CURIE Syntax 1.0

A syntax for expressing Compact URIs

W3C Working Draft 2 April 2008

This version:
  http://www.w3.org/TR/2008/WD-curie-20080402
Latest version:
  http://www.w3.org/TR/curie
Previous version:
  http://www.w3.org/TR/2007/WD-curie-20071126
Diff from previous version:
  curie-diff.html
Editors:
  Mark Birbeck, webBackplane mark.birbeck@webBackplane.com
  Shane McCarron, Applied Testing and Technology, Inc.

This document is also available in these non-normative formats: PostScript version, PDF version.

The English version of this specification is the only normative version. Non-normative translations may also be available.

Abstract

The aim of this document is to outline a syntax for expressing URIs in a generic, abbreviated syntax. While it has been produced in conjunction with the XHTML 2 Working Group, it is not specifically targeted at use by XHTML Family Markup Languages. Note that the target audience for this document is Language designers, not the users of those Languages.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This document is a Working Draft of the CURIE Syntax specification. Originally this document was based upon work done in the definition of [XHTML2 [p.15]], and work done by the RDF-in-HTML Task Force, a joint task force of the Semantic Web Best Practices and Deployment Working Group and XHTML 2 Working Group. It is being released in a separate, stand-alone specification in order to speed its adoption and facilitiate its use in various specifications. We believe the syntax rules defined in this document to be stable, and plan on proceeding to Last Call very soon.

This document has been produced by the W3C XHTML 2 Working Group as part of the HTML Activity. The goals of the XHTML 2 Working Group are discussed in the XHTML 2 Working Group charter.

This document was produced by a group operating under the 5 February 2004 W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

Publication as a Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

Please report errors in this specification to www-html-editor@w3.org (archive). It is inappropriate to send discussion email to this address. Public discussion may take place on www-html@w3.org (archive).

Table of Contents

1. Introduction

This section is informative.

More and more languages need a mechanism to permit the use of *extensible* value collections. These are primarily found in XML attribute values, but also found in other, similar spaces in non-XML languages (e.g., [SPARQL [p.15]]). Typically such extension mechanisms utilize the concept of *scoping*, where values are created within a unique *scope*, and that value space is managed by whomever defines it. Using such a mechanism allows independent organizations to define values without the risk of collision.

At the same time, language designers are trying to ensure that their languages mesh smoothly into the *semantic web*. Since the basis of the *semantic web* is the notion that meaning can be derived through the relationship among resources, these extension mechanisms need a ready way of mapping their scoped values to resources (via URIs).

In many cases, language designers are attempting to use QNames for this extension mechanism [XML-SCHEMA-QNAME [p.15]]. QNames do permit independent management of the value space, and *can* map the values to a resource. Unfortunately, QNames are unsuitable in most cases because 1) they are NOT intended for use in attribute values, and 2) the syntax of QNames is overly-restrictive and does not allow all possible URIs to be expressed.

A specific example of the problem this causes comes from attempting to define the value space for books. In a QName, the part after the colon must be a valid element name, making an example such as the following *invalid*: `isbn:0321154991`

This is not a valid QName simply because ’0321154991’ is not a valid element name. Yet, in the example given, we don’t really want to define a valid element name anyway. The whole reason for using a QName was to reference an item in a private value space - that of ISBNs. Moreover, in this example, we want that value to map to a URI that will reveal the meaning of that ISBN. As you can see, the definition of QNames and this (relatively common) use case are in conflict with one another.

This specification addresses the problem by creating a new data type whose purpose is specifically to allow for the definition of scoped values that map to URIs in exactly this way. This type is called a "CURIE" or a "Compact URI", and values that are syntactically valid QNames are a subset of this.

Note that this specification is targeted at markup language designers, not document authors. Any language designer considering the use of QNames in attribute values should consider instead using CURIEs, since CURIEs are designed for this purpose, while QNames are not.

1.1. Existing Uses of CURIEs

Although they are not currently called CURIEs, the technique described here is in widespread usage. However, taken literally, QNames would not support many of the examples that we would find ’in the wild’ — the fact that they do is mainly because systems and authors take a very lax approach to QNames.

In other words, the *principle* used in QNames — that of combining a *namespace name* with a *local part* to generate a URI — is widely used, but little checking is done on the *local part* to ensure that the string is a valid element name. However, this does mean that CURIEs can be easily used in a number of places, since there is already a large amount of ’mind-share’.

2. Usage

This section is informative.

CURIEs can be used in exactly the same syntactic way QNames have been used in attribute values, with the modification that the format of the strings after the colon is looser. In all cases a parsed CURIE will produce an IRI. However, the process of evaluating involves replacing the CURIE with a concatenation of the value represented by the prefix and the part after the colon (the *reference*).

Note that if CURIEs are to be used in the context of scripting, accessing a CURIE via standard mechanisms such as the XML DOM will return the raw CURIE, not its lexical value. In order to develop portable applications that evaluate CURIEs, a script author must transform CURIEs into their lexical value before evaluating them (e.g., dereferencing the resulting URI or comparing two CURIEs).

