HTML+RDFa 1.1

Support for RDFa in HTML4 and HTML5

W3C Recommendation 22 August 2013

This version:
http://www.w3.org/TR/2013/REC-html-rdfa-20130822/
Latest version:
http://www.w3.org/TR/html-rdfa/
Previous version:
http://www.w3.org/TR/2013/PR-html-rdfa-20130625/
Editor:
Manu Sporny, Digital Bazaar, Inc.
Authors:
Shane McCarron, Applied Testing and Technology, Inc.
Ben Adida, Creative Commons
Mark Birbeck, Sidewinder Labs
Gregg Kellogg, Kellogg Associates
Ivan Herman, W3C
Steven Pemberton, CWI

Please refer to the **errata** for this document, which may include some normative corrections.

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

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

Abstract

This specification defines rules and guidelines for adapting the RDFa Core 1.1 and RDFa Lite 1.1 specifications for use in HTML5 and XHTML5. The rules defined in this specification not only apply to HTML5 documents in non-XML and XML mode, but also to HTML4 and XHTML documents interpreted through the HTML5 parsing rules.

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 has been reviewed by W3C Members, by software developers, and by other W3C groups and interested parties, and is endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

This specification is an extension to the HTML5 language. All normative content in the HTML5 specification, unless specifically overridden by this specification, is intended to be the basis for this specification.

Note

There are two features in this specification, `@datetime` processing and `rdf:HTML` literals, that are currently defined as non-normative features. The intent is that these features will eventually become normative features once the specification that describes the `@datetime` attribute [*HTML5 [p.27]*] and the specification that defines `rdf:HTML` [*RDF-CONCEPTS [p.27]*] become W3C Recommendations. Implementers should implement these features now; a 2nd (or later) edition of this specification will clarify the long-term stability of those features. Based on the discussion between the RDFa Working Group, the HTML Working Group, and the RDF Working Group, it is not expected that implementation changes will be necessary as HTML5 and RDF 1.1 advance to Recommendation.

A sample test harness is available for software developers. This set of tests is not intended to be exhaustive. A community-maintained website contains more information on further reading, developer tools, and software libraries that can be used to extract and process RDFa data from web documents.

This document was published by the RDFa Working Group as a Recommendation. If you wish to make comments regarding this document, please send them to public-rdfa-wg@w3.org (subscribe, archives). All comments are welcome.

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.

Table of Contents

1. Introduction

This section is non-normative.

Today's web is built predominantly for human readers. Even as machine-readable data begins to permeate the web, it is typically distributed in a separate file, with a separate format, and very limited correspondence between the human and machine versions. As a result, web browsers can provide only minimal assistance to humans in parsing and processing web pages: browsers only see presentation information. RDFa is intended to solve the problem of marking up machine-readable data in HTML documents. RDFa provides a set of HTML attributes to augment visual data with machine-readable hints. Using RDFa, authors may turn their existing human-visible text and links into machine-readable data without repeating content.

2. Conformance

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words *MUST*, *MUST NOT*, *REQUIRED*, *SHOULD*, *SHOULD NOT*, *RECOMMENDED*, *MAY*, and *OPTIONAL* in this specification are to be interpreted as described in [*RFC2119 [p.27]*].

2.1 Document Conformance

There are two types of document conformance criteria for HTML documents containing RDFa semantics; **HTML+RDFa** and **HTML+RDFa Lite**.

The following conformance criteria apply to any HTML document including RDFa markup:

- All document conformance requirements stated as mandatory in the HTML5 specification *MUST* be met.
- The appropriate Extensions to the HTML5 Syntax [p.16] , as described in this document, *MUST* be considered valid and conforming. Note that there are fewer supported attributes if the RDFa Lite syntax [*RDFA-LITE [p.27]*] is desired over the more advanced set of RDFa attributes outlined in RDFa Core.
- All HTML5 elements and attributes *SHOULD* be used in a way that conforms to [*HTML5 [p.27]*]. All RDFa attributes *SHOULD* be used in a way that is conforms to [*RDFA-CORE [p.27]*] and this document.

