

RichViewActions

RichViewActions © trichview.com

Table of Contents

Part I RichViewActions 15

1..Overview 15
Installing RichViewActions 15
Using RichViewActions 15
Style templates 18
History 19
2..Components 25
TRVAControlPanel 26
Properties 27
TRVAControlPanel.ActionsEnabled 27
TRVAControlPanel.AddColorNameToHints 28
TRVAControlPanel.AutoDeleteUnusedStyles 28
TRVAControlPanel.ColorDialog 28
TRVAControlPanel.ColorDialogInterface 28
TRVAControlPanel.DefaultColor 28
TRVAControlPanel.DefaultControl 29
TRVAControlPanel.DefaultCustomFilterIndex 29
TRVAControlPanel.DefaultDocParameters 29
TRVAControlPanel.DefaultExt 30
TRVAControlPanel.DefaultFileFormat 30
TRVAControlPanel.DefaultFileName 30
TRVAControlPanel.DefaultMargin 30
TRVAControlPanel.DialogFontName 31
TRVAControlPanel.DialogFontSize 31
TRVAControlPanel.DownloadInterface 31
TRVAControlPanel.FirstPageNumber 31
TRVAControlPanel.Header, Footer 31
TRVAControlPanel.Language 32
TRVAControlPanel.MetafileCompatibility 32
TRVAControlPanel.PixelBorders 32
TRVAControlPanel.RVFFilter 33
TRVAControlPanel.RVFLocalizable 33
TRVAControlPanel.RVFormatTitle 33
TRVAControlPanel.HTMLComponent 33
TRVAControlPanel.RVPrint 34
TRVAControlPanel.RVStylesFilter, RVStylesExt 34
TRVAControlPanel.SearchScope 34
TRVAControlPanel.ShowSoftPageBreaks 35
TRVAControlPanel.SpellInterface 35
TRVAControlPanel.TableGridStyle 35
TRVAControlPanel.UnitsDisplay 35
TRVAControlPanel.UnitsProgram 35
TRVAControlPanel.UseHelpFiles 36
TRVAControlPanel.UserInterface 36
TRVAControlPanel.UseTextCodePageDialog 37
TRVAControlPanel.UseXPThemes 37
TRVAControlPanel.XMLComponent 37
TRVAControlPanel.XMLFilter 38
TRVAControlPanel.XMLLocalizable 38
Methods 38

TRVAControlPanel.Activate ... 38
TRVAControlPanel.InitImportPictures, DoneImportPictures, DoImportPicture ... 39
Events ... 39
TRVAControlPanel.OnAddStyle ... 39
TRVAControlPanel.OnBackgroundChange ... 40
TRVAControlPanel.OnCreateForm ... 40
TRVAControlPanel.OnCustomFileOperation ... 40
TRVAControlPanel.OnDownload ... 41
TRVAControlPanel.OnGetActionControlCoords ... 42
TRVAControlPanel.OnGetHeaderFooterCode ... 42
TRVAControlPanel.OnMarginsChanged ... 43
TRVAControlPanel.OnStyleNeeded ... 43
TRVAPopupMenu ... 43
Properties ... 45
TRVAPopupMenu.ActionList ... 45
TRVAPopupMenu.MaxSuggestionsCount ... 45
Events ... 45
TRVAPopupMenu.OnLiveSpellAdd ... 45
TRVAPopupMenu.OnLiveSpellGetSuggestions ... 46
TRVAPopupMenu.OnLiveSpellIgnoreAll ... 46
TRVAPopupMenu.OnLiveSpellWordReplace ... 47
TCustomRVFontComboBox ... 47
Properties ... 48
TCustomRVFontComboBox.ActionFont ... 48
TCustomRVFontComboBox.Editor ... 48
TRVFontCharsetComboBox ... 48
Properties ... 49
TRVFontCharsetComboBox.AddDefaultCharset ... 49
TRVFontCharsetComboBox.DefaultCharsetCaption ... 49
TRVFontCharsetComboBox.FontName ... 49
TRVFontComboBox ... 49
Properties ... 50
TRVFontComboBox.DropDownWidth ... 50
TRVFontComboBox.Preview ... 50
TRVFontComboBox.SymbolPreviewString ... 50
TRVFontSizeComboBox ... 50
Properties ... 51
TRVFontSizeComboBox.FontName ... 51
TRVStyleTemplateComboBox, TRVStyleTemplateListBox ... 51
Properties ... 52
TRVStyleTemplateComboBox.*ImageIndex ... 52
TRVStyleTemplateComboBox.ControlPanel ... 52
TRVStyleTemplateComboBox.Editor ... 52
TRVStyleTemplateComboBox.Images ... 53
TRVStyleTemplateComboBox.ShowAllStyles ... 53
TRVStyleTemplateComboBox.ShowClearFormat ... 53
TRVStyleTemplateComboBox.SmartHeadings ... 53
Methods ... 54
TRVStyleTemplateComboBox.Localize ... 54
Events ... 54
TRVStyleTemplateComboBox.OnClickAllStyles ... 54
3..Actions ... 54
File ... 58
TrvActionCustomFileIO ... 58
Properties ... 58
TrvActionCustomFileIO.CustomFilter ... 58
TrvActionCustomIO ... 59

Properties ... 59
TrvActionCustomIO.AutoUpdateFileName ... 59
TrvActionCustomIO.DialogTitle ... 59
TrvActionCustomIO.FileName ... 60
TrvActionCustomIO.InitialDir ... 60
TrvActionCustomNew ... 60
Properties ... 61
TrvActionCustomNew .ActionSave ... 61
Events ... 61
TrvActionCustomNew .OnNew ... 61
TrvActionNew ... 61
Properties ... 62
TrvActionNew .StyleTemplates ... 62
Methods ... 63
TrvActionNew .Reset ... 63
TrvActionOpen ... 63
Properties ... 64
TrvActionOpen.ActionNew ... 64
TrvActionOpen.CustomFilter ... 64
TrvActionOpen.DialogTitle ... 65
TrvActionOpen.Filter ... 65
TrvActionOpen.InitialDir ... 65
TrvActionOpen.LastFilterIndex ... 65
Methods ... 65
TrvActionOpen.LoadFile ... 65
Events ... 66
TrvActionOpen.OnOpenFile ... 66
TrvActionExport ... 66
Properties ... 67
TrvActionExport.CreateDirectoryForHTMLImages ... 67
TrvActionExport.FileTitle ... 67
TrvActionExport.Filter ... 67
TrvActionExport.ImagePrefix ... 68
TrvActionExport.SaveOptions ... 68
Methods ... 68
TrvActionExport.ExportToFile ... 68
TrvActionPageSetup ... 69
Properties ... 69
TrvActionPageSetup.MarginsUnits ... 69
Events ... 70
TrvActionPageSetup.OnChange ... 70
TrvActionPrint ... 70
Properties ... 70
TrvActionPrint.Title ... 70
TrvActionPrintPreview ... 71
Properties ... 71
TrvActionPrintPreview .ActionPageSetup ... 71
TrvActionPrintPreview .ActionPrint ... 71
TrvActionPrintPreview .Maximized ... 72
Events ... 72
TrvActionPrintPreview .OnGetPreview FormClass ... 72
TrvActionQuickPrint ... 72
Properties ... 73
TrvActionQuickPrint.Title ... 73
TrvActionSave ... 73
Properties ... 74
TrvActionSave.ActionSaveAs ... 74
TrvActionSave.CreateDirectoryForHTMLImages ... 74

TrvActionSave.DisableWhenUnmodified ... 74
TrvActionSave.Documents ... 74
TrvActionSave.FileTitle ... 75
TrvActionSave.ImagePrefix ... 75
TrvActionSave.LostFormatWarning ... 75
TrvActionSave.SaveOptions ... 75
TrvActionSave.SuppressNextErrorMessage ... 76
Methods ... 76
TrvActionSave.CanCloseDoc ... 76
TrvActionSave.FindDoc ... 76
Events ... 76
TrvActionSave.OnDocumentFileChange ... 76
TrvActionSave.OnSaving ... 77
TrvActionSaveAs ... 77
Properties ... 78
TrvActionSaveAs.ActionSave ... 78
TrvActionSaveAs.Filter ... 78
TrvCustomPrintAction ... 78
Edit ... 79
TrvActionCharCase ... 79
TrvActionCopy ... 79
TrvActionCut ... 80
TrvActionFind ... 80
Properties ... 81
TrvActionFind.ActionReplace ... 81
Methods ... 81
TrvActionFind.CloseDialog ... 81
TrvActionFindNext ... 81
Properties ... 82
TrvActionFindNext.ActionFind ... 82
TrvActionPaste ... 82
TrvActionPasteAsText ... 82
TrvActionPasteSpecial ... 83
Properties ... 84
TrvActionPasteSpecial.StoreFileNameInItemName ... 84
Methods ... 84
TrvActionPasteSpecial.AddFormat ... 84
Events ... 84
TrvActionPasteSpecial.OnCanPaste ... 84
TrvActionPasteSpecial.OnCustomPaste ... 85
TrvActionPasteSpecial.OnShowing ... 85
TrvActionRedo ... 85
TrvActionReplace ... 86
Properties ... 86
TrvActionReplace.ActionFind ... 86
TrvActionReplace.ShowReplaceAllSummary ... 86
Methods ... 87
TrvActionReplace.CloseDialog ... 87
Events ... 87
TrvActionReplace.OnReplaceAllEnd ... 87
TrvActionReplace.OnReplaceAllStart ... 87
TrvActionReplace.OnReplacing ... 87
TrvActionSelectAll ... 87
TrvActionUndo ... 88
TrvCustomEditAction ... 88
Font ... 89
TrvActionFontAllCaps ... 89
TrvActionFontBackColor ... 89

TrvActionFontBold ... 90
TrvActionFontColor ... 90
TrvActionFontCustomColor ... 91
TrvActionFontEx ... 91
Properties ... 92
TrvActionFontEx.AutoApplySymbolCharset ... 92
TrvActionFontEx.BackColor ... 92
TrvActionFontEx.CharScale ... 93
TrvActionFontEx.CharSpacing ... 93
TrvActionFontEx.FontSizeDouble ... 93
TrvActionFontEx.FontStyleEx ... 94
TrvActionFontEx.Preview InList ... 94
TrvActionFontEx.SubSuperScriptType ... 94
TrvActionFontEx.UnderlineColor ... 94
TrvActionFontEx.UnderlineType ... 94
TrvActionFontEx.ValidProperties ... 95
TrvActionFontEx.VShift ... 95
TrvActionFontGrow ... 95
Properties ... 96
TrvActionFontGrow .MaxSize ... 96
TrvActionFontGrow OnePoint ... 96
Properties ... 96
TrvActionFontGrow OnePoint.MaxSize ... 96
TrvActionFontItalic ... 97
TrvActionFontOverline ... 97
TrvActionFonts ... 98
Properties ... 98
TrvActionFonts.Font ... 98
TrvActionFonts.UserInterface ... 99
TrvActionFontShrink ... 99
Properties ... 99
TrvActionFontShrink.MinSize ... 99
TrvActionFontShrinkGrow ... 99
Properties ... 100
TrvActionFontShrinkGrow .Percent ... 100
TrvActionFontShrinkOnePoint ... 100
Properties ... 101
TrvActionFontShrinkOnePoint.MinSize ... 101
TrvActionFontStrikeout ... 101
TrvActionFontStyle ... 101
TrvActionFontStyleEx ... 102
TrvActionFontUnderline ... 103
TrvActionSSScript ... 103
TrvActionSubscript ... 104
TrvActionSuperscript ... 104
TrvActionTextBiDi ... 105
TrvActionTextLTR ... 106
TrvActionTextRTL ... 106
TrvActionTextStyles ... 107
Paragraph ... 108
TrvActionAlignCenter ... 108
TrvActionAlignCustomJustify ... 108
Properties ... 109
TrvActionAlignCustomJustify.LastLineAlignment, UseLastLineAlignment ... 109
TrvActionAlignDistribute ... 109
TrvActionAlignJustify ... 110
TrvActionAlignLeft ... 110
TrvActionAlignment ... 111

TrvActionAlignRight ... 111
TrvActionClearBoth ... 112
TrvActionClearLeft ... 112
TrvActionClearNone ... 113
TrvActionClearRight ... 113
TrvActionClearTextFlow ... 114
TrvActionCustomParaListSwitcher ... 114
Properties ... 115
TrvActionCustomParaListSwitcher.IndentStep ... 115
TrvActionCustomParaListSwitcher.ListLevels ... 115
TrvActionIndent ... 115
Properties ... 116
TrvActionIndent.IndentStep ... 116
TrvActionIndentDec ... 116
TrvActionIndentInc ... 117
Properties ... 117
TrvActionIndentInc.IndentMax ... 117
TrvActionLineSpacing ... 117
TrvActionLineSpacing100 ... 118
TrvActionLineSpacing150 ... 118
TrvActionLineSpacing200 ... 119
TrvActionParaBiDi ... 119
TrvActionParaBorder ... 120
Properties ... 121
TrvActionParaBorder.Background ... 121
TrvActionParaBorder.Border ... 121
TrvActionParaBorder.UserInterface ... 121
TrvActionParaBorder.ValidProperties ... 121
TrvActionParaBullets ... 122
TrvActionParaColor ... 122
TrvActionParaColorAndPadding ... 123
Properties ... 123
TrvActionParaColorAndPadding.Padding ... 123
TrvActionParaColorAndPadding.UsePadding ... 124
TrvActionParaCustomColor ... 124
TrvActionParagraph ... 124
Properties ... 125
TrvActionParagraph.Alignment ... 125
TrvActionParagraph.DeleteAllTabs ... 125
TrvActionParagraph.FirstIndent ... 125
TrvActionParagraph.KeepLinesTogether ... 126
TrvActionParagraph.KeepWithNext ... 126
TrvActionParagraph.LastLineAlignment ... 126
TrvActionParagraph.LeftIndent ... 126
TrvActionParagraph.LineSpacing ... 127
TrvActionParagraph.LineSpacingType ... 127
TrvActionParagraph.OutlineLevel ... 127
TrvActionParagraph.RightIndent ... 127
TrvActionParagraph.SpaceAfter ... 128
TrvActionParagraph.SpaceBefore ... 128
TrvActionParagraph.Tabs ... 128
TrvActionParagraph.TabsToDelete ... 129
TrvActionParagraph.UserInterface ... 129
TrvActionParagraph.ValidProperties ... 129
TrvActionParaList ... 129
Properties ... 130
TrvActionParaList.IndentStep ... 130
TrvActionParaList.UpdateAllActionsOnForm ... 130

TrvActionParaList.ActionParaNumbering ... 131
TrvActionParaList.ActionParaBullets ... 131
TrvActionParaLTR ... 131
TrvActionParaNumbering ... 132
TrvActionParaRTL ... 132
TrvActionParaStyles ... 133
TrvActionWordWrap ... 134
Styles ... 134
TrvActionStyleTemplates ... 134
Properties ... 135
TrvActionStyleTemplates.Images ... 135
TrvActionStyleTemplates.StandardStyleTemplates ... 135
TrvActionStyleTemplates.*StyleImageIndex ... 136
TrvActionAddStyleTemplate ... 136
Properties ... 136
TrvActionAddStyleTemplate.ActionStyleTemplates ... 136
TrvActionStyleInspector ... 137
Properties ... 137
TrvActionStyleInspector.Images ... 137
TrvActionStyleInspector.FontImageIndex, ParaImageIndex ... 137
Methods ... 138
TrvActionStyleInspector.UpdateInfo ... 138
Events ... 138
TrvActionCustomInfoWindow.OnShowing ... 138
TrvActionClearFormat ... 138
TrvActionClearTextFormat ... 139
Insert ... 139
TrvActionBookmarks ... 139
Properties ... 140
TrvActionBookmarks.AllowedCharacters ... 140
TrvActionBookmarks.ScrollToCenter ... 140
TrvActionInsertCaption ... 140
Properties ... 141
TrvActionInsertCaption.ExcludeLabel ... 141
TrvActionInsertCaption.InsertAbove ... 141
TrvActionInsertCustomPageNumber ... 141
Properties ... 142
TrvActionInsertCustomPageNumber.NumberType ... 142
TrvActionInsertEquation ... 142
Properties ... 143
TrvActionInsertEquation.UserInterface ... 143
TrvActionInsertEquation.Expression ... 143
TrvActionInsertFile ... 143
Properties ... 144
TrvActionInsertFile.Filter ... 144
Methods ... 144
TrvActionInsertFile.InsertFile ... 144
TrvActionInsertHLine ... 144
Properties ... 145
TrvActionInsertHLine.Color ... 145
TrvActionInsertHLine.Style ... 145
TrvActionInsertHLine.Width ... 145
TrvActionInsertHyperlink ... 145
Properties ... 147
TrvActionInsertHyperlink.ActionStyleTemplates ... 147
TrvActionInsertHyperlink.AutoAddHyperlinkStyleTemplate ... 147
TrvActionInsertHyperlink.BackColor ... 147
TrvActionInsertHyperlink.Color ... 147

TrvActionInsertHyperlink.Cursor ... 147
TrvActionInsertHyperlink.HoverBackColor ... 148
TrvActionInsertHyperlink.HoverColor ... 148
TrvActionInsertHyperlink.HoverEffects ... 148
TrvActionInsertHyperlink.HoverUnderlineColor ... 148
TrvActionInsertHyperlink.SetToPictures ... 149
TrvActionInsertHyperlink.SpaceFiller ... 149
TrvActionInsertHyperlink.Style ... 149
TrvActionInsertHyperlink.StyleTemplateName ... 149
TrvActionInsertHyperlink.UnderlineColor ... 150
TrvActionInsertHyperlink.ValidProperties ... 150
Methods ... 150
TrvActionInsertHyperlink.DetectURL ... 150
TrvActionInsertHyperlink.EncodeTarget ... 151
TrvActionInsertHyperlink.GetHyperlinkStyleNo ... 151
TrvActionInsertHyperlink.GetNormalStyleNo ... 152
TrvActionInsertHyperlink.GoToLink ... 152
TrvActionInsertHyperlink.SetTags ... 152
TrvActionInsertHyperlink.TerminateHyperlink ... 153
Events ... 153
TrvActionInsertHyperlink.OnApplyHyperlinkToItem ... 153
TrvActionInsertHyperlink.OnGetHyperlinkTargetFromItem ... 153
TrvActionInsertHyperlink.OnHyperlinkForm ... 154
TrvActionInsertNumber ... 154
TrvActionInsertNumSequence ... 155
Properties ... 155
TrvActionInsertNumSequence.NumberType ... 155
TrvActionInsertNumSequence.SeqName ... 155
TrvActionInsertNumSequence.UseDefaults ... 156
TrvActionInsertPageBreak ... 156
TrvActionInsertPageCount ... 156
TrvActionInsertPageNumber ... 157
TrvActionInsertPicture ... 157
Properties ... 158
TrvActionInsertPicture.BackgroundColor ... 158
TrvActionInsertPicture.BorderWidth, BorderColor ... 158
TrvActionInsertPicture.DefaultExt ... 158
TrvActionInsertPicture.Filter ... 159
TrvActionInsertPicture.MaxImageSize ... 159
TrvActionInsertPicture.OuterHSpacing, OuterVSpacing ... 159
TrvActionInsertPicture.Spacing ... 159
TrvActionInsertPicture.StoreFileNameInItemName ... 160
TrvActionInsertPicture.VAlign ... 160
Events ... 160
TrvActionInsertPicture.OnInserting ... 160
TrvActionInsertSymbol ... 160
Properties ... 161
TrvActionInsertSymbol.CharCode ... 161
TrvActionInsertSymbol.FontName ... 161
TrvActionInsertSymbol.Protection ... 161
TrvActionInsertSymbol.UseCurrentFont ... 161
TrvActionInsertText ... 162
Events ... 162
TrvActionInsertText.OnInsertText ... 162
Notes ... 162
TrvActionCustomInsertSidenote ... 162
Properties ... 163
TrvActionCustomInsertSidenote.BoxPosition ... 163

TrvActionCustomInsertSidenote.BoxProperties ... 164
TrvActionCustomInsertSidenote.StyleTemplateName ... 164
TrvActionEditNote ... 164
Properties ... 165
TrvActionEditNote.JumpToNextNote ... 165
TrvActionInsertEndnote ... 165
TrvActionInsertFootnote ... 166
TrvActionInsertNote ... 167
Properties ... 168
TrvActionInsertNote.ActionEditNote ... 168
TrvActionInsertNote.Font ... 168
TrvActionInsertSidenote ... 168
Properties ... 169
TrvActionInsertSidenote.RefStyleTemplateName ... 169
TrvActionInsertTextBox ... 169
Tables ... 170
TrvActionInsertTable ... 170
Properties ... 171
TrvActionInsertTable.BackgroundPicture ... 171
TrvActionInsertTable.BackgroundStyle ... 171
TrvActionInsertTable.BestWidth ... 171
TrvActionInsertTable.BorderColor ... 171
TrvActionInsertTable.BorderHSpacing ... 171
TrvActionInsertTable.BorderLightColor ... 172
TrvActionInsertTable.BorderStyle ... 172
TrvActionInsertTable.BorderVSpacing ... 172
TrvActionInsertTable.BorderWidth ... 172
TrvActionInsertTable.CellBorderColor ... 173
TrvActionInsertTable.CellBorderLightColor ... 173
TrvActionInsertTable.CellBorderStyle ... 173
TrvActionInsertTable.CellBorderWidth ... 173
TrvActionInsertTable.CellHPadding,CellVPadding ... 173
TrvActionInsertTable.CellHSpacing ... 174
TrvActionInsertTable.CellVSpacing ... 174
TrvActionInsertTable.ColBandSize, RowBandSize ... 174
TrvActionInsertTable.ColCount ... 174
TrvActionInsertTable.Color ... 175
TrvActionInsertTable.HeadingRowCount ... 175
TrvActionInsertTable.HOutermostRule ... 175
TrvActionInsertTable.HRuleColor ... 175
TrvActionInsertTable.HRuleWidth ... 175
TrvActionInsertTable.ItemText ... 176
TrvActionInsertTable.RowCount ... 176
TrvActionInsertTable.RowsKeepTogether ... 176
TrvActionInsertTable.RowsVAlign ... 176
TrvActionInsertTable.TableOptions ... 176
TrvActionInsertTable.TablePrintOptions ... 177
TrvActionInsertTable.VisibleBorders ... 177
TrvActionInsertTable.VOutermostRule ... 177
TrvActionInsertTable.VRuleColor ... 177
TrvActionInsertTable.VRuleWidth ... 177
TrvActionInsertTable's row and column colors ... 178
Methods ... 178
TrvActionInsertTable.ShowTableSizeDialog ... 178
Events ... 178
TrvActionInsertTable.OnCloseTableSizeDialog ... 178
TrvActionInsertTable.OnInserting ... 178
TrvActionTableCell ... 179

TrvActionTableCellAllBorders ... 179
TrvActionTableCellBorder ... 180
TrvActionTableCellBottomBorder ... 180
TrvActionTableCellLeftBorder ... 181
TrvActionTableCellNoBorders ... 182
TrvActionTableCellRightBorder ... 182
TrvActionTableCellRotation ... 183
TrvActionTableCellRotationNone ... 183
TrvActionTableCellRotation90 ... 184
TrvActionTableCellRotation180 ... 185
TrvActionTableCellRotation270 ... 185
TrvActionTableCellTopBorder ... 186
TrvActionTableCellVAlign ... 186
TrvActionTableCellVAlignBottom ... 187
TrvActionTableCellVAlignDefault ... 187
TrvActionTableCellVAlignMiddle ... 188
TrvActionTableCellVAlignTop ... 189
TrvActionTableDeleteCols ... 189
TrvActionTableDeleteRows ... 190
TrvActionTableDeleteTable ... 190
TrvActionTableGrid ... 191
TrvActionTableInsertColLeft ... 191
TrvActionTableInsertColRight ... 192
TrvActionTableInsertRowsAbove ... 192
TrvActionTableInsertRowsBelow ... 193
Properties ... 193
TrvActionTableInsertRowsBelow.AllowMultiple ... 193
TrvActionTableMergeCells ... 193
TrvActionTableMultiCellAttributes ... 194
TrvActionTableProperties ... 194
Properties ... 195
TrvActionTableProperties.ActionInsertTable ... 195
TrvActionTableProperties.BackgroundGraphicFilter ... 195
TrvActionTableRCBase ... 196
TrvActionTableSelectCell ... 196
TrvActionTableSelectCols ... 197
TrvActionTableSelectRows ... 197
TrvActionTableSelectTable ... 198
TrvActionTableSort ... 199
TrvActionTableSplit ... 199
TrvActionTableSplitCells ... 200
TrvActionTableToText ... 200
Spelling check and thesaurus ... 201
TrvActionSpellingCheck ... 201
TrvActionThesaurus ... 201
Others ... 202
TrvAction ... 202
Properties ... 202
TrvAction.Control ... 202
TrvActionBackground ... 202
Properties ... 203
TrvActionBackground.CanChangeMargins ... 203
TrvActionBackground.ImageFileName ... 203
Events ... 204
TrvActionBackground.OnChange ... 204
TrvActionColor ... 204
TrvActionCustomColor ... 204
Properties ... 205

TrvActionCustomColor.CallerControl ... 205
TrvActionCustomColor.Color ... 205
TrvActionCustomColor.UserInterface ... 205
Events ... 206
TrvActionCustomColor.OnShow ColorPicker ... 206
TrvActionCustomColor.OnHideColorPicker ... 206
TrvActionEvent ... 206
Events ... 206
TrvActionEvent.OnExecute ... 206
TrvActionEvent.OnUpdate ... 207
TrvActionFillColor ... 207
Properties ... 207
TrvActionFillColor.Allow ApplyingTo ... 207
TrvActionHide ... 208
TrvActionItemProperties ... 208
Properties ... 209
TrvActionItemProperties.ActionInsertHLine ... 209
TrvActionItemProperties.ActionInsertTable, UpdateAllInsertTableActions ... 209
TrvActionItemProperties.BackgroundGraphicFilter ... 209
TrvActionItemProperties.GraphicFilter ... 210
TrvActionItemProperties.StoreImageFileName ... 210
Events ... 210
TrvActionItemProperties.OnCanApply ... 210
TrvActionItemProperties.OnCustomItemPropertiesDialog ... 211
TrvActionRemoveHyperlinks ... 211
Properties ... 211
TrvActionRemoveHyperlinks.ActionInsertHyperlink ... 211
TrvActionRemovePageBreak ... 211
TrvActionShow SpecialCharacters ... 212
Properties ... 213
TrvActionShow SpecialCharacters.Show Checkpoints ... 213
TrvActionVAlign ... 213
TrvCustomAction ... 213
Properties ... 214
TrvCustomAction.Caption ... 214
TrvCustomAction.ControlPanel ... 214
TrvCustomAction.Disabled ... 214
TrvCustomAction.Hint ... 215
Methods ... 215
TrvCustomAction.GetControlPanel ... 215
TrvCustomEditorAction ... 215
Properties ... 216
TrvCustomEditorAction.Form ... 216
TrvCustomEditorAction.SubDocEditor ... 216
Events ... 216
TrvCustomEditorAction.OnFormCreate ... 216
TrvCustomEditorAction.OnShow ing, OnHide ... 216

.4..Localization of RichViewActions ... 218

RVALocalize unit ... 219

Procedures and Functions ... 220
RVA_EnumLanguages procedure ... 220
RVA_FillLanguageList procedure ... 220
RVA_GetCharset function ... 220
RVA_GetHelpFile function ... 221
RVA_GetLanguageName function ... 221
RVA_GetS function ... 221
RVA_Sw itchLanguage procedure ... 222

RVA_TranslateUnits procedure ... 222
Functions for progress messages ... 222
Types ... 224
TRVALanguageName ... 224
TRVALocString, TRVALocStrings ... 224
RichViewActions unit ... 224
Procedures and functions ... 224
RVA_ChooseLanguage function ... 224
RVA_GetColorName function ... 225
RVA_LocalizeForm procedure ... 225
Other units ... 226
GetAddictSpellLanguage, GetAddictThesLanguage functions ... 226
RVALocalizeRuler procedure ... 226
.5 Interfaces for third-party components ... 226
Download interfaces ... 226
TRVAClDownloadInterface ... 227
TRVACustomDownloadInterface ... 227
TRVAIndyDownloadInterface ... 228
Spelling check interfaces ... 228
TRVAAddictSpellInterface ... 229
TRVASpellInterface ... 230
TRVACustomSpellInterface ... 230
TRVADXSpellInterface ... 231
TRVAHunSpellInterface ... 231
TRVAPolarSpellInterface ... 232
Color dialog interfaces ... 232
.6 Third-party and additional tools ... 232
Components ... 233
Spelling checkers ... 233
Addict 3 and 4 ... 233
ASpell ... 234
ExpressSpellChecker ... 235
HunSpell ... 236
Polar SpellChecker ActiveX ... 236
Downloaders ... 237
CleverComponents ... 237
Indy ... 238
File readers and writers ... 238
RichView XML ... 238
RvHtmlImporter ... 238
RvHtmlView Importer ... 239
User interface ... 239
SpTBXLib ... 239
TNT Controls ... 239
TBX ... 240
Toolbar2000 ... 240
Toolbar Images ... 241
TRichView Icons ... 241
GlyFX toolbar images ... 241
Glyfz toolbar images ... 242
Fugue Icons toolbar images ... 242
FamFamFam Silk Icons toolbar images ... 243
.7 Procedures and Functions ... 243
RichViewActions unit ... 243
ConvertToUnicode procedure ... 243
GoToCheckpoint function ... 244
PasteHTML procedure ... 244

RVA_ChooseStyle function ... 244
RVA_EditorControlFunctionDef function ... 244
RVA_ConvertToPixels, RVA_ConvertToTwips procedures ... 244
RvHtmlHelp function ... 245
RVARibbonUtils unit ... 245
FillActionClientContents procedure ... 245
RemoveEllipsis, RemoveEllipsisFromRibbon procedures ... 245
LocalizeRibbonPage procedure ... 245
SetRulerColors procedure ... 246
Other units ... 246
LoadCSV function ... 246
MarkSubString functions ... 246
NormalizeRichView procedure ... 247
RVChangeCharCase, RVGetCharCase procedures ... 248
.8..Types ... 249
TRVAEditEvent ... 249
TRVAHFInfo ... 249
TRVASpellStrings, TRVASpellStringList ... 249
TrvFileExportFilter, TrvFileSaveFilter ... 250
TRVFileFormatComponent ... 250
TrvFileImportFilter, TrvFileOpenFilter ... 251
TRVShowFormEvent ... 252
.9..Global Variables ... 252
MainRVAControlPanel ... 252
RVA_EditForceDefControl ... 252
RVA_EditorControlFunction ... 253
ShowUntranslatedControls ... 253
10..How to ... 254
How to change RVF and XML format name ... 254

Index 255

1 RichViewActions

RichViewActions is a set of actions for Delphi and C++Builder 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 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 15
- What's new? 19

1.1 Overview

- Installing RichViewActions 15
- Using RichViewActions 15
- Style Templates 18
- What's new 19

1.1.1 Installing RichViewActions

Installing

RichViewActions is installed by the automatic installer together with TRichView.

If you want to use third-party tools 232, read the installation instructions in the topics about them.

Read **Install.txt** for additional information.

1.1.2 Using RichViewActions

Overview

All actions have Control 202 property of TCustomRVControl type. If this property is not assigned, actions work with the focused control (or GetControlPanel 215.DefaultControl 28 component).

All actions have Disabled 214 property (Boolean). If it is *True*, all actions are disabled. If it is *False* (default), actions update their Enabled property automatically.

TRVAControlPanel 26 component has a set of properties affecting all (or many) actions.

TRVAPopupMenu 43 is a popup menu for TCustomRichViewEdit that maintains items itself. Set its ActionList 45 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, 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 linked image-list is 16x16, an action image will be added automatically.

Alternatively, you can use dmActions.pas unit. This is a datamodule containing a ready to use ActionList and ImageList. If you will not modify this unit, you can include it in your project directly. But it's highly recommended to create a copy of this unit (under a new name) and use it instead. Important: if you access actions in OnCreate event of some form, this form must be created after this 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 218.

Ruler

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

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

In TRVAControlPanel.OnMarginsChanged 43 call RVRuler.UpdateRulerMargins.

The ruler supports units: 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 202 property for the following actions:

- TrvActionStyleInspector 137;
- TrvActionEditNote 164.

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 164.

Files

File-related actions (TrvActionNew 60, TrvActionOpen 63, TrvActionSave 73, TrvActionSaveAs 77) 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 66, TrvActionInsertFile 143 are independent – they are not linked with other actions, they just allow to export/insert a file.

Printing

TRVAControlPanel 26 has RVPrint 34: TRVPrint property. All printing actions (TrvActionPrintPreview 71, TrvActionQuickPrint 72, TrvActionPrint 70) 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 26 – ShowSoftPageBreaks 35: Boolean. If True (default), soft page breaks are shown when possible (after print preview or printing – until the next change in the document).

TrvActionPageSetup 69 is a special printing action. It does not require RichViewEdit, but requires RVPrint 34 assigned to TRVAControlPanel 26.

Color

All the main coloring actions (TrvActionFontColor 90, TrvActionFontBackColor 89, TrvActionParaColor 122, TrvActionColor 204) can work in three modes, depending on UserInterface 205 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 205 = *rvacNone*) and one for displaying a color dialog). Or you can implement your own interface for choosing colors.

All coloring actions have OnShowColorPicker 206 and OnHideColorPicker 206 events helping to implement a custom color combo.

Formatting

TrvActionFontGrow 95 and TrvActionFontShink 99 change the size of selected font by the given number of percents (Percent 100 property). If you change its value, do not forget to update hints for the actions.

TrvActionFontGrow 95, TrvActionFontGrowOnePoint 96 have MaxSize property, TrvActionFontShrink 99 TrvActionFontShrinkOnePoint 100 have MinSize property.

TrvActionParagraph 124 can work in two modes, depending on UserInterface 129: 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 91 is similar to TrvActionParagraph 124, it can also work in two modes.

Live spelling check

TRVAPopupMenu supports live spelling. If you use Addict 3-4 233, suggestions, "add to dictionary" and "ignore all" commands appear in the menu automatically. Otherwise, you need to process menu events: OnLiveSpellGetSuggestions 46, OnLiveSpellIgnoreAll 46, OnLiveSpellAdd 45.

If your spellchecker can store a user choice to generate better suggestions in future, you can also process OnLiveSpellWordReplace 47.

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

Bidirected text

Use these actions only if you really need this feature in your application (bidirected text is not completely supported in all versions of Windows).

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

TrvActionParaLTR 131, TrvActionParaRTL 132 – set default bidi mode for the selected paragraphs.

TrvActionTextLTR[106], TrvActionTextRTL[106] – set bidi mode for the selected text.

1.1.3 Style templates

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

TrvActionNew[60], TrvActionOpen[63]

If Editor.UseStyleTemplates=*True*, TrvActionNew resets the Editor.Style's TextStyles and ParaStyles according to StyleTemplates[62], and clears its ListStyles.

If Editor.UseStyleTemplates=*False*, TrvActionNew does not reset styles; it may only delete unused styles (if GetControlPanel[215].AutoDeleteUnusedStyles[28]=*True*). If you want to reset styles, use OnNew[61] event.

TrvActionOpen uses its linked[64] TrvActionNew to perform the same operations before loading.

TrvActionPasteSpecial[83]

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[145]

If Editor.UseStyleTemplates=*True*, the action applies StyleTemplateName[149] 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[150] (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[211] is affected as well.

