

SaveAsXML Developer Information for Creating or Modifying Mapping Tables

ADOBE SYSTEMS INCORPORATED

Copyright 2001 Adobe Systems Incorporated
All Rights Reserved

Updated: 19 November 2002 - 10:43

1 Introduction

SaveAsXML is a plug-in for Acrobat 6.0 which extends the 'Save as type' choices in the SaveAs dialog to allow a Tagged PDF document to be saved as a number of XML, HTML, or similar text-based formats.

Mapping Tables are used to control the conversion process for the SaveAsXML feature. The Mapping Tables are a script of hierarchically organized directives written in a custom language defined in XML syntax. This allows developers to create custom Mapping Tables for formats other than those provided in this package. This document provides an overview of that language.

2 Overview of the SaveAsXML Process

As the SaveAsXML plug-in registers itself with Acrobat 6.0, it inspects the set of XML files in the MappingTables folder to determine the number of conversion services that are available.

Note: The MappingTables folder must be found in the same folder as the SaveAsXML.api file. Only files in this MappingTables folder will be inspected as potential conversion services supported by this plug-in. This folder may not contain any files ending in a _____.xml suffix that are not Mapping Tables.

If it finds the Root element and its menu-name attribute, it adds the menu-name to the list of file format choices available in the SaveAs dialog.

Note: One should note that the menu-name must be unique, or the user may be confused by similarly identified entries in the SaveAs dialog's file formats pulldown.

The following simplified & incomplete sample Mapping Table will be used to explain the basic operation of the SaveAsXML processing. It is shown first in its entirety, then is followed by one that is annotated.

```
<Root    File-format = "Xml-1-00" Menu-name = "Sample Mapping Table"
        Mac-creator = "MSIE" Mac-type = "TEXT" Win-suffix = ".xml"
        Encode-out = "Utf-8-out">
  <Emit-string ... >&lt;XML-Doc&gt;</Emit-string>
  <Walk-structure Use-event-list = "Block-events"></Walk-structure>
  <Emit-string ...>&lt;/XML-Doc&gt;</Emit-string>
  <Define-event-list Name = "Block-events">
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
              Node-name = "Div" Alternate-name = "-none-"
              Node-content = "Has-kids" Event-class = "Enter">
      <Emit-string ...>&lt;Div&gt;</Emit-string>
      <Call-proc-list Name = "Block-attributes"></Call-proc-list>
      <Emit-string ...>&gt;</Emit-string>
      <Walk-children Use-event-list = "Inline-events"></Walk-children>
    </Event>
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
              Node-name = "Div" Alternate-name = "-none-"
              Node-content = "Has-kids" Event-class = "Exit">
      <Emit-string ...>&lt;/Div&gt;</Emit-string>
    </Event>
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
              Node-name = "Div" Alternate-name = "-none-"
              Node-content = "Empty" Event-class = "Enter">
      <Emit-string ...>&lt;Div&gt;</Emit-string>
      <Call-proc-list Name = "Block-attributes"></Call-proc-list>
      <Emit-string ...>/&gt;</Emit-string>
    </Event>
  </Define-event-list>
  <Define-event-list Name = "Inline-events">
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
              Node-name = "Span" Alternate-name = "-none-"
              Node-content = "Has-kids" Event-class = "Enter">
      <Emit-string ...>&lt;Span&gt;</Emit-string>
      <Call-proc-list Name = "Span-attributes"></Call-proc-list>
      <Emit-string ...>&gt;</Emit-string>
      <Walk-children Use-event-list = "Inline-events"></Walk-children>
    </Event>
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
              Node-name = "Span" Alternate-name = "-none-"
              Node-content = "Has-kids" Event-class = "Exit">
```

```

        <Emit-string ...>&lt;/Span>></Emit-string>
    </Event>
    <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
                Node-name = "Span" Alternate-name = "-none-"
                Node-content = "Empty" Event-class = "Enter">
        <Emit-string ...>&lt;/Span></Emit-string>
        <Call-proc-list Name = "Span-attributes"></Call-proc-list>
        <Emit-string ...>&gt;</Emit-string>
    </Event>
    <Event      Inf-type = "Pds-mc" Name-type = "Any" Node-name = "-none-"
                Alternate-name = "-none-" Node-content = "Has-text-only"
                Event-class = "Enter">
        <Proc-doc-text do-br-substitution = "do-br-substitution"></Proc-doc-text>
    </Event>
</Define-event-list>
<Define-proc-list Name = "Block-attributes">
    <Proc-var    Pdf-var = "Alt" Owner = "Structelem" Type = "String"
                Has-enum = "No-enum" Inherit = "Not-inherited" Default = "-none-"
                Condition = "Has-value">
        <Emit-string ...>alt=</Emit-string>
        <Proc-string></Proc-string>
        <Emit-string ...>"</Emit-string>
    </Proc-var>
</Define-proc-list>
<Define-proc-list Name = "Span-attributes">
    <Proc-var    Pdf-var = "ActualText" Owner = "Structelem" Type = "String"
                Has-enum = "No-enum" Inherit = "Not-inherited" Default = "-none-"
                Condition = "Always">
        <Emit-string ...>actual-text=</Emit-string>
        <Proc-string></Proc-string>
        <Emit-string ...>"</Emit-string>
    </Proc-var>
</Define-proc-list>
</Root>

```

Once the user selects an applicable file format in the SaveAs dialog, the dialog handler activates the SaveAsXML plug-in. It reads in the associated Mapping Table and converts it to a binary in-memory format. This is used to control the processing of the current TaggedPDF document. Processing begins with the root node of the Mapping Table and generally proceeds as a pre-order hierarchical traversal of the control nodes.

```

<Root      File-format = "Xml-1-00" Menu-name = "Sample Mapping Table"
            Mac-creator = "MSIE" Mac-type = "TEXT" Win-suffix = "xml"
            Encode-out = "Utf-8-out">

```

In processing the Root node of the Mapping Table, the SaveAsXML processor opens the output file using the filepath and name of the PDF document to be saved, replacing the file suffix with that specified by the Win-suffix attribute in this node (and on the Macintosh, the Mac-creator and Mac-type are also used to open the output file.) The remaining attributes in the Root node are available to the SaveAsXML processor and are internally used to control or optimize the conversion.

```

    <Emit-string ... >&lt;/XML-Doc>></Emit-string>

```

The Emit-string directive causes its content to be translated to the output encoding specified in the Encode-out attribute of the Root node and then emits the converted data to the output file. In this case, it issues the start tag for the document:

```

<XML-Doc>

```

Note: For clarity, the additional attributes of the Emit-string directive have been omitted.

