Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the Microsoft [Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting [iplg@microsoft.com](mailto:iplg@microsoft.com).
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit [www.microsoft.com/trademarks](http://www.microsoft.com/trademarks).
- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights.** All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact [dochelp@microsoft.com](mailto:dochelp@microsoft.com).

**Preliminary Documentation.** This particular Open Specifications document provides documentation for past and current releases and/or for the pre-release version of this technology. This document provides final documentation for past and current releases and preliminary documentation, as applicable and specifically noted in this document, for the pre-release version. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. Because this documentation might change between the pre-release version and the final
version of this technology, there are risks in relying on this preliminary documentation. To the extent
that you incur additional development obligations or any other costs as a result of relying on this
preliminary documentation, you do so at your own risk.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/13/2009</td>
<td>0.1</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>8/28/2009</td>
<td>0.2</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/6/2009</td>
<td>0.3</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>2/19/2010</td>
<td>1.0</td>
<td>Major</td>
<td>Updated and revised the technical content</td>
</tr>
<tr>
<td>3/31/2010</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>4/30/2010</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/7/2010</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/29/2010</td>
<td>1.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>7/23/2010</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/27/2010</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/18/2011</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>6/10/2011</td>
<td>1.04</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>1/20/2012</td>
<td>1.5</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>4/11/2012</td>
<td>1.5</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>1.6</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>1.7</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>1.7</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/30/2013</td>
<td>1.7</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>1.7</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>1.7</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>1.7</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>1.8</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>------------</td>
<td>------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>1.9</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>3/16/2015</td>
<td>2.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/4/2015</td>
<td>2.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/15/2016</td>
<td>2.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>2.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/27/2018</td>
<td>3.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>8/28/2018</td>
<td>4.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/22/2021</td>
<td>5.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction ......................................................................................... 8  
  1.1 Glossary ......................................................................................... 8  
  1.2 References ...................................................................................... 9  
    1.2.1 Normative References ............................................................ 9  
    1.2.2 Informative References ........................................................... 9  
  1.3 Overview .......................................................................................... 9  
  1.4 Relationship to Protocols and Other Structures .............................. 10  
  1.5 Applicability Statement ................................................................. 10  
  1.6 Versioning and Localization ............................................................ 10  
  1.7 Vendor-Extensible Fields ................................................................. 10  

2 Structures ............................................................................................. 11  
  2.1 Part Enumerations ........................................................................... 11  
    2.1.1 Ribbon and Backstage Customizations .................................... 11  
  2.2 Global Elements ............................................................................. 11  
    2.2.1 customUI ................................................................................ 11  
    2.2.2 menu ....................................................................................... 11  
  2.3 Complex Types ............................................................................... 12  
    2.3.1 CT_Backstage .......................................................................... 12  
    2.3.2 CT_BackstageButtonBase ....................................................... 12  
    2.3.3 CT_BackstageCheckBox ......................................................... 14  
    2.3.4 CT_BackstageCheckBoxBase .................................................. 16  
    2.3.5 CT_BackstageComboBox ....................................................... 17  
    2.3.6 CT_BackstageDropDown ........................................................ 19  
    2.3.7 CT_BackstageEditBox ............................................................ 21  
    2.3.8 CT_BackstageFastCommandButton ...................................... 23  
    2.3.9 CT_BackstageGroup .............................................................. 25  
    2.3.10 CT_BackstageGroupButton .................................................. 27  
    2.3.11 CT_BackstageGroups ............................................................ 29  
    2.3.12 CT_BackstageItem ................................................................. 29  
    2.3.13 CT_BackstageLabelControl .................................................. 30  
    2.3.14 CT_BackstageMenuBase ....................................................... 31  
    2.3.15 CT_BackstageMenuButton ................................................... 33  
    2.3.16 CT_BackstageMenuCheckBox .............................................. 35  
    2.3.17 CT_BackstageMenuGroup ...................................................... 36  
    2.3.18 CT_BackstageMenuToggleButton ....................................... 37  
    2.3.19 CT_BackstagePrimaryMenu .................................................. 39  
    2.3.20 CT_BackstageRegularButton .............................................. 41  
    2.3.21 CT_BackstageSubMenu ........................................................ 42  
    2.3.22 CT_BackstageTab ................................................................. 44  
    2.3.23 CT_Box .................................................................................. 46  
    2.3.24 CT_Button .............................................................................. 48  
    2.3.25 CT_ButtonGroup ................................................................... 51  
    2.3.26 CT_ButtonRegular .............................................................. 53  
    2.3.27 CT_CheckBox ......................................................................... 56  
    2.3.28 CT_ComboBox ........................................................................ 58  
    2.3.29 CT_Command ...................................................................... 61  
    2.3.30 CT_Commands ...................................................................... 62  
    2.3.31 CT_ContextMenu ................................................................. 62  
    2.3.32 CT_ContextMenus ................................................................. 63  
    2.3.33 CT_ContextualTabs ............................................................... 64  
    2.3.34 CT_Control .......................................................................... 64  
    2.3.35 CT_ControlBase ..................................................................... 67  
    2.3.36 CT_ControlClone ................................................................. 69  
    2.3.37 CT_ControlCloneQat ............................................................. 71
<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>CT_ControlCloneRegular</td>
<td>2.3.38</td>
</tr>
<tr>
<td>CT_CustomUI</td>
<td>2.3.39</td>
</tr>
<tr>
<td>CT_DialogLauncher</td>
<td>2.3.40</td>
</tr>
<tr>
<td>CT_DropDownRegular</td>
<td>2.3.41</td>
</tr>
<tr>
<td>CT_DynamicMenu</td>
<td>2.3.42</td>
</tr>
<tr>
<td>CT_DynamicMenuRegular</td>
<td>2.3.43</td>
</tr>
<tr>
<td>CT_EditBox</td>
<td>2.3.44</td>
</tr>
<tr>
<td>CT_Gallery</td>
<td>2.3.45</td>
</tr>
<tr>
<td>CT_GalleryRegular</td>
<td>2.3.46</td>
</tr>
<tr>
<td>CT_Group</td>
<td>2.3.47</td>
</tr>
<tr>
<td>CT_GroupBox</td>
<td>2.3.48</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.3.49</td>
</tr>
<tr>
<td>CT_HeaderGroup</td>
<td>2.3.50</td>
</tr>
<tr>
<td>CT_Hyperlink</td>
<td>2.3.51</td>
</tr>
<tr>
<td>CT_ImageControl</td>
<td>2.3.52</td>
</tr>
<tr>
<td>CT_Item</td>
<td>2.3.53</td>
</tr>
<tr>
<td>CT_LabelControl</td>
<td>2.3.54</td>
</tr>
<tr>
<td>CT_LayoutContainer</td>
<td>2.3.55</td>
</tr>
<tr>
<td>CT_Menu</td>
<td>2.3.56</td>
</tr>
<tr>
<td>CT_MenuRegular</td>
<td>2.3.57</td>
</tr>
<tr>
<td>CT_MenuRoot</td>
<td>2.3.58</td>
</tr>
<tr>
<td>CT_MenuSeparator</td>
<td>2.3.59</td>
</tr>
<tr>
<td>CT_MenuSeparatorNoTitle</td>
<td>2.3.60</td>
</tr>
<tr>
<td>CT_MenuWithTitle</td>
<td>2.3.61</td>
</tr>
<tr>
<td>CT_PreviewDisplayControl</td>
<td>2.3.62</td>
</tr>
<tr>
<td>CT_Qat</td>
<td>2.3.63</td>
</tr>
<tr>
<td>CT_QatItems</td>
<td>2.3.64</td>
</tr>
<tr>
<td>CT_RadioGroup</td>
<td>2.3.65</td>
</tr>
<tr>
<td>CT_Ribbon</td>
<td>2.3.66</td>
</tr>
<tr>
<td>CT_SimpleGroups</td>
<td>2.3.67</td>
</tr>
<tr>
<td>CT_SplitButton</td>
<td>2.3.68</td>
</tr>
<tr>
<td>CT_SplitButtonBase</td>
<td>2.3.69</td>
</tr>
<tr>
<td>CT_SplitButtonRegular</td>
<td>2.3.70</td>
</tr>
<tr>
<td>CT_SplitButtonRestricted</td>
<td>2.3.71</td>
</tr>
<tr>
<td>CT_TaskFormGroup</td>
<td>2.3.72</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.3.73</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.3.74</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.3.75</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.3.76</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.3.77</td>
</tr>
<tr>
<td>CT_ToggleButton</td>
<td>2.3.78</td>
</tr>
<tr>
<td>CT_ToggleButtonRegular</td>
<td>2.3.79</td>
</tr>
<tr>
<td>CT_VisibleButton</td>
<td>2.3.80</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.3.81</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.3.82</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.3.83</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.3.84</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.3.85</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.3.86</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.3.87</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.3.88</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.3.89</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.3.90</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.3.91</td>
</tr>
<tr>
<td>CT_SplitButton</td>
<td>2.3.92</td>
</tr>
<tr>
<td>CT_SplitButtonBase</td>
<td>2.3.93</td>
</tr>
<tr>
<td>CT_SplitButtonRegular</td>
<td>2.3.94</td>
</tr>
<tr>
<td>CT_SplitButtonRestricted</td>
<td>2.3.95</td>
</tr>
<tr>
<td>CT_Gallery</td>
<td>2.3.96</td>
</tr>
<tr>
<td>CT_GalleryRegular</td>
<td>2.3.97</td>
</tr>
<tr>
<td>CT_SimpleGroups</td>
<td>2.3.98</td>
</tr>
<tr>
<td>CT_SplitButton</td>
<td>2.3.99</td>
</tr>
<tr>
<td>CT_SplitButtonBase</td>
<td>2.4.00</td>
</tr>
<tr>
<td>CT_SplitButtonRegular</td>
<td>2.4.01</td>
</tr>
<tr>
<td>CT_SplitButtonRestricted</td>
<td>2.4.02</td>
</tr>
<tr>
<td>CT_VisibleButton</td>
<td>2.4.03</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.04</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.05</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.4.06</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.4.07</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.4.08</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.4.09</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.4.10</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.4.11</td>
</tr>
<tr>
<td>CT_ToggleButton</td>
<td>2.4.12</td>
</tr>
<tr>
<td>CT_ToggleButtonRegular</td>
<td>2.4.13</td>
</tr>
<tr>
<td>CT_ToggleButtonRestricted</td>
<td>2.4.14</td>
</tr>
<tr>
<td>CT_VisibleButton</td>
<td>2.4.15</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.16</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.17</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.4.18</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.4.19</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.4.20</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.4.21</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.4.22</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.4.23</td>
</tr>
<tr>
<td>CT_ToggleButton</td>
<td>2.4.24</td>
</tr>
<tr>
<td>CT_ToggleButtonRegular</td>
<td>2.4.25</td>
</tr>
<tr>
<td>CT_ToggleButtonRestricted</td>
<td>2.4.26</td>
</tr>
<tr>
<td>CT_VisibleButton</td>
<td>2.4.27</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.28</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.29</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.4.30</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.4.31</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.4.32</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.4.33</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.4.34</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.4.35</td>
</tr>
<tr>
<td>CT_ToggleButton</td>
<td>2.4.36</td>
</tr>
<tr>
<td>CT_ToggleButtonRegular</td>
<td>2.4.37</td>
</tr>
<tr>
<td>CT_ToggleButtonRestricted</td>
<td>2.4.38</td>
</tr>
<tr>
<td>CT_VisibleButton</td>
<td>2.4.39</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.40</td>
</tr>
<tr>
<td>CT_VisibleToggleButton</td>
<td>2.4.41</td>
</tr>
<tr>
<td>CT_GroupControls</td>
<td>2.4.42</td>
</tr>
<tr>
<td>CT_TaskFormGroupCategory</td>
<td>2.4.43</td>
</tr>
<tr>
<td>CT_TaskFormGroupTask</td>
<td>2.4.44</td>
</tr>
<tr>
<td>CT_TaskGroup</td>
<td>2.4.45</td>
</tr>
<tr>
<td>CT_TaskGroupCategory</td>
<td>2.4.46</td>
</tr>
<tr>
<td>CT_TaskGroupTask</td>
<td>2.4.47</td>
</tr>
<tr>
<td>CT_ToggleButton</td>
<td>2.4.48</td>
</tr>
<tr>
<td>CT_ToggleButtonRestricted</td>
<td>2.4.49</td>
</tr>
</tbody>
</table>
### 2.4.9 ST_secondColumnMinWidth .............................................................. 169
### 2.4.10 ST_style1 .................................................................................. 170
### 2.4.11 ST_columnWidthPercent ............................................................... 170
### 2.4.12 ST_Delegate ............................................................................. 171
### 2.4.13 ST_expand ............................................................................... 171
### 2.4.14 ST_GalleryItemWidthHeight ....................................................... 172
### 2.4.15 ST_GalleryRowColumnCount ...................................................... 172
### 2.4.16 ST_style .................................................................................. 172
### 2.4.17 ST_ID ...................................................................................... 173
### 2.4.18 ST_ItemSize ............................................................................ 174
### 2.4.19 ST_Keytip ............................................................................... 174
### 2.4.20 ST_layoutChildren .................................................................. 175
### 2.4.21 ST_LongString ......................................................................... 175
### 2.4.22 ST_QID .................................................................................. 176
### 2.4.23 ST_Size .................................................................................. 176
### 2.4.24 ST_String .............................................................................. 177
### 2.4.25 ST_StringLength ..................................................................... 178
### 2.4.26 ST_TaskSizes ......................................................................... 178
### 2.4.27 ST_UniqueID ......................................................................... 179
### 2.4.28 ST_Uri .................................................................................. 179

### 3 Structure Examples .............................................................................. 181
#### 3.1 Ribbon ..................................................................................... 181
#### 3.2 Context Menu ............................................................................ 181
#### 3.3 Backstage .................................................................................. 182

### 4 Security ............................................................................................ 183
#### 4.1 Security Considerations for Implementers ..................................... 183
#### 4.2 Index of Security Fields ................................................................. 183

### 5 Appendix A: Full XML Schemas ......................................................... 184

### 6 Appendix B: Product Behavior ............................................................. 203

### 7 Change Tracking ................................................................................ 204

### 8 Index .................................................................................................. 205
1 Introduction

This document specifies the Custom UI XML Markup Version 2, which is used to implement UI customization in the context of the standards for Microsoft Office Open XML file formats as described in [ISO/IEC29500:2012]. Examples in this document describe customizations in the context of the Microsoft Office Fluent interface (UI), which includes the ribbon, Quick Access Toolbar, context menus, and Microsoft Office Backstage view, but the concepts extend naturally to any user interface.

Sections 1.7 and 2 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

contextual tab: A tab on the ribbon that displays commands related to the active selection or object.

Enhanced ScreenTip: A small, pop-up window that provides context-sensitive Help when users point to an item on the ribbon.

gallery: A type of ribbon control that displays a set of items from which users can choose.

KeyTip: A small, pop-up window that appears over commands on the ribbon when users press the ALT key. By pressing the key that is displayed in a KeyTip, users can execute the command that is associated with the KeyTip.

Microsoft Office Backstage view: A full-screen, user-interface model that exposes file-level functionality for the active document. It is part of the Microsoft Office Fluent user interface, which replaces the previous system of layered menus, toolbars, and task panes.

Quick Access Toolbar: A set of controls that represents tasks or commands within an application. By default, it provides access to frequently used commands on the ribbon. Users can customize it by adding or removing commands.

ribbon: A set of controls that represents tasks or commands within an application. The tasks and commands are organized into tabs. The ribbon appears at the top of an application window and is part of the Microsoft Office Fluent user interface, which replaces the previous system of layered menus, toolbars, and task panes.

TaskGroup: A group of buttons that are displayed on a tab in the Microsoft Office Backstage view.

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring XML schemas.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.
1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.


1.2.2 Informative References


1.3 Overview

In creating an interoperable implementation, it can be helpful to understand specific implementation choices used by other products implementing the same standard. For example, portions of the standard might provide only general guidance, leaving specific implementation choices up to the
application implementer; in some circumstances it can be helpful for other implementers to understand those choices.

This document provides information about how to implement UI customization in the context of [ISO/IEC29500-1:2016]. Examples in this document describe customizations in the context of the Microsoft Office Fluent interface (UI), which includes the ribbon, Quick Access Toolbar, context menus, and Backstage, but the concepts extend naturally to any user interface.

1.4 Relationship to Protocols and Other Structures

Customization of the UI is accomplished by adding a ribbon and Backstage customizations part to the document package described in [ISO/IEC29500-1:2016].

1.5 Applicability Statement

The structures in this document represent how to store customizations to the ribbon, Quick Access Toolbar, context menus, and Backstage within the [ISO/IEC29500-1:2016] document package and are not appropriate for stand-alone use. Custom UI XML can be used whenever it is desirable to have UI customizations travel with a file from one system to another. These UI customizations include, but are not limited to, rearranging the location of commands and adding additional functionality to the UI.

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

None.
2 Structures

2.1 Part Enumerations

This section specifies UI-related extensions to [ISO/IEC 29500-1:2016].

2.1.1 Ribbon and Backstage Customizations

<table>
<thead>
<tr>
<th>Content type</th>
<th>application/xml</th>
</tr>
</thead>
</table>

An instance of this part specifies ribbon and Backstage customizations. A package is permitted to contain at most one ribbon and Backstage Customizations part, and each part MUST be the target of an explicit relationship from the root package, as specified in [ISO/IEC 29500-1:2016] section 9.2.

A ribbon and Backstage Customizations part is permitted to contain explicit relationships to Image Parts, as specified in [ISO/IEC 29500-1:2016] section 15.2.14.

The syntax of the structures contained in this part uses XML schema definition (XSD), as specified in [XMLSCHEMA1] and [XMLSCHEMA2].

This specification defines and references various XML namespaces by using the mechanisms specified in [XMLNS].

The content of the part is XML. The root element for the part is the `customUI` element, as specified in section 2.2.1.

2.2 Global Elements

2.2.1 customUI


A `CT_CustomUI` element, as specified in section 2.3.39, that specifies the root tag in a Custom UI document.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this element.

```xml
<xsd:element name="customUI" type="CT_CustomUI"/>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.2.2 menu


A `CT_MenuRoot` element, as specified in section 2.3.58, that specifies the root element of the customization XML used in the contents determined by the `getcontent` attribute of `CT_DynamicMenu`, as specified in section 2.3.42.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this element.

    <xsd:element name="menu" type="CT_MenuRoot"/>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3 Complex Types

2.3.1 CT_Backstage


Referenced by: CT_CustomUI

Specifies a reference to the Backstage.

Child Elements:

- **tab**: A CT_BackstageTab element, as specified in section 2.3.22.
- **button**: A CT_BackstageFastCommandButton element, as specified in section 2.3.8.

Attributes:

- **onShow**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when the Backstage is shown.
- **onHide**: An ST_Delegate attribute that specifies the name of a callback function that is called when the Backstage is hidden.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

    <xsd:complexType name="CT_Backstage">
        <xsd:sequence>
            <xsd:choice minOccurs="0" maxOccurs="255">
                <xsd:element name="tab" type="CT_BackstageTab"/>
                <xsd:element name="button" type="CT_BackstageFastCommandButton"/>
            </xsd:choice>
        </xsd:sequence>
        <xsd:attribute name="onShow" type="ST_Delegate" use="optional"/>
        <xsd:attribute name="onHide" type="ST_Delegate" use="optional"/>
    </xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.2 CT_BackstageButtonBase


Referenced by: CT_BackstageRegularButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton

An abstract base class that specifies attributes common to all types of Backstage buttons.

Attributes:
id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

isDefinitive: A Boolean attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to false.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

image: An ST_URi attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are
mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageButtonBase">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Definitive"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attributeGroup ref="AG_Image"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.3 CT_BackstageCheckBox

**Target namespace:** http://schemas.microsoft.com.office/2009/07/customui

**Referenced by:** CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a Backstage check box control.

**Attributes:**

**expand:** An **ST_expand** attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

**description:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.
**getSupertip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The `supertip` and `getSupertip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**id:** An `ST_UniqueID` attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An `ST_String` attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**onAction:** An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**getPressed:** An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

**enabled:** A Boolean attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An `ST_String` attribute that specifies a string that is displayed as the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A Boolean attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An `ST_Keytip` attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageCheckBox">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageCheckBoxBase">
      <xsd:attributeGroup ref="AG_Expand"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_Screentip"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.4 CT_BackstageCheckBoxBase

**Target namespace:** http://schemas.microsoft.com_office/2009/07/customui

**Referenced by:** CT_BackstageCheckBox, CT_BackstageMenuCheckBox

An abstract base class that specifies attributes common to check box controls in the Backstage.

**Attributes:**

- **id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **onAction:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

- **getPressed:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

- **enabled:** A Boolean attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **label:** An ST_String attribute that specifies a string that is displayed as the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
得到标签: 一个ST_Delegate属性，它指定一个回调函数的名称，该函数被调用以确定控件的标签。得到标签和标签属性是互斥的。它们不能一起使用。如果没有指定任何属性，那么标签将不会显示。

可见性: 一个布尔属性，它指定控件的可见性状态。可见性和得到可见性的属性是互斥的。它们不能一起使用。如果没有指定任何属性，控件将默认为可显示。

得到可见性: 一个ST_Delegate属性，它指定一个回调函数的名称，该函数被调用以确定控件的可见性状态。可见性和得到可见性属性是互斥的。它们不能一起使用。如果没有指定任何属性，控件将默认为可显示。

密钥提示: 一个ST_密钥提示属性，如2.4.19所指定，它指定一个字符串，该字符串将作为控件的密钥提示显示。得到密钥提示和密钥提示属性是互斥的。它们不能一起使用。如果没有指定任何属性，密钥提示将不会显示。

得到密钥提示: 一个ST_Delegate属性，它指定一个回调函数的名称，该函数被调用以确定控件的密钥提示。得到密钥提示和密钥提示属性是互斥的。它们不能一起使用。如果没有指定任何属性，密钥提示将不会显示。

以下W3C XML Schema ([XMLSCHEMA1/2]节2.1)片段指定此复杂类型的内容。

