
 BorderColor 253, 276
 TrvActionInsertPicture 253
 TrvActionInsertTable 276
 BorderHSpacing 276
 TrvActionInsertTable 276
 BorderLightColor 276
 TrvActionInsertTable 276
 BorderStyle 277
 TrvActionInsertTable 277
 BorderVSpacing 277
 TrvActionInsertTable 277
 BorderWidth 253, 277
 TrvActionInsertPicture 253
 TrvActionInsertTable 277
 BoxPosition 261
 TrvActionCustomInsertSidenote 261
 BoxProperties 262
 TrvActionCustomInsertSidenote 262

- C -

CallerControl 321
 TrvActionCustomColor 321
 CanChangeMargins 316
 TrvActionBackground 316
 CanCloseDoc 127
 TrvActionSave 127
 Caption 336
 TrvCustomAction 336
 CellBorderColor 277
 TrvActionInsertTable 277
 CellBorderLightColor 278
 TrvActionInsertTable 278
 CellBorderStyle 278
 TrvActionInsertTable 278
 CellBorderWidth 278
 TrvActionInsertTable 278
 CellHPadding 278
 TrvActionInsertTable 278
 CellHSpacing 279
 TrvActionInsertTable 279
 CellPadding 278
 TrvActionInsertTable 278
 CellVPadding 278
 TrvActionInsertTable 278
 CellVSpacing 279
 TrvActionInsertTable 279
 CharCode 257
 TrvActionInsertSymbol 257
 CharScale 154
 TrvActionFontEx 154
 CharSpacing 154
 TrvActionFontEx 154
 ClearFormatImageIndex 91
 TRVStyleTemplateComboBox 91
 TRVStyleTemplateListBox 91
 CleverComponents 369
 CloseDialog 136, 145
 TrvActionFind 136
 TrvActionReplace 145
 ColBandSize 279
 TrvActionInsertTable 279
 ColCount 280
 TrvActionInsertTable 280
 Color 234, 238, 280, 321
 TrvActionCustomColor 321
 TrvActionInsertHLine 234
 TrvActionInsertHyperlink 238
 TrvActionInsertTable 280
 ColorDialog 42
 TRVAControlPanel 42
 ColorDialogInterface 43
 TRVAControlPanel 43
 Control 314
 TrvAction 314
 ControlPanel 91, 337
 TrvCustomAction 337
 TRVStyleTemplateComboBox 91
 TRVStyleTemplateListBox 91
 CreateDirectoryForHTMLImages 115, 126
 TrvActionExport 115
 TrvActionSave 126
 Cursor 238
 TrvActionInsertHyperlink 238
 CustomFilter 100, 110

CustomFilter 100, 110
 TrvActionCustomFileIO 100
 TrvActionOpen 110

- D -

DefaultCharsetCaption 75
 TRVFontCharsetComboBox 75
 DefaultChecked 305, 329
 TrvActionItemProperties 329
 TrvActionTableProperties 305
 DefaultColor 43
 TRVAControlPanel 43
 DefaultControl 43
 TRVAControlPanel 43
 DefaultCustomFilterIndex 43
 TRVAControlPanel 43
 DefaultDocParameters 44
 TRVAControlPanel 44
 DefaultExt 44, 254
 TRVAControlPanel 44
 TrvActionInsertPicture 254
 DefaultFileFormat 44
 TRVAControlPanel 44
 DefaultFileName 45
 TRVAControlPanel 45
 DefaultMargin 45
 TRVAControlPanel 45
 DefaultPersistent 305, 329
 TrvActionItemProperties 329
 TrvActionTableProperties 305
 DeleteAllTabs 202
 TrvActionParagraph 202
 DetectURL 242
 TrvActionInsertHyperlink 242
 DialogFontName 45
 TRVAControlPanel 45
 DialogFontNameLin 45
 TRVAControlPanel 45
 DialogFontSize 46
 TRVAControlPanel 46
 DialogIcons 46
 TRVAControlPanel 46
 DialogPosition 47
 TRVAControlPanel 47
 DialogTitle 102, 110
 TrvActionCustomIO 102
 TrvActionOpen 110
 DialogZoomPercent 48
 TRVAControlPanel 48

Disabled 337
 TrvCustomAction 337
 DisableWhenUnmodified 126
 TrvActionSave 126
 Documents 126
 TrvActionSave 126
 DoNotImportPicturesAndFiles 61
 TRVAControlPanel 61
 DownloadInterface 21, 48
 TRVAControlPanel 48
 DropDownWidth 77
 TRVFontComboBox 77

- E -

Editor 35, 37, 91
 TCustomRVFontComboBox 35
 TCustomRVFontListBox 37
 TRVStyleTemplateComboBox 91
 TRVStyleTemplateListBox 91
 EncodeTarget 243
 TrvActionInsertHyperlink 243
 EvenColumnsColor 283
 TrvActionInsertTable 283
 EvenRowsColor 283
 TrvActionInsertTable 283
 ExcludeLabel 226
 TrvActionInsertCaption 226
 ExportToFile 116
 TrvActionExport 116
 Expression 230
 TrvActionInsertEquation 230
 ExpressSpellChecker 366

- F -

FamFamFam Silk Icons toolbar images 378
 FileName 102
 TrvActionCustomIO 102
 FillActionClientContents 381
 Filter 110, 115, 131, 232, 254
 TrvActionExport 115
 TrvActionInsertFile 232
 TrvActionInsertPicture 254
 TrvActionOpen 110
 TrvActionSaveAs 131
 FindDoc 128
 TrvActionSave 128
 FindText 136, 144
 TrvActionFind 136

FindText 136, 144
 TrvActionReplace 144

FirstColumnColor 283
 TrvActionInsertTable 283

FirstIndent 203
 TrvActionParagraph 203

FirstPageNumber 48
 TRVAControlPanel 48

Flat 85
 TCustomRuler 85

Font 163, 268
 TrvActionFonts 163
 TrvActionInsertNote 268

FontImageIndex 220
 TrvActionStyleInspector 220

FontName 75, 82, 258
 TrvActionInsertSymbol 258
 TRVFontCharsetComboBox 75
 TRVFontSizeComboBox 82

