DITA 1.3 proposed feature #13118

Contents

DITA 1.3 proposed feature #13118

Provide a domain that provides the ruby markup from HTML5 as used by Japanese and other ideographic languages.

Date and version information

- Proposal Completed: 1 Oct 2012
- Change log:
  - 21 Oct 2012:
    - Completed TBD in costs
    - Added RNC and XSD declarations
    - Removed <rb> element—has been obsoleted in latest HTML5 draft (October 2012)
    - Added statement about which package to include in.
  - 22 Oct 2012:
    - Allow <ruby> to nest per 10/2012 version of HTML5 specification.
- Champion: Eliot Kimber,
- Email discussion: *https://lists.oasis-open.org/archives/dita/201201/msg00061.html*.

Original requirement

In Japanese, the pronunciation of ideographic characters cannot always (or often) be known from context. The ideographic characters are annotated with their phonetic transliteration, a "ruby", which is rendered above or beside or following the ideographs. This is standard Japanese typography.

Use cases

Japanese and other ideographic or similar languages where correct pronunciation may require phonetic annotation.

Benefits

- Who benefits: DITA users who create Japanese- and similar ideographic language content.
- Expected benefit: Enables correct typographic rendering of ruby annotations.
- Potential users: At a minimum, all or most companies that write in or localize to Japanese.
- Degree of positive impact: Significant. Makes producing Japanese documents with proper ruby annotation possible without the need to implement a custom vocabulary module.

Costs

Costs:

- Maintainers of the DTDs and XSDs: Adds a new vocabulary module, which must be integrated into the appropriate shell document types.
- Editors of the DITA specification:
  - How many new topics will be required? 1 new reference topic.
  - How many existing topics will need to be edited?
    - Topic "Domain elements" in the Language Reference will need to reflect this domain.
  - Will the feature require substantial changes to the information architecture of the DITA specification? No architectural change.
- Vendors of tools: Processors that render DITA content visually should provide appropriate rendering of ruby content. For HTML-based outputs, ruby support is built in to most, if not all, browsers.

- DITA community-at-large. Will this feature add to the perception that DITA is becoming too complex? Will it be simple for end users to understand?

  This feature adds a new optional vocabulary module. Users who need it will appreciate having it readily available. Users who do not need it may safely ignore it. The general architecture and semantics of DITA are not affected by this proposal.

Technical requirements

Define a new vocabulary module, rubyDomain, that defines the <ruby> element type with the same content as defined in HTML5.

ruby	Contains the subelements <rp> and <rt> or nested <ruby>. See the HTML5 definition of the <ruby> element.

This module should be included in the base vocabulary package as it is applicable to all documents that use languages for which ruby annotations are needed.

rubyDomain.ent:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- ============================================================
     DITA Ruby Domain

     Defines equivalent of HTML ruby elements for marking up
     Japanese language documents.

     Copyright (c) 2012 OASIS Open

     ============================================================ -->


<!-- ============================================================ -->
<!--                    Ruby DOMAIN ENTITIES                      -->
<!-- ============================================================ -->

