Standard Element Types

Technical Note #5401

ADOBE SYSTEMS INCORPORATED

Corporate Headquarters

345 Park Avenue

San Jose, CA 95110-2704

(408) 536-6000

October 10, 2000

Date/Rev #	Comments
Feb., 2000	Written from information supplied by Dan Rabin and Bennet Leeds in October, 1999.

Standard Element Types

1. Introduction

Many elements of your applications can be used with various other applications, if you have the foresight to define them in ways that other applications can understand. If you record the structural elements of your application, other applications can use your programs in ways that you may not have thought of when you were designing them. For instance, other, more general applications might inspect and otherwise use PDF documents that your application created. Conversely, if you are creating applications that use structured PDF documents, you can use the structural information that other developers have recorded for you.

Adobe has defined a set of standard element types to facilitate information reuse among applications, and to make such general programs as useful as possible. These standard element types cover very common purposes. By relating your native element types to the Adobe standard set, you can be assured that the PDF documents your software creates can be used by other tools. Your applicatons can search other applications' PDF documents for element types in the standard set as well.

2. Using Standard Element Types

This section describes both how to create an application using Adobe's Standard Element Types, and also how your applications can make use of other applications that have related their types to the standard set.

2.1 Creating structure in your applications

If your application already uses an element type with the same name and meaning as one of the standard element types, your job is done. Other applications that access yours will search for the standard element types and apply the standard meaning.

However, if you have an element type that has approximately the same meaning, but not exactly the same name, as one of the standard types, you need to help other applications recognize this. Adobe PDF 1.3 provides a *role map* that relates your element type to the standard one of the same meaning. Other applications that know about standard types will consult the role map to determine the meaning of your unknown element types.

If your application creates PDF documents with logical structure, you can relate your element types to the standard set by creating appropriate entries in the role map of the structure tree root of the created documents.

Some sets of element types, such as HTML, contain element types, such as HTML tables, that you could manipulate into representing layout information instead of logical strucutre. This is not a good idea. To preserve your element structure, you should develop a parallel set of element types to represent the non-logical uses of the element types. You can then use the role map to relate your types to the special role `NonStruct`.

NOTE: *Only use this strategy if you have no choice of element types. If you are devising new sets of element types, avoid incorporating non-structural types.*

If your application uses an element type that has the same name as one of the standard element types, but has a completely differernt meaning, you should use the role map to relate your element types to the special standard element type `Private`.

2.2 Using Structure In Other Applications' PDF Documents

When designing your applications to use standard element types in structured PDF documents, there are three issues to keep in mind:

- Documents can contain elements that are not related to (via the role map) any standard element types. Your application should ignore unknown element types.
- Some documents may use element types that thave no logical structure associated with them. You can see this by checking the role map for element types related to the special role `NonStruct`.
- Some documents may use element types that have standard names, but their meaning might be different. You can see this by checking the role map for element types related to the special role `Private`.

If your application will generically use logically structured PDF documents, then it should be lenient about interpreting the structure. Use the role map as a guide rather than as a perspective schema.

If your application performs a lot of navigation, you will be most interested in subdivisions of structured PDF documents, such as articles and sections. You will also be interested in links. If your application reuses a lot of data from structured PDF documents, you will likely use most of the features of the standard element type set to identify interesting information.

3. Standard Element Features

The standard element types were have these features:

- Standard element types are *generally applicable*: they occur in many documents.
- Standard element types are *flexible*, applying in different contexts without significant distortion in meaning.
- The set of standard element types are *economical*: two different element types should not do the work of one.
- Standard element types reflect *logical* structure rather than presentational artifacts.

4. Standard Element Type Definition

This section defines all of the standard element types. Each subsection presents a group of element types. The definition of each element type includes both its intended meaning and its intended containment relationship with elements of other types.

You should use the standard element types definitions leniently: applications that use logical structure should accept reasonable deviations from the containment hierarchies specified here.

4.1 Standard attribute objects

Some of the standard element types are defined with attributes. To set values for these attributes, you provide an attribute object with owner `Standard` on an element whose element type relates (via the role map) to the given standard element type.

4.2 Standard attributes for links

The parent-child connection between structural elements is not always appropriate for modeling a given relationship, and it is not always available because it is typically used to model logical containment of content. To allow linking of structural elements independently of the containment hierarchy, the standard element types also include a standardized way to attach attributes serving as links to any structural element. An attribute object with owner Link may contain a key SE (StructElem) whose associated value is the ID value of another structural element. The key S (Subtype) in such an attribute object distinguishes among the various semantics that may be associated with a link. There are two such subtypes, L (Link) and R (Reference).

NOTE: *A preliminary public version of this document specified that* Link *attributes should indicate their targets via an indirect reference to the target element as a Cos object. This mechanism was actually difficult for applications to create, though, and is no longer recommended.*

L (Link)

A Link represents a hypertext navigational aid. A Link offers a way of getting to its target, but does not indicate that the target is in any other way related to the source.