Note: < is the entity for the "<" character, > is the entity for the ">" character, and & is the entity for the "&" character. Entities should be used in the Mapping Tables for these 3 characters when part of the data to a Mapping Table directive.

```
<Walk-structure Use-event-list = "Block-events"></Walk-structure>
```

The Walk-structure directive causes the SaveAsXML processor to walk the first level *Structural Elements* (**Kids** array of the **StructRoot**) of the Tagged PDF document to be saved. (A discussion of how to walk subsequent levels appears in the discussion of the Walk-children directive, below.)

Structural Elements are traversed in the order found in the *Logical Structure Tree*. An event is generated on entering and on exiting each *Structural Element*. The event-list specified by the Use-event-list attribute of the Walk-structure directive is searched for a matching Event directive (described below under the Define-event-list directive).

- If a match is found then the directives within that Event directive are processed (which may include the recursive processing of children of the current *Structural Element* via a Walk-children directive). Searching of the event-list is terminated and the next event is generated.
- If no match is found (or when processing is completed on the matching Event directive) then the next event is generated.

Processing continues until all first-level *Structural Elements* (**Kids** of the **StructRoot**) have been traversed, then the directive following the Walk-structure directive will be processed. In this case, it is:

```
<Emit-string Emit-space-after = "Emit-space-after" ...>
&#lt;/XML-Doc&gt;
</Emit-string>
```

This Emit-string directive issues the end tag: </XML-Doc>. Since newlines and spaces are often modified or stripped by various XML tools, the Emit-space-after attribute (and the other related attributes of the Emit-string directive) guarantees the retention of these characters.

```
<Define-event-list Name = "Block-events">
```

The Define-event-list directive is similar to a macro or subroutine definition in most programming languages. It encapsulates and names a set of event directives that are activated by a Walk-structure, a Walk-children, or a Call-event-list directive having a corresponding name in its Use-event-list attribute.

```
<Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
            Node-name = "Div" Alternate-name = "-none-"
            Node-content = "Has-kids" Event-class = "Enter">
```

The Event directive includes a set of attributes that are used to determine if the directives within it are to be processed. For a complete description of these attributes, see the full specification of this directive in the next section of this document. The directive above will be activated by a Entering (either from a parent element or from the prior peer element) a *Structural Element* (Inf-type = "Struct-elem"),

when the element is role-mapped (Name-type = "Structure-role") to "Div" and the element has children (see the 2nd event directive below for an element that has no children).

When an Event directive is activated, the directives within it (before its </Event> tag) are processed. In this case:

```
<Emit-string ...>&lt;Div</Emit-string>
```

Issue the "<Div " portion of the output element's start-tag.

```
<Call-proc-list Name = "Block-attributes"></Call-proc-list>
```

The Call-proc-list directive will process the properties associated with this *Structural Element*, using the processing list specified by the Name property on the Call-proc-list directive.

Although the event-list processing stops on the first match, the proc-list processing will continue for every directive in the selected processing list. (The Block-attributes proc-list is described later in this example.)

```
<Emit-string ...>&gt;</Emit-string>
```

Issues the closing ">" on the output element's start-tag.

```
<Walk-children Use-event-list = "Inline-events"></Walk-children>
```

The Walk-children directive is functionally identical to the Walk-structure directive (described earlier in this example), except that it walks the first level children of the current *Structural Element*.

```
</Event>
```

The </Event> tag indicates the end of the processing for this event. Remaining entries in this event-list follow a similar model.

The next Event included in this event-list handles events that will be generated when exiting Div elements that have children. This will generate the close tag on the output element.

```
<Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
            Node-name = "Div" Alternate-name = "-none-"
            Node-content = "Has-kids" Event-class = "Exit">
  <Emit-string ...>&lt;/Div&gt;</Emit-string>
</Event>
```

The final Event directive included in this event-list handles events that will be generated on entering an element which has no children. (Note that it does not and should not contain a Walk-children directive.)

```
<Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
            Node-name = "Div" Alternate-name = "-none-"
            Node-content = "Empty" Event-class = "Enter">
  <Emit-string ...>&lt;Div</Emit-string>
  <Call-proc-list Name = "Block-attributes"></Call-proc-list>
  <Emit-string ...>/&gt;</Emit-string>
</Event>
</Define-event-list>
```

The </Define-event-list> tag ends the list of entries in the Block-events event-list.