FontSizeDouble 155
 TrvActionFontEx 155

FontStyleEx 155
 TrvActionFontEx 155

Footer 49
 TRVAControlPanel 49

Form 339
 TrvCustomEditorAction 339

Fugue Icons toolbar images 377

- G -

GetAddictSpellLanguage 353

GetAddictThesLanguage 353

GetControlPanel 338
 TrvCustomAction 338

GetHyperlinkStyleNo 243
 TrvActionInsertHyperlink 243

GetNormalStyleNo 244
 TrvActionInsertHyperlink 244

GetRealDialogFontName 61
 TRVAControlPanel 61

GlyFX toolbar images 376

Glyfz toolbar images 377

GoToCheckpoint 379

GoToLink 244
 TrvActionInsertHyperlink 244

GraphicFilter 330
 TrvActionItemProperties 330

- H -

Header 49
 TRVAControlPanel 49

HeadingRowColor 283
 TrvActionInsertTable 283

HeadingRowCount 280
 TrvActionInsertTable 280

HelpType 49
 TRVAControlPanel 49

Hint 337
 TrvCustomAction 337

HOutermostRule 280
 TrvActionInsertTable 280

HoverBackColor 238
 TrvActionInsertHyperlink 238

HoverColor 239
 TrvActionInsertHyperlink 239

HoverEffects 239
 TrvActionInsertHyperlink 239

HoverUnderlineColor 239
 TrvActionInsertHyperlink 239

How to
 change RVF format name 392

How to change RVF format name 392

HRuleColor 280
 TrvActionInsertTable 280

HRuleWidth 281
 TrvActionInsertTable 281

HTMLComponent 53
 TRVAControlPanel 53

HunSpell 367

- I -

If 178

ImageFileName 317
 TrvActionBackground 317

Images 92, 216, 220
 TrvActionStyleInspector 220
 TrvActionStyleTemplates 216
 TRVStyleTemplateComboBox 92
 TRVStyleTemplateListBox 92

IndentColor 85
 TCustomRuler 85

IndentMax 190
 TrvActionIndentInc 190

IndentSaturation 86
 TCustomRuler 86

IndentStep 186, 188, 209
 TrvActionCustomParaListSwitcher 186
 TrvActionIndent 188
 TrvActionParaList 209

Indy 370

InitialDir 102, 111
 TrvActionCustomIO 102
 TrvActionOpen 111

InitImportPicturesAndFiles 61
 TRVAControlPanel 61

InsertAbove 227
 TrvActionInsertCaption 227

InsertFile 232
 TrvActionInsertFile 232

Interfaces for downloader third-party components in RichViewActions 355

Interfaces for spell checker third-party components in RichViewActions 357

Interfaces for third-party color-dialog components in RichViewActions 362

Interfaces for third-party components in RichViewActions 354

ItemHeight 77, 80
 TRVFontComboBox 77
 TRVFontListBox 80

ItemText 281
 TrvActionInsertTable 281

- J -

JumpToNextNote 264
 TrvActionEditNote 264

- K -

KeepLinesTogether 203
 TrvActionParagraph 203

KeepWithNext 203
 TrvActionParagraph 203

- L -

Language 53
 TRVAControlPanel 53

LastColumnColor 283
 TrvActionInsertTable 283

LastFilterIndex 111
 TrvActionOpen 111

LastLineAlignment 178, 204
 TrvActionAlignCustomJustify 178
 TrvActionParagraph 204

LastRowColor 283
 TrvActionInsertTable 283

LeftIndent 204
 TrvActionParagraph 204

LineSpacing 204
 TrvActionParagraph 204

LineSpacingType 205
 TrvActionParagraph 205

ListLevels 186
 TrvActionCustomParaListSwitcher 186

LoadCSV 382

LoadFile 111
 TrvActionOpen 111

Localization of RichViewActions 342

Localize 93
 TRVStyleTemplateComboBox 93
 TRVStyleTemplateListBox 93

LocalizeRibbonPage 382

LostFormatWarning 127
 TrvActionSave 127

- M -

MainRVAControlPanel 390

MarginColor 85
 TCustomRuler 85

MarginSaturation 86
 TCustomRuler 86

MarginsUnits 117
 TrvActionPageSetup 117

MarkSubStringA 383

MarkSubStringW 383

MaxImageSize 254
 TrvActionInsertPicture 254

Maximized 122
 TrvActionPrintPreview 122

MaxSize 159, 160
 TrvActionFontGrow 159
 TrvActionFontGrowOnePoint 160

MaxSuggestionsCount 71
 TRVAPopupMenu 71
 TRVASPTBXPopupMenu 71
 TRVATBXPopupMenu 71
 TRVATBXPopupMenu 71

MetafileCompatibility 53
 TRVAControlPanel 53

MinSize 165, 167
 TrvActionFontShrink 165
 TrvActionFontShrinkOnePoint 167

- N -

NormalizeRichView 384
 NumberType 228, 249
 TrvActionInsertCustomPageNumber 228
 TrvActionInsertNumSequence 249

- O -

OddColumnsColor 283
 TrvActionInsertTable 283
 OddRowsColor 283
 TrvActionInsertTable 283
 OnAddStyle 62
 TRVAControlPanel 62
 OnApplyHyperlinkToItem 246
 TrvActionInsertHyperlink 246
 OnBackgroundChange 63
 TRVAControlPanel 63
 OnCanApply 330
 TrvActionItemProperties 330
 OnCanPaste 141
 TrvActionPasteSpecial 141
 OnChange 118, 317
 TrvActionBackground 317
 TrvActionPageSetup 118
 OnChoosePicture 63
 TRVAControlPanel 63
 OnClickAllStyles 94
 TRVStyleTemplateComboBox 94
 TRVStyleTemplateListBox 94
 OnCloseTableSizeDialog 284
 TrvActionInsertTable 284
 OnColorSelected 319
 TrvActionUserDefinedColor 319
 OnCreateForm 64
 TRVAControlPanel 64
 OnCustomFileOperation 64
 TRVAControlPanel 64
 OnCustomItemPropertiesDialog 331
 TrvActionItemProperties 331
 OnCustomPaste 142
 TrvActionPasteSpecial 142
 OnDocumentFileChange 128
 TrvActionSave 128
 OnDownload 66
 TRVAControlPanel 66
 OnExecute 324
 TrvActionEvent 324

OnFormCreate 340
 TrvCustomEditorAction 340
 OnGetActionControlCoords 66
 TRVAControlPanel 66
 OnGetColor 320
 TrvActionUserDefinedColor 320
 OnGetHeaderFooterCode 67
 TRVAControlPanel 67
 OnGetHyperlinkTargetFromItem 246
 TrvActionInsertHyperlink 246
 OnGetPreviewFormClass 122
 TrvActionPrintPreview 122
 OnHide 340
 TrvCustomEditorAction 340
 OnHideColorPicker 323
 TrvActionCustomColor 323
 OnHyperlinkForm 246
 TrvActionInsertHyperlink 246
 OnInserting 256, 284
 TrvActionInsertPicture 256
 TrvActionInsertTable 284
 OnInsertText 259
 TrvActionInsertText 259
 OnLiveSpellAdd 71
 TRVAPopupMenu 71
 TRVASPTBXPopupMenu 71
 TRVATBPopupMenu 71
 TRVATBXPopupMenu 71
 OnLiveSpellGetSuggestions 72
 TRVAPopupMenu 72
 TRVASPTBXPopupMenu 72
 TRVATBPopupMenu 72
 TRVATBXPopupMenu 72
 OnLiveSpellIgnoreAll 72
 TRVAPopupMenu 72
 TRVASPTBXPopupMenu 72
 TRVATBPopupMenu 72
 TRVATBXPopupMenu 72
 OnLiveSpellWordReplace 73
 TRVAPopupMenu 73
 TRVASPTBXPopupMenu 73
 TRVATBPopupMenu 73
 TRVATBXPopupMenu 73
 OnMarginsChanged 67
 TRVAControlPanel 67
 OnNew 103
 TrvActionCustomNew 103
 OnOpenFile 112
 TrvActionOpen 112
 OnReplaceAllEnd 145