TrvActionClearFormat[138], TrvActionClearTextFormat[139]

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[134], TrvActionAddStyleTemplate[136]

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

TrvActionInsertFootnote[166], TrvActionInsertEndnote[165], TrvActionInsertSidenote[168]

If Editor.UseStyleTemplates=*True,* TrvActionInsertFootnote[166] inserts footnote characters using "footnote reference" style template, and formats footnote text using "footnote text" style template. TrvActionInsertEndnote[165] uses "endnote reference" and "endnote text" style templates, TrvActionInsertSidenote[168] uses RefStyleTemplateName[169] ("Sidenote Reference" by default) and StyleTemplateName[164] ("Sidenote Text" by default) style templates.

"footnote reference", "footnote text", "endnote reference", "endnote text" are included in TrvActionNew.StyleTemplates[62] by default. "Sidenote Reference" and "Sidenote Text" are not included.

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

TrvActionInsertCaption 140

TrvActionInsertCaption uses "caption" style template. It is included in TrvActionNew.StyleTemplates 62 by default.

1.1.4 History

Changes after version 7 (between TRichView 17 and 18)

Compatibility issues:

- The following properties of TrvActionInsertSymbol 160 are removed: AlwaysInsertUnicode, DisplayUnicodeBlocks, SymbolType
- String parameters are changed to TRVUnicodeString in the events:
  - TRVAControlPanel.OnCustomFileOperation 40, OnDownload 41, OnGetHeaderFooterCode 42.
  - TRVAPopupMenu.OnLiveSpellAdd 45, OnLiveSpellGetSuggestions 46, OnLiveSpellIgnoreAll 46, OnLiveSpellWordReplace 47
  - TrvActionOpen.OnOpenFile 66
  - TrvActionSave.OnSaving 77, OnDocumentFileChange 76
  - TrvActionReplace.OnReplacing 87
  - TrvActionInsertHyperlink.OnApplyHyperlinkToItem 153, OnGetHyperlinkTargetFromItem 153, OnHyperlinkForm 154
  - TrvActionInsertPicture.OnInserting 160
  - TrvActionInsertText.OnInsertText 162
- Support for KSDev ThemeEngine is discontinued

Changes in "Insert Symbol" dialog

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

- TrvActionInsertSymbol 160 has new properties: CharCode 161, FontName 161, Protection 161.

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 246 unit
- No more compiler $defines for using Addict 233, you cannot add RVAddictSpell3 and RVAddictThesaurus3 properties to TRVAControlPanel 26. Use SpellInterface 35 property instead. TrvActionAddictSpell3, TrvActionAddictThesaurus3 are removed, they are superseded by TrvActionSpellingCheck 201 and TrvActionThesaurus 201. RVA_Addict3AutoCorrect function is removed, use TRVAAddictSpellInterface 229's auto-correct methods
- No more compiler $defines for using TIdHttp (Indy 238) and TClHttp (CleverComponents 237), you cannot add IdHttp and ClHttp properties to TRVAControlPanel 26. Use DownloadInterface 31 property instead.

- No more compiler $defines for using RichViewXML 238, RvHtmlImporter 238 and RvHtmlViewImporter 239, you cannot add RVXML, RVHTMLImporter, RVHTMLViewImporter properties to TRVAControlPanel. Use XMLComponent 37 and HTMLComponent 33 properties instead.
- The compiler $define for using TNT Controls 239 is moved from RichViewActions.inc to RV_Defs.inc.
- TrvActionShowSpecialCharacters 212 shows/hides checkpoints by default.
- The following properties are removed: TrvActionPasteSpecial 83.StoreFileName, TrvActionInsertPicture 157.StoreFileName, TrvActionTableProperties 194.StoreImageFileName, TrvActionItemProperties 208.StoreImageFileName. Instead of these properties, the actions check *rvoAssignImageFileNames* in the Options property of the target editor.

New "interface" components

Interface components 226 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 26: ColorDialogInterface 28, DownloadInterface 31, SpellInterface 35.

Interface components for using TdxColorDialog, Indy 238, CleverComponents 237, Addict 233, ASpell 234, ExpessSpellChecker 235, HunSpell 236, Polar SpellChecker 236 are added.

File loading and saving components

Previously, programmers needed to add $defines in RichViewActions.inc to allow using RichViewXML 238, RvHtmlImporter 238 and RvHtmlViewImporter 239 in RichViewActions. Now, you can use XMLComponent 37 and HTMLComponent 33 properties of TRVAControlPanel 26.

Checkpoints (Bookmarks)

New action  TrvActionBookmarks 139 allows adding and managing checkpoints.

In TrvActionInsertHyperlink 145, the editor of the link target is now a combo-box containing a list of checkpoint names.

TrvActionShowSpecialCharacters 212 can show/hide checkpoints.

Mathematical formulas (equations)

New action:  TrvActionInsertEquation 142. TrvActionItemProperties 208 can edit properties of equation items.

Font preview

TRVFontComboBox 49 can display preview of fonts. It has new properties: Preview 50, DropDownWidth 50, SymbolPreviewString 50.

The dialog of TrvActionFontEx 91 can display a preview of font names in the list as well. It has a new property: PreviewInList 94.

Changes related to tables

- TrvActionInsertTable 170 has new properties to assign to table (HeadingRowColor and other colors 178, ColBandSize, RowBandSize 174) and to table rows (RowsVAlign 176, RowsKeepTogether 176).
- The table properties dialog (displayed by TrvActionItemProperties 208 and TrvActionTableProperties 194) allows defining colors of rows and columns.
- new properties TrvActionItemProperties.UpdateAllInsertTableActions 209 and TrvActionTableProperties.UpdateAllInsertTableActions 195 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 156 and TrvActionRemovePageBreak 211, when called from a table cell, add and remove page breaks before the current table row.
- PageNumberType 142 property is added to TrvActionInsertPageCount 156. For inserted "page count" fields, a number type can be changed by TrvActionItemProperties 208.
- New TRVAControlPanel 26.OnGetHeaderFooterCode 42 event.

Other changes

- OnInserting 160 event is added to TrvActionInsertPicture 157.
- new property TRuler.TickColor for drawing ticks and tab stops.
- New untranslated features can be hidden using ShowUntranslatedControls 253 variable.
- MarkSearch 246 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 208 and TrvActionTableProperties 194 update linked TrvActionInsertTable 170 and TrvActionInsertHLine 144 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 109 aligns the selected paragraph to the both left and right sides by adding space between all characters.

 TrvActionInsertPageCount 156 inserts a page count field.

Changes related to paragraph alignments

TrvActionAlignJustify 110 (as well as TrvActionAlignDistribute 109) has new properties: LastLineAlignment and UseLastLineAlignment 109.

TrvActionParagraph 124 has a new property: LastLineAlignment 126.

New properties:

- TRVAControlPanel 26.MetafileCompatibility 32 affects printing of TSRichViewEdit components;
- TrvActionSave 73.DisableWhenUnmodified 74 allows to disable the action for unmodified documents.

TrvActionRemovePageBreak 211 removes a page break from the current paragraph (not the current item as before).

TrvActionItemProperties 208 and TrvActionTableProperties 194 update linked TrvActionInsertTable 170 and TrvActionInsertHLine 144 actions when the user checks "Default" checkbox.

Changes in version 5 (between TRichView 14 and 15)

Compatibility issues:

- TrvActionFontEx[91].Font[98].Size is no longer used. Existing code must be changed to use TrvActionFontEx[91].FontSizeDouble[93] instead.
- 'Francais' cannot be used as a lanuage name; use 'French' or 'Français'.

New actions

- TrvActionInsertTextBox[169] inserts a floating box
- TrvActionInsertFootnote[166] inserts a footnote
- TrvActionInsertEndnote[165] inserts an endnote
- TrvActionInsertSidenote[168] inserts a note in a floating box
- TrvActionEditNote[164] displays a window for editing a note or a floating box
- TrvActionInsertNumber[154] inserts a "numbered sequence" item
- TrvActionInsertCaption[140] inserts a caption for an image or a table
- TrvActionInsertPageNumber[157] inserts a "page number" field

Changes

TrvActionExport[66] can export DocX files (without office converters). Make sure that *ffeDocX* is included in Filter[67].

Changes made by TrvActionColor[204], TrvActionBackground[202] can be undone.

New property: TrvActionFontEx[91].FontSizeDouble[93].

TRVFontSizeComboBox[50] supports fractional font sizes.

Changes in TrvActionItemProperties[208]:

- 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[209] and ActionInsertHLine[209] (in addition to the current item)

Changes in TrvActionNew[61]:

- Reset[63] is made public.
- new item is included in StyleTemplates[62]: 'caption'.

New TrvActionInsertPicture[157] properties to assign to inserted pictures: BackgroundColor[158], BorderWidth, BorderColor[158], OuterHSpacing, OuterVSpacing[159].

New TrvActionInsertTable[170] properties to assign to inserted tables: CellHPadding and CellVPadding[173] (they replace CellPadding), VisibleBorders[177], BackgroundStyle[171], BackgroundPicture[171].

TRVAControlPanel[26].DialogFontName[31]'s default value is changed to 'Tahoma'.

New RemoveEllipsisFromRibbon[245] 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[26].TableGridStyle property is removed.

Redesigned control panel

TRVAControlPanel 26 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 31.

New actions

Styles: 18

 TrvActionStyleTemplates 134
 TrvActionAddStyleTemplate 136
 TrvActionClearFormat 138
 TrvActionClearTextFormat 139
 TrvActionStyleInspector 137

Cell rotation:

 TrvActionTableCellRotationNone 183
 TrvActionTableCellRotation90 184
 TrvActionTableCellRotation180 185
 TrvActionTableCellRotation270 185

Table operations:

 TrvActionTableSplit 199
 TrvActionTableSort 199
 TrvActionTableToText 200

New and improved components

New components for applying style templates 18:

 TRVStyleTemplateComboBox 51
 TRVStyleTemplateListBox 51

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

 TRVFontComboBox 49
 TRVFontSizeComboBox 50
 TRVFontCharsetComboBox 48

 ScaleRichView support

You can assign not only TRichViewEdit, but also TSRichViewEdit to TRVAControlPanel.DefaultControl 29, or Control 202 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 43.

Visual changes

Delphi XE2+'s visual styles are supported in components and dialogs. The function RVA_ChooseStyle 244 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 37=*False*.

New features, properties and events

A new event TRVAControlPanel.OnGetActionControlCoords[42] 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[61] resets styles according to StyleTemplates[62], if style templates are used.
TrvActionOpen[63] can use a linked[64] TrvActionNew to reset document before loading.

TrvActionPasteSpecial[83]:

- allows pasting URLs;
- respects RichViewEdit.AcceptPasteFormats property;
- if style templates[18] are used, displays style modes for pasting RTF and RVF.

TrvActionFontEx[91] can change a text background color (new BackColor[92] property).

TrvActionInsertHyperlink[145] can apply HoverUnderlineColor and HoverEffects properties (if style templates are not used). If style templates are used, it applies the chosen StyleTemplateName[149].

TrvActionTableGrid[191] shows/hides grid only in the target editor.

In the print preview form (TrvActionPrintPreview[71]), a mouse wheel changes pages, Ctrl+mouse wheel zooms in/out

New optional parameters in NormalizeRichView[247].

Localization

New localization functions: RVA_GetProgressMessage, RVA_GetPrintingMessage[222].

New languages:

- Portuguese (European)
- Catalan
- Hindi (in Delphi 4-2007, it is available only if TNT Controls[239] are used);
- Thai

Changes added between TRichView v12 and v13

New units of measurement

All dialogs display values measured according to new TRVAControlPanel[26].UnitsDisplay[35] and TRVAControlPanel.PixelBorders[32] properties.

Properties of several actions are measured in TRVAControlPanel[26].UnitsProgram[35].

New images

A new set of toolbar images[241] 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[29] define a page layout for new documents (especially useful for ScaleRichView)

New TRVAControlPanel[26].DialogFontSize[31] property.

Changes in menus

New TRVAPopupActionBar[43] component (for Delphi 2006 or newer).

The type of ActionList[45] property for menus[43] was changed from TActionList to TCustomActionList, to allow using TActionManager.

Changes in actions

New action: TrvActionHide 208. TrvActionShowSpecialCharacters 212 shows/hides a hidden text as well as special characters.

New properties:

- TrvActionFontEx 91.AutoApplySymbolCharset 92
- TrvActionInsertHLine 144.Style 145 defines the style for inserted horizontal lines
- TrvActionBackground 202.CanChangeMargins 203 allows/disallows changing margins
- TrvActionParaBullets 122's and TrvActionParaNumbering 132's IndentStep 115 defines default indents for list levels.

TrvActionParaList 129.IndentStep 130 now affects indents of list style templates in non-HTML dialog too.

Other changes

New RVARibbonUtils 245 unit.

Ability to use CleverComponents 237 for downloading images.

Changes added between TRichView v11 and v12

New actions:

- TrvActionPasteAsText 82
- TrvActionVAlign 213
- TrvActionClearLeft 112
- TrvActionClearRight 113
- TrvActionClearBoth 112
- TrvActionClearNone 113
- TrvActionRemoveHyperlinks 211

New methods, properties and events

- TrvActionInsertFile.InsertFile 144 method
- TrvActionInsertPicture.Spacing 159 property
- TrvActionParagraph.OutlineLevel 127 property (paragraph dialog has a new combobox for specifying a paragraph outline level)
- TrvActionItemProperties.OnCanApply 210 event

New features:

- New types of vertical alignment for pictures in TrvActionItemProperties 208.
- Properties dialog for breaks (TrvActionItemProperties 208) has a new combobox for specifying a break style (line/rectangle/3d/dotted/dashed).
- Pasting HTML from the Clipboard using rvHtmlViewImporter 239.
- New datamodules with alternative toolbar images (GlyFX 241 and Fugue Icons 242)

1.2 Components

Components

Settings

TRVAControlPanel 26

Popup Menus

TRVAPopupMenu 43

TRVAPopupActionBar 43

TRVATBPopupMenu 43
TRVASPTBXPopupMenu 43
TRVATBXPopupMenu 43

Combo- and List-Boxes

TRVFontComboBox 49
TRVFontSizeComboBox 50
TRVFontCharsetComboBox 48
TRVStyleTemplateComboBox 51
TRVStyleTemplateListBox 51

1.2.1 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 an application does not contain TRVAControlPanel on a form, a default control panel is used (accessible as MainRVAControlPanel 252). If an application contains a single TRVControlPanel on a form, it becomes the main control panel automatically (assigned to MainRVAControlPanel). Another control panel can be made a main one if you call its Activate 38 method.

Linking

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

- TrvCustomAction 213.ControlPanel 214
- TRVStyleTemplateCombo/ListBox 51.ControlPanel 52

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 28 and ColorDialogInterface 28
  - DownloadInterface 31
  - SpellInterface 35
  - HTMLComponent 33

- RVPrint 34
- XMLComponent 37
- properties for customizing user interface:
  - Language 32
  - AddColorNameToHints 28
  - UserInterface 36
  - UseXPThemes 37
  - DialogFontName 31, DialogFontSize 31
  - UnitsDisplay 35, PixelBorders 32
  - UseHelpFiles 36
- properties of new documents:
  - DefaultColor 28
  - DefaultCustomFilterIndex 29
  - DefaultExt 30
  - DefaultFileFormat 30
  - DefaultFileName 30
  - DefaultMargin 30
  - DefaultDocParameters 29
- properties controlling all actions:
  - DefaultControl 29
  - ActionsEnabled 27
- properties for changing names of file formats:
  - RVFFilter 33
  - RVFLocalizable 33
  - RVFormatTitle 33
  - RVStylesFilter 34
  - RVStylesExt 34
  - XMLFilter 38
  - XMLLocalizable 38
- other properties:
  - AutoDeleteUnusedStyles 28
  - FirstPageNumber 31
  - SearchScope 34
  - ShowSoftPageBreaks 35
  - UseTextCodePageDialog 37
  - MetafileCompatibility 32

Events

If you use TRVRuler, process OnMarginsChanged 43.

If you want to implement opening, saving, exporting and inserting files in additional formats, use OnCustomFileOperation 40.

1.2.1.1 Properties

1.2.1.1.1 TRVAControlPanel.ActionsEnabled

Allows to disable all actions.

```
property ActionsEnabled: Boolean;
```

Set to *False* to disable all actions inherited from TrvAction 202, linked to this control panel.

Default value:

True

1.2.1.1.2 TRVAControlPanel.AddColorNameToHints

Allowing adding color names to hints of actions inherited from TrvActionCustomColor[204].

```
property AddColorNameToHints: Boolean;
```

A color name can be added to the action's hint, if its UserInterface[205]=*rvacNone* (i.e. the action applies a specific color without displaying a dialog).

Default value:

True

1.2.1.1.3 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[60] and TrvActionOpen[63].

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[18], this property is ignored: instead of deleting unused styles, "New" and "Open" actions resets styles according to StyleTemplates[62].

Default value:

True

1.2.1.1.4 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 if optional.

This property can be overridden by ColorDialogInterface[28] property.

1.2.1.1.5 TRVAControlPanel.ColorDialogInterface

Specifies a custom color dialog for using in RichViewActions.

```
property ColorDialogInterface: TRVACustomColorDialogInterface[232];
```

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

ColorDialogInterface has a higher priority than ColorDialog[28]: if both **ColorDialogInterface** and ColorDialog[28] are specified, **ColorDialogInterface** is used, ColorDialog[28] is ignored.

1.2.1.1.6 TRVAControlPanel.DefaultColor

Specifies the default background color.

```
property DefaultColor: TColor;
```

This property is assigned to TCustomRichViewEdit.Color property when executing TrvActionNew[60] or TrvActionOpen[63] (before loading).

See also:

- DefaultMargin 30
- DefaultDocParameters 29
- TrvActionNew 61.StyleTemplates 62

1.2.1.1.7 TRVAControlPanel.DefaultControl

Specifies the editor component for using in RichViewActions.

```
property DefaultControl: TCustomRVControl;
```

If this link is defined, all actions inherited from TrvAction 202 (and linked with this control panel) work with this editor. This property can be overridden by Control 202 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 202 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.

1.2.1.1.8 TRVAControlPanel.DefaultCustomFilterIndex

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

```
property DefaultCustomFilterIndex: Integer;
```

This property is used if DefaultFileFormat 30 =*rfeCustom*. This is an index of custom format (from 1); custom formats are listed in TrvActionSaveAs.CustomFilter 58 property.

This is a temporal format assigned to new documents after executing TrvActionNew 60. This format will be offered as a default format on the first execution of TrvActionSave 73 or TrvActionSaveAs 77 for new documents.

OnCustomFileOperation 40 occurs when files of this format need to be saved.

Default value:

1

1.2.1.1.9 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 60 is executed;
- before TrvActionOpen 63 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 30 property is ignored.

See also:

- DefaultColor 28
- TrvActionNew 61.StyleTemplates 62

1.2.1.1.10 TRVAControlPanel.DefaultExt

Specifies a default file extension for dialogs in TrvActionOpen [63], TrvActionSaveAs [77], and TrvActionExport [66] actions.

```
property DefaultExt: TRVALocString [224];
```

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

Default value:

'rvf'

1.2.1.1.11 TRVAControlPanel.DefaultFileFormat

Specifies the default format for new documents.

```
property DefaultFileFormat: TrvFileSaveFilter [250];
```

If it equals to *ffeCustom*, DefaultCustomFilterIndex [29] property defines a subformat.

This is a temporal format assigned to new documents after executing TrvActionNew [60]. This format will be offered as a default format on the first execution of TrvActionSave [73] or TrvActionSaveAs [77] for new documents.

Default value:

ffeRVF

See also:

- DefaultExt [30]

1.2.1.1.12 TRVAControlPanel.DefaultFileName

Specifies a default file name for new documents.

```
property DefaultFileName: TRVALocString [224];
```

This is a temporal file name assigned to new documents after executing TrvActionNew [60]. This file name will be offered as a default file name on the first execution of TrvActionSave [73] or TrvActionSaveAs [77] for new documents.

Default value:

'Untitled.rvf'

See also:

- DefaultExt [30]

1.2.1.1.13 TRVAControlPanel.DefaultMargin

Specifies default margins for new documents.

```
property DefaultMargin: Integer;
```

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

- when TrvActionNew [60] is executed;
- before TrvActionOpen [63] loads a file.

ScaleRichView note: for TSRichViewEdit , this property is ignored, see DefaultDocParameters [29].

Default value:

5

See also:

- DefaultColor [28]
- DefaultDocParameters [29]

- TrvActionNew 61.StyleTemplates 62

1.2.1.1.14 TRVAControlPanel.DialogFontName

Defines the default font name for all dialogs displayed by RichViewActions.

```
property DialogFontName: TFontName;
```

Default value:

'Tahoma'

See also:

- DialogFontSize 31

1.2.1.1.15 TRVAControlPanel.DialogFontSize

Defines the default font size for all dialogs displayed by RichViewActions.

```
property DialogFontSize: Integer;
```

Default value:

8

See also:

- DialogFontName 31

1.2.1.1.16 TRVAControlPanel.DownloadInterface

Allows using a third-party components to download images from HTTP-links.

```
property DownloadInterface: TRVACustomDownloadInterface 227;
```

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

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

- TRVAIndyDownloadInterface 228 (to use Indy 238)
- TRVAClDownloadInterface 227 (to use CleverComponents 237)

See also

- OnDownload 41

1.2.1.1.17 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 31

1.2.1.1.18 TRVAControlPanel.Header, Footer

The properties define the page headers and footers.

```
property Header: TRVAHFInfo 249;
property Footer: TRVAHFInfo 249;
```

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

TrvActionPageSetup 69 changes these properties as well.

The following actions print headers and footers:

- TrvActionPrint 70
- TrvActionQuickPrint 72
- TrvActionPrintPreview 71

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 31

1.2.1.1.19 TRVAControlPanel.Language

Defines the language for user interface.

```
property Language: TRVALanguageName 224;
```

This property returns a language name in English.

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

See Localization of RichViewActions 218.

1.2.1.1.20 TRVAControlPanel.MetafileCompatibility

Specifies whether the printing output is compatible with metafiles.

```
property MetafileCompatibility: Boolean;
```

This property is ignored when printing TRichViewEdit, because TrvActionPrint 70, TrvActionQuickPrint 72, and TrvActionPrintPreview 71 use RVPrint 34.MetafileCompatibility property instead.

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

Default value:

False

1.2.1.1.21 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 35 <> *rvuPixels*.

This property is used by the following actions:

- TrvActionParaBorder 120 for border widths, including an internal width;
- TrvActionTableProperties 194 for widths of a table and cells borders;
- TrvActionItemProperties 208 for widths of a table and cells borders; for width of horizontal lines.

Default value:

False

1.2.1.1.22 TRVAControlPanel.RVFFilter

Defines the file filter for RVF format.

```
property RVFFilter: TRVALocString[224];
```

This value is used only if RVFLocalizable[33]=*False*. This string will appear in file filters in dialogs used in TrvActionOpen[63], TrvActionSaveAs[77], TrvActionExport[66], TrvActionInsertFile[143], TrvActionStyleTemplates[134] (style import) actions.

If RVFLocalizable[33]=*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[34]
- DefaultExt[30]

1.2.1.1.23 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[33], RVFormatTitle[33], RVStylesFilter, RVStylesExt[34] properties are used.

Default value:

True

See also:

- XMLLocalizable[38]
- DefaultExt[30]

1.2.1.1.24 TRVAControlPanel.RVFormatTitle

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

```
property RVFormatTitle: TRVALocString[224];
```

This value is used only if RVFLocalizable[33]=*False*. This string will appear in the dialog used in TrvActionPasteSpecial[83] action and in RVA_GetProgressMessage[222] function.

If RVFLocalizable[33]=*True*, this property is ignored (RVF title is assigned automatically when a new language is applied).

Default value:

'RichView Format'

1.2.1.1.25 TRVAControlPanel.HTMLComponent

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

```
property HTMLComponent: TRVHTMLComponent[250];
```

You can assign either TRVHtmlImporter[238] or TrvHtmlViewImporter[239] component to this property.

If this link is defined, HTML format can be used in TrvActionInsertFile[143], TrvActionPaste[82], and TrvActionPasteSpecial[83] actions.

1.2.1.1.26 TRVAControlPanel.RVPrint

Contains a link to TRVPrint component.

```
property RVPrint: TRVPrint;
```

This link (if defined) is used by TrvActionPrint[70], TrvActionQuickPrint[72], TrvActionPrintPreview[71], TrvActionPageSetup[69] actions.

For TrvActionPageSetup[69], this link *must* be defined, otherwise the action will be disabled (because this action reads/writes margins from/to properties of this component).

1.2.1.1.27 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[224];
property RVStylesExt: TRVALocString[224];
```

These values are used only if RVFLocalizable[33]=*False*. They will appear in file filters in the style import and export dialogs used in TrvActionStyleTemplates[134], TrvActionAddStyleTemplate[136] actions.

If RVFLocalizable[33]=*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[33]

1.2.1.1.28 TRVAControlPanel.SearchScope

Defines a search scope for TrvActionFind[80], TrvActionFindNext[81], and TrvActionReplace[86] actions.

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

property SearchScope: TRVASearchScope;
```

Value	Meaning
rvssFromCursor	Searching from the caret position to the end/beginning of the document.
rvssAskUser	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.
rvssGlobal	Searching from the caret position to the end/beginning of the document, then searching from the beginning/end.

Default value:

rvssAskUser

1.2.1.1.29 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[70], TrvActionQuickPrint[72], or TrvActionPrintPreview[71] actions) until it is changed.

Default value:

True

1.2.1.1.30 TRVAControlPanel.SpellInterface

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

```
property SpellInterface: TRVACustomSpellInterface[230];
```

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

- TRVAAddictSpellInterface[229] (to use Addict[233])
- TRVAASpellInterface[230] (to use ASpell[234])
- TRVADXSpellInterface[231] (to use ExpressSpellChecker[235])
- TRVAHunSpellInterface[231] (to use HunSpell[236])
- TRVAPolarSpellInterface[232] (to use Polar SpellChecker[236])

1.2.1.1.31 TRVAControlPanel.TableGridStyle

Defines the pen style for drawing grid lines in table.

```
property TableGridStyle: TPenStyle;
```

This property is used in TrvActionTableGrid[191] action.

Default value:

psDot

1.2.1.1.32 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[32]=*True*.

Default value:

rvuPixels

1.2.1.1.33 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[61],
- TrvActionIndentInc[117],
- TrvActionIndentDec[116],
- TrvActionParaList[129],
- TrvActionParaBullets[122],

- TrvActionParaNumbering 132,
- TrvActionStyleTemplates 134,
- TrvActionInsertPicture 157,
- TrvActionInsertHLine 144,
- TrvActionInsertTable 170.

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 244 must be called before changing value of this property.

Default value:

rvstuPixels

1.2.1.1.34 TRVAControlPanel.UseHelpFiles

Specifies whether help files should be used if available.

```
property UseHelpFiles: Boolean;
```

Help files can be provided together with translations. A help file name can be specified as the last (optional) parameter of RVA_RegisterLanguage procedure (see RVAL_*.pas files).

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

Current version of RichViewActions does not include any help files.

Default value:

True

1.2.1.1.35 TRVAControlPanel.UserInterface

Allows hiding UI options incompatible with some formats.

```
type
  TRVAUserInterface = (rvauiFull, rvauiHTML, rvauiRTF, rvauiText);

property UserInterface: TRVAUserInterface;
```

Value	Meaning
rvauiFull	All user interface controls are available. Optimal for RVF documents.
rvauiHTML	All user interface controls providing options incompatible with HTML are hidden. The most noticeable changes: • an alternative HTML-style dialog in TrvActionParaList 129 action; • a hidden "Tabs" page in the TrvActionParagraph 124 action's dialog.
rvauiRTF	All user interface controls providing options incompatible with HTML are hidden. The most noticeable changes: • TrvActionColor 204, TrvActionVAlign 213, TrvActionBackground 202, TrvActionTableCellRotation180 185 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
rvauiText	All user interface controls providing options incompatible with a plain text are hidden or disabled.

Default value:

rvauiFull

1.2.1.1.36 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 63
- TrvActionExport 66
- TrvActionSave 73
- TrvActionInsertFile 143

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

1.2.1.1.37 TRVAControlPanel.UseXPThemes

Specifies whether controls on RichViewActions dialog should use Windows XP/Vista/7 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 204

Other controls use themes if it is supported by Delphi (and a manifest is included). If you use Delphi 4-5, you can use a free ***TThemeManager*** component. Newer versions of Delphi have a built-in support of themes.

Default value:

True

1.2.1.1.38 TRVAControlPanel.XMLComponent

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

```
property XMLComponent: TRVXMLComponent 250;
```

You can assign TRichViewXML 238 component to this property.

If this link is defined, XML format can be used in TrvActionOpen[63], TrvActionSaveAs[77], TrvActionExport[66] and TrvActionInsertFile[143] actions.

See also:

- XMLFilter[38]
- XMLLocalizable[38]

1.2.1.1.39 TRVAControlPanel.XMLFilter

Defines the file filter for XML format.

```
property XMLFilter: TRVALocString[224];
```

This property is used if XMLComponent[37] is assigned.

This value is used only if XMLLocalizable[38]=*False*. This string will appear in file filters in dialogs used in TrvActionOpen[63], TrvActionSaveAs[77], TrvActionExport[66], TrvActionInsertFile[143] actions.

If XMLLocalizable[38]=*True*, this property is ignored (XML file filter name is assigned automatically when a new language is applied).

Default value:

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

1.2.1.1.40 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[37] 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[38] property is used.

Default value:

True

See also:

- RVFLocalizable[33]

1.2.1.2 Methods

1.2.1.2.1 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[252] variable.

See also:

- TrvCustomAction.ControlPanel[214]
- TRVStyleTemplateCombo/ListBox.ControlPanel[52]

1.2.1.2.2 TRVAControlPanel.InitImportPictures, DoneImportPictures, DoImportPicture

Allows using DownloadInterface[31] component for a loading operation, not initialized by RichViewActions.

```
procedure InitImportPictures(Sender: TrvAction; OriginalEvent: TRVImportPictureEvent);
procedure DoneImportPictures;
procedure DoImportPicture(Sender: TCustomRichView; const Location: TRVUnicodeString;
  Width, Height: Integer; var Graphic: TGraphic);
```

Normally, when loading is initiated by an action, DownloadInterface[31] is used automatically. The following actions initializes loading:

- TrvActionOpen[63]
- TrvActionInsertFile[143]
- TrvActionPaste[82]
- TrvActionPasteSpecial[83]

However, when loading is not initiated by an action, DownloadInterface[31] is not used. To solve this problem, call **InitImportPictures** before loading, and **DoneImportPicture** after, and assign **DoImportPicture** to OnImportPicture of the target editor.

Parameters:

Sender will be used as a parameter of OnDownload[41] event

OriginalEvent is a procedure that is called (if defined) before the RichViewActions procedure to download images.

Example:

```
procedure TMyForm.MyRichViewEditBeforeOleDrop(Sender: TObject);
begin
  MyRVAControlPanel.InitImportPictures(nil, nil);
  (Sender as TCustomRichViewEdit).OnImportPicture := MyRVAControlPanel.DoImportPicture;
end;
procedure TForm3.RichViewEdit1AfterOleDrop(Sender: TObject);
begin
  (Sender as TCustomRichViewEdit).OnImportPicture := nil;
  MyRVAControlPanel.DoneImportPictures;
end;
```

1.2.1.3 Events

1.2.1.3.1 TRVAControlPanel.OnAddStyle

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

```
type
  TRVAddStyleEvent = procedure (Sender: TrvAction[202]; 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, TParaInfo, or TRVListInfo class.

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

See also:

- OnStyleNeeded[43]
- TCustomRichView.OnAddStyle

1.2.1.3.2 TRVAControlPanel.OnBackgroundChange

Occurs after TrvActionBackground[202] or TrvActionColor[204] actions change background properties.

```
property OnBackgroundChange: TRVAEditEvent[249];
```

1.2.1.3.3 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.

1.2.1.3.4 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: TrvAction[202]; Edit: TCustomRichViewEdit;
    const FileName: TRVUnicodeString; Operation: TRVAFileOperation;
    var SaveFormat: TrvFileSaveFilter[250];
    var CustomFilterIndex: Integer; var Success: Boolean) of object;

property OnCustomFileOperation: TRVCustomFileOperationEvent;
```

This event occurs when the following actions are executed:

- TrvActionOpen[63] (**Operation**=*rvafoOpen*)
- TrvActionSave[73] (**Operation**=*rvafoSave*)
- TrvActionExport[66] (**Operation**=*rvafoExport*)
- TrvActionInsertFile[143] (**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.CustomFilter[64]; on output, it must correspond to TrvActionSaveAs.CustomFilter[58], 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.SuppressNextErrorMessage[76]

You can use LoadCSV[246] function to implement an insertion of CSV files.

Example

By default, RichViewActions allows opening and saving HTML files only with rvHtmlViewImporter[239].

This example allows using rvHtmlImporter[238] for this. Use it at your own risk, because a quality of rvHtmlImporter is not guaranteed.

First, assign CustomFilter property of TrvActionOpen[63] and TrvActionSaveAs[77]. You can do it at design time in the Object Inspector. But if you want to make it localizable, do it at run time in

the localization procedure (in the ActionTest project, do it in the procedure TForm3.Localize):

```
  rvActionsResource.rvActionOpen1.CustomFilter :=  RVA_GetS(221)(rvam_flt_HTMLOpen);
  rvActionsResource.rvActionSaveAs1.CustomFilter :=  RVA_GetS(221)(rvam_flt_HTMLOpen);
```

Make sure that *ffiCustom* is in TrvActionOpen.Filter(65), and *ffeCustom* is in TrvActionSaveAs.Filter(78).

Next, process RVAContolPanel.OnCustomFileOperation event:

```
procedure TForm3.RVAControlPanel1CustomFileOperation(Sender: TrvAction(202);
  Edit: TCustomRichViewEdit; const FileName: TRVUnicodeString;
  Operation: TRVAFileOperation; var SaveFormat: TrvFileSaveFilter(250);
  var CustomFilterIndex: Integer; var Success: Boolean);
var Stream: TFileStream;
    s: TRVRawByteString;
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)
          Stream := TFileStream.Create(FileName, fmOpenRead);
          try
            SetLength(s, Stream.Size);
            Stream.ReadBuffer(PRVAnsiChar(s)^, Length(s));
          finally
            Stream.Free;
          end;
          RvHtmlImporter1.RichView := Edit;
          RvHtmlImporter1.LoadHtml(s);
          // This example assumes that there is only one custom save format,
          // so it does not assign CustomFilterIndex and SaveFormat
          // (they are already 1 and ffeCustom correspondingly)
          Success := True;
        except;
          Success := False;
        end;
      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 := Edit.SaveHTML(FileName, rvActionsResource.rvActionExport1.FileTitle
            rvActionsResource.rvActionExport1.ImagePrefix(68),
            rvActionsResource.rvActionExport1.SaveOptions(68)+[rvsoOverrideImages]);
      end;
  end;
end;
```

1.2.1.3.5 TRVAControlPanel.OnDownload

Occurs when DownloadInterface(31) downloads an image.

```
type
  TRVADownloadEvent = procedure (Sender: TrvAction; const Source: TRVUnicodeString) of ob

property OnDownload: TRVADownloadEvent;
```

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[222]

1.2.1.3.6 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[90]
- TrvActionFontBackColor[89]
- TrvActionParaColor[122]
- TrvActionColor[204]

(if UserInterface[205]=*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;
```

1.2.1.3.7 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[31].Text[249] and Footer[31].Text[249]. It can be edited in the dialog displayed by TrvActionPageSetup[69].

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.

1.2.1.3.8 TRVAControlPanel.OnMarginsChanged

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

```
property OnMarginsChanged: TRVAEditEvent[249];
```

Margins can be changed by TrvActionOpen[63] and TrvActionBackground[202] actions.

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

```
procedure TForm1.RVAControlPanel1MarginsChanged(Sender: TrvAction[202];
  Edit: TCustomRichViewEdit);
begin
  RVRuler1.UpdateRulerMargins;
end;
```

1.2.1.3.9 TRVAControlPanel.OnStyleNeeded

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

```
type
  TRVStyleNeededEvent = procedure (Sender: TrvAction[202]; 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**).