The following event-list is for handling inline elements and is similar to the one above.

```
<Define-event-list Name = "Inline-events">
  <Event      Inf-type = "Struct-elem" Name-type = "Structure-role"
```

```

Node-name = "Span" Alternate-name = "-none-"
Node-content = "Has-kids" Event-class = "Enter">
<Emit-string ...>&lt;Span</Emit-string>
<Call-proc-list Name = "Span-attributes"></Call-proc-list>
<Emit-string ...>&gt;</Emit-string>
<Walk-children Use-event-list = "Inline-events"></Walk-children>
</Event>
<Event Inf-type = "Struct-elem" Name-type = "Structure-role"
Node-name = "Span" Alternate-name = "-none-"
Node-content = "Has-kids" Event-class = "Exit">
<Emit-string ...>&lt;/Span&gt;</Emit-string>
</Event>
<Event Inf-type = "Struct-elem" Name-type = "Structure-role"
Node-name = "Span" Alternate-name = "-none-"
Node-content = "Empty" Event-class = "Enter">
<Emit-string ...>&lt;Span</Emit-string>
<Call-proc-list Name = "Span-attributes"></Call-proc-list>
<Emit-string ...>/&gt;</Emit-string>
</Event>

```

For event-lists that process *Structural Elements* that contains text or graphics, an Event entry like the following is needed. The code in the SaveAsXML plug-in that traverses the *Logical Structure Tree* also reports entering and exiting of the marked content containers (the wrappers around the low-level text and graphic content in the PDF page's marking stream). The labels on these nodes are hidden in the Tags view in Acrobat. (The corresponding Event for a Pds-mc element where the content is Image is slightly more complex. You should look at the Mapping Tables distributed with SaveAsXML for complete examples.)

```

<Event Inf-type = "Pds-mc" Name-type = "Any" Node-name = "-none-"
Alternate-name = "-none-" Node-content = "Has-text-only"
Event-class = "Enter">

```

This Event directive processes the low-level marked content containers (Inf-type = "Pds-mc") that actually contain the text (Node-content = "Has-text-only"). A corresponding exit directive is not required.

```

<Proc-doc-text do-br-substitution = "do-br-substitution"></Proc-doc-text>

```

The Proc-doc-text directive converts the text from the active marked content container in the PDF page's marking stream to the output encoding specified in the Encode-out attribute of the Root node and then emits the converted data to the output file. The do-br-substitution attribute controls whether the LF character is to be converted to a
 tag in the output stream, converted to a space, or discarded.

```

</Event>
</Define-event-list>

```

```

<Define-proc-list Name = "Block-attributes">

```

The Define-proc-list directive is also a macro/subroutine similar to the Define-event-list directive. Whereas the event-list describes how to process transition events in traversing the *Logical Structure Tree*, the proc-list describes how to process the properties (attributes) of a *Structural Element*.

```

<Proc-var Pdf-var = "Alt" Owner = "Structelem" Type = "String"
Has-enum = "No-enum" Inherit = "Not-inherited" Default = "-none-"
Condition = "Has-value">

```

The Proc-var directive searches an internal cache of the properties on the current *Structural Element* for the value of the property specified by its Pdf-var and Owner attributes. If inheritance is enabled, it also searches the cached properties of all

ancestors of the current *Structural Element* for an applicable value. Once it determines if there is (or is not) a value, it then uses the remaining attributes to determine if the value should be processed. If it determines it should be processed, then the directives contained in this Proc-var directive will be processed.

```
<Emit-string ...>alt="</Emit-string>
<Proc-string></Proc-string>
```

The Proc-string directive causes the string selected by the containing Proc-var directive to be translated to the output encoding specified in the Encode-out attribute of the Root node and then emits the converted data to the output file.

```
<Emit-string ...>"</Emit-string>
</Proc-var>
</Define-proc-list>
```

The </Define-proc-list> tag indicates the end of this proc-list.

The following proc-list has a similar organization to the one, above, for Block-attributes.

```
<Define-proc-list Name = "Span-attributes">
  <Proc-var Pdf-var = "ActualText" Owner = "Structelem" Type = "String"
    Has-enum = "No-enum" Inherit = "Not-inherited" Default = "-none-"
    Condition = "Always">
    <Emit-string ...>actual-text="</Emit-string>
    <Proc-string></Proc-string>
    <Emit-string ...>"</Emit-string>
  </Proc-var>
</Define-proc-list>
```

```
</Root>
```

The </Root> tag is the last line of a Mapping Table file. It indicated the end of the Root directive.

For the details on the full list of directives and their attributes, consult the following section of this document.

Note: You may also wish to look at any or all of the Mapping Tables distributed with SaveAsXML for examples of the usage of these directives. Every directive that is currently supported has been used in one or more of these tables.

3 Mapping Table Elements

3.1 Root

This is the root node of a Mapping Table.

Its attributes specify the name of the filter to appear in the menu and information necessary to properly generate the output file name and type information.

DTD content rule for this element:

(Comment | Emit-string | Define-event-list | Define-proc-list | Walk-metadata |
Emit-all-metadata | Walk-cached-property-sets | Walk-structure | Walk-layout)+

Table 1 *Attribute list for <Root>*

| Name | Type | Description |
|-------------|--------|---|
| File-format | Choice | Required — Internal name that describes the format of the output file (must be unique). The following formats are provided at release: Html-3-02 Html-4-01-with-css-1-00 Xml-1-00 Plain Text |
| Menu-name | String | Required — The text string describing the file format that will appear in the SaveAs dialog's pulldown menu. |
| Mac-creator | String | Required — The file creator field for a Macintosh file. |
| Mac-type | String | Required — The file type field for a Macintosh file. |
| Win-suffix | String | Required — The 3 letter filetype suffix for the Windows environment. Also used on Macintosh files. |
| Encode-out | Choice | Required — The encoding of the output file. Utf-8-out The file will be encoded in UTF-8 (8-bit Unicode). Utf-16-out The file will be encoded in UTF-16 (16-bit Unicode). Ucs-4-out The file will be encoded in UCS-4 (32-bit Unicode) Iso-latin-1-out The file will be encoded as ISO-Latin-1. All Unicode values above 0x00FF will be output as numeric character entities (). Html-ascii-out The file will be encoded as 7-bit ASCII. All Unicode values above 0x007F will be output as numeric character entities (). |

3.2 Walk-layout

THIS DIRECTIVE IS NOT SUPPORTED IN THIS VERSION OF SaveAsXML.

3.3 Walk-metadata

Directs the SaveAs processor to walk the DocInfo metadata portion of the PDF document.

DTD content rule for this element:

Void?