- OnReplaceAllEnd 145
 - TrvActionReplace 145
- OnReplaceAllStart 145
 - TrvActionReplace 145
- OnReplacing 146
 - TrvActionReplace 146
- OnSave 128
 - TrvActionSave 128
- OnSaving 128
 - TrvActionSave 128
- OnShowColorPicker 323
 - TrvActionCustomColor 323
- OnShowing 142, 221, 340
 - TrvActionCustomInfoWindow 221
 - TrvActionPasteSpecial 142
 - TrvCustomEditorAction 340
- OnShowingDialog 157, 196, 207
 - TrvActionFontEx 157
 - TrvActionParaBorder 196
 - TrvActionParagraph 207
- OnStartEditNote 265
 - TrvActionEditNote 265
- OnStyleNeeded 68
 - TRVAControlPanel 68
- OnUpdate 324
 - TrvActionEvent 324
- OnViewChanged 68
 - TRVAControlPanel 68
- Opacity 322
 - TrvActionCustomColor 322
- OuterHSpacing 255
 - TrvActionInsertPicture 255
- OuterVSpacing 255
 - TrvActionInsertPicture 255
- OutlineLevel 205
 - TrvActionParagraph 205

- P -

- Padding 199
 - TrvActionParaColorAndPadding 199
- ParaImageIndex 220
 - TrvActionStyleInspector 220
- ParaStyleImageIndex 91, 217
 - TrvActionStyleTemplates 217
 - TRVStyleTemplateComboBox 91
 - TRVStyleTemplateListBox 91
- ParaTextStyleImageIndex 91, 217
 - TrvActionStyleTemplates 217
 - TRVStyleTemplateComboBox 91

- TRVStyleTemplateListBox 91
- PasteHTML 380
- Percent 166
 - TrvActionFontShrinkGrow 166
- PixelBorders 53
 - TRVAControlPanel 53
- Polar SpellChecker ActiveX 368
- Preview 78, 80
 - TRVFontComboBox 78
 - TRVFontListBox 80
- PreviewInList 155
 - TrvActionFontEx 155
- Protection 258
 - TrvActionInsertSymbol 258

- R -

- RefStyleTemplateName 271
 - TrvActionInsertSidenote 271
- RemoveEllipsis 381
- RemoveEllipsisFromRibbon 381
- ReplaceText 144
 - TrvActionReplace 144
- Reset 107
 - TrvActionNew 107
- RichViewActions
 - downloading images 369, 370
 - FamFamFam Silk Icons toolbar images 378
 - Fugue Icons toolbar images 377
 - GlyFX toolbar images 376
 - Glyfz toolbar images 377
 - history 21
 - inserting and pasting HTML 370
 - interfaces for downloader third-party components 355
 - interfaces for spell checker third-party components 357
 - interfaces for third-party color-dialog components 362
 - interfaces for third-party components 354
 - localizing 342
 - opening-inserting-saving HTML 371
 - opening-inserting-saving XML 370
 - Style Templates 20
 - Third-party tools 362
 - TNT Controls 372
 - TRichView toolbar images 374
 - Types 385
 - using 16
 - using Addict 3 components 364
 - using ASpell 365

RichViewActions
 using ExpressSpellChecker 366
 using HunSpell 367
 using Polar SpellChecker 368
 using SpTBXLib 372
 using TBX 373
 using Toolbar 2000 373
 RichViewActions overview 16
 RichViewActions unit 351
 RichViewActions.Overview 16
 RichViewEdit 88
 TRVRuler 88
 RichViewXML 370
 RightIndent 205
 TrvActionParagraph 205
 RowBandSize 279
 TrvActionInsertTable 279
 RowCount 281
 TrvActionInsertTable 281
 RowsKeepTogether 281
 TrvActionInsertTable 281
 RowsVAlign 282
 TrvActionInsertTable 282
 Ruler 39
 TCustomRulerItemSelector 39
 RulerSaturation 85, 86
 TCustomRuler 85, 86
 RulerType 86
 TCustomRuler 86
 RVA_ChooseLanguage 351
 RVA_ChooseStyle 380
 RVA_ConvertToEMU 380
 RVA_ConvertToPixels 380
 RVA_ConvertToTwips 380
 RVA_ConvertToUnits 380
 RVA_EditForceDefControl 391
 RVA_EditorControlFunction 391
 RVA_EditorControlFunctionDef 380
 RVA_EnumLanguages 346
 RVA_FillLanguageList 346
 RVA_GetCharset 347
 RVA_GetColorName 352
 RVA_GetHelpFile 347
 RVA_GetLanguageName 348
 RVA_GetPC 348
 RVA_GetPrintingMessage 345
 RVA_GetProgressMessage 345
 RVA_GetS 348
 RVA_GetSH 348

RVA_LocalizeForm 352
 RVA_SetHelpFile 349
 RVA_SwitchLanguage 349
 RVA_TranslateUnits 349
 RVAAddictLanguages unit 353
 RVALocalize unit 343
 RVALocalizeRuler 353
 RVAlocRuler unit 353
 RVARibbonUtils unit 378, 381
 RVChangeCharCase 385
 RVCharCase 385
 rvcsv unit 382
 RVFFilter 54
 TRVAControlPanel 54
 RVFLocalizable 54
 TRVAControlPanel 54
 RVFormatTitle 54
 TRVAControlPanel 54
 RVGetCharCase 385
 RVGetHelpKeyword 354
 RVHelp unit 354
 RVHtmlHelp 381
 RVHTMLImporter 370
 RVHTMLViewImporter 371
 RVNormalize 384
 RVPrint 55
 TRVAControlPanel 55
 RVSetHelpKeyword 354
 RVStylesExt 55
 TRVAControlPanel 55
 RVStylesFilter 55
 TRVAControlPanel 55

- S -

ScreenRes 86
 TCustomRuler 86
 ScrollToCenter 224, 240
 TrvActionBookmarks 224
 TrvActionInsertHyperlink 240
 SearchScope 55
 TRVAControlPanel 55
 SeqName 249
 TrvActionInsertNumSequence 249
 SetRulerColors 382
 SetTags 245
 TrvActionInsertHyperlink 245
 SetToPictures 240
 TrvActionInsertHyperlink 240

