

Revised Disposition for CA-0072

Agreed; we should provide a mechanism to associate a navigation index to content controls and form controls in order to meet accessibility guidelines. We should also modify the specification to include specific details regarding the navigation order among elements in a presentation slide. [We should also provide informative text in the Accessibility Annex that documents the best practice for achieving predictable navigation.](#)

The following changes will be made:

In Part 4, §2.5.2, add a new subclause, which becomes the new §2.5.2.40:

tabIndex (Structured Document Navigation Order Index)

This element specifies the position of the current structured document tag in the navigation (tab) order used in the document. The index shall be stored on this element's val attribute and is analogous to the tabIndex attribute in HTML.

Objects that support tab index shall be navigated by consumers in the following order:

1. Objects for which the XML specifies a non-zero tabIndex value are navigated first. Navigation proceeds with the element with the lowest resolved value of tabIndex to the element with the highest resolved value of tabIndex.
 - a. Objects that specify identical resolved values of tabIndex will be navigated in the lexical order in which the elements appear in the underlying WordprocessingML.
2. Objects for which the XML does not specify an index or objects for which the XML specifies a resolved tabIndex value of 0 are navigated last. These objects are navigated in the lexical order in which they appear in the underlying WordprocessingML.

[*Example:* Consider the following two structured document tags where each structured document tag specifies a tab index:

```
<w:sdt>
  <w:sdtPr>
    <w:id w:val="5" />
    <w:tabIndex w:val="1" />
  </w:sdtPr>
  <w:sdtContent>
    <w:p>
      <w:r>
        <w:t>First Name</w:t>
      </w:r>
```

```

    </w:p>
  </w:sdtContent>
</w:sdt>
...
<w:sdt>
  <w:sdtPr>
    <w:id w:val="6" />
    <w:tabIndex w:val="2" />
  </w:sdtPr>
  <w:sdtContent>
    <w:p>
      <w:r>
        <w:t>Last Name</w:t>
      </w:r>
    </w:p>
  </w:sdtContent>
</w:sdt>

```

The `tabIndex` element specifies that the structured document tag with an identifier value of 5 shall be the first content to be reached via tabbing, whereas the structured document tag with an identifier value of 6 shall be the second content to be reached via tabbing. *end example*]

Attributes	Description
val (Positive Decimal Number Value)	<p>Specifies that the contents of this attribute will contain a positive decimal number.</p> <p>The contents of this positive decimal number are interpreted based on the context of the parent XML element.</p> <p><i>[Example: Consider the following numeric WordprocessingML property of type ST_UnsignedDecimalNumber:</i></p> <pre><w:... w:val="15" /></pre> <p>The value of the <code>val</code> attribute is a decimal number whose value must be interpreted in the context of the parent element. <i>end example</i>]</p> <p>The possible values for this attribute are defined by the <code>ST_UnsignedDecimalNumber</code> simple type (\$Error! Reference source not found.).</p>

In Part 4, §2.5.2.37, page 585, line 22, one more child element will be included in the Child Elements table

Child Elements	Subclause
tabIndex (Structured Document Navigation Order Index)	\$Error! Reference source

Child Elements	Subclause
	not found.
tag (Programmatic Tag)	§Error! Reference source not found.
temporary (Remove Structured Document Tag When Contents Are Edited)	§Error! Reference source not found.
text (Plain Text Structured Document Tag)	§Error! Reference source not found.

In Part 4, §2.16.17, page 1,586, line 16, one more child element will be included in the Child Elements table

Child Elements	Subclause
tabIndex (Form Field Navigation Order Index)	§Error! Reference source not found.
textInput (Text Box Form Field Properties)	§Error! Reference source not found. ³⁴

In Part 4, §2.16, add a new subclause, which becomes the new §2.16.33:

tabIndex (Form Field Navigation Order Index)

This element specifies the position of the current form field in the navigation (tab) order used in the document. The tabbing index shall be stored on this element's val attribute and is analogous to the tabIndex attribute in HTML.