1.2.2 TRVAPopupMenu

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

Unit RichViewActions, RVAPopupActionBar;

Syntax, if TNT Controls[239] are not used:

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

Syntax, if TNT Controls[239] 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[239] 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[239] are used, from TTntPopupMenu.
- TRVAPopupActionBar is inherited from TPopupActionBar, available for Delphi 2006 or newer, defined in RVAPopupActionBar unit.
- TRVATBPopupMenu is available if Toolbar2000[240] is used.
- TRVATBXPopupMenu is available if TBX[240] is used.
- TRVASPTBXPopupMenu is available if SpTBXLib[239] is used.

Standard actions

This menu is automatically filled with RichViewActions from ActionList[45].

The following actions are always added (if available):

- TrvActionCut[80]
- TrvActionCopy[79]
- TrvActionPaste[82]
- TrvActionFontEx[91]
- TrvActionParagraph[124]
- TrvActionParaList[129]

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

- TrvActionItemProperties[208]

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

- TrvActionTableInsertRowsAbove[192]
- TrvActionTableInsertRowsBelow[193]
- TrvActionTableInsertColLeft[191]
- TrvActionTableInsertColRight[192]

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

- TrvActionTableDeleteRows[190]
- TrvActionTableDeleteCols[189]

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

- TrvActionFillColor[207]

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

- TrvActionTableMergeCells 193

Othewise, the following action is added:

- TrvActionTableSplitCells 200

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

- TrvActionTableProperties 194

Live spelling

This menu supports live spelling. If Addict 3 or 4 233 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 45
- OnLiveSpellGetSuggestions 46
- OnLiveSpellIgnoreAll 46

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.

1.2.2.1 Properties

1.2.2.1.1 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.

1.2.2.1.2 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 3 233 or OnLiveSpellGetSuggestions 46 event are ignored.

Default value:

0 (all suggestions are displayed)

1.2.2.2 Events

1.2.2.2.1 TRVAPopupMenu.OnLiveSpellAdd

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

```
type
  TRVALiveSpellAddEvent = procedure (Sender: TRVAPopupMenu 43;
    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[35] 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).

1.2.2.2.2 TRVAPopupMenu.OnLiveSpellGetSuggestions

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

```
type
  TRVALiveSpellGetSuggestionsEvent = procedure (Sender: TRVAPopupMenu[43];
    Edit: TCustomRichViewEdit; const Word: TRVUnicodeString; StyleNo: Integer;
    Suggestions: TRVASpellStrings[249]) of object;
  TRVA2LiveSpellGetSuggestionsEvent = procedure (Sender: TObject;
    Edit: TCustomRichViewEdit; const Word: TRVUnicodeString; StyleNo: Integer;
    Suggestions: TRVASpellStrings[249]) 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[35] is used, this command is processed automatically.

Otherwise, use this event for generating suggestions.

See also:

- MaxSuggestionsCount[45]

1.2.2.2.3 TRVAPopupMenu.OnLiveSpellIgnoreAll

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

```
type
  TRVALiveSpellIgnoreAllEvent = procedure (Sender: TRVAPopupMenu[43];
    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[35] 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).

1.2.2.2.4 TRVAPopupMenu.OnLiveSpellWordReplace

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

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

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.

1.2.3 TCustomRVFontComboBox

TCustomRVFontComboBox is the base class for font combo-boxes.

Unit RVFontCombos;

Syntax, if TNT Controls[239] are not used:

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

Syntax, if TNT Controls[239] are used:

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

Hierarchy

(Hierarchy of TCustomRVFontComboBox, if TNT Controls[239] 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[49]
- TRVFontSizeComboBox[50]
- TRVFontCharsetComboBox[48]

1.2.3.1 Properties

1.2.3.1.1 TCustomRVFontComboBox.ActionFont

Links this component to TrvActionFontEx[91] action.

```
property ActionFont: TrvActionFontEx[91];
```

This action is required to apply the selection to Editor[48].

1.2.3.1.2 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[48] must be defined as well)

1.2.4 TRVFontCharsetComboBox

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

Unit RVFontCombos;

```
TRVFontCharsetComboBox = class(TCustomRVFontComboBox[47])
```

Hierarchy

(Hierarchy of TRVFontCharsetComboBox, if TNT Controls[239] are not used)

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomComboBox
TComboBox
TCustomRVFontComboBox[47]

Description

This combo-box allows changing a font Charset for a selected text in Editor[48].

This component works automatically: place it on a form, assign Editor[48] and ActionFont[48] 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[49]=*True*.

See also:

- TRVFontComboBox[49]
- TRVFontSizeComboBox[50]

1.2.4.1 Properties

1.2.4.1.1 TRVFontCharsetComboBox.AddDefaultCharset

Specifies whether the combo-box must contain DEFAULT_CHARSET item.

```
property AddDefaultCharset: Boolean;
```

The caption for this item is defined in DefaultCharsetCaption[49] property.

Default value:

False

1.2.4.1.2 TRVFontCharsetComboBox.DefaultCharsetCaption

Specifies the caption for DEFAULT_CHARSET item.

```
property DefaultCharsetCaption: TRVALocString[224];
```

This item is added if AddDefaultCharset[49]=*True*.

The recommended value for this property: RVA_GetS[221](*rvam_font_DefaultCharset*).

Default value:

'(Default)'

1.2.4.1.3 TRVFontCharsetComboBox.FontName

Contains a font name.

```
property FontName: TRVALocString[224];
```

This property may be used when the combo-box is not linked with Editor[48]. If linked, the combo-box items are maintained automatically.

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

1.2.5 TRVFontComboBox

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

Unit RVFontCombos;

```
TRVFontComboBox = class(TCustomRVFontComboBox[47])
```

Hierarchy

(Hierarchy of TRVFontComboBox, if TNT Controls[239] are not used)

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomComboBox
TComboBox
TCustomRVFontComboBox[47]

Description

This combo-box allows changing a font name for a selected text in Editor[48].

This component works automatically: place it on a form, assign Editor[48] and ActionFont[48] properties; no additional code is required.

The component displays preview of fonts in items. To disable this feature, assign Preview[50] = *False*.

See also:

- TRVFontSizeComboBox 50
- TRVFontCharsetComboBox 48

1.2.5.1 Properties

1.2.5.1.1 TRVFontComboBox.DropDownWidth

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

```
property DropDownWidth: Integer;
```

This property is used if Preview 50 = *True*.

Default value:

300

1.2.5.1.2 TRVFontComboBox.Preview

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

```
property Preview: Boolean;
```

Default value:

True

See also:

- DropDownWidth 50
- ItemHeight
- SymbolPreviewString 50

1.2.5.1.3 TRVFontComboBox.SymbolPreviewString

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

```
property SymbolPreviewString: TRVALocString 224;
```

This property is used if Preview 50 = *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'

1.2.6 TRVFontSizeComboBox

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

Unit RVFontCombos;

```
TRVFontSizeComboBox = class(TCustomRVFontComboBox 47)
```

Hierarchy

(Hierarchy of TRVFontSizeComboBox, if TNT Controls 239 are not used)

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomComboBox
TComboBox
TCustomRVFontComboBox 47

Description

This combo-box allows changing a font size for a selected text in Editor[48].

This component works automatically: place it on a form, assign Editor[48] and ActionFont[48] properties; no additional code is required.

See also:

- TRVFontComboBox[49]
- TRVFontCharsetComboBox[48]

1.2.6.1 Properties

1.2.6.1.1 TRVFontSizeComboBox.FontName

Contains a font name.

```
property FontName: TFontName;
```

This property may be used when the combo-box is not linked with Editor[48]. 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).

1.2.7 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[239] are not used:

```
TRVStyleTemplateComboBox = class(TCustomComboBox)
TRVStyleTemplateListBox = class(TCustomListBox)
```

Syntax, if TNT Controls[239] are used:

```
TRVStyleTemplateComboBox = class(TTntCustomComboBox)
TRVStyleTemplateListBox = class(TTntCustomListBox)
```

Hierarchy

(Hierarchy of TRVStyleTemplateComboBox/TRVStyleTemplateListBox, if TNT Controls[239] are not used:)

TObject
TPersistent
TComponent
TControl
TWinControl
TCustomComboBox
TCustomComboBox/TCustomListBox

Description

These component allow applying style templates to Editor[52]. The components require Editor[52] .UseStyleTemplates=*True*, otherwise their Items are empty.

The components work automatically: place them on a form, assign Editor[48] property; no additional code is required, unless you want to process "All Styles" command in OnClickAllStyles[54] event.

By default, the components show only style templates in use, "quick access" style templates, and heading[53] style templates. You can display all style templates by assigning ShowAllStyles[53]=*True*.

If ShowClearFormat[53]=*True*, the components display "Clear Format" command.

You can display icons in items by assigning Images[53] and image indices[52].

When the application language is changed, call Localize[54].

1.2.7.1 Properties

1.2.7.1.1 TRVStyleTemplateComboBox.*ImageIndex

The properties contain indexes in Images[53].

```
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[53]=*True*.

AllStylesImageIndex – index of an image for "All Styles" item, shown if ShowAllStyles[53]=*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

1.2.7.1.2 TRVStyleTemplateComboBox.ControlPanel

Links this component with a control panel.

```
property ControlPanel: TComponent;
```

You can assign a TRVAControlPanel[26] component to this property. If this property is not assigned (*nil*), the component uses MainRVAControlPanel[252].

ControlPanel is used to display localized captions according to its Language[32].

1.2.7.1.3 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**

1.2.7.1.4 TRVStyleTemplateComboBox.Images

A link to TImageList component containing images for items.

```
property Images: TCustomImageList;
```

See also:

- AllStylesImageIndex, ClearFormatImageIndex, ParaStyleImageIndex, ParaTextStyleImageIndex, TextStyleImageIndex 52

1.2.7.1.5 TRVStyleTemplateComboBox.ShowAllStyles

Specifies whether the component displays all available style templates.

```
property ShowAllStyles: Boolean;
```

If *True*, the component displays all style templates from Editor 52.Style.StyleTemplates.

If *False*, the component displays:

- style templates in use (i.e. style templates referred by Editor 52.Style.TextStyles and ParaStyles);
- style templates having QuickAccess=*True*
- "All styles" item (if not all style templates are displayed)

If SmartHeadings 53=*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 54 is assigned, it is called
- otherwise, the combo-/list-box temporarily displays all style templates.

Default value:

False

See also:

- AllStylesImageIndex 52

1.2.7.1.6 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 52

1.2.7.1.7 TRVStyleTemplateComboBox.SmartHeadings

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

```
property SmartHeadings: Boolean;
```

If ShowAllStyles 53=*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

1.2.7.2 Methods

1.2.7.2.1 TRVStyleTemplateComboBox.Localize

Rebuilds the content of the component.

```
procedure Localize;
```

Call this method when Language [32] of ControlPanel [52] is changed.

1.2.7.3 Events

1.2.7.3.1 TRVStyleTemplateComboBox.OnClickAllStyles

Occurs when "All Styles" item is clicked.

```
property OnClickAllStyles: TNotifyEvent;
```

See ShowAllStyles [53] for details.

1.3 Actions

Actions

Actions in the order as they appear in the menu of the ActionTest demo:

File

- TrvActionNew [60]
- TrvActionOpen [63]
- TrvActionSave [73]
- TrvActionSaveAs [77]
- TrvActionExport [66]

- TrvActionPageSetup [69]
- TrvActionPrintPreview [71]
- TrvActionPrint [70]

Edit

- TrvActionUndo [88]
- TrvActionRedo [85]

- TrvActionCut [80]
- TrvActionCopy [79]
- TrvActionPaste [82]
- TrvActionPasteAsText [82]
- TrvActionPasteSpecial [83]

- TrvActionFind [80]
- TrvActionFindNext [81]
- TrvActionReplace [86]

- TrvActionCharCase [79]

- TrvActionInsertPageBreak [156]
- TrvActionRemovePageBreak [211]

TrvActionSelectAll 87

Font

TrvActionFonts 98
TrvActionFontEx 91

TrvActionFontBold 90
TrvActionFontItalic 97
TrvActionFontUnderline 103
TrvActionFontStrikeout 101

TrvActionFontAllCaps 89
TrvActionFontOverline 97

TrvActionSubscript 104
TrvActionSuperscript 104

TrvActionFontShrink 99
TrvActionFontGrow 95

TrvActionFontShrinkOnePoint 100
TrvActionFontGrowOnePoint 96

TrvActionFontColor 90
TrvActionFontBackColor 89

Paragraph

TrvActionParagraph 124
TrvActionParaBorder 120

TrvActionAlignLeft 110
TrvActionAlignCenter 108
TrvActionAlignRight 111
TrvActionAlignJustify 110
TrvActionAlignDistribute 109

TrvActionParaList 129
TrvActionParaBullets 122
TrvActionParaNumbering 132

TrvActionWordWrap 134

TrvActionIndentInc 117
TrvActionIndentDec 116

TrvActionLineSpacing100 118
TrvActionLineSpacing150 118

TrvActionLineSpacing200 119

TrvActionClearLeft 112
TrvActionClearRight 113
TrvActionClearBoth 112
TrvActionClearNone 113

TrvActionParaColor 122

Format

TrvActionStyleTemplates 134
TrvActionAddStyleTemplate 136
TrvActionClearFormat 138
TrvActionClearTextFormat 139
TrvActionStyleInspector 137

TrvActionBackground 202
TrvActionColor 204

TrvActionFillColor 207

TrvActionHide 208
TrvActionRemoveHyperlinks 211

TrvActionItemProperties 208
TrvActionInsertCaption 140
TrvActionVAlign 213

Insert

TrvActionInsertFile 143
TrvActionInsertPicture 157
TrvActionInsertHLine 144
TrvActionInsertHyperlink 145
TrvActionBookmarks 139
TrvActionInsertSymbol 160
TrvActionInsertEquation 142*
TrvActionInsertNumber 154
TrvActionInsertTextBox 169
TrvActionInsertPageNumber 157
TrvActionInsertPageCount 156

TrvActionInsertFootnote 166
TrvActionInsertEndnote 165
TrvActionInsertSidenote 168
TrvActionEditNote 164

Table

TrvActionInsertTable 170


TrvActionTableInsertColLeft 191

TrvActionTableInsertColRight 192


TrvActionTableInsertRowsAbove 192

TrvActionTableInsertRowsBelow 193


TrvActionTableDeleteCols 189

TrvActionTableDeleteRows 190

TrvActionTableDeleteTable 190

TrvActionTableToText 200


TrvActionTableSelectTable 198

TrvActionTableSelectCols 197

TrvActionTableSelectRows 197

TrvActionTableSelectCell 196


TrvActionTableCellVAlignTop 189

TrvActionTableCellVAlignMiddle 188

TrvActionTableCellVAlignBottom 187

TrvActionTableCellVAlignDefault 187


TrvActionTableCellRotationNone 183

TrvActionTableCellRotation90 184

TrvActionTableCellRotation180 185

TrvActionTableCellRotation270 185


TrvActionTableCellLeftBorder 181

TrvActionTableCellTopBorder 186

TrvActionTableCellRightBorder 182

TrvActionTableCellBottomBorder 180

TrvActionTableCellAllBorders 179

TrvActionTableCellNoBorders 182


TrvActionTableSplit 199

TrvActionTableSplitCells 200

TrvActionTableMergeCells 193


TrvActionTableGrid 191


TrvActionTableSort 199

TrvActionTableProperties 194

Toolbar only


TrvActionQuickPrint 72

TrvActionShowSpecialCharacters 212

Not used in ActionTest

TrvActionTextLTR 106

TrvActionTextRTL 106

TrvActionParaLTR 131
TrvActionParaRTL 132

TrvActionSpellingCheck 201
TrvActionThesaurus 201

Customized

TrvActionEvent 206
TrvActionInsertText 162

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

1.3.1 File

1.3.1.1 TrvActionCustomFileIO

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

Unit RichViewActions 54;

Syntax

```
TrvActionCustomFileIO = class(TrvActionCustomIO 59)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomIO 59

Description

This action is not used directly.

Descendants:

- TrvActionSaveAs 77
- TrvActionExport 66
- TrvActionInsertFile 143

1.3.1.1.1 Properties

1.3.1.1.1.1 TrvActionCustomFileIO.CustomFilter

Specifies the file filter for custom file formats.

```
property CustomFilter: TRVALocString 224;
```

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 78
- TRVActionExport.Filter 67
- TRVActionInsertFile.Filter 144
- TrvActionOpen.CustomFilter 64
- TRVAControlPanel.OnCustomFileOperation 40

1.3.1.2 TrvActionCustomIO

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

Unit RichViewActions 54;

Syntax

```
TrvActionCustomIO = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action is not used directly.

Direct descendants:

- TrvActionCustomFileIO 58
- TrvActionInsertPicture 157

1.3.1.2.1 Properties

1.3.1.2.1.1 TrvActionCustomIO.AutoUpdateFileName

Specifies whether FileName 60 is automatically updated by the action.

```
property AutoUpdateFileName: Boolean;
```

If this property is *True*, FileName 60 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 66
- TrvActionInsertFile 143

Default value:
True

1.3.1.2.1.2 TrvActionCustomIO.DialogTitle

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

```
property DialogTitle: TRVALocString 224;
```

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

Default value:

" (empty string)

See also:

- TrvActionOpen.DialogTitle 65

1.3.1.2.1.3 TrvActionCustomIO.FileName

Specifies the default file name.

```
property FileName: TRVALocString 224;
```

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 66
- TrvActionInsertFile 143
- TrvActionInsertPicture 157

Default value:
" (empty string)

See also properties:

- InitialDir 60
- AutoUpdateFileName 59

1.3.1.2.1.4 TrvActionCustomIO.InitialDir

Specifies the initial directory for the file dialog.

```
property InitialDir: TRVALocString 224;
```

This property is used only if FileName 60 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 54;

Syntax

```
TrvActionCustomNew = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction

TAction
TrvCustomAction 213
TrvAction 202

Description

This action is not used directly.

Descendants:

- TrvActionNew 61
- TrvActionOpen 63

1.3.1.3.1 Properties

1.3.1.3.1.1 TrvActionCustomNew .ActionSave

Links this action to TrvActionSave 73 action.

```
property ActionSave: TrvActionSave 73;
```

If this link is not defined, the first found TrvActionSave 73 action on the same form/datamodule is used. If no TrvActionSave 73 action found, an exception occurs on executing.

1.3.1.3.2 Events

1.3.1.3.2.1 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 60.

TrvActionOpen 63 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 64 or implicitly) with TrvActionNew. If linked, TrvActionOpen calls the event of TrvActionNew instead.

If style templates are not used 18, 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.

1.3.1.4 TrvActionNew

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

Unit RichViewActions 54;

Syntax

```
TrvActionNew = class(TrvActionCustomNew[60])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomNew[60]

Description

If you use this action, you must also use TrvActionSave[73] and TrvActionSaveAs[77] actions.

This action must be linked to TrvActionSave[73] action, because TrvActionSave[73] contains a list of all documents opened in TCustomRichViewEdit controls. This link may be defined explicitly (via ActionSave[61] property) or implicitly (this action finds and uses TrvActionSave[73] action placed on the same form/datamodule).

This action performs the following tasks, if style templates are not used[18]:

1. If the document was modified, asks for saving and saves it (using TrvActionSave[73]).
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[215]: DefaultColor[28], DefaultMargin[30], DefaultDocParameters[29].
4. Calls OnNew[61] event.
5. Calls OnMarginsChanged[43] event of GetControlPanel[215] component.
6. Formats the document (calls TCustomRichViewEdit.Format).
7. If GetControlPanel[215].AutoDeleteUnusedStyles[28]=*True*, deletes unused styles (calls TCustomRichViewEdit.DeleteUnusedStyles(True, True, True)).
8. Stores the following values for this TCustomRichViewEdit using the properties of GetControlPanel[215]: temporal file name = DefaultFileName[30], file format = DefaultFileFormat[30], custom filter index (used if file format is *ffeCustom*) = DefaultCustomFilterIndex[29]. These values are stored in the list of documents maintained by TrvActionSave[73].
9. Calls OnDocumentFileChange[76] event of TrvActionSave[73] action.

If style templates are used[18]:

The sequence of operations is the same, but the step #7 is different. The action resets text and paragraph styles according to StyleTemplates[62].

1.3.1.4.1 Properties

1.3.1.4.1.1 TrvActionNew .StyleTemplates

A collection of style templates for assigning to new documents.

```
property StyleTemplates: TRVStyleTemplateCollection;
```

This property is used:

- when TrvActionNew is executed
- when TrvActionOpen[63] (linked to this action) clears a target editor before loading a file.

Values in this collection are measured in GetControlPanel[215].UnitsProgram[35].

If style templates are used[18] (i.e. the target editor's UseStyleTemplates=*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 TSRichViewEdit, 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 215:

- DefaultColor 28
- DefaultMargin 30
- DefaultDocParameters 29

See also:

- TrvActionStyleTemplates 134.StandardStyleTemplates 135

1.3.1.4.2 Methods

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 62.

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 62 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 61, TrvActionSave 73, TrvActionSaveAs 77, TrvActionOpen 63) 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 54;

Syntax

```
TrvActionOpen = class(TrvActionCustomNew 60)
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomNew 60

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 73 action save content of this TCustomRichViewEdit component in this file in this format.

The following formats are supported:

- RichView Format (RVF)
- Rich Text Format (RTF)
- Text files (TXT, Unicode and ANSI)
- HTML (only if RvHtmlViewImporter 239 is used)
- RichViewXML Format (XML, only if RichViewXML 238 is used)
- custom formats, supported in GetControlPanel 215.OnCustomFileOperation 40 event.

If you use this action, you must also use TrvActionSave 73 and TrvActionSaveAs 77 actions.

This action must be linked to TrvActionSave 73 action, because TrvActionSave 73 contains a list of all documents opened in TCustomRichViewEdit controls. This link may be defined explicitly (via ActionSave 61 property) or implicitly (this action finds and uses TrvActionSave 73 action placed on the same form/datamodule).

This action may be linked to TrvActionNew 61 action. This link may be defined explicitly (via ActionNew 64 property) or implicitly (this action finds and uses TrvActionNew 61 action placed on the same form/datamodule). If linked, TrvActionNew 61 is used to reset editor before loading (it is especially important, if style templates are used 18).

This action performs the following tasks:

1. If the document was modified, asks for saving and saves it (using TrvActionSave 73).
2. Shows a file opening dialog allowing the user to choose the file for opening.
3. If the user chose a file, calls LoadFile 65 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

1.3.1.5.1.1 TrvActionOpen.ActionNew

Links this action to TrvActionNew 60 action.

```
property ActionNew: TrvActionNew 60;
```

If this link is not defined, the first found TrvActionNew 60 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 224;
```

This property allows opening files in custom formats (in addition to the standard formats listed in the Filter 65 property of this action).

Value of this property is appended to the Filter property of the file opening dialog, if the Filter 65 property includes *ffiCustom*.

Default value:

" (empty string)

See also:

- GetControlPanel 215.OnCustomFileOperation 40
- TrvActionCustomFileIO.CustomFilter 58

1.3.1.5.1.3 TrvActionOpen.DialogTitle

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

```
property DialogTitle: TRVALocString 224;
```

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

Default value:

" (empty string)

See also:

- TrvActionCustomIO.DialogTitle 59

1.3.1.5.1.4 TrvActionOpen.Filter

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

```
property Filter: TrvFileOpenFilterSet 251;
```

If *ffiCustom* is included, formats listed in the CustomFilter 64 property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel 215.RVFFilter 33.

Default value:

all formats

1.3.1.5.1.5 TrvActionOpen.InitialDir

Specifies the initial directory for the file dialog.

```
property InitialDir: TRVALocString 224;
```

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

1.3.1.5.2.1 TrvActionOpen.LoadFile

Loads the specified file in **rve**.

```
procedure LoadFile(rve: TCustomRichViewEdit; const FileName: TRVUnicodeString;
  FileFormat: TrvFileOpenFilter 251; 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[64] 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[215]: DefaultColor[28], DefaultMargin[30].
3. Calls OnNew[61] event.
4. Loads the file in **rve**.
5. Associates **rve** with this file and file format (in the list of documents maintained by TrvActionSave[73]).
6. Calls OnDocumentFileChange[76] event of TrvActionSave[73] action.
7. Calls OnMarginsChanged[43] event of GetControlPanel[215] component.
8. Formats the document (calls TCustomRichViewEdit.Format).
9. If GetControlPanel[215].AutoDeleteUnusedStyles[28]=*True*, deletes unused styles (calls TCustomRichViewEdit.DeleteUnusedStyles(True, True, True)).
10. Calls OnOpenFile[66] event.

Subsequent executions of TrvActionSave[73] will save **rve** content in this file in this format.

1.3.1.5.3 Events

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[251]; CustomFilterIndex: Integer) of object;

property OnOpenFile: TRVOpenFileEvent;
```

Parameters are the same as in LoadFile[65] method.

1.3.1.6 TrvActionExport

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

Unit RichViewActions[54];

Syntax

```
TrvActionExport = class(TrvActionCustomFileIO[58])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomIO[59]
TrvActionCustomFileIO[58]

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 77 action, this action supports:

- HTML (even if rvHtmlViewImporter 239 is not used)
- 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 60, TrvActionOpen 63 or TrvActionSaveAs 77 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 68 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

1.3.1.6.1.1 TrvActionExport.CreateDirectoryForHTMLImages

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

```
property CreateDirectoryForHTMLImages: Boolean;
```

This property is used when exporting to HTML formats (*ffeHTMLCSS or ffeHTML* 250).

If *True*, images from 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 *rvsoUseItemImageFileNames* is not included in SaveOptions 68 property.

Default value:

True

See also:

- TrvActionSave.CreateDirectoryForHTMLImages 74

1.3.1.6.1.2 TrvActionExport.FileTitle

Defines title for HTML files.

```
property FileTitle: TRVUnicodeString;
```

This property is used when exporting to HTML formats (*ffeHTMLCSS or ffeHTML* 250). This property is used as the Title parameter of TCustomRichViewEdit.SaveHTML and SaveHTMLEx methods.

See also:

- TrvActionSave.FileTitle 75

1.3.1.6.1.3 TrvActionExport.Filter

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

```
property Filter: TrvFileExportFilterSet 250;
```

If *ffeCustom* is included, formats listed in the CustomFilter 58 property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel 215.RVFFilter 33.

Default value:

all formats

1.3.1.6.1.4 TrvActionExport.ImagePrefix

Prefix of image files saved together with HTML file.

```
property ImagePrefix: TRVUnicodeString;
```

This property is used when exporting to HTML formats (*ffeHTMLCSS or ffeHTML* 250). This property (probably with subdirectory name) is used as the ImagesPrefix parameter of TCustomRichViewEdit.SaveHTML and SaveHTMLEx methods.

This property is used for images with undefined (empty) file names, or when *rvsoUseItemImageFileNames* is not included in SaveOptions 68 property.

See also properties:

- CreateDirectoryForHTMLImages 67

See also:

- TrvActionSave.ImagePrefix 75

1.3.1.6.1.5 TrvActionExport.SaveOptions

Defines options for saving HTML files.

```
property SaveOptions: TRVSaveOptions;
```

This property is used when exporting to HTML formats (*ffeHTMLCSS or ffeHTML* 250). This property is used as the SaveOptions parameter of TCustomRichViewEdit.SaveHTML and SaveHTMLEx methods.

Default value:

[]

See also:

- TrvActionSave.SaveOptions 75

1.3.1.6.2 Methods

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: TrvFileExportFilter; 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 CustomFilter 58 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 RichViewActions[54];

Syntax

```
TrvActionPageSetup = class(TrvCustomAction[213])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]

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 GetControlPanel[215].RVPrint[34] 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 GetControlPanel[215].RVPrint[34] properties;
- headers and footers are stored in GetControlPanel[215].Header and Footer[31] 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

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;
```

Internally, millimeters are used.

Default value:

rvpmuMillimeters

1.3.1.7.2 Events

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[54];

Syntax

```
TrvActionPrint = class(TrvCustomPrintAction[78])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvCustomPrintAction[78]

Description

This action displays a printing dialog (TPrintDialog) and prints the document from TCustomRichViewEdit component.

This action uses settings defined in TrvActionPageSetup[69]. If GetControlPanel[215].RVPrint[34] 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:

- TrvActionQuickPrint[72]
- TrvActionPrintPreview[71]

1.3.1.8.1 Properties

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[73]

1.3.1.9 TrvActionPrintPreview

TrvActionPrintPreview is the action for "File | Print Preview" command.

Unit RichViewActions [54];

Syntax

```
TrvActionPrintPreview = class(TrvCustomPrintAction [78])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction [213]
TrvAction [202]
TrvCustomPrintAction [78]

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 [69] action is used. If it is not linked (via ActionPageSetup [71] property), a temporal TrvActionPageSetup [69] action is used. For the "Print" button, TrvActionPrint [70] action is used. If it is not linked (via ActionPrint [71] property), the first TrvActionPrint [70] 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 [69]. If GetControlPanel [215].RVPrint [34] 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 [72]
- TrvActionPrint [70]

1.3.1.9.1 Properties

1.3.1.9.1.1 TrvActionPrintPreview .ActionPageSetup

A link to TrvActionPageSetup [69] action.

```
property ActionPageSetup: TrvActionPageSetup [69];
```

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 [70] action.

```
property ActionPrint: TrvActionPrint [70];
```

This link is used for the "Print" button on the print preview dialog.

If this link is not assigned, the first found TrvActionPrint [70] 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

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 54;

Syntax

```
TrvActionQuickPrint = class(TrvCustomPrintAction 78)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomPrintAction 78

Description

This action prints the document from TCustomRichViewEdit component without displaying a printing dialog.

This action uses settings defined in TrvActionPageSetup 69 action. If GetControlPanel 215.RVPrint 34 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 70
- TrvActionPrintPreview 71

1.3.1.10.1 Properties

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 70

1.3.1.11 TrvActionSave

TrvActionSave is the action for "File | Save" command.

Unit RichViewActions 54;

Syntax

```
TrvActionSave = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action saves the document from TCustomRichViewEdit to a file. In addition, this action contains list of all documents (TCustomRichViewEdits associated with their file names and formats).

If the document has a temporal name (assigned after executing TrvActionNew 60 action), this action executes TrvActionSaveAs 77 action. So if you use this action, you must also use TrvActionSaveAs 77 action. The link between the actions may be defined explicitly (via ActionSaveAs 74 property) or implicitly (this action finds and uses TrvActionSaveAs 77 action placed on the same form/datamodule).

If the document has a permanent name and format (assigned after executing TrvActionOpen 63 or TrvActionSaveAs 77 action), the action just saves the file.

If the file cannot be saved, an error message is displayed.

CanCloseDoc 76 method should be used in Form.OnCloseQuery event.

1.3.1.11.1 Properties

1.3.1.11.1.1 TrvActionSave.ActionSaveAs

Links this action to TrvActionSaveAs (77) action.

```
property ActionSaveAs: TrvActionSaveAs (77);
```

TrvActionSaveAs (77) action is used when the document has a temporal name (after the execution of TrvActionNew (60) action). In this case, instead of saving, the action executes TrvActionSaveAs (77) action.

If this link is not defined, the first found TrvActionSaveAs (77) action on the same form/datamodule is used. If no TrvActionSaveAs (77) action found, an exception occurs on executing.

1.3.1.11.1.2 TrvActionSave.CreateDirectoryForHTMLImages

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

```
property CreateDirectoryForHTMLImages: Boolean;
```

This property is defined only if RvHTMLViewImporter (239) is used.

This property is used when saving to HTML format (*ffeHTMLCSS* (250)).

If *True*, images from 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 *rvsoUseItemImageFileNames* is not included in SaveOptions (75) property.

Default value:

True

See also:

TrvActionExport.CreateDirectoryForHTMLImages (67)

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 (250) – 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 (58) property of the linked TrvActionSaveAs (77) action.
- Defined – *False* if FileName and FileFormat contain temporal values assigned for new document.

This list is maintained automatically.

See also methods:

- FindDoc 76

1.3.1.11.1.5 TrvActionSave.FileTitle

Defines title for HTML files.

```
property FileTitle: TRVUnicodeString;
```

This property is defined only if RvHTMLViewImporter 239 is used.

This property is used when saving to HTML format (*ffeHTMLCSS* 250). This property is used as the Title parameter of TCustomRichViewEdit.SaveHTMLEx method.

See also:

- TrvActionExport.FileTitle 67

1.3.1.11.1.6 TrvActionSave.ImagePrefix

Prefix of image files saved together with HTML file.

```
property ImagePrefix: TRVUnicodeString;
```

This property is defined only if RvHTMLViewImporter 239 is used.

This property is used when saving to HTML format (*ffeHTMLCSS* 250). This property (probably with subdirectory name) is used as the ImagesPrefix parameter of TCustomRichViewEdit.SaveHTMLEx method.

This property is used for images with undefined (empty) file names, or when *rvsoUseItemImageFileNames* is not included in SaveOptions 75 property.

See also properties:

- CreateDirectoryForHTMLImages 74

See also:

- TrvActionExport.ImagePrefix 68

1.3.1.11.1.7 TrvActionSave.LostFormatWarning

Lists "lossy" file formats.

```
property LostFormatWarning: TrvFileSaveFilterSet 250;
```

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)

1.3.1.11.1.8 TrvActionSave.SaveOptions

Defines options for saving HTML files.

```
property SaveOptions: TRVSaveOptions;
```

This property is defined only if RvHTMLViewImporter 239 is used.

This property is used when saving to HTML format (*ffeHTMLCSS* 250). This property is used as the SaveOptions parameter of TCustomRichViewEdit.SaveHTMLEx method.

Default value:

[]

See also:

- TrvActionExport.SaveOptions 68

1.3.1.11.1.9 TrvActionSave.SuppressNextErrorMessage

Turns off the default error message when saving to the custom format (*ffeCustom* 250).

```
property SuppressNextErrorMessage: Boolean;
```

This property is set to *False* before calling GetControlPanel 215.OnCustomFileOperation 40.

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

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 74 for the given editor **rve**. Returns -1 if not found.

```
function FindDoc(rve: TCustomRichViewEdit): Integer;
```

1.3.1.11.3 Events

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 250; IsNew: Boolean) of object;

property OnDocumentFileChange: TRVFileChangeEvent;
```

This event occurs in the following cases:

- TrvActionNew 60 action is executed;
- TrvActionOpen 63 action is executed, or TrvActionOpen.LoadFile 65 is called;
- TrvActionSaveAs 77 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[60] action.

If **FileFormat**=*ffeCustom*, this event does not allow to identify the specific custom format. You can get it using Documents[74] property.

1.3.1.11.3.2 TrvActionSave.OnSaving

Occurs before saving.

```
type
  TRVSaveFileEvent = procedure (Sender: TObject;
    Editor: TCustomRichViewEdit; const FileName: TRVUnicodeString;
    FileFormat: TrvFileSaveFilter[250]; CustomFilterIndex: Integer) of object;

property OnSaving: TRVSaveFileEvent;
```

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[58] property of the linked TrvActionSaveAs[77] action.

