Data-centric XML

Namespaces

Overview and Rationale

- Single XML documents used for multiple applications
  - e.g. documents containing both MathML and SVG
  - A single, centralized DTD cannot readily be produced
  - Applications may overlap in their usage of elements and attributes
- Solution: Provide a mechanism to put elements and attributes into multiple “namespaces”
- Uniqueness of names achieved by denoting namespaces by URI reference
  - Each application author should allocate names from “her” URL space
  - Duplicating URLs to scope element names is tedious, so an abbreviation for namespaces is necessary
    - XML names cannot contain /, %, ~, anyway
  - Namespaces in XML 1.1: IRIs, not URIs
- http://www.w3.org/TR/REC-xml-names/

Syntax Overview

- Extended Name syntax:
  - colon gets special meaning: separator for namespace prefix and local part
  - qualified name: prefix:local_part
  - “xml” is reserved as a prefix (e.g. xml:lang, xml:space)
- Namespace prefixes must be declared
  - declaration of individual prefixes as xmlns:prefix=“URI”
  - declaration of default namespace: xmlns=“URI”

Example

```
<rdf:RDF xmlns:rdf="http://www.w3.org/TR/REC-rdf-syntax">
 <rdf:Description
        xmlns:dc="http://purl.org/dc"
        about="http://www.cafeconleche.org/examples/
 impressionists.xml">
 <dc:title>Impressionist Paintings</dc:title>
 <dc:creator>Elliot Rusty Harold</dc:creator>
 <dc:description>
        A list of famous impressionist paintings organized by painter and
 date
 </dc:description>
 <dc:date>2000-08-23</dc:date>
 </rdf:Description>
</rdf:RDF>
```

Declaring Namespaces

[1] NSAttName ::= PrefixedAttName | DefaultAttName

- Attributes with NSAttNames are reserved for namespace declaration
- Namespace declarations nest

[2] PrefixedAttName ::= 'xmlns:' NCName

- Namespace Constraint (NSC): NCNames starting with “x”, “m”, “l” (any case) are reserved
- Attribute value must not be empty
  - Namespaces in XML 1.1: Empty value unbinds prefix

[3] DefaultAttName ::= 'xmlns'

- Non-empty attribute value declares default namespace
- Empty attribute removes declaration of default namespace

[4] NCName ::= (Letter | '_') (NCNameChar)*

[5] NCNameChar ::= Letter | Digit | '.' | '-' | '_' | CombiningChar | Extender

- Declarations of namespace attributes may occur in DTD

Qualified Names

[6] QName ::= (Prefix ':')? LocalPart

[7] Prefix ::= NCName

[8] LocalPart ::= NCName

- The Prefix must have been declared
  - either by namespace declaration at the current element, or a parent element

Using Qualified Names

- Namespace-conforming documents must use QNames as element names:

  [9] STag ::= '<' QName (S Attribute)* S? '>'

  [10] ETag ::= '</' QName S? '>'

  [11] EmptyElemTag ::= '<' QName (S Attribute)* S? '/>'

  [12] Attribute ::= NSAttName Eq AttValue
  | QName Eq AttValue

- NSC: Prefixes in QNames must have been declared

Using Qualified Names (2)

[13] doctypedecl ::= '<!DOCTYPE' S QName (S ExternalID)? S? ('[' (markupdecl | PEReference | S)* ']' S?)? '>'

[14] elementdecl ::= '<!ELEMENT' S QName S contentspec S? '>'

[15] cp ::= (QName | choice | seq) ('?' | '*' | '+')?

[16] Mixed ::= '(' S? '#PCDATA' (S? '|' S? QName)* S? ')*' | '(' S? '#PCDATA' S? ')'

[17] AttlistDecl ::= '<!ATTLIST' S QName AttDef* S? '>'

[18] AttDef ::= S (QName | NSAttName) S AttType S DefaultDecl

Namespace Scoping

- Namespace declaration applies to element where it is declared, and all child elements unless overridden
- Multiple namespaces can be declared in a single start tag

Namespace Defaulting

- Applies to element and all content elements that have no prefix
- Does not **apply** to attributes
- Can be overridden in nested elements
- Can be removed by declaring xmlns=“”
- Declaration in DTD is possible:

```
<!ATTLIST html xmlns CDATA #FIXED "http://www.w3.org/1999/xhtml">
```

Conformance

- For a NS-conforming document:
  - All attribute and element names match QName
  - All namespace constraints (NSC) must hold
  - All other Names must match NCName (e.g. PI targets, entity names, …)
  - no two attributes of an element may have the same namespace and local name
    - I.e. they must not have prefixes bound to the same namespace

Namespaces in Applications

- Namespace-aware and namespace-unaware parsers
- Namespace-aware parser keep track of declarations, and report namespaces to application
  - Specific APIs for reporting namespace, prefix, localname
- Usage of namespaces in valid documents is difficult:
  - not only namespace must be correct, but also the namespace prefix
  - Solution: parameter entities allow customization of element names

```
<!ENTITY % dc-prefix “dc”>
<!ENTITY % dc-colon “:”>
<!ENTITY % dc-title “%dc-prefix;%dc-colon;title”>
<!ELEMENT %dc-title; (#PCDATA)>
```