```xml
<xsd:complexType name="CT_BackstageCheckBoxBase">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Pressed"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
```

见5.1节的完整W3C XML Schema ([XMLSCHEMA1/2]节2.1)。

2.3.5 CT_BackstageComboBox


引用: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

它在一个Backstage中指定一个组合框控制。

子元素:

item: 一个CT_BackstageItem元素，如2.3.12节所指定。

属性:

id: 一个ST_UniqueID属性，如2.4.27节所指定，它指定一个自定义控件的标识符。id, idQ, 和idMso属性是互斥的。每个控件必须有唯一的标识符。

idQ: 一个ST_QID属性，如2.4.22节所指定，它指定控件或容器的资格标识符。id, idQ, 和idMso属性是互斥的。每个控件必须有唯一的标识符。
tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

alignLabel: An ST_alignLabel attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

expand: An ST_expand attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getText: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the text displayed in this control. If this attribute is omitted, the text defaults to an empty string.

onChange: An ST_Delegate attribute that specifies the name of a callback function that is called when the text inside this control changes.

sizeString: An ST_String attribute, as specified in section 2.4.24, that specifies a string for which the size is used to determine the size of the text input area of this control. If this attribute is omitted, the size of the control is determined by the application.

getItemCount: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the number of child items in this control. If this attribute is omitted, the control displays the selection items which are specified as child elements. If this attribute is omitted and no child items are specified, the control displays the selection items which are specified as child elements.
are specified, the control is empty. If both this attribute and child items are specified, the child items are ignored.

**getItemLabel**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the label of a child item, identified by index. If this attribute is omitted, dynamically created selection items do not display labels.

**getItemID**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the identifier of a child item, identified by index. If this attribute is omitted, dynamically created selection items have blank identifiers.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageComboBox">
  <xsd:sequence>
    <xsd:element name="item" type="CT_BackstageItem" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attribute name="getText" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="onChange" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
  <xsd:attribute name="getItemCount" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemLabel" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.6 CT_BackstageDropDown

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a Backstage drop-down control.

**Child Elements**: 

- **item**: A `CT_BackstageItem` element, as specified in section 2.3.12.

**Attributes**:

- **id**: An `ST_UniqueID` attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes must be specified on all controls.

- **tag**: An `ST_String` attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.
alignLabel: An ST_alignLabel attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

expand: An ST_expand attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**getSelectedIndex**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine which selection item is selected, identified by index. If this attribute is omitted, the first item is initially selected.

**sizeString**: An **ST_String** attribute, as specified in section 2.4.24, that specifies a string for which the size is used to determine the size of this control. If this attribute is omitted, the control has an application-defined size.

**getItemCount**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the number of selection items in this control. If this attribute is omitted, the control displays the selection items that are specified as child elements. If this attribute is omitted and no child items are specified, the control is empty. If both this attribute and child items are specified, the child items are ignored.

**getItemLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the labels of dynamically created selection items, identified by index. If this attribute is omitted, dynamically created selection items do not display labels.

**getItemID**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the identifiers of dynamically created selection items, identified by index. If this attribute is omitted, dynamically created selection items have blank identifiers.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageDropDown">
  <xsd:sequence>
    <xsd:element name="item" type="CT_BackstageItem" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_ENABLED"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Screentip"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attribute name="getSelectedIndex" type="ST_Delegate" use="optional"></xsd:attribute>
  <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
  <xsd:attribute name="getItemCount" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemLabel" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.7 CT_BackstageEditBox

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a Backstage edit box control.

**Attributes**: 

**id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
idQ: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary **string** that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

alignLabel: An **ST_alignLabel** attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

expand: An **ST_expand** attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

enabled: A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getText: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the text in this control. If this attribute is omitted, the control defaults to being empty.

onChange: An **ST_Delegate** attribute that specifies the name of a callback function that is called when the text in this control changes.

maxLength: An **ST_StringLength** attribute, as specified in section 2.4.25, that specifies the maximum length, in characters, of strings allowed in this control. If this attribute is omitted, the length of the input string is limited only by application-specific constraints.
**sizeString:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string for which the length is used to determine the size of the text input area of this control. If this attribute is omitted, the size of the control is determined by the application.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageEditBox">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attribute name="getText" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="onChange" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="maxLength" type="ST_StringLength" use="optional"/>
  <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.8 CT_BackstageFastCommandButton

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Backstage

Specifies a button that exists in the navigation pane of the **Backstage**.

**Attributes:**

- **idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

- **insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.
onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.
isDefinitive: A Boolean attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to false.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are
mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageFastCommandButton">
    <xsd:complexContent>
        <xsd:extension base="CT_BackstageButtonBase">
            <xsd:attributeGroup ref="AG_IMAGEMso"/>
            <xsd:attributeGroup ref="AG_PositionAttributes"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.9 CT_BackstageGroup

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_TaskFormGroupTask, CT_BackstageGroups, CT_SimpleGroups

Specifies a **Backstage** group control.

**Child Elements**:

- **primaryItem**: A **CT_PrimaryItem** element, as specified in section 2.3.62.
- **topItems**: A **CT_GroupControls** element, as specified in section 2.3.49.
- **bottomItems**: A **CT_GroupControls** element, as specified in section 2.3.49.

**Attributes**:

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **idMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

- **insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is
specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An `ST_ID` attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An `ST_QID` attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**label**: An `ST_String` attribute that specifies a string that is displayed as the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible**: A `Boolean` attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**style**: An `ST_style` attribute, as specified in section 2.4.16, that specifies the visual style of this Backstage group control. The `style` and `getStyle` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control defaults to the normal style.

**getStyle**: An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the visual style of this Backstage group control. The `style` and `getStyle` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control defaults to the normal style.

**helperText**: An `ST_LongString` attribute, as specified in section 2.4.21, that specifies additional text that explains the functionality of this Backstage group control. The `helperText` and `getHelperText` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.

**getHelperText**: An `ST_Delegate` attribute that specifies the name of a callback that is called to determine the additional text for this Backstage group control. The `helperText` and `getHelperText` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.
**showLabel**: A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageGroup">
    <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="1">
            <xsd:element name="primaryItem" type="CT_PrimaryItem" minOccurs="0" maxOccurs="1"/>
        </xsd:choice>
        <xsd:element name="topItems" type="CT_GroupControls" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="bottomItems" type="CT_GroupControls" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="AG_IDAttributes"/>
    <xsd:attributeGroup ref="AG_PositionAttributes"/>
    <xsd:attributeGroup ref="AG_Label"/>
    <xsd:attributeGroup ref="AG_Visible"/>
    <xsd:attributeGroup ref="AG_GroupStyle"/>
    <xsd:attributeGroup ref="AG_HelperText"/>
    <xsd:attributeGroup ref="AG_ShowLabel"/>
</xsd:complexType>
```

See section [5.1](#) for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.10 CT_BackstageGroupButton

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a **Backstage** button that exists within a Backstage group control.

**Attributes**:

**expand**: An **ST_expand** attribute, as specified in section [2.4.13](#), that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

**style**: An **ST_style1** attribute, as specified in section [2.4.10](#), that specifies the visual style of this button control. If this attribute is omitted, the normal button style is used.

**screentip**: An **ST_String** attribute, as specified in section [2.4.24](#), that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip**: An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.
mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**isDefinitive:** A **Boolean** attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to **false**.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageGroupButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageRegularButton">
      <xsd:attributeGroup ref="AG_Expand"/>
      <xsd:attributeGroup ref="AG_ButtonStyle"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.11 CT_BackstageGroups


Referenced by: CT_BackstageTab

Specifies a list of Backstage group controls.

Child Elements:

taskFormGroup: A CT_TaskFormGroup element, as specified in section 2.3.77.

group: A CT_BackstageGroup element, as specified in section 2.3.9.

taskGroup: A CT_TaskGroup element, as specified in section 2.3.80.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageGroups">
  <xsd:choice minOccurs="1" maxOccurs="1">
    <xsd:choice minOccurs="0" maxOccurs="1">
      <xsd:element name="taskFormGroup" type="CT_TaskFormGroup"/>
    </xsd:choice>
    <xsd:group ref="EG_SimpleGroups"/>
  </xsd:choice>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.12 CT_BackstageItem

Specifies an item in a selection-type control in the **Backstage**.

**Attributes:**

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. They MUST NOT be used together. Exactly one of these attributes MUST be specified on all controls.

- **label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

- **getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageItem">
  <xsd:attribute name="id" type="ST_UniqueID" use="optional"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.13 CT_BackstageLabelControl

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a **Backstage** control that displays a simple string of text.

**Attributes:**

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary **string** that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **alignLabel**: An **ST_alignLabel** attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

- **expand**: An **ST_expand** attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.
enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g.getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

g.getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

noWrap: A Boolean attribute that specifies that this control's text will appear on a single line. If this attribute is omitted, its value defaults to false.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageLabelControl">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attribute name="noWrap" type="xsd:boolean" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

**2.3.14 CT_BackstageMenuBase**


Referenced by: CT_BackstageSubMenu, CT_BackstagePrimaryMenu

An abstract base class that specifies attributes common to all types of Backstage menu controls.

Child Elements:

menuGroup: A CT_BackstageMenuGroup element, as specified in section 2.3.17.

Attributes:
id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
getKeytip: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageMenuBase">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:element name="menuGroup" type="CT_BackstageMenuGroup"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.15 CT_BackstageMenuButton

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_BackstageMenuGroup

Specifies a Backstage button that exists within a menu.

**Attributes:**

- **description:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

- **getDescription:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

- **id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.
isDefinitive: A Boolean attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to false.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g.getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

g.getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

g.getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageMenuButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageButtonBase">
      <xsd:attributeGroup ref="AG_Description"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.16 CT_BackstageMenuCheckBox


Referenced by: CT_BackstageMenuToggleButton, CT_BackstageMenuGroup

Specifies a Backstage check box control that exists within a menu.

Attributes:

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

getPressed: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageMenuCheckBox">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageCheckBoxBase">
      <xsd:attributeGroup ref="AG_Description"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.17 CT_BackstageMenuGroup

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_BackstagePrimaryMenu, CT_BackstageSubMenu, CT_BackstageMenuBase

Specifies a grouping of controls within a **Backstage** menu control.

**Child Elements:**

- **button:** A **CT_BackstageMenuButton** element, as specified in section 2.3.15.
- **checkBox:** A **CT_BackstageMenuCheckBox** element, as specified in section 2.3.16.
- **menu:** A **CT_BackstageSubMenu** element, as specified in section 2.3.21.
- **toggleButton:** A **CT_BackstageMenuToggleButton** element, as specified in section 2.3.18.

**Attributes:**

- **id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI
documents. The \texttt{id}, \texttt{idQ}, and \texttt{idMso} attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An \texttt{ST\_String} attribute, as specified in section 2.4.24, that specifies an arbitrary \texttt{string} that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**label:** An \texttt{ST\_String} attribute that specifies a string that is displayed as the label of this control. The \texttt{label} and \texttt{getLabel} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An \texttt{ST\_Delegate} attribute that specifies the name of a callback function that is called to determine the label of this control. The \texttt{label} and \texttt{getLabel} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**itemSize:** An \texttt{ST\_ItemSize} attribute, as specified in section 2.4.18, that specifies the size of child items within this grouping container.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageMenuGroup">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:group ref="EG_BackstageMenuControls"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.18 CT\_BackstageMenuToggleButton

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT\_BackstageMenuGroup

Specifies a Backstage toggle button control that exists within a menu.

**Attributes:**

**image:** An \texttt{ST\_Uri} attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The \texttt{image}, \texttt{getImage}, and \texttt{imageMso} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An \texttt{ST\_ID} attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The \texttt{image}, \texttt{getImage}, and \texttt{imageMso} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An \texttt{ST\_Delegate} attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The \texttt{image}, \texttt{getImage}, and \texttt{imageMso} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
**description:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDefinition** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDefinition:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDefinition** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id, idQ,** and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id, idQ,** and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**getPressed:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageMenuToggleButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageMenuCheckBox">
      <xsd:attributeGroup ref="AG_Image"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.19 CT_BackstagePrimaryMenu

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui  
**Referenced by**: CT_PrimaryItem

Specifies a **Backstage** menu control that acts as the primary UI for the containing group.

**Child Elements**:

- **menuGroup**: A **CT_BackstageMenuGroup** element, as specified in section 2.3.17.

**Attributes**:

- **screentip**: An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

- **getScreentip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

- **supertip**: An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

- **getSupertip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary **string** that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.
enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xs:complexType name="CT_BackstagePrimaryMenu">
  <xs:complexContent>
    <xs:extension base="CT_BackstageMenuBase">
      <xs:attributeGroup ref="AG_Screentip"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.20 CT_BackstageRegularButton


Referenced by: CT_BackstageGroupButton, CT_PrimaryItem, CT_HeaderGroup

Specifies a Backstage button control.

Attributes:

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

isDefinitive: A Boolean attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to false.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageRegularButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageButtonBase">
      <xsd:attributeGroup ref="AG_Screentip"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.21 CT_BackstageSubMenu

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_BackstageMenuGroup

Specifies a **Backstage** menu control that is a submenu of another menu.

**Child Elements:**
menuGroup: A CT_BackstageMenuGroup element, as specified in section 2.3.17.

Attributes:

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getAddress: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are
mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema (XMLSCHEMA1/2 section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageSubMenu">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageMenuBase">
      <xsd:attributeGroup ref="AG_Description"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema (XMLSCHEMA1/2 section 2.1).

### 2.3.22 CT_BackstageTab

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Backstage

Specifies a Backstage tab control.

**Child Elements:**

**firstColumn:** A **CT_BackstageGroups** element, as specified in section 2.3.11.

**secondColumn:** A **CT_SimpleGroups** element, as specified in section 2.3.68.

**Attributes:**

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.
**idMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**enabled**: A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**title:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as a title of the menu. The **title** and **getTitle** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

**getTitle:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the title of the menu. The **title** and **getTitle** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

**columnWidthPercent:** An **ST_columnWidthPercent** attribute, as specified in section 2.4.11, that specifies the width of the first column in this Backstage tab as a percentage of the width of the entire Backstage tab. The second column fills the remainder of the tab. If the value of this attribute creates conflicts with the pixel widths given in the **firstColumnMinWidth**, **firstColumnMaxWidth**, **secondColumnMinWidth**, or **secondColumnMaxWidth** attributes, this attribute is ignored.

**firstColumnMinWidth:** An **ST_firstColumnMinWidth** attribute, as specified in section 2.4.6, that specifies the minimum width of the first column of this Backstage tab, in pixels.

**firstColumnMaxWidth:** An **ST_firstColumnMaxWidth** attribute, as specified in section 2.4.5, that specifies the maximum width of the first column in this Backstage tab, in pixels. The value of this attribute MUST be greater than or equal to the value of the **firstColumnMinWidth** attribute.

**secondColumnMinWidth:** An **ST_secondColumnMinWidth** attribute, as specified in section 2.4.9, that specifies the minimum width of the second column in this Backstage tab, in pixels.

**secondColumnMaxWidth:** An **ST_secondColumnMaxWidth** attribute, as specified in section 2.4.8, that specifies the maximum width of the second column in this Backstage tab, in pixels. The value of this attribute MUST be greater than or equal to the value of the **secondColumnMinWidth** attribute.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_BackstageTab">
  <xsd:sequence>
    <xsd:element name="firstColumn" type="CT_BackstageGroups" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="secondColumn" type="CT_SimpleGroups" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attributeGroup ref="AG_Title"/>
  <xsd:attribute name="columnWidthPercent" use="optional" type="ST_columnWidthPercent"/>
  <xsd:attribute name="firstColumnMinWidth" use="optional" type="ST_firstColumnMinWidth"/>
  <xsd:attribute name="firstColumnMaxWidth" use="optional" type="ST_firstColumnMaxWidth"/>
  <xsd:attribute name="secondColumnMinWidth" use="optional" type="ST_secondColumnMinWidth"/>
  <xsd:attribute name="secondColumnMaxWidth" use="optional" type="ST_secondColumnMaxWidth"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.23 CT_Box

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui
Referenced by: **CT_Group**

Specifies a grouping container control that aligns child controls horizontally or vertically.

**Child Elements:**

- **control:** A **CT_ControlClone** element, as specified in section 2.3.36.
- **labelControl:** A **CT_LabelControl** element, as specified in section 2.3.54.
- **button:** A **CT_Button** element, as specified in section 2.3.24.
- **toggleButton:** A **CT_ToggleButton** element, as specified in section 2.3.83.
- **checkBox:** A **CT_CheckBox** element, as specified in section 2.3.27.
- **editBox:** A **CT_EditBox** element, as specified in section 2.3.44.
- **comboBox:** A **CT_ComboBox** element, as specified in section 2.3.28.
- **dropDown:** A **CT_DropDownRegular** element, as specified in section 2.3.41.
- **gallery:** A **CT_Gallery** element, as specified in section 2.3.45.
- **menu:** A **CT_Menu** element, as specified in section 2.3.56.
- **dynamicMenu:** A **CT_DynamicMenu** element, as specified in section 2.3.42.
- **splitButton:** A **CT_SplitButton** element, as specified in section 2.3.69.
- **box:** A **CT_Box** element.
- **buttonGroup:** A **CT_ButtonGroup** element, as specified in section 2.3.25.

**Attributes:**

- **id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **visible:** A **Boolean** attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is
specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**boxStyle:** An ST_BoxStyle attribute, as specified in section 2.4.3, that specifies the alignment direction of the child controls. If this attribute is omitted, the child controls are laid out horizontally.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Box">
  <xsd:group ref="EG_Controls" minOccurs="0" maxOccurs="1000"/>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attribute name="boxStyle" type="ST_BoxStyle" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.24 CT_Button

**Target namespace:** http://schemas.microsoft.com.office/2009/07/customui

**Referenced by:** `CT_ControlClone`, `CT_Box`, `CT_Group`

Specifies a standard push-button control.

**Attributes:**

**size:** An ST_Size attribute, as specified in section 2.4.23, that specifies the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**getSize:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**onAction:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.
enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

generated: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

generatedDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getgeneratedDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

generatedImage: An ST_Imaginary attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The generatedImage, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

generatedImageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

generatedImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.
**supertip**: An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**showLabel:** A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Button">
  <xsd:complexContent>
    <xsd:extension base="CT_ButtonRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section **5.1** for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.25 CT_ButtonGroup

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Box, CT_Group

Specifies a grouping container that visually groups child controls together.

**Child Elements:**

- **control:** A **CT_ControlCloneRegular** element, as specified in section **2.3.38**.
- **button:** A **CT_ButtonRegular** element, as specified in section **2.3.26**.
- **toggleButton:** A **CT_ToggleButtonRegular** element, as specified in section **2.3.84**.
- **gallery:** A **CT_GalleryRegular** element, as specified in section **2.3.46**.
- **menu:** A **CT_MenuRegular** element, as specified in section **2.3.57**.
- **dynamicMenu:** A **CT_DynamicMenuRegular** element, as specified in section **2.3.43**.
- **splitButton:** A **CT_SplitButtonRegular** element, as specified in section **2.3.71**.
- **separator:** A **CT_Separator** element, as specified in section **2.3.67**.

**Attributes:**

---

[MS-CUSTOMUI2] - v20210422
Custom UI XML Markup Version 2 Specification
Copyright © 2021 Microsoft Corporation
Release: April 22, 2021
id: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

visible: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

insertAfterMso: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ButtonGroup">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:element name="control" type="CT_ControlCloneRegular"/>
      <xsd:element name="button" type="CT_ButtonRegular"/>
      <xsd:element name="ToggleButton" type="CT_ToggleButtonRegular"/>
      <xsd:element name="gallery" type="CT_GalleryRegular"/>
      <xsd:element name="menu" type="CT_MenuRegular"/>
      <xsd:element name="dynamicMenu" type="CT_DynamicMenuRegular"/>
      <xsd:element name="splitButton" type="CT_SplitButtonRegular"/>
      <xsd:element name="separator" type="CT_Separator"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.26 CT_ButtonRegular


Referenced by: CT_Button, CT_VisibleButton, CT_ToggleButtonRegular, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_DialogLauncher, CT_ButtonGroup, CT_QatItems, CT_ContextMenu, CT_MenuRoot

Specifies a button control that has a fixed size because of its location. This type is otherwise identical to the CT_Button type.

Attributes:

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertBeforeQ**: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage**: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ButtonRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_Control">
      <xsd:attributeGroup ref="AG_Action"/>
      <xsd:attributeGroup ref="AG_Enabled"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_Image"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.27 CT_CheckBox


Referenced by: CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_Box, CT_Group, CT_ContextMenu, CT_Menuroot

Specifies a standard check-box control.

Attributes:

getPressed: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.
**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the **Enhanced Screen Tip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
getKeytip: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```
<xsd:complexType name="CT_CheckBox">
  <xsd:complexContent>
    <xsd:restriction base="CT_ToggleButtonRegular">
      <xsd:attribute name="image" use="prohibited"/>
      <xsd:attribute name="imageMso" use="prohibited"/>
      <xsd:attribute name="getImage" use="prohibited"/>
      <xsd:attribute name="showImage" use="prohibited"/>
      <xsd:attribute name="showLabel" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.28 CT_ComboBox

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Box, CT_Group

Specifies a standard combo box control.