ShowAllStyles 92
 TRVStyleTemplateComboBox 92
 TRVStyleTemplateListBox 92

ShowCheckpoints 334
 TrvActionShowSpecialCharacters 334

ShowClearFormat 92
 TRVStyleTemplateComboBox 92
 TRVStyleTemplateListBox 92

ShowReplaceAllSummary 145
 TrvActionReplace 145

ShowSoftPageBreaks 56
 TRVAControlPanel 56

ShowTableSizeDialog 284
 TrvActionInsertTable 284

ShowUntranslatedControls 392

SkinType 87
 TCustomRuler 87

SmartHeadings 92
 TRVStyleTemplateComboBox 92
 TRVStyleTemplateListBox 92

SpaceAfter 205
 TrvActionParagraph 205

SpaceBefore 206
 TrvActionParagraph 206

SpaceFiller 240
 TrvActionInsertHyperlink 240

Spacing 255
 TrvActionInsertPicture 255

SpellInterface 56
 TRVAControlPanel 56

SpTBXLib 372

StandardStyleTemplates 216
 TrvActionStyleTemplates 216

StoreFileNameInItemName 140, 255
 TrvActionInsertPicture 255
 TrvActionPasteSpecial 140

StoreImageFileName 330
 TrvActionItemProperties 330

Style 234, 240
 TrvActionInsertHLine 234
 TrvActionInsertHyperlink 240

Style Templates in RichViewActions 20

StyleTemplateName 241, 262
 TrvActionCustomInsertSidenote 262
 TrvActionInsertHyperlink 241

StyleTemplates 106
 TrvActionNew 106

SubDocEditor 339
 TrvCustomEditorAction 339

SubSuperScriptType 155

TrvActionFontEx 155

SuppressNextErrorMessage 127
 TrvActionSave 127

SymbolPreviewString 78, 80
 TRVFontComboBox 78
 TRVFontListBox 80

- T -

TableGridStyle 56
 TRVAControlPanel 56

TableOptions 282
 TrvActionInsertTable 282

TablePrintOptions 282
 TrvActionInsertTable 282

Tabs 206
 TrvActionParagraph 206

TabsToDelete 206
 TrvActionParagraph 206

TBiDiModeRuler 84

TBX 373

TCustomRuler.BiDiModeRuler 84

TCustomRuler.Flat 85

TCustomRuler.IndentColor 85

TCustomRuler.IndentSaturation 86

TCustomRuler.MarginColor 85

TCustomRuler.MarginSaturation 86

TCustomRuler.RulerColor 85

TCustomRuler.RulerSaturation 86

TCustomRuler.RulerType 86

TCustomRuler.ScreenRes 86

TCustomRuler.SkinType 87

TCustomRuler.UnitsDisplay 87

TCustomRuler.UnitsPixelsPerInch 87

TCustomRuler.UnitsProgram 87

TCustomRuler.Zoom 88

TCustomRulerItemSelector.Ruler 39

TCustomRVFontComboBox 33

TCustomRVFontComboBox.ActionFont 35

TCustomRVFontComboBox.Editor 35

TCustomRVFontListBox 35

TCustomRVFontListBox.ActionFont 37

TCustomRVFontListBox.AllowFocusEditor 37

TCustomRVFontListBox.Editor 37

TerminateHyperlink 245
 TrvActionInsertHyperlink 245

TextStyleImageIndex 91, 217
 TrvActionStyleTemplates 217
 TRVStyleTemplateComboBox 91

- TextStyleImageIndex 91, 217
- TRVStyleTemplateListBox 91
- Title 120, 124
 - TrvActionPrint 120
 - TrvActionQuickPrint 124
- TNT Controls 372
- Toolbar 2000 373
- TRichView toolbar images 374
- TRulerType 86
- TRulerUnits 386
- TRVA2LiveSpellAddEvent type 71
- TRVA2LiveSpellGetSuggestionsEvent type 72
- TRVA2LiveSpellIgnoreAllEvent 72
- TRVA2LiveSpellReplaceEvent type 73
- TRVAAddictSpellInterface 358
- TRVAASpellInterface 359
- TRVABoxPosition 261
- TRVABoxProperties 262
- TRVACanPasteEvent 141
- TRVACIDownloadInterface 355
- TrvaColorInterface 322
- TRVAControlPanel 39
 - TRVAControlPanel.ActionsEnabled 42
 - TRVAControlPanel.Activate 61
 - TRVAControlPanel.AddColorNameToHints 42
 - TRVAControlPanel.AutoDeleteUnusedStyles 42
 - TRVAControlPanel.ColorDialog 42
 - TRVAControlPanel.ColorDialogInterface 43
 - TRVAControlPanel.DefaultColor 43
 - TRVAControlPanel.DefaultControl 43
 - TRVAControlPanel.DefaultCustomFilterIndex 43
 - TRVAControlPanel.DefaultDocParameters 44
 - TRVAControlPanel.DefaultExt 44
 - TRVAControlPanel.DefaultFileFormat 44
 - TRVAControlPanel.DefaultFileName 45
 - TRVAControlPanel.DefaultMargin 45
 - TRVAControlPanel.DialogFontName 45
 - TRVAControlPanel.DialogFontNameLin 45
 - TRVAControlPanel.DialogFontSize 46
 - TRVAControlPanel.DialogIcons 46
 - TRVAControlPanel.DialogPosition 47
 - TRVAControlPanel.DialogZoomPercent 48
 - TRVAControlPanel.DonImportPicturesAndFiles 61
 - TRVAControlPanel.DownloadInterface 48
 - TRVAControlPanel.FirstPageNumber 48
 - TRVAControlPanel.Footer 49
 - TRVAControlPanel.GetRealDialogFontName 61
 - TRVAControlPanel.Header 49
 - TRVAControlPanel.HelpType 49
 - TRVAControlPanel.HTMLComponent 53
 - TRVAControlPanel.InitImportPicturesAndFiles 61
 - TRVAControlPanel.Language 53
 - TRVAControlPanel.MetafileCompatibility 53
 - TRVAControlPanel.OnAddStyle 62
 - TRVAControlPanel.OnBackgroundChange 63
 - TRVAControlPanel.OnChoosePicture 63
 - TRVAControlPanel.OnCreateForm 64
 - TRVAControlPanel.OnCustomFileOperation 64
 - TRVAControlPanel.OnDownload 66
 - TRVAControlPanel.OnGetActionControlCoords 66
 - TRVAControlPanel.OnGetHeaderFooterCode 67
 - TRVAControlPanel.OnMarginsChanged 67
 - TRVAControlPanel.OnStyleNeeded 68
 - TRVAControlPanel.OnViewChanged 68
 - TRVAControlPanel.PixelBorders 53
 - TRVAControlPanel.RVFFilter 54
 - TRVAControlPanel.RVFLocalizable 54
 - TRVAControlPanel.RVFormatTitle 54
 - TRVAControlPanel.RVPrint 55
 - TRVAControlPanel.RVStylesExt 55
 - TRVAControlPanel.RVStylesFilter 55
 - TRVAControlPanel.SearchScope 55
 - TRVAControlPanel.ShowSoftPageBreaks 56
 - TRVAControlPanel.SpellInterface 56
 - TRVAControlPanel.TableGridStyle 56
 - TRVAControlPanel.UnitsDisplay 56
 - TRVAControlPanel.UnitsProgram 57
 - TRVAControlPanel.UseDefaultHelpFile 57
 - TRVAControlPanel.UseHelpFiles 58
 - TRVAControlPanel.UserInterface 58
 - TRVAControlPanel.UseTextCodePageDialog 59
 - TRVAControlPanel.UseXPTThemes 59
 - TRVAControlPanel.XMLComponent 60
 - TRVAControlPanel.XMLFilter 60
 - TRVAControlPanel.XMLLocalizable 60
- TrvAction 313
 - TrvAction.Control 314
 - TrvActionAddStyleTemplate 217
 - TrvActionAddStyleTemplate.ActionStyleTemplates 218
 - TrvActionAlignCenter 176
 - TrvActionAlignCustomJustify 177
 - TrvActionAlignCustomJustify.LastLineAlignment 178
 - TrvActionAlignCustomJustify.UseLastLineAlignment 178
 - TrvActionAlignDistribute 178
 - TrvActionAlignJustify 179
 - TrvActionAlignLeft 180
 - TrvActionAlignment 180
 - TrvActionAlignRight 181