1.3.1.12 TrvActionSaveAs

TrvActionSaveAs is the action for "File | Save As..." command.

Unit RichViewActions[54];

Syntax

```
TrvActionSaveAs = class(TrvActionCustomFileIO[58])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomIO[59]
TrvActionCustomFileIO[58]

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[73] action save its content in this file in this format.

The following formats are supported:

- RichView Format (RVF)
- Rich Text Format (RTF)
- Text files (TXT, Unicode and ANSI)
- HTML (only if RvHtmlViewImporter[239] is used)
- RichViewXML Format (XML, only if RichViewXML[238] is used)

- custom formats, supported in GetControlPanel[215].OnCustomFileOperation[40] event.

If you use this action, you must also use TrvActionSave[73] action.

This action must be linked to TrvActionSave[73] action, because TrvActionSave[73] contains a list of all documents opened in TCustomRichViewEdit controls. Besides, this action uses TrvActionSave[73] to save the file when file name and format are chosen. This link may be defined explicitly (via ActionSave[78] property) or implicitly (this action finds and uses TrvActionSave[73] 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[75] of TrvActionSave[73], displays a warning about lossy saving.
3. Saves the file, associates TCustomRichViewEdit control with this file name and file format.
4. Calls OnDocumentFileChange[76] event of TrvActionSave[73] action.

1.3.1.12.1 Properties

1.3.1.12.1.1 TrvActionSaveAs.ActionSave

Links this action to TrvActionSave[73] action.

```
property ActionSave: TrvActionSave[73];
```

If this link is not defined, the first found TrvActionSave[73] action on the same form/datamodule is used. If no TrvActionSave[73] 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[250];
```

If *ffeCustom* is included, formats listed in the CustomFilter[58] property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel[215].RVFFilter[33].

Default value:

all formats

1.3.1.13 TrvCustomPrintAction

TrvCustomPrintAction is a base class for the printing actions.

Unit RichViewActions[54];

Syntax

```
TrvCustomPrintActions = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly, this is the base class for the following actions:

- TrvActionPrintPreview 71
- TrvActionQuickPrint 72
- TrvActionPrint 70

All these actions are enabled only if at least one printer installed in the system (Printer.Printers.Count>0).

1.3.2 Edit

1.3.2.1 TrvActionCharCase

TrvActionCharCase is the action for "Character Case" command.

Unit RichViewActions 54;

Syntax

```
TrvActionCharCase = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 of TrvAction.

Limitation: this action cannot be applied to multicell selection in table.

See also:

- RVChangeCharCase, RVGetCharCase 248 procedures.
- TrvActionFontAllCaps 89

1.3.2.2 TrvActionCopy

TrvActionCopy is the action for "Edit | Copy" command.

Unit RichViewActions 54;

Syntax

```
TrvActionCopy = class(TrvCustomEditAction 88)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomEditAction 88

Description

This action copies the selection to the Clipboard (calls TCustomRichViewEdit.CopyDef).

This action does not introduce any new properties in addition to properties of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction 88 topic.

1.3.2.3 TrvActionCut

TrvActionCut is the action for "Edit | Cut" command.

Unit RichViewActions 54;

Syntax

```
TrvActionCut = class(TrvCustomEditAction 88)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomEditAction 88

Description

This action cuts the selection to the Clipboard (calls TCustomRichViewEdit.CutDef).

This action does not introduce any new properties in addition to properties of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction 88 topic.

1.3.2.4 TrvActionFind

TrvActionFind is the action for "Edit | Find" command.

Unit RichViewActions 54;

Syntax

```
TrvActionFind = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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[215].SearchScope[34] 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[239] are used, the action searches for Unicode string. Otherwise, it searches for ANSI string.

See also:

TrvActionFindNext[81]
TrvActionReplace[86]

1.3.2.4.1 Properties

1.3.2.4.1.1 TrvActionFind.ActionReplace

Links this action to TrvActionReplace[86] action.

```
property ActionReplace: TrvActionReplace[86];
```

If this link is not defined, the first TrvActionReplace[86] action found in the same form/datamodule is used.

When this action is executed, it calls TrvActionReplace.CloseDialog[87].

1.3.2.4.2 Methods

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[54];

Syntax

```
TrvActionFindNext = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action continues the search started by TrvActionFind[80] action. The link between the actions may be defined explicitly (via ActionFind[82] property) or implicitly (this action finds and uses TrvActionFind[80] action placed on the same form/datamodule).

1.3.2.5.1 Properties

1.3.2.5.1.1 TrvActionFindNext.ActionFind

Links this action to TrvActionFind[80] action.

```
property ActionFind: TrvActionFind[80];
```

If this link is not defined, the first TrvActionFind[80] 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[80] action. If search was not started, the action executes TrvActionFind[80] action instead.

1.3.2.6 TrvActionPaste

TrvActionPaste is the action for "Edit | Paste" command.

Unit RichViewActions[54];

Syntax

```
TrvActionPaste = class(TrvCustomEditAction[88])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvCustomEditAction[88]

Description

This action pastes data from the Clipboard (calls TCustomRichViewEdit.Paste).

If RvHtmlViewImporter[239] or rvHtmlImporter[238] is used, the action can paste also in HTML format. This format is pasted only if:

- the Clipboard does not contain data in RVF format, and
- the Clipboard does not contain data in RTF format (or it contains empty RTF data; it happens when MS Internet Explorer copies a complex fragment with nested tables).

This action does not introduce any new properties in addition to properties of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction[88] topic.

1.3.2.7 TrvActionPasteAsText

TrvActionPasteAsText is the action for "Edit | Paste as Text" command.

Unit RichViewActions[54];

Syntax

```
TrvActionPasteAsText = class(TrvCustomEditAction[88])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction

TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomEditAction 88

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 of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction 88 topic.

1.3.2.8 TrvActionPasteSpecial

TrvActionPasteSpecial is the action for "Edit | Paste Special" command.

Unit RichViewActions 54;

Syntax

```
TrvActionPasteSpecial = class(TrvActionPaste 82)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomEditAction 88
TrvActionPaste 82

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 (CF_TEXT)
- Unicode text (CF_UNICODETEXT)
- Windows bitmap (CF_BITMAP)
- Windows metafile (CF_METAFILEPICT)
- URL ("UniformResourceLocator")
- RTF ("Rich Text Format")
- RVF ("RichView Format")
- HTML ("HTML Format") (only if RvHtmlViewImporter 239 or RvHtmlImporter 238 is used)
- Graphic files (CF_HDROP)
- custom formats (see OnShowing 85 and OnCustomPaste 85 events).

The action displays only formats listed in in TCustomRichViewEdit.AcceptPasteFormats. **Note:** HTML and URL formats are not included in AcceptPasteFormats by default.

If GetControlPanel 215.RVFLocalizable 33=*True*, GetControlPanel 215.RVFFormatTitle 33 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 84 = 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 88, this action cannot work with editor types other than TCustomRichViewEdit and TSRichViewEdit.

1.3.2.8.1 Properties

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 *rvespImageFileName* 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 160
- TrvActionItemProperties 208 (does not support storing paths in item names)
- RTF reading (does not support storing paths in item names)
- TRvHtmlViewImporter 239.StoreFileNameInItemName
- TRvHtmlImporter 238.Options (*rvhtmloImgSrcToImageName* option)

1.3.2.8.2 Methods

1.3.2.8.2.1 TrvActionPasteSpecial.AddFormat

Adds a new format in the paste-special dialog.

```
procedure AddFormat(const FormatName: TRVALocString 224; Format: Word);
```

This method must be called only from OnShowing 85 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 85 event will occur.

1.3.2.8.3 Events

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 85
- OnCustomPaste 85

1.3.2.8.3.2 TrvActionPasteSpecial.OnCustomPaste

Occurs when the user chose a Clipboard format added with AddFormat 84 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 84 method to add additional formats in the dialog.

1.3.2.9 TrvActionRedo

TrvActionRedo is the action for "Edit | Redo" command.

Unit RichViewActions 54;

Syntax

```
TrvActionRedo = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action redoes the last undone editing operation (calls TCustomRichViewEdit.Redo).

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.2.10 TrvActionReplace

TrvActionReplace is the action for "Edit | Replace" command

Unit RichViewActions 54;

Syntax

```
TrvActionReplace = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 215.SearchScope 34 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 239 are used, the action searches for Unicode string and replaces it with Unicode string. Otherwise, it works with ANSI strings.

See also:

TrvActionFindNext 81
TrvActionFind 80

1.3.2.10.1 Properties

1.3.2.10.1.1 TrvActionReplace.ActionFind

Links this action to TrvActionFind 80 action.

```
property ActionFind: TrvActionFind 80;
```

If this link is not defined, the first TrvActionFind 80 action found in the same form/datamodule is used.

When this action is executed, it calls TrvActionFind.CloseDialog 81.

1.3.2.10.1.2 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

1.3.2.10.2.1 TrvActionReplace.CloseDialog

Closes the replace dialog (if it is shown).

```
procedure CloseDialog;
```

1.3.2.10.3 Events

1.3.2.10.3.1 TrvActionReplace.OnReplaceAllEnd

Occurs when "replace all" operation finishes.

```
property OnReplaceAllEnd: TRVAEditEvent[249];
```

If GetControlPanel[215].SearchScope[34] 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[249];
```

If GetControlPanel[215].SearchScope[34] 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[54];

Syntax

```
TrvActionSelectAll = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action selects the whole document (calls TCustomRichViewEdit.SelectAll).

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.2.12 TrvActionUndo

TrvActionUndo is the action for "Edit | Undo" command.

Unit RichViewActions 54;

Syntax

```
TrvActionUndo = class(TrvCustomEditAction 88)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvCustomEditAction 88

Description

This action undoes the last editing operation (calls TCustomRichViewEdit.Undo).

This action does not introduce any new properties in addition to properties of TrvAction.

This action can work with several types of editors, see the comments in the TrvCustomEditAction 88 topic.

1.3.2.13 TrvCustomEditAction

TrvCustomEditAction is a base class for several editing actions.

Unit RichViewActions 54;

Syntax

```
TrvCustomEditAction = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionUndo 88
- TrvActionCut 80
- TrvActionCopy 79
- TrvActionPaste 82

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[253] variable.

See also RVA_EditForceDefControl[252] global variable.

1.3.3 Font

1.3.3.1 TrvActionFontAllCaps

TrvActionFontAllCaps is the action for "All Capitals" command.

Unit RichViewActions[54];

Syntax

```
TrvActionFontAllCaps = class(TrvActionFontStyleEx[102])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionTextStyles[107]
TrvActionFontStyleEx[102]

Description

This action does not introduce any new properties in addition to properties 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[79]

1.3.3.2 TrvActionFontBackColor

TrvActionFontBackColor is the action for changing background color of the selected text.

Unit RichViewActions[54];

Syntax

```
TrvActionFontBackColor = class(TrvActionFontCustomColor[91])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction

TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomColor 204
TrvActionFontCustomColor 91

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor 204.

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 204.

1.3.3.3 TrvActionFontBold

TrvActionFontBold is the action for "Bold" command.

Unit RichViewActions 54;

Syntax

```
TrvActionFontBold = class(TrvActionFontStyle 101)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionFontStyle 101

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionFontColor = class(TrvActionFontCustomColor 91)
```

Hierarchy

Hierarchy
TObject

TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomColor 204
TrvActionFontCustomColor 91

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor 204. However, it changes the default value of Color 205 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 204.

1.3.3.5 TrvActionFontCustomColor

TrvActionFontCustomColor is a base class for the actions changing color properties of the selected text.

Unit RichViewActions 54;

Syntax

```
TrvActionFontCustomColor = class(TrvActionCustomColor 204)
```

Hierarchy

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomColor 204

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor 204.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontColor 90
- TrvActionFontBackColor 89

1.3.3.6 TrvActionFontEx

TrvActionFontEx is the action for "Font" command. It uses an advanced font dialog.

Unit RichViewActions 54;

Syntax

```
TrvActionFontEx = class(TrvActionFonts 98)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionFonts 98

Description

If UserInterface 99=*False*, the action applies its properties to the selected text. The list of properties to apply is specified in ValidProperties 95.

If UserInterface 99=*True*, the action displays a font dialog (initialized with font properties of the selected text), assigns the chosen font to the action's properties, and then applies these 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 99 := False;
      ValidProperties 95 := [rvfimFontName, rvfimSize, rvfimBold, rvfimUnderline, rvfimUnder
      Font 98.Name := FontName;
      Font 98.Size := 12;
      Font 98.Style := [fsUnderline];
      UnderlineType 94 := rvutDotted;
      ExecuteTarget(RichViewEdit1);
      UserInterface 99 := True;
    end;
```

See also:

- TrvActionFonts 98

1.3.3.6.1 Properties

1.3.3.6.1.1 TrvActionFontEx.AutoApplySymbolCharset

Allows to correct charset for symbol fonts.

```
property AutoApplySymbolCharset: Boolean;
```

This property works if UserInterface 99=*False*.

If *True*, then when the action applies a font name without charset (i.e *rvfimFontName* in ValidProperties 95, *rvfimCharset* not in ValidProperties 95), if this is a symbol font (i.e. if it supports only SYMBOL_CHARSET), the action applies SYMBOL_CHARSET to the selection.

Default value:

True

1.3.3.6.1.2 TrvActionFontEx.BackColor

Specifies the background color for applying to the selected text.

```
property BackColor: TColor;
```

If UserInterface 99=*False* and *rvfimBackColor* in ValidProperties 95, the action applies this property to the selected text.

If UserInterface[99]=*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, *rvfimBackColor* is included in ValidProperties[95], 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[99]=*False* and *rvfimCharScale* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False* and *rvfimCharSpacing* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False* and *rvfimSize* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], and the action applies this property to the selected text.

This property is applied to the text style's (RVStyle.TextStyles[]) property SizeDouble.

This property overrides Font[98].Size.

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[99]=*False* and *rvfimOverline* or *rvfimAllCaps* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False* and *rvfimSubSuperScriptType* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False* and *rvfimUnderlineColor* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False* and *rvfimUnderlineType* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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[99]=*False,* assign a subset of properties (for applying) to ValidProperties.

If UserInterface[99]=*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[98].Style and FontStyleEx[94]) each value in the set is represented by one value in this property. For example:

- if *rvfimOverline* in ValidProperties, *rvfsOverline* in FontStyleEx[94], the selected text will be overlined;
- if *rvfimOverline* in ValidProperties, *rvfsOverline* not in FontStyleEx[94], 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[99]=*False* and *rvfimVShift* in ValidProperties[95], the action applies this property to the selected text.

If UserInterface[99]=*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[95], 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.

1.3.3.7 TrvActionFontGrow

TrvActionFontGrow is the action for "Grow Font" command.

Unit RichViewActions[54];

Syntax

```
TrvActionFontGrow = class(TrvActionFontShrinkGrow[99])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]

TrvAction 202
TrvActionTextStyles 107
TrvActionFontShrinkGrow 99

Description

The action increases font size of the selected text by Percent 100 (until it reaches MaxSize 96).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow 99.

1.3.3.7.1 Properties

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 54;

Syntax

```
TrvActionFontGrowOnePoint = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

The action increases font size of the selected text by 1 (until it reaches MaxSize 96).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow 99.

1.3.3.8.1 Properties

1.3.3.8.1.1 TrvActionFontGrow OnePoint.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 54;

Syntax

```
TrvActionFontItalic = class(TrvActionFontStyle 101)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionFontStyle 101

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionFontOverline = class(TrvActionFontStyleEx 102)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionFontStyleEx 102

Description

This action does not introduce any new properties in addition to properties 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[54];

Syntax

```
TrvActionFonts = class(TrvActionTextStyles[107])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionTextStyles[107]

Description

If UserInterface[99]=*False*, the action applies Font[98] to the selected text.

If UserInterface[99]=*True*, the action displays a TFontDialog (initialized with font properties of the current text style), assigns the chosen font to Font[98], and then applies Font[98] to the selected text.

It's recommended to use TrvActionFontEx[91] 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[91], 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[91] allows defining much more text properties.

1.3.3.11.1 Properties

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[99]=*False*, the action applies this property to the selected text. For TrvActionFontEx[91], only properties specified in ValidProperties[95] are applied (a subset of *rvfimFontName*, *rvfimSize*, *rvfimCharset*, *rvfimBold*, *rvfimItalic*, *rvfimUnderline*, *rvfimStrikeout*, *rvfimColor*).

If UserInterface[99]=*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[91], it is overridden by FontSizeDouble[93])
- 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 [54];

Syntax

```
TrvActionFontShrink = class(TrvActionFontShrinkGrow [99])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction [213]
TrvAction [202]
TrvActionTextStyles [107]
TrvActionFontShrinkGrow [99]

Description

The action decreases font size of the selected text by Percent [100] (until it reaches MinSize [99]).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow [99].

1.3.3.12.1 Properties

1.3.3.12.1.1 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 [54];

Syntax

```
TrvActionFontShrinkGrow = class(TrvActionTextStyles [107])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

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 99
- TrvActionFontGrow 95

See also:

- TrvActionFontShrinkOnePoint 100
- TrvActionFontGrowOnePoint 96

1.3.3.13.1 Properties

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 54;

Syntax

```
TrvActionFontShrinkOnePoint = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

The action decreases font size of the selected text by 1 (until it reaches MinSize 101).

See the list of font size changing actions in the topic about TrvActionFontShrinkGrow 99.

1.3.3.14.1 Properties

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[54];

Syntax

```
TrvActionFontStrikeout = class(TrvActionFontStyle[101])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionTextStyles[107]
TrvActionFontStyle[101]

Description

This action does not introduce any new properties in addition to properties 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[54];

Syntax

```
TrvActionFontStyle = class(TrvActionTextStyles[107])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontBold 90
- TrvActionFontItalic 97
- TrvActionFontUnderline 103
- TrvActionFontStrikeout 101

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 54;

Syntax

```
TrvActionFontStyleEx = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionFontAllCaps 89
- TrvActionFontOverline 97

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 RichViewActions 54;

Syntax

```
TrvActionFontUnderline = class(TrvActionFontStyle 101)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionFontStyle 101

Description

This action does not introduce any new properties in addition to properties 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 RichViewActions 54;

Syntax

```
TrvActionTextStyles = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

This action does not introduce any new properties in addition to properties 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:

- TrvActionSubscript 104
- TrvActionSuperscript 104

1.3.3.20 TrvActionSubscript

TrvActionSubscript is the action for "Subscript" command.

Unit RichViewActions 54;

Syntax

```
TrvActionSubscript = class(TrvActionSSScript 103)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionSSScript 103

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionSuperscript = class(TrvActionSSScript 103)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction

TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionSSScript 103

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionTextBiDi = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

Description

This action does not introduce any new properties in addition to properties 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 106
- TrvActionTextRTL 106

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 119
- TrvActionParaLTR 131
- TrvActionParaRTL 132

1.3.3.23 TrvActionTextLTR

TrvActionTextLTR is the action for "Left to Right Text" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTextLTR = class(TrvActionTextBiDi 105)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionTextBiDi 105

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 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 106
- TrvActionParaLTR 131
- TrvActionParaRTL 132

1.3.3.24 TrvActionTextRTL

TrvActionTextRTL is the action for "Right to Left Text" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTextRTL = class(TrvActionTextBiDi 105)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107
TrvActionTextBiDi 105

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 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 106
- TrvActionParaLTR 131
- TrvActionParaRTL 132

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 145.

Unit RichViewActions 54;

Syntax

```
TrvActionTextStyles = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionFonts 98
- TrvActionFontStyle 101
- TrvActionFontStyleEx 102
- TrvActionSSScript 103
- TrvActionTextBiDi 105
- TrvActionFontShrinkGrow 99
- TrvActionFontShrinkOnePoint 100
- TrvActionFontGrowOnePoint 96
- TrvActionInsertHyperlink 145

1.3.4 Paragraph

1.3.4.1 TrvActionAlignCenter

TrvActionAlignCenter is the action for "Align Center" command.

Unit RichViewActions 54;

Syntax

```
TrvActionAlignCenter = class(TrvActionAlignment 111)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionAlignment 111

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionAlignCustomJustify = class(TrvActionAlignment 111)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionAlignment 111

Description

Properties:

LastLineAlignment[109] is applied to the selected paragraphs, if UseLastLineAlignment[109]=*True*.

This action is not used directly. The following actions are inherited from it:

- TrvActionAlignLeft[110]
- TrvActionAlignRight[111]
- TrvActionAlignCenter[108]
- TrvActionAlignJustify[110]

1.3.4.2.1 Properties

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[110]: *rvllaDefault*
  - in TrvActionAlignDistribute: *rvllaJustify*

1.3.4.3 TrvActionAlignDistribute

TrvActionAlignDistribute is the action for "Distribute" command.

Unit RichViewActions[54];

Syntax

```
TrvActionAlignDistribute = class(TrvActionAlignCustomJustify[108])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionParaStyles[133]
TrvActionAlignment[111]
TrvActionAlignCustomJustify[108]

Description

This action does not introduce any new properties in addition to properties 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[109].

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 54;

Syntax

```
TrvActionAlignJustify = class(TrvActionAlignCustomJustify 108)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionAlignment 111
TrvActionAlignCustomJustify 108

Description

This action does not introduce any new properties in addition to properties 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 109.

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 54;

Syntax

```
TrvActionAlignLeft = class(TrvActionAlignment 111)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionAlignment 111

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionAlignment = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionAlignLeft 110
- TrvActionAlignRight 111
- TrvActionAlignCenter 108
- TrvActionAlignJustify 110
- TrvActionAlignDistribute 109

1.3.4.7 TrvActionAlignRight

TrvActionAlignRight is the action for "Align Right" command.

Unit RichViewActions 54;

Syntax

```
TrvActionAlignRight = class(TrvActionAlignment 111)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionAlignment 111

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionClearBoth = class(TrvActionClearTextFlow 114)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionClearTextFlow 114

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 of TrvAction.

1.3.4.9 TrvActionClearLeft

TrvActionClearLeft is the action for "Clear Text Flow at Left Side"

Unit RichViewActions 54;

Syntax

```
TrvActionClearLeft = class(TrvActionClearTextFlow 114)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionClearTextFlow 114

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 of TrvAction.

1.3.4.10 TrvActionClearNone

TrvActionClearNone is the action for "Normal Text Flow"

Unit RichViewActions 54;

Syntax

```
TrvActionClearNone = class(TrvActionClearTextFlow 114)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionClearTextFlow 114

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 of TrvAction.

1.3.4.11 TrvActionClearRight

TrvActionClearRight is the action for "Clear Text Flow at Right Side"

Unit RichViewActions 54;

Syntax

```
TrvActionClearRight = class(TrvActionClearTextFlow 114)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionClearTextFlow 114

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 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[54];

Syntax

```
TrvActionClearTextFlow = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionClearLeft[112]
- TrvActionClearRight[113]
- TrvActionClearBoth[112]
- TrvActionClearNone[113]

1.3.4.13 TrvActionCustomParaListSwitcher

TrvActionCustomParaListSwitcher is a base class for the actions toggling paragraphs' bullets or numbering on/off.

Unit RichViewActions[54];

Syntax

```
TrvActionCustomParaListSwitcher = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action is checked when the current paragraph has a list marker with properties identical to the properties of ListLevels[115]. 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 ListLevels[115] 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:

- TrvActionParaBullets 122
- TrvActionParaNumbering 132

1.3.4.13.1 Properties

1.3.4.13.1.1 TrvActionCustomParaListSwitcher.IndentStep

Defines the step for increasing LeftIndent and MarkerIndent of ListLevels 115.

```
property IndentStep: TRVStyleLength;
```

This value is measured in GetControlPanel 215.UnitsProgram 35.

Assigning value to this property resets ListLevels 115 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 115.IndentStep 116
- TrvActionParaList 129.IndentStep 130

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 215.UnitsProgram 35.

Default value for TrvActionParaBullets 122:

9 levels, all with ListType=*rvlstBullet*, FirstIndent=0, MarkerIndent is increased by IndentStep 115*2 starting from 0, LeftIndent is increased by IndentStep 115*2 starting from IndentStep 115, Font.Size=12, disk/circle/square cycled 3 times.

Default value for TrvActionParaNumbering 132:

9 levels, all with ListType=*rvlstDecimal*, FirstIndent=0, MarkerIndent is increased by IndentStep 115*2 starting from 0, LeftIndent is increased by IndentStep 115*2 starting from IndentStep 115, 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 54;

Syntax

```
TrvActionIndent = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

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 116
- TrvActionIndentInc 117

1.3.4.14.1 Properties

1.3.4.14.1.1 TrvActionIndent.IndentStep

Defines the step for paragraph indenting or unindenting.

```
property IndentStep: TRVStyleLength;
```

This value is measured in GetControlPanel 215.UnitsProgram 35.

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 114.IndentStep 115
- TrvActionParaList 129.IndentStep 130

1.3.4.15 TrvActionIndentDec

TrvActionIndentDec is the action for "Decrease Indent" command.

Unit RichViewActions 54;

Syntax

```
TrvActionIndentDec = class(TrvActionIndent 115)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionIndent 115

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionIndentInc = class(TrvActionIndent 115)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionIndent 115

Description

The action does not allow changing the paragraph indent above the MaxIndent 117.

1.3.4.16.1 Properties

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 215.UnitsProgram 35.

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 54;

Syntax

```
TrvActionLineSpacing = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionLineSpacing100 118
- TrvActionLineSpacing150 118
- TrvActionLineSpacing200 119

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 54;

Syntax

```
TrvActionLineSpacing100 = class(TrvActionLineSpacing 117)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionLineSpacing 117

Description

This action does not introduce any new properties in addition to properties 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 54;

Syntax

```
TrvActionLineSpacing150 = class(TrvActionLineSpacing 117)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionLineSpacing 117

Description

This action does not introduce any new properties in addition to properties 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 RichViewActions 54;

Syntax

```
TrvActionLineSpacing200 = class(TrvActionLineSpacing 117)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133
TrvActionLineSpacing 117

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This actions changes the following properties of styles of the selected paragraphs (RVStyle.ParaStyles[]):

- LineSpacingType := *rvlsPercent*
- 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 54;

Syntax

```
TrvActionParaBiDi = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

Description

This action does not introduce any new properties in addition to properties 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 131
- TrvActionParaRTL 132

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 105
- TrvActionTextLTR 106
- TrvActionTextRTL 106

1.3.4.22 TrvActionParaBorder

TrvActionParaBorder is the action for "Paragraph Border and Background" command.

Unit RichViewActions 54;

Syntax

```
TrvActionParaBorder = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

Description

If UserInterface 121=*False*, the action applies its properties to the selected paragraphs. The list of properties to apply is specified in ValidProperties 121.

If UserInterface 121=*True*, the action displays a dialog (initialized with properties of the selected paragraphs), assigns the changes to the action's properties, and then applies these properties to the selected paragraph.

This action allows changing attributes of paragraph border and background. For changing other paragraph attributes, see TrvActionParagraph[124].

1.3.4.22.1 Properties

1.3.4.22.1.1 TrvActionParaBorder.Background

Specifies background properties for applying to the selected paragraphs.

```
property Background: TRVBackgroundRect;
```

If UserInterface[121]=*False* and *rvpibBackground_**** in ValidProperties[121], the action applies selected subproperties of this property to the selected paragraphs.

If UserInterface[121]=*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[121], 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[121]=*False* and *rvpibBorder_**** in ValidProperties[121], the action applies selected subproperties of this property to the selected paragraphs.

If UserInterface[121]=*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[121], 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, rvpibBackg
    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_Bot
```

```
  TRVParaInfoBorderProperties = set of TRVParaInfoBorderProperty;

property ValidProperties: TRVParaInfoBorderProperty;
```

If UserInterface[121]=*False,* assign a subset of properties (for applying) to ValidProperties .

If UserInterface[121]=*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.

1.3.4.23 TrvActionParaBullets

TrvActionParaBullets is the action for "Bullets" command.

Unit RichViewActions[54];

Syntax

```
TrvActionParaBullets = class(TrvActionCustomParaListSwitcher[114])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomParaListSwitcher[114]

Description

This action applies or removes bullets to the selected paragraphs.

This action does not introduce any new properties in addition to properties of TrvActionCustomParaListSwitcher.

ListLevels[115] 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[54];

Syntax

```
TrvActionParaColor = class(TrvActionParaCustomColor[124])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomColor[204]
TrvActionParaCustomColor[124]

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor[204].

On execution, this action changes Background.Color property of styles of the selected paragraphs (RVStyle.ParaStyles[].Background.Color).

See also the list of color changing actions in the topic about TrvActionCustomColor[204].

1.3.4.25 TrvActionParaColorAndPadding

TrvActionParaColorAndPadding is the action for changing background color and padding of the selected paragraphs.

Unit RichViewActions[54];

Syntax

```
TrvActionParaColorAndPadding = class(TrvActionParaColor[122])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomColor[204]
TrvActionParaCustomColor[124]
TrvActionParaColor[122]

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[204].

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[123] property is assigned only if the corresponding property of UsePadding[124] is *True*. If the padding exceeds the paragraph's indents/spacing (LeftIndent/SpaceBefore/RightIndent/SpaceAfter), indents/spacing is increased.

Values of Padding[123] and UsePadding[124] properties must be assigned by the programmer.

1.3.4.25.1 Properties

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.ParaStyles[].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[124] property specifies which sides of padding are applied.

1.3.4.25.1.2 TrvActionParaColorAndPadding.UsePadding

Specifies sides of paragraphs for applying Padding[123].

```
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[54];

Syntax

```
TrvActionParaCustomColor = class(TrvActionCustomColor[204])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomColor[204]
TrvActionParaCustomColor[124]

Description

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor[204].

This action is not used directly. The following actions are inherited from it:

- TrvActionParaColor[122]
- TrvActionParaColorAndPadding[123]

1.3.4.27 TrvActionParagraph

TrvActionParagraph is the action for "Paragraph" command.

Unit RichViewActions[54];

Syntax

```
TrvActionParagraph = class(TrvActionParaStyles[133])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction

TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 133

Description

If UserInterface 129=*False*, the action applies its properties to the selected paragraphs. The list of properties to apply is specified in ValidProperties 129.

If UserInterface 129=*True*, the action displays a paragraph dialog (initialized with properties of the selected paragraphs), assigns the changes to the action's properties, and then applies these properties to the selected paragraph.

This action allows changing almost all paragraph attributes, except for border and background (see TrvActionParaBorder 120).

1.3.4.27.1 Properties

1.3.4.27.1.1 TrvActionParagraph.Alignment

Specifies the alignment for applying to the selected paragraphs.

```
property Alignment: TRVAlignment;
```

If UserInterface 129=*False* and *rvpimAlignment* in ValidProperties 129, the action applies this property to the selected paragraphs.

If UserInterface 129=*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 129, 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 129 are deleted.

After that, tab-stops from Tabs 128 are added.

All these properties are applied only if ValidProperties 129 contains *rvpimTabs*.

If UserInterface 129=*True*, you should assign these properties yourself.

If UserInterface 129=*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 129=*False* and *rvpimFirstIndent* in ValidProperties 129, the action applies this property to the selected paragraphs.

If UserInterface 129=*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 129, 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[129]=*False* and *rvpimKeepLinesTogether* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimKeepWithNext* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimLastLineAlignment* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimLeftIndent* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimLineSpacing* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[127] property. These two properties are applied together.

If LineSpacingType[127]<>*rvlsPercent*, 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[129]=*False* and *rvpimLineSpacing* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[127] 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[129]=*False* and *rvpimOutlineLevel* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimRightIndent* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimSpaceAfter* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]=*False* and *rvpimSpaceBefore* in ValidProperties[129], the action applies this property to the selected paragraphs.

If UserInterface[129]=*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[129], 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[129]) or all (DeleteAllTabs[125]) tab-stops are deleted.

All these properties are applied only if ValidProperties[129] contains *rvpimTabs*.

If UserInterface[129]=*True*, you should assign these properties yourself.

If UserInterface[129]=*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[125]=*True*, all tab-stops are deleted instead.

After that, tab-stops from Tabs[128] are added.

All these properties are applied only if ValidProperties[129] contains *rvpimTabs*.

If UserInterface[129]=*True*, you should assign these properties yourself.

If UserInterface[129]=*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[129]=*False,* assign a subset of properties (for applying) to ValidProperties .

If UserInterface[129]=*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.

1.3.4.28 TrvActionParaList

TrvActionParaList is the action for "Bullets and Numbering" command.

Unit RichViewActions[54];

Syntax

```
TrvActionParaList = class(TrvActionParaStyles[133])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction[213]
TrvAction[202]
TrvActionParaStyles[133]

Description

This action displays a bullet & numbering dialog. Two kinds of dialogs are used, depending on the value of GetControlPanel[215].UserInterface[36]: a general dialog and an HTML dialog.

In the general kind of dialog, the user can choose from predefined templates (may be after customizing them). If the caret is placed in 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). For numbered styles, the user can choose to continue or to reset numbering, or to create a new list.

In the HTML kind of dialog, templates cannot be customized. All templates have 1 level. The "level" field in this dialog actually does not mean a list level, it affects only indenting (MarkerIndent of list level is IndentStep[130] *level). For numbered styles, the user can choose to continue or to reset numbering.

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[122] 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[132] actions.

1.3.4.28.1 Properties

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[215].UnitsProgram[35].

This property is used:

- in an HTML-style list&numbering dialog (if GetControlPanel[215].UserInterface[36]=*rvauiHTML*); this dialog creates a paragraph list with MarkerIndent = level*IndentStep, LeftIndent = (level+1)*IndentStep;
- in a general-style dialog, to adjust indents of a predefines set of styles (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[114].IndentStep[115]
- TrvActionIndent[115].IndentStep[116]

1.3.4.28.1.2 TrvActionParaList.UpdateAllActionsOnForm

Allows updating all TrvActionParaBullets[122] and TrvActionParaNumbering[132] 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[115] of all TrvActionParaBullets[122] actions on the same form/datamodule.
- when the user applies a list with all numbered levels, the action assigns these levels to ListLevels[115] of all TrvActionParaNumbering[132] actions on the same form/datamodule.

Default value:

True

See also properties:

- ActionParaNumbering 131
- ActionParaBullets 131

1.3.4.28.1.3 TrvActionParaList.ActionParaNumbering

Defines a link to the "Numbering" action.

```
property ActionParaNumbering: TrvActionParaNumbering 132;
```

When the user applies a list with all numbered levels, the action assigns these levels to ListLevels 115 properties to the action specified in this link. If UpdateAllActionsOnForm 130=*True*, these levels are assigned to ListLevels 115 of all TrvActionParaNumbering 132 actions on the same form/datamodule.

1.3.4.28.1.4 TrvActionParaList.ActionParaBullets

Defines a link to the "Bullets" action.

```
property ActionParaBullets: TrvActionParaBullets 122;
```

When the user applies a list without numbered levels, the action assigns these levels to ListLevels 115 properties to the action specified in this link. If UpdateAllActionsOnForm 130=*True*, these levels are assigned to ListLevels 115 of all TrvActionParaBullets 122 actions on the same form/datamodule.

1.3.4.29 TrvActionParaLTR

TrvActionParaLTR is the action for "Left to Right" command.

Unit RichViewActions 54;

Syntax

```
TrvActionParaLTR = class(TrvActionParaBiDi 119)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 107
TrvActionParaBiDi 119

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 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 106
- TrvActionTextRTL 106
- TrvActionParaRTL 132

1.3.4.30 TrvActionParaNumbering

TrvActionParaNumbering is the action for "Numbering" command.