**Child Elements:**

- **item:** A `CT_Item` element.

**Attributes:**

- **showItemImage:** A Boolean attribute that specifies whether this control displays icons on its selection items.
- **getItemCount:** An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the number of dynamically created selection items in this control. If this attribute is omitted, the control displays the selection items that are specified as child elements. If this attribute is omitted and no child items are specified, the control is empty. If both this attribute and child items are specified, the child items are ignored.
- **getItemLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the labels of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display labels.
- **getItemScreentip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the tooltips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display tooltips.
- **getItemSupertip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display Enhanced ScreenTips.
**getItemImage:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the icons of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display icons.

**getItemID:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the identifiers of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items have empty identifiers.

**sizeString:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a **string** of which the size is used to determine the width of the text input area of this control. If this attribute is omitted, the application determines the width of the text input area automatically.

**invalidateContentOnDrop:** A **Boolean** attribute that specifies whether this control invalidates its contents and requeries for them each time the user opens its drop-down menu.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image, getImage, and imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image, getImage, and imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image, getImage, and imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**maxLength:** An **ST_StringLength** attribute, as specified in section 2.4.25, that specifies an **integer** that is used as the maximum length, in characters, of **strings** that are entered into the control. If this attribute is omitted, the length of the input **string** is not limited except by application-specific constraints.

**getText:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the text inside of the edit box control. If this attribute is omitted, the control defaults to no text.

**onChange:** An **ST_Delegate** attribute that specifies the name of a callback function that is called when the text in the edit box has been changed.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id, idQ, and idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id, idQ, and idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary **string** that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.
**getVisible:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An `ST_Keytip` attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A Boolean attribute that specifies whether the icon for this control will be displayed. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ComboBox">
  <xsd:complexContent>
    <xsd:extension base="CT_EditBox">
      <xsd:sequence>
        <xsd:element name="item" type="CT_Item" minOccurs="0" maxOccurs="1000"/>
      </xsd:sequence>
      <xsd:attributeGroup ref="AG_DropDownAttributes"/>
      <xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.29 CT_Command

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Commands

Specifies a particular built-in command in the application for which the functionality will be repurposed.

**Attributes:**
**onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Command" mixed="false">
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_IDMso"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.30 CT_Commands

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_CustomUI

Specifies a list of commands that are repurposed.

**Child Elements:**

**command:** A **CT_Command** element, as specified in section 2.3.29.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Commands">
  <xsd:sequence>
    <xsd:element name="command" type="CT_Command" minOccurs="1" maxOccurs="5000"/>
  </xsd:sequence>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.31 CT_ContextMenu

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_ContextMenus

Specifies a built-in context menu control.
Child Elements:

**control:** A `CT_ControlCloneRegular` element, as specified in section 2.3.38.
**button:** A `CT_ButtonRegular` element, as specified in section 2.3.26.
**checkBox:** A `CT_CheckBox` element, as specified in section 2.3.27.
**gallery:** A `CT_GalleryRegular` element, as specified in section 2.3.46.
**toggleButton:** A `CT_ToggleButtonRegular` element, as specified in section 2.3.84.
**splitButton:** A `CT_SPLITButtonRegular` element, as specified in section 2.3.71.
**menu:** A `CT_MenuRegular` element, as specified in section 2.3.57.
**dynamicMenu:** A `CT_DynamicMenuRegular` element, as specified in section 2.3.43.
**menuSeparator:** A `CT_MenuSeparatorNoTitle` element, as specified in section 2.3.60.

Attributes:

**idMso:** An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ContextMenu">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:group ref="EG_ContextMenuControls"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDMso"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.32 CT_ContextMenus

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_CustomUI

Specifies a list of built-in context menu controls.

Child Elements:

**contextMenu:** A `CT_ContextMenu` element, as specified in section 2.3.31.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ContextMenus">
  <xsd:sequence>
    <xsd:element name="contextMenu" type="CT_ContextMenu" minOccurs="1" maxOccurs="1000"/>
  </xsd:sequence>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.33 CT_ContextualTabs


Referenced by: CT_Ribbon

Specifies a list of contextual tab sets.

Child Elements:

**tabSet**: A CT_TabSet element, as specified in section 2.3.76.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ContextualTabs">
  <xsd:sequence>
    <xsd:element name="tabSet" type="CT_TabSet" minOccurs="1" maxOccurs="100"/>
  </xsd:sequence>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.34 CT_Control


Referenced by: CT_ControlCloneRegular, CT_ButtonRegular, CT_LabelControl, CT_EditBox, CT_DropDownRegular, CT_SplitButtonBase

An abstract base class that specifies attributes common to all types of custom controls.

Attributes:

**id**: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**image**: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together.
together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A Boolean attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A Boolean attribute that specifies whether the icon for this control will be displayed. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Control">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:attributeGroup ref="AG_IDAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.35 CT_ControlBase


Referenced by: CT_Control, CT_ControlCloneQat, CT_DynamicMenuRegular, CT_MenuRegular, CT_MenuWithTitle

An abstract base class that specifies attributes common to all types of ribbon controls.

Attributes:

**image**: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip**: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip**: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled**: A Boolean attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label**: An ST_String attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ControlBase">
  <xsd:attributeGroup ref="AG_ControlAttributes"/>
</xsd:complexType>
```

See section [5.1](#) for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.36 CT_ControlClone

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_Box](#), [CT_Group](#)

Specifies a clone of an existing control. Built-in controls are specified with the `idMso` attribute. Custom controls are specified with the `idQ` attribute.

**Attributes:**

- **size:** An `ST_Size` attribute, as specified in section 2.4.23, that specifies the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

- **getSize:** An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

- **enabled:** A Boolean attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **description:** An `ST_LongString` attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The `description` and `getDescription` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

- **getDescription:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The `description` and `getDescription` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

- **image:** An `ST_Uri` attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

- **imageMso:** An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image, getImage,** and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id, idQ,** and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id, idQ,** and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso,** and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso,** and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso,** and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertBeforeQ:** An `ST_QID` attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A `Boolean` attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An `ST_Keytip` attribute, as specified in section 2.4.19, that specifies a `string` that is displayed as the `KeyTip` for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A `Boolean` attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A `Boolean` attribute that specifies whether the icon for this control will be displayed. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ControlClone">
  <xsd:complexContent>
    <xsd:restriction base="CT_Button">
      <xsd:attribute name="id" use="prohibited"/>
      <xsd:attribute name="onAction" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.37 CT_ControlCloneQat

Specifies a clone of an existing control. This complex type is specific to the Quick Access Toolbar, but otherwise this complex type is identical to the CT_ControlClone type.

Attributes:

id: An ST_ID attribute, as specified in section 2.4.17 that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

size: An ST_Size attribute, as specified in section 2.4.23, that specifies the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

getSize: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.
**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually
exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ControlCloneQat">
    <xsd:complexContent>
        <xsd:extension base="CT_ControlBase">
            <xsd:attribute name="id" type="ST_ID" use="optional"/>
            <xsd:attribute name="idQ" type="ST_QID" use="optional"/>
            <xsd:attributeGroup ref="AG_IDMso"/>
            <xsd:attributeGroup ref="AG_Description"/>
            <xsd:attributeGroup ref="AG_SizeAttributes"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.38 CT_ControlCloneRegular

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a clone of a control whose size is determined by its location. This complex type is otherwise equivalent to the CT_ControlClone type.

**Attributes:**

**idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI
documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

createnip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The createnip and getCreatenip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

getCreatenip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The createnip and getCreatenip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ControlCloneRegular">
  <xsd:complexContent>
    <xsd:restriction base="CT_Control">
      <xsd:attribute name="id" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.39 CT_CustomUI


Referenced by: customUI

Specifies the root tag in a Custom UI document.

Child Elements:

- **commands**: A CT_Commands element, as specified in section 2.3.30.
- **ribbon**: A CT_Ribbon element, as specified in section 2.3.66.
- **backstage**: A CT_Backstage element, as specified in section 2.3.1.
- **contextMenus**: A CT_ContextMenus element, as specified in section 2.3.32.

Attributes:

- **onLoad**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when the custom UI in this document is loaded by the application.
- **loadImage**: An ST_Delegate attribute that specifies the name of a callback function that is called when the application needs to load an icon image.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_CustomUI">
  <xsd:sequence>
    <xsd:element name="commands" type="CT_Commands" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="ribbon" type="CT_Ribbon" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="backstage" type="CT_Backstage" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="contextMenus" type="CT_ContextMenus" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="onLoad" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="loadImage" type="ST_Delegate" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.40 CT_DialogLauncher

Referenced by: **CT_Group**

Specifies a button that opens additional UI.

**Child Elements:**

**button:** A **CT_ButtonRegular** element, as specified in section 2.3.26.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_DialogLauncher">
  <xsd:sequence>
    <xsd:element name="button" type="CT_ButtonRegular" minOccurs="1" maxOccurs="1"/>
  </xsd:sequence>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.41 CT_DropDownRegular

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

Referenced by: **CT_GalleryRegular**, **CT_Box**, **CT_Group**

Specifies a standard drop-down control. The drop-down control optionally has a group of buttons below its selection items.

**Child Elements:**

**item:** A **CT_Item** element, as specified in section 2.3.53.

**button:** A **CT_ButtonRegular** element, as specified in section 2.3.26.

**Attributes:**

**onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and
imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**showItemImage:** A Boolean attribute that specifies whether this control displays icons on its selection items.

** getItemCount:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the number of dynamically created selection items in this control. If this attribute is omitted, the control displays the selection items that are specified as child elements. If this attribute is omitted and no child items are specified, the control is empty. If both this attribute and child items are specified, the child items are ignored.

**getItemLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the labels of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display labels.

**getItemScreentip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display tooltips.

**getItemSupertip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display Enhanced ScreenTips.

**getItemImage:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the icons of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display icons.

**getItemID:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the identifiers of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items have empty identifiers.

**sizeString:** An ST_String attribute, as specified in section 2.4.24, that specifies a string of which the size is used to determine the width of the text input area of this control. If this attribute is omitted, the application determines the width of the text input area automatically.

**getSelectedItemId:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the identifier of the item that is selected in this control. The getSelectedItemId and getSelectedItemIndex attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control does not display a selected item.

**getSelectedItemIndex:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the index of the item that is selected in this control. The getSelectedItemId and getSelectedItemIndex attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control does not display a selected item.

**showItemImage:** A Boolean attribute that specifies whether this control displays labels on its selection items. If this attribute is omitted, the control displays labels on its selection items.

**id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

g getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

g getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.
**getVisible:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An `ST_Keypad` attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A Boolean attribute that specifies whether the icon for this control will be displayed. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_DropDownRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_Control">
      <xsd:sequence>
        <xsd:element name="item" type="CT_Item" minOccurs="0" maxOccurs="1000"/>
        <xsd:element name="button" type="CT_ButtonRegular" minOccurs="0" maxOccurs="16"/>
      </xsd:sequence>
      <xsd:attributeGroup ref="AG_Action"/>
      <xsd:attributeGroup ref="AG_Enabled"/>
      <xsd:attributeGroup ref="AG_Image"/>
      <xsd:attributeGroup ref="AG_DropDownAttributes"/>
      <xsd:attribute name="getSelectedItemID" type="ST_Delegate" use="optional"/>
      <xsd:attribute name="getSelectedItemIndex" type="ST_Delegate" use="optional"/>
      <xsd:attribute name="showItemLabel" type="xsd:boolean" use="optional"/>
      <xsd:attribute name="showImage" type="xsd:boolean" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

## 2.3.42 CT_DynamicMenu

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Box, CT_Group

Specifies a menu control in which the contents are dynamically populated at run time.
Attributes:

**size**: An ST_Size attribute, as specified in section 2.4.23, that specifies the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**getSize**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**description**: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**id**: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**getContent**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the contents of this menu control.

**invalidateContentOnDrop**: A Boolean attribute that specifies whether this control invalidates its contents and requeries for them each time the user opens its drop-down menu.

**image**: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

g getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

g getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_DynamicMenu">
  <xsd:complexContent>
    <xsd:extension base="CT_DynamicMenuRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.43 CT_DynamicMenuRegular


Referenced by: CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_DynamicMenu, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a dynamically populated menu control that has a fixed size because of its location. It is otherwise identical to the CT_DynamicMenu complex type.

Attributes:
**description:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**getContent:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the contents of this menu control.

**invalidateContentOnDrop:** A **Boolean** attribute that specifies whether this control invalidates its contents and requeries for them each time the user opens its drop-down menu.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.
getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage**: A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_DynamicMenuRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_IDAttributes"/>
      <xsd:attributeGroup ref="AG_GetContentAttributes"/>
      <xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.44 CT_EditBox

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_ComboBox, CT_Box, CT_Group

Specifies a standard edit box control.

**Attributes**:

**enabled**: A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**image**: An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso**
attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**maxLength**: An ST_StringLength attribute, as specified in section 2.4.25, that specifies an integer that is used as the maximum length, in characters, of strings that are entered into the control. If this attribute is omitted, the length of the input string is not limited except by application-specific constraints.

**getText**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the text inside of the edit box control. If this attribute is omitted, the control defaults to no text.

**onChange**: An ST_Delegate attribute that specifies the name of a callback function that is called when the text in the edit box has been changed.

**sizeString**: An ST_String attribute, as specified in section 2.4.24, that specifies a string for which the size is used to determine the width of the text input area of this control. If this attribute is omitted, the size of the control is determined by the application.

**id**: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip**: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip**: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are
mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.
**showImage**: A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_EditBox">
  <xsd:complexContent>
    <xsd:extension base="CT_Control">
      <xsd:attributeGroup ref="AG_Enabled"/>
      <xsd:attributeGroup ref="AG_Image"/>
      <xsd:attribute name="maxLength" type="ST_StringLength" use="optional"/>
      <xsd:attribute name="getText" type="ST_Delegate" use="optional"/>
      <xsd:attribute name="onChange" type="ST_Delegate" use="optional"/>
      <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.45 CT_Gallery

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_Box, CT_Group

Specifies a **gallery** control that displays a drop-down grid of selection items.

**Child Elements**:

- **item**: A **CT_Item** element, as specified in section 2.3.53.
- **button**: A **CT_ButtonRegular** element, as specified in section 2.3.26.

**Attributes**:

- **size**: An **ST_Size** attribute, as specified in section 2.4.23, that specifies the size of this control. The **size** and **getSize** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

- **getSize**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The **size** and **getSize** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

- **description**: An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

- **getDescription**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription**
attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a
detailed description is not displayed.

**invalidateContentOnDrop**: A **Boolean** attribute that specifies whether this control invalidates its
contents and requeries for them each time the user opens its drop-down menu.

**columns**: An **ST_GalleryRowColumnCount** attribute, as specified in section 2.4.15, that specifies
the number of columns displayed in this gallery control. If this attribute is omitted, the number of
columns is chosen automatically based on the total number of items.

**rows**: An **ST_GalleryRowColumnCount** attribute that specifies the number of rows displayed in this
gallery control. If this attribute is omitted, the number of rows is chosen automatically based on the
total number of items.

**itemWidth**: An **ST_GalleryItemWidthHeight** attribute, as specified in section 2.4.14, that specifies
the width, in pixels, of the selection items in this control. The **itemWidth** and **getItemWidth**
attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified,
the size of the first item's contents determines the width of all the items in the control.

**itemHeight**: An **ST_GalleryItemWidthHeight** attribute that specifies the height, in pixels, of the
selection items in this control. The **itemHeight** and **getItemHeight** attributes are mutually exclusive.
They MUST NOT be used together. If neither attribute is specified, the items will all take the height of
the first item, based on its contents.

**getItemWidth**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a
callback function that is called to determine the width, in pixels, of the selection items in this control.
The **itemWidth** and **getItemWidth** attributes are mutually exclusive. They MUST NOT be used
together. If neither attribute is specified, the items will all take the width of the first item, based on its
contents.

**getItemHeight**: An **ST_Delegate** attribute that specifies the name of a callback function that is
called to determine the height, in pixels, of the selection items in this control. The **itemHeight** and **getItemHeight**
attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the items will all take the height of the first item, based on its contents.

**showItemLabel**: A **Boolean** attribute that specifies whether the selection items in this control
display labels. If this attribute is omitted, the selection items display labels.

**showInRibbon**: An **ST_GalleryShowInRibbon** attribute, as specified in section 2.4.7. This attribute
MUST be ignored.

**onAction**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a
callback function that is called when this control is invoked by the user.

**enabled**: A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and
**setEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute
is specified, the control defaults to being enabled.

**setEnabled**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to
determine the enabled state of this control. The **enabled** and **setEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to
being enabled.

**image**: An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso**
attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are
specified, an icon is not displayed.

**imageMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are
mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an
icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a
callback function that is called to determine the icon of this control. The **image**, **getImage**, and
**imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these
attributes are specified, an icon is not displayed.

**showItemImage:** A **Boolean** attribute that specifies whether this control displays icons on its
selection items.

**getItemCount:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the number of
dynamically created selection items in this control. If this attribute is omitted, the control displays the selection items that are specified as
child elements. If this attribute is omitted and no child items are specified, the control is empty. If
both this attribute and child items are specified, the child items are ignored.

**getItemLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called
to determine the labels of dynamically created selection items in this control. If this attribute is
omitted, dynamically created selection items do not display labels.

**getItemScreenTip:** An **ST_Delegate** attribute that specifies the name of a callback function that is
called to determine the tooltips of dynamically created selection items in this control. If this attribute is
omitted, dynamically created selection items do not display tooltips.

**getItemSuperTip:** An **ST_Delegate** attribute that specifies the name of a callback function that is
called to determine the **Enhanced ScreenTips** of dynamically created selection items in this control.
If this attribute is omitted, dynamically created selection items do not display Enhanced ScreenTips.

**getItemImage:** An **ST_Delegate** attribute that specifies the name of a callback function that is
called to determine the icons of dynamically created selection items in this control. If this
attribute is omitted, dynamically created selection items do not display icons.

**getItemID:** An **ST_Delegate** attribute that specifies the name of a callback function that is called
to determine the identifiers of dynamically created selection items in this control. If this attribute is
omitted, dynamically created selection items have empty identifiers.

**sizeString:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a **string** of which
the size is used to determine the width of the text input area of this control. If this attribute is
omitted, the application determines the width of the text input area automatically.

**getSelectedItemID:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the identifier of the item that is selected in this
control. The **getSelectedItemID** and **getSelectedItemIndex** attributes are mutually exclusive.
They MUST NOT be used together. If neither attribute is specified, the control does not display a
selected item.

**getSelectedItemIndex:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the index of the item that is selected in this
control. The **getSelectedItemID** and **getSelectedItemIndex** attributes are mutually exclusive.
They MUST NOT be used together. If neither attribute is specified, the control does not display a
selected item.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom
control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a
control. This attribute is used to reference controls or containers created by other Custom UI
documents. The \texttt{id}, \texttt{idQ}, and \texttt{idMso} attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

\textbf{tag}: An \texttt{ST\_String} attribute, as specified in section 2.4.24, that specifies an arbitrary \texttt{string} that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

\textbf{idMso}: An \texttt{ST\_ID} attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The \texttt{id}, \texttt{idQ}, and \texttt{idMso} attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

\textbf{screentip}: An \texttt{ST\_String} attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The \texttt{screentip} and \texttt{getScreentip} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

\textbf{getScreentip}: An \texttt{ST\_Delegate} attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The \texttt{screentip} and \texttt{getScreentip} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

\textbf{supertip}: An \texttt{ST\_String} attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The \texttt{supertip} and \texttt{getSupertip} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

\textbf{getSupertip}: An \texttt{ST\_Delegate} attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The \texttt{supertip} and \texttt{getSupertip} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

\textbf{label}: An \texttt{ST\_String} attribute that specifies a string that is displayed as the label of this control. The \texttt{label} and \texttt{getLabel} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

\textbf{getLabel}: An \texttt{ST\_Delegate} attribute that specifies the name of a callback function that is called to determine the label of this control. The \texttt{label} and \texttt{getLabel} attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

\textbf{insertAfterMso}: An \texttt{ST\_ID} attribute that specifies the identifier of a built-in control after which this control will be inserted. The \texttt{insertAfterMso}, \texttt{insertAfterQ}, \texttt{insertBeforeMso}, and \texttt{insertBeforeQ} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

\textbf{insertBeforeMso}: An \texttt{ST\_ID} attribute that specifies the identifier of a built-in control before which this control will be inserted. The \texttt{insertAfterMso}, \texttt{insertAfterQ}, \texttt{insertBeforeMso}, and \texttt{insertBeforeQ} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

\textbf{insertAfterQ}: An \texttt{ST\_QID} attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The \texttt{insertAfterMso}, \texttt{insertAfterQ}, \texttt{insertBeforeMso}, and \texttt{insertBeforeQ} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

\textbf{insertBeforeQ}: An \texttt{ST\_QID} attribute that specifies the qualified identifier of a control before which this control will be inserted. The \texttt{insertAfterMso}, \texttt{insertAfterQ}, \texttt{insertBeforeMso}, and \texttt{insertBeforeQ} attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Gallery">
  <xsd:complexContent>
    <xsd:extension base="CT_GalleryRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.46 CT_GalleryRegular


Referenced by: CT_Gallery, CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a gallery control that is of fixed size because of its location. It is otherwise identical to the CT_Gallery complex type.

Child Elements:
item: A CT_Item element, as specified in section 2.3.53.

button: A CT_ButtonRegular element, as specified in section 2.3.26.

Attributes:

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getchDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

invalidateContentOnDrop: A Boolean attribute that specifies whether this control invalidates its contents and requeries for them each time the user opens its drop-down menu.

columns: An ST_GalleryRowColumnCount attribute, as specified in section 2.4.15, that specifies the number of columns displayed in this gallery control. If this attribute is omitted, the number of columns is chosen automatically based on the total number of items.

rows: An ST_GalleryRowColumnCount attribute that specifies the number of rows displayed in this gallery control. If this attribute is omitted, the number of rows is chosen automatically based on the total number of items.

itemWidth: An ST_GalleryItemWidthHeight attribute, as specified in section 2.4.14, that specifies the width, in pixels, of the selection items in this control. The itemWidth and getItemWidth attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the size of the first item’s contents determines the width of all the items in the control.

itemHeight: An ST_GalleryItemWidthHeight attribute that specifies the height, in pixels, of the selection items in this control. The itemHeight and getItemHeight attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the items will all take the height of the first item, based on its contents.

getchItemWidth: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the width, in pixels, of the selection items in this control. The itemWidth and getItemWidth attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the items will all take the width of the first item, based on its contents.

getchItemHeight: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the height, in pixels, of the selection items in this control. The itemHeight and getItemHeight attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the items will all take the height of the first item, based on its contents.

showItemLabel: A Boolean attribute that specifies whether the selection items in this control display labels. If this attribute is omitted, the selection items display labels.

showInRibbon: An ST_GalleryShowInRibbon attribute, as specified in section 2.4.7. This attribute MUST be ignored.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.
**getEnabled:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**image:** An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**showItemImage:** A Boolean attribute that specifies whether this control displays icons on its selection items.

**getItemCount:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the number of dynamically created selection items in this control. If this attribute is omitted, the control displays the selection items that are specified as child elements. If this attribute is omitted and no child items are specified, the control is empty. If both this attribute and child items are specified, the child items are ignored.

**getItemLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the labels of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display labels.

**getItemScreentip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display tooltips.

**getItemSupertip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTips of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display Enhanced ScreenTips.

**getItemImage:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the icons of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items do not display icons.

**getItemID:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the identifiers of dynamically created selection items in this control. If this attribute is omitted, dynamically created selection items have empty identifiers.

**sizeString:** An ST_String attribute, as specified in section 2.4.24, that specifies a string of which the size is used to determine the width of the text input area of this control. If this attribute is omitted, the application determines the width of the text input area automatically.

**getSelectedItemID:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the identifier of the item that is selected in this control. The getSelectedItemID and getSelectedItemIndex attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control does not display a selected item.
**getSelectedItemIndex**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the index of the item that is selected in this control. The **getSelectedItemID** and **getSelectedItemIndex** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control does not display a selected item.

**id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a custom control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip**: An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip**: An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **String** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage**: A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_GalleryRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_DropDownRegular">
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
      <xsd:attribute name="columns" type="ST_GalleryRowColumnCount" use="optional"/>
      <xsd:attribute name="rows" type="ST_GalleryRowColumnCount" use="optional"/>
      <xsd:attribute name="itemWidth" type="ST_GalleryItemWidthHeight" use="optional"/>
      <xsd:attribute name="itemHeight" type="ST_GalleryItemWidthHeight" use="optional"/>
      <xsd:attribute name="getItemWidth" type="ST_Delegate" use="optional"/>
      <xsd:attribute name="getItemHeight" type="ST_Delegate" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.47 CT_Group


Referenced by: CT_Tab

Specifies a grouping of controls within a ribbon tab.

Child Elements:

control: A CT_ControlClone element, as specified in section 2.3.36.

labelControl: A CT_LabelControl element, as specified in section 2.3.54.

button: A CT_Button element, as specified in section 2.3.24.

toggleButton: A CT_ToggleButton element, as specified in section 2.3.83.

checkBox: A CT_CheckBox element, as specified in section 2.3.27.

editBox: A CT_EditBox element, as specified in section 2.3.44.

comboBox: A CT_ComboBox element, as specified in section 2.3.28.

dropDown: A CT_DropDownRegular element, as specified in section 2.3.41.

gallery: A CT_Gallery element, as specified in section 2.3.45.

menu: A CT_Menu element, as specified in section 2.3.56.

dynamicMenu: A CT_DynamicMenu element, as specified in section 2.3.42.

splitButton: A CT_SplitButton element, as specified in section 2.3.69.

box: A CT_Box element.

buttonGroup: A CT_ButtonGroup element, as specified in section 2.3.25.

separator: A CT_Separator element, as specified in section 2.3.67.

dialogBoxLauncher: A CT_DialogLauncher element, as specified in section 2.3.40.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.
**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the **Enhanced ScreenTip** of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a string that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**autoScale:** A **Boolean** attribute that specifies whether the contents of this group control automatically resize as the size of the containing window changes. If this attribute is omitted, the contents of this group control will not automatically resize.

**centerVertically:** A **Boolean** attribute that specifies whether the contents of this group control are centered vertically within the container. If this attribute is omitted, the contents of this group control will be top-aligned rather than vertically centered.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Group">
    <xsd:sequence>
        <xsd:sequence>
            <xsd:choice minOccurs="0" maxOccurs="1000">
                <xsd:group ref="EG_Controls"/>
                <xsd:element name="separator" type="CT_Separator"/>
            </xsd:choice>
        </xsd:sequence>
        <xsd:element name="dialogBoxLauncher" type="CT_DialogLauncher" minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="AG_IDAttributes"/>
    <xsd:attributeGroup ref="AG_Label"/>
    <xsd:attributeGroup ref="AG_Image"/>
    <xsd:attributeGroup ref="AG_PositionAttributes"/>
    <xsd:attributeGroup ref="AG_Screentip"/>
    <xsd:attributeGroup ref="AG_Visible"/>
    <xsd:attributeGroup ref="AG_Keytip"/>
    <xsd:attribute name="autoScale" type="xsd:boolean" use="optional"/>
    <xsd:attribute name="centerVertically" type="xsd:boolean" use="optional"/>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.48 CT_GroupBox


Referenced by: CT_GroupControls, CT_LayoutContainer

Specifies a grouping container that visually groups child controls together.

Child Elements:

**button:** A CT_BackstageGroupButton element, as specified in section 2.3.10.

**checkBox:** A CT_BackstageCheckBox element, as specified in section 2.3.3.

**editBox:** A CT_BackstageEditBox element, as specified in section 2.3.7.

**dropDown:** A CT_BackstageDropDown element, as specified in section 2.3.6.

**radioGroup:** A CT_RadioGroup element, as specified in section 2.3.65.

**comboBox:** A CT_BackstageComboBox element, as specified in section 2.3.5.

**hyperlink:** A CT_Hyperlink element, as specified in section 2.3.51.

**labelControl:** A CT_BackstageLabelControl element, as specified in section 2.3.13.

**groupBox:** A CT_GroupBox element, as specified in section 2.3.48.

**layoutContainer:** A CT_LayoutContainer element, as specified in section 2.3.55.

**imageControl:** A CT_ImageControl element, as specified in section 2.3.52.

Attributes:

**id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**expand:** An ST_expand attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

**label:** An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_GroupBox">
  <xsd:sequence>
    <xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.49 CT_GroupControls

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_BackstageGroup](#)

Specifies a list of controls that are child controls of a Backstage group control.

**Child Elements:**

- **button:** A CT_BackstageGroupButton element, as specified in section 2.3.10.
- **checkBox:** A CT_BackstageCheckBox element, as specified in section 2.3.3.
- **editBox:** A CT_BackstageEditBox element, as specified in section 2.3.7.
- **dropDown:** A CT_BackstageDropDown element, as specified in section 2.3.6.
- **radioGroup:** A CT_RadioGroup element, as specified in section 2.3.65.
- **comboBox:** A CT_BackstageComboBox element, as specified in section 2.3.5.
- **hyperlink:** A CT_Hyperlink element, as specified in section 2.3.51.
- **labelControl:** A CT_BackstageLabelControl element, as specified in section 2.3.13.
- **groupBox:** A CT_GroupBox element, as specified in section 2.3.48.
- **layoutContainer:** A CT_LayoutContainer element, as specified in section 2.3.55.
- **imageControl:** A CT_ImageControl element, as specified in section 2.3.52.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_GroupControls">
  <xsd:choice minOccurs="0" maxOccurs="1000">
    <xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
  </xsd:choice>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.50 CT_HeaderGroup


Specifies a list of button controls.

Child Elements:

button: A CT_BackstageRegularButton element, as specified in section 2.3.20.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_HeaderGroup">
  <xsd:sequence>
    <xsd:element name="button" type="CT_BackstageRegularButton" minOccurs="1" maxOccurs="100"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.51 CT_Hyperlink

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a hyperlink control.

**Attributes:**

- **id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **alignLabel:** An ST_alignLabel attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

- **expand:** An ST_expand attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

- **enabled:** A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **visible:** A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **getVisible:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **keytip:** An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

- **getKeytip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.
**label:** An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**onAction:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**image:** An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip:** An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**target:** An ST_String attribute, as specified in section 2.4.24, that specifies the URL for this hyperlink control. The target, getTarget, and onAction attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the hyperlink control does not perform any action when invoked.

**getTarget:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the target URL for this hyperlink control. The target, getTarget, and onAction attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the hyperlink control does not perform any action when invoked.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Hyperlink">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.52 CT_ImageControl

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_GroupBox, CT_LayoutContainer, CT_GroupControls

Specifies a control that displays an image.

**Attributes:**

- **id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **enabled:** A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **visible:** A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **getVisible:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **image:** An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
imageMso: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

altText: An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the alternative text of this control. The **altText** and **getAltText** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, alternative text is not displayed.

getAltText: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the alternative text of this control. The **altText** and **getAltText** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, alternative text is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ImageControl">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_AltText"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.53 CT_Item

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery

Specifies an item in a selection-type control.

**Attributes:**

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**label:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the label of this control. If this attribute is not specified, a label is not displayed.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image** and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image** and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an icon is not displayed.
screentip: An ST_String attribute that specifies a string that is displayed as the tooltip for this control. If this attribute is not specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. If this attribute is not specified, an Enhanced ScreenTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Item">
  <xsd:attribute name="id" type="ST_UniqueID" use="optional"/>
  <xsd:attribute name="label" type="ST_String" use="optional"/>
  <xsd:attribute name="image" type="ST_Uri" use="optional"/>
  <xsd:attribute name="imageMso" type="ST_ID" use="optional"/>
  <xsd:attribute name="screentip" type="ST_String" use="optional"/>
  <xsd:attribute name="supertip" type="ST_String" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.54 CT_LabelControl


*Referenced by:* CT_Box, CT_Group

Specifies a control that displays a string of text.

*Attributes:*

**id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip:** An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.
**getSupertip:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The `supertip` and `getSupertip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled:** A `Boolean` attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label:** An `ST_String` attribute that specifies a string that is displayed as the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An `ST_ID` attribute that specifies the identifier of a built-in control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An `ST_ID` attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An `ST_QID` attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A `Boolean` attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**showLabel:** A `Boolean` attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.
**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_LabelControl">
  <xsd:complexContent>
    <xsd:restriction base="CT_Control">
      <xsd:attribute name="image" use="prohibited"/>
      <xsd:attribute name="imageMso" use="prohibited"/>
      <xsd:attribute name="getImage" use="prohibited"/>
      <xsd:attribute name="keytip" use="prohibited"/>
      <xsd:attribute name="getKeytip" use="prohibited"/>
      <xsd:attribute name="showImage" use="prohibited"/>
      <xsd:attribute name="getShowImage" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.55 CT_LayoutContainer

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: **CT_GroupBox**, **CT_GroupControls**

Specifies a grouping container control that aligns child controls horizontally or vertically.

**Child Elements**:

- **button**: A **CT_BackstageGroupButton** element, as specified in section 2.3.10.
- **checkBox**: A **CT_BackstageCheckBox** element, as specified in section 2.3.3.
- **editBox**: A **CT_BackstageEditBox** element, as specified in section 2.3.7.
- **dropDown**: A **CT_BackstageDropDown** element, as specified in section 2.3.6.
- **radioGroup**: A **CT_RadioGroup** element, as specified in section 2.3.65.
- **comboBox**: A **CT_BackstageComboBox** element, as specified in section 2.3.5.
- **hyperlink**: A **CT_Hyperlink** element, as specified in section 2.3.51.
- **labelControl**: A **CT_BackstageLabelControl** element, as specified in section 2.3.13.
- **groupBox**: A **CT_GroupBox** element, as specified in section 2.3.48.
- **layoutContainer**: A **CT_LayoutContainer** element, as specified in section 2.3.55.
- **imageControl**: A **CT_ImageControl** element, as specified in section 2.3.52.

**Attributes**:

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

align: An ST_align attribute, as specified in section 2.4.1, that specifies where child controls are aligned within this container control. If this attribute is omitted, the child controls are positioned at the top left of the container.

expand: An ST_expand1 attribute, as specified in section 2.4.4, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

layoutChildren: An ST_layoutChildren attribute, as specified in section 2.4.20, that specifies the direction that child controls are laid out within this container. If this attribute is omitted, child controls are laid out horizontally.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_LayoutContainer">
  <xsd:sequence>
    <xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attribute name="align" use="optional" type="ST_align"/>
  <xsd:attribute name="expand" use="optional" type="ST_expand1"/>
  <xsd:attribute name="layoutChildren" use="optional" type="ST_layoutChildren"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.56 CT_Menu

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_Box, CT_Group

Specifies a standard drop-down menu control.

**Child Elements:**

- control: A CT_ControlCloneRegular element, as specified in section 2.3.38.
- button: A CT_ButtonRegular element, as specified in section 2.3.26.
- checkBox: A CT_CheckBox element, as specified in section 2.3.27.
- gallery: A CT_GalleryRegular element, as specified in section 2.3.46.
- toggleButton: A CT_ToggleButtonRegular element, as specified in section 2.3.84.
- menuSeparator: A CT_MenuSeparator element, as specified in section 2.3.59.
- splitButton: A CT_SplitButtonRegular element, as specified in section 2.3.71.
- menu: A CT_MenuRegular element.
**dynamicMenu**: A `CT_DynamicMenuRegular` element, as specified in section 2.3.43.

**Attributes:**

**size**: An `ST_Size` attribute, as specified in section 2.4.23, that specifies the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**getSize**: An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**itemSize**: An `ST_ItemSize` attribute, as specified in section 2.4.18, that specifies the size of the items in this menu control.

**description**: An `ST_LongString` attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The `description` and `getDescription` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The `description` and `getDescription` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**id**: An `ST_UniqueID` attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An `ST_String` attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**image**: An `ST_Uri` attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

g getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

g getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Menu">
  <xsd:complexContent>
    <xsd:extension base="CT_MenuRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
      <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.57 CT_MenuRegular


Referenced by: CT_SplitButtonRegular, CT_Menu, CT_SplitButton, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a menu control that is of fixed size because of its location. It is otherwise identical to the CT_Menu complex type.
Child Elements:

**control**: A `CT_ControlCloneRegular` element, as specified in section 2.3.38.

**button**: A `CT_ButtonRegular` element, as specified in section 2.3.26.

**checkBox**: A `CT_CheckBox` element, as specified in section 2.3.27.

**gallery**: A `CT_GalleryRegular` element, as specified in section 2.3.46.

**toggleButton**: A `CT_ToggleButtonRegular` element, as specified in section 2.3.84.

**menuSeparator**: A `CT_MenuSeparator` element, as specified in section 2.3.59.

**splitButton**: A `CT_SplitButtonRegular` element, as specified in section 2.3.71.

**menu**: A `CT_MenuRegular` element.

**dynamicMenu**: A `CT_DynamicMenuRegular` element, as specified in section 2.3.43.

Attributes:

**itemSize**: An `ST_ItemSize` attribute, as specified in section 2.4.18, that specifies the size of the items in this menu control.

**description**: An `ST_LongString` attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getdescription**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**id**: An `ST_UniqueID` attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An `ST_String` attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**image**: An `ST_Uri` attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip:** An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the condition defaults to being enabled.

**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these
attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage**: A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_MenuRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="1000">
          <xsd:group ref="EG_MenuControlsBase"/>
          <xsd:group ref="EG_MenuOrSplitButtonRegular"/>
        </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_IDAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.58 CT_MenuRoot


Referenced by: menu

Specifies the root element of the customization XML used by the CT_DynamicMenu complex type, as specified in section 2.3.42.

Child Elements:

control: A CT_ControlCloneRegular element, as specified in section 2.3.38.

clickButton: A CT_ClickButtonRegular element, as specified in section 2.3.26.

checkBox: A CT_CheckBox element, as specified in section 2.3.27.

gallery: A CT_GalleryRegular element, as specified in section 2.3.46.

toggleButton: A CT_ToggleButtonRegular element, as specified in section 2.3.46.

menuSeparator: A CT_MenuSeparator element, as specified in section 2.3.59.

splitButton: A CT_SplitButtonRegular element, as specified in section 2.3.71.

menu: A CT_MenuRegular element.

dynamicMenu: A CT_DynamicMenuRegular element, as specified in section 2.3.43.

Attributes:

title: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as a title of the menu. The title and getTitle attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

gETTitle: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the title of the menu. The title and getTitle attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

itemSize: An ST_ItemSize attribute, as specified in section 2.4.18, that specifies the size of the menu items in this control.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

<xsd:complexType name="CT_MenuRoot">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:group ref="EG_MenuControlsBase"/>
      <xsd:group ref="EG_MenuOrSplitButtonRegular"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_Title"/>
  <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.59 CT_MenuSeparator


Referenced by: CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_MenuRoot

Specifies a horizontal menu separator control that optionally displays a textual heading.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

title: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as a title of the menu. The title and getTitle attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

getTitle: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the title of the menu. The title and getTitle attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_MenuSeparator">
```

[MS-CUSTOMUI2] - v20210422
Custom UI XML Markup Version 2 Specification
Copyright © 2021 Microsoft Corporation
Release: April 22, 2021
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
<xsd:attributeGroup ref="AG_Title"/>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.60 CT_MenuSeparatorNoTitle


Referenced by: CT_ContextMenu

Specifies a menu separator control that visually separates other controls.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

<xsd:complexType name="CT_MenuSeparatorNoTitle"/>
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.61 CT_MenuWithTitle


Referenced by: CT_SplitButtonWithTitle

Specifies a menu control that optionally displays a title string at the top of the menu. It is otherwise identical to the CT_Menu complex type.

Child Elements:

control: A CT_ControlCloneRegular element, as specified in section 2.3.38.

button: A CT_ButtonRegular element, as specified in section 2.3.26.

checkBox: A CT_CheckBox element, as specified in section 2.3.27.

gallery: A CT_GalleryRegular element, as specified in section 2.3.46.

toggleButton: A CT_ToggleButtonRegular element, as specified in section 2.3.84.

menuSeparator: A CT_MenuSeparator element, as specified in section 2.3.59.

splitButton: A CT_SplitButtonWithTitle element, as specified in section 2.3.73.

menu: A CT_MenuWithTitle element.

dynamicMenu: A CT_DynamicMenuRegular element, as specified in section 2.3.43.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The Id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

itemSize: An ST_ItemSize attribute, as specified in section 2.4.18, that specifies the size of the items in this menu control.

title: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as a title of the menu. The title and getTitle attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.
**getTitle**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the title of the menu. The **title** and **getTitle** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the menu does not display a title.

**image**: An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage**: An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip**: An **ST_String** attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The **screentip** and **getScreentip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip**: An **ST_String** attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The **supertip** and **getSupertip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**enabled**: A Boolean attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertBeforeMso**: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A Boolean attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A Boolean attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage**: A Boolean attribute that specifies whether the icon for this control will be displayed. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The `showImage` and `getShowImage` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_MenuWithTitle">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:sequence>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```
<xsd:choice minOccurs="0" maxOccurs="1000">
    <xsd:group ref="EG_MenuControlsBase"/>
    <xsd:group ref="EG_MenuOrSplitButtonWithTitle"/>
</xsd:choice>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
<xsd:attributeGroup ref="AG_Title"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.62 CT_PrimaryItem


Referenced by: CT_BackstageGroup

Specifies a container for the primary UI of the containing Backstage group control.

Child Elements:

button: A CT_BackstageRegularButton element, as specified in section 2.3.20.

menu: A CT_BackstagePrimaryMenu element, as specified in section 2.3.19.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

<xsd:complexType name="CT_PrimaryItem">
    <xsd:choice minOccurs="1" maxOccurs="1">
        <xsd:element name="button" type="CT_BackstageRegularButton" minOccurs="0" maxOccurs="1"/>
        <xsd:element name="menu" type="CT_BackstagePrimaryMenu" minOccurs="0" maxOccurs="1"/>
    </xsd:choice>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.63 CT_Qat


Referenced by: CT_Ribbon

Specifies the Quick Access Toolbar. This element MUST NOT be specified unless the startFromScratch attribute on the containing ribbon element is set to true.

Child Elements:

sharedControls: A CT_QatItems element, as specified in section 2.3.64, that specifies the list of controls that are shared among all documents.

documentControls: A CT_QatItems element that specifies the list of controls that are specific to the current document.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.
2.3.64 CT_QatItems


Referenced by: CT_Qat

Specifies a grouping of controls on the Quick Access Toolbar.

Child Elements:

- control: A CT_ControlCloneQat element, as specified in section 2.3.37.
- button: A CT_ButtonRegular element, as specified in section 2.3.26.
- separator: A CT_Separator element, as specified in section 2.3.67.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_QatItems">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:element name="control" type="CT_ControlCloneQat"/>
      <xsd:element name="button" type="CT_ButtonRegular"/>
      <xsd:element name="separator" type="CT_Separator"/>
    </xsd:choice>
  </xsd:sequence>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.65 CT_RadioGroup


Referenced by: CT_GroupControls, CT_GroupBox, CT_LayoutContainer

Specifies a group of radio buttons.