An example of a conforming HTML+RDFa document, with the RDFa portions highlighted in green:

Example 1: Example of an HTML+RDFa 1.1 document

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Example Document</title>
  </head>
  <body vocab="http://schema.org/">
    <p typeof="Blog">
      Welcome to my <a property="url" href="http://example.org/">blog</a>.
    </p>
  </body>
</html>
```

The following data will be extracted by a conforming RDFa processor, shown in Turtle format [*TURTLE [p.27]*]:
Example 2: Turtle output of Example Document

```
[] a <http://schema.org/Blog>;
   <http://schema.org/url> <http://example.org/> .
```

Non-XML mode HTML+RDFa 1.1 documents *SHOULD* be labeled with the Internet Media Type `text/html` as defined in section 12.1 of the HTML5 specification [*HTML5 [p.27]*].

XML mode XHTML5+RDFa 1.1 documents *SHOULD* be labeled with the Internet Media Type `application/xhtml+xml` as defined in section 12.3 of the HTML5 specification [*HTML5 [p.27]*], *MUST NOT* use a `DOCTYPE` declaration for XHTML+RDFa 1.0 or XHTML+RDFa 1.1, and *SHOULD NOT* use the `@version` attribute.

2.2 RDFa Processor Conformance

The RDFa processor conformance criteria are listed below, all of which are mandatory:

- An RDFa processor *MUST* implement all of the mandatory features specified in the RDFa Core 1.1 specification [*RDFA-CORE [p.27]*].
- An RDFa processor *MUST* support any mandatory features described in this specification.

2.3 User Agent Conformance

A user agent is considered to be a type of RDFa processor when the user agent stores or processes RDFa attributes and their values. The reason there are separate *RDFa Processor Conformance* and a *User Agent Conformance* sections is because one can be a valid HTML5 RDFa processor but not a valid HTML5 user agent (for example, by only providing a very small subset of rendering functionality).

The user agent conformance criteria are listed below, all of which are mandatory:

- A user agent *MUST* conform to all requirements listed in Section 2.2: Conformance Requirements of the HTML5 specification.
- A user agent *MUST* implement all of the features required by this specification.
- A user agent *MUST* implement all of the features required in the RDFa Core 1.1 specification, excluding those features which are specifically overridden by this specification as detailed in the Extensions to RDFa Core 1.1 [p.8] .

3. Extensions to RDFa Core 1.1

The RDFa Core 1.1 [*RDFA-CORE [p.27]*] specification is the base document on which this specification builds. RDFa Core 1.1 specifies the attributes and syntax, in Section 5: Attributes and Syntax, and processing model, in Section 7: Processing Model, for extracting RDF from a web document. This section specifies changes to the attributes and processing model defined in RDFa Core 1.1 in order to support extracting RDF from HTML documents.

The requirements and rules, as specified in RDFa Core and further extended in this document, apply to all HTML5 documents. An RDFa processor operating on both HTML and XHTML documents, specifically on their resulting DOMs or infosets, *MUST* apply these processing rules for HTML4, HTML5 and XHTML5 serializations, DOMs and/or infosets.

3.1 Additional RDFa Processing Rules

Documents conforming to the rules in this specification are processed according to [*RDFA-CORE [p.27]*] with the following extensions:

1. The default vocabulary URI is undefined.
2. HTML+RDFa uses an additional initial context by default, `http://www.w3.org/2011/rdfa-context/html-rdfa-1.1`, which must be applied after the initial context for [*RDFA-CORE [p.27]*] (`http://www.w3.org/2011/rdfa-context/rdfa-1.1`).
3. The base can be set using the `base` element. For XHTML5+RDFa 1.1 documents, base can also be set using the `@xml:base` attribute.
4. The current language can be set using either the `@lang` or `@xml:lang` attributes. When the `@lang` attribute and the `@xml:lang` attribute are specified on the same element, the `@xml:lang` attribute takes precedence. When both `@lang` and `@xml:lang` are specified on the same element, they *MUST* have the same value. Further details related to setting the current language can be found in section 3.3 Specifying the Language for a Literal [p.11] .
5. When determining which set of RDFa processing rules to use for documents served with the `application/xhtml+xml` media type, a conforming RDFa processor *MUST* look at the value in the DOCTYPE declaration of the document. If a DOCTYPE declaration exists, then the processing rules are:
   - XHTML1+RDFa 1.0 for a DOCTYPE of `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN" "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-1.dtd">`, or
   - XHTML1+RDFa 1.1 for a DOCTYPE of `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.1//EN" "http://www.w3.org/MarkUp/DTD/xhtml-rdfa-2.dtd">`, or
   - XHTML5+RDFa 1.1 for all other values of DOCTYPE.

   Documents served as `application/xhtml+xml`, that don't contain a DOCTYPE declaration, and don't specify a `@version` attribute *MUST* be interpreted as XHTML5+RDFa 1.1 documents.
6. In Section 7.5: Sequence, processing step 3, if the processor graph feature is supported and if an IRI mapping overwrites a previously existing mapping in the local list of IRI

mappings with a different value, the processor *MUST* generate an appropriate rdfa:PrefixRedefinition warning and place the associated triples into the processor graph.

7. In Section 7.5: Sequence, immediately after processing step 4, if the `@property` attribute and the `@rel` and/or `@rev` attribute exists on the same element, the non-CURIE and non-URI `@rel` and `@rev` values are ignored. If, after this, the value of `@rel` and/or `@rev` becomes empty, then the processor *MUST* act as if the respective attribute is not present.
8. In Section 7.5, processing step 5, and processing step 6, if no IRI is provided by a resource attribute (e.g., `@about`, `@href`, `@resource`, or `@src`), then first check to see if the element is the `head` or `body` element. If it is, then set new subject to parent object.
9. In Section 7.5: Sequence, processing step 11, the HTML5 `@datetime` attribute *MUST* be utilized when generating the current property value, unless `@content` is also present on the same element. Otherwise, if `@datetime` is present, the current property value must be generated as follows. The literal value is the value contained in the `@datetime` attribute. If `@datatype` is present, it is to be used as defined in the RDFa Core [*RDFA-CORE [p.27]*] processing rules. Otherwise, if the value of `@datetime` lexically matches a valid `xsd:date`, `xsd:time`, `xsd:dateTime`, `xsd:duration`, `xsd:gYear`, or `xsd:gYearMonth` a typed literal must be generated, with its datatype set to the matching xsd datatype. Otherwise, a plain literal *MUST* be generated, taking into account the current language. Implementers should note that the correct order of match testing should be: `xsd:duration`, `xsd:dateTime`, `xsd:date`, `xsd:time`, `xsd:gYearMonth`, and `xsd:gYear`. This feature is currently non-normative, see the note [p.2] on when it will become normative.
10. In Section 7.5: Sequence, processing step 11, if the element is `time`, and the element does not have `@datetime` or `@content` attributes, the processor *MUST* act as if there is a `@datetime` attribute containing exactly the elements text value. This feature is currently non-normative, see the note [p.2] on when it will become normative.
11. In Section 7.5: Sequence, step 11, immediately after sub-step 2, if the `@datatype` attribute is present and evaluates to `http://www.w3.org/1999/02/22-rdf-syntax-ns#HTML`, the value of the HTML Literal is a string created by serializing all child nodes to text. This applies to all nodes that are descendants of the current element, not including the element itself. The HTML Literal is given a datatype of `http://www.w3.org/1999/02/22-rdf-syntax-ns#HTML` as defined in *Section 5.2: The rdf:HTML Datatype* of [*RDF-CONCEPTS [p.27]*]. This feature is currently non-normative, see the note [p.2] on when it will become normative.
12. Once the output graph is generated following the processing steps defined in Section 7.5: Sequence of the RDFa Core 1.1 specification [*RDFA-CORE [p.27]*], and the steps in this section, perform the operations defined in Implementing Property Copying [p.15] .

The `@version` attribute is not supported in HTML5 and is non-conforming. However, if an HTML+RDFa document contains the `@version` attribute on the `html` element, a conforming RDFa processor *MUST* examine the value of this attribute. If the value matches that of a defined version of RDFa, then the processing rules for that version *MUST* be used. If the value does not match a defined version, or there is no `@version` attribute, then the processing rules for the most recent version of RDFa 1.1 *MUST* be used.

3.2 Modifying the Input Document