- TrvActionBackground 315
- TrvActionBackground.CanChangeMargins 316
- TrvActionBackground.ImageFileName 317
- TrvActionBackground.OnChange 317
- TrvActionBookmarks 223
- TrvActionBookmarks.AllowedCharacters 224
- TrvActionBookmarks.ScrollToCenter 224
- TrvActionCharCase 132
- TrvActionClearBoth 182
- TrvActionClearFormat 221
- TrvActionClearLeft 182
- TrvActionClearNone 183
- TrvActionClearRight 183
- TrvActionClearTextFlow 184
- TrvActionClearTextFormat 221
- TrvActionColor 317
- TrvActionCopy 133
- TrvActionCustomColor 320
- TrvActionCustomColor.CallerControl 321
- TrvActionCustomColor.Color 321
- TrvActionCustomColor.OnHideColorPicker 323
- TrvActionCustomColor.OnShowColorPicker 323
- TrvActionCustomColor.Opacity 322
- TrvActionCustomColor.UserInterface 322
- TrvActionCustomFileIO 98
- TrvActionCustomFileIO.CustomFilter 100
- TrvActionCustomInfoWindow 218
- TrvActionCustomInfoWindow.OnShowing 221
- TrvActionCustomInsertSidenote 260
- TrvActionCustomInsertSidenote.BoxPosition 261
- TrvActionCustomInsertSidenote.BoxProperties 262
- TrvActionCustomInsertSidenote.StyleTemplateName 262
- TrvActionCustomIO 100
- TrvActionCustomIO.AutoUpdateFileName 101
- TrvActionCustomIO.DialogTitle 102
- TrvActionCustomIO.FileName 102
- TrvActionCustomIO.InitialDir 102
- TrvActionCustomNew 103
- TrvActionCustomNew.ActionSave 104
- TrvActionCustomNew.OnNew 103
- TrvActionCustomParaListSwitcher 184
- TrvActionCustomParaListSwitcher.IndentStep 186
- TrvActionCustomParaListSwitcher.ListLevels 186
- TrvActionCut 133
- TrvActionEditNote 262
- TrvActionEditNote.JumpToNextNote 264
- TrvActionEditNote.OnStartEditNote 265
- TrvActionEvent 323
- TrvActionEvent.OnExecute 324
- TrvActionEvent.OnUpdate 324
- TrvActionExport 112
- TrvActionExport.CreateDirectoryForHTMLImages 115
- TrvActionExport.ExportToFile 116
- TrvActionExport.Filter 115
- TrvActionFillColor 324
- TrvActionFillColor.AllowApplyingTo 326
- TrvActionFind 134
- TrvActionFind.ActionReplace 135
- TrvActionFind.CloseDialog 136
- TrvActionFind.FindText 136
- TrvActionFindNext 136
- TrvActionFindNext.ActionFind 137
- TrvActionFontAllCaps 148
- TrvActionFontBackColor 149
- TrvActionFontBold 149
- TrvActionFontColor 150
- TrvActionFontCustomColor 151
- TrvActionFontEx 151
- TrvActionFontEx.AutoApplySymbolCharset 153
- TrvActionFontEx.BackColor 154
- TrvActionFontEx.CharScale 154
- TrvActionFontEx.CharSpacing 154
- TrvActionFontEx.FontSizeDouble 155
- TrvActionFontEx.FontStyleEx 155
- TrvActionFontEx.OnShowingDialog 157
- TrvActionFontEx.PreviewInList 155
- TrvActionFontEx.SubSuperScriptType 155
- TrvActionFontEx.UnderlineColor 156
- TrvActionFontEx.UnderlineType 156
- TrvActionFontEx.ValidProperties 156
- TrvActionFontEx.VShift 157
- TrvActionFontGrow 157
- TrvActionFontGrow.MaxSize 159
- TrvActionFontGrowOnePoint 159
- TrvActionFontGrowOnePoint.MaxSize 160
- TrvActionFontItalic 160
- TrvActionFontOverline 161
- TrvActionFonts 161
- TrvActionFonts.Font 163
- TrvActionFonts.UserInterface 163
- TrvActionFontShrink 163
- TrvActionFontShrink.MinSize 165
- TrvActionFontShrinkGrow 165
- TrvActionFontShrinkGrow.Percent 166
- TrvActionFontShrinkOnePoint 166
- TrvActionFontShrinkOnePoint.MinSize 167
- TrvActionFontStrikeout 168