Child Elements:

- radioButton: A CT_BackstageItem element, as specified in section 2.3.12.

Attributes:

- id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

alignLabel: An ST_alignLabel attribute, as specified in section 2.4.2, that specifies how the label of this control is aligned relative to the control. If this attribute is omitted, the label defaults to the top left alignment.

expand: An ST_expand attribute, as specified in section 2.4.13, that specifies how this control expands within its container. If this attribute is omitted, the control does not expand.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getSelectedItemIndex: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine which radio button is selected, identified by index. If this attribute is omitted, the first radio button is selected by default.

getItemCount: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the number of radio buttons in this group. If this attribute is omitted, the control displays the radio buttons which are specified as child elements. If this attribute is omitted and no child
elements are specified, the radio group is empty. If both this attribute and child elements are specified, the child elements are ignored.

**get[,]ItemLabel**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of a radio button, identified by index. If this attribute is omitted, dynamically created radio buttons will not display labels.

**get[,]ItemID**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the identifier of a radio button, identified by index. If this attribute is omitted, dynamically created radio buttons will have blank identifiers.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_RadioGroup">
  <xsd:sequence>
    <xsd:element name="radioButton" type="CT_BackstageItem" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attribute name="getSelectedItemIndex" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemCount" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemLabel" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.66 CT_Ribbon


*Referenced by:* CT_CustomUI

Specifies a reference to the main *ribbon* of the application.

**Child Elements:**

- **qat**: A CT_Qat element, as specified in section 2.3.63.
- **tabs**: A CT_Tabs element, as specified in section 2.3.75.
- **contextualTabs**: A CT_ContextualTabs element, as specified in section 2.3.33.

**Attributes:**

- **startFromScratch**: A Boolean attribute that specifies that the ribbon is reduced to a minimal set of features from which a completely custom ribbon can be built. If this attribute is omitted, the ribbon is unchanged.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Ribbon">
  <xsd:all>
```
<xsd:element name="qat" type="CT_Qat" minOccurs="0" maxOccurs="1">
    <xsd:unique name="qatControls">
        <xsd:selector xpath="*"/>
        <xsd:field xpath="@id"/>
    </xsd:unique>
</xsd:element>
</xsd:complexType>
</xsd:schema>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.67 CT_Separator

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_ButtonGroup, CT_Group, CT_QatItems

Specifies a vertical line separator control.

**Attributes**:

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

- **insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Separator">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.68 CT_SimpleGroups


Referenced by: CT_BackstageTab

Specifies a list of Backstage group controls.

**Child Elements:**

**group:** A CT_BackstageGroup element, as specified in section 2.3.9.

**taskGroup:** A CT_TaskGroup element, as specified in section 2.3.80.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_SimpleGroups">
  <xsd:choice minOccurs="0" maxOccurs="1000">
    <xsd:group ref="EG_SimpleGroups"/>
  </xsd:choice>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.69 CT_SplitButton


Referenced by: CT_Box, CT_Group

Specifies a split button control that contains a primary button, as well as a drop-down menu.

**Child Elements:**

**button:** A CT_VisibleButton element, as specified in section 2.3.85.
**ToggleButton**: A `CT_VisibleToggleButton` element, as specified in section 2.3.86.

**Menu**: A `CT_MenuRegular` element, as specified in section 2.3.57.

**Attributes**:

**size**: An `ST_Size` attribute, as specified in section 2.4.23, that specifies the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**getSize**: An `ST_Delegate` attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The `size` and `getSize` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

**enabled**: A Boolean attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An `ST_Delegate` attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**id**: An `ST_UniqueID` attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag**: An `ST_String` attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso**: An `ST_ID` attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**insertAfterMso**: An `ST_ID` attribute that specifies the identifier of a built-in control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An `ST_ID` attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An `ST_QID` attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertBeforeQ:** An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_SplitButton">
  <xsd:complexContent>
    <xsd:extension base="CT_SplitButtonRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.70 CT_SplitButtonBase


Referenced by: **CT_SplitButtonRestricted**

Abstract base type that specifies common attributes for split button type controls.

**Attributes:**

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.
**getEnabled:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**id:** An *ST_UniqueID* attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An *ST_QID* attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An *ST_String* attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An *ST_ID* attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**image:** An *ST_Uri* attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An *ST_ID* attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An *ST_Delegate* attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The `image`, `getImage`, and `imageMso` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**screentip:** An *ST_String* attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The `screentip` and `getScreentip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The `screentip` and `getScreentip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An *ST_String* attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The `supertip` and `getSupertip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The `supertip` and `getSupertip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**label:** An *ST_String* attribute that specifies a string that is displayed as the label of this control. The `label` and `getLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.
getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to
determine the label of this control. The label and getLabel attributes are mutually exclusive. They
MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this
control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ
attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is
specified, the controls will be appended to the existing set of controls, in the order they are defined in
the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which
this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and
insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these
attributes is specified, the controls will be appended to the existing set of controls, in the order they
are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified
identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ,
insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used
together. If none of these attributes is specified, the controls will be appended to the existing set of
controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which
this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and
insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these
attributes is specified, the controls will be appended to the existing set of controls, in the order they
are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and
getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is
specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to
determine the visibility state of this control. The visible and getVisible attributes are mutually
exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to
being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is
displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to
determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed.
The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used
together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is
called to determine whether to display the label of this control. The showLabel and getShowLabel
attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

showImage: A Boolean attribute that specifies whether the icon for this control will be displayed.
The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used
together. If neither attribute is specified, the icon of the control will be displayed.

getShowImage: An ST_Delegate attribute that specifies the name of a callback function that is
called to determine whether to display the icon of this control. The showImage and getShowImage

attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_SplitButtonBase">
  <xsd:complexContent>
    <xsd:extension base="CT_Control">
      <xsd:attributeGroup ref="AG_Enabled"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.71 CT_SplitButtonRegular


Referenced by: CT_MenuRegular, CT_Menu, CT_SplitButton, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a split button control that is of fixed size because of its location. It is otherwise identical to the CT_SplitButton complex type.

Child Elements:

- **button**: A CT_VisibleButton element, as specified in section 2.3.85.
- **toggleButton**: A CT_VisibleToggleButton element, as specified in section 2.3.86.
- **menu**: A CT_MenuRegular element, as specified in section 2.3.57.

Attributes:

- **enabled**: A Boolean attribute that specifies the enabled state of this control. The *enabled* and *getEnabled* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The *enabled* and *getEnabled* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **id**: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The *id*, *idQ*, and *idMso* attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The *id*, *idQ*, and *idMso* attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.
idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The _id_, _idQ_, and _idMso_ attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The _visible_ and _getVisible_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The _visible_ and _getVisible_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a _string_ that is displayed as the _KeyTip_ for this control. The _keytip_ and _getKeytip_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The _keytip_ and _getKeytip_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The _showLabel_ and _getShowLabel_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The _showLabel_ and _getShowLabel_ attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_SplitButtonRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_SplitButtonRestricted">
      ...
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.72 CT_SplitButtonRestricted

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_SplitButtonRegular](#), [CT_SplitButtonWithTitle](#)

Abstract base type that restricts several attributes from being specified on split button controls. The restricted attributes are instead provided by the primary button within the split button control.

**Attributes:**

- **enabled:** A **Boolean** attribute that specifies the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The `enabled` and `getEnabled` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

- **id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary **string** that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The `id`, `idQ`, and `idMso` attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

- **insertAfterMso:** An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeMso:** An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these
attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The `insertAfterMso`, `insertAfterQ`, `insertBeforeMso`, and `insertBeforeQ` attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A Boolean attribute that specifies the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The `visible` and `getVisible` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The `keytip` and `getKeytip` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A Boolean attribute that specifies whether the label for this control will be displayed. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The `showLabel` and `getShowLabel` attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xs:complexType name="CT_SplitButtonRestricted">
  <xs:complexContent>
    <xs:restriction base="CT_SplitButtonBase">
      <xs:attribute name="label" use="prohibited"/>
      <xs:attribute name="getLabel" use="prohibited"/>
      <xs:attribute name="screentip" use="prohibited"/>
      <xs:attribute name="getScreentip" use="prohibited"/>
      <xs:attribute name="supertip" use="prohibited"/>
      <xs:attribute name="getSupertip" use="prohibited"/>
      <xs:attribute name="image" use="prohibited"/>
      <xs:attribute name="imageMso" use="prohibited"/>
      <xs:attribute name="getImage" use="prohibited"/>
      <xs:attribute name="showImage" use="prohibited"/>
      <xs:attribute name="getShowImage" use="prohibited"/>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.73 CT_SplitButtonWithTitle


Referenced by: CT_MenuWithTitle

Specifies a split button control that optionally displays a title string at the top of the drop-down menu. It is otherwise identical to the CT_SplitButton complex type.

Child Elements:

button: A CT_VisibleButton element, as specified in section 2.3.85.
toggleButton: A CT_VisibleToggleButton element, as specified in section 2.3.86.

menu: A CT_MenuWithTitle element, as specified in section 2.3.61.

Attributes:

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_SplitButtonWithTitle">
  <xsd:complexContent>
    <xsd:extension base="CT_SplitButtonRestricted">
      <xsd:sequence minOccurs="0">
        <xsd:choice minOccurs="0">
          <xsd:element name="button" type="CT_VisibleButton"/>
          <xsd:element name="toggleButton" type="CT_VisibleToggleButton"/>
        </xsd:choice>
        <xsd:element name="menu" type="CT_MenuWithTitle"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.74 CT_Tab

**Target namespace:** http://schemas.microsoft.com_office/2009/07/customui
Referenced by: **CT_Tabs, CT_TabSet**

Specifies a **ribbon** tab control.

**Child Elements:**

- **group**: A **CT_Group** element, as specified in section 2.3.47.

**Attributes:**

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **idMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

- **label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

- **getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

- **insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

ggetKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```
<xsd:complexType name="CT_Tab">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="100">
      <xsd:element name="group" type="CT_Group"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.75 CT_Tabs


Referenced by: CT_Ribbon

Specifies a list of tab controls.

Child Elements:

**tab**: A CT_Tab element, as specified in section 2.3.74.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```
<xsd:complexType name="CT_Tabs">
  <xsd:sequence>
    <xsd:element name="tab" type="CT_Tab" minOccurs="1" maxOccurs="100"/>
  </xsd:sequence>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.3.76 CT_TabSet


Referenced by: CT_ContextualTabs

Specifies a list of contextual tab controls.

Child Elements:

**tab**: A CT_Tab element, as specified in section 2.3.74.

Attributes:

**idMso**: An ST_ID attribute, as specified in section 2.4.17, that specifies the ID of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**visible**: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_TabSet">
  <xsd:sequence>
    <xsd:element name="tab" type="CT_Tab" minOccurs="0" maxOccurs="50"/>
  </xsd:sequence>
  <xsd:attribute name="idMso" type="ST_ID" use="required"/>
  <xsd:attributeGroup ref="AG_Visible"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.77 CT_TaskFormGroup


Referenced by: CT_BackstageGroups

Specifies a TaskGroup control that additionally displays an associated Backstage group control for each of its selectable options.

Child Elements:

**category**: A CT_TaskFormGroupCategory element, as specified in section 2.3.78.

Attributes:

**id**: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.
idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

helperText: An ST_LongString attribute, as specified in section 2.4.21, that specifies additional text that explains the functionality of this Backstage group control. The helperText and getHelperText attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.

getHelperText: An ST_Delegate attribute that specifies the name of a callback that is called to determine the additional text for this Backstage group control. The helperText and getHelperText attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

getShowLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

allowedTaskSizes: An ST_TaskSizes attribute, as specified in section 2.4.26, that specifies the allowed sizes for tasks within this control.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_TaskFormGroup">
  <xsd:sequence>
    <xsd:element name="category" type="CT_TaskFormGroupCategory" minOccurs="0" maxOccurs="100"/>
  </xsd:sequence>
</xsd:complexType>
```
2.3.78 CT_TaskFormGroupCategory


Referenced by: CT_TaskFormGroup

Specifies a grouping of child controls within a TaskGroup control.

Child Elements:

task: A CT_TaskFormGroupTask element, as specified in section 2.3.79.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_TaskFormGroupCategory">
  <xsd:sequence>
    <xsd:element name="task" type="CT_TaskFormGroupTask" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.79 CT_TaskFormGroupTask


Referenced by: CT_TaskFormGroupCategory

Specifies a selectable option within a TaskGroup control. When this control is selected, the child Backstage group control is displayed.

Child Elements:

group: A CT_BackstageGroup element, as specified in section 2.3.9.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI
The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**insertAfterMso:** An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**image:** An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**enabled:** A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.
**label:** An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**visible:** A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**description:** An **ST_LongString** attribute, as specified in section [2.4.21](#), that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**keytip:** An **ST_Keytip** attribute, as specified in section [2.4.19](#), that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2](#) section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_TaskFormGroupTask">
  <xsd:sequence>
    <xsd:element name="group" type="CT_BackstageGroup" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Description"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
```

See section [5.1](#) for the full W3C XML Schema ([XMLSCHEMA1/2](#) section 2.1).

### 2.3.80 CT_TaskGroup


*Referenced by:* [CT_BackstageGroups](#), [CT_SimpleGroups](#)
Specifies a **TaskGroup** control.

**Child Elements:**

- **category**: A **CT_TaskGroupCategory** element, as specified in section 2.3.81.

**Attributes:**

- **id**: An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The *id*, *idQ*, and *idMso* attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **idQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The *id*, *idQ*, and *idMso* attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

- **tag**: An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

- **idMso**: An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The *id*, *idQ*, and *idMso* attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

- **insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

- **label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The *label* and *getLabel* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

- **getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The *label* and *getLabel* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

- **visible**: A **Boolean** attribute that specifies the visibility state of this control. The *visible* and *getVisible* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.
**getVisible:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**helperText:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies additional text that explains the functionality of this Backstage group control. The **helperText** and **getHelperText** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.

**getHelperText:** An **ST_Delegate** attribute that specifies the name of a callback that is called to determine the additional text for this Backstage group control. The **helperText** and **getHelperText** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the Backstage group control does not display additional text.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**allowedTaskSizes:** An **ST_TaskSizes** attribute, as specified in section 2.4.26, that specifies the allowed sizes for tasks within this control.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xs:complexType name="CT_TaskGroup">
  <xs:sequence>
    <xs:element name="category" type="CT_TaskGroupCategory" minOccurs="0" maxOccurs="100"/>
  </xs:sequence>
  <xs:attributeGroup ref="AG_IDAttributes"/>
  <xs:attributeGroup ref="AG_PositionAttributes"/>
  <xs:attributeGroup ref="AG_Label"/>
  <xs:attributeGroup ref="AG_Visible"/>
  <xs:attributeGroup ref="AG_HelperText"/>
  <xs:attributeGroup ref="AG_ShowLabel"/>
  <xs:attribute name="allowedTaskSizes" type="ST_TaskSizes" use="optional"/>
</xs:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.81 CT_TaskGroupCategory

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_TaskGroup](#)

Specifies a grouping of child controls within a [TaskGroup](#) control.

**Child Elements:**

- **task:** A [CT_TaskGroupTask](#) element, as specified in section 2.3.82.

**Attributes:**
id: An **ST_UniqueID** attribute, as specified in section **2.4.27**, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An **ST_QID** attribute, as specified in section **2.4.22**, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An **ST_String** attribute, as specified in section **2.4.24**, that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An **ST_ID** attribute, as specified in section **2.4.17**, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

insertAfterMso: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An **ST_QID** attribute, as specified in section **2.4.22**, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

visible: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

label: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.
<xsd:complexType name="CT_TaskGroupCategory">
  <xsd:sequence>
    <xsd:element name="task" type="CT_TaskGroupTask" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.82 CT_TaskGroupTask


Referenced by: CT_TaskGroupCategory

Specifies a selectable choice within a TaskGroup control.

Attributes:

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and
insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

isDefinitive: A Boolean attribute that specifies that the Backstage closes when this command is invoked. If this attribute is omitted, its value defaults to false.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

visible: A Boolean attribute that specifies the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

getVisible: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The visible and getVisible attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.
keytip: An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_TaskGroupTask">
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Definitive"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Description"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.83 CT_ToggleButton

Target namespace: http://schemas.microsoft.com офис/2009/07/customui

Referenced by: CT_Box, CT_Group

Specifies a standard toggle button control.

Attributes:

size: An ST_Size attribute, as specified in section 2.4.23, that specifies the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

getSize: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the size of this control. The size and getSize attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to the normal size.

getPressed: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.
**description:** An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**image:** An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**id:** An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

**screentip:** An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**getScreentip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

**supertip:** An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

**getSupertip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.
**label**: An **ST_String** attribute that specifies a string that is displayed as the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**getLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the label of this control. The **label** and **getLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

**insertAfterMso**: An **ST_ID** attribute that specifies the identifier of a built-in control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso**: An **ST_ID** attribute that specifies the identifier of a built-in control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ**: An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ**: An **ST_QID** attribute that specifies the qualified identifier of a control before which this control will be inserted. The **insertAfterMso**, **insertAfterQ**, **insertBeforeMso**, and **insertBeforeQ** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible**: A **Boolean** attribute that specifies the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The **visible** and **getVisible** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip**: An **ST_Keytip** attribute, as specified in section 2.4.19, that specifies a **string** that is displayed as the **KeyTip** for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The **keytip** and **getKeytip** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel**: A **Boolean** attribute that specifies whether the label for this control will be displayed. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel**: An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.
**showImage**: A Boolean attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_ToggleButton">
  <xsd:complexContent>
    <xsd:extension base="CT_ToggleButtonRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.84 CT_ToggleButtonRegular

**Target namespace**: http://schemas.microsoft.com/office/2009/07/customui

**Referenced by**: CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_ButtonGroup, CT_ContextMenu, CT_MenuRoot

Specifies a toggle button control that has a fixed size because of its location. This type is otherwise identical to the **CT_ToggleButton** type.

**Attributes**:

**getPressed**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

**onAction**: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**enabled**: A Boolean attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**description**: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription**: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.
image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

ggetImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

gScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

gSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

gLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is
specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeMso:** An *ST_ID* attribute that specifies the identifier of a built-in control before which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertAfterQ:** An *ST_QID* attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**insertBeforeQ:** An *ST_QID* attribute that specifies the qualified identifier of a control before which this control will be inserted. The *insertAfterMso*, *insertAfterQ*, *insertBeforeMso*, and *insertBeforeQ* attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**visible:** A *Boolean* attribute that specifies the visibility state of this control. The *visible* and *getVisible* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**getVisible:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine the visibility state of this control. The *visible* and *getVisible* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being visible.

**keytip:** An *ST_Keytip* attribute, as specified in section 2.4.19, that specifies a *string* that is displayed as the *KeyTip* for this control. The *keytip* and *getKeytip* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The *keytip* and *getKeytip* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A *Boolean* attribute that specifies whether the label for this control will be displayed. The *showLabel* and *getShowLabel* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The *showLabel* and *getShowLabel* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A *Boolean* attribute that specifies whether the icon for this control will be displayed. The *showImage* and *getShowImage* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An *ST_Delegate* attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The *showImage* and *getShowImage* attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.
<xsd:complexContent>
  <xsd:extension base="CT_ButtonRegular">
    <xsd:attributeGroup ref="AG_Pressed"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.3.85 CT_VisibleButton


Referenced by: CT_SplitButtonRegular, CT_SplitButtonWithTitle, CT_SplitButton

Specifies a button control that is visible because of its location. This type is otherwise identical to the CT_Button type.

Attributes:

onAction: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

enabled: A Boolean attribute that specifies the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

getEnabled: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The enabled and getEnabled attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

description: An ST_LongString attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

getDescription: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The description and getDescription attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

image: An ST_Uri attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

imageMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

getImage: An ST_Delegate attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The image, getImage, and imageMso attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.
id: An ST_UniqueID attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

idQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

tag: An ST_String attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

idMso: An ST_ID attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The id, idQ, and idMso attributes are mutually exclusive. Exactly one of these attributes MUST be specified.

screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

g getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.
**insertBeforeQ:** An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

**keytip:** An ST_Keytip attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**getKeytip:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

**showLabel:** A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**getShowLabel:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A Boolean attribute that specifies whether the icon for this control will be displayed. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An ST_Delegate attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The showImage and getShowImage attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_VisibleButton">
  <xsd:complexContent>
    <xsd:restriction base="CT_ButtonRegular">
      <xsd:attribute name="visible" use="prohibited"/>
      <xsd:attribute name="getVisible" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.3.86 CT_VisibleToggleButton

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_SplitButtonRegular, CT_SplitButtonWithTitle, CT_SplitButton

Specifies a toggle button control that is visible because of its location. This type is otherwise identical to the CT_ToggleButtonRegular type.

**Attributes:**
**getPressed:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine whether this toggle button is in the pressed state. If this attribute is omitted, the control defaults to the un-pressed state.

**onAction:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called when this control is invoked by the user.

**enabled:** A **Boolean** attribute that specifies the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**getEnabled:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the enabled state of this control. The **enabled** and **getEnabled** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the control defaults to being enabled.

**description:** An **ST_LongString** attribute, as specified in section 2.4.21, that specifies a string that is displayed as the description of the control in detailed views. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**getDescription:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine the detailed description of this control. The **description** and **getDescription** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a detailed description is not displayed.

**image:** An **ST_Uri** attribute, as specified in section 2.4.28, that specifies the relationship identifier of an image file that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**imageMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in image that is used as the icon for this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**getImage:** An **ST_Delegate** attribute, as specified in section 2.4.12, that specifies the name of a callback function that is called to determine the icon of this control. The **image**, **getImage**, and **imageMso** attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes are specified, an icon is not displayed.

**id:** An **ST_UniqueID** attribute, as specified in section 2.4.27, that specifies the identifier of a custom control. All custom controls MUST have unique identifiers. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**idQ:** An **ST_QID** attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control. This attribute is used to reference controls or containers created by other Custom UI documents. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified on all controls.

**tag:** An **ST_String** attribute, as specified in section 2.4.24, that specifies an arbitrary string that provides additional information about this control during callback function calls. If this attribute is omitted, the value defaults to an empty string.

**idMso:** An **ST_ID** attribute, as specified in section 2.4.17, that specifies the identifier of a built-in control. The **id**, **idQ**, and **idMso** attributes are mutually exclusive. Exactly one of these attributes MUST be specified.
 screentip: An ST_String attribute, as specified in section 2.4.24, that specifies a string that is displayed as the tooltip for this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

g getScreentip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the tooltip of this control. The screentip and getScreentip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a tooltip is not displayed.

supertip: An ST_String attribute that specifies a string that is displayed as the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

g getSupertip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the Enhanced ScreenTip of this control. The supertip and getSupertip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, an Enhanced ScreenTip is not displayed.

label: An ST_String attribute that specifies a string that is displayed as the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

g getLabel: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the label of this control. The label and getLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a label is not displayed.

insertAfterMso: An ST_ID attribute that specifies the identifier of a built-in control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeMso: An ST_ID attribute that specifies the identifier of a built-in control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertAfterQ: An ST_QID attribute, as specified in section 2.4.22, that specifies the qualified identifier of a control after which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

insertBeforeQ: An ST_QID attribute that specifies the qualified identifier of a control before which this control will be inserted. The insertAfterMso, insertAfterQ, insertBeforeMso, and insertBeforeQ attributes are mutually exclusive. They MUST NOT be used together. If none of these attributes is specified, the controls will be appended to the existing set of controls, in the order they are defined in the XML.

keytip: An ST_Keypad attribute, as specified in section 2.4.19, that specifies a string that is displayed as the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

g getKeytip: An ST_Delegate attribute that specifies the name of a callback function that is called to determine the KeyTip for this control. The keytip and getKeytip attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, a KeyTip is not displayed.

showLabel: A Boolean attribute that specifies whether the label for this control will be displayed. The showLabel and getShowLabel attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.
**getShowLabel:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the label of this control. The **showLabel** and **getShowLabel** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the label of the control will be displayed.

**showImage:** A **Boolean** attribute that specifies whether the icon for this control will be displayed. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

**getShowImage:** An **ST_Delegate** attribute that specifies the name of a callback function that is called to determine whether to display the icon of this control. The **showImage** and **getShowImage** attributes are mutually exclusive. They MUST NOT be used together. If neither attribute is specified, the icon of the control will be displayed.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_VisibleToggleButton">
  <xsd:complexContent>
    <xsd:restriction base="CT_ToggleButtonRegular">
      <xsd:attribute name="visible" use="prohibited"/>
      <xsd:attribute name="getVisible" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4 Simple Types

#### 2.4.1 ST_align

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** **CT_LayoutContainer**

Specifies the position where the child controls of a container control are aligned.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>topLeft</td>
<td>Specifies that the child controls are aligned at the top left of the container.</td>
</tr>
<tr>
<td>top</td>
<td>Specifies that the child controls are aligned at the top of the container.</td>
</tr>
<tr>
<td>topRight</td>
<td>Specifies that the child controls are aligned at the top right of the container.</td>
</tr>
<tr>
<td>left</td>
<td>Specifies that the child controls are aligned at the left side of the container.</td>
</tr>
<tr>
<td>center</td>
<td>Specifies that the child controls are aligned at the center of the container.</td>
</tr>
<tr>
<td>right</td>
<td>Specifies that the child controls are aligned at the right side of the container.</td>
</tr>
<tr>
<td>bottomLeft</td>
<td>Specifies that the child controls are aligned at the bottom left of the container.</td>
</tr>
<tr>
<td>bottom</td>
<td>Specifies that the child controls are aligned at the bottom of the container.</td>
</tr>
<tr>
<td>bottomRight</td>
<td>Specifies that the child controls are aligned at the bottom right of the container.</td>
</tr>
</tbody>
</table>
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_align">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="topLeft"/>
        <xsd:enumeration value="top"/>
        <xsd:enumeration value="topRight"/>
        <xsd:enumeration value="left"/>
        <xsd:enumeration value="center"/>
        <xsd:enumeration value="right"/>
        <xsd:enumeration value="bottomLeft"/>
        <xsd:enumeration value="bottom"/>
        <xsd:enumeration value="bottomRight"/>
    </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.2 **ST_alignLabel**

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_BackstageEditBox, CT_BackstageDropDown, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl

Specifies the position where the relevant control’s label is aligned.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>topLeft</td>
<td>Specifies that the label is aligned at the top left of the control.</td>
</tr>
<tr>
<td>top</td>
<td>Specifies that the label is aligned at the top of the control.</td>
</tr>
<tr>
<td>topRight</td>
<td>Specifies that the label is aligned at the top right of the control.</td>
</tr>
<tr>
<td>left</td>
<td>Specifies that the label is aligned at the left side of the control.</td>
</tr>
<tr>
<td>center</td>
<td>Specifies that the label is aligned at the center of the control.</td>
</tr>
<tr>
<td>right</td>
<td>Specifies that the label is aligned at the right side of the control.</td>
</tr>
<tr>
<td>bottomLeft</td>
<td>Specifies that the label is aligned at the bottom left of the control.</td>
</tr>
<tr>
<td>bottom</td>
<td>Specifies that the label is aligned at the bottom of the control.</td>
</tr>
<tr>
<td>bottomRight</td>
<td>Specifies that the label is aligned at the bottom right of the control.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_alignLabel">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="topLeft"/>
        <xsd:enumeration value="top"/>
        <xsd:enumeration value="topRight"/>
        <xsd:enumeration value="left"/>
        <xsd:enumeration value="center"/>
        <xsd:enumeration value="right"/>
        <xsd:enumeration value="bottomLeft"/>
        <xsd:enumeration value="bottom"/>
        <xsd:enumeration value="bottomRight"/>
    </xsd:restriction>
</xsd:simpleType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.3 **ST_BoxStyle**

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_Box]

Specifies the layout style of a container control.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontal</td>
<td>Specifies that the child controls are laid out horizontally.</td>
</tr>
<tr>
<td>vertical</td>
<td>Specifies that the child controls are laid out vertically.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_BoxStyle">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="horizontal"/>
    <xsd:enumeration value="vertical"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.4 **ST_expand1**

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** [CT_LayoutContainer]

Specifies the direction or directions that the relevant control expands within its container.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontal</td>
<td>Specifies that the control expands horizontally.</td>
</tr>
<tr>
<td>vertical</td>
<td>Specifies that the control expands vertically.</td>
</tr>
<tr>
<td>both</td>
<td>Specifies that the control expands both horizontally and vertically.</td>
</tr>
<tr>
<td>neither</td>
<td>Specifies that the control does not expand.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.5 ST_firstColumnMaxWidth


Referenced by: CT_BackstageTab

Specifies the maximum width of the first column within a Backstage tab, in pixels.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```
<xsd:simpleType name="ST_firstColumnMaxWidth">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="10000"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.6 ST_firstColumnMinWidth


Referenced by: CT_BackstageTab

Specifies the minimum width of the first column within a Backstage tab, in pixels.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```
<xsd:simpleType name="ST_firstColumnMinWidth">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="10000"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.7 ST_GalleryShowInRibbon


Referenced by: CT_GalleryRegular, CT_Gallery
This type has no meaning and MUST be ignored.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>This value has no meaning and MUST be ignored.</td>
</tr>
<tr>
<td>0</td>
<td>This value has no meaning and MUST be ignored.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_GalleryShowInRibbon">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="false"/>
    <xsd:enumeration value="0"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.8 ST_secondColumnMaxWidth

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui  
**Referenced by:** CT_BackstageTab

Specifies the maximum width of the second column within a Backstage tab, in pixels.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_secondColumnMaxWidth">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>  
    <xsd:maxInclusive value="10000"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.9 ST_secondColumnMinWidth

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui  
**Referenced by:** CT_BackstageTab

Specifies the minimum width of the second column within a Backstage tab, in pixels.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_secondColumnMinWidth">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>  
    <xsd:maxInclusive value="10000"/>
  </xsd:restriction>
</xsd:simpleType>
```
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.10 ST_style1