Also note that it is possible to define a CURIE prefix such that the resulting CURIEs in a document would resemble URIs (e.g., a prefix named ’mailto’ would engender CURIEs that look like `mailto:someone` but would expand to something else). Such CURIEs would only occur in contexts where a normal URI COULD NOT occur, but might still cause confusion for people reviewing the source of a document.

2.1. Example CURIEs

All of the following are valid CURIEs — even though they are not valid QNames — and they take advantage of the fact that the part after the colon no longer needs to conform to the rules for element names:

```
home:#start
joseki:
google:xforms+or+’xml+forms’
```

2.2. Ambiguities Between CURIEs and URIs

In some cases language designers will want to use both URIs and CURIEs as the value of an attribute. For example, in XHTML+RDFa [XHTMLRDFa [p.15]] the `about` attribute allows a URI to be specified that some metadata is "about", but it is also be useful to abbreviate this URI, using the compact syntax. However, the problem is that it is not possible for the language parser to be completely sure whether it has located a CURIE or a URI. For example, a resource could be specified as follows:

```
<p rel="foaf:homePage" about="http://www.example.org/home.html">home</p>
```

There is no way to be sure that this is a normal URI, or a CURIE. Therefore the syntax for carrying a CURIE when there is any possibility of ambiguity is to enclose the CURIE in square brackets, as in the following example:

```
<html xmlns:ex="http://www.example.org/">
    <head>...</head>
    <body>
        <p rel="foaf:homePage" about="[ex:home.html]">home</p>
    </body>
</html>
```

A CURIE enclosed in brackets is called a safe_curie [p.7] .

3. Syntax

This section is normative.

A CURIE is by definition a syntactic superset of a QName. It is comprised of two components, a *prefix* and a *reference*. The prefix is separated from the reference by a colon (`:`). It is possible to omit both the prefix and the colon, or to omit just the prefix and leave the colon. To disambiguate a CURIE when it appears in a context where a normal [URI [p.15]] may also be used, the entire CURIE is permitted to be enclosed in brackets (`[`, `]`).

```
safe_curie  :=   '[' curie ']'

curie       :=   [ [ prefix ] ':' ] reference

prefix      :=   NCName

reference   :=   irelative-ref (as defined in IRI)
```

When CURIES are used in an XML-based host language, prefix values MUST be able to be defined using the 'xmlns:' syntax specified in [XMLNAMES [p.15]]. Such host languages MAY also provide additional prefix mapping definition mechanisms.

When CURIES are used in a non-XML host language, the host language MUST provide a mechanism for defining the mapping from the `prefix` to an IRI.

A host language MAY interpret a *reference* value that is not preceded by a *prefix* and a colon as being a member of a host-language defined set of reserved values. Such reserved values MUST translate into an IRI, just as with any other CURIE.

A host language MAY declare a default prefix value, or MAY provide a mechanism for defining a default prefix value. In such a host language, when the `prefix` is omitted from a CURIE, the default prefix value MUST be used. Conversely, if such a language does not define a default prefix value mechanism and does not define a set of reserved values, CURIEs MUST NOT be used without a leading *prefix* and colon.

The concatenation of the prefix value associated with a CURIE and its `reference` MUST be an IRI [IRI [p.15]]. Note that while the set of IRIs represents the *lexical space* of a CURIE, the *value space* is the set of URIs (IRIs after canonicalization - see [IRI [p.15]]).

The CURIE prefix '_' is reserved. For this reason, the prefix '_' SHOULD be avoided by authors.

Host languages MAY define additional constraints on these syntax rules when CURIES are used in the context of those host languages. Host languages MUST NOT relax the constraints defined this specification.

A CURIE processor that encounters a value that does not conform the constraints defined by this specification and by the host language SHOULD ignore that value. A host language MAY require other behavior.

The `safe_curie` production is for use in attribute values where it would be otherwise impossible to disambiguate between a CURIE and a URI.

Language designers SHOULD only use CURIEs (or safe_curies) as the datatype of new attributes in their markup language, since using them in values where historically an attribute has taken a URI as its datatype could break backward compatibility.

4. Incorporating CURIEs into Host Languages

This section is informative.

CURIEs can be used in a variety of ways in host languages. This section shows a few simple examples.

4.1. SPARQL

The [SPARQL [p.15]] language provides a `PREFIX` keyword for defining the prefix used in their CURIE-like identifiers.