TrvActionFontStyle	168	TrvActionInsertHyperlink.SpaceFiller	240
TrvActionFontStyleEx	169	TrvActionInsertHyperlink.Style	240
TrvActionFontUnderline	170	TrvActionInsertHyperlink.StyleTemplateName	241
TrvActionHide	326	TrvActionInsertHyperlink.TerminateHyperlink	245
TrvActionIndent	186	TrvActionInsertHyperlink.UnderlineColor	241
TrvActionIndent.IndentStep	188	TrvActionInsertHyperlink.ValidProperties	241
TrvActionIndentDec	188	TrvActionInsertNote	267
TrvActionIndentInc	189	TrvActionInsertNote.ActionEditNote	268
TrvActionIndentInc.IndentMax	190	TrvActionInsertNote.Font	268
TrvActionInsertCaption	225	TrvActionInsertNumber	247
TrvActionInsertCaption.ExcludeLabel	226	TrvActionInsertNumSequence	248
TrvActionInsertCaption.InsertAbove	227	TrvActionInsertNumSequence.NumberType	249
TrvActionInsertCustomPageNumber	227	TrvActionInsertNumSequence.SeqName	249
TrvActionInsertCustomPageNumber.NumberType	228	TrvActionInsertNumSequence.UseDefaults	249
TrvActionInsertEndnote	265	TrvActionInsertPageBreak	250
TrvActionInsertEquation	228	TrvActionInsertPageCount	250
TrvActionInsertEquation.Expression	230	TrvActionInsertPageNumber	251
TrvActionInsertEquation.UserInterface	230	TrvActionInsertPicture	251
TrvActionInsertFile	230	TrvActionInsertPicture.BackgroundColor	253
TrvActionInsertFile.Filter	232	TrvActionInsertPicture.BorderColor	253
TrvActionInsertFile.InsertFile	232	TrvActionInsertPicture.BorderWidth	253
TrvActionInsertFootnote	266	TrvActionInsertPicture.DefaultExt	254
TrvActionInsertHLine	233	TrvActionInsertPicture.Filter	254
TrvActionInsertHLine.Color	234	TrvActionInsertPicture.MaxImageSize	254
TrvActionInsertHLine.Style	234	TrvActionInsertPicture.OnInserting	256
TrvActionInsertHLine.Width	234	TrvActionInsertPicture.OuterHSpacing	255
TrvActionInsertHyperlink	234	TrvActionInsertPicture.OuterVSpacing	255
TrvActionInsertHyperlink.ActionStyleTemplates	237	TrvActionInsertPicture.Spacing	255
TrvActionInsertHyperlink.AutoAddHyperlinkStyleTemplate	237	TrvActionInsertPicture.StoreFileNameInItemName	255
TrvActionInsertHyperlink.BackColor	237	TrvActionInsertPicture.VAlign	255
TrvActionInsertHyperlink.Color	238	TrvActionInsertSidenote	269
TrvActionInsertHyperlink.Cursor	238	TrvActionInsertSidenote.RefStyleTemplateName	271
TrvActionInsertHyperlink.DetectURL	242	TrvActionInsertSymbol	256
TrvActionInsertHyperlink.EncodeTarget	243	TrvActionInsertSymbol.CharCode	257
TrvActionInsertHyperlink.GetHyperlinkStyleNo	243	TrvActionInsertSymbol.FontName	258
TrvActionInsertHyperlink.GetNormalStyleNo	244	TrvActionInsertSymbol.Protection	258
TrvActionInsertHyperlink.GoToLink	244	TrvActionInsertSymbol.UseCurrentFont	258
TrvActionInsertHyperlink HoverBackColor	238	TrvActionInsertTable	273
TrvActionInsertHyperlink HoverColor	239	TrvActionInsertTable.BackgroundPicture	275
TrvActionInsertHyperlink HoverEffects	239	TrvActionInsertTable.BackgroundProperties	275
TrvActionInsertHyperlink HoverUnderlineColor	239	TrvActionInsertTable.BestWidth	276
TrvActionInsertHyperlink.OnApplyHyperlinkToItem	246	TrvActionInsertTable.BorderColor	276
TrvActionInsertHyperlink.OnGetHyperlinkTargetFromItem	246	TrvActionInsertTable.BorderHSpacing	276
TrvActionInsertHyperlink.OnHyperlinkForm	246	TrvActionInsertTable.BorderLightColor	276
TrvActionInsertHyperlink.ScrollToCenter	240	TrvActionInsertTable.BorderStyle	277
TrvActionInsertHyperlink.SetTags	245	TrvActionInsertTable.BorderVSpacing	277
TrvActionInsertHyperlink.SetToPictures	240	TrvActionInsertTable.BorderWidth	277
		TrvActionInsertTable.CellBorderColor	277
		TrvActionInsertTable.CellBorderLightColor	278

- TrvActionInsertTable.CellBorderStyle 278
- TrvActionInsertTable.CellBorderWidth 278
- TrvActionInsertTable.CellHPadding 278
- TrvActionInsertTable.CellHSpacing 279
- TrvActionInsertTable.CellPadding 278
- TrvActionInsertTable.CellVPadding 278
- TrvActionInsertTable.CellVSpacing 279
- TrvActionInsertTable.ColBandSize 279
- TrvActionInsertTable.ColCount 280
- TrvActionInsertTable.Color 280
- TrvActionInsertTable.EvenColumnsColor 283
- TrvActionInsertTable.EvenRowsColor 283
- TrvActionInsertTable.FirstColumnColor 283
- TrvActionInsertTable.HeadingRowColor 283
- TrvActionInsertTable.HeadingRowCount 280
- TrvActionInsertTable.HOutermostRule 280
- TrvActionInsertTable.HRuleColor 280
- TrvActionInsertTable.HRuleWidth 281
- TrvActionInsertTable.ItemText 281
- TrvActionInsertTable.LastColumnColor 283
- TrvActionInsertTable.LastRowColor 283
- TrvActionInsertTable.OddColumnsColor 283
- TrvActionInsertTable.OddRowsColor 283
- TrvActionInsertTable.OnCloseTableSizeDialog 284
- TrvActionInsertTable.OnInserting 284
- TrvActionInsertTable.RowBandSize 279
- TrvActionInsertTable.RowCount 281
- TrvActionInsertTable.RowsKeepTogether 281
- TrvActionInsertTable.RowsVAlign 282
- TrvActionInsertTable.ShowTableSizeDialog 284
- TrvActionInsertTable.TableOptions 282
- TrvActionInsertTable.TablePrintOptions 282
- TrvActionInsertTable.VisibleBorders 282
- TrvActionInsertTable.VOutermostRule 282
- TrvActionInsertTable.VRuleColor 283
- TrvActionInsertTable.VRuleWidth 283
- TrvActionInsertText 258
- TrvActionInsertText.OnInsertText 259
- TrvActionInsertTextBox 271
- TrvActionItemProperties 326
- TrvActionItemProperties.ActionInsertHLine 328
- TrvActionItemProperties.ActionInsertTable 328
- TrvActionItemProperties.BackgroundGraphicFilter 329
- TrvActionItemProperties.DefaultChecked 329
- TrvActionItemProperties.DefaultPersistent 329
- TrvActionItemProperties.GraphicFilter 330
- TrvActionItemProperties.OnCanApply 330
- TrvActionItemProperties.OnCustomItemPropertiesDialog 331
- TrvActionItemProperties.StoreImageFileName 330
- TrvActionItemProperties.UpdateAllInsertTableActions 328
- TrvActionLineSpacing 190
- TrvActionLineSpacing100 191
- TrvActionLineSpacing150 191
- TrvActionLineSpacing200 192
- TrvActionNew 105
- TrvActionNew.Reset 107
- TrvActionNew.StyleTemplates 106
- TrvActionOpen 108
- TrvActionOpen.ActionNew 110
- TrvActionOpen.CustomFilter 110
- TrvActionOpen.DialogTitle 110
- TrvActionOpen.Filter 110
- TrvActionOpen.InitialDir 111
- TrvActionOpen.LastFilterIndex 111
- TrvActionOpen.LoadFile 111
- TrvActionOpen.OnOpenFile 112
- TrvActionPageSetup 116
- TrvActionPageSetup.MarginsUnits 117
- TrvActionPageSetup.OnChange 118
- TrvActionPageSetup.UnitsPixelsPerInch 118
- TrvActionParaBiDi 193
- TrvActionParaBorder 193
- TrvActionParaBorder.Background 195
- TrvActionParaBorder.Border 195
- TrvActionParaBorder.OnShowingDialog 196
- TrvActionParaBorder.UserInterface 196
- TrvActionParaBorder.ValidProperties 196
- TrvActionParaBullets 197
- TrvActionParaColor 197
- TrvActionParaColorAndPadding 198
- TrvActionParaColorAndPadding.Padding 199
- TrvActionParaColorAndPadding.UsePadding 200
- TrvActionParaCustomColor 200
- TrvActionParagraph 200
- TrvActionParagraph.Alignment 202, 204
- TrvActionParagraph.DeleteAllTabs 202
- TrvActionParagraph.FirstIndent 203
- TrvActionParagraph.KeepLinesTogether 203
- TrvActionParagraph.KeepWithNext 203
- TrvActionParagraph.LastLineAlignment 204
- TrvActionParagraph.LeftIndent 204
- TrvActionParagraph.LineSpacing 204
- TrvActionParagraph.LineSpacingType 205
- TrvActionParagraph.OnShowingDialog 207
- TrvActionParagraph.OutlineLevel 205
- TrvActionParagraph.RightIndent 205