*Referenced by:* CT_BackstageGroupButton

Specifies the visual style for a Backstage button control.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>The button appears normally.</td>
</tr>
<tr>
<td>borderless</td>
<td>The button does not display a border around its contents.</td>
</tr>
<tr>
<td>large</td>
<td>The button has a large size.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_style1">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="normal"/>
    <xsd:enumeration value="borderless"/>
    <xsd:enumeration value="large"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.11 ST_columnWidthPercent


*Referenced by:* CT_BackstageTab

Specifies the width of a column within a Backstage tab as a percentage of the width of its parent container.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_columnWidthPercent">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="99"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.4.12 ST_Delegate


Referenced by: CT_Command, CT_ControlBase, CT_Control, CT_ControlCloneRegular, CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_LabelControl, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_EditBox, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_MenuSeparator, CT_SplitButtonRegular, CT_DynamicMenuRegular, CT_SplitButtonRestricted, CT_SplitButtonBase, CT_MenuWithTitle, CT_SplitButtonWithTitle, CT_Menu, CT_DynamicMenu, CT_SplitButton, CT_Box, CT_ButtonGroup, CT_Separator, CT_Group, CT_Tab, CT_TabSet, CT_MenuRoot, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageCheckBoxBase, CT_BackstageCheckButton, CT_BackstageMenuCheckBox, CT_BackstageMenuToqleButton, CT_BackstageEditBox, CT_BackstageDropDown, CT_BackstageItem, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl, CT_BackstagePrimaryMenu, CT_BackstageMenuGroup, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_ImageControl, CT_GroupBox, CT_BackstageGroup, CT_TaskGroup, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupCategory, CT_TaskFormGroupTask, CT_BackstageTab, CT_Backstage, CT_CustomUI

Specifies the name of a callback function. The format of this string is application defined.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```
<xsd:simpleType name="ST_Delegate">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="1024"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.13 ST_expand


Referenced by: CT_BackstageGroupButton, CT_BackstageCheckButton, CT_BackstageEditBox, CT_BackstageDropDown, CT_BackstageItem, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl, CT_BackstagePrimaryMenu, CT_BackstageMenuGroup, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_ImageControl, CT_GroupBox, CT_BackstageGroup, CT_TaskGroup, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupCategory, CT_TaskFormGroupTask, CT_BackstageTab, CT_Backstage, CT_CustomUI

Specifies the direction or directions that the relevant control expands within its container.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontal</td>
<td>Specifies that the control expands horizontally.</td>
</tr>
<tr>
<td>vertical</td>
<td>Specifies that the control expands vertically.</td>
</tr>
<tr>
<td>both</td>
<td>Specifies that the control expands both horizontally and vertically.</td>
</tr>
<tr>
<td>neither</td>
<td>Specifies that the control does not expand.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.
<xsd:simpleType name="ST_expand">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="horizontal"/>
    <xsd:enumeration value="vertical"/>
    <xsd:enumeration value="both"/>
    <xsd:enumeration value="neither"/>
  </xsd:restriction>
</xsd:simpleType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.14 ST_GalleryItemWidthHeight


Referenced by: CT_GalleryRegular, CT_Gallery

Specifies the width or height of a gallery item, in pixels.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_GalleryItemWidthHeight">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="4096"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.15 ST_GalleryRowColumnCount


Referenced by: CT_GalleryRegular, CT_Gallery

Specifies the count of rows or columns in a gallery control.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_GalleryRowColumnCount">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="1024"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.16 ST_style


Referenced by: CT_BackstageGroup
Specifies the visual style for a Backstage group control.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>The control is displayed with the normal visual appearance.</td>
</tr>
<tr>
<td>warning</td>
<td>The control’s visual style indicates a warning.</td>
</tr>
<tr>
<td>error</td>
<td>The control’s visual style indicates an error.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_style">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="normal"/>
        <xsd:enumeration value="warning"/>
        <xsd:enumeration value="error"/>
    </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.17 ST_ID

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:**
- CT_Command
- CT_ControlBase
- CT_Control
- CT_ControlCloneRegular
- CT_ControlClone
- CT_Button
- CT_ButtonRegular
- CT_ControlCloneQat
- CT_LabelControl
- CT_VisibleButton
- CT_ToggleButtonRegular
- CT_ToggleButton
- CT_VisibleToggleButton
- CT_CheckBox
- CT_EditBox
- CT_Item
- CT_ComboBox
- CT_DropDownRegular
- CT_DropDown
- CT_GalleryRegular
- CT_Gallery
- CT_MenuRegular
- CT_MenuSeparator
- CT_SplitButtonRegular
- CT_DynamicMenuRegular
- CT_SplitButtonRestricted
- CT_SplitButtonBase
- CT_MenuWithTitle
- CT_SplitButtonWithTitle
- CT_Menu
- CT_DynamicMenu
- CT_SplitButton
- CT_Box
- CT_ButtonGroup
- CT_Separator
- CT_MenuSeparatorNoTitle
- CT_Group
- CT_Tab
- CT_TabSet
- CT_ContextMenu
- CT_BackstageButtonBase
- CT_BackstageRegularButton
- CT_BackstageGroupButton
- CT_BackstageMenuButton
- CT_BackstageFastCommandButton
- CT_BackstageMenuToggleButton
- CT_Hyperlink
- CT_BackstagePrimaryMenu
- CT_BackstageSubMenu
- CT_BackstageMenuBase
- CT_ImageControl
- CT_Backstage
- CT_HeaderGroup
- CT_TaskGroup
- CT_TaskGroupCategory
- CT_TaskGroupTask
- CT_TaskFormGroup
- CT_TaskFormGroupCategory
- CT_TaskFormGroupTask
- CT_BackstageTab

Specifies the identifier of a built-in control. The acceptable values for control identifiers are defined on a per-application basis, as specified in the following Custom UI Command Tables.

- Word Custom UI Command Table [MSDN-CUICT/Word]
- Excel Custom UI Command Table [MSDN-CUICT/Excel]
- PowerPoint Custom UI Command Table [MSDN-CUICT/PPT]
- Custom UI Image ID Table [MSDN-CUIDT/Image]

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_ID">
</xsd:simpleType>
```
```
<xsd:restriction base="xsd:NCName">
  <xsd:minLength value="1"/>
  <xsd:maxLength value="1024"/>
</xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.18 ST_ItemSize

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_MenuRegular, CT_MenuWithTitle, CT_Menu, CT_MenuRoot, CT_BackstageMenuGroup

Specifies the size of child controls within a menu control.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>Specifies that the child controls are of normal size.</td>
</tr>
<tr>
<td>large</td>
<td>Specifies that the child controls are of large size.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```
<xsd:simpleType name="ST_ItemSize">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="normal"/>
    <xsd:enumeration value="large"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.19 ST_Keytip

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_ControlBase, CT_Control, CT_ControlCloneRegular, CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_EditBox, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_SplitButtonRegular, CT_DynamicMenuRegular, CT_SplitButtonRestricted, CT_SplitButtonBase, CT_MenuWithTitle, CT_SplitButtonWithTitle, CT_Menu, CT_DynamicMenu, CT_SplitButton, CT_Group, CT_Tab, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageCheckBoxBase, CT_BackstageCheckBox, CT_BackstageMenuCheckBox, CT_BackstageMenuToggleButton, CT_BackstageEditBox, CT_BackstageDropdown, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstagePrimaryMenu, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_TaskGroupTask, CT_TaskFormGroupTask, CT_BackstageTab

Specifies a KeyTip string.
The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_Keytip">
  <xsd:restriction base="xsd:token">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="3"/>
    <xsd:whiteSpace value="collapse"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.20 ST_layoutChildren

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui  
**Referenced by:** CT_LayoutContainer

Specifies the direction that the child controls of a container control are laid out.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>horizontal</td>
<td>Specifies that the child controls are laid out horizontally.</td>
</tr>
<tr>
<td>vertical</td>
<td>Specifies that the child controls are laid out vertically.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_layoutChildren">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="horizontal"/>
    <xsd:enumeration value="vertical"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.21 ST_LongString

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui  
**Referenced by:** CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_DynamicMenuRegular, CT_Menu, CT_DynamicMenu, CT_BackstageMenuButton, CT_BackstageCheckBox, CT_BackstageMenuCheckButton, CT_BackstageMenuToggleButton, CT_BackstageSubMenu, CT_ImageControl, CT_BackstageGroup, CT_TaskGroup, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupTask

Specifies a string of potentially long length.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.
See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.22 ST_QID


Referenced by: CT_ControlBase, CT_Control, CT_ControlCloneRegular, CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_LabelControl, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_EditBox, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_MenuSeparator, CT_SplitButtonRegular, CT_DynamicMenuRegular, CT_SplitButtonRestricted, CT_SplitButtonBase, CT_MenuWithTitle, CT_SplitButtonWithTitle, CT_Menu, CT_DynamicMenu, CT_SplitButton, CT_Box, CT_ButtonGroup, CT_Separator, CT_MenuSeparatorNoTitle, CT_Group, CT_Tab, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageCheckBox, CT_BackstageMenuCheckBox, CT_BackstageMenuButton, CT_BackstageEditBox, CT_BackstageDropDown, CT_RadioButton, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl, CT_BackstagePrimaryMenu, CT_BackstageMenuGroup, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_IconControl, CT_GroupBox, CT_LayoutContainer, CT_BackstageGroup, CT_HeaderGroup, CT_TaskGroup, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupCategory, CT_TaskFormGroupTask, CT_BackstageTab

Specifies a control identifier that is qualified by an XML namespace prefix. The prefix determines which namespace the control belongs to.

If the prefix is set to a namespace other than the namespace of the containing document, the qualified identifier implicitly defines a new custom namespace. If multiple Custom UI documents refer to controls in the same namespace, the documents are able to share common containers.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```
<xsd:simpleType name="ST_QID">
  <xsd:restriction base="xsd:QName">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="1024"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.23 ST_Size


Referenced by: CT_ControlClone, CT_Button, CT_ControlCloneQat, CT_ToggleButton, CT_Gallery, CT_Menu, CT_DynamicMenu, CT_SplitButton

Specifies the size of a control.
<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>normal</td>
<td>Specifies the normal control size.</td>
</tr>
<tr>
<td>large</td>
<td>Specifies the large control size.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_Size">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="normal"/>
    <xsd:enumeration value="large"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

### 2.4.24 ST_String

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:**
- CT_ControlBase, CT_Control, CT_ControlCloneRegular, CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_LabelControl, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_EditBox, CT_Item, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_MenuSeparator, CT_SplitButtonRegular, CT_DynamicMenuRegular, CT_SplitButtonRestricted, CT_SplitButtonBase, CT_MenuWithTitle, CT_SplitButtonWithTitle, CT_Menu, CT_DynamicMenu, CT_SplitButton, CT_Box, CT_ButtonGroup, CTSeparator, CT_MenuSeparatorNoTitle, CT_Group, CT_Tab, CT_MenuRoot, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageCheckBoxBase, CT_BackstageCheckBox, CT_BackstageMenuCheckBox, CT_BackstageMenuToggleButton, CT_BackstageEditBox, CT_BackstageDropDown, CT_BackstageItem, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl, CT_BackstagePrimaryMenu, CT_BackstageMenuGroup, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_ImageControl, CT_GroupBox, CT_LayoutContainer, CT_BackstageGroup, CT_HeaderGroup, CT_TaskGroup, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupCategory, CT_TaskFormGroupTask, CT_BackstageTab

Specifies a string with limited length.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_String">
  <xsd:restriction base="xsd:string">
    <xsd:maxLength value="1024"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.4.25 ST_StringLength

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_EditBox, CT_ComboBox, CT_BackstageEditBox

Specifies the length of a string, in characters.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_StringLength">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="1024"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.26 ST_TaskSizes

**Target namespace:** http://schemas.microsoft.com/office/2009/07/customui

**Referenced by:** CT_TaskGroup, CT_TaskFormGroup

Specifies the allowed sizes of the relevant control’s tasks.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>largeMediumSmall</td>
<td>Specifies that the tasks are of large, medium, or small size.</td>
</tr>
<tr>
<td>largeMedium</td>
<td>Specifies that the tasks are of large or medium size.</td>
</tr>
<tr>
<td>large</td>
<td>Specifies that the tasks are of large size.</td>
</tr>
<tr>
<td>mediumSmall</td>
<td>Specifies that the tasks are of medium or small size.</td>
</tr>
<tr>
<td>medium</td>
<td>Specifies that the tasks are of medium size.</td>
</tr>
<tr>
<td>small</td>
<td>Specifies that the tasks are of small size.</td>
</tr>
</tbody>
</table>

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_TaskSizes">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="largeMediumSmall"/>
    <xsd:enumeration value="largeMedium"/>
    <xsd:enumeration value="large"/>
    <xsd:enumeration value="mediumSmall"/>
    <xsd:enumeration value="medium"/>
    <xsd:enumeration value="small"/>
  </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
2.4.27 ST_UniqueID


Referenced by: CT_Control, CT_Button, CT_ButtonRegular, CT_LabelControl, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_VisibleToggleButton, CT_CheckBox, CT_EditBox, CT_Item, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_MenuSeparator, CT_SplitButtonRegular, CT_DynamicMenuRegular, CT_SplitButton Restricted, CT_SplitButtonBase, CT_MenuWithTitle, CT_SplitButtonClickTitle, CT_Menu, CT_DynamicMenu, CT_SplitButton, CT_Box, CT_ButtonGroup, CT_Separator, CT_MenuSeparatorNoTitle, CT_Group, CT_Tab, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageCheckBoxBase, CT_BackstageCheckButton, CT_BackstageMenuCheckBox, CT_BackstageMenuToggleButton, CT_BackstageEditBox, CT_BackstageDropDown, CT_BackstageItem, CT_RadioGroup, CT_BackstageComboBox, CT_Hyperlink, CT_BackstageLabelControl, CT_BackstagePrimaryMenu, CT_BackstageMenuGroup, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_ImageControl, CT_GroupBox, CT_LayoutContainer, CT_BackstageGroup, CT_StencilGroup, CT_TaskGroup, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupGroup, CT_TaskFormGroupTask, CT_BackstageTab

Specifies a custom control identifier. All custom control identifiers MUST be unique within this document.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_UniqueID">
    <xsd:restriction base="xsd:ID">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="1024"/>
    </xsd:restriction>
</xsd:simpleType>
```

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).

2.4.28 ST_Uri


Referenced by: CT_ControlBase, CT_Control, CT_ControlCloneRegular, CT_ControlClone, CT_Button, CT_ButtonRegular, CT_ControlCloneQat, CT_VisibleButton, CT_ToggleButtonRegular, CT_ToggleButton, CT_EditBox, CT_Item, CT_ComboBox, CT_DropDownRegular, CT_GalleryRegular, CT_Gallery, CT_MenuRegular, CT_DynamicMenuRegular, CT_SplitButtonBase, CT_MenuWithTitle, CT_Menu, CT_DynamicMenu, CT_Group, CT_BackstageButtonBase, CT_BackstageRegularButton, CT_BackstageGroupButton, CT_BackstageMenuButton, CT_BackstageFastCommandButton, CT_BackstageMenuToggleButton, CT_Hyperlink, CT_BackstagePrimaryMenu, CT_BackstageSubMenu, CT_BackstageMenuBase, CT_ImageControl, CT_TaskGroupBase, CT_TaskGroupCategory, CT_TaskGroupTask, CT_TaskFormGroup, CT_TaskFormGroupGroup, CT_TaskFormGroupTask, CT_BackstageTab

Specifies the identifier of an XML part that is related to the containing Custom UI document. The target part MUST contain an image file.

The following W3C XML Schema ([XMLSCHEMA1/2] section 2.1) fragment specifies the contents of this simple type.