Unit RichViewActions 54;

Syntax

```
TrvActionParaNumbering = class(TrvActionCustomParaListSwitcher 114)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionCustomParaListSwitcher 114

Description

This action applies or removes numbering to the selected paragraphs.

This action does not introduce any new properties in addition to properties of TrvActionCustomParaListSwitcher.

ListLevels 115 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 115, applies this list style;
2. otherwise, if RVStyle.ListStyles has unused list style with the same properties as ListLevels 115, applies this list style;
3. otherwise, creates a new list style, assigns ListLevels 115 to it, applies this list style.

See also:

- TrvActionParaBullets 122
- TrvActionParaList 129

1.3.4.31 TrvActionParaRTL

TrvActionParaRTL is the action for "Right to Left" command.

Unit RichViewActions 54;

Syntax

```
TrvActionParaRTL = class(TrvActionParaBiDi 119)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 107
TrvActionParaBiDi 119

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 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 106
- TrvActionTextRTL 106
- TrvActionParaLTR 131

1.3.4.32 TrvActionParaStyles

TrvActionParaStyles is a base class for the actions applying changes to the selected paragraphs.

Unit RichViewActions 54;

Syntax

```
TrvActionParaStyles = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionParagraph 124
- TrvActionAlignment 111
- TrvActionParaBiDi 119
- TrvActionIndent 115
- TrvActionWordWrap 134
- TrvActionLineSpacing 117
- TrvActionParaBorder 120

1.3.4.33 TrvActionWordWrap

TrvActionWordWrap is the action for "Word Wrap" command.

Unit RichViewActions 54;

Syntax

```
TrvActionWordWrap = class(TrvActionParaStyles 133)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionParaStyles 107

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 properties 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

1.3.5.1 TrvActionStyleTemplates

TrvActionStyleTemplates is the action for "Format | Styles" command.

Unit RichViewActions 54;

Syntax

```
TrvActionStyleTemplates = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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, RTF 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 135 property.

You can customize the appearance of the dialog window using Images[135], TextStyleImageIndex, ParaStyleImageIndex, ParaTextStyleImageIndex[136] 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[215].RVFLocalizable[33], you can change them in the control panel properties: RVFFilter[33], RVStylesFilter, RVStylesExt[34].

This action is used by other actions:

- TrvActionAddStyleTemplate[136] (to open a dialog window)
- TrvActionInsertHyperlink[145] (to get default properties of "Hyperlink" style template)

1.3.5.1.1 Properties

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[136] >=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[91], TrvActionParagraph[124], TrvActionParaBorder[120], TrvActionInsertHyperlink[145] 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. It 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[215].UnitsProgram[35].

This property may also be used by TrvActionInsertHyperlink, when it adds "Hyperlink" style template.

See also:

TrvActionNew[61].StyleTemplates[62]
TrvActionInsertHyperlink[145].AutoAddHyperlinkStyleTemplate[147].

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[135] 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[54];

Syntax

```
TrvActionAddStyleTemplate = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

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[134] action. It can be linked using ActionStyleTemplates[136] property; if this property is not assigned, the action searches for TrvActionStyleTemplates[134] action on the same form.

1.3.5.2.1 Properties

1.3.5.2.1.1 TrvActionAddStyleTemplate.ActionStyleTemplates

Links this action to TrvActionStyleTemplates[134] action.

```
property ActionStyleTemplates: TrvActionStyleTemplates[134];
```

If this property is not assigned, the action searches for TrvActionStyleTemplates[134] action on the same form. If not found, the action does nothing.

TrvActionStyleTemplates[134] defines properties for the style management dialog window.

1.3.5.3 TrvActionStyleInspector

TrvActionStyleInspector is the action for "Format | Style Inspector" command.

Unit RichViewActions[54];

Syntax

```
TrvActionCustomInfoWindow = class(TrvAction[202]);
TrvActionStyleInspector = class(TrvActionCustomInfoWindow);
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

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[202] property of this action, and the info window will be updated automatically according to changes in Control[202]. When the action is executed for some editor, the action automatically assign this editor to Control[202].

You can customize the appearance of the info window using Images[137], FontImageIndex and ParaImageIndex[137] properties.

1.3.5.3.1 Properties

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[137] properties.

1.3.5.3.1.2 TrvActionStyleInspector.FontImageIndex, ParaImageIndex

Indices in Images[137].

```
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

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[215].Language[32] is changed.

1.3.5.3.3 Events

1.3.5.3.3.1 TrvActionCustomInfoWindow.OnShowing

Occurs before showing the info window.

```
property OnShowing: TRVShowFormEvent[252];
```

Example (embedding the info window in the main window TfrmMain, at the right side):

```
procedure TfrmMain.rvActionStyleInspector1Showing(Sender: TrvAction[202];
  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[54];

Syntax

```
TrvActionClearFormat = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

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 of TrvAction.

1.3.5.5 TrvActionClearTextFormat

TrvActionClearFormat is the action for "Format | Clear Text Format" command.

Unit RichViewActions [54];

Syntax

```
TrvActionClearTextFormat = class(TrvAction [202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction [213]
TrvAction [202]

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 properties of TrvAction.

1.3.6 Insert

1.3.6.1 TrvActionBookmarks

TrvActionBookmarks is the action for "Insert Bookmark..." command.

Unit RichViewActions [54];

Syntax

```
TrvActionBookmarks = class(TrvAction [202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction [213]
TrvAction [202]

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 [140].

If you use this action with Report Workshop, assign '{}' to AllowedCharacters[140].

1.3.6.1.1 Properties

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 allows 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 affect 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.

Initial value:

True

1.3.6.2 TrvActionInsertCaption

TrvActionInsertCaption is the action for "Insert Object Caption..." command.

Unit RichViewActions[54];

Syntax

```
TrvActionInsertCaption = class(TrvActionInsertNumSequence[155])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionInsertNumSequence[155]

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[156]=*True*, the dialog is initialized with a label 'Figure' (localized string), a decimal numbering type, ExcludeLabel[141] and InsertAbove[141] properties.

When the user clicks the "Insert" button, a caption is inserted above or below the selected object, UseDefaults[156] is reset to *False*, and chosen values are stored in SeqName[155], NumberType[155], ExcludeLabel[141] and InsertAbove[141] properties (they are used to initialize a dialog when it will be displayed for the next time).

1.3.6.2.1 Properties

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[54];

Syntax

```
TrvActionInsertCustomPageNumber = class(TrvAction[202])
```

Hierarchy

TObject

TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action is not used directly. It is a parent for the following actions:

- TrvActionInsertPageNumber 157
- TrvActionInsertPageCount 156

1.3.6.3.1 Properties

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 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 143 = *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

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[143] to its text.

If **UserInterface** = *False*, the action displays a dialog for editing properties of the new equation, initializing it with Expression[143].

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[143]

1.3.6.5 TrvActionInsertFile

TrvActionInsertFile is the action for "Insert | File" command.

Unit RichViewActions[54];

Syntax

```
TrvActionInsertFile = class(TrvActionCustomFileIO[58])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomIO[59]
TrvActionCustomFileIO[58]

Description

This action displays a file opening dialog and inserts the chosen file in the caret position.

In addition to formats supported by TrvActionOpen[63] action, this action supports:

- HTML (not only if rvHtmlViewImporter[239], but also if rvHtmlImporter[238] is used)
- formats supported by Microsoft Office file import converters.

1.3.6.5.1 Properties

1.3.6.5.1.1 TrvActionInsertFile.Filter

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

```
property Filter: TrvFileImportFilterSet[251];
```

If *ffiCustom* is included, formats listed in the CustomFilter[58] property are used.

Name and extension of RichView Format (RVF) may be customized, see GetControlPanel[215].RVFFilter[33].

Default value:

all formats

1.3.6.5.2 Methods

1.3.6.5.2.1 TrvActionInsertFile.InsertFile

Inserts the specified file.

```
function InsertFile(rve: TCustomRichViewEdit;
  const FileName: TRVUnicodeString;
  FileFormat: TrvFileImportFilter[251];
  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[58] 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[54];

Syntax

```
TrvActionInsertHLine = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action inserts a horizontal line ("break" item type) in the caret position in TCustomRichViewEdit component.

The inserted line has the specified Color 145 and Width 145, and Style 145.

1.3.6.6.1 Properties

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 215.UnitsProgram 35.

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 54;

Syntax

```
TrvActionInsertHyperlink = class(TrvActionTextStyles 107)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction

TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTextStyles 107

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 153 and OnGetHyperlinkTargetFromItem 153 events.

Using style templates

Visual appearance of a hyperlink depends on whether style templates are used 18.

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 150 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 149 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 151 returns the index of hypertext style for the given text style;
- GetNormalStyleNo 152 returns the index of non-hypertext style for the given text style;
- DetectURL 150 analyses text to the left of the caret; if it is URL, converts it to hyperlink;
- TerminateHyperlink 153 converts the current text style to non-hypertext, if the caret is at the end of hyperlink;
- GoToLink 152 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

1.3.6.7.1.1 TrvActionInsertHyperlink.ActionStyleTemplates

Links this action to TrvActionStyleTemplates[134] action.

```
property ActionStyleTemplates: TrvActionStyleTemplates[134];
```

This property is used only if style templates are used[18] (i.e. the target editor's UseStyleTemplates=*True*).

If this link is not defined, the first found TrvActionStyleTemplates[134] action on the same form/datamodule is used.

See StyleTemplateName[149] 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[18] (i.e. the target editor's UseStyleTemplates=*True*).

See StyleTemplateName[149] 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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlBackColor* is included in ValidProperties[150].

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clNone

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlColor* is included in ValidProperties[150].

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlCursor* is included in ValidProperties[150].

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlHoverBackColor* is included in ValidProperties[150].

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlHoverColor* is included in ValidProperties[150].

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clBlue

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlHoverEffects* is included in ValidProperties[150].

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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlHoverUnderlineColor* is included in ValidProperties[150].

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clNone (instructs using UnderlineColor[150])

1.3.6.7.1.10 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

1.3.6.7.1.11 TrvActionInsertHyperlink.SpaceFiller

Specifies the string used for replacing space characters in hyperlink targets (by EncodeTarget[151]).

```
property SpaceFiller: TRVUnicodeString;
```

Default value:

' ' (space character, so nothing is replaced)

1.3.6.7.1.12 TrvActionInsertHyperlink.Style

Defines the font style for hyperlinks.

```
property Style: TFontStyles;
```

This property is used only if style templates are not used[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlBold*, *rvhlItalic*, *rvhlUnderline*, or *rvhlStrikeout* are included in ValidProperties[150].

For example:

- if *rvhlBold* is included in ValidProperties[150] and *fsBold* is included in Style, hyperlinks will be bold;
- if *rvhlBold* is included in ValidProperties[150] and *fsBold* is not included in Style, hyperlinks will not be bold;
- if *rvhlBold* is not included in ValidProperties[150] (default), boldness is not changed.

Default value:

[*fsUnderline*]

1.3.6.7.1.13 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[18] (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[147]=*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[147] TrvActionStyleTemplates[134] action, if it has "Hyperlink" in its StandardStyleTemplates[135]; otherwise, the action uses hard-coded default values.

Default value:

'Hyperlink'

1.3.6.7.1.14 TrvActionInsertHyperlink.UnderlineColor

Defines the underline color for hyperlinks.

```
property UnderlineColor: TColor;
```

This property is used only if style templates are not used[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

This property is used when converting text to hypertext, if *rvhlHoverUnderlineColor* is included in ValidProperties[150].

A value of this property can be changed by the user in the hyperlink properties dialog.

Default value:

clNone (instructs using Color[147])

1.3.6.7.1.15 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[18] (i.e. the target editor's UseStyleTemplates=*False*). If style templates are used, the action applies StyleTemplateName[149] instead.

When this action converts text to hypertext, properties of this action listed in this property are applied:

- BackColor[147]
- Color[147]
- HoverBackColor[148]
- HoverColor[148]
- HoverEffects[148]
- HoverUnderlineColor[148]
- UnderlineColor[150]
- Style[149]

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

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[151] for this conversion.

Example:

```
procedure TForm1.RichViewEdit1KeyPress(Sender: TObject; var Key: Char);
begin
  if Key in [#13, #9, ' ', ',', ';'] then begin
    rvActionInsertHyperlink1.DetectURL(RichViewEdit1);
    rvActionInsertHyperlink1.TerminateHyperlink[153](RichViewEdit1);
  end;
end;
```

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[149].

Normally, the returned value is the same as **Target** (because the action does not allow entering multiline targets, and SpaceFiller[149]=' ').

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[18], 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[150]. For these properties, values specified in properties of this action are used.

If style templates are used[18], the method returns the index of style (in **rve**.Style.TextStyles) having all properties of the specified style, by with the style template StyleTemplateName[149] 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[150] 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[151](Target);
end;
```

See also methods:

- GetNormalStyleNo[152]

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 [18], 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 [150]. For these properties, values of properties of **rve**.Style.TextStyles[0] are used.

If style templates are used [18], 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 [153] method, or when user entered an empty target in the dialog.

See also methods:

- GetHyperlinkStyleNo [151]

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 '#').

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 [244] 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 152 is used.

Example:

```
procedure TForm1.RichViewEdit1KeyPress(Sender: TObject; var Key: Char);
begin
  if Key in [#13, #9, ' ', ',', ';'] then begin
    rvActionInsertHyperlink1.DetectURL 150 (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.

1.3.6.7.3 Events

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 54;

Syntax

```
TrvActionInsertNumber = class(TrvActionInsertNumSequence 155)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertNumSequence 155

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[156]=*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[156] is reset to *False*, and chosen values are stored in SeqName[155] and NumberType[155] 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[54];

Syntax

```
TrvActionInsertNumSequence = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action is not used directly.

The following actions are inherited from it:

- TrvActionInsertNumber
- TrvActionInsertCaption

1.3.6.9.1 Properties

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[156]=*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[156]=*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[155] and NumberType[155] 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[155] and NumberType[155] 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[54];

Syntax

```
TrvActionInsertPageBreak = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

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 of TrvAction.

See also:

- TrvActionRemovePageBreak[211]

1.3.6.11 TrvActionInsertPageCount

TrvActionInsertPageCount is the action for "Insert | Page Count" command.

Unit RichViewActions[54];

Syntax

```
TrvActionInsertPageCount = class(TrvActionInsertCustomPageNumber[141])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction

TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertCustomPageNumber 141

Description

This action inserts "page count" field in the position of the caret. The number is formatted according to NumberType 142 property.

See also

- TrvActionInsertPageNumber 157

1.3.6.12 TrvActionInsertPageNumber

TrvActionInsertPageNumber is the action for "Insert | Page Number" command.

Unit RichViewActions 54;

Syntax

```
TrvActionInsertPageNumber = class(TrvActionInsertCustomPageNumber 141)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertCustomPageNumber 141

Description

This action inserts "page number" field in the position of the caret. The number is formatted according to NumberType 142 property.

See also

- TrvActionInsertPageCount 156

1.3.6.13 TrvActionInsertPicture

TrvActionInsertPicture is the action for "Insert | Picture" command.

Unit RichViewActions 54;

Syntax

```
TrvActionInsertPicture = class(TrvActionCustomIO 59)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213

TrvAction 202
TrvActionCustomIO 59

Description

This action inserts a page picture from file in 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 following additional properties can be applied to inserted pictures:

- VAlign 160 – the image alignment relative to text;
- Spacing 159 – padding (spacing between the image itself and its border);
- OuterHSpacing, OuterVSpacing 159 – horizontal and vertical spacing around the image border;
- BorderWidth, BorderColor 158 – border width and color;
- BackgroundColor 158 – background color;
- MaxImageSize 159 – maximal allows size (the image will be stretched if it exceeds this size);
- StoreFileNameInItemName 160 allows storing file name with the image (It's not recommended to use this option. The recommended place for storing image file names is *rvespImageFileName* 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 160 event.

1.3.6.13.1 Properties

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 215.UnitsProgram 35.

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 210
- TrvActionItemProperties.BackgroundGraphicFilter 209
- TrvActionTableProperties.BackgroundGraphicFilter 195

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 *rvepImageHeight* and *rvepImageWidth* properties are assigned, see the help file on TRVExtraItemProperty).

This value is measured in GetControlPanel 215.UnitsProgram 35.

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 215.UnitsProgram 35.

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 215.UnitsProgram 35.

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 *rvespImageFileName* 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 84
- TrvActionItemProperties 208 (does not support storing paths in item names)
- RTF reading (does not support storing paths in item names)
- TRvHtmlViewImporter 239.StoreFileNameInItemName
- TRvHtmlImporter 238.Options (*rvhtmloImgSrcToImageName* option)

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

1.3.6.13.2.1 TrvActionInsertPicture.OnInserting

Occurs before inserting image in the target editor.

```
type
  TRVAInsertPictureEvent = procedure(Sender: TObject;
    Editor: TCustomRichViewEdit; var Graphic: TGraphic; const FileName: TRVUnicodeString)
    of object;
property OnInserting: TRVAInsertPictureEvent;
```

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 54;

Syntax

```
TrvActionInsertSymbol = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action inserts a character in the caret position of the target editor.

1.3.6.14.1 Properties

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 224;
```

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 161 = *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[161], 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[54];

Syntax

```
TrvActionInsertText = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action calls OnInsertText[162] 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 of TrvAction.

See also:

- TrvActionEvent[206]

1.3.6.15.1 Events

1.3.6.15.1.1 TrvActionInsertText.OnInsertText

Requests text for insertion.

```
type
  TRVAInsertTextEvent = procedure (Sender: TObject; Editor: TCustomRichViewEdit;
    var Text: TRVUnicodeString) of object;

property OnInsertText: TRVAInsertTextEvent;
```

The string assigned to **Text** will be inserted in **Editor**.

1.3.7 Notes

1.3.7.1 TrvActionCustomInsertSidenote

TrvActionCustomInsertSidenote is a base class for the actions inserting sidenotes and text box items.

Unit RichViewActions[54];

Syntax

```
TrvActionCustomInsertSidenote = class(TrvActionInsertNote[167])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertNote 167

Description

This action introduces properties that are assigned to properties of an inserted sidenote (or a text box item):

- BoxProperties 164
- BoxPosition 163

Text in the note's Document is formatted either according to StyleTemplateName 164 (if style templates are used) or according to Font 168 (otherwise).

This class is not used directly. The following actions are inherited from TrvActionCustomInsertSidenote:

- TrvActionInsertSidenote 168
- TrvActionInsertTextBox 169

1.3.7.1.1 Properties

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 168**:**

- HorizontalAnchor=*rvhanMainTextArea*;
- HorizontalPositionKind=*rvfpkAlignment*;
- HorizontalAlignment=*rvfphalRight*;
- VerticalAnchor=*rvvanParagraph*;
- VerticalPositionKind=*rvfpkAbsPosition*;
- VerticalOffset=0.

TrvActionInsertTextBox 169**:**

- 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[168] are used.

Default values are different in inherited actions:

TrvActionInsertSidenote[168]:

'Sidenote Text' (note: this style template is standard for RichViewActions, but it is not standard for Microsoft Word)

TrvActionInsertTextBox[169]:

'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[54];

Syntax

```
TrvActionInsertNote = class(TrvCustomEditorAction[215])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvCustomEditorAction[215]

Description

ScaleRichView note: it's not recommended to use this action for TSRichViewEdit. Use TsrvActionEditNote instead.

This action can be executed directly, or it is called by TrvActionInsertFootnote[166], TrvActionInsertEndnote[165], TrvActionInsertSidenote[168], TrvActionInsertTextBox[169].

It's recommended to assign Control[202] 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 165=*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 202): 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 202;
- saving Control 202 to a file or a stream;
- printing or displaying a print preview of Control 202;
- showing or hiding the note editor window;
- assigning a different value to Control 202.

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 216 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 28 when the note editor acquires and releases the input focus (see the example in the topic about OnShowing and OnHide 216)

The latter is implemented in the ActionTest demos.

1.3.7.2.1 Properties

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

1.3.7.3 TrvActionInsertEndnote

TrvActionInsertEndnote inserts a new endnote in the document.

Unit RichViewActions 54;

Syntax

```
TrvActionInsertEndnote = class(TrvActionInsertNote 167)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertNote 167

Description

ScaleRichView note: it's not recommended to use this action for TSRichViewEdit. Use TsrvActionInsertEndnote instead.

TrvActionInsertEndnote inserts a new endnote in the document, then executes ActionEditNote 168 to open an editor for this note.

If TrvActionEditNote 164'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 168 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 54;

Syntax

```
TrvActionInsertFootnote = class(TrvActionInsertNote 167)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionInsertNote 167

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[168] to open an editor for this note.

If TrvActionEditNote[164]'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[168] 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[54];

Syntax

```
TrvActionInsertNote = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This class is not used directly.

The following actions are inherited from TrvActionInsertNote:

- TrvActionInsertFootnote[166]
- TrvActionInsertEndnote[165]
- TrvActionInsertSidenote[168]
- TrvActionInsertTextBox[169]

1.3.7.5.1 Properties

1.3.7.5.1.1 TrvActionInsertNote.ActionEditNote

Links this action to TrvActionEditNote[164] action.

```
property ActionEditNote: TrvActionEditNote[164];
```

The action executes ActionEditNote after inserting a note.

If this link is not defined, the first found TrvActionEditNote[164] 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[164] 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[54];

Syntax

```
TrvActionCustomInsertSidenote = class(TrvActionCustomInsertSidenote[162])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionInsertNote[167]
TrvActionCustomInsertSidenote[162]

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[168] to open an editor for this note.

If TrvActionEditNote[164]'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[169] 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[164] style template. Otherwise, or if this style template does not exist, sidenote text is formatted using Font[168] property.

The following properties are assigned to the inserted sidenote:

- BoxProperties[164] (default: width: 20% of page width, height: auto)
- BoxPosition[163] (default: horizontal - right aligned to the main text area, vertical - aligned to the top of the paragraph)

1.3.7.6.1 Properties

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[54];

Syntax

```
TrvActionInsertTextBox = class(TrvActionCustomInsertSidenote[162])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionInsertNote[167]
TrvActionCustomInsertSidenote[162]

Description

ScaleRichView note: it's not recommended to use this action for TSRichViewEdit. Use TsrvActionInsertTextBox instead.

TrvActionInsertTextBox inserts a new text box item in the document, then executes ActionEditNote[168] 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[164] style template. Otherwise, or if this style template does not exist, this text is formatted using Font[168] property.

The following properties are assigned to the inserted text box:

- BoxProperties[164] (default: width: 20% of page width, height: auto)
- BoxPosition[163] (default: horizontal - centered in the main text area, vertical - 20 pixels below the top of the line)

1.3.8 Tables

1.3.8.1 TrvActionInsertTable

TrvActionInsertTable is the action for "Table | Insert Table" command.

Unit RichViewActions[54];

Syntax

```
TrvActionInsertTable = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

The action can insert a table in two ways:

- using a table dialog (when executed)
- using a special table size window (when ShowTableSizeDialog[178] 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[171]. 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[176], ColCount[174] and BestWidth[171] 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[176] and RowsVAlign[176] are assigned to properties of all rows of the table,

- OnInserting 178 event occurs.

The user can assign properties of this action by checking "Default" check box in the table properties dialog displayed by TrvActionTableProperties 194 or TrvActionItemProperties 208.

1.3.8.1.1 Properties

1.3.8.1.1.1 TrvActionInsertTable.BackgroundPicture

Defines the table background image.

```
property BackgroundPicture: TPicture;
```

BackgroundPicture.Graphic is assigned to the table's BackgroundImage

1.3.8.1.1.2 TrvActionInsertTable.BackgroundStyle

Defines the table background style.

```
property BackgroundStyle: TRVItemBackgroundStyle;
```

This property is assigned to the table property of the same name.

Default value:

rvbsColor

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 178 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 215.UnitsProgram 35.

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 215.UnitsProgram 35.

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 215.UnitsProgram 35.

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 215.UnitsProgram 35.

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[215].UnitsProgram[35].

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[215].UnitsProgram[35].

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[215].UnitsProgram[35].

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[215].UnitsProgram[35].

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[178]

1.3.8.1.1.18 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[178] method.

Default value:

2

1.3.8.1.1.19 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

1.3.8.1.1.20 TrvActionInsertTable.HeadingRow Count

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

1.3.8.1.1.21 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[175]>0.

Default value:

False

1.3.8.1.1.22 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[175]>0.

Default value:

clWindowText

1.3.8.1.1.23 TrvActionInsertTable.HRuleWidth

Defines the width of horizontal rules.

```
property HRuleWidth: TRVStyleLength;
```

This value is measured in GetControlPanel[215].UnitsProgram[35].

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 178 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: TSRichViewEdit 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[177]>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[177]>0.

Default value:

clWindowText

1.3.8.1.1.33 TrvActionInsertTable.VRuleWidth

Defines the width of vertical rules.

```
property VRuleWidth: TRVStyleLength;
```

This value is measured in GetControlPanel[215].UnitsProgram[35].

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

clNone

See also

- ColBandSize, RowBandSize

1.3.8.1.2 Methods

1.3.8.1.2.1 TrvActionInsertTable.Show TableSizeDialog

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); overlo
```

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

1.3.8.1.3.1 TrvActionInsertTable.OnCloseTableSizeDialog

Occurs when a table sizing window (shown by ShowTableSizeDialog 178 method) is closed.

```
property OnCloseTableSizeDialog: TNotifyEvent;
```

This event can be used to uncheck a button pressed before calling ShowTableSizeDialog 178 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
    TRVAInsertTableEvent = procedure (Sender: TrvActionInsertTable; table: TRVTableItemInfo

property OnInserting: TRVAInsertTableEvent;
```

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 54;

Syntax

```
TrvActionTableCell = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionTableRCBase 196
- TrvActionTableDeleteTable 190
- TrvActionTableMergeCells 193
- TrvActionTableSplitCells 200
- TrvActionTableSelectTable 198
- TrvActionTableSelectCell 196
- TrvActionTableMultiCellAttributes 194
- TrvActionTableProperties 194

1.3.8.3 TrvActionTableCellAllBorders

TrvActionTableCellAllBorders is the action for "Table | Cell Borders | All Borders" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellAllBorders = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 of TrvAction.

See also:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellRightBorder 182
- TrvActionTableCellTopBorder 186
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellNoBorders 182

1.3.8.4 TrvActionTableCellBorder

TrvActionTableCellBorder is a base class for the actions changing cell borders.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellBorder = class(TrvActionTableMultiCellAttributes 194)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellRightBorder 182
- TrvActionTableCellTopBorder 186
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellAllBorders 179
- TrvActionTableCellNoBorders 182

These actions change table.Cells[].VisibleBorders properties for the selected cells (or the edited cell). All these actions (except for TrvActionTableCellNoBorders 182) 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 54;

Syntax

```
TrvActionTableCellBottomBorder = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 properties of TrvAction.

See also:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellRightBorder 182
- TrvActionTableCellTopBorder 186
- TrvActionTableCellAllBorders 179
- TrvActionTableCellNoBorders 182

1.3.8.6 TrvActionTableCellLeftBorder

TrvActionTableCellLeftBorder is the action for "Table | Cell Borders | Left Border" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellLeftBorder = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 of TrvAction.

See also:

- TrvActionTableCellRightBorder 182
- TrvActionTableCellTopBorder 186
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellAllBorders 179
- TrvActionTableCellNoBorders 182

1.3.8.7 TrvActionTableCellNoBorders

TrvActionTableCellNoBorders is the action for "Table | Cell Borders | No Borders" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellNoBorders = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 of TrvAction.

This is a redundant action, because it only provides a subset of functionality of TrvActionTableCellAllBorders 179.

See also:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellRightBorder 182
- TrvActionTableCellTopBorder 186
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellAllBorders 179

1.3.8.8 TrvActionTableCellRightBorder

TrvActionTableCellRightBorder is the action for "Table | Cell Borders | Right Border" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRightBorder = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 of TrvAction.

See also:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellTopBorder 186
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellAllBorders 179
- TrvActionTableCellNoBorders 182

1.3.8.9 TrvActionTableCellRotation

TrvActionTableCellRotation is a base class for the actions changing a rotation in table cells.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRotation = class(TrvActionTableMultiCellAttributes 194)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableCellRotationNone 183
- TrvActionTableCellRotation90 184
- TrvActionTableCellRotation180 185
- TrvActionTableCellRotation270 185

1.3.8.10 TrvActionTableCellRotationNone

TrvActionTableCellRotationNone is the action for "Table | No Cell Rotation" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRotationNone = class(TrvActionTableCellRotation 183)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellRotation 183

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 of TrvAction.

See also:

- TrvActionTableCellRotation90 184
- TrvActionTableCellRotation180 185
- TrvActionTableCellRotation270 185

1.3.8.11 TrvActionTableCellRotation90

TrvActionTableCellRotation90 is the action for "Table | Rotate Cell by 90°" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRotation90 = class(TrvActionTableCellRotation 183)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellRotation 183

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 of TrvAction.

See also:

- TrvActionTableCellRotationNone 183
- TrvActionTableCellRotation180 185
- TrvActionTableCellRotation270 185

1.3.8.12 TrvActionTableCellRotation180

TrvActionTableCellRotation180 is the action for "Table | Rotate Cell by 180°" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRotation180 = class(TrvActionTableCellRotation 183)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellRotation 183

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 of TrvAction.

See also:

- TrvActionTableCellRotationNone 183
- TrvActionTableCellRotation90 184
- TrvActionTableCellRotation270 185

1.3.8.13 TrvActionTableCellRotation270

TrvActionTableCellRotation270 is the action for "Table | Rotate Cell by 270°" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellRotation270 = class(TrvActionTableCellRotation 183)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellRotation 183

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 of TrvAction.

See also:

- TrvActionTableCellRotationNone 183
- TrvActionTableCellRotation90 184
- TrvActionTableCellRotation180 185

1.3.8.14 TrvActionTableCellTopBorder

TrvActionTableCellTopBorder is the action for "Table | Cell Borders | Top Border" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellTopBorder = class(TrvActionTableCellBorder 180)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellBorder 180

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 of TrvAction.

See also:

- TrvActionTableCellLeftBorder 181
- TrvActionTableCellRightBorder 182
- TrvActionTableCellBottomBorder 180
- TrvActionTableCellAllBorders 179
- TrvActionTableCellNoBorders 182

1.3.8.15 TrvActionTableCellVAlign

TrvActionTableCellVAlign is a base class for the actions changing a vertical alignment in table cells.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellVAlign = class(TrvActionTableMultiCellAttributes 194)
```

Hierarchy

TObject
TPersistent
TComponent

TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableCellVAlignTop 189
- TrvActionTableCellVAlignMiddle 188
- TrvActionTableCellVAlignBottom 187
- TrvActionTableCellVAlignDefault 187

1.3.8.16 TrvActionTableCellVAlignBottom

TrvActionTableCellVAlignBottom is the action for "Table | Align Cell to the Bottom" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellVAlignBottom = class(TrvActionTableCellVAlign 186)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellVAlign 186

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 of TrvAction.

See also:

- TrvActionTableCellVAlignTop 189
- TrvActionTableCellVAlignMiddle 188
- TrvActionTableCellVAlignDefault 187

1.3.8.17 TrvActionTableCellVAlignDefault

TrvActionTableCellVAlignDefault is the action for "Table | Default Cell Vertical Alignment" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellVAlignDefault = class(TrvActionTableCellVAlign 186)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellVAlign 186

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 of TrvAction.

See also:

- TrvActionTableCellVAlignTop 189
- TrvActionTableCellVAlignMiddle 188
- TrvActionTableCellVAlignBottom 187

1.3.8.18 TrvActionTableCellVAlignMiddle

TrvActionTableCellVAlignMiddle is the action for "Table | Align Cell to the Middle" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellVAlignMiddle = class(TrvActionTableCellVAlign 186)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellVAlign 186

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 of TrvAction.

See also:

- TrvActionTableCellVAlignTop 189
- TrvActionTableCellVAlignBottom 187

- TrvActionTableCellVAlignDefault 187

1.3.8.19 TrvActionTableCellVAlignTop

TrvActionTableCellVAlignTop is the action for "Table | Align Cell to the Top" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableCellVAlignTop = class(TrvActionTableCellVAlign 186)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableMultiCellAttributes 194
TrvActionTableCellVAlign 186

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 of TrvAction.

See also:

- TrvActionTableCellVAlignMiddle 188
- TrvActionTableCellVAlignBottom 187
- TrvActionTableCellVAlignDefault 187

1.3.8.20 TrvActionTableDeleteCols

TrvActionTableDeleteCols is the action for "Table | Delete Columns" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableDeleteColumns = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableDeleteRows 190

1.3.8.21 TrvActionTableDeleteRows

TrvActionTableDeleteRows is the action for "Table | Delete Rows" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableDeleteRows = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableDeleteCols 189

1.3.8.22 TrvActionTableDeleteTable

TrvActionTableDeleteTable is the action for "Table | Delete Table" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableDeleteTable = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

Description

The action deletes the table at the caret position.

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.8.23 TrvActionTableGrid

TrvActionTableGrid is the action for "Table | Show Grid Lines" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableGrid = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 of TrvAction.

1.3.8.24 TrvActionTableInsertColLeft

TrvActionTableInsertColLeft is the action for "Table | Insert Column Left" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableInsertColLeft = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableInsertRowsAbove 192
- TrvActionTableInsertRowsBelow 193
- TrvActionTableInsertColRight 192

1.3.8.25 TrvActionTableInsertColRight

TrvActionTableInsertColRight is the action for "Table | Insert Column Right" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableInsertColRight = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableInsertRowsAbove 192
- TrvActionTableInsertRowsBelow 193
- TrvActionTableInsertColLeft 191

1.3.8.26 TrvActionTableInsertRowsAbove

TrvActionTableInsertRowsAbove is the action for "Table | Insert Row Above" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableInsertRowsAbove = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableInsertRowsBelow 193
- TrvActionTableInsertColLeft 191
- TrvActionTableInsertColRight 192

1.3.8.27 TrvActionTableInsertRowsBelow

TrvActionTableInsertRowsBelow is the action for "Table | Insert Row Below" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableInsertRowsBelow = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 193 = *True*, the action requests a number of rows to insert and inserts this count of rows.

See also:

- TrvActionTableInsertRowsAbove 192
- TrvActionTableInsertColLeft 191
- TrvActionTableInsertColRight 192

1.3.8.27.1 Properties

1.3.8.27.1.1 TrvActionTableInsertRow sBelow .Allow Multiple

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 54;

Syntax

```
TrvActionTableMergeCells = class(TrvActionTableCell[179])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionTableCell[179]

Description

The action merges the selected table cells in one.

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.8.29 TrvActionTableMultiCellAttributes

TrvActionTableMultiCellAttributes is the base class for actions changing properties of table cells.

Unit RichViewActions[54];

Syntax

```
TrvActionTableMultiCellAttributes = class(TrvActionTableCell[179])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionTableCell[179]

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. This action has the following direct descendants:

- TrvActionTableCellVAlign[186]
- TrvActionTableCellBorder[180]

1.3.8.30 TrvActionTableProperties

TrvActionTableProperties is the action for "Table | Table Properties" command.

Unit RichViewActions[54];

Syntax

```
TrvActionTableProperties = class(TrvActionTableCell[179])
```

Hierarchy

TObject
TPersistent

TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 208 action. But TrvActionItemProperties 208 edits properties of an item at the position of caret. For example, if a picture inside a table is selected, TrvActionItemProperties 208 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

1.3.8.30.1.1 TrvActionTableProperties.ActionInsertTable

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 170;
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 170 action on the same form/datamodule (if **UpdateAllInsertTableActions**=*False*), or to all TrvActionInsertTable 170 actions on the same form/datamodule (if **UpdateAllInsertTableActions**=*True*)

Default value

UpdateAllInsertTableActions = *True*

See also

- TRVActionItemProperties 208.ActionInsertTable 209

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 224;
```

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 159
- TrvActionItemProperties.GraphicFilter 210
- TrvActionItemProperties.BackgroundGraphicFilter 209

1.3.8.31 TrvActionTableRCBase

TrvActionTableRCBase is a base class for the actions for insertion and deletion of table rows and columns.