Table 2 *Attribute list for <Walk-metadata>*

| Name | Type | Description |
|---------------|--------|--|
| Use-proc-list | String | Required — The name of an event processing list (see <define-proc-list>) to be used to process the attributes found by walking the metadata portion of the document. |

3.4 Emit-all-metadata

The full set of XAP metadata will be copied to the output file.

DTD content rule for this element:

Void?

Table 3 *Attribute list for <Emit-all-metadata>*

| Name | Type | Description |
|---------------------|--------|--|
| Emit-space-before | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-space-before Emit a space before emitting any content text. |
| Emit-space-after | Choice | No-space-before Do not emit a space before emitting any content text. |
| | | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| Emit-space-after | Choice | Emit-space-after Emit a space after emitting any content text. |
| | | No-space-after Do not emit a space after emitting any content text. |
| Emit-newline-before | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-newline-before Emit a newline before emitting any content text. |
| Emit-newline-after | Choice | No-newline-before Do not emit a newline before emitting any content text. |
| | | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| Emit-newline-after | Choice | Emit-newline-after Emit a newline after emitting any content text. |
| | | No-newline-after Do not emit a newline after emitting any content text. |

3.5 Walk-structure

Directs the SaveAs processor to walk the *Logical Structure Tree* and associated content of the PDF document.

DTD content rule for this element:

Void?

Table 4 Attribute list for <Walk-structure>

| Name | Type | Description |
|----------------|--------|---|
| Use-event-list | String | Required — The name of an event processing list (see <define-event-list>) to be used to process the events generated by walking the structure tree (PDF Logical Structure) of the document. |

3.6 Walk-cached-property-sets

Directs the SaveAs processor to construct a stylesheet cache and walk the stylesheet data.

DTD content rule for this element:

Void?

Table 5 Attribute list for <Walk-cached-property-sets>

| Name | Type | Description |
|----------------|--------|---|
| Use-event-list | String | Required — The name of an event processing list (see <define-event-list>) to be used to process the events generated by walking the stylesheet data (ClassMap and class information) of the document. |

3.7 Walk-children

Directs the SaveAs processor to walk the kids list of the current Structural Element.

DTD content rule for this element:

Void?

Table 6 Attribute list for <Walk-children>

| Name | Type | Description |
|----------------|--------|--|
| Use-event-list | String | Required— The name of an event processing list (see <define-event-list>) to be used to process the events generated by walking the first-level children of the current <i>Structural Element</i> . |

3.8 Define-event-list

Event-lists and proc-lists are like macros, they allow the user to define a series of processing directives which may be used in multiple locations within the SaveAs Mapping Table.

- Event-lists govern the selection and processing of elements in the layout, metadata, logical structure, or stylesheet trees.
- Proc-lists govern the processing of attributes/properties associated with a given event/*Structural Element*.

DTD content rule for this element:

(Comment | Event | Call-event-list)+

Table 7 Attribute list for <Define-event-list>

| Name | Type | Description |
|------|--------|--|
| Name | String | Required — The name to be applied to the event processing list being defined by this element. This is referenced in the <Walk-*> elements via the "Use-event-list" attribute. The name must be unique across all Define-event-list elements within a given Mapping Table file. |

3.9 Call-event-list

Identical to a macro call, inserts the named event-list at this point in the Mapping Table.

DTD content rule for this element:

Void?

Table 8 Attribute list for <Call-proc-list>

| Name | Type | Description |
|------|--------|---|
| Name | String | Required — The name of a event list (see <Define-event-list>) to be included at this point in the current event list. |

3.10 Event

Governs the processing of a node in the layout, logical-structure, metadata, or stylesheet trees. Used to specify the processing that is to be performed on entering or exiting the named node.

DTD content rule for this element:

(Comment | Emit-string | Conditional-prefix | Element-name | Proc-var | Walk-proplist | Call-proc-list | Conditional-suffix | Proc-graphic-content | Proc-image-content | Proc-doc-text | Walk-children | Walk-metadata | Emit-all-metadata | Walk-cached-property-sets | Walk-structure | Walk-layout | Evaluate-var)+

Table 9 *Attribute list for <Event>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|--------------|----------------------|---|
| Node-type | Choice | Required — The Node-name attribute is matched against either the /S key of the StructElem (Structure-user-label) or against the result of processing that key via the RoleMap (Structure-role). |
| | Any | Attempt match on Structure-user-label then on Structure-role. Also used for matching within metadata an stylesheet construction. |
| | Structure-role | Compare Node-name to the result of processing the StructElem's /S key via the RoleMap. |
| | Structure-user-label | Compare Node-name to the StructElem's /S key. |
| Node-name | String | Required — Name of the element/role to match, in order to select this event descriptor for processing. |
| Node-content | Choice | Required |
| | Empty | Node has no children or direct content. |
| | Has-text-only | Node has only text content (no other elements). |
| | Has-kids | Node has child elements (including possible text-only spans). |
| | Graphic | Node contains (vector) graphic data. |
| | Image | Node contains bitmapped image data. |
| | Other | Node is something other than those listed above. |
| Event-class | Choice | Required — Identifies which transition into or out of the node is to be processed using this event description. |
| | Enter | Node is being entered from either parent or peer. |
| | Enter-from-parent | Node is being entered from parent, but not from peer. |
| | Enter-from-peer | Node is being entered from peer, but not from parent. |
| | Exit | Node is being exited to either parent or to peer. |
| | Exit-to-parent | Node is being exited to parent, but not to peer. |
| | Exit-to-peer | Node is being exited to peer, but not to parent. |
| | Begin-children | Node is being exited to begin processing its children. |
| | End-children | Node is being re-entered after processing its children. |

3.11 Define-proc-list

Event-lists and proc-lists are like macros, they allow the user to define a series of processing directives which may be used in multiple locations within the SaveAs Mapping Table.

- Event-lists govern the selection and processing of elements in the layout, metadata, logical structure, or stylesheet trees.
- Proc-lists govern the processing of attributes/properties associated with a given event/*Structural Element*.