```xml
<xsd:simpleType name="ST_Uri">
    <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
    </xsd:restriction>
</xsd:simpleType>
```
<xsd:maxLength value="1024"/>
</xsd:restriction>
</xsd:simpleType>

See section 5.1 for the full W3C XML Schema ([XMLSCHEMA1/2] section 2.1).
3 Structure Examples

The following sections provide examples of some of the structures specified in the preceding sections.

3.1 Ribbon

The following shows an example of a ribbon customization.

```xml
  <ribbon>
    <tabs>
      <tab idMso="TabHome" visible="false" />
      <tab id="customTab"
        insertAfterMso="TabView"
        label="Tab">
        <group id="customGroup"
          label="Group">
          <button id="customButton"
            label="Button"
            imageMso="HappyFace"
            size="large"
            onAction="OnButtonClicked" />
          <editBox id="customEditBox"
            label="Edit Box"
            onChange="OnEditBoxTextChanged" />
        </group>
      </tab>
    </tabs>
  </ribbon>
</customUI>
```

This example hides the built-in tab with identifier TabHome and creates a custom tab. The custom tab contains one group, which in turn contains a button and an edit box. The button is labeled Button, displays the built-in HappyFace icon, and calls the OnButtonClicked function when it is invoked. The edit box is labeled Edit Box and calls the OnEditBoxTextChanged function when text is entered.

3.2 Context Menu

The following shows an example of a context menu customization.

```xml
  <contextMenus>
    <contextMenu idMso="ContextMenuText">
      <control idMso="Cut"
        visible="false" />
      <control idMso="Copy"
        visible="false" />
      <button id="customButton"
        label="Click Me"
        getImage="LoadCustomImage"
        onAction="OnButtonClicked" />
      <ToggleButton id="customToggleButton"
        label="Toggle Me"
        getImage="LoadCustomImage"
        onAction="OnToggleButtonToggled" />
    </contextMenu>
  </contextMenus>
</customUI>
```
This example customizes the built-in context menu with identifier `ContextMenuText`. It hides the two built-in controls with identifiers "Cut" and "Copy". It also adds a button and a toggle button with labels Click Me and Toggle Me, respectively. The icons for these controls are loaded by using the `LoadCustomImage` callback function. The `OnButtonClicked` callback function is called when the button control is invoked, and the `OnButtonToggled` callback function is called when the toggle button is switched.

### 3.3 Backstage

The following shows an example of a Backstage customization.

```xml
  <backstage>
    <tab id="customTab"
      label="Custom Tab">
      <firstColumn>
        <group id="customGroup"
          label="Custom Group">
          <primaryItem>
            <menu id="customMenu"
              imageMso="HappyFace"
              label="Actions">
              <menuGroup id="menuGroup1">
                <button id="button1"
                  imageMso="Bold"
                  label="Action 1"
                  onAction="OnButtonClicked" />
                <button id="button2"
                  imageMso="Italic"
                  label="Action 2"
                  onAction="OnButtonClicked" />
              </menuGroup>
            </menu>
          </primaryItem>
          <topItems>
            <radioGroup id="radioGroup1"
              label="Choose Color:">
              <radioButton id="radioButton1"
                label="Red" />
              <radioButton id="radioButton2"
                label="Green" />
              <radioButton id="radioButton3"
                label="Blue" />
            </radioGroup>
          </topItems>
        </group>
      </firstColumn>
    </tab>
  </backstage>
</customUI>
```

This example creates a custom tab within the Backstage labeled Custom Tab. The tab contains one group, which in turn contains a menu and a radio group. The menu contains two button controls labeled Action 1 and Action 2. The `OnButtonClicked` callback function is called when the buttons are invoked. The radio group displays three radio button controls labeled Red, Green, and Blue. The `OnRadioButtonSelected` callback function is called when a radio button is invoked.
4 Security

4.1 Security Considerations for Implementers
None.

4.2 Index of Security Fields
None.
5 Appendix A: Full XML Schemas


```xml
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" version="1.0"
    attributeFormDefault="unqualified"
    elementFormDefault="qualified">
    <xsd:simpleType name="ST_QID">
        <xsd:restriction base="xsd:QName">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_ID">
        <xsd:restriction base="xsd:NCName">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_UniqueID">
        <xsd:restriction base="xsd:ID">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_Delegate">
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_StringLength">
        <xsd:restriction base="xsd:positiveInteger">
            <xsd:minInclusive value="1"/>
            <xsd:maxInclusive value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_GalleryRowColumnCount">
        <xsd:restriction base="xsd:positiveInteger">
            <xsd:minInclusive value="1"/>
            <xsd:maxInclusive value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_GalleryShowInRibbon">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="false"/>
            <xsd:enumeration value="0"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_String">
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="1024"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="ST_LongString">
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="4096"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```
<xsd:simpleType name="ST_Url">
    <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="1024"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_Size">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="normal"/>
        <xsd:enumeration value="large"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_ItemSize">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="normal"/>
        <xsd:enumeration value="large"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_BoxStyle">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="horizontal"/>
        <xsd:enumeration value="vertical"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_Keytip">
    <xsd:restriction base="xsd:token">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="3"/>
        <xsd:whiteSpace value="collapse"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_TaskSizes">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="largeMediumSmall"/>
        <xsd:enumeration value="largeMedium"/>
        <xsd:enumeration value="large"/>
        <xsd:enumeration value="mediumSmall"/>
        <xsd:enumeration value="medium"/>
        <xsd:enumeration value="small"/>
    </xsd:restriction>
</xsd:simpleType>

<xsd:attributeGroup name="AG_IDCustom">
    <xsd:attribute name="id" type="ST_UniqueID" use="optional"/>
    <xsd:attribute name="idQ" type="ST_QID" use="optional"/>
    <xsd:attribute name="tag" type="ST_String" use="optional"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_IDMso">
    <xsd:attribute name="idMso" type="ST_ID" use="optional"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_Title">
    <xsd:attribute name="title" type="ST_String" use="optional"/>
    <xsd:attribute name="getTitle" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_IDAttributes">
    <xsd:attributeGroup ref="AG_IDCustom"/>
    <xsd:attributeGroup ref="AG_IDMso"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_Image">
    <xsd:attribute name="image" type="ST_Url" use="optional"/>
    <xsd:attribute name="imageMso" type="ST_ID" use="optional"/>
    <xsd:attribute name="getImage" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_CommonAttributes">
    <xsd:attributeGroup ref="AG_Enabled"/>
</xsd:attributeGroup>

<xsd:attributeGroup name="AG_PositionAttributes">
    <xsd:attribute name="insertAfterMso" type="ST_ID" use="optional"/>
    <xsd:attribute name="insertBeforeMso" type="ST_ID" use="optional"/>
</xsd:attributeGroup>
<xsd:attribute name="insertAfterQ" type="ST_QID" use="optional"/>
<xsd:attributeGroup>
  <xsd:attribute name="insertBeforeQ" type="ST_QID" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Enabled">
  <xsd:attribute name="enabled" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="getEnabled" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Visible">
  <xsd:attribute name="visible" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="getVisible" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Label">
  <xsd:attribute name="label" type="ST_String" use="optional"/>
  <xsd:attribute name="getLabel" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Keytip">
  <xsd:attribute name="keytip" type="ST_Keytip" use="optional"/>
  <xsd:attribute name="getKeytip" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Screentip">
  <xsd:attribute name="screentip" type="ST_String" use="optional"/>
  <xsd:attribute name="getScreentip" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="supertip" type="ST_String" use="optional"/>
  <xsd:attribute name="getSupertip" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Description">
  <xsd:attribute name="description" type="ST_LongString" use="optional"/>
  <xsd:attribute name="getDescription" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_AltText">
  <xsd:attribute name="altText" type="ST_LongString" use="optional"/>
  <xsd:attribute name="getAltText" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_ShowLabel">
  <xsd:attribute name="showLabel" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="getShowLabel" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_HelperText">
  <xsd:attribute name="helperText" type="ST_LongString" use="optional"/>
  <xsd:attribute name="getHelperText" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_UIAttributes">
  <xsd:attributeGroup ref="AG_CommonAttributes"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_ItemAttributes">
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_Screentip"/>
  <xsd:attributeGroup ref="AG_UIAttributes"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_ControlAttributes">
  <xsd:attributeGroup ref="AG_ItemAttributes"/>
  <xsd:attributeGroup ref="AG_ShowLabel"/>
  <xsd:attributeGroup ref="AG_HelperText"/>
  <xsd:attributeGroup ref="AG_UIAttributes"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Action">
  <xsd:attribute name="onAction" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Pressed">
  <xsd:attribute name="getPressed" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_Definitive">
  <xsd:attribute name="isDefinitive" type="xsd:boolean" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_SizeAttributes"/>
<xsd:attribute name="size" type="ST_Size" use="optional"/>
<xsd:attribute name="getSize" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_DropDownAttributes">
<xsd:attribute name="showItemImage" type="xsd:boolean" use="optional"/>
<xsd:attribute name="getItemCount" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemLabel" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemScreentip" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemSupertip" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemImage" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_GetContentAttributes">
<xsd:attribute name="getContent" type="ST_Delegate" use="required"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG_DynamicContentAttributes">
<xsd:attribute name="invalidateContentOnDrop" type="xsd:boolean" use="optional"/>
</xsd:attributeGroup>
</xsd:simpleType>
<xsd:simpleType name="ST_alignLabel">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="topLeft"/>
<xsd:enumeration value="top"/>
<xsd:enumeration value="topRight"/>
<xsd:enumeration value="left"/>
<xsd:enumeration value="center"/>
<xsd:enumeration value="right"/>
<xsd:enumeration value="bottomLeft"/>
<xsd:enumeration value="bottom"/>
<xsd:enumeration value="bottomRight"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:attributeGroup name="AG_AlignAttributes">
<xsd:attribute name="alignLabel" use="optional" type="ST_alignLabel"/>
</xsd:attributeGroup>
<xsd:simpleType name="ST_expand">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="horizontal"/>
<xsd:enumeration value="vertical"/>
<xsd:enumeration value="both"/>
<xsd:enumeration value="neither"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:attributeGroup name="AG_Expand">
<xsd:attribute name="expand" use="optional" type="ST_expand"/>
</xsd:attributeGroup>
<xsd:simpleType name="ST_style">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="normal"/>
<xsd:enumeration value="warning"/>
<xsd:enumeration value="error"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:attributeGroup name="AG_GroupStyle">
<xsd:attribute name="style" use="optional" type="ST_style"/>
<xsd:attribute name="getStyle" type="ST_Delegate" use="optional"/>
</xsd:attributeGroup>
<xsd:simpleType name="ST_style1">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="normal"/>
<xsd:enumeration value="borderless"/>
<xsd:enumeration value="large"/>  
</xsd:restriction>
</xsd:simpleType>
<xsd:attributeGroup ref="AG_Enabled"/>
<xsd:attributeGroup ref="AG_IDMso"/>
</xsd:complexType>
<xsd:complexType name="CT_ControlBase">
  <xsd:attributeGroup ref="AG_ControlAttributes"/>
</xsd:complexType>
<xsd:complexType name="CT_Control">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:attributeGroup ref="AG_IDAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_ControlCloneRegular">
  <xsd:complexContent>
    <xsd:restriction base="CT_Control">
      <xsd:attribute name="id" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_ControlClone">
  <xsd:complexContent>
    <xsd:restriction base="CT_Button">
      <xsd:attribute name="id" use="prohibited"/>
      <xsd:attribute name="onAction" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_ControlCloneQat">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:attribute name="id" type="ST_ID" use="optional"/>
      <xsd:attribute name="idQ" type="ST_QID" use="optional"/>
      <xsd:attributeGroup ref="AG_IDMso"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_LabelControl">
  <xsd:complexContent>
    <xsd:restriction base="CT_Control">
      <xsd:attribute name="image" use="prohibited"/>
      <xsd:attribute name="imageMso" use="prohibited"/>
      <xsd:attribute name="getImage" use="prohibited"/>
      <xsd:attribute name="keytip" use="prohibited"/>
      <xsd:attribute name="getKeytip" use="prohibited"/>
      <xsd:attribute name="showImage" use="prohibited"/>
      <xsd:attribute name="getShowImage" use="prohibited"/>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_ButtonRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_Control">
      <xsd:attributeGroup ref="AG_Action"/>
      <xsd:attributeGroup ref="AG_Enabled"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_Image"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_Button">
  <xsd:complexContent>
    <xsd:extension base="CT_ButtonRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_VisibleButton">
    <xsd:complexContent>
        <xsd:restriction base="CT_ButtonRegular">
            <xsd:attribute name="visible" use="prohibited"/>
            <xsd:attribute name="getVisible" use="prohibited"/>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_ToggleButtonRegular">
    <xsd:complexContent>
        <xsd:extension base="CT_ButtonRegular">
            <xsd:attributeGroup ref="AG_Pressed"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_ToggleButton">
    <xsd:complexContent>
        <xsd:extension base="CT_ToggleButtonRegular">
            <xsd:attributeGroup ref="AG_SizeAttributes"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_VisibleToggleButton">
    <xsd:complexContent>
        <xsd:restriction base="CT_ToggleButtonRegular">
            <xsd:attribute name="visible" use="prohibited"/>
            <xsd:attribute name="getVisible" use="prohibited"/>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_CheckBox">
    <xsd:complexContent>
        <xsd:restriction base="CT_ToggleButtonRegular">
            <xsd:attribute name="image" use="prohibited"/>
            <xsd:attribute name="imageMso" use="prohibited"/>
            <xsd:attribute name="getImage" use="prohibited"/>
            <xsd:attribute name="showImage" use="prohibited"/>
            <xsd:attribute name="showLabel" use="prohibited"/>
            <xsd:attribute name="getShowLabel" use="prohibited"/>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_EditBox">
    <xsd:complexContent>
        <xsd:extension base="CT_Control">
            <xsd:attributeGroup ref="AG_Enabled"/>
            <xsd:attributeGroup ref="AG_Image"/>
            <xsd:attribute name="maxLength" type="ST_StringLength" use="optional"/>
            <xsd:attribute name="getText" type="ST_Delegate" use="optional"/>
            <xsd:attribute name="onChange" type="ST_Delegate" use="optional"/>
            <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_Item">
    <xsd:attribute name="id" type="ST_UniqueID" use="optional"/>
    <xsd:attribute name="label" type="ST_String" use="optional"/>
    <xsd:attribute name="image" type="ST_Uri" use="optional"/>
    <xsd:attribute name="imageMso" type="ST_ID" use="optional"/>
    <xsd:attribute name="screentip" type="ST_String" use="optional"/>
    <xsd:attribute name="supertip" type="ST_String" use="optional"/>
</xsd:complexType>

<xsd:complexType name="CT_ComboBox">
    <xsd:complexContent>
        <xsd:extension base="CT_EditBox">
            <xsd:sequence>
                <xsd:element name="item" type="CT_Item" minOccurs="0" maxOccurs="1000"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:attributeGroup ref="AG_DropDownAttributes"/>
<xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_DropDownRegular">
<xsd:complexContent>
<xsd:extension base="CT_Control">
<xsd:sequence>
<xsd:element name="item" type="CT_Item" minOccurs="0" maxOccurs="1000"/>
<xsd:element name="button" type="CT_ButtonRegular" minOccurs="0" maxOccurs="16"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_Action"/>
<xsd:attributeGroup ref="AG_Enabled"/>
<xsd:attributeGroup ref="AG_Image"/>
<xsd:attributeGroup ref="AG_DropDownAttributes"/>
<xsd:attribute name="getSelectedItemID" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getSelectedItemIndex" type="ST_Delegate" use="optional"/>
<xsd:attribute name="showItemLabel" type="xsd:boolean" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_GalleryRegular">
<xsd:complexContent>
<xsd:extension base="CT_DropDownRegular">
<xsd:attributeGroup ref="AG_Description"/>
<xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
<xsd:attribute name="columns" type="ST_GalleryRowColumnCount" use="optional"/>
<xsd:attribute name="rows" type="ST_GalleryRowColumnCount" use="optional"/>
<xsd:attribute name="itemWidth" type="ST_GalleryItemWidthHeight" use="optional"/>
<xsd:attribute name="itemHeight" type="ST_GalleryItemWidthHeight" use="optional"/>
<xsd:attribute name="getItemWidth" type="ST_Delegate" use="optional"/>
<xsd:attribute name="getItemHeight" type="ST_Delegate" use="optional"/>
<xsd:attribute name="showItemLabel" type="xsd:boolean" use="optional"/>
<xsd:attribute name="showInRibbon" type="ST_GalleryShowInRibbon" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_Gallery">
<xsd:complexContent>
<xsd:extension base="CT_GalleryRegular">
<xsd:attributeGroup ref="AG_SizeAttributes"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:group name="EG_MenuControlsBase">
<xsd:choice>
<xsd:element name="control" type="CT_ControlCloneRegular"/>
<xsd:element name="button" type="CT_ButtonRegular"/>
<xsd:element name="checkBox" type="CT_CheckBox"/>
<xsd:element name="gallery" type="CT_GalleryRegular"/>
<xsd:element name="toggleButton" type="CT_ToggleButtonRegular"/>
<xsd:element name="menuSeparator" type="CT_MenuSeparator"/>
</xsd:choice>
</xsd:group>
<xsd:group name="EG_MenuOrSplitButtonRegular">
<xsd:choice>
<xsd:element name="splitButton" type="CT_SplitButtonRegular"/>
<xsd:element name="menu" type="CT_MenuRegular"/>
<xsd:element name="dynamicMenu" type="CT_DynamicMenuRegular"/>
</xsd:choice>
</xsd:group>
<xsd:group name="EG_MenuOrSplitButtonWithTitle">
<xsd:choice>
<xsd:element name="splitButton" type="CT_SplitButtonWithTitle"/>
<xsd:element name="menu" type="CT_MenuWithTitle"/>
<xsd:element name="dynamicMenu" type="CT_DynamicMenuRegular"/>
</xsd:choice>
</xsd:group>
<xsd:group name="EG_ContextMenuControls">
  <xsd:choice>
    <xsd:element name="control" type="CT_ControlCloneRegular"/>
    <xsd:element name="button" type="CT_ButtonRegular"/>
    <xsd:element name="checkBox" type="CT_CheckBox"/>
    <xsd:element name="gallery" type="CT_GalleryRegular"/>
    <xsd:element name="toggleButton" type="CT_ToggleButtonRegular"/>
    <xsd:element name="splitButton" type="CT_SplitButtonRegular"/>
    <xsd:element name="menu" type="CT_MenuRegular"/>
    <xsd:element name="dynamicMenu" type="CT_DynamicMenuRegular"/>
    <xsd:element name="menuSeparator" type="CT_MenuSeparatorNoTitle"/>
  </xsd:choice>
</xsd:group>