Objects that support tab index shall be navigated by consumers in the following order:

1. Objects for which the XML specifies a non-zero `tabIndex` value are navigated first. Navigation proceeds with the element with the lowest resolved value of `tabIndex` to the element with the highest resolved value of `tabIndex`.
 - a. Objects that specify identical resolved values of `tabIndex` will be navigated in the lexical order in which the elements appear in the underlying WordprocessingML.
2. Objects for which the XML does not specify an index or objects for which the XML specifies a resolved `tabIndex` value of 0 are navigated last. These objects are navigated in the lexical order in which they appear in the underlying WordprocessingML.

[*Example:* Consider the following two text box form fields where form field specifies a tab index:

```

<w:ffData>
  <w:name w:val="FirstName" />
  <w:enabled />
  <w:textInput />
  <w:tabIndex w:val="1" />
</w:ffData>
...
<w:ffData>
  <w:name w:val="LastName" />
  <w:enabled />
  <w:textInput />
  <w:tabIndex w:val="2" />
</w:ffData>

```

The `tabIndex` element specifies that the `FirstName` form field shall be the first content to be reached via tabbing, whereas the `LastName` form field shall be the second content to be reached via tabbing. *end example*]

Attributes	Description
val (Positive Decimal Number Value)	<p>Specifies that the contents of this attribute will contain a positive decimal number.</p> <p>The contents of this positive decimal number are interpreted based on the context of the parent XML element.</p> <p>[<i>Example:</i> Consider the following numeric WordprocessingML property of type <code>ST_UnsignedDecimalNumber</code>:</p> <pre><w:... w:val="15" /></pre> <p>The value of the <code>val</code> attribute is a decimal number whose value must be interpreted in the context of the parent element. <i>end example</i>]</p> <p>The possible values for this attribute are defined by the <code>ST_UnsignedDecimalNumber</code> simple type (\$Error! Reference source not found.).</p>

In wml.xsd:

The tabIndex element will be added to the CT_SdtPr and CT_FFData content models.

In Part 4, §4.4.1.42, page 3,051, line 16

This element specifies all shape-based, either grouped or not, that can be referenced on a given slide.

In Part 4, §4.4.1.42, page 3,052, after line 5

Each shape-based object within the shape tree, whether grouped or not, shall represent one unique level of z-ordering on the slide. The z-order for each shape-based object shall be determined by the lexical ordering of each shape-based object within the shape tree: the first shape-based object shall have the lowest z-order, while the last shape-based object shall have the highest z-order.

The z-ordering of shape-based objects within the shape tree shall also determine the navigation (tab) order of the shape-based objects: the shape-based object with the lowest z-order (the first shape in lexical order) shall be first in navigation order, with objects being navigated in ascending z-order.

[*Example:* Consider the following PresentationML slide with two shapes

```
<p:sld>
  <p:cSld>
    <p:spTree>
      ...
      <p:sp>
        <p:nvSpPr>
          <p:cNvPr id="5" name="Oval 4" />
          ...
        </p:nvSpPr>
        ...
      </p:sp>
      <p:sp>
        <p:nvSpPr>
          <p:cNvPr id="4" name="Isosceles Triangle 3" />
          ...
        </p:nvSpPr>
        ...
      </p:sp>
    </p:spTree>
  </p:cSld>
  ...
</p:sld>
```

In the above example the shape with name Oval 4 has the lowest z-order value since that shape is the first shape in the shape tree. Oval 4 is also the first shape in navigation order. The shape with name

Isosceles Triangle 3 has the highest z positioning value since that shape is the last shape in the shape tree. Isosceles Triangle 3 is also the last shape in navigation order. *end example*]

In Part 4, Annex G. Accessibility Best Practices, Best Practice for Document and Template Authors

Delivering Predictable Navigation within Forms

Although it is often the case that navigating documents is simple by virtual of the natural flow of documents, it can become complex in the case of forms being found in the document. It is highly desirable for document and template authors to define those forms such that the navigation from control to control is both intelligent and predictable. Office Open XML provides a mechanism for defining the navigation flow (also known as tab order) for such controls.