DTD content rule for this element:

(Comment | Proc-var | Walk-proplist | Call-proc-list)+

Table 10 *Attribute list for <Define-proc-list>*

| Name | Type | Description |
|------|--------|---|
| Name | String | Required — The name to be applied to the variable processing list being defined by this element. This is referenced in the <Call-proc-list> element via its "Name" attribute. The name must be unique across all Define-proc-list elements within a given Mapping Table file. |

3.12 Call-proc-list

Identical to a macro call, inserts the named proc-list at this point in the Mapping Table.

DTD content rule for this element:

Void?

Table 11 *Attribute list for <Call-proc-list>*

| Name | Type | Description |
|------|--------|--|
| Name | String | Required — The name of a variable processing list (see <define-proc-list>) to be included at this point in the current event or proc-list. |

3.13 Proc-var

Specifies the formatting/conversion of the named attribute/property (PDF-variable). The proc-var directive also caches the data value and type of the value specified for use by various proc-* directives within this proc-var element.

DTD content rule for this element:

(Comment | Conditional-delimiter | Emit-string | Conditional-prefix | Element-name | Proc-string | Proc-integer | Proc-fixed | Proc-length | Proc-pixels | Proc-enum | Proc-doc-text | Proc-graphic-content | Proc-image-content | Conditional-suffix)+

Table 12 *Attribute list for <Proc-var>*

| Name | Type | Description |
|---------|------------|---|
| Pdf-var | String | Required — The name of a property in a given property dictionary (see Owner) to be processed/evaluated. |
| Owner | Choice | Required — The owner of the property dictionary. |
| | Metadata | This is a pseudo-owner for entries in the document's metadata. |
| | Structelem | This is a pseudo-owner for properties specified directly in the StructElem's obj dictionary. |

| | | | |
|-----------|--------|-----------------------------|--|
| | | Layout | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Layout. |
| | | Link | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Link. |
| | | List | Properties in the StructElem's Attribute dictionary list within the dictionary owned by List. |
| | | Table | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Table. |
| | | Auto-span | This is a pseudo-owner generated by the SaveAs processor for each span it synthesizes by consolidating Tj operators having common styling properties (font, size, color, etc.) |
| | | Inline-markup | This is a pseudo-owner generated by the SaveAs processor when the inline marking of /Span << ... >> BDC (abbrev.) Tj EMC is encountered. |
| Type | Choice | Required | — The primary PDF datatype of the property (see Has-enum for a possible secondary datatype). |
| | | Fixed | Fixed-point number. |
| | | Int32 | A signed integer. |
| | | Atom | A PDF key (/XYZ). |
| | | String | A PDF string. |
| | | Color | An RGB color (array of 3 Fixed values) |
| | | BBox | A bounding box (array of 4 Fixed values) |
| Inherit | Choice | Optional | — Indicates if the property value can be inherited from a parent. |
| | | Inheritable | This property can be inherited. |
| | | Not-inherited | This property can not be inherited (Default). |
| Default | String | Optional | — The value to be used if the property is not found on this element (or through inheritance). This should be the same "Type" (Fixed, Int32, Atom, String) as the property. |
| Condition | Choice | Required | — Indicates whether the directives that are children of the Proc-var directive are to be executed. |
| | | Always | Always execute the children of this Proc-var directive. |
| | | Has-value | Execute the children of this Proc-var directive if a value is found on this node (either explicit or Default). |
| | | Diff-from-default-for-event | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by Default. |
| | | Diff-from-ancestor | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by searching the inheritance tree for any ancestor. |

| | | | |
|---------|--------|--|---|
| | | Diff-from-parent | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by examining the inheritance cache of the parent. |
| | | Diff-from-predecessor | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by examining the inheritance cache of the preceding peer. |
| | | Diff-from-value | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by Compare. (Can be used with any type) |
| | | Matches-value | Execute the children of this Proc-var directive if a value is found and that value matches that specified by Compare. (Can be used with any type) |
| | | Less-than-value | Execute the children of this Proc-var directive if a value is found and that value is less than that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | Less-equal-value | Execute the children of this Proc-var directive if a value is found and that value is less than or equal to that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | More-than-value | Execute the children of this Proc-var directive if a value is found and that value is greater than that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | More-equal-value | Execute the children of this Proc-var directive if a value is found and that value is greater than or equal to that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| Compare | String | Optional — The value used to determine Diff-from-value, Matches-value, Less-than-value, or More-than-value. This should be the same "Type" (Fixed, Int32, Atom, String) as the property. | |
| | | Default: | -none- |

3.14 Evaluate-var

Evaluate-var does exactly the same processing as Proc-var, except it does not make the data value available to the other Proc-* directives.

DTD content rule for this element:

```
(Comment | Conditional-delimiter | Emit-string | Conditional-prefix | Element-name |
Proc-string | Proc-integer | Proc-fixed | Proc-length | Proc-pixels | Proc-enum | Proc-var |
Walk-proplist | Call-proc-list | Proc-graphic-content | Proc-image-content | Proc-doc-text |
Walk-children | Walk-metadata | Emit-all-metadata | Walk-cached-property-sets |
Walk-structure | Walk-layout | Conditional-suffix )+
```