A Link target element may contain a PDF Link Annotation object that specifies the traversal action to be taken by a PDF viewer. Link Annotations can be included as children of structural elements by using an OBJR object as an intermediate.

R (Reference)

Unlike Links, References indicate that their targets are semantically referred to at the source location. Reference elements are appropriate for references to footnotes or bibliography entries from body text, as well as from index and table-of-contents entries to appropriate locations. You can use References in your applications to perform the same navigational behavior as Links.

4.3 The Standard Element Types

These standard element types are intended to contain linear text, perhaps organized into a hierarchy of sections. Document components, such as figures, that do not form part of the linear text are treated in Section 4.2 of this document. Linear text refers to such elements via a `Reference` element.

Art (Article)

An Article is a relatively self-contained body of text considered to be a single narrative or exposition. An Article may contain a Heading, Sections, Paragraphs, and other content considered to be in the logical sequence of the article. An Article should not contain any other Article, so that applications that examine logical structure may assume that Articles are disjoint.

Part

A Part is a large division of an entire document. Parts are appropriate for grouping Articles; a Part may contain another Part.

H (Heading)

A Heading is a label for a subdivision of a document's content. A Heading should appear as the first child of the division that it heads.

Sect (Section)

A Section is the most general text container type, comparable to DIV in HTML. A Section may contain a Heading as well as other Sections.

P (Paragraph)

A Paragraph should be a low-level division of text: a Paragraph should not contain any other Paragraphs, although it may contain in-line objects.

NOTE: *A preliminary version of this document required that* P *elements should not contain any other* P *element. This requirement has been relaxed.*

Span

A Span is any segment of text. Spans are independent of paragraphs, sentences, and so forth. You typically use a Span as the text associated with a Link or Reference.

L (List)

A List can be any sequence of items of like meaning and importance. The immediate children of a List element should be ListItem elements.

LI (ListItem)

A ListItem is one member of a List. It may contain a ListItemLabel.

Lbl (Label)

A Label is a name that distinguishes an element from others in the same list or other group of like items.

Quote

A Quote is a portion of text attributed to someone other than the author of the text surrounding it.

Formula

A Formula is a mathematical formula. [This is only really useful for navigation: we'd need to represent encoding in order to be able to reuse a formula].

Code

Code represents part of computer program text embedded within a document.

Quote, Formula, and Code have in common that, although they form part of a text sequence, they are not themselves paragraphs. They may, however, be contained in or contain paragraphs.

Caption

A Caption is a brief portion of text that describes a Table or Figure. It is different from a Label in that it describes rather than merely identifying. Items that have Captions may also have Labels.

Table

A Table represents a table of data, possibly having a complex substructure. A Table contains TableRows as children, and may have a Caption as a child. The substructure of element types that may be included within a Table mimics the highly prevalent structure of HTML tables.

TR (TableRow)

A TableRow is one row of headings or data in a table. A TableRow may contain TableHeaderCells and TableDataCells.

TH (TableHeaderCell)

A TableHeaderCell contains text describing one or more rows or columns of a table. A TableHeaderCell can have a standard attribute object with owner Table and attributes RowSpan and ColSpan that describe the space taken up by the cell within the table. If not given, the value of each of these attributes is assumed to be 1.

TD (TableDataCell)

A TableDataCell is like a TableHeaderCell, except that it should contain data. Like a TableHeaderCell, a TableDataCell can have RowSpan and a ColumnSpan attributes.

The association of headers with rows and columns of data is to be determined heuristically by applications and is not defined here.

Figure

A Figure is graphic material associated with text but not in a fixed place with respect to the text. Text can contain a Reference to a figure. Figures may contain Groups.

Group

A Group is a portion of an illustration. It may correspond to a subfigure, layer, or other portion of a graphic that is meaningful to extract or examine on its own.

Note

A Note is some explanatory text, such as a footnote or an endnote, that is referred to from body text. A Note may have a Label as a child element. You can include a Note as a child in the body text element that refers to it, or elsewhere (such as in an endnotes section) and access it via a Reference.

BibEntry

A BibEntry gives information on where some cited information can be found. A BibEntry may contain a Label as a child element. It is likely that a BibEntry will have substructure identifying author, work, publisher, and so forth; but standard element types are not provided at this level of detail.

Index

An Index contains a sequence of entries that contain both identifying text and Reference elements that point out the occurrence of the text in the main text body of a document. An Index contains a List whose items are Links to occurrences of the subject described in their text.

TOC (TableOfContents)

A TableOfContents contains a List of ListItems, each of which may contain either a ContentItem or another such List along with a Reference to a text body (or other) element. A

TableOfContents is thus potentially hierarchical. You should create such a hierarchy to structurally match the hierarchy of the text body of the document.

You can treat lists of figures and tables, as well as bibliographies, as TablesOfContents for the purpose of standard element types.

TOCI (ContentItem)

A ContentItem contains some identifying text, along with a Reference to a text body element.

Form

A Form element designates a PDF Form Annotation—something that can be or has been filled out. A Form element refers to its underlying annotation via an object reference (OBJR).