[*Example:* Consider part of a well known form:

1. Number of qualifying children: _____ × \$1,000. Enter the result.	1
---	---

The following mark up can be used to make the form controls on that line accessible:

```
<w:p>
  <w:sdt>
    <w:sdtPr>
      <w:id w:val="1" />
    </w:sdtPr>
    <w:sdtContent>
      <w:sdt>
        <w:sdtPr>
          <w:id w:val="2" />
        </w:sdtPr>
        <w:sdtContent>
          <w:r>
            <w:t>Number of qualifying children:</w:t>
          </w:r>
        </w:sdtContent>
      </w:sdt>
    </w:sdtContent>
  </w:p>
  ...
  <w:r>
    ...
    <w:fldChar w:fldCharType="begin">
      <w:ffData>
        <w:name w:val="NumberOfChildren" />
        <w:enabled />
```

```

        ...
        <w:textInput />
        <w:label w:val="2" />
        <w:tabIndex w:val="1" />
    </w:ffData>
</w:fldChar>
</w:r>
...
<w:r>
    <w:t xml:space="preserve"> X $1,000</w:t>
</w:r>
</w:sdtContent>
</w:sdt>
</w:p>
<w:p>
    ...
    <w:sdt>
        <w:sdtPr>
            <w:id w:val="3" />
        </w:sdtPr>
        <w:sdtContent>
            <w:r>
                <w:t>Enter the result</w:t>
            </w:r>
        </w:sdtContent>
    </w:sdt>
</w:p>
<w:tbl>
    ...
    <w:sdt>
        <w:sdtPr>
            <w:id w:val="4" />
        </w:sdtPr>
        <w:sdtContent>
            <w:tr>
                ...
                <w:tc>
                    ...
                    <w:p>
                        <w:r>
                            <w:t>1</w:t>
                        </w:r>
                    </w:p>
                </w:tc>
            </w:tr>
        </w:sdtContent>
    </w:sdt>
</w:tbl>

```

```

    </w:tc>
    <w:tc>
      ...
      <w:p>
        <w:r>
          <w:fldChar w:fldCharType="begin">
            <w:ffData>
              <w:name w:val="Line1Value" />
              <w:enabled />
              ...
              <w:textInput />
              <w:label w:val="1" />
              <w:label w:val="3" />
              <w:tabIndex w:val="2" />
            </w:ffData>
          </w:fldChar>
        </w:r>
        ...
      </w:p>
    </w:tc>
  </w:tr>
</w:sdtContent>
</w:sdt>
</w:tbl>

```

The first line of this form is broken up into two Form Text Fields and four content controls that are used as labels. The Form Text Field with a name of NumberOfChildren is associated in terms of its label with the content control that has an id value of 2. In this case, the Form Text Field with a name of NumberOfChildren has a label of Number of qualifying children:. According to the tabIndex element the first Form Text Field is navigated first. *end example*]

In Part 4, Annex G. Accessibility Best Practices, Best Practice for Customers

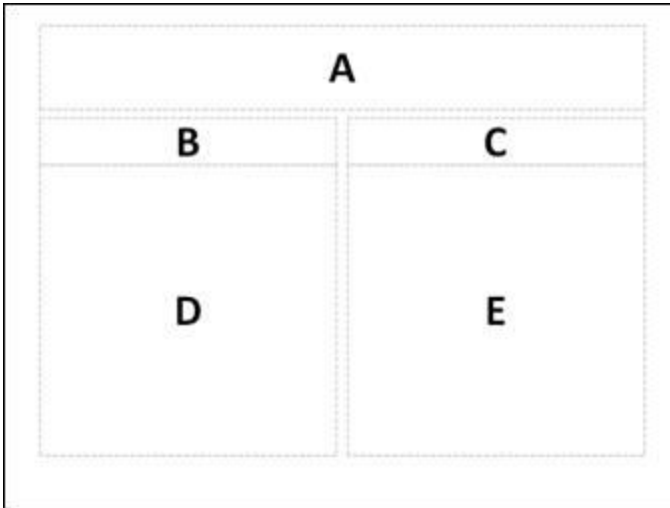
Techniques for Improving Document Navigation

It is highly desirable for customers to author their documents in such a way that there is predictability in terms of navigation. The following describes the techniques that the customer can use to make use built-in primitives in Office Open XML, as well as how such techniques change the underlying document data, to achieve the desired outcome.