Table 13 *Attribute list for <Evaluate-var>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|-------------|---------------|---|
| Pdf-var | String | Required — The name of a property in a given property dictionary (see Owner) to be processed/evaluated. |
| Owner | Choice | Required — The owner of the property dictionary. |
| | Metadata | This is a pseudo-owner for entries in the document's metadata. |
| | Structelem | This is a pseudo-owner for properties specified directly in the StructElem's obj dictionary. |
| | Layout | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Layout. |
| | Link | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Link. |
| | List | Properties in the StructElem's Attribute dictionary list within the dictionary owned by List. |
| | Table | Properties in the StructElem's Attribute dictionary list within the dictionary owned by Table. |
| | Auto-span | This is a pseudo-owner generated by the SaveAs processor for each span it synthesizes by consolidating Tj operators having common styling properties (font, size, color, etc.) |
| Type | Inline-markup | This is a pseudo-owner generated by the SaveAs processor when the inline marking of /Span << ... >> BDC (abbrev.) Tj EMC is encountered. |
| | Choice | Required — The primary PDF datatype of the property (see Has-enum for a possible secondary datatype). |
| | Fixed | Fixed-point number. |
| | Int32 | A signed integer. |
| | Atom | A PDF key (/XYZ). |
| | String | A PDF string. |
| | Color | An RGB color (array of 3 Fixed values) |
| | BBox | A bounding box (array of 4 Fixed values) |
| Inherit | Choice | Optional — Indicates if the property value can be inherited from a parent. |
| | Inheritable | This property can be inherited. |
| | Not-inherited | This property can not be inherited (Default). |
| Default | String | Optional — The value to be used if the property is not found on this element (or through inheritance). This should be the same "Type" (Fixed, Int32, Atom, String) as the property. |
| Condition | Choice | Required — Indicates whether the directives that are children of the Proc-var directive are to be executed. |
| | Always | Always execute the children of this Proc-var directive. |

| | | | |
|---------|--------|--|---|
| | | Has-value | Execute the children of this Proc-var directive if a value is found on this node (either explicit or Default). |
| | | Diff-from-default-for-event | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by Default. |
| | | Diff-from-ancestor | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by searching the inheritance tree for any ancestor. |
| | | Diff-from-predecessor | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by examining the inheritance cache of the preceding peer. |
| | | Diff-from-value | Execute the children of this Proc-var directive if a value is found and that value differs from that specified by Compare. (Can be used with any type) |
| | | Matches-value | Execute the children of this Proc-var directive if a value is found and that value matches that specified by Compare. (Can be used with any type) |
| | | Less-than-value | Execute the children of this Proc-var directive if a value is found and that value is less than that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | Less-equal-value | Execute the children of this Proc-var directive if a value is found and that value is less than or equal to that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | More-than-value | Execute the children of this Proc-var directive if a value is found and that value is greater than that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| | | More-equal-value | Execute the children of this Proc-var directive if a value is found and that value is greater than or equal to that specified by Compare. (Can only be used with: Fixed, Int32, Atom, String) |
| Compare | String | Optional — The value used to determine Diff-from-value, Matches-value, Less-than-value, or More-than-value. This should be the same "Type" (Fixed, Int32, Atom, String) as the property. | |
| | | Default: | -none- |

3.15 Walk-proplist

Directs the SaveAs processor to walk the specified generic property list (property lists owned by XML, HTML-3.20, HTML-4.01). This is used to process arbitrary, user-supplied attributes on the current *Structural Element*.

DTD content rule for this element:

(Comment | Conditional-delimiter | Emit-string | Proc-property)+

Table 14 *Attribute list for <Walk-proplist>*

| Name | Type | Description |
|-------|--------|--|
| Owner | Choice | Required — Selects the attribute list owned by: Xml, Html-3.20, Html-4.01, Css-1.00, Css-2.00 |

3.16 Proc-property

Process an arbitrary property. Is similar in function to proc-var, except that it does not select/filter which properties are processed, it simply takes each property owned by the current owner in turn.

DTD content rule for this element:

(Comment | Conditional-delimiter | Emit-string | Property-name | Property-type)+

3.17 Property-type

Process the data portion of an arbitrary property.

DTD content rule for this element:

(Comment | Conditional-delimiter | Emit-string | Proc-string | Proc-integer | Proc-fixed | Proc-length | Proc-pixels | Proc-enum | Proc-doc-text | Proc-graphic-content | Proc-image-content)+

Table 15 *Attribute list for <Property-type>*

| Name | Type | Description |
|------|--------|---|
| Type | Choice | Required — The primary PDF datatype of the property. Fixed Fixed-point number. Int32 A signed integer. Atom A PDF key (/XYZ). String A PDF string. Color An RGB color (array of 3 Fixed values) BBox A bounding box (array of 4 Fixed values) |

3.18 Property-name

Process the name/key portion of an arbitrary property.

DTD content rule for this element:

Void?

3.19 Element-name

Output the Element-name (used in the XML output filter to generate the user-supplied element tag.

DTD content rule for this element:

Void?

Table 16 *Attribute list for <Element-name>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|-------------|----------------------|---|
| Node-type | Choice | Required — Used to select whether the <i>Structural Element</i> name from either the /S key of the StructElem (Structure-user-label) or the result of processing that key via the RoleMap (Structure-role) should be emitted. |
| | Structure-role | Use the result of processing the StructElem's /S key via the RoleMap. |
| | Structure-user-label | Use the StructElem's /S key. |

3.20 Conditional-delimiter

The text contained in this Mapping Table element will be emitted if this proc-var is not the first one to be accepted and processed after the start of an event or the first proc-var to be processed after a conditional-prefix.

DTD content rule for this element:

<TEXT>

Table 17 *Attribute list for <Conditional-delimiter>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|---------------------|-------------------|--|
| Emit-space-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | Emit-space-before | Emit a space before emitting any content text. |
| | No-space-before | Do not emit a space before emitting any content text. |
| Emit-space-after | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | Emit-space-after | Emit a space after emitting any content text. |
| | No-space-after | Do not emit a space after emitting any content text. |
| Emit-newline-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |

| | | | |
|--------------------|--------|--|---|
| Emit-newline-after | Choice | Emit-newline-before | Emit a newline before emitting any content text. |
| | | No-newline-before | Do not emit a newline before emitting any content text. |
| | | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. | |
| | | Emit-newline-after | Emit a newline after emitting any content text. |
| | | No-newline-after | Do not emit a newline after emitting any content text. |

3.21 Conditional-prefix

The text contained in this Mapping Table element will be cached and will be emitted if any proc-var is accepted to be processed before the end of the current event or the next Conditional-suffix control element.

