XML Support in XMF

Tony Clark

May 12, 2004

1 Introduction

XML is a standard data format for storage and application communication. XMF provides extensive support for XML: models can be written and read using the XMF XML format; XML parsers can be constructed that read XML input and synthesize XMF values; XMF models can be annotated with mappings describing how to export instances of the models in XML format; XMF contains a model of XML allowing the features of XMF to be used to construct, manipulate and transform XML documents.

2 XML Model

This section defines the classes defined in the package `XML`.

2.1 `Document`

Represents an XML document containing single root element node. A document is an XMF resource and therefore has a resource name string that describes the location of the document (empty if unknown). The following attributes are defined:

`resourceName:String`

Defines the name of the resource containing the document.

`root:Node`

The root node of the document.

The following operations are defined:

`Document(resourceName:String)`

Constructs an instance.

`Document(resourceName:String,root:Node)`

Constructs an instance.

`print()`

Prints the XML document to its resource. An error occurs if no resource is set.

`print(out:OutputChannel)`

Prints the XML document to the supplied output channel.

`pprint(out:OutputChannel)`

Prints the XML document to the supplied output channel using indentation to show nesting.

`stripWhiteSpace():Document`

Removes any occurrences of `Text` elements in the document that contain just white-pace characters. Returns a fresh copy of the document containing no white-space.

`reduce():Element`

Translates an XML document that conforms to the XMF.dtd (see appendix A) to the appropriate XMF element.

2.2 `Node`

An abstract class that is used as the super class of all document nodes.

2.3 `Element`

A sub-class of `Node` that has a tag, a set of attributes and a sequence of child nodes. The following attributes are defined:

`tag:String`

The tag of the element.

`attributes:Set(Attribute)`

The attributes for the element.

`children:Seq(Node)`

The child nodes of the element.

The following operations are defined for elements:

`filter(tag:String):Seq(Element)`

Returns the sequence of children with the given tag in the order that they occur in the element.

`getAtt(name:String):String`

Returns the value of the attribute with the given name. Throws an exception if no attribute with the given name exists.

`hasAtt(name:String):Boolean`

Tests whether the element has an attribute with the given name.

`print(out:OutputChannel)`

Prints the element to the supplied output channel.

`put(name:String,value:String)`

Sets the value of the attribute with the given name. Throws an exception if no attribute with the given name exists.

2.4 `Attribute`

An XML attribute that consists of a name and a value. Both the name and value are strings.

2.5 `Text`

A sub-class of `Node` that represents text. An instance of this class has a single attribute `text` of type string.

3 XML Input and Output

4 Parsing XML Files

5 XML Mappings

A The XMF DTD

```
<!DEFINE Value
  (Boolean     |
   Integer     |
   String      |
   Object      |
   IdRef       |
   Set         |
   Seq         |
   Null        |
   Operation   |
   EmptySeq)>

<!ELEMENT Boolean ()>
<!ATTLIST Boolean value PCDATA #REQUIRED>

<!ELEMENT EmptySeq ()>

<!ELEMENT IdRef ()>
<!ATTLIST IdRef id ID #REQUIRED>

<!ELEMENT Integer ()>
<!ATTLIST Integer value PCDATA #REQUIRED>

<!ELEMENT NameSpaceRef ()>
<!ATTLIST NameSpaceRef name PCDATA #REQUIRED>

<!ELEMENT Null ()>

<!ELEMENT Object (Ref Slot*)>
<!ATTLIST Object id ID #REQUIRED>

<!ELEMENT Operation ()>
<!ATTLIST Operation name PCDATA #REQUIRED>

<!ELEMENT Ref (NameSpaceRef*)>
<!ATTLIST Ref root PCDATA #REQUIRED>

<!ELEMENT Set (Value)*>

<!ELEMENT Seq (Value Value)>

<!ELEMENT Slot (Value)>
<!ATTLIST Slot name PCDATA #REQUIRED>

<!ELEMENT String ()>
<!ATTLIST String value PCDATA #REQUIRED>
```