Unit RichViewActions 54;

Syntax

```
TrvActionTableRCBase = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

Description

This action does not introduce any new properties in addition to properties of TrvAction.

This action is not used directly. The following actions are inherited from it:

- TrvActionTableInsertRowsAbove 192
- TrvActionTableInsertRowsBelow 193
- TrvActionTableInsertColLeft 191
- TrvActionTableInsertColRight 192
- TrvActionTableDeleteRows 190
- TrvActionTableDeleteCols 189
- TrvActionTableSelectCols 197
- TrvActionTableSelectRows 197

1.3.8.32 TrvActionTableSelectCell

TrvActionTableSelectCell is the action for "Table | Select Cell" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSelectCell = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

Description

The action selects the edited table cell.

This action does not introduce any new properties in addition to properties of TrvAction.

See also:

- TrvActionTableSelectCols 197
- TrvActionTableSelectRows 197
- TrvActionTableSelectTable 198

1.3.8.33 TrvActionTableSelectCols

TrvActionTableSelectCols is the action for "Table | Select Columns" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSelectCols = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableSelectRows 197
- TrvActionTableSelectCell 196
- TrvActionTableSelectTable 198

1.3.8.34 TrvActionTableSelectRows

TrvActionTableSelectRows is the action for "Table | Select Rows" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSelectRows = class(TrvActionTableRCBase 196)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction

TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179
TrvActionTableRCBase 196

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 of TrvAction.

See also:

- TrvActionTableSelectCols 197
- TrvActionTableSelectCell 196
- TrvActionTableSelectTable 198

1.3.8.35 TrvActionTableSelectTable

TrvActionTableSelectTable is the action for "Table | Select Table" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSelectTable = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 197
- TrvActionTableSelectRows 197
- TrvActionTableSelectCell 196

1.3.8.36 TrvActionTableSort

TrvActionTableSort is the action for "Table | Sort" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSort = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 of TrvAction.

1.3.8.37 TrvActionTableSplit

TrvActionTableSplit is the action for "Table | Split Table" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSplit = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 of TrvAction.

1.3.8.38 TrvActionTableSplitCells

TrvActionTableSplitCells is the action for "Table | Split Cells" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableSplitCells = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 or rows and columns (optionally with preliminary merging).

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.8.39 TrvActionTableToText

TrvActionTableToText is the action for "Table | Convert to Text" command.

Unit RichViewActions 54;

Syntax

```
TrvActionTableToText = class(TrvActionTableCell 179)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202
TrvActionTableCell 179

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 of TrvAction.

1.3.9 Spelling check and thesaurus

1.3.9.1 TrvActionSpellingCheck

TrvActionSpellingCheck is the action for "Spell Check" command.

Unit RichViewActions[54];

Syntax

```
TrvActionSpellingCheck = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action is enabled a spelling check interface component is assigned to ControlPanel[214].SpellInterface[35].

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 properties of TrvAction.

1.3.9.2 TrvActionThesaurus

TrvActionThesaurus is the action for "Thesaurus" command.

Unit RichViewActions[54];

Syntax

```
TrvActionThesaurus = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action is enabled a spelling check interface component is assigned to ControlPanel[214].SpellInterface[35], and this component supports a thesaurus.

In this version, a thesaurus is provided only by TRVAAddictSpellInterface[229] component.

The action offers synonyms for the current word.

This action does not introduce any new properties in addition to properties of TrvAction.

1.3.10 Others

1.3.10.1 TrvAction

TrvAction is a base class for all RichViewActions working with TCustomRichViewEdit.

Unit RichViewActions 54;

Syntax

```
TrvAction = class(TrvCustomAction 213)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213

Description

This action is not used directly, but all RichViewActions working with TCustomRichViewEdit are inherited from it.

1.3.10.1.1 Properties

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 215.DefaultControl 29 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 137;
- TrvActionEditNote 164.

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 54;

Syntax

```
TrvActionBackground = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action changes background (Color, BackgroundBitmap, BackgroundStyle) and "padding" (LeftMargin, TopMargin, RightMargin, BottomMargin) properties of the target editor. "Padding" can be changed only if CanChangeMargins 203 = *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 215.OnMarginsChanged 43 (only if the action changed values of LeftMargin, TopMargin, RightMargin, BottomMargin properties of TCustomRichViewEdit)
- GetControlPanel 215.OnBackgroundChange 40

1.3.10.2.1 Properties

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 TSRichViewEdit 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 204
- GetControlPanel 215.OnBackgroundChange 40

1.3.10.2.2 Events

1.3.10.2.2.1 TrvActionBackground.OnChange

Occurs after the background is changed by this action.

```
property OnChange: TRVAEditEvent[249];
```

You can use either this event or GetControlPanel[215].OnBackgroundChange[40].

1.3.10.3 TrvActionColor

TrvActionColor is the action for "Background Color" command.

Unit RichViewActions[54];

Syntax

```
TrvActionCustomColor = class(TrvActionCustomColor[204])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]
TrvActionCustomColor[204]

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[215].OnBackgroundChange[40] event.

This action does not introduce any new properties and events in addition to properties and events of TrvActionCustomColor[204].

1.3.10.4 TrvActionCustomColor

TrvActionCustomColor is a base class of the actions for color changing.

Unit RichViewActions[54];

Syntax

```
TrvActionCustomColor = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action is not used directly. The following actions are inherited from it:

- TrvActionFontCustomColor 91
- TrvActionFontColor 90
- TrvActionFontBackColor 89
- TrvActionParaCustomColor 124
- TrvActionParaColor 122
- TrvActionColor 204

1.3.10.4.1 Properties

1.3.10.4.1.1 TrvActionCustomColor.CallerControl

Specifies the control (for example a toolbar button) which executed this action.

```
property CallerControl: TControl;
```

Assign this property if you use Delphi 4 or 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.

1.3.10.4.1.2 TrvActionCustomColor.Color

Specifies the color to apply.

```
property Color: TColor;
```

See the topic about UserInterface 205 property for details.

Default value for TrvActionCustomColor:

clNone

Default value for TrvActionFontColor 90:

clWindowText

1.3.10.4.1.3 TrvActionCustomColor.UserInterface

Defines the way how the user chooses a color.

```
type
  TrvaColorInterface = (rvacNone, rvacColorDialog, rvacAdvanced);

property UserInterface: TrvaColorInterface;
```

Value	Meaning
rvacNone	Color is not chosen by the user. When the action is executed, value specified in Color 205 property is applied.
rvacColorDialog	Color is chosen using TColorDialog. Value of Color 205 property before the execution is ignored. When the color is chosen, it is assigned to Color 205 property.
rvacAdvanced	Color is chosen using a special popup color-picker window. Value of Color 205 property before the execution is ignored. When the color is chosen, it is assigned to Color 205 property.

This action uses GetControlPanel 215.ColorDialog 28, if assigned. Otherwise it creates a temporal color dialog.

In *rvacAdvanced* mode, GetControlPanel[215].OnGetActionControlCoords[42] may be used to position a color-picker window.

Default value:

rvacAdvanced

1.3.10.4.2 Events

1.3.10.4.2.1 TrvActionCustomColor.OnShow ColorPicker

Occurs before the color-picker window is displayed.

```
property OnShowColorPicker: TNotifyEvent;
```

This event occurs if UserInterface[205] = *rvacAdvanced.*

1.3.10.4.2.2 TrvActionCustomColor.OnHideColorPicker

Occurs when the color-picker window is destroyed.

```
property OnHideColorPicker: TNotifyEvent;
```

This event occurs if UserInterface[205] = *rvacAdvanced.*

1.3.10.5 TrvActionEvent

TrvActionEvent is the action to perform custom operations.

Unit RichViewActions[54];

Syntax

```
TrvActionEvent = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

Description

This action calls OnExecute[206] and OnUpdate[207] 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 of TrvAction.

See also:

- TrvActionInsertText[162]

1.3.10.5.1 Events

1.3.10.5.1.1 TrvActionEvent.OnExecute

Occurs when the action is executed.

```
type
  TRVAEvent = procedure(Sender: TObject;
    Editor: TCustomRichViewEdit) of object;
```

```
property OnExecute: TRVAEvent;
```

1.3.10.5.1.2 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[214]=*True*, or GetControlPanel[215].ActionsEnabled[27]=*False*.

1.3.10.6 TrvActionFillColor

TrvActionFillColor is the action for "Fill Color" command.

Unit RichViewActions[54];

Syntax

```
TrvActionFillColor = class(TrvAction[202])
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction[213]
TrvAction[202]

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.

1.3.10.6.1 Properties

1.3.10.6.1.1 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*]

1.3.10.7 TrvActionHide

TrvActionHide switches visibility of the selected fragment.

Unit RichViewActions 54;

Syntax

```
TrvActionHide = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 of TrvAction.

See also

- TrvActionShowSpecialCharacters 212

1.3.10.8 TrvActionItemProperties

TrvActionItemProperties is the action for "Object Properties" command.

Unit RichViewActions 54;

Syntax

```
TrvActionItemProperties = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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[211] 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 *rvespImageFileName* 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[194]

1.3.10.8.1 Properties

1.3.10.8.1.1 TrvActionItemProperties.ActionInsertHLine

Specifies the "insert horizontal line" action for applying default properties to.

```
property ActionInsertHLine: TrvActionInsertHLine[144];
```

If this link is not defined, the first found TrvActionInsertHLine[144] action on the same form/datamodule is used.

If "Default" checkbox is checked, when properties are applied, they are assigned to properties of this action as well, so they become default properties for new horizontal lines.

See also:

- ActionInsertTable[209]

1.3.10.8.1.2 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[170];
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[170] action on the same form/datamodule (if **UpdateAllInsertTableActions**=*False*), or to all TrvActionInsertTable[170] actions on the same form/datamodule (if **UpdateAllInsertTableActions**=*True*)

Default value

UpdateAllInsertTableActions = *True*

See also:

- ActionInsertHLine[209]
- TrvActionTableProperties[194].ActionInsertTable[195]

1.3.10.8.1.3 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: TRVALocString[224];
```

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 159
- GraphicFilter 210
- TrvActionTableProperties.BackgroundGraphicFilter 195

1.3.10.8.1.4 TrvActionItemProperties.GraphicFilter

Determines the file masks (filters) available in the file opening dialog for pictures and "hot-pictures".

```
property GraphicFilter: TRVALocString 224;
```

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 159
- TrvActionItemProperties.BackgroundGraphicFilter 209
- TrvActionTableProperties.BackgroundGraphicFilter 195

1.3.10.8.1.5 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 *rvespImageFileName* item property (see the RichView help file, TRVExtraItemStrProperty 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 *rvsoUseItemImageFileNames* in TrvActionSave.SaveOptions 75 and TrvActionExport.SaveOptions 68).

Default value

False

See also:

- TCustomRichView.RTFReadProperties.StoreImagesFileNames
- HTML Import (for images inserted with HTML, paths to graphic files are always stored in *rvespImageFileName*)

1.3.10.8.2 Events

1.3.10.8.2.1 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 211 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*.

1.3.10.8.2.2 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 210

1.3.10.9 TrvActionRemoveHyperlinks

TrvActionRemoveHyperlinks is the action for "Remove Hyperlinks" command.

Unit RichViewActions 54;

Syntax

```
TrvActionRemoveHyperlinks = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

This action converts all hyperlinks in the selected fragment to a plain text. This action does the same work as TrvActionInsertHyperlink 145, if the user entered an empty link target.

This action is a wrapper for TrvActionInsertHyperlink 145 action referenced in ActionInsertHyperlink property.

1.3.10.9.1 Properties

1.3.10.9.1.1 TrvActionRemoveHyperlinks.ActionInsertHyperlink

Links this action to TrvActionInsertHyperlink 145 action.

```
property ActionInsertHyperlink: TrvActionInsertHyperlink 145;
```

If this link is not defined, the first found TrvActionInsertHyperlink 145 action on the same form/datamodule is used. If no TrvActionInsertHyperlink 145 action found, this action is disabled.

1.3.10.10 TrvActionRemovePageBreak

TrvActionRemovePageBreak is the action for "Remove Page Break" command.

Unit RichViewActions 54;

Syntax

```
TrvActionRemovePageBreak = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 of TrvAction.

See also:

- TrvActionInsertPageBreak 156

1.3.10.11 TrvActionShowSpecialCharacters

TrvActionShowSpecialCharacters is the action for "Non-printing Characters" command.

Unit RichViewActions 54;

Syntax

```
TrvActionShowSpecialCharacters = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 213 = 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 208

1.3.10.11.1 Properties

1.3.10.11.1.1 TrvActionShow SpecialCharacters.Show Checkpoints

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/excluded 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

1.3.10.12 TrvActionVAlign

TrvActionVAlign is the action for "Object Position" command.

Unit RichViewActions 54;

Syntax

```
TrvActionVAlign = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

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 of TrvAction.

1.3.10.13 TrvCustomAction

TrvCustomAction is a base class for all RichViewActions.

Unit RichViewActions 54;

Syntax (normal)

```
TrvCustomAction = class(TAction)
```

Syntax (if TNT Controls 239 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 214 links this action to a control panel defining additional properties for the action.
- Disabled 214 allows to disable this action.

Direct descendants:

- TrvAction 202
- TrvActionPageSetup 69

1.3.10.13.1 Properties

1.3.10.13.1.1 TrvCustomAction.Caption

Represents the caption of client controls and menu items.

If TNT Controls 239 are used, this property overrides the standard action's Caption property.

```
property Caption: WideString;
```

This property is defined only if TNT Controls 239 are used. It overrides the standard Caption property.

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 215

1.3.10.13.1.2 TrvCustomAction.ControlPanel

Links this action with a control panel.

```
property ControlPanel: TRVAControlPanel 26;
```

A control panel defines properties used for many actions.

If this property is not assigned (*nil*), the action uses MainRVAControlPanel 252.

See also:

- GetControlPanel 215.

1.3.10.13.1.3 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 215.ActionsEnabled 27

1.3.10.13.1.4 TrvCustomAction.Hint

Indicates the Help hint for client controls and menu items.

If TNT Controls 239 are used, this property overrides the standard action's Hint property.

```
property Hint: WideString;
```

This property is defined only if TNT Controls 239 are used. It overrides the standard Hint property.

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 214

1.3.10.13.2 Methods

1.3.10.13.2.1 TrvCustomAction.GetControlPanel

Returns a control panel used by this action.

```
function GetControlPanel: TRVAControlPanel 26;
```

If ControlPanel 214 is assigned, the method returns it. Otherwise it returns MainRVAControlPanel 252.

1.3.10.14 TrvCustomEditorAction

TrvCustomEditorAction is a base class for action displaying a non-modal window containing TRichViewEdit component.

Unit RichViewActions 54;

Syntax

```
TrvCustomEditorAction = class(TrvAction 202)
```

Hierarchy

TObject
TPersistent
TComponent
TBasicAction
TContainedAction
TCustomAction
TAction
TrvCustomAction 213
TrvAction 202

Description

When executed, this action shows Form 216 containing SubDocEditor 216.

This class is not used directly. The following actions are inherited from TrvCustomEditorAction:

- TrvActionEditNote 164

1.3.10.14.1 Properties

1.3.10.14.1.1 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 216
- OnShowing and OnHide 216
- SubDocEditor 216

1.3.10.14.1.2 TrvCustomEditorAction.SubDocEditor

Returns a TRichViewEdit control inserted in Form 216.

```
property SubDocEditor: TRichViewEdit;
```

This control is created and destroyed together with Form 216,

1.3.10.14.2 Events

1.3.10.14.2.1 TrvCustomEditorAction.OnFormCreate

Occurs after the Form 216 is created.

```
property OnFormCreate: TRVShowFormEvent 252;
```

See also:

- OnShowing, OnHide 216 (contains example)

1.3.10.14.2.2 TrvCustomEditorAction.OnShow ing, OnHide

Occurs before Form 216 is shown, and when it is hidden.

```
property OnShowing: TRVShowFormEvent 252;
property OnHide: TNotifyEvent;
```

Example

This example shows how to work with a form displayed by TrvActionEditNote 164.

Let the main form TfrmMain have:

- RichViewEdit: TRichViewEdit – the main editor;
- RVAControlPanel1: TRVAControlPanel 26;
- RVAPopupMenu1: TRVAPopupMenu 43;
- rvActionEditNote1: TrvActionEditNote;
- rvActionStyleInspector1: TrvActionStyleInspector 137;
- cmbFont: TRVFontComboBox 49;
- cmbFontSize: TRVFontSizeComboBox 50;
- cmbStyles: TRVStyleTemplateComboBox 51.

We want to insert Form 216 at the bottom of TfrmMain.

Initial property settings:

- RVAControlPanel1.DefaultControl 29 := RichViewEdit1;
- rvActionStyleInspector1.Control 202 := RichViewEdit1;
- cmbFont.Editor 48 := RichViewEdit1;
- cmbFontSize.Editor 48 := RichViewEdit1;
- cmbStyles.Editor 52 := RichViewEdit1

```
// OnFormCreate event
procedure TfrmMain.rvActionEditNoteFormCreate(Sender: TrvAction; 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
  RVAControlPanel1.DefaultControl := rvActionsResource.rvActionEditNote1.SubDocEditor;
  UpdateLinkedControls;
end;

// OnExit: making the main editor default
procedure TfrmMain.NoteEditorExit(Sender: TObject);
begin
  RVAControlPanel1.DefaultControl := RichViewEdit1;
  UpdateLinkedControls;
end;

procedure TfrmMain.UpdateLinkedControls;
begin
  rvActionStyleInspector1.Control := RVAControlPanel1.DefaultControl;
  cmbFont.Editor := RVAControlPanel1.DefaultControl;
  cmbFontSize.Editor := RVAControlPanel1.DefaultControl;
  cmbStyles.Editor := RVAControlPanel1.DefaultControl;
end;
```

Note: a note editor is special, it redirects many actions (for example, TrvActionNew[61], TrvActionSave[73], TrvActionPrint[70]) 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[32] 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[225] to localize them.

For changing UI language, use RVA_ChooseLanguage[224] function. The topic about this function contains an example.

Other localization functions are declared in RVALocalize[219] 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[253] 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_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_RegisterLanguage 222 to RVA_RegisterLanguage('Greek', Ελληνικά', GREEK_CHARSET, @Messages).
4. Remove the call of RVA_SwitchLanguage 222.
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 RichViewActions 15.

Procedures and functions declared in this unit

Procedures returning names of registered UI languages:

- RVA_EnumLanguages 220
- RVA_FillLanguageList 220

Functions returning properties of the current language:

- RVA_GetLanguageName 221
- RVA_GetCharset 220
- RVA_GetHelpFile 221

Functions returning a translated string:

- RVA_GetS, RVA_GetSH, RVA_GetPC 221
- RVA_TranslateUnits 222
- RVA_GetProgressMessage, RVA_GetPrintingMessage 222

Procedure for changing the current language:

- RVA_SwitchLanguage 222

Other procedures

The following procedures and functions are declared in other units, but related to localization:

- RVA_ChooseLanguage 224 (displays a dialog to choose a language name, changes the current language)
- RVA_LocalizeForm 225 (localizes all actions on the specified form/datamodule)
- RVALocalizeRuler 226 (localizes TRVRuler component)
- RVA_GetColorName 225 (returns the color name)

Additional information

See also:

- Localization of RichViewActions 218

1.4.1.1 Procedures and Functions

1.4.1.1.1 RVA_EnumLanguages procedure

This procedure allows to enumerate all UI languages available for RichViewActions.

Unit RVALocalize[219].

```
//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[32] or used as a parameter for RVA_SwitchLanguage[222] procedure.

See also:

- RVA_FillLanguageList[220]

1.4.1.1.2 RVA_FillLanguageList procedure

This procedure adds all UI language names available for RichViewActions in **sl**.

Unit RVALocalize[219].

```
procedure RVA_FillLanguageList(sl: TRVALocStrings[224]; 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[32] or used as a parameter for RVA_SwitchLanguage[222] procedure.

1.4.1.1.3 RVA_GetCharset function

This method returns the charset for the current UI language.

Unit RVALocalize[219].

```
function RVA_GetCharset(ControlPanel: TComponent=nil): TFontCharset;
```

If **ControlPanel** is a TRVAControlPanel[26] component, the function uses it. Otherwise, it uses MainRVAControlPanel[252].

The name of the current language is returned as ControlPanel.Language[32], or by RVA_GetLanguageName[221] function.

The current language can be changed by assigning a new value to ControlPanel.Language[32], or by calling RVA_SwitchLanguage[222] 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 4-2007.

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
```

1.4.1.1.4 RVA_GetHelpFile function

This method returns the help file name for the current UI language.

Unit RVALocalize[219].

```
function RVA_GetHelpFile(ControlPanel: TComponent): String;
```

If **ControlPanel** is a TRVAControlPanel[26] component.

The name of the current language is returned as **ControlPanel**.Language[32], or by RVA_GetLanguageName[221] function.

The current language can be changed by assigning a new value to **ControlPanel**.Language[32], or by calling RVA_SwitchLanguage[222] procedure.

See also:

- TRVAControlPanel.UseHelpFiles[36]

1.4.1.1.5 RVA_GetLanguageName function

This method returns the name of the current UI language.

Unit RVALocalize[219].

```
function RVA_GetLanguageName(ControlPanel: TComponent=nil): TRVALanguageName[224];
```

If **ControlPanel** is a TRVAControlPanel[26] component, the function uses it. Otherwise, it uses MainRVAControlPanel[252].

The same value is returned in ControlPanel.Language[32] property.

The current language can be changed by assigning a new value to ControlPanel.Language[32], or by calling RVA_SwitchLanguage[222] procedure.

1.4.1.1.6 RVA_GetS function

This method returns a localized string.

Unit RVALocalize[219].

```
type
  TRVAMessageID = (rvam_Empty, rvam_menu_File, rvam_menu_Edit, ...);
function RVA_GetS(MsgID: TRVAMessageID; ControlPanel: TComponent=nil): TRVALocString[224];
function RVA_GetSH(MsgID: TRVAMessageID; ControlPanel: TComponent=nil): TRVALocString[224];
function RVA_GetPC(MsgID: TRVAMessageID; ControlPanel: TComponent=nil): PChar // or WideS
```

If **ControlPanel** is a TRVAControlPanel[26] component, the functions use it. Otherwise, they use MainRVAControlPanel[252].

These functions return strings corresponding to MsgID, translated to the current language. The name of the current language is returned as ControlPanel.Language[32], or by RVA_GetLanguageName[221] function.

The current language can be changed by assigning a new value to ControlPanel.Language[32], or by calling RVA_SwitchLanguage[222] 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 239, these functions return WideString.

1.4.1.1.7 RVA_SwitchLanguage procedure

Changes the current language.

```
procedure RVA_SwitchLanguage(const LanguageName: TRVALanguageName;
  ControlPanel: TComponent=nil);
```

If **ControlPanel** is a TRVAControlPanel 26 component, the function uses it. Otherwise, it uses MainRVAControlPanel 252.

Assigning a new value to ControlPanel.Language 32 has the same effect.

To get a list of valid languages, use RVA_EnumLanguages 220 or RVA_FillLanguageList 220.

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 225 for localization of Captions and Hints of all actions on the given form/datamodule;
- RVALocalizeRuler 226 for localizations of hints in TRVRuler component;
- RVA_GetCharset 220 for changing Charsets of menus, hints and your forms;
- RVA_GetS 221 for localizing standard menu items in your menu (such as "File", "Edit"...).

See also:

- RVA_ChooseLanguage 224

1.4.1.1.8 RVA_TranslateUnits procedure

Translates units of measurement to the current language.

```
procedure RVA_TranslateUnits(SL: TStrings; ControlPanel: TComponent=nil);
```

If **ControlPanel** is a TRVAControlPanel 26 component, the function uses it. Otherwise, it uses MainRVAControlPanel 252.

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 32, or by RVA_GetLanguageName 221 function.

The current language can be changed by assigning a new value to ControlPanel.Language 32, or by calling RVA_SwitchLanguage 222 procedure.

1.4.1.1.9 Functions for progress messages

```
function RVA_GetProgressMessage(Operation: TRVLongOperation;
  ControlPanel: TComponent=nil): TRVALocString 224;
function RVA_GetPrintingMessage(PageCompleted: Integer; Step:TRVPrintingStep;
  ControlPanel: TComponent=nil): TRVALocString 224;
```

If **ControlPanel** is a TRVAControlPanel[26] component, the functions use it. Otherwise, they use MainRVAControlPanel[252].

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[32], or by RVA_GetLanguageName[221] function.

The current language can be changed by assigning a new value to ControlPanel.Language[32], or by calling RVA_SwitchLanguage[222] procedure.

Example:

This example assumes that Application.Hint is displayed in a status bar.

```
// TRVAControlPanel.OnDownload[41]
procedure TForm3.RVAControlPanel1Download(Sender: TrvAction;
  const Source: TRVUnicodeString);
begin
  if Source='' then
    Application.Hint := ''
  else
    Application.Hint := RVAFormat(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;
```

1.4.1.2 Types

1.4.1.2.1 TRVALanguageName

A type for UI language name.

Unit RVALocalize[219].

```
type
  TRVALanguageName = type String;
```

See also:

- TRVAControlPanel.Language[32]
- RVA_GetLanguageName[221]
- RVA_SwitchLanguage[222]

1.4.1.2.2 TRVALocString, TRVALocStrings

unit RVALocalize;

If TNT controls[239] are not used:

```
type
  TRVALocString = String;
  TRVALocStrings = TStrings;
```

If TNT controls are used:

```
type
  TRVALocString = WideString;
  TRVALocStrings = TTntStrings;
```

1.4.2 RichViewActions unit

1.4.2.1 Procedures and functions

1.4.2.1.1 RVA_ChooseLanguage function

Displays a dialog for choosing UI language and applies the chosen language.

Unit RichViewActions.

```
function RVA_ChooseLanguage: Boolean;
```

This function displays a dialog containing a list with all available UI languages.

If the user pressed "OK", this function calls RVA_SwitchLanguage[222] and returns *True*.

You may need to perform additional operations on the language changing, see RVA_SwitchLanguage[222] for details.

In Delphi 4-2007, this function displays language names in English.

In new versions of Delphi (or if TNT Controls[239] 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_ChooseLanguage then
    Localize;
end;

procedure TForm3.FormCreate(Sender: TObject);
begin
  ...
```

```
  Localize;
  ...
end;

procedure TForm3.Localize;
var Index: Integer;
begin
  // Fonts (for Delphi 4-2007)
  Font.Charset            := RVA_GetCharset(220);
  StatusBar1.Font.Charset := RVA_GetCharset(220);
  // Fonts (for Delphi 6-2007)
  Screen.HintFont.Charset := RVA_GetCharset(220);
  Screen.MenuFont.Charset := RVA_GetCharset(220);
  // Localizing all actions on rvActionsResource
  RVA_LocalizeForm(225)(rvActionsResource);
  // Localizing all actions on this form
  RVA_LocalizeForm(225)(Self);
  // Localizing ruler
  RVALocalizeRuler(226)(RVRuler1);
  // Localizing menus
  mitFile.Caption := RVA_GetS(221)(rvam_menu_File);
  mitEdit.Caption := RVA_GetS(221)(rvam_menu_Edit);
  // Styles
  rvActionsResource.rvActionStyleInspector1.UpdateInfo(138);
  RVStyleTemplateComboBox1.Localize(54);
  // Localizing combobox with units of measurement
  Index := cmbUnits.ItemIndex;
  RVA_TranslateUnits(222)(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): String;
```

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 (218).

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 (213)) owned by this form/datamodule.

The name of the current language is returned as Action.GetControlPanel (215).Language (32).

The current language can be changed by assigning a new value to TRVAControlPanel.Language[32], or by calling RVA_SwitchLanguage[222] procedure.

See also:

- RVALocalize unit[219].

1.4.3 Other units

1.4.3.1 GetAddictSpellLanguage, GetAddictThesLanguage functions

The functions convert a language of RichViewActions to languages used by Addict 3[233] components.

Unit RVAAddictLanguages.

```
function GetAddictSpellLanguage(const s: String): TSpellLanguageType;
function GetAddictThesLanguage(const s: String): TThesaurusLanguageType;
```

Example:

```
  RVAddictSpell31.UILanguage := GetAddictSpellLanguage(RVA_GetLanguageName[221]);
  RVThesaurus31.UILanguage := GetAddictThesLanguage(RVA_GetLanguageName[221]);
```

1.4.3.2 RVALocalizeRuler procedure

Localizes a TRVRuler component.

Unit RVALocRuler.

```
procedure RVALocalizeRuler(Ruler: TRVRuler);
```

The method assigns localized strings to hints displayed by **Ruler**.

See also:

- Localization of RichViewActions[218]
- RVALocalize unit[219]

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[26].

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[226]
- spelling checkers[228]
- color dialogs[232]

1.5.1 Download interfaces

Download interfaces allow using third-party components to download external images referred by RTF and HTML files.

To use a download interface component, assign it to DownloadInterface[31] property of TRVAControlPanel[26] component.

The following download interface components are provided:

Component	Requires
 TRVAIndyDownloadInterface 228	Indy: included in Delphi or ***http://www.indyproject.org***
 TRVAClDownloadInterface 227	Clever Internet Suite: **http://www.clevercomponents.com**

See also

- RichViewActions interfaces for third-party components 226

1.5.1.1 TRVAClDownloadInterface

Allows using TClHttp component (Clever Internet Suite) to download external images when loading files.

Unit RVAClDownloadInterface;

```
TRVAClDownloadInterface = class (TRVACustomDownloadInterface 227);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomDownloadInterface 227

Description

To use this component, assign it to DownloadInterface 31 property of TRVAControlPanel 26 component.

TRVAClDownloadInterface has property: **ClHttp**. Assign TClHttp component to this property, and actions will use it to download images. If this property is not assigned, a temporary TClHttp component is created each time when an action needs to download an image.

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 226

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 228
- TRVAClDownloadInterface 227

To use a download interface component, assign it to DownloadInterface 31 property of TRVAControlPanel 26 component.

See also:

- RichViewActions download interfaces 226

1.5.1.3 TRVAIndyDownloadInterface

Allows using TIdHttp component (Indy) to download external images when loading files.

Unit RVAIndyDownloadInterface;

```
TRVAIndyDownloadInterface = class (TRVACustomDownloadInterface 227);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomDownloadInterface 227

Description

To use this component, assign it to DownloadInterface 31 property of TRVAControlPanel 26 component.

TRVAIndyDownloadInterface has property: **IdHttp**. Assign TIdHttp component to this property, and actions will use it to download images. If this property is not assigned, a temporary TIdHttp component is created each time when an action needs to download an image.

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 226

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 35 property of TRVAControlPanel 26 component.

Then you can:

- use TrvActionSpellingCheck 201 action
- use TrvActionThesaurus 201 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-correction	Dialog Localization	Dialog Type	Win32/Win64	Parser is implemented by
TRVAAddictSpellInterface 229	Addict 3 or 4 233	Yes	Yes	Yes[1]	Classic/Word	Win32	Spell checker

TRVAASpellInterface 230	ASpell 234			Yes	Classic/Word	Win32	TRichView
TRVADXSpellInterface 231	ExpressSpellChecker 235		Yes	Yes[2]	Classic/Word	Win32+Win64	TRichView
TRVAHunSpellInterface 231	HunSpell 236			Yes	Classic/Word	Win32+Win64	TRichView
TRVAPolarSpellInterface 232	Polar SpellChecker 236		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 26.Language 32 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 226

1.5.2.1 TRVAAddictSpellInterface

Allows using Addict (by Addictive Software) 233 to correct spelling errors and to find synonyms.

Unit RVAAddictSpellInterface;

```
TRVAAddictSpellInterface = class (TRVACustomSpellInterface 230);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface 230

Description

To use this component, assign it to SpellInterface 35 property of TRVAControlPanel 26 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 201 will be disabled.

The TRichView installer installs this component automatically if Addict is already installed, and "build control" option is changed 233 for Addict packages.

Addict can be downloaded from **http://www.addictivesoftware.com**.

See also:

- RichViewActions spelling check interfaces 228

1.5.2.2 TRVAASpellInterface

Allows using ASpell[234] to correct spelling errors.

Unit RVAASpellInterface;

```
TRVAASpellInterface = class (TRVACustomSpellInterface[230]);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface[230]

Description

To use this component, assign it to SpellInterface[35] property of TRVAControlPanel[26] 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[228]

1.5.2.3 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[229] (to use Addict[233])
- TRVAASpellInterface[230] (to use ASpell[234])
- TRVADXSpellInterface[231] (to use ExpressSpellChecker[235])
- TRVAHunSpellInterface[231] (to use HunSpell[236])
- TRVAPolarSpellInterface[232] (to use Polar SpellChecker[236])

To use a spelling check interface component, assign it to SpellInterface[35] property of TRVAControlPanel[26] component.

Then you can:

- use TrvActionSpellingCheck[201] action
- use TrvActionThesaurus[201] 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.

Methods:

```
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;
```

See also:

- RichViewActions spelling check interfaces 228

1.5.2.4 TRVADXSpellInterface

Allows using ExpressSpellChecker 235 to correct spelling errors.

Unit RVADXSpellInterface;

```
TRVADXSpellInterface = class (TRVACustomSpellInterface 230);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface 230

Description

To use this component, assign it to SpellInterface 35 property of TRVAControlPanel 26 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 228

1.5.2.5 TRVAHunSpellInterface

Allows using HunSpell 236 to correct spelling errors.