DTD content rule for this element:

<TEXT>

Table 18 *Attribute list for <Conditional-prefix>*

| Name | Type | Description |
|---------------------|--------|--|
| Emit-space-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-space-before Emit a space before emitting any content text. |
| | | No-space-before Do not emit a space before emitting any content text. |
| Emit-space-after | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-space-after Emit a space after emitting any content text. |
| | | No-space-after Do not emit a space after emitting any content text. |
| Emit-newline-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-newline-before Emit a newline before emitting any content text. |
| | | No-newline-before Do not emit a newline before emitting any content text. |
| Emit-newline-after | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. |
| | | Emit-newline-after Emit a newline after emitting any content text. |
| | | No-newline-after Do not emit a newline after emitting any content text. |

3.22 Conditional-suffix

The text contained in this Mapping Table element will be emitted if the preceding Conditional-prefix within the current event was emitted.

DTD content rule for this element:

<TEXT>

Table 19 *Attribute list for <Conditional-suffix>*

| Name | Type | Description |
|---------------------|--------|---|
| Emit-space-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-space-before Emit a space before emitting any content text. No-space-before Do not emit a space before emitting any content text. |
| Emit-space-after | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-space-after Emit a space after emitting any content text. No-space-after Do not emit a space after emitting any content text. |
| Emit-newline-before | Choice | Required — Since XML strips the first/last space in each element and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-newline-before Emit a newline before emitting any content text. No-newline-before Do not emit a newline before emitting any content text. |
| Emit-newline-after | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-newline-after Emit a newline after emitting any content text. No-newline-after Do not emit a newline after emitting any content text. |

3.23 Comment

Does no processing. Provided to allow documentation or notes to be included in the Mapping Table.

DTD content rule for this element:

<TEXT>

3.24 Emit-string

The text contained in this Mapping Table element will be emitted.

DTD content rule for this element:

<TEXT>

Table 20 *Attribute list for <Emit-string>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|---------------------|-------------|---|
| Emit-space-before | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-space-before Emit a space before emitting any content text. No-space-before Do not emit a space before emitting any content text. |
| Emit-space-after | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-space-after Emit a space after emitting any content text. No-space-after Do not emit a space after emitting any content text. |
| Emit-newline-before | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-newline-before Emit a newline before emitting any content text. No-newline-before Do not emit a newline before emitting any content text. |
| Emit-newline-after | Choice | Required — Since XML strips the first/last space and most newlines from the parsed result, it is necessary to have this set of flags to control explicit insertion of these control codes. Emit-newline-after Emit a newline after emitting any content text. No-newline-after Do not emit a newline after emitting any content text. |

3.25 Proc-doc-text

Emit the text that is the content of the current Structural Element.

DTD content rule for this element:

Void?

Table 21 *Attribute list for <Proc-integer>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|--------------------|-------------|--|
| do-br-substitution | Choice | Required. do-br-substitution Emit a for every newline found in the doc text. do-xml-br-substitution Emit a for every newline found in the doc text. no-substitution Disregard newlines in doc text. |

3.26 Proc-string

If the data cached by the containing Proc-var directive is a string or an atom, the text content of the string or a text representation of the atom's name will be emitted.

DTD content rule for this element:

Void?

3.27 Proc-integer

If the data cached by the containing Proc-var directive is an Int32 or a Uns32, the text representation of the value will be emitted. This value will be scaled using the attributes of this directive:

1. The original value will be multiplied by the value of the Mul attribute.
2. The value of the Add attribute will be added to the result of step 1.
3. The result of step 2 will be divided by Div and the fraction will be discarded.
4. The result of step 3 will be converted to a string.

DTD content rule for this element:

Void?

Table 22 *Attribute list for <Proc-integer>*

| Name | Type | Description |
|------|--------|-------------|
| Mul | String | Optional |
| | | Default: 1 |
| Add | String | Optional |
| | | Default: 0 |
| Div | String | Optional |
| | | Default: 1 |

3.28 Proc-hex

If the data cached by the containing Proc-var directive is an Int32, an Uns32, or a Fixed, the text representation of the integer portion of the value after the scaling algorithm is applied will be emitted. This value will be scaled using the attributes of this directive:

1. The original value will be multiplied by the value of the Mul attribute.
2. The value of the Add attribute will be added to the result of step 1.
3. The result of step 2 will be divided by Div and the fraction will be discarded.
4. The result of step 3 will be converted to a string.

DTD content rule for this element:

Void?

Table 23 *Attribute list for <Proc-hex>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|-------------|-------------|--------------------|
| Mul | String | Optional |
| | | Default: 1 |
| Add | String | Optional |
| | | Default: 0 |
| Div | String | Optional |
| | | Default: 1 |
| Num-digits | String | Optional |
| | | Default: 2 |

3.29 Proc-fixed

If the data cached by the containing Proc-var directive is a FixedPoint number, the text representation of the value will be emitted. This value will be scaled using the attributes of this directive:

1. The original value will be multiplied by the value of the Mul attribute.
2. The value of the Add attribute will be added to the result of step 1.
3. The result of step 2 will be divided by Div.
4. The result of step 3 will be converted to a string. Frac-len controls the number of digits to the right of the decimal point. Frac-dlm is the fraction-radix character to be issued if Frac-len is greater than 0.

Proc-fixed, Proc-length, and Proc-pixels vary only in the default values for Mul, Div, and Add.

DTD content rule for this element:

Void?

Table 24 *Attribute list for <Proc-fixed>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|-------------|-------------|--------------------|
| Mul | String | Optional |
| | | Default: 1 |
| Add | String | Optional |
| | | Default: 0 |
| Div | String | Optional |
| | | Default: 1 |
| Frac-len | String | Optional |
| | | Default: 2 |
| Frac-dlm | String | Optional |

Default: .