Navigating Shapes on Slides

In cases where a reader is navigating slides, navigation is complicated by the 2D-nature of the slide. Office Open XML provides a remedy for this by allowing customers to define the reading order, or navigation order, of shapes on a slide through the use of z-order.

[*Example:* Consider the case that a presentation author elects to construct a slide that contains five placeholders: one for the title (A) , two for list headings (B, C) and two lists (D, E). Visually, this could be represented as follows:



Further, suppose that the default tab order (aka reading order) for such a slide, in a LTR-orientation, is A, B, D, C, E.

In DrawingML, this ordering is accomplished by the lexical ordering of the five shapes, A - E. In the shape tree for this slide, you would see this:

```
<p:spTree>
  <p:nvGrpSpPr>...</p:nvGrpSpPr>
  <p:grpSpPr>...</p:grpSpPr>
  <p:sp>
    ...
    <p:txBody>...
      <a:r>
        <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
        <a:t>A</a:t>
      </a:r>...
    </p:txBody>
  </p:sp>
  <p:sp>
    ...
    <p:txBody>...
      <a:r>
        <a:rPr lang="en-US" sz="4800" dirty="0" smtClean="0"/>
        <a:t>B</a:t>
      </a:r>...
```

```

        </p:txBody>
    </p:sp>
    <p:sp>
        ...
        <p:txBody>...
            <a:r>
                <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
                <a:t>D</a:t>
            </a:r>...
        </p:txBody>
    </p:sp>
    <p:sp>
        ...
        <p:txBody>...
            <a:r>
                <a:rPr lang="en-US" sz="4800" dirty="0" smtClean="0"/>
                <a:t>C</a:t>
            </a:r>...
        </p:txBody>
    </p:sp>
    <p:sp>
        ...
        <p:txBody>...
            <a:r>
                <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
                <a:t>E</a:t>
            </a:r>...
        </p:txBody>
    </p:sp>
</p:spTree>

```

If this default reading order was not what the presentation author preferred, the author could specify the z-order for each shape to determine the ordering of the shapes in the shape tree, and by extension, the ordering that shapes would be read in the presentation. The shapes will show up in the shape tree starting with the shape farthest “back” and ending with the shape in the “front.”

Using our example above, if the author wanted to change the ordering of the shapes to facilitate a more RTL-like reading order (A, C, E, B, D), she could select shape E and send it to the back, then select shape C and sent it to the back and finally select shape A and send it to the back. Naturally the user experience here is arbitrary; the important aspect is that the ordering of the shapes within the shape tree is changed to the preferred reading order. With these three changes, the shape tree is changed to this:

```

<p:spTree>
  <p:nvGrpSpPr>...</p:nvGrpSpPr>
  <p:grpSpPr>...</p:grpSpPr>
  <p:sp>
    ...
    <p:txBody>...
      <a:r>
        <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
        <a:t>A</a:t>
      </a:r>...
    </p:txBody>
  </p:sp>
  <p:sp>
    ...
    <p:txBody>...
      <a:r>
        <a:rPr lang="en-US" sz="4800" dirty="0" smtClean="0"/>
        <a:t>C</a:t>
      </a:r>...
    </p:txBody>
  </p:sp>
  <p:sp>
    ...
    <p:txBody>...
      <a:r>
        <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
        <a:t>E</a:t>
      </a:r>...
    </p:txBody>
  </p:sp>
  <p:sp>
    ...

```

```
<p:txBody>...
  <a:r>
    <a:rPr lang="en-US" sz="4800" dirty="0" smtClean="0"/>
    <a:t>B</a:t>
  </a:r>...
</p:txBody>
</p:sp>
<p:sp>
...
<p:txBody>...
  <a:r>
    <a:rPr lang="en-US" sz="4800" b="1" dirty="0"
smtClean="0"/>
    <a:t>D</a:t>
  </a:r>...
</p:txBody>
</p:sp>
</p:spTree>
end example]
```