Unit RVAASpellInterface;

```
TRVAHunSpellInterface = class (TRVACustomSpellInterface 230);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface 230

Description

To use this component, assign it to SpellInterface 35 property of TRVAControlPanel 26 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 228

1.5.2.6 TRVAPolarSpellInterface

Allows using Polar SpellChecker[236] to correct spelling errors.

Unit RVAPolarSpellInterface;

```
TRVAPolarSpellInterface = class (TRVACustomSpellInterface[230]);
```

Hierarchy

TObject
TPersistent
TComponent
TRVACustomSpellInterface[230]

Description

To use this component, assign it to SpellInterface[35] property of TRVAControlPanel[26] 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[228]

1.5.3 Color dialog interfaces

By default, RichViewActions use the standard TColorDialog. You can assign a specific TColorDialog component to ColorDialog[28] property of TRVAControlPanel[26], 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[28] property of TRVAControlPanel[26] component.

The following download interface components are provided:

Component	Requires	Dialog Component
TRVADXColorDialogInterface	DevExpress VCL Components	TdxColorDialog

See also

- RichViewActions interfaces for third-party components[226]

1.6 Third-party and additional tools

Components

The following additional components can be optionally used in RichViewActions.

Spell checkers:

- Addict 3 or 4[233]
- ASpell[234]
- ExpressSpellChecker[235]
- HunSpell[236]
- Polar Spell Checker ActiveX
-
- Indy[238] (freeware) or CleverComponents[237] (shareware) for downloading external images

- RvHtmlImporter[238] for inserting and pasting HTML files (freeware)
- RvHtmlViewImporter[239] for opening, inserting and pasting HTML files (freeware)
- RichViewXML[238] for opening and saving documents in XML format (freeware)
- Toolbar 2000[240] (free for non-commercial use), TBX[240], SpTBXLib[239] for providing alternative versions of TRVAPopupMenu[43]
- TNT Controls[239] for Unicode support in Delphi 4-2007.

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[241], or the following alternative toolbar images:

- GlyFX[241]
- Glyfz[242]
- Fugue Icons[242]
- FamFamFam Silk Icons[243]

1.6.1 Components

1.6.1.1 Spelling checkers

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.

All registered users of RichView Package get a 10% discount when registering Addict Professional + PlusPack (spell checker and thesaurus).

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 TRVAAddictSpellInterface[229] component, assign it to SpellInterface[35] property of TRVAControlPanel[26] component.

Create TRVAddictSpell3 component, assign it to AddictSpell property of this TRVAAddictSpellInterface[229] component.

Create TRVThesaurus3 component, assign it to Thesaurus property of this TRVAAddictSpellInterface[229] component.

Assign TrvActionSpellingCheck[201] and TrvActionThesaurus[201] to menu items or toolbar buttons.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface[230].

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 66 event of TrvActionOpen 63 action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject; Editor: TCustomRichViewEdit;
  const FileName: String; FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu 43 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 230 component, assign it to SpellInterface 35 property of TRVAControlPanel 26 component.

Create TRVASpell component, assign it to ASpell property of this TRVAASpellInterface 230 component.

Assign TrvActionSpellingCheck 201 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 *rvlspOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile 66 event of TrvActionOpen 63 action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject; Editor: TCustomRichViewEdit;
  const FileName: String; FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu 43 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 instals the component integrating ExpressSpellChecker in TRichView and RichViewActions automatically.

Using

Create TRVADXSpellInterface 231 component, assign it to SpellInterface 35 property of TRVAControlPanel 26 component.

Create TRvDxSpellChecker component, assign it to SpellChecker property of this TRVADXSpellInterface 231 component. Add the code to initialize the spell checker.

Assign TrvActionSpellingCheck 201 to a menu item or a toolbar button.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface 230.

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 43 component to RichViewEdit1.PopupMenu, however, TcxTRichViewEdit can display a popup menu with live spelling options even without it.

Live spelling check in TRichViewEdit (and TSRichViewEdit)

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://cgit.freedesktop.org/libreoffice/dictionaries/tree/***.

Using

Create TRVAHunSpellInterface[231] component, assign it to SpellInterface[35] property of TRVAControlPanel[26] component.

Create TRVHunSpell component, assign it to HunSpell property of this TRVAHunSpellInterface[231] component. Assign its DictDir, DllDir, DllName properties, if necessary. Load dictionaries (for example, by calling LoadAllDictionaries).

Assign TrvActionSpellingCheck[201] 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 *rvlspOnChange* to LiveSpellingMode property of TCustomRichViewEdit component.

If you want to start a live spelling check when the user loaded a document, use OnOpenFile[66] event of TrvActionOpen[63] action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject; Editor: TCustomRichViewEdit;
  const FileName: String; FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu[43] 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: D10_1

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 232 component, assign it to SpellInterface 35 property of TRVAControlPanel 26 component.

Create TRVPolarSpellChecker component, assign it to PolarSpellChecker property of this TRVAPolarSpellInterface 232 component.

Assign TrvActionSpellingCheck 201 to a menu item or a toolbar button.

If you need an auto-correction feature, see the information in the topic for TRVACustomSpellInterface 230.

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 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 66 event of TrvActionOpen 63 action:

```
procedure TForm1.RvActionOpen1OpenFile(Sender: TObject; Editor: TCustomRichViewEdit;
  const FileName: String; FileFormat: TrvFileOpenFilter; CustomFilterIndex: Integer);
begin
  Editor.StartLiveSpelling;
end;
```

Assign TRVAPopupMenu 43 component to RichViewEdit1.PopupMenu.

1.6.1.2 Downloaders

1.6.1.2.1 CleverComponents

HTML files (imported with RvHtmlImporter 238 or RvHtmlViewImporter 239) 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 CleverComponents (TClHTTP) component.

Visit ***http://www.clevercomponents.com*** for additional information.

CleverComponents can be used via TRVACIDownloadInterface[227] component.

1.6.1.2.2 Indy

HTML files (imported with RvHtmlImporter[238] or RvHtmlViewImporter[239]) 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[228] component.

1.6.1.3 File readers and writers

1.6.1.3.1 RichView XML

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[63]
- TrvActionSaveAs[77]
- TrvActionExport[66]
- TrvActionInsertFile[143]

RichViewXML is installed by TRichView Setup.

How to use

Place TRVAControlPanel[26] component on a form. If you have more than one TRVAControlPanel component in your application, link[214] it to actions.

Assign TRichViewXML component to TRVAControlPanel.XMLComponent[37].

Make sure that *ffiXML* is in Filter of TrvActionOpen[63], TrvActionInsertFile[143], *ffeXML* in Filter of TrvActionSaveAs[77], TrvActionExport[66].

You can change a file filter for XML files: see TRVAControlPanel.XMLFilter[38] property.

1.6.1.3.2 RvHtmlImporter

RVHTMLImporter allows importing HTML files in TCustomRichView components.

RichViewAction support RVHTMLImporter in the following actions:

- TrvActionInsertFile[143] (RVHTMLImporter replaces the MS Office HTML import converter in the file insertion dialog)
- TrvActionPasteSpecial[83]
- TrvActionPaste[82] (pasted if not RVF or non-empty RTF available in the Clipboard)

RVHTMLImporter is installed by TRichView Setup

How to use

Place TRVAControlPanel[26] component on a form. If you have more than one TRVAControlPanel component in your application, link[214] it to actions.

Assign TRvHtmlImporter component to TRVAControlPanel.HTMLComponent[33].

Make sure that *ffiHTML* is in Filter[144] property of TrvActionInsertFile[143] action.

Include *rvddHTML* in AcceptPasteFormats property of the editor.

See also:

- PasteHTML 244 procedure
- RvHtmlViewImporter 239

1.6.1.3.3 RvHtmlView Importer

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 63
- TrvActionSaveAs 77 (RVHTMLViewImporter is not used for saving, but HTML is added in the "Save As" action because it is added in the "Open" action)
- TrvActionInsertFile 143
- TrvActionPasteSpecial 83
- TrvActionPaste 82 (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 26 component on a form. If you have more than one TRVAControlPanel component in your application, link 214 it to actions.

Assign TRvHtmlViewImporter component to TRVAControlPanel.HTMLComponent 33.

Assign THTMLViewer component to HTMLViewer property of TRVHTMLViewImporter (make THTMLViewer hidden).

Make sure that *ffiHTML* is in Filter 144 property of TrvActionInsertFile 143 action.

Include *rvddHTML* in AcceptPasteFormats property of the editor.

See also:

- PasteHTML 244 procedure
- RvHtmlImporter 238

1.6.1.4 User interface

1.6.1.4.1 SpTBXLib

SpTBXLib is an expansion package for TB2K 240 components that adds Unicode support and skinning. You can download it from ***www.silverpointdevelopment.com/sptbxlib/***.

If activated, TRVASPTBXPopupMenu 43 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 15 (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[214] and Hints[215] properties of RichViewActions are changed to Unicode. TRVAPopupMenu[43], 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[240] components. You can download it from ***code.google.com/p/tbxlib/***

If activated, TRVATBXPopupMenu[43] 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[15] (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[240]
- SpTBXLib[239]

If activated, TRVATBPopupMenu[43] 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[15] (or rebuild it, if already installed).

1.6.2 Toolbar Images

1.6.2.1 TRichView Icons

TRichView Icons

This set of icons was created specially for RichViewActions.

It is used in this manual, and in ActionTest demos for Delphi 2009 and newer (including Ribbon demos).

The set includes 16x16 icons for almost all actions, and 32x32 icons for the main actions. Additionally, it includes several icons for using in the scrollbar areas of TSRichViewEdit (ScaleRichView).

For Delphi 2009+ only (because of semitransparent images).

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.

Additional information

Additional information is available on the support forum.

1.6.2.2 GlyFX toolbar images

GlyFX Images

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

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

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

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

1.7.1 RichViewActions unit

1.7.1.1 ConvertToUnicode procedure

Converts all text in **rv** to Unicode.

Unit RichViewActions

```
procedure ConvertToUnicode(rv: TCustomRichView);
```

This procedure changes Unicode properties of all **rv**.Style.TextStyles to *True*, and converts all non-Unicode text in **rv** to Unicode.

This method is called by PasteHTML[244].

1.7.1.2 GoToCheckpoint function

Moves the caret in **rv** to the checkpoint having the specified name.

Unit RichViewActions

```
function GoToCheckpoint(rv: TCustomRichViewEdit; const CPName: String): Boolean;
```

Returns *True* if the caret was successfully moved. Returns *False* if the checkpoint is not found.

This function is used in TrvActionInsertHyperlink.GoToLink 152 method.

1.7.1.3 PasteHTML procedure

Pastes HTML from the Clipboard to **rve** (if available)

Unit RichViewActions.

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

This procedure is available only if RvHtmlViewImporter 239 or RvHtmlImporter 238 is used.

The procedure does the following operations:

- loading HTML from the Clipboard to a temporal TRichView component;
- if **rve**.RTFReadProperties.UnicodeMode=*rvruOnlyUnicode*, converting a document in this temporal TRichView component to Unicode (see ConvertToUnicode 243);
- inserting the content of this temporal TRichView component in **rve**, in the caret position (via TMemoryStream in RVF format).

This procedure is used in TrvActionPaste 82 and TrvActionPasteSpecial 83 actions.

1.7.1.4 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.

1.7.1.5 RVA_EditorControlFunctionDef function

The function allows for actions inherited from TrvCustomEditAction 88 working with additional editor types.

Unit RichViewActions;

```
function RVA_EditorControlFunctionDef(Control: TControl;
    Command: TRVAEditorControlCommand): Boolean;
```

This function is a default value of RVA_EditorControlFunction 253 variable.

It allows working with:

- TCustomEdit (TEdit, TMemo, TRichEdit and others);
- TComboBox (having csDropDown or csSimple style).

1.7.1.6 RVA_ConvertToPixels, RVA_ConvertToTwips procedures

The procedures convert all properties measured in TRVAControlPanel 26.UnitsProgram 35 for all actions on the **Form**.

Unit RichViewActions.

```
procedure RVA_ConvertToPixels(Form: TComponent);
procedure RVA_ConvertToTwips(Form: TComponent);
```

Form is a form or datamodule. This procedure changes values of properties of all actions (inherited from TrvCustomAction[213]) owned by this form/datamodule.

RVA_ConvertToPixels must be called before assigning *rvstuPixels* to TRVAControlPanel[26].UnitsProgram[35].

RVA_ConvertToTwips must be called before assigning *rvstuTwips* to TRVAControlPanel[26].UnitsProgram[35].

These procedures do not change value of TRVAControlPanel[26].UnitsProgram[35].

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_ConvertToPixels**/**RVA_ConvertToTwips**, next, change UnitsProgram[35] for all these control panels).

1.7.1.7 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

This unit contains procedures and functions for working with TRibbon control.

- FillActionClientContents[245]
- RemoveEllipsis[245], RemoveEllipsisFromRibbon[245]
- LocalizeRibbonPage[245]
- SetRulerColors[246]

1.7.2.1 FillActionClientContents procedure

For all action items in Manager, copies Action.Hint to CommandProperties.Content;

Unit RVARibbonUtils[245].

```
procedure FillActionClientContents(Manager: TActionManager);
```

1.7.2.2 RemoveEllipsis, RemoveEllipsisFromRibbon procedures

Unit RVARibbonUtils[245].

```
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[245].

```
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 245.

```
procedure SetRulerColors(Ruler: TRVRuler; ColorMap: TCustomRibbonColorMap);
```

1.7.3 Other units

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 40 event.

1.7.3.2 MarkSubString functions

Marks all occurrences of **s** in **rv**.

Unit MarkSearch.

```
type
  TRVMarkStringOption = (
    rvmsoIgnoreCase, rvmsoWholeWords, rvmsoNormalize,
    rvmsoStripDiacritic, rvmsoIgnorePunctuation);
  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
rvmsoIgnoreCase	If set, character case is ignored when comparing
rvmsoWholeWords	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
rvmsoNormalize	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 'ト' and find 'ｱﾊﾟｰﾄ'.
rvmsoStripDiacritic	Works like *rvmsoNormalize*, and also all diacritic characters are ignored. For example, you can search for 'Ä' and find 'A' (and vice versa).
rvmsoIgnorePunctuation	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:

- *rvmsoIgnoreCase* for Unicode text works only on WinNT-based systems (WinNT4, Win2000, WinXP and newer)
- *rvmsoNormalize* and *rvmsoStripDiacritic* 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;
```

This procedure is called, for example, after loading HTML files with rvHtmlImporter[238].

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
rvccLowerCase	change all characters to lower case
rvccUpperCase	change all characters to upper case
rvccTitleWord	change the first character of each word to upper case, other characters to lower case.

For Unicode text items, this action requires NT-based systems (WinNT4/2000/XP/Vista). In Win9x/ME, Unicode text items are skipped.

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[79]

1.8 Types

1.8.1 TRVAEditEvent

This type is used for several events.

```
type
  TRVAEditEvent = procedure (Sender: TrvAction[202];
    Edit: TCustomRichViewEdit) of object;
```

1.8.2 TRVAHFInfo

TRVAHFInfo is the class used for storing properties of a header and a footer in RichViewActions[15].

Unit RichViewActions

Syntax

```
TRVAHFInfo = class(TPersistent);
```

Hierarchy

TObject
TPersistent

Description

This is the type of TRVAControlPanel.Header and Footer[31] 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[31]);
- &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[42] 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[34]) already has this event assigned, the temporal event handler calls the old handler before drawing text.

1.8.3 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
  TRVASpellStrings = TStrings;
  TRVASpellStringList = TStringList;
```

If TNT Controls are used[239], these strings are always Unicode:

```
type
  TRVASpellStrings = TTntStrings;
  TRVASpellStringList = TTntStringList;
```

1.8.4 TrvFileExportFilter, TrvFileSaveFilter

Types for Filter property of "Save As" and "Export" actions.

Unit RichViewActions;

```
type
  TrvFileExportFilter = (ffeRVF, ffeRTF, ffeXML, ffeTextANSI, ffeTextUnicode,
    ffeCustom, ffeHTMLCSS, ffeHTML, ffeDocX, 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.

ffeXML is declared only if RichViewXML 238 is used.

ffeHTMLCSS is included in TrvFileSaveFilter only if RvHtmlViewImporter 239 is used.

Value	File Type
ffeRVF	RichView Format (RVF)
ffeRTF	Rich Text Format (RTF)
ffeXML	Format of RichViewXML 238 (XML)
ffeTextANSI	ANSI text files (TXT)
ffeTextUnicode	Unicode (UTF-16) text files (TXT)
ffeCustom	Custom file formats defined in CustomFilter property of the corresponding action
ffeHTMLCSS	HTML files (HTM, HTML) saved by TCustomRichViewEdit.SaveHTMLEx method
ffeHTML	Simplified HTML files (HTM, HTML) saved by TCustomRichViewEdit.SaveHTML method
ffeDocX	Microsoft Word 2007+ Format (DocX)
ffeOfficeConverters	Formats supported by Microsoft Office text export converters

See also:

- TrvActionSaveAs.Filter 78
- TrvActionExport.Filter 67

1.8.5 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 238: allows loading, saving, and inserting XML files (an alternative to RVF);
- TrvHtmlImporter 238: basic component for inserting and pasting HTML files; it can be used to load and save them, but it is not recommended;
- TrvHtmlViewImporter 239: component for loading, saving, inserting and pasting HTML files.

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 37
- HTMLComponent 33

1.8.6 TrvFileImportFilter, TrvFileOpenFilter

Types for Filter property of "Open" and "Insert File" actions.

Unit RichViewActions;

```
type
  TrvFileImportFilter = (ffiRVF, ffiRTF, ffiXML, ffiTextANSI, ffiTextUnicode,
    ffiTextAuto, ffiCustom, 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 is declared only if RichViewXML 238 is used.

ffiHTML is declared only if RvHtmlImporter 238 or RvHtmlViewImporter 239 are used.

ffiHTML is included in TrvFileOpenFilter only if RvHtmlViewImporter 239 is used.

Value	File Type
ffiRVF	RichView Format (RVF)
ffiRTF	Rich Text Format (RTF)
ffiXML	Format of RichViewXML 238 (XML)
ffiTextANSI	ANSI text files (TXT)
ffiTextUnicode	Unicode (UTF-16) text files (TXT)
ffiTextAuto	ANSI or Unicode (UTF-16) text files (TXT), autodetected

ffiCustom	Custom file formats defined in CustomFilter property of the corresponding action
ffiHTML	HTML files (HTM, HTML)
ffiOfficeConverters	Formats supported by Microsoft Office text export converters

See also:

- TrvActionOpen.Filter 65
- TrvActionInsertFile.Filter 144

1.8.7 TRVShowFormEvent

This type is used in the following events:

- TrvActionCustomInfoWindow.OnShowing 138
- TrvCustomEditorAction.OnShowing 216
- TrvCustomEditorAction.OnFormCreate 216

```
type
  TRVShowFormEvent = procedure (Sender: TrvAction 202; Form: TForm) of object;
```

1.9 Global Variables

1.9.1 MainRVAControlPanel

The main control panel for RichViewActions.

Unit RichViewActions;

```
var MainRVAControlPanel: TRVAControlPanel 26;
```

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 38 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 88 (such as Cut, Copy, Paste) than Control 202 or GetControlPanel 215.DefaultControl 29 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 29=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[88] 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[244] function.

Command	The function must...
rvaeccIsEditorControl	... return True if Control is a supported editor control.
rvaeccHasSelection	... return True if the selection in Control is not empty
rvaeccCanPaste	... return True if Control can paste data from the Clipboard.
rvaeccCanPasteText	... return True if Control can paste text from the Clipboard.
rvaeccCanUndo	... return True if Control can undo the last editing operation
rvaeccCopy	... copy the selection from Control to the Clipboard
rvaeccCut	... cut the selection from Control to the Clipboard
rvaeccPaste	... paste data from the Clipboard into Control
rvaeccPasteText	... paste text from the Clipboard into Control
rvaeccUndo	... 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

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 26 component on the form. Change the following properties (in the code or in the Object Inspector):

```
RVAControlPanel1.RVFLocalizable 33 := False;
RVAControlPanel1.RVFFilter 33 := 'SuperApp Files (*.saf)|*.saf';
RVAControlPanel1.DefaultExt 30 := 'saf';
RVAControlPanel1.RVFormatTitle 33 := 'SuperApp Format';
```

Additionally, you can change names of files used by TrvActionStyleTemplates 134 to save and load style templates:

```
RVAControlPanel1.RVStylesFilter 34 := 'SuperApp Styles (*.sas)|*.sas';
RVAControlPanel1.RVStylesExt 34 := 'sas';
```

Now, if you want to localize these properties 218, you need to assign them yourself every time when the application language is changed.

For XML format, use XMLLocalizable 38 and XMLFilter 38 properties.

Index

- A -

ActionEditNote 168
TrvActionInsertNote 168
ActionFind 82, 86
TrvActionFindNext 82
TrvActionReplace 86
ActionFont 48
TCustomRVFontComboBox 48
ActionInsertHLine 209
TrvActionItemProperties 209
ActionInsertHyperlink 211
TrvActionRemoveHyperlinks 211
ActionInsertTable 195, 209
TrvActionItemProperties 209
TrvActionTableProperties 195
ActionList 45
TRVAPopupMenu 45
TRVASPTBXPopupMenu 45
TRVATBPopupMenu 45
TRVATBXPopupMenu 45
ActionNew 64
TrvActionOpen 64
ActionPageSetup 71
TrvActionPrintPreview 71
ActionParaBullets 131
TrvActionParaList 131
ActionParaNumbering 131
TrvActionParaList 131
ActionPrint 71
TrvActionPrintPreview 71
ActionReplace 81
TrvActionFind 81
ActionSave 61, 78
TrvActionCustomNew 61
TrvActionSaveAs 78
ActionSaveAs 74
TrvActionSave 74
ActionsEnabled 27
TRVAControlPanel 27
ActionStyleTemplates 136, 147
TrvActionAddStyleTemplate 136
TrvActionInsertHyperlink 147
Activate 38
TRVAControlPanel 38
AddColorNameToHints 28
TRVAControlPanel 28
AddDefaultCharset 49
TRVFontCharsetComboBox 49
AddFormat 84
TrvActionPasteSpecial 84
Addict 3 233
Alignment 125, 126
TrvActionParagraph 125, 126
AllowApplyingTo 207
TrvActionFillColor 207
AllowedCharacters 140
TrvActionBookmarks 140
AllowMultiple 193
TrvActionTableInsertRowsBelow 193
AllStylesImageIndex 52
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
ASpell 234
AutoAddHyperlinkStyleTemplate 147
TrvActionInsertHyperlink 147
AutoApplySymbolCharset 92
TrvActionFontEx 92
AutoDeleteUnusedStyles 28
TRVAControlPanel 28
AutoUpdateFileName 59
TrvActionCustomIO 59

- B -

BackColor 92, 147
TrvActionFontEx 92
TrvActionInsertHyperlink 147
Background 121
TrvActionParaBorder 121
BackgroundColor 158
TrvActionInsertPicture 158
BackgroundGraphicFilter 195, 209
TrvActionItemProperties 209
TrvActionTableProperties 195
BackgroundPicture 171
TrvActionInsertTable 171
BackgroundStyle 171
TrvActionInsertTable 171
BestWidth 171
TrvActionInsertTable 171
Border 121
TrvActionParaBorder 121
BorderColor 158, 171
TrvActionInsertPicture 158

BorderColor 158, 171
TrvActionInsertTable 171
BorderHSpacing 171
TrvActionInsertTable 171
BorderLightColor 172
TrvActionInsertTable 172
BorderStyle 172
TrvActionInsertTable 172
BorderVSpacing 172
TrvActionInsertTable 172
BorderWidth 158, 172
TrvActionInsertPicture 158
TrvActionInsertTable 172
BoxPosition 163
TrvActionCustomInsertSidenote 163
BoxProperties 164
TrvActionCustomInsertSidenote 164

- C -

CallerControl 205
TrvActionCustomColor 205
CanChangeMargins 203
TrvActionBackground 203
CanCloseDoc 76
TrvActionSave 76
Caption 214
TrvCustomAction 214
CellBorderColor 173
TrvActionInsertTable 173
CellBorderLightColor 173
TrvActionInsertTable 173
CellBorderStyle 173
TrvActionInsertTable 173
CellBorderWidth 173
TrvActionInsertTable 173
CellHPadding 173
TrvActionInsertTable 173
CellHSpacing 174
TrvActionInsertTable 174
CellPadding 173
TrvActionInsertTable 173
CellVPadding 173
TrvActionInsertTable 173
CellVSpacing 174
TrvActionInsertTable 174
CharCode 161
TrvActionInsertSymbol 161
CharScale 93
TrvActionFontEx 93
CharSpacing 93
TrvActionFontEx 93
ClearFormatImageIndex 52
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
CleverComponents 237
CloseDialog 81, 87
TrvActionFind 81
TrvActionReplace 87
ColBandSize 174
TrvActionInsertTable 174
ColCount 174
TrvActionInsertTable 174
Color 145, 147, 175, 205
TrvActionCustomColor 205
TrvActionInsertHLine 145
TrvActionInsertHyperlink 147
TrvActionInsertTable 175
ColorDialog 28
TRVAControlPanel 28
ColorDialogInterface 28
TRVAControlPanel 28
Control 202
TrvAction 202
ControlPanel 52, 214
TrvCustomAction 214
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
ConvertToUnicode 243
CreateDirectoryForHTMLImages 67, 74
TrvActionExport 67
TrvActionSave 74
Cursor 147
TrvActionInsertHyperlink 147
CustomFilter 58, 64
TrvActionCustomFileIO 58
TrvActionOpen 64

- D -

DefaultCharsetCaption 49
TRVFontCharsetComboBox 49
DefaultColor 28
TRVAControlPanel 28
DefaultControl 29
TRVAControlPanel 29
DefaultCustomFilterIndex 29
TRVAControlPanel 29
DefaultDocParameters 29
TRVAControlPanel 29

DefaultExt 30, 158
TRVAControlPanel 30
TrvActionInsertPicture 158
DefaultFileFormat 30
TRVAControlPanel 30
DefaultFileName 30
TRVAControlPanel 30
DefaultMargin 30
TRVAControlPanel 30
DeleteAllTabs 125
TrvActionParagraph 125
DetectURL 150
TrvActionInsertHyperlink 150
DialogFontName 31
TRVAControlPanel 31
DialogFontSize 31
TRVAControlPanel 31
DialogTitle 59, 65
TrvActionCustomIO 59
TrvActionOpen 65
Disabled 214
TrvCustomAction 214
DisableWhenUnmodified 74
TrvActionSave 74
Documents 74
TrvActionSave 74
DoImportPicture 39
TRVAControlPanel 39
DoneImportPictures 39
TRVAControlPanel 39
DownloadInterface 19, 31
TRVAControlPanel 31
DropDownWidth 50
TRVFontComboBox 50

- E -

Editor 48, 52
TCustomRVFontComboBox 48
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
EncodeTarget 151
TrvActionInsertHyperlink 151
EvenColumnsColor 178
TrvActionInsertTable 178
EvenRowsColor 178
TrvActionInsertTable 178
ExcludeLabel 141
TrvActionInsertCaption 141
ExportToFile 68
TrvActionExport 68
Expression 143
TrvActionInsertEquation 143
ExpressSpellChecker 235

- F -

FamFamFam Silk Icons toolbar images 243
FileName 60
TrvActionCustomIO 60
FileTitle 67, 75
TrvActionExport 67
TrvActionSave 75
FillActionClientContents 245
Filter 65, 67, 78, 144, 159
TrvActionExport 67
TrvActionInsertFile 144
TrvActionInsertPicture 159
TrvActionOpen 65
TrvActionSaveAs 78
FindDoc 76
TrvActionSave 76
FirstColumnColor 178
TrvActionInsertTable 178
FirstIndent 125
TrvActionParagraph 125
FirstPageNumber 31
TRVAControlPanel 31
Font 98, 168
TrvActionFonts 98
TrvActionInsertNote 168
FontImageIndex 137
TrvActionStyleInspector 137
FontName 49, 51, 161
TrvActionInsertSymbol 161
TRVFontCharsetComboBox 49
TRVFontSizeComboBox 51
FontSizeDouble 93
TrvActionFontEx 93
FontStyleEx 94
TrvActionFontEx 94
Footer 31
TRVAControlPanel 31
Form 216
TrvCustomEditorAction 216
Fugue Icons toolbar images 242

- G -

GetAddictSpellLanguage 226

GetAddictThesLanguage 226
GetControlPanel 215
  TrvCustomAction 215
GetHyperlinkStyleNo 151
  TrvActionInsertHyperlink 151
GetNormalStyleNo 152
  TrvActionInsertHyperlink 152
GlyFX toolbar images 241
Glyfz toolbar images 242
GoToCheckpoint 244
GoToLink 152
  TrvActionInsertHyperlink 152
GraphicFilter 210
  TrvActionItemProperties 210

- H -

Header 31
  TRVAControlPanel 31
HeadingRowColor 178
  TrvActionInsertTable 178
HeadingRowCount 175
  TrvActionInsertTable 175
Hint 215
  TrvCustomAction 215
HOutermostRule 175
  TrvActionInsertTable 175
HoverBackColor 148
  TrvActionInsertHyperlink 148
HoverColor 148
  TrvActionInsertHyperlink 148
HoverEffects 148
  TrvActionInsertHyperlink 148
HoverUnderlineColor 148
  TrvActionInsertHyperlink 148
How to
  change RVF format name 254
How to change RVF format name 254
HRuleColor 175
  TrvActionInsertTable 175
HRuleWidth 175
  TrvActionInsertTable 175
HTMLComponent 33
  TRVAControlPanel 33
HunSpell 236

- I -

If 109
ImageFileName 203
  TrvActionBackground 203
ImagePrefix 68, 75
  TrvActionExport 68
  TrvActionSave 75
Images 53, 135, 137
  TrvActionStyleInspector 137
  TrvActionStyleTemplates 135
  TRVStyleTemplateComboBox 53
  TRVStyleTemplateListBox 53
IndentMax 117
  TrvActionIndentInc 117
IndentStep 115, 116, 130
  TrvActionCustomParaListSwitcher 115
  TrvActionIndent 116
  TrvActionParaList 130
Indy 238
InitialDir 60, 65
  TrvActionCustomIO 60
  TrvActionOpen 65
InitImportPictures 39
  TRVAControlPanel 39
InsertAbove 141
  TrvActionInsertCaption 141
InsertFile 144
  TrvActionInsertFile 144
Installing RichViewActions 15
Interfaces for downloader third-party components in RichViewActions 226
Interfaces for spell checker third-party components in RichViewActions 228
Interfaces for third-party color-dialog components in RichViewActions 232
Interfaces for third-party components in RichViewActions 226
ItemText 176
  TrvActionInsertTable 176

- J -

JumpToNextNote 165
  TrvActionEditNote 165

- K -

KeepLinesTogether 126
  TrvActionParagraph 126
KeepWithNext 126
  TrvActionParagraph 126

- L -

Language 32
TRVAControlPanel 32
LastColumnColor 178
TrvActionInsertTable 178
LastFilterIndex 65
TrvActionOpen 65
LastLineAlignment 109, 126
TrvActionAlignCustomJustify 109
TrvActionParagraph 126
LastRowColor 178
TrvActionInsertTable 178
LeftIndent 126
TrvActionParagraph 126
LineSpacing 127
TrvActionParagraph 127
LineSpacingType 127
TrvActionParagraph 127
ListLevels 115
TrvActionCustomParaListSwitcher 115
LoadCSV 246
LoadFile 65
TrvActionOpen 65
Localization of RichViewActions 218
Localize 54
TRVStyleTemplateComboBox 54
TRVStyleTemplateListBox 54
LocalizeRibbonPage 245
LostFormatWarning 75
TrvActionSave 75

- M -

MainRVAControlPanel 252
MarginsUnits 69
TrvActionPageSetup 69
MarkSubStringA 246
MarkSubStringW 246
MaxImageSize 159
TrvActionInsertPicture 159
Maximized 72
TrvActionPrintPreview 72
MaxSize 96
TrvActionFontGrow 96
TrvActionFontGrowOnePoint 96
MaxSuggestionsCount 45
TRVAPopupMenu 45
TRVASPTBXPopupMenu 45
TRVATBPopupMenu 45
TRVATBXPopupMenu 45
MetafileCompatibility 32
TRVAControlPanel 32
MinSize 99, 101
TrvActionFontShrink 99
TrvActionFontShrinkOnePoint 101

- N -

NormalizeRichView 247
NumberType 142, 155
TrvActionInsertCustomPageNumber 142
TrvActionInsertNumSequence 155

- O -

OddColumnsColor 178
TrvActionInsertTable 178
OddRowsColor 178
TrvActionInsertTable 178
OnAddStyle 39
TRVAControlPanel 39
OnApplyHyperlinkToItem 153
TrvActionInsertHyperlink 153
OnBackgroundChange 40
TRVAControlPanel 40
OnCanApply 210
TrvActionItemProperties 210
OnCanPaste 84
TrvActionPasteSpecial 84
OnChange 70, 204
TrvActionBackground 204
TrvActionPageSetup 70
OnClickAllStyles 54
TRVStyleTemplateComboBox 54
TRVStyleTemplateListBox 54
OnCloseTableSizeDialog 178
TrvActionInsertTable 178
OnCreateForm 40
TRVAControlPanel 40
OnCustomFileOperation 40
TRVAControlPanel 40
OnCustomItemPropertiesDialog 211
TrvActionItemProperties 211
OnCustomPaste 85
TrvActionPasteSpecial 85
OnDocumentFileChange 76
TrvActionSave 76
OnDownload 41

OnDownload 41
TRVAControlPanel 41
OnExecute 206
TrvActionEvent 206
OnFormCreate 216
TrvCustomEditorAction 216
OnGetActionControlCoords 42
TRVAControlPanel 42
OnGetHeaderFooterCode 42
TRVAControlPanel 42
OnGetHyperlinkTargetFromItem 153
TrvActionInsertHyperlink 153
OnGetPreviewFormClass 72
TrvActionPrintPreview 72
OnHide 216
TrvCustomEditorAction 216
OnHideColorPicker 206
TrvActionCustomColor 206
OnHyperlinkForm 154
TrvActionInsertHyperlink 154
OnInserting 160, 178
TrvActionInsertPicture 160
TrvActionInsertTable 178
OnInsertText 162
TrvActionInsertText 162
OnLiveSpellAdd 45
TRVAPopupMenu 45
TRVASPTBXPopupMenu 45
TRVATBPopupMenu 45
TRVATBXPopupMenu 45
OnLiveSpellGetSuggestions 46
TRVAPopupMenu 46
TRVASPTBXPopupMenu 46
TRVATBPopupMenu 46
TRVATBXPopupMenu 46
OnLiveSpellIgnoreAll 46
TRVAPopupMenu 46
TRVASPTBXPopupMenu 46
TRVATBPopupMenu 46
TRVATBXPopupMenu 46
OnLiveSpellWordReplace 47
TRVAPopupMenu 47
TRVASPTBXPopupMenu 47
TRVATBPopupMenu 47
TRVATBXPopupMenu 47
OnMarginsChanged 43
TRVAControlPanel 43
OnNew 61
TrvActionCustomNew 61
OnOpenFile 66
TrvActionOpen 66
OnReplaceAllEnd 87
TrvActionReplace 87
OnReplaceAllStart 87
TrvActionReplace 87
OnReplacing 87
TrvActionReplace 87
OnSaving 77
TrvActionSave 77
OnShowColorPicker 206
TrvActionCustomColor 206
OnShowing 85, 138, 216
TrvActionCustomInfoWindow 138
TrvActionPasteSpecial 85
TrvCustomEditorAction 216
OnStyleNeeded 43
TRVAControlPanel 43
OnUpdate 207
TrvActionEvent 207
OuterHSpacing 159
TrvActionInsertPicture 159
OuterVSpacing 159
TrvActionInsertPicture 159
OutlineLevel 127
TrvActionParagraph 127

- P -

Padding 123
TrvActionParaColorAndPadding 123
ParaImageIndex 137
TrvActionStyleInspector 137
ParaStyleImageIndex 52, 136
TrvActionStyleTemplates 136
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
ParaTextStyleImageIndex 52, 136
TrvActionStyleTemplates 136
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
PasteHTML 244
Percent 100
TrvActionFontShrinkGrow 100
PixelBorders 32
TRVAControlPanel 32
Polar SpellChecker ActiveX 236
Preview 50
TRVFontComboBox 50
PreviewInList 94
TrvActionFontEx 94