<!ENTITY % ruby-d-ph
   " ruby
   "
>



<!ENTITY   ruby-d-att
  "(topic ruby-d)"
>


<!-- ================== End DITA For Publishers Ruby Domain Entities
 ============ -->
```

rubyDomain.mod:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- ============================================================
     DITA Ruby Domain

     Defines equivalent of HTML ruby elements for marking up
     Japanese language documents.

     Copyright (c) 2012 OASIS Open
```

```
     ============================================================= -->

 <!ENTITY % ruby           "ruby" >

 <!ENTITY % rp             "rp" >
 <!ENTITY % rt             "rt" >


<!-- ============================================================= -->
<!--                    ELEMENT NAME ENTITIES                      -->
<!-- ============================================================= -->


<!-- ============================================================= -->
<!--                     ELEMENT DECLARATIONS                      -->
<!-- ============================================================= -->


<!-- In order to support HTML5, which allows a mix of PCDATA, other phrase-
     level elements, and <rt> and <rp>, the content model must allow
     %ph;, which means that the DTD allows <ruby> within <ruby>. However,
     <ruby> should *not* be used within <ruby>, per the HTML
     constraints on <ruby>. Likewise, if <rp> is used, it should be
     used as <rp>(</rp><rt>...</rt><rp>)</rp> per the HTML5 spec.
  -->
<!ENTITY % ruby.content
"
  (#PCDATA |
   %ruby; |
   %text; |
   %rp; |
   %rt;)*
  "
>
<!ENTITY % ruby.attributes
 "
   %id-atts;
  %localization-atts;
  base
    CDATA
    #IMPLIED
  %base-attribute-extensions;
  outputclass
    CDATA
    'ruby'
 "
>
<!ELEMENT ruby %ruby.content; >
<!ATTLIST ruby %ruby.attributes; >


<!-- LONG NAME: Ruby parenthesis -->
<!ENTITY % rp.content
"
  (#PCDATA
  )*
  "
>
<!ENTITY % rp.attributes
 "
   %id-atts;
  %localization-atts;
  base
```

```
    CDATA
    #IMPLIED
  %base-attribute-extensions;
  outputclass
    CDATA
    'rp'
 "
>
<!ELEMENT rp %rp.content; >
<!ATTLIST rp %rp.attributes; >

<!-- LONG NAME: Ruby Text -->

<!ENTITY % rt.content
"
  (#PCDATA |
   %text;
  )*
  "
>
<!ENTITY % rt.attributes
 "
   %id-atts;
  %localization-atts;
  base
    CDATA
    #IMPLIED
  %base-attribute-extensions;
  outputclass
    CDATA
    'rt'
 "
>
<!ELEMENT rt %rt.content; >
<!ATTLIST rt %rt.attributes; >



<!-- ============================================================= -->
<!--                    SPECIALIZATION ATTRIBUTE DECLARATIONS      -->
<!-- ============================================================= -->

<!ATTLIST ruby             %global-atts;  class CDATA "+ topic/ph    ruby-
d/ruby ">
<!ATTLIST rp               %global-atts;  class CDATA "+ topic/ph    ruby-
d/rp ">
<!ATTLIST rt               %global-atts;  class CDATA "+ topic/ph    ruby-
d/rt ">


<!-- ================== End Ruby Domain ======================= -->
```

Figure 1: DTD Syntax domain module declarations

rubyDomainMod.xsd:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <!-- DITA 1.3 XML Domain -->
```

```
    <xs:annotation>
      <xs:appinfo>
        <dita:domainsModule xmlns:dita="http://dita.oasis-open.org/
architecture/2005/">(topic ruby-d)</dita:domainsModule>
      </xs:appinfo>
      <xs:documentation>

      </xs:documentation>
    </xs:annotation>


  <xs:group name="ruby-d-ph">
    <xs:choice>
      <xs:element ref="ruby"/>
    </xs:choice>
  </xs:group>

  <xs:group name="ruby.content">
    <xs:sequence>
      <xs:choice>
        <xs:element ref="text" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="ruby" minOccurs="0" maxOccurs="1"/>
      </xs:choice>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="rp"/>
        <xs:element ref="rt"/>
      </xs:choice>
    </xs:sequence>
  </xs:group>

  <!-- Basic form: ruby -->
  <xs:element name="ruby">
    <xs:annotation>
      <xs:documentation>

      </xs:documentation>
    </xs:annotation>
    <xs:complexType mixed="true">
      <xs:complexContent>
        <xs:extension base="ruby.class">
          <xs:attribute ref="class" default="+ topic/ph ruby-d/ruby "/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="ruby.class" mixed="true">
        <xs:sequence>
          <xs:group ref="ruby.content"/>
        </xs:sequence>
    <xs:attributeGroup ref="ruby.attributes"/>
  </xs:complexType>

  <xs:attributeGroup name="ruby.attributes">
    <xs:attributeGroup ref="global-atts"/>
    <xs:attributeGroup ref="univ-atts"/>
    <xs:attribute name="outputclass" type="xs:string"/>
  </xs:attributeGroup>


  <xs:element name="rp">
    <xs:annotation>
      <xs:documentation>
        <p>Ruby parenthesis</p>
      </xs:documentation>
```