TrvActionParagraph.SpaceAfter	205	TrvActionSave.DisableWhenUnmodified	126
TrvActionParagraph.SpaceBefore	206	TrvActionSave.Documents	126
TrvActionParagraph.Tabs	206	TrvActionSave.FindDoc	128
TrvActionParagraph.TabsToDelete	206	TrvActionSave.LostFormatWarning	127
TrvActionParagraph.UserInterface	207	TrvActionSave.OnDocumentFileChange	128
TrvActionParagraph.ValidProperties	207	TrvActionSave.OnSave	128
TrvActionParaList	208	TrvActionSave.OnSaving	128
TrvActionParaList.ActionParaBullets	210	TrvActionSave.SuppressNextErrorMessage	127
TrvActionParaList.ActionParaNumbering	210	TrvActionSaveAs	129
TrvActionParaList.IndentStep	209	TrvActionSaveAs.ActionSave	131
TrvActionParaList.UpdateAllActionsOnForm	209	TrvActionSaveAs.Filter	131
TrvActionParaLTR	210	TrvActionSelectAll	146
TrvActionParaNumbering	211	TrvActionShowSpecialCharacters	333
TrvActionParaRTL	212	TrvActionShowSpecialCharacters.ShowCheckpoints	334
TrvActionParaStyles	212	TrvActionSpellingCheck	312
TrvActionPaste	137	TrvActionSSScript	170
TrvActionPasteAsText	138	TrvActionStyleInspector	218
TrvActionPasteSpecial	139	TrvActionStyleInspector.FontImageIndex	220
TrvActionPasteSpecial.AddFormat	141	TrvActionStyleInspector.Images	220
TrvActionPasteSpecial.OnCanPaste	141	TrvActionStyleInspector.ParImageIndex	220
TrvActionPasteSpecial.OnCustomPaste	142	TrvActionStyleInspector.UpdateInfo	220
TrvActionPasteSpecial.OnShowing	142	TrvActionStyleTemplates	214
TrvActionPasteSpecial.StoreFileNameInItemName	140	TrvActionStyleTemplates.Images	216
TrvActionPrint	118	TrvActionStyleTemplates.ParaStyleImageIndex	217
TrvActionPrint.Title	120	TrvActionStyleTemplates.ParaTextStyleImageIndex	217
TrvActionPrintPreview	120	TrvActionStyleTemplates.StandardStyleTemplates	216
TrvActionPrintPreview.ActionPageSetup	121	TrvActionStyleTemplates.TextStyleImageIndex	217
TrvActionPrintPreview.ActionPrint	121	TrvActionSubscript	171
TrvActionPrintPreview.Maximized	122	TrvActionSuperscript	172
TrvActionPrintPreview.OnGetPreviewFormClass	122	TrvActionTableCell	284
TrvActionQuickPrint	122	TrvActionTableCell.AllBorders	285
TrvActionQuickPrint.Title	124	TrvActionTableCell.Border	286
TrvActionRedo	142	TrvActionTableCell.BottomBorder	286
TrvActionRemoveHyperlinks	331	TrvActionTableCell.LeftBorder	287
TrvActionRemoveHyperlinks.ActionInsertHyperlink	332	TrvActionTableCell.NoBorders	288
TrvActionRemovePageBreak	332	TrvActionTableCell.RightBorder	289
TrvActionReplace	143	TrvActionTableCell.Rotation	289
TrvActionReplace.ActionFind	144	TrvActionTableCell.Rotation180	291
TrvActionReplace.CloseDialog	145	TrvActionTableCell.Rotation270	292
TrvActionReplace.FindText	144	TrvActionTableCell.Rotation90	291
TrvActionReplace.OnReplaceAllEnd	145	TrvActionTableCell.RotationNone	290
TrvActionReplace.OnReplaceAllStart	145	TrvActionTableCell.TopBorder	293
TrvActionReplace.OnReplacing	146	TrvActionTableCell.VAlign	293
TrvActionReplace.ReplaceText	144	TrvActionTableCell.VAlignBottom	294
TrvActionReplace.ShowReplaceAllSummary	145	TrvActionTableCell.VAlignDefault	295
TrvActionSave	124	TrvActionTableCell.VAlignMiddle	295
TrvActionSave.ActionSaveAs	125	TrvActionTableCell.VAlignTop	296
TrvActionSave.CanCloseDoc	127	TrvActionTableDeleteCols	297
TrvActionSave.CreateDirectoryForHTMLImages	126	TrvActionTableDeleteRows	297