<xsd:complexType name="CT_MenuRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="1000">
          <xsd:group ref="EG_MenuControlsBase"/>
          <xsd:group ref="EG_MenuOrSplitButtonRegular"/>
        </xsd:choice>
      </xsd:sequence>
      <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_IDAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_DynamicMenuRegular">
  <xsd:complexContent>
    <xsd:extension base="CT_ControlBase">
      <xsd:attributeGroup ref="AG_Description"/>
      <xsd:attributeGroup ref="AG_IDAttributes"/>
      <xsd:attributeGroup ref="AG_GetContentAttributes"/>
      <xsd:attributeGroup ref="AG_DynamicContentAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_MenuWithTitle">
  <xsd:complexContent>
    <xsd:extension base="CT_MenuRegular">
      <xsd:attributeGroup ref="AG_IDAttributes"/>
      <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
      <xsd:attributeGroup ref="AG_Title"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_Menu">
  <xsd:complexContent>
    <xsd:extension base="CT_MenuRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
      <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_DynamicMenu">
  <xsd:complexContent>
    <xsd:extension base="CT_DynamicMenuRegular">
      <xsd:attributeGroup ref="AG_SizeAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_SplitButtonBase"/>
<xsd:complexType name="CT_SplitButton">
    <xsd:complexContent>
        <xsd:extension base="CT_SplitButtonRegular">
            <xsd:attributeGroup ref="AG_SizeAttributes"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<xsd:group name="EG_Controls">
    <xsd:choice>
        <xsd:element name="control" type="CT_ControlClone"/>
        <xsd:element name="labelControl" type="CT_LabelControl"/>
        <xsd:element name="button" type="CT_Button"/>
        <xsd:element name="toggleButton" type="CT_ToggleButton"/>
        <xsd:element name="checkBox" type="CT_CheckBox"/>
        <xsd:element name="editBox" type="CT_EditBox"/>
        <xsd:element name="comboBox" type="CT_ComboBox"/>
        <xsd:element name="dropDown" type="CT_DropDownRegular"/>
        <xsd:element name="gallery" type="CT_Gallery"/>
        <xsd:element name="menu" type="CT_Menu"/>
        <xsd:element name="dynamicMenu" type="CT_DynamicMenu"/>
        <xsd:element name="menu" type="CT_MenuWithTitle"/>
        <xsd:element name="dynamicMenu" type="CT_DynamicMenuWithTitle"/>
        <xsd:element name="dynamicMenu" type="CT_DynamicMenuWithDropDown"/>
    </xsd:choice>
</xsd:group>
<xsd:element name="splitButton" type="CT_SplitButton"/>
<xsd:element name="box" type="CT_Box"/>
<xsd:element name="buttonGroup" type="CT_ButtonGroup"/>
</xsd:choice>
</xsd:group>
<xsd:complexType name="CT_DialogLauncher">
<xsd:sequence>
<xsd:element name="button" type="CT_ButtonRegular" minOccurs="1" maxOccurs="1"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CT_Box">
<xsd:group ref="EG_Controls" minOccurs="0" maxOccurs="1000"/>
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
<xsd:attribute name="boxStyle" type="ST_BoxStyle" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_Separator">
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
<xsd:complexType name="CT_MenuSeparator">
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
<xsd:complexType name="CT_MenuSeparatorNoTitle">
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
<xsd:complexType name="CT_ButtonGroup">
<xsd:sequence>
<xsd:choice minOccurs="0" maxOccurs="1000">
<xsd:element name="control" type="CT_ControlCloneRegular"/>
<xsd:element name="button" type="CT_ButtonRegular"/>
<xsd:element name="toggleButton" type="CT_ToggleButtonRegular"/>
<xsd:element name="gallery" type="CT_GalleryRegular"/>
<xsd:element name="menu" type="CT_MenuRegular"/>
<xsd:element name="dynamicMenu" type="CT_DynamicMenuRegular"/>
<xsd:element name="splitButton" type="CT_SplitButtonRegular"/>
<xsd:element name="separator" type="CT_Separator"/>
</xsd:choice>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDCustom"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>
<xsd:complexType name="CT_Group">
<xsd:sequence>
<xsd:choice minOccurs="0" maxOccurs="1000">
<xsd:group ref="EG_Controls"/>
<xsd:element name="separator" type="CT_Separator"/>
</xsd:choice>
</xsd:sequence>
<xsd:element name="dialogBoxLauncher" type="CT_DialogLauncher" minOccurs="0" maxOccurs="1"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Image"/>
<xsd:attributeGroup ref="AG_PositionAttributes"/>
<xsd:attributeGroup ref="AG_Screentip"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_Keytip"/>
<xsd:attribute name="autoScale" type="xsd:boolean" use="optional"/>
<xsd:attribute name="centerVertically" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_Tab">
    <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="100">
            <xsd:element name="group" type="CT_Group"/>
        </xsd:choice>
    </xsd:sequence>
    <xsd:attributeGroup ref="AG_IDAttributes"/>
    <xsd:attributeGroup ref="AG_Label"/>
    <xsd:attributeGroup ref="AG_PositionAttributes"/>
    <xsd:attributeGroup ref="AG_Visible"/>
    <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>

<xsd:complexType name="CT_QatItems">
    <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="1000">
            <xsd:element name="control" type="CT_ControlCloneQat"/>
            <xsd:element name="button" type="CT_ButtonRegular"/>
            <xsd:element name="separator" type="CT_Separator"/>
        </xsd:choice>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CT_Qat">
    <xsd:sequence>
        <xsd:element name="sharedControls" type="CT_QatItems" minOccurs="0"/>
        <xsd:element name="documentControls" type="CT_QatItems" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CT_Tabs">
    <xsd:sequence>
        <xsd:element name="tab" type="CT_Tab" minOccurs="1" maxOccurs="100"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CT_TabSet">
    <xsd:sequence>
        <xsd:element name="tab" type="CT_Tab" minOccurs="0" maxOccurs="50"/>
    </xsd:sequence>
    <xsd:attribute name="idMso" type="ST_ID" use="required"/>
</xsd:complexType>

<xsd:complexType name="CT_ContextualTabs">
    <xsd:sequence>
        <xsd:element name="tabSet" type="CT_TabSet" minOccurs="1" maxOccurs="100"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CT_ContextMenu">
    <xsd:sequence>
        <xsd:choice minOccurs="0" maxOccurs="1000">
            <xsd:group ref="EG_ContextMenuControls"/>
        </xsd:choice>
    </xsd:sequence>
    <xsd:attributeGroup ref="AG_IDMso"/>
</xsd:complexType>

<xsd:complexType name="CT_Commands">
    <xsd:sequence>
        <xsd:element name="command" type="CT_Command" minOccurs="1" maxOccurs="5000"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="CT_Ribbon">
    <xsd:all>
        <xsd:element name="qat" type="CT_Qat" minOccurs="0" maxOccurs="1"/>
        <xsd:unique name="qatControls">
            <xsd:selector xpath="*/*"/>
            <xsd:field xpath="@id"/>
        </xsd:unique>
    </xsd:all>
</xsd:complexType>
<xsd:attribute name="startFromScratch" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_ContextMenus">
  <xsd:sequence>
    <xsd:element name="contextMenu" type="CT_ContextMenu" minOccurs="1" maxOccurs="1000"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CT_MenuRoot">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:group ref="EG_MenuControlsBase"/>
      <xsd:group ref="EG_MenuOrSplitButtonRegular"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_Title"/>
  <xsd:attribute name="itemSize" type="ST_ItemSize" use="optional"/>
</xsd:complexType>
<xsd:element name="menu" type="CT_MenuRoot"/>
<xsd:complexType name="CT_BackstageButtonBase">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Definitive"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attributeGroup ref="AG_Image"/>
</xsd:complexType>
<xsd:complexType name="CT_BackstageRegularButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageButtonBase">
      <xsd:attributeGroup ref="AG_Screentip"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_BackstageGroupButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageRegularButton">
      <xsd:attributeGroup ref="AG_Expand"/>
      <xsd:attributeGroup ref="AG_ButtonStyle"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_BackstageMenuButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageButtonBase">
      <xsd:attributeGroup ref="AG_Description"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_BackstageFastCommandButton">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageButtonBase">
      <xsd:attributeGroup ref="AG_IDMso"/>
      <xsd:attributeGroup ref="AG_PositionAttributes"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CT_BackstageCheckBoxBase">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Pressed"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
<xsd:complexType name="CT_BackstageCheckBox">
  <xsd:complexContent>
<xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_BackstageComboBox">
  <xsd:sequence>
    <xsd:element name="item" type="CT_BackstageItem" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attribute name="getText" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="onChange" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="sizeString" type="ST_String" use="optional"/>
  <xsd:attribute name="getItemCount" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemLabel" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="getItemID" type="ST_Delegate" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_BackstageItem">
  <xsd:attribute name="id" type="ST_UniqueID" use="optional"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>
<xsd:complexType name="CT_Hyperlink">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Action"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_Screentip"/>
  <xsd:attribute name="target" type="ST_String" use="optional"/>
  <xsd:attribute name="getTarget" type="ST_Delegate" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_BackstageLabelControl">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_AlignAttributes"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attribute name="noWrap" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_PrimaryItem">
  <xsd:choice minOccurs="1" maxOccurs="1">
    <xsd:element name="button" type="CT_BackstageRegularButton" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="menu" type="CT_BackstagePrimaryMenu" minOccurs="0" maxOccurs="1"/>
  </xsd:choice>
</xsd:complexType>
<xsd:group name="EG_BackstageMenuControls">
  <xsd:choice>
    <xsd:element name="button" type="CT_BackstageMenuButton"/>
    <xsd:element name="checkBox" type="CT_BackstageMenuCheckBox"/>
    <xsd:element name="menu" type="CT_BackstageSubMenu"/>
    <xsd:element name="toggleButton" type="CT_BackstageMenuToggleButton"/>
  </xsd:choice>
</xsd:group>
<xsd:complexType name="CT_BackstageMenuGroup">
  <xsd:sequence minOccurs="0" maxOccurs="1000">
    <xsd:group ref="EG_BackstageMenuControls"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="CT_BackstageMenuBase">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1000">
      <xsd:element name="menuGroup" type="CT_BackstageMenuGroup"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>

<xsd:complexType name="CT_BackstagePrimaryMenu">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageMenuBase">
      <xsd:attributeGroup ref="AG_Screentip"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_BackstageSubMenu">
  <xsd:complexContent>
    <xsd:extension base="CT_BackstageMenuBase">
      <xsd:attributeGroup ref="AG_Description"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:complexType name="CT_ImageControl">
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Image"/>
  <xsd:attributeGroup ref="AG_AltText"/>
</xsd:complexType>

<xsd:complexType name="CT_GroupControls">
  <xsd:choice minOccurs="0" maxOccurs="1000">
    <xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
  </xsd:choice>
</xsd:complexType>

<xsd:complexType name="CT_BackstageGroup">
  <xsd:sequence>
    <xsd:choice minOccurs="0" maxOccurs="1">
      <xsd:element name="primaryItem" type="CT_PrimaryItem" minOccurs="0" maxOccurs="1"/>
    </xsd:choice>
    <xsd:element name="topItems" type="CT_GroupControls" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="bottomItems" type="CT_GroupControls" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_GroupStyle"/>
  <xsd:attributeGroup ref="AG_HelperText"/>
  <xsd:attributeGroup ref="AG_ShowLabel"/>
</xsd:complexType>

<xsd:complexType name="CT_HeaderGroup">
  <xsd:sequence>
    <xsd:element name="button" type="CT_BackstageRegularButton" minOccurs="1" maxOccurs="100"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
</xsd:complexType>

<xsd:complexType name="CT_TaskGroup">
  <xsd:sequence>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="category" type="CT_TaskGroupCategory" minOccurs="0" maxOccurs="100"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_HelperText"/>
<xsd:attributeGroup ref="AG_ShowLabel"/>
<xsd:attribute name="allowedTaskSizes" type="ST_TaskSizes" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_TaskGroupCategory">
<xsd:sequence>
<xsd:element name="task" type="CT_TaskGroupTask" minOccurs="0" maxOccurs="1000"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
</xsd:complexType>
<xsd:complexType name="CT_TaskGroupTask">
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_Definitive"/>
<xsd:attributeGroup ref="AG_Image"/>
<xsd:attributeGroup ref="AG_Enabled"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_Description"/>
<xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
<xsd:complexType name="CT_TaskFormGroup">
<xsd:sequence>
<xsd:element name="category" type="CT_TaskFormGroupCategory" minOccurs="0" maxOccurs="100"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_HelperText"/>
<xsd:attributeGroup ref="AG_ShowLabel"/>
<xsd:attribute name="allowedTaskSizes" type="ST_TaskSizes" use="optional"/>
</xsd:complexType>
<xsd:complexType name="CT_TaskFormGroupCategory">
<xsd:sequence>
<xsd:element name="task" type="CT_TaskFormGroupTask" minOccurs="0" maxOccurs="1000"/>
</xsd:sequence>
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Visible"/>
</xsd:complexType>
<xsd:complexType name="CT_TaskFormGroupTask">
<xsd:attributeGroup ref="AG_IDAttributes"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_Image"/>
<xsd:attributeGroup ref="AG_Enabled"/>
<xsd:attributeGroup ref="AG_Label"/>
<xsd:attributeGroup ref="AG_Visible"/>
<xsd:attributeGroup ref="AG_Description"/>
<xsd:attributeGroup ref="AG_Keytip"/>
</xsd:complexType>
<xsd:complexType name="CT_GroupBox">
<xsd:sequence>
<xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
</xsd:sequence>
</xsd:complexType>
<xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attributeGroup ref="AG_Expand"/>
  <xsd:attributeGroup ref="AG_Label"/>
</xsd:complexType>

<xsd:simpleType name="ST_align">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="topLeft"/>
    <xsd:enumeration value="top"/>
    <xsd:enumeration value="topRight"/>
    <xsd:enumeration value="left"/>
    <xsd:enumeration value="center"/>
    <xsd:enumeration value="right"/>
    <xsd:enumeration value="bottomLeft"/>
    <xsd:enumeration value="bottom"/>
    <xsd:enumeration value="bottomRight"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_expand1">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="horizontal"/>
    <xsd:enumeration value="vertical"/>
    <xsd:enumeration value="both"/>
    <xsd:enumeration value="neither"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:simpleType name="ST_layoutChildren">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="horizontal"/>
    <xsd:enumeration value="vertical"/>
  </xsd:restriction>
</xsd:simpleType>

<xsd:complexType name="CT_LayoutContainer">
  <xsd:sequence>
    <xsd:group ref="EG_GroupControls" minOccurs="0" maxOccurs="1000"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDCustom"/>
  <xsd:attribute name="align" use="optional" type="ST_align"/>
  <xsd:attribute name="expand" use="optional" type="ST_expand1"/>
  <xsd:attribute name="layoutChildren" use="optional" type="ST_layoutChildren"/>
</xsd:complexType>

<xsd:group name="EG_GroupControls">
  <xsd:choice>
    <xsd:element name="button" type="CT_BackstageGroupButton"/>
    <xsd:element name="checkBox" type="CT_BackstageCheckBox"/>
    <xsd:element name="editBox" type="CT_BackstageEditBox"/>
    <xsd:element name="dropDown" type="CT_BackstageDropDown"/>
    <xsd:element name="radioGroup" type="CT_RadioGroup"/>
    <xsd:element name="comboBox" type="CT_BackstageComboBox"/>
    <xsd:element name="hyperlink" type="CT_Hyperlink"/>
    <xsd:element name="labelControl" type="CT_BackstageLabelControl"/>
    <xsd:element name="groupBox" type="CT_GroupBox"/>
    <xsd:element name="layoutContainer" type="CT_LayoutContainer"/>
    <xsd:element name="imageControl" type="CT_ImageControl"/>
  </xsd:choice>
</xsd:group>

<xsd:group name="EG_SimpleGroups">
  <xsd:choice>
    <xsd:element name="group" type="CT_BackstageGroup"/>
    <xsd:element name="taskGroup" type="CT_TaskGroup"/>
  </xsd:choice>
</xsd:group>

<xsd:complexType name="CT_BackstageGroups">
  <xsd:choice minOccurs="1" maxOccurs="1">
    <xsd:choice minOccurs="0" maxOccurs="1"/>
    <xsd:element name="taskFormGroup" type="CT_TaskFormGroup"/>
  </xsd:choice>
  <xsd:choice minOccurs="0" maxOccurs="1000"/>
  <xsd:group ref="EG_SimpleGroups"/>
</xsd:complexType>
<xsd:complexType>
  <xsd:sequence>
    <xsd:element name="firstColumn" type="CT_BackstageGroups" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="secondColumn" type="CT_SimpleGroups" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attributeGroup ref="AG_IDAttributes"/>
  <xsd:attributeGroup ref="AG_PositionAttributes"/>
  <xsd:attributeGroup ref="AG_Enabled"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Label"/>
  <xsd:attributeGroup ref="AG_Visible"/>
  <xsd:attributeGroup ref="AG_Title"/>
  <xsd:attribute name="columnWidthPercent" use="optional" type="ST_columnWidthPercent"/>
  <xsd:attribute name="firstColumnMinWidth" use="optional" type="ST_firstColumnMinWidth"/>
  <xsd:attribute name="firstColumnMaxWidth" use="optional" type="ST_firstColumnMaxWidth"/>
  <xsd:attribute name="secondColumnMinWidth" use="optional" type="ST_secondColumnMinWidth"/>
  <xsd:attribute name="secondColumnMaxWidth" use="optional" type="ST_secondColumnMaxWidth"/>
</xsd:complexType>
<xsd:complexType name="CT_CustomUI">
  <xsd:sequence>
    <xsd:element name="commands" type="CT_Commands" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="ribbon" type="CT_Ribbon" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="backstage" type="CT_Backstage" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="contextMenus" type="CT_ContextMenus" minOccurs="0" maxOccurs="1"/>
  </xsd:sequence>
  <xsd:attribute name="onLoad" type="ST_Delegate" use="optional"/>
  <xsd:attribute name="loadImage" type="ST_Delegate" use="optional"/>
</xsd:complexType>

<xsd:element name="customUI" type="CT_CustomUI"/>
</xsd:complexType>
6 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Office 2010 suites
- Microsoft Office 2013
- Microsoft Office 2016
- Microsoft Office 2019
- Microsoft Office 2021

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.
7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Revision class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 Relationship to Protocols and Other Structures</td>
<td>Updated ISO/IEC29500-1:2012 to its latest reference.</td>
<td>Minor</td>
</tr>
<tr>
<td>2.4.17 ST_ID</td>
<td>Updated Custom UI Command Table names.</td>
<td>Minor</td>
</tr>
<tr>
<td>6 Appendix B: Product Behavior</td>
<td>Updated list of supported products.</td>
<td>major</td>
</tr>
</tbody>
</table>
8 Index

A

Applicability 10

B

Backstage example 182

C

Change tracking 204
Complex types
CT_Backstage 12
CT_BackstageButtonBase 12
CT_BackstageCheckBox 14
CT_BackstageCheckBoxBase 16
CT_BackstageComboBox 17
CT_BackstageDropDown 19
CT_BackstageEditBox 21
CT_BackstageFastCommandButton 23
CT_BackstageGroup 25
CT_BackstageGroupButton 27
CT_BackstageGroups 29
CT_BackstageItem 29
CT_BackstageLabelControl 30
CT_BackstageMenuBase 31
CT_BackstageMenuButton 33
CT_BackstageMenuCheckBox 35
CT_BackstageMenuGroup 36
CT_BackstageMenuToggleButton 37
CT_BackstagePrimaryMenu 39
CT_BackstageRegularButton 41
CT_BackstageSubMenu 42
CT_BackstageTab 44
CT_Box 46
CT_Button 48
CT_ButtonGroup 51
CT_ButtonRegular 53
CT_CheckBox 56
CT_ComboBox 58
CT_Command 61
CT_Commands 62
CT_ContextMenu 62
CT_ContextMenus 63
CT_ContextualTabs 64
CT_Control 64
CT_ControlBase 67
CT_ControlClone 69
CT_ControlCloneQat 71
CT_ControlCloneRegular 74
CT_CustomUI 77
CT_DialogLauncher 77
CT_DropDownRegular 78
CT_DynamicMenu 81
CT_DynamicMenuRegular 84
CT_EditBox 87
CT_Gallery 90
CT_GalleryRegular 94
CT_Group 99
CT_GroupBox 102
CT_GroupControls 103
CT_HeaderGroup 104
CT_Hyperlink 105
CT_ImageControl 107
CT_Item 108
CT_LabelControl 109
CT_LayoutContainer 111
CT_Menu 112
CT_MenuRegular 115
CT_MenuRoot 119
CT_MenuSeparator 120
CT_MenuSeparatorNoTitle 121
CT_MenuWithTitle 122
CT_PrimaryItem 125
CT_Qat 125
CT_QatItems 126
CT_RadioButton 126
CT_Ribbon 128
CT_Separator 129
CT_SimpleGroups 130
CT_SplitButton 130
CT_SplitButtonBase 132
CT_SplitButtonRegular 135
CT_SplitButtonRestricted 137
CT_SplitButtonWithTitle 139
CT_Tab 140
CT_Tabs 142
CT_TabSet 143
CT_TaskFormGroup 143
CT_TaskFormGroupCategory 145
CT_TaskFormGroupTask 146
CT_TaskGroup 148
CT_TaskGroupCategory 150
CT_TaskGroupTask 152
CT_ToggleButton 154
CT_ToggleButtonRegular 157
CT_VisibleButton 160
CT_VisibleToggleButton 162
Context Menu example 181

E

Examples 181
Backstage 182
Context Menu 181
Ribbon 181

F

Fields - security index 183
Fields - vendor-extensible 10

G

Global elements
customUI 11
menu 11
Glossary 8

I

Implementer - security considerations 183