```
    </xs:annotation>
    <xs:complexType mixed="true">
      <xs:complexContent>
        <xs:extension base="rp.class">
          <xs:attribute ref="class" default="+ topic/ph ruby-d/rp "/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="rp.class" mixed="true">
    <xs:attributeGroup ref="rp.attributes"/>
  </xs:complexType>

  <xs:attributeGroup name="rp.attributes">
    <xs:attributeGroup ref="global-atts"/>
    <xs:attributeGroup ref="univ-atts"/>
    <xs:attribute name="outputclass" type="xs:string"/>
  </xs:attributeGroup>

  <xs:group name="rt.content">
    <xs:choice>
      <xs:element ref="text" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
  </xs:group>

  <xs:element name="rt">
    <xs:annotation>
      <xs:documentation>
        <p>Ruby text</p>
      </xs:documentation>
    </xs:annotation>
    <xs:complexType mixed="true">
      <xs:complexContent>
        <xs:extension base="rt.class">
          <xs:attribute ref="class" default="+ topic/ph ruby-d/rt "/>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="rt.class" mixed="true">
    <xs:sequence>
      <xs:group ref="rt.content"/>
    </xs:sequence>
    <xs:attributeGroup ref="rt.attributes"/>
  </xs:complexType>

  <xs:attributeGroup name="rt.attributes">
    <xs:attributeGroup ref="global-atts"/>
    <xs:attributeGroup ref="univ-atts"/>
    <xs:attribute name="outputclass" type="xs:string"/>
  </xs:attributeGroup>

</xs:schema>
```

Figure 2: XSD domain module declarations

rubyDomainMod.rnc:

```
datatypes xsd = "http://www.w3.org/2001/XMLSchema-datatypes"

# ================================================================
# MODULE:    DITA Ruby Domain - RNC
```

```
# VERSION:   1.3
# DATE:      October 2012
# ============================================================


namespace a = "http://relaxng.org/ns/compatibility/annotations/1.0"

# Define the domain values of this module
domains-atts-value |= "(topic ruby-d)"
# Define domain extension patterns
ruby-d-ph =
   ruby.element

# Extend the patterns with the domain contribution
ph |= ruby-d-ph
# Define elements content and attributes

ruby = ruby.element
rp = rp.element
rt = rt.element

# LONG NAME: Ruby annotation
ruby.content =
  (((text |
     \text) |
    ruby),
    (rp |
     rt)*)
ruby.attributes =
  univ-atts,
  attribute outputclass { text }?
ruby.element =

  element ruby {
    ruby.attlist,
    ruby.content
   }
ruby.attlist &= ruby.attributes

# LONG NAME: Ruby parenthesis
rp.content =
  (text)*
rp.attributes =
  univ-atts,
  attribute outputclass { text }?
rp.element =

  element rp {
    rp.attlist,
    rp.content
   }
rp.attlist &= rp.attributes

# LONG NAME: Ruby text
rt.content =
  (text |
   \text)*
rt.attributes =
  univ-atts,
  attribute outputclass { text }?
rt.element =

  element rt {
    rt.attlist,
```

```
    rt.content
   }
rt.attlist &= rt.attributes



# Specialization attributes. Global attributes and class defaults
ruby.attlist &=
  global-atts,
  [ a:defaultValue = "+ topic/ph ruby-d/ruby " ] attribute class { text }?
rp.attlist &=
  global-atts,
  [ a:defaultValue = "+ topic/ph ruby-d/rp " ] attribute class { text }?
rt.attlist &=
  global-atts,
  [ a:defaultValue = "+ topic/ph ruby-d/rt " ] attribute class { text }?
```

Figure 3: RelaxNG Compact domain module declarations

Examples