Protection 161
TrvActionInsertSymbol 161

- R -

RefStyleTemplateName 169
TrvActionInsertSidenote 169
RemoveEllipsis 245
RemoveEllipsisFromRibbon 245
Reset 63
TrvActionNew 63
RichViewActions
downloading images 237, 238
FamFamFam Silk Icons toolbar images 243
Fugue Icons toolbar images 242
GlyFX toolbar images 241
Glyfz toolbar images 242
history 19
inserting and pasting HTML 238
installing 15
interfaces for downloader third-party components 226
interfaces for spell checker third-party components 228
interfaces for third-party color-dialog components 232
interfaces for third-party components 226
localizing 218
opening-inserting-saving HTML 239
opening-inserting-saving XML 238
Style Templates 18
Third-party tools 232
TNT Controls 239
TRichView toolbar images 241
using 15
using Addict 3 components 233
using ASpell 234
using ExpressSpellChecker 235
using HunSpell 236
using Polar SpellChecker 236
using SpTBXLib 239
using TBX 240
using Toolbar 2000 240
RichViewActions overview 15
RichViewActions.Overview 15
RichViewXML 238
RightIndent 127
TrvActionParagraph 127
RowBandSize 174
TrvActionInsertTable 174
RowCount 176
TrvActionInsertTable 176
RowsKeepTogether 176
TrvActionInsertTable 176
RowsVAlign 176
TrvActionInsertTable 176
RVA_ChooseLanguage 224
RVA_ChooseStyle 244
RVA_ConvertToPixels 244
RVA_ConvertToTwips 244
RVA_EditForceDefControl 252
RVA_EditorControlFunction 253
RVA_EditorControlFunctionDef 244
RVA_EnumLanguages 220
RVA_FillLanguageList 220
RVA_GetCharset 220
RVA_GetColorName 225
RVA_GetHelpFile 221
RVA_GetLanguageName 221
RVA_GetPC 221
RVA_GetPrintingMessage 222
RVA_GetProgressMessage 222
RVA_GetS 221
RVA_GetSH 221
RVA_LocalizeForm 225
RVA_SwitchLanguage 222
RVA_TranslateUnits 222
RVAAddictLanguages unit 226
RVALocalize 219
RVALocalizeRuler 226
RVALocRuler unit 226
RVARibbonUtils unit 245
RVChangeCharCase 248
RVCharCase 248
rvcsv unit 246
RVFFilter 33
TRVAControlPanel 33
RVFLocalizable 33
TRVAControlPanel 33
RVFormatTitle 33
TRVAControlPanel 33
RVGetCharCase 248
RvHtmlHelp 245
RVHTMLImporter 238
RVHTMLViewImporter 239
RVNormalize 247
RVPrint 34
TRVAControlPanel 34
RVStylesExt 34
TRVAControlPanel 34

RVStylesFilter 34
TRVAControlPanel 34

- S -

SaveOptions 68, 75
TrvActionExport 68
TrvActionSave 75
ScrollToCenter 140
TrvActionBookmarks 140
SearchScope 34
TRVAControlPanel 34
SeqName 155
TrvActionInsertNumSequence 155
SetRulerColors 246
SetTags 152
TrvActionInsertHyperlink 152
SetToPictures 149
TrvActionInsertHyperlink 149
ShowAllStyles 53
TRVStyleTemplateComboBox 53
TRVStyleTemplateListBox 53
ShowCheckpoints 213
TrvActionShowSpecialCharacters 213
ShowClearFormat 53
TRVStyleTemplateComboBox 53
TRVStyleTemplateListBox 53
ShowReplaceAllSummary 86
TrvActionReplace 86
ShowSoftPageBreaks 35
TRVAControlPanel 35
ShowTableSizeDialog 178
TrvActionInsertTable 178
ShowUntranslatedControls 253
SmartHeadings 53
TRVStyleTemplateComboBox 53
TRVStyleTemplateListBox 53
SpaceAfter 128
TrvActionParagraph 128
SpaceBefore 128
TrvActionParagraph 128
SpaceFiller 149
TrvActionInsertHyperlink 149
Spacing 159
TrvActionInsertPicture 159
SpellInterface 35
TRVAControlPanel 35
SpTBXLib 239
StandardStyleTemplates 135
TrvActionStyleTemplates 135
StoreFileNameInItemName 84, 160
TrvActionInsertPicture 160
TrvActionPasteSpecial 84
StoreImageFileName 210
TrvActionItemProperties 210
Style 145, 149
TrvActionInsertHLine 145
TrvActionInsertHyperlink 149
Style Templates in RichViewActions 18
StyleTemplateName 149, 164
TrvActionCustomInsertSidenote 164
TrvActionInsertHyperlink 149
StyleTemplates 62
TrvActionNew 62
SubDocEditor 216
TrvCustomEditorAction 216
SubSuperScriptType 94
TrvActionFontEx 94
SuppressNextErrorMessage 76
TrvActionSave 76
SymbolPreviewString 50
TRVFontComboBox 50

- T -

TableGridStyle 35
TRVAControlPanel 35
TableOptions 176
TrvActionInsertTable 176
TablePrintOptions 177
TrvActionInsertTable 177
Tabs 128
TrvActionParagraph 128
TabsToDelete 129
TrvActionParagraph 129
TBX 240
TCustomRVFontComboBox 47
TCustomRVFontComboBox.ActionFont 48
TCustomRVFontComboBox.Editor 48
TerminateHyperlink 153
TrvActionInsertHyperlink 153
TextStyleImageIndex 52, 136
TrvActionStyleTemplates 136
TRVStyleTemplateComboBox 52
TRVStyleTemplateListBox 52
Title 70, 73
TrvActionPrint 70
TrvActionQuickPrint 73
TNT Controls 239
Toolbar 2000 240

TRichView toolbar images 241
TRVA2LiveSpellAddEvent type 45
TRVA2LiveSpellGetSuggestionsEvent type 46
TRVA2LiveSpellIgnoreAllEvent 46
TRVA2LiveSpellReplaceEvent type 47
TRVAAddictSpellInterface 229
TRVAASpellInterface 230
TRVABoxPosition 163
TRVABoxProperties 164
TRVACanPasteEvent 84
TRVAClDownloadInterface 227
TrvaColorInterface 205
TRVAControlPanel 26
TRVAControlPanel.ActionsEnabled 27
TRVAControlPanel.Activate 38
TRVAControlPanel.AddColorNameToHints 28
TRVAControlPanel.AutoDeleteUnusedStyles 28
TRVAControlPanel.ColorDialog 28
TRVAControlPanel.ColorDialogInterface 28
TRVAControlPanel.DefaultColor 28
TRVAControlPanel.DefaultControl 29
TRVAControlPanel.DefaultCustomFilterIndex 29
TRVAControlPanel.DefaultDocParameters 29
TRVAControlPanel.DefaultExt 30
TRVAControlPanel.DefaultFileFormat 30
TRVAControlPanel.DefaultFileName 30
TRVAControlPanel.DefaultMargin 30
TRVAControlPanel.DialogFontName 31
TRVAControlPanel.DialogFontSize 31
TRVAControlPanel.DoImportPicture 39
TRVAControlPanel.DoneImportPictures 39
TRVAControlPanel.DownloadInterface 31
TRVAControlPanel.FirstPageNumber 31
TRVAControlPanel.Footer 31
TRVAControlPanel.Header 31
TRVAControlPanel.HTMLComponent 33
TRVAControlPanel.InitImportPictures 39
TRVAControlPanel.Language 32
TRVAControlPanel.MetafileCompatibility 32
TRVAControlPanel.OnAddStyle 39
TRVAControlPanel.OnBackgroundChange 40
TRVAControlPanel.OnCreateForm 40
TRVAControlPanel.OnCustomFileOperation 40
TRVAControlPanel.OnDownload 41
TRVAControlPanel.OnGetActionControlCoords 42
TRVAControlPanel.OnGetHeaderFooterCode 42
TRVAControlPanel.OnMarginsChanged 43
TRVAControlPanel.OnStyleNeeded 43
TRVAControlPanel.PixelBorders 32
TRVAControlPanel.RVFFilter 33
TRVAControlPanel.RVFLocalizable 33
TRVAControlPanel.RVFormatTitle 33
TRVAControlPanel.RVPrint 34
TRVAControlPanel.RVStylesExt 34
TRVAControlPanel.RVStylesFilter 34
TRVAControlPanel.SearchScope 34
TRVAControlPanel.ShowSoftPageBreaks 35
TRVAControlPanel.SpellInterface 35
TRVAControlPanel.TableGridStyle 35
TRVAControlPanel.UnitsDisplay 35
TRVAControlPanel.UnitsProgram 35
TRVAControlPanel.UseHelpFiles 36
TRVAControlPanel.UserInterface 36
TRVAControlPanel.UseTextCodePageDialog 37
TRVAControlPanel.UseXPThemes 37
TRVAControlPanel.XMLComponent 37
TRVAControlPanel.XMLFilter 38
TRVAControlPanel.XMLLocalizable 38
TrvAction 202
TrvAction.Control 202
TrvActionAddStyleTemplate 136
TrvActionAddStyleTemplate.ActionStyleTemplates 136
TrvActionAlignCenter 108
TrvActionAlignCustomJustify 108
TrvActionAlignCustomJustify.LastLineAlignment 109
TrvActionAlignCustomJustify.UseLastLineAlignment 109
TrvActionAlignDistribute 109
TrvActionAlignJustify 110
TrvActionAlignLeft 110
TrvActionAlignment 111
TrvActionAlignRight 111
TrvActionBackground 202
TrvActionBackground.CanChangeMargins 203
TrvActionBackground.ImageFileName 203
TrvActionBackground.OnChange 204
TrvActionBookmarks 139
TrvActionBookmarks.AllowedCharacters 140
TrvActionBookmarks.ScrollToCenter 140
TrvActionCharCase 79
TrvActionClearBoth 112
TrvActionClearFormat 138
TrvActionClearLeft 112
TrvActionClearNone 113
TrvActionClearRight 113
TrvActionClearTextFlow 114
TrvActionClearTextFormat 139

TrvActionColor 204
TrvActionCopy 79
TrvActionCustomColor 204
TrvActionCustomColor.CallerControl 205
TrvActionCustomColor.Color 205
TrvActionCustomColor.OnHideColorPicker 206
TrvActionCustomColor.OnShowColorPicker 206
TrvActionCustomColor.UserInterface 205
TrvActionCustomFileIO 58
TrvActionCustomFileIO.CustomFilter 58
TrvActionCustomInfoWindow 137
TrvActionCustomInfoWindow.OnShowing 138
TrvActionCustomInsertSidenote 162
TrvActionCustomInsertSidenote.BoxPosition 163
TrvActionCustomInsertSidenote.BoxProperties 164
TrvActionCustomInsertSidenote.StyleTemplateName 164
TrvActionCustomIO 59
TrvActionCustomIO.AutoUpdateFileName 59
TrvActionCustomIO.DialogTitle 59
TrvActionCustomIO.FileName 60
TrvActionCustomIO.InitialDir 60
TrvActionCustomNew 60
TrvActionCustomNew.ActionSave 61
TrvActionCustomNew.OnNew 61
TrvActionCustomParaListSwitcher 114
TrvActionCustomParaListSwitcher.IndentStep 115
TrvActionCustomParaListSwitcher.ListLevels 115
TrvActionCut 80
TrvActionEditNote 164
TrvActionEditNote.JumpToNextNote 165
TrvActionEvent 206
TrvActionEvent.OnExecute 206
TrvActionEvent.OnUpdate 207
TrvActionExport 66
TrvActionExport.CreateDirectoryForHTMLImages 67
TrvActionExport.ExportToFile 68
TrvActionExport.FileTitle 67
TrvActionExport.Filter 67
TrvActionExport.ImagePrefix 68
TrvActionExport.SaveOptions 68
TrvActionFillColor 207
TrvActionFillColor.AllowApplyingTo 207
TrvActionFind 80
TrvActionFind.ActionReplace 81
TrvActionFind.CloseDialog 81
TrvActionFindNext 81
TrvActionFindNext.ActionFind 82
TrvActionFontAllCaps 89
TrvActionFontBackColor 89
TrvActionFontBold 90
TrvActionFontColor 90
TrvActionFontCustomColor 91
TrvActionFontEx 91
TrvActionFontEx.AutoApplySymbolCharset 92
TrvActionFontEx.BackColor 92
TrvActionFontEx.CharScale 93
TrvActionFontEx.CharSpacing 93
TrvActionFontEx.FontSizeDouble 93
TrvActionFontEx.FontStyleEx 94
TrvActionFontEx.PreviewInList 94
TrvActionFontEx.SubSuperScriptType 94
TrvActionFontEx.UnderlineColor 94
TrvActionFontEx.UnderlineType 94
TrvActionFontEx.ValidProperties 95
TrvActionFontEx.VShift 95
TrvActionFontGrow 95
TrvActionFontGrow.MaxSize 96
TrvActionFontGrowOnePoint 96
TrvActionFontGrowOnePoint.MaxSize 96
TrvActionFontItalic 97
TrvActionFontOverline 97
TrvActionFonts 98
TrvActionFonts.Font 98
TrvActionFonts.UserInterface 99
TrvActionFontShrink 99
TrvActionFontShrink.MinSize 99
TrvActionFontShrinkGrow 99
TrvActionFontShrinkGrow.Percent 100
TrvActionFontShrinkOnePoint 100
TrvActionFontShrinkOnePoint.MinSize 101
TrvActionFontStrikeout 101
TrvActionFontStyle 101
TrvActionFontStyleEx 102
TrvActionFontUnderline 103
TrvActionHide 208
TrvActionIndent 115
TrvActionIndent.IndentStep 116
TrvActionIndentDec 116
TrvActionIndentInc 117
TrvActionIndentInc.IndentMax 117
TrvActionInsertCaption 140
TrvActionInsertCaption.ExcludeLabel 141
TrvActionInsertCaption.InsertAbove 141
TrvActionInsertCustomPageNumber 141
TrvActionInsertCustomPageNumber.NumberType 142
TrvActionInsertEndnote 165

TrvActionInsertEquation 142
TrvActionInsertEquation.Expression 143
TrvActionInsertEquation.UserInterface 143
TrvActionInsertFile 143
TrvActionInsertFile.Filter 144
TrvActionInsertFile.InsertFile 144
TrvActionInsertFootnote 166
TrvActionInsertHLine 144
TrvActionInsertHLine.Color 145
TrvActionInsertHLine.Style 145
TrvActionInsertHLine.Width 145
TrvActionInsertHyperlink 145
TrvActionInsertHyperlink.ActionStyleTemplates 147
TrvActionInsertHyperlink.AutoAddHyperlinkStyleTemplate 147
TrvActionInsertHyperlink.BackColor 147
TrvActionInsertHyperlink.Color 147
TrvActionInsertHyperlink.Cursor 147
TrvActionInsertHyperlink.DetectURL 150
TrvActionInsertHyperlink.EncodeTarget 151
TrvActionInsertHyperlink.GetHyperlinkStyleNo 151
TrvActionInsertHyperlink.GetNormalStyleNo 152
TrvActionInsertHyperlink.GoToLink 152
TrvActionInsertHyperlink.HoverBackColor 148
TrvActionInsertHyperlink.HoverColor 148
TrvActionInsertHyperlink.HoverEffects 148
TrvActionInsertHyperlink.HoverUnderlineColor 148
TrvActionInsertHyperlink.OnApplyHyperlinkToItem 153
TrvActionInsertHyperlink.OnGetHyperlinkTargetFromItem 153
TrvActionInsertHyperlink.OnHyperlinkForm 154
TrvActionInsertHyperlink.SetTags 152
TrvActionInsertHyperlink.SetToPictures 149
TrvActionInsertHyperlink.SpaceFiller 149
TrvActionInsertHyperlink.Style 149
TrvActionInsertHyperlink.StyleTemplateName 149
TrvActionInsertHyperlink.TerminateHyperlink 153
TrvActionInsertHyperlink.UnderlineColor 150
TrvActionInsertHyperlink.ValidProperties 150
TrvActionInsertNote 167
TrvActionInsertNote.ActionEditNote 168
TrvActionInsertNote.Font 168
TrvActionInsertNumber 154
TrvActionInsertNumSequence 155
TrvActionInsertNumSequence.NumberType 155
TrvActionInsertNumSequence.SeqName 155
TrvActionInsertNumSequence.UseDefaults 156
TrvActionInsertPageBreak 156
TrvActionInsertPageCount 156
TrvActionInsertPageNumber 157
TrvActionInsertPicture 157
TrvActionInsertPicture.BackgroundColor 158
TrvActionInsertPicture.BorderColor 158
TrvActionInsertPicture.BorderWidth 158
TrvActionInsertPicture.DefaultExt 158
TrvActionInsertPicture.Filter 159
TrvActionInsertPicture.MaxImageSize 159
TrvActionInsertPicture.OnInserting 160
TrvActionInsertPicture.OuterHSpacing 159
TrvActionInsertPicture.OuterVSpacing 159
TrvActionInsertPicture.Spacing 159
TrvActionInsertPicture.StoreFileNameInItemName 160
TrvActionInsertPicture.VAlign 160
TrvActionInsertSidenote 168
TrvActionInsertSidenote.RefStyleTemplateName 169
TrvActionInsertSymbol 160
TrvActionInsertSymbol.CharCode 161
TrvActionInsertSymbol.FontName 161
TrvActionInsertSymbol.Protection 161
TrvActionInsertSymbol.UseCurrentFont 161
TrvActionInsertTable 170
TrvActionInsertTable.BackgroundPicture 171
TrvActionInsertTable.BackgroundStyle 171
TrvActionInsertTable.BestWidth 171
TrvActionInsertTable.BorderColor 171
TrvActionInsertTable.BorderHSpacing 171
TrvActionInsertTable.BorderLightColor 172
TrvActionInsertTable.BorderStyle 172
TrvActionInsertTable.BorderVSpacing 172
TrvActionInsertTable.BorderWidth 172
TrvActionInsertTable.CellBorderColor 173
TrvActionInsertTable.CellBorderLightColor 173
TrvActionInsertTable.CellBorderStyle 173
TrvActionInsertTable.CellBorderWidth 173
TrvActionInsertTable.CellHPadding 173
TrvActionInsertTable.CellHSpacing 174
TrvActionInsertTable.CellPadding 173
TrvActionInsertTable.CellVPadding 173
TrvActionInsertTable.CellVSpacing 174
TrvActionInsertTable.ColBandSize 174
TrvActionInsertTable.ColCount 174
TrvActionInsertTable.Color 175
TrvActionInsertTable.EvenColumnsColor 178
TrvActionInsertTable.EvenRowsColor 178
TrvActionInsertTable.FirstColumnColor 178

TrvActionInsertTable.HeadingRowColor 178
TrvActionInsertTable.HeadingRowCount 175
TrvActionInsertTable.HOutermostRule 175
TrvActionInsertTable.HRuleColor 175
TrvActionInsertTable.HRuleWidth 175
TrvActionInsertTable.ItemText 176
TrvActionInsertTable.LastColumnColor 178
TrvActionInsertTable.LastRowColor 178
TrvActionInsertTable.OddColumnsColor 178
TrvActionInsertTable.OddRowsColor 178
TrvActionInsertTable.OnCloseTableSizeDialog 178
TrvActionInsertTable.OnInserting 178
TrvActionInsertTable.RowBandSize 174
TrvActionInsertTable.RowCount 176
TrvActionInsertTable.RowsKeepTogether 176
TrvActionInsertTable.RowsVAlign 176
TrvActionInsertTable.ShowTableSizeDialog 178
TrvActionInsertTable.TableOptions 176
TrvActionInsertTable.TablePrintOptions 177
TrvActionInsertTable.VisibleBorders 177
TrvActionInsertTable.VOutermostRule 177
TrvActionInsertTable.VRuleColor 177
TrvActionInsertTable.VRuleWidth 177
TrvActionInsertText 162
TrvActionInsertText.OnInsertText 162
TrvActionInsertTextBox 169
TrvActionItemProperties 208
TrvActionItemProperties.ActionInsertHLine 209
TrvActionItemProperties.ActionInsertTable 209
TrvActionItemProperties.BackgroundGraphicFilter 209
TrvActionItemProperties.GraphicFilter 210
TrvActionItemProperties.OnCanApply 210
TrvActionItemProperties.OnCustomItemPropertiesDialog 211
TrvActionItemProperties.StoreImageFileName 210
TrvActionItemProperties.UpdateAllInsertTableActions 209
TrvActionLineSpacing 117
TrvActionLineSpacing100 118
TrvActionLineSpacing150 118
TrvActionLineSpacing200 119
TrvActionNew 61
TrvActionNew.Reset 63
TrvActionNew.StyleTemplates 62
TrvActionOpen 63
TrvActionOpen.ActionNew 64
TrvActionOpen.CustomFilter 64
TrvActionOpen.DialogTitle 65
TrvActionOpen.Filter 65
TrvActionOpen.InitialDir 65
TrvActionOpen.LastFilterIndex 65
TrvActionOpen.LoadFile 65
TrvActionOpen.OnOpenFile 66
TrvActionPageSetup 69
TrvActionPageSetup.MarginsUnits 69
TrvActionPageSetup.OnChange 70
TrvActionParaBiDi 119
TrvActionParaBorder 120
TrvActionParaBorder.Background 121
TrvActionParaBorder.Border 121
TrvActionParaBorder.UserInterface 121
TrvActionParaBorder.ValidProperties 121
TrvActionParaBullets 122
TrvActionParaColor 122
TrvActionParaColorAndPadding 123
TrvActionParaColorAndPadding.Padding 123
TrvActionParaColorAndPadding.UsePadding 124
TrvActionParaCustomColor 124
TrvActionParagraph 124
TrvActionParagraph.Alignment 125, 126
TrvActionParagraph.DeleteAllTabs 125
TrvActionParagraph.FirstIndent 125
TrvActionParagraph.KeepLinesTogether 126
TrvActionParagraph.KeepWithNext 126
TrvActionParagraph.LastLineAlignment 126
TrvActionParagraph.LeftIndent 126
TrvActionParagraph.LineSpacing 127
TrvActionParagraph.LineSpacingType 127
TrvActionParagraph.OutlineLevel 127
TrvActionParagraph.RightIndent 127
TrvActionParagraph.SpaceAfter 128
TrvActionParagraph.SpaceBefore 128
TrvActionParagraph.Tabs 128
TrvActionParagraph.TabsToDelete 129
TrvActionParagraph.UserInterface 129
TrvActionParagraph.ValidProperties 129
TrvActionParaList 129
TrvActionParaList.ActionParaBullets 131
TrvActionParaList.ActionParaNumbering 131
TrvActionParaList.IndentStep 130
TrvActionParaList.UpdateAllActionsOnForm 130
TrvActionParaLTR 131
TrvActionParaNumbering 132
TrvActionParaRTL 132
TrvActionParaStyles 133
TrvActionPaste 82
TrvActionPasteAsText 82

TrvActionPasteSpecial 83
TrvActionPasteSpecial.AddFormat 84
TrvActionPasteSpecial.OnCanPaste 84
TrvActionPasteSpecial.OnCustomPaste 85
TrvActionPasteSpecial.OnShowing 85
TrvActionPasteSpecial.StoreFileNameInItemName 84
TrvActionPrint 70
TrvActionPrint.Title 70
TrvActionPrintPreview 71
TrvActionPrintPreview.ActionPageSetup 71
TrvActionPrintPreview.ActionPrint 71
TrvActionPrintPreview.Maximized 72
TrvActionPrintPreview.OnGetPreviewFormClass 72
TrvActionQuickPrint 72
TrvActionQuickPrint.Title 73
TrvActionRedo 85
TrvActionRemoveHyperlinks 211
TrvActionRemoveHyperlinks.ActionInsertHyperlink 211
TrvActionRemovePageBreak 211
TrvActionReplace 86
TrvActionReplace.ActionFind 86
TrvActionReplace.CloseDialog 87
TrvActionReplace.OnReplaceAllEnd 87
TrvActionReplace.OnReplaceAllStart 87
TrvActionReplace.OnReplacing 87
TrvActionReplace.ShowReplaceAllSummary 86
TrvActionSave 73
TrvActionSave.ActionSaveAs 74
TrvActionSave.CanCloseDoc 76
TrvActionSave.CreateDirectoryForHTMLImages 74
TrvActionSave.DisableWhenUnmodified 74
TrvActionSave.Documents 74
TrvActionSave.FileTitle 75
TrvActionSave.FindDoc 76
TrvActionSave.ImagePrefix 75
TrvActionSave.LostFormatWarning 75
TrvActionSave.OnDocumentFileChange 76
TrvActionSave.OnSaving 77
TrvActionSave.SaveOptions 75
TrvActionSave.SuppressNextErrorMessage 76
TrvActionSaveAs 77
TrvActionSaveAs.ActionSave 78
TrvActionSaveAs.Filter 78
TrvActionSelectAll 87
TrvActionShowSpecialCharacters 212
TrvActionShowSpecialCharacters.ShowCheckpoints 213
TrvActionSpellingCheck 201
TrvActionSSScript 103
TrvActionStyleInspector 137
TrvActionStyleInspector.FontImageIndex 137
TrvActionStyleInspector.Images 137
TrvActionStyleInspector.ParaImageIndex 137
TrvActionStyleInspector.UpdateInfo 138
TrvActionStyleTemplates 134
TrvActionStyleTemplates.Images 135
TrvActionStyleTemplates.ParaStyleImageIndex 136
TrvActionStyleTemplates.ParaTextStyleImageIndex 136
TrvActionStyleTemplates.StandardStyleTemplates 135
TrvActionStyleTemplates.TextStyleImageIndex 136
TrvActionSubscript 104
TrvActionSuperscript 104
TrvActionTableCell 179
TrvActionTableCellAllBorders 179
TrvActionTableCellBorder 180
TrvActionTableCellBottomBorder 180
TrvActionTableCellLeftBorder 181
TrvActionTableCellNoBorders 182
TrvActionTableCellRightBorder 182
TrvActionTableCellRotation 183
TrvActionTableCellRotation180 185
TrvActionTableCellRotation270 185
TrvActionTableCellRotation90 184
TrvActionTableCellRotationNone 183
TrvActionTableCellTopBorder 186
TrvActionTableCellVAlign 186
TrvActionTableCellVAlignBottom 187
TrvActionTableCellVAlignDefault 187
TrvActionTableCellVAlignMiddle 188
TrvActionTableCellVAlignTop 189
TrvActionTableDeleteCols 189
TrvActionTableDeleteRows 190
TrvActionTableDeleteTable 190
TrvActionTableGrid 191
TrvActionTableInsertColLeft 191
TrvActionTableInsertColRight 192
TrvActionTableInsertRowsAbove 192
TrvActionTableInsertRowsBelow 193
TrvActionTableInsertRowsBelow.AllowMultiple 193
TrvActionTableMergeCells 193
TrvActionTableMultiCellAttributes 194
TrvActionTableProperties 194
TrvActionTableProperties.ActionInsertTable 195

TrvActionTableProperties.BackgroundGraphicFilter 195
TrvActionTableProperties.UpdateAllInsertTableActions 195
TrvActionTableRCBase 196
TrvActionTableSelectCell 196
TrvActionTableSelectCols 197
TrvActionTableSelectRows 197
TrvActionTableSelectTable 198
TrvActionTableSort 199
TrvActionTableSplit 199
TrvActionTableSplitCells 200
TrvActionTableToText 200
TrvActionTextBiDi 105
TrvActionTextLTR 106
TrvActionTextRTL 106
TrvActionTextStyles 107
TrvActionThesaurus 201
TrvActionUndo 88
TrvActionVAlign 213
TrvActionWordWrap 134
TRVACustomDownloadInterface 227
TRVACustomPasteEvent 85
TRVACustomSpellInterface 230
TRVAddStyleEvent 39
TrvaDocumentInfo 74
TRVADownloadEvent 41
TRVADXSpellInterface 231
TRVAEditEvent 249
TRVAEditorControlCommand 253
TRVAEditorControlCommandFunction 253
TRVAEvent 206, 207
TRVAFileOperation 40
TRVAFillColorApplyToElement 207
TRVAFillColorApplyToSet 207
TRVAHFInfo 249
TRVAHunSpellInterface 231
TRVAIndyDownloadInterface 228
TRVAInsertPictureEvent 160
TRVAInsertTableEvent 178
TRVAInsertTextEvent 162
TRVALanguageName 224
TRVALiveSpellAddEvent type 45
TRVALiveSpellGetSuggestionsEvent type 46
TRVALiveSpellIgnoreAllEvent type 46
TRVALiveSpellReplaceEvent type 47
TRVALocString 224
TRVAMessageID 221
TRVAPolarSpellInterface 232
TRVAPopupActionBar 43
TRVAPopupMenu 43
TRVAPopupMenu.ActionList 45
TRVAPopupMenu.MaxSuggestionsCount 45
TRVAPopupMenu.OnLiveSpellAdd 45
TRVAPopupMenu.OnLiveSpellGetSuggestions 46
TRVAPopupMenu.OnLiveSpellIgnoreAll 46
TRVAPopupMenu.OnLiveSpellWordReplace 47
TRVApplyHyperlinkToItemEvent 153
TRVAReplacingEvent 87
TRVASearchScope 34
TRVASpellStringList 249
TRVASpellStrings 249
TRVASPTBXPopupMenu 43
TRVASPTBXPopupMenu.ActionList 45
TRVASPTBXPopupMenu.MaxSuggestionsCount 45
TRVASPTBXPopupMenu.OnLiveSpellAdd 45
TRVASPTBXPopupMenu.OnLiveSpellGetSuggestions 46
TRVASPTBXPopupMenu.OnLiveSpellIgnoreAll 46
TRVASPTBXPopupMenu.OnLiveSpellWordReplace 47
TRVATBPopupMenu 43
TRVATBPopupMenu.ActionList 45
TRVATBPopupMenu.MaxSuggestionsCount 45
TRVATBPopupMenu.OnLiveSpellAdd 45
TRVATBPopupMenu.OnLiveSpellGetSuggestions 46
TRVATBPopupMenu.OnLiveSpellIgnoreAll 46
TRVATBPopupMenu.OnLiveSpellWordReplace 47
TRVATBXPopupMenu 43
TRVATBXPopupMenu.ActionList 45
TRVATBXPopupMenu.MaxSuggestionsCount 45
TRVATBXPopupMenu.OnLiveSpellAdd 45
TRVATBXPopupMenu.OnLiveSpellGetSuggestions 46
TRVATBXPopupMenu.OnLiveSpellIgnoreAll 46
TRVATBXPopupMenu.OnLiveSpellWordReplace 47
TRVAUserInterface 36
TRVCanApplyEvent 210
TRVCharCase 248
TRVCreateFormEvent 40
TrvCustomAction 213
TrvCustomAction.Caption 214
TrvCustomAction.ControlPanel 214
TrvCustomAction.Disabled 214
TrvCustomAction.GetControlPanel 215
TrvCustomAction.Hint 215
TrvCustomEditAction 88

TrvCustomEditorAction 215
TrvCustomEditorAction.Form 216
TrvCustomEditorAction.OnFormCreate 216
TrvCustomEditorAction.OnHide 216
TrvCustomEditorAction.OnShowing 216
TrvCustomEditorAction.SubDocEditor 216
TRVCustomFileOperationEvent 40
TRVCustomItemPropertiesDialog 211
TrvCustomPrintAction 78
TrvFileExportFilter 250
TrvFileExportFilterSet 250
TRVFileFormatComponent 250
TrvFileImportFilter 251
TrvFileImportFilterSet 251
TrvFileOpenFilter 251
TrvFileOpenFilterSet 251
TrvFileSaveFilter 250
TrvFileSaveFilterSet 250
TRVFontCharsetComboBox 48
TRVFontCharsetComboBox.AddDefaultCharset 49
TRVFontCharsetComboBox.DefaultCharsetCaption 49
TRVFontCharsetComboBox.FontName 49
TRVFontComboBox 49
TRVFontComboBox.DropDownWidth 50
TRVFontComboBox.Preview 50
TRVFontComboBox.SymbolPreviewString 50
TRVFontInfoMainProperties 95
TRVFontInfoMainProperty 95
TRVFontSizeComboBox 50
TRVFontSizeComboBox.FontName 51
TRVGetFormClassEvent 72
TRVGetHyperlinkTargetFromItem 153
TRVHeaderFooterCodeEvent 42
TRVHTMLComponent 250
TRVHyperlinkFormEvent 154
TRVHyperlinkProperties 150
TRVHyperlinkProperty 150
TrvPaperMarginsUnits 69
TRVParaInfoBorderProperties 121
TRVParaInfoBorderProperty 121
TRVParaInfoMainProperties 129
TRVParaInfoMainProperty 129
TRVSaveFileEvent 77
TRVShowFormEvent 138, 252
TRVStyleNeededEvent 43
TRVStyleTemplateComboBox 51
TRVStyleTemplateComboBox.AllStylesImageIndex 52
TRVStyleTemplateComboBox.ClearFormatImageIndex 52
TRVStyleTemplateComboBox.ControlPanel 52
TRVStyleTemplateComboBox.Editor 52
TRVStyleTemplateComboBox.Images 53
TRVStyleTemplateComboBox.Localize 54
TRVStyleTemplateComboBox.OnClickAllStyles 54
TRVStyleTemplateComboBox.ParaStyleImageIndex 52
TRVStyleTemplateComboBox.ParaTextStyleImageIndex 52
TRVStyleTemplateComboBox.ShowAllStyles 53
TRVStyleTemplateComboBox.ShowClearFormat 53
TRVStyleTemplateComboBox.SmartHeadings 53
TRVStyleTemplateComboBox.TextStyleImageIndex 52
TRVStyleTemplateListBox.AllStylesImageIndex 52
TRVStyleTemplateListBox.ClearFormatImageIndex 52
TRVStyleTemplateListBox.ControlPanel 52
TRVStyleTemplateListBox.Editor 52
TRVStyleTemplateListBox.Images 53
TRVStyleTemplateListBox.Localize 54
TRVStyleTemplateListBox.OnClickAllStyles 54
TRVStyleTemplateListBox.ParaStyleImageIndex 52
TRVStyleTemplateListBox.ParaTextStyleImageIndex 52
TRVStyleTemplateListBox.ShowAllStyles 53
TRVStyleTemplateListBox.ShowClearFormat 53
TRVStyleTemplateListBox.SmartHeadings 53
TRVStyleTemplateListBox.TextStyleImageIndex 52
TRVXMLComponent 250

- U -

UnderlineColor 94, 150
  TrvActionFontEx 94
  TrvActionInsertHyperlink 150
UnderlineType 94
  TrvActionFontEx 94
UnitsDisplay 35
  TRVAControlPanel 35
UnitsProgram 35
  TRVAControlPanel 35
UpdateAllActionsOnForm 130
  TrvActionParaList 130
UpdateAllInsertTableActions 195, 209
  TrvActionTableProperties 195
UpdateAllInsertTableActions.TrvActionItemProperties 209

UpdateInfo 138
TrvActionStyleInspector 138
UseCurrentFont 161
TrvActionInsertSymbol 161
UseDefaults 156
TrvActionInsertNumSequence 156
UseHelpFiles 36
TRVAControlPanel 36
UseLastLineAlignment 109
TrvActionAlignCustomJustify 109
UsePadding 124
TrvActionParaColorAndPadding 124
UserInterface 36, 99, 121, 129, 143, 205
TRVAControlPanel 36
TrvActionCustomColor 205
TrvActionFonts 99
TrvActionInsertEquation 143
TrvActionParaBorder 121
TrvActionParagraph 129
UseTextCodePageDialog 37
TRVAControlPanel 37
UseXPThemes 37
TRVAControlPanel 37
Using RichViewActions 15

- V -

ValidProperties 95, 121, 129, 150
TrvActionFontEx 95
TrvActionInsertHyperlink 150
TrvActionParaBorder 121
TrvActionParagraph 129
VAlign 160
TrvActionInsertPicture 160
VisibleBorders 177
TrvActionInsertTable 177
VOutermostRule 177
TrvActionInsertTable 177
VRuleColor 177
TrvActionInsertTable 177
VRuleWidth 177
TrvActionInsertTable 177
VShift 95
TrvActionFontEx 95

- W -

Width 145
TrvActionInsertHLine 145

- X -

XMLComponent 37
TRVAControlPanel 37
XMLFilter 38
TRVAControlPanel 38
XMLLocalizable 38
TRVAControlPanel 38