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
SELECT ?x ?name
WHERE  { ?x foaf:name ?name }
```

4.2. XHTML+RDFa

The RDFa Syntax specification defines an extended version of XHTML 1.1 called XHTML+RDFa. Because XHTML+RDFa is an XML markup language, the CURIE prefixes are defined using the `xmlns:` attribute.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN"
         "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">
<html version="XHTML+RDFa 1.0"
      xmlns="http://www.w3.org/1999/xhtml"
      xmlns:dc="http://purl.org/dc/elements/1.1/">
      <head version="XHTML+RDFa 1.0"
            profile="http://www.w3.org/1999/xhtml/vocab">
            <title>An XHTML+RDFa document using CURIEs</title>
      </head>
      <body>
          <p rel="cite">
              this content was written by
              <span property="dc:creator">some author</span>
          </p>
      </body>
</html>
```

4.3. XHTML 2

XHTML 2 includes the `role` attribute. This attribute takes advantage of CURIEs to permit the easy definition of additional roles for the content of a page.

```
<html version="XHTML2"
      xmlns="http://www.w3.org/1999/xhtml"
      xmlns:MR="http://www.example.org/roles/myRoles#">
      <head profile="http://www.w3.org/1999/xhtml/vocab">
          <title>An XHTML 2 document using Role</title>
      </head>
      <body>
          <p role="MR:main">The main content</p>
          <p role="MR:music">Some musical support for the page</p>
      </body>
</html>
```

5. Conformance Requirements

This section is *normative*.

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119 [p.15]].

5.1. CURIE Processor Conformance

A conforming CURIE processor must support all of the features required in this specification.

A. XML DTD and Schema Datatypes

This section is normative.

In order to facilitate the use of CURIEs in markup languages, this specification defines some additional datatypes in the XHTML datatype space (`http://www.w3.org/1999/xhtml/datatypes/`). Markup languages that use XHTML Modularization can find these definitions in the Modularization support file "datatypes" for their schema grammar.

CURIE	A single CURIE
CURIEs	A whitespace separated list of CURIEs
SafeCURIE	A single SafeCURIE
SafeCURIEs	A whitespace separated list of SafeCURIEs
URIorCURIE	A URI or a SafeCURIE (since you need a SafeCURIE to disambiguate between a common URI and a CURIE)

The following *informative* XML Schema definition for these datatypes is included as an example:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="http://www.w3.org/1999/xhtml/datatypes/"
 xmlns:xh11d="http://www.w3.org/1999/xhtml/datatypes/"
 targetNamespace="http://www.w3.org/1999/xhtml/datatypes/"
 elementFormDefault="qualified"
>
    <xs:simpleType name="CURIE">
        <xs:restriction base="xs:string">
            <xs:pattern value="[\i-[:]][\c-[:]]*:.+" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="CURIEs">
        <xs:list itemType="xh11d:CURIE" />
    </xs:simpleType>

    <xs:simpleType name="SafeCURIE">
        <xs:restriction base="xs:string">
            <xs:pattern value="\[[\i-[:]][\c-[:]]*:.+\]" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="SafeCURIEs">
        <xs:list itemType="xh11d:SafeCURIE" />
    </xs:simpleType>
```

```
        <xs:simpleType name="URIorCURIE">
            <xs:union memberTypes="xs:anyURI xh11d:SafeCURIE" />
        </xs:simpleType>
    </xs:schema>
```

B. References

B.1. Required Specifications

This section is normative.

IRI
"*Internationalized Resource Identifiers (IRI)*", RFC 3987, M.Duerst, M. Suignard January 2005.
Available at: http://www.ietf.org/rfc/rfc3987.txt

RFC2119
"*Key words for use in RFCs to indicate requirement levels*", RFC 2119, S. Bradner, March 1997.
Available at: http://www.ietf.org/rfc/rfc2119.txt

URI
"*Uniform Resource Identifiers (URI): Generic Syntax*", RFC 3986, T. Berners-Lee *et al.*, January 2005.
Available at: http://www.rfc-editor.org/rfc/rfc3986.txt. This RFC updates RFC 1738 [URI [p.15]] and obsoletes RFC 2732, 2396 and 1808.

XML
"*Extensible Markup Language (XML) 1.0 (Third Edition)*", W3C Recommendation, T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, *eds.*, 2 February 2004.
Available at: http://www.w3.org/TR/2004/REC-xml-20040204

XMLNAMES
"*Namespaces in XML*", W3C Recommendation, T. Bray, D. Hollander, A. Layman, *eds.*, 14 January 1999.
Available at: http://www.w3.org/TR/1999/REC-xml-names-19990114

B.2. Related Specifications

This section is informative.

XHTML2
"*XHTML™ 2.0*". J. Axelsson *et al.*, 26 July 2006.
Available at: http://www.w3.org/TR/2006/WD-xhtml2-20060726
The latest version is available at: http://www.w3.org/TR/xhtml2

RDFa in XHTML: Syntax and Processing
RDFa in XHTML: Syntax and Processing (See http://www.w3.org/TR/2008/WD-rdfa-syntax-20080221.)

XML-SCHEMA-QNAME
XML Schema Part 2: Datatypes Second Edition: Section 3.2.18 QName (See http://www.w3.org/TR/xmlschema-2/#QName.)

SPARQL
"*SPARQL Query Language for RDF*". Eric Prud'hommeaux *et al.*, 15 January 2008.
Available at: http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115

The latest version is available at: http://www.w3.org/TR/rdf-sparql-query