3.30 Proc-length

If the data cached by the containing Proc-var directive is a FixedPoint number, the text representation of the value will be emitted. This value will be scaled using the attributes of this directive:

1. The original value will be multiplied by the value of the Mul attribute.
2. The value of the Add attribute will be added to the result of step 1.
3. The result of step 2 will be divided by Div.
4. The result of step 3 will be converted to a string. Frac-len controls the number of digits to the right of the decimal point. Frac-dlm is the fraction-radix character to be issued if Frac-len is greater than 0.

Proc-fixed, Proc-length, and Proc-pixels vary only in the default values for Mul, Div, and Add.

DTD content rule for this element:

Void?

Table 25 *Attribute list for <Proc-length>*

| <i>Name</i> | <i>Type</i> | <i>Description</i> |
|-------------|-------------|-------------------------|
| Mul | String | Optional Default: 72 |
| Add | String | Optional Default: 0 |
| Div | String | Optional Default: 72 |
| Frac-len | String | Optional Default: 2 |
| Frac-dlm | String | Optional Default: . |

3.31 Proc-pixels

If the data cached by the containing Proc-var directive is a FixedPoint number, the text representation of the value will be emitted. This value will be scaled using the attributes of this directive:

1. The original value will be multiplied by the value of the Mul attribute.
2. The value of the Add attribute will be added to the result of step 1.
3. The result of step 2 will be divided by Div.
4. The result of step 3 will be converted to a string. Frac-len controls the number of

digits to the right of the decimal point. Frac-dlm is the fraction-radix character to be issued if Frac-len is greater than 0.

Proc-fixed, Proc-length, and Proc-pixels vary only in the default values for Mul, Div, and Add.

DTD content rule for this element:

Void?

Table 26 *Attribute list for <Proc-pixels>*

| Name | Type | Description |
|----------|--------|-------------------------|
| Mul | String | Optional Default: 96 |
| Add | String | Optional Default: 36 |
| Div | String | Optional Default: 72 |
| Frac-len | String | Optional Default: 0 |
| Frac-dlm | String | Optional Default: . |

3.32 Proc-enum

If the data cached by the containing Proc-var directive is a string or an atom, then the proc-enum choice elements which are children of this control element are searched for a match. If a match is found, the Value-out value of the matching Proc-enum-choice directive is issued as a string.

DTD content rule for this element:

Proc-enum-choice+

3.33 Proc-enum-choice

DTD content rule for this element:

Void?

Table 27 *Attribute list for <Proc-enum-choice>*

| Name | Type | Description |
|-----------|--------|---|
| Value-in | String | Required — This value is compared to the value cached by the containing proc-var directive. |
| Value-out | String | Required — This value will be emitted as a string if a match against Value-in is found. |

3.34 Proc-graphic-content

Process the content of the current structural element as a vector graphic.

DTD content rule for this element:

Void?

3.35 Proc-image-content

Process the content of the current structural element as a bitmapped graphic.

DTD content rule for this element:

Void?

3.36 Void

The "Void" node is used to avoid the <empty/> syntax of XML and force the <name></name> syntax of SGML (this allows editing on any SGML editor as well as any XML editor).

Many of the above elements have the content rule "Void?". *However, the Void element should never be specified, thereby leaving the containing node empty.*

DTD content rule for this element:

<EMPTY>

4 Editing Mapping Tables

You can edit the _____.xml versions of the Mapping Tables in any XML (or SGML) editor. The files were created using FrameMaker+SGML 6.0, and detailed instructions for that editor are included below, followed by general instructions for using another editor.

4.1 Editing Mapping Tables in FrameMaker+SGML 6.0.

To edit/modify the Mapping Tables in FrameMaker+SGML 6.0:

- Copy all the files in the SaveAsXML/DeveloperInfo folder and all the mapping tables from the SaveAsXML/MappingTables folder to a single folder on your machine.
Be sure to include:
 - sgmlapps.fm,
 - sgml.dec,
 - MappingTable.edd.fm,
 - MappingTable.dtd,
 - MappingTable.fm and
 - the _____.xml (or _____.fm) file for the conversion you wish to edit.
- Open the sgmlapps.fm file:
 - Select the menu command:
File => Developer Tools => Reread SGML Application File
 - Close this file but *do not exit* FrameMaker, the reread sgmlapps file remains valid only for the current session.
- Choose the SGML Application
 - Select the menu command:
File => Set SGML Application
 - Choose **Mapping Table** from the pulldown in the dialog.
 - Select **Set** to close the dialog.
- Open the _____.xml version of the Mapping Table file you wish to edit (note that FrameMaker will change the file extension to _____.fm, do not change it back to _____.xml, see instructions later in this section):
 - Make the necessary changes.
 - Select the menu command:
Element => Validate then click "**Start Validating**"
Correct what is necessary until you get a "Document is valid" response.
- Select the menu command:
File => Save to save the _____.fm version of the file.
- Select the menu command:
File => SaveAs
 - In the save as type field, choose **SGML** (Note: Do *NOT* choose XML)
 - **BE SURE TO CHANGE THE FILE SUFFIX TO .xml**

- Click **"Save"**
- Copy the _____.xml file to the **Plug-Ins/SaveAsXML/MappingTables** folder.

4.2 Guidelines for Editors other than FrameMaker+SGML 6.0

The DTD is included in the DeveloperInfo folder is an SGML syntax DTD. To convert it to XML syntax, remove the " - -" (space-hyphen-space-hyphen, which indicates starttag and endtag are required) from each ELEMENT directive.

It may be necessary to modify the file path:

"D:\Adobe Docs\AcroStructure\MappingTables\MappingTable.dtd"

in the DOCTYPE directive to point to your local copy of the DTD.

Note: The SaveAsXML processor requires that the Mapping Tables must be valid in accordance with this DTD or Acrobat may crash during the SaveAs operation.

---End of document---