```
<topic id="ruby-test-topic-01" xml:lang="jp-JP">
  <title>Ruby Domain Test (DTD)</title>
  <body>
    <p> Ruby with bare PCDATA first child: <ruby>
      ##
      <rp>#</rp>
      <rt>####</rt>
      <rp>#</rp>
    </ruby>##########################</p>
    <p> Ruby with &lt;text&gt; element first child: <ruby>
      <text>##</text>
      <rp>#</rp>
      <rt>####</rt>
      <rp>)</rp>
    </ruby>##########################</p>
    <p>Ruby with nested &lt;ruby&gt; element:<ruby>
      <ruby>##<rt>2342</rt></ruby>
      <rp>#</rp>
      <rt><text>####</text></rt>
      <rp>#</rp>
    </ruby></p>
  </body>
</topic>
```

Figure 4: Sample topic with ruby markup.

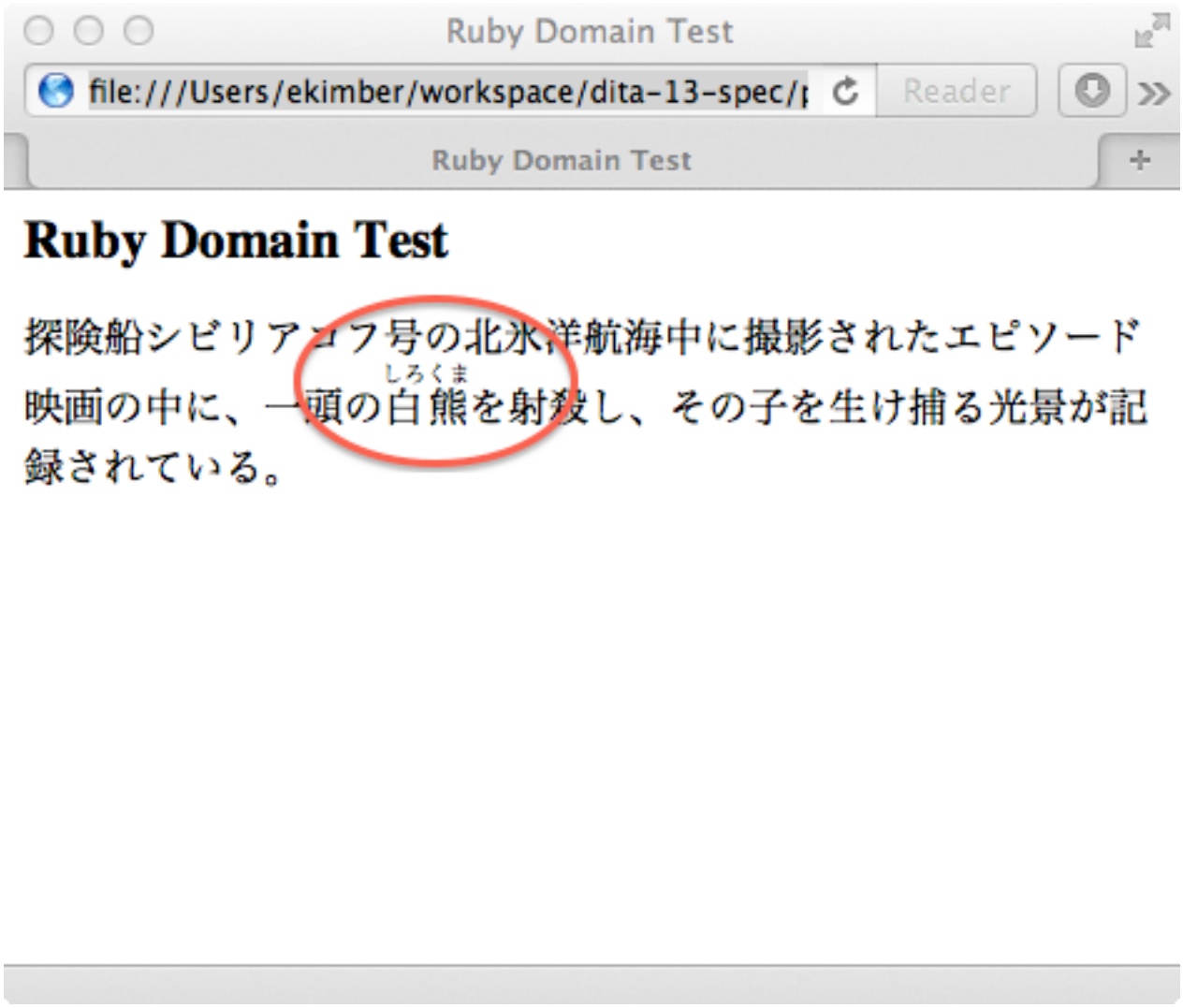

Figure 5: Ruby as rendered in HTML by Safari browser