RichViewActions © trichview.com

TRVATBXPopupMenu.OnLiveSpellAdd	71	TRVFontListBox.Preview	80
TRVATBXPopupMenu.OnLiveSpellGetSuggestions	72	TRVFontListBox.SymbolPreviewString	80
TRVATBXPopupMenu.OnLiveSpellIgnoreAll	72	TRVFontSizeComboBox	81
TRVATBXPopupMenu.OnLiveSpellWordReplace	73	TRVFontSizeComboBox.FontName	82
TRVUserInterface	58	TRVGetFormClassEvent	122
TRVCanApplyEvent	330	TRVGetHyperlinkTargetFromItem	246
TRVCharCase	385	TRVHeaderFooterCodeEvent	67
TRVChoosePictureEvent	63	TRVHTMLComponent	389
TRVCreateFormEvent	64	TRVHyperlinkFormEvent	246
TrvCustomAction	335	TRVHyperlinkProperties	241
TrvCustomAction.Caption	336	TRVHyperlinkProperty	241
TrvCustomAction.ControlPanel	337	TrvPaperMarginsUnits	117
TrvCustomAction.Disabled	337	TRVParaInfoBorderProperties	196
TrvCustomAction.GetControlPanel	338	TRVParaInfoBorderProperty	196
TrvCustomAction.Hint	337	TRVParaInfoMainProperties	207
TrvCustomEditAction	147	TRVParaInfoMainProperty	207
TrvCustomEditorAction	338	TRVRuler	82
TrvCustomEditorAction.Form	339	TRVRuler.RichViewEdit	88
TrvCustomEditorAction.OnFormCreate	340	TRVRuler.UpdateRulerMargins	89
TrvCustomEditorAction.OnHide	340	TRVRulerItemSelector	37
TrvCustomEditorAction.OnShowing	340	TRVSaveFileEvent	128
TrvCustomEditorAction.SubDocEditor	339	TRVShowFormEvent	221, 390
TRVCustomFileOperationEvent	64	TRVSpellInterface	358
TRVCustomItemPropertiesDialog	331	TRVStyleNeededEvent	68
TrvCustomPrintAction	131	TRVStyleTemplateComboBox	89
TrvFileExportFilter	388	TRVStyleTemplateComboBox.AllStylesImageIndex	91
TrvFileExportFilterSet	388	TRVStyleTemplateComboBox.ClearFormatImageIndex	91
TRVFileFormatComponent	389	TRVStyleTemplateComboBox.ControlPanel	91
TrvFileImportFilter	389	TRVStyleTemplateComboBox.Editor	91
TrvFileImportFilterSet	389	TRVStyleTemplateComboBox.Images	92
TrvFileOpenFilter	389	TRVStyleTemplateComboBox.Localize	93
TrvFileOpenFilterSet	389	TRVStyleTemplateComboBox.OnClickAllStyles	94
TrvFileSaveFilter	388	TRVStyleTemplateComboBox.ParaStyleImageIndex	91
TrvFileSaveFilterSet	388	TRVStyleTemplateComboBox.ParaTextStyleImageIndex	91
TRVFontCharsetComboBox	73	TRVStyleTemplateComboBox.ShowAllStyles	92
TRVFontCharsetComboBox.AddDefaultCharset	75	TRVStyleTemplateComboBox.ShowClearFormat	92
TRVFontCharsetComboBox.DefaultCharsetCaption	75	TRVStyleTemplateComboBox.SmartHeadings	92
TRVFontCharsetComboBox.FontName	75	TRVStyleTemplateComboBox.TextStyleImageIndex	91
TRVFontComboBox	75	TRVStyleTemplateListBox.AllStylesImageIndex	91
TRVFontComboBox.AutoCharset	77	TRVStyleTemplateListBox.ClearFormatImageIndex	91
TRVFontComboBox.DropDownWidth	77	TRVStyleTemplateListBox.ControlPanel	91
TRVFontComboBox.ItemHeight	77	TRVStyleTemplateListBox.Editor	91
TRVFontComboBox.Preview	78	TRVStyleTemplateListBox.Images	92
TRVFontComboBox.SymbolPreviewString	78	TRVStyleTemplateListBox.Localize	93
TRVFontInfoMainProperties	156	TRVStyleTemplateListBox.OnClickAllStyles	94
TRVFontInfoMainProperty	156	TRVStyleTemplateListBox.ParaStyleImageIndex	91
TRVFontListBox	78	TRVStyleTemplateListBox.ParaTextStyleImageIndex	91
TRVFontListBox.AutoCharset	80		
TRVFontListBox.ItemHeight	80		

TRVStyleTemplateListBox.ShowAllStyles 92
 TRVStyleTemplateListBox.ShowClearFormat 92
 TRVStyleTemplateListBox.SmartHeadings 92
 TRVStyleTemplateListBox.TextStyleImageIndex 91
 TRVXMLComponent 389
 TSkinType 87
 TZoomRange 88

- U -

UnderlineColor 156, 241
 TrvActionFontEx 156
 TrvActionInsertHyperlink 241
 UnderlineType 156
 TrvActionFontEx 156
 UnitsDisplay 56, 87
 TCustomRuler 87
 TRVAControlPanel 56
 UnitsPixelsPerInch 87, 118
 TCustomRuler 87
 TrvActionPageSetup 118
 UnitsProgram 57, 87
 TCustomRuler 87
 TRVAControlPanel 57
 UpdateAllActionsOnForm 209
 TrvActionParaList 209
 UpdateAllInsertTableActions 305, 328
 TrvActionTableProperties 305
 UpdateAllInsertTableActions.TrvActionItemProperties 328
 UpdateInfo 220
 TrvActionStyleInspector 220
 UpdateRulerMargins 89
 TRVRuler 89
 UseCurrentFont 258
 TrvActionInsertSymbol 258
 UseDefaultHelpFile 57
 TRVAControlPanel 57
 UseDefaults 249
 TrvActionInsertNumSequence 249
 UseHelpFiles 58
 TRVAControlPanel 58
 UseLastLineAlignment 178
 TrvActionAlignCustomJustify 178
 UseOpacity 319
 TrvActionUserDefinedColor 319
 UsePadding 200
 TrvActionParaColorAndPadding 200
 UserInterface 58, 163, 196, 207, 230, 322
 TRVAControlPanel 58

TrvActionCustomColor 322
 TrvActionFonts 163
 TrvActionInsertEquation 230
 TrvActionParaBorder 196
 TrvActionParagraph 207
 UseTextCodePageDialog 59
 TRVAControlPanel 59
 UseXPThemes 59
 TRVAControlPanel 59
 Using RichViewActions 16

- V -

ValidProperties 156, 196, 207, 241
 TrvActionFontEx 156
 TrvActionInsertHyperlink 241
 TrvActionParaBorder 196
 TrvActionParagraph 207
 VAlign 255
 TrvActionInsertPicture 255
 VisibleBorders 282
 TrvActionInsertTable 282
 VOutermostRule 282
 TrvActionInsertTable 282
 VRuleColor 283
 TrvActionInsertTable 283
 VRuleWidth 283
 TrvActionInsertTable 283
 VShift 157
 TrvActionFontEx 157

- W -

Width 234
 TrvActionInsertHLine 234

- X -

XMLComponent 60
 TRVAControlPanel 60
 XMLFilter 60
 TRVAControlPanel 60
 XMLLocalizable 60
 TRVAControlPanel 60

- Z -

Zoom 88
 TCustomRuler 88