RDFa's tree-based processing rules, outlined in Section 7.5: Sequence of the RDFa Core 1.1 specification [*RDFA-CORE [p.27]*], allow an input document to be automatically corrected, cleaned-up, re-arranged, or modified in any way that is approved by the host language prior to processing. Element nesting issues in HTML documents *SHOULD* be corrected before the input document is translated into the DOM, a valid tree-based model, on which the RDFa processing rules will operate.

Any mechanism that generates a data structure equivalent to the HTML5 or XHTML5 DOM, such as the html5lib library, *MAY* be used as the mechanism to construct the tree-based model provided as input to the RDFa processing rules.

3.3 Specifying the Language for a Literal

According to RDFa Core 1.1 the current language *MAY* be specified by the host language. In order to conform to this specification, RDFa processors *MUST* use the mechanism described in *The lang and xml:lang attributes* section of the [*HTML5 [p.27]*] specification to determine the language of a node.

If the final encapsulating MIME type for an HTML fragment is not decided on while editing, it is *RECOMMENDED* that the author specify both `@lang` and `@xml:lang` where the value in both attributes is exactly the same.

Note

The HTML5 specification takes the `Content-Language` HTTP header into account when determining the language of an element. Some RDFa processor implementations, like those written in JavaScript, may not have access to this header and will be non-conforming in the edge case where the language is only specified in the `Content-Language` HTTP header. In these instances, RDFa document authors are urged to set the language in the document via the `@lang` attribute on the `html` element in order to ensure that the document is interpreted correctly across all RDFa processors.

3.4 Invalid XMLLiteral Values

When generating literals of type XMLLiteral, the processor *MUST* ensure that the output XMLLiteral is a namespace well-formed XML fragment. A namespace well-formed XML fragment has the following properties:

- The XML fragment, when placed inside of a single root element, *MUST* validate as well-formed XML. The normative language that describes a well-formed XML document is specified in Section 2.1 "Well-Formed XML Documents" of the XML specification.
- The XML fragment, when placed inside of a single root element, *MUST* retain all active namespace information. The currently active attributes declared using `@xmlns` and `@xmlns:` that are stored in the RDFa processor's current evaluation context in the IRI mappings *MUST* be preserved in the generated XMLLiteral. The *PREFIX* value for

`@xmlns:PREFIX` *MUST* be entirely transformed into lower-case characters when preserving the value in the XMLLiteral. All active namespaces declared via `@xmlns`, `@xmlns:`, and `@prefix` *MUST* be placed in each top-level element in the generated XMLLiteral, taking care to not overwrite pre-existing namespace values.

An RDFa processor that transforms the XML fragment *MUST* use the Coercing an HTML DOM into an infoset algorithm, as specified in the HTML5 specification, followed by the algorithm defined in the Serializing XHTML Fragments section of the HTML5 specification. If an error or exception occurs at any point during the transformation, the triple containing the XMLLiteral *MUST NOT* be generated.

Transformation to a namespace well-formed XML fragment is required because an application that consumes XMLLiteral data expects that data to be a namespace well-formed XML fragment.

The transformation requirement does not apply to plain text input data that are text-only, such as literals that contain a `@datatype` attribute with an empty value (`""`), or input data that contain only text nodes.

An example transformation demonstrating the preservation of namespace values is provided below. The â symbol is used to denote that the line is a continuation of the previous line and is included purely for the purposes of readability:

Example 3: Namespace preservation markup

```
<p xmlns:ex="http://example.org/vocab#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 Two rectangles (the example markup for them are stored in a triple):
 <svg xmlns="http://www.w3.org/2000/svg"
      property="ex:markup" datatype="rdf:XMLLiteral">
 â<rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:1; stroke:rgb(0,0,0)"/>
 â<rect width="50" height="50" style="fill:rgb(255,0,0);stroke-width:2;stroke:rgb(0,0,0)"/></svg>
</p>
```

The markup above *SHOULD* produce the following triple, which preserves the xmlns declaration in the markup by injecting the `@xmlns` attribute in the `rect` elements:

Example 4: Namespace preservation triple

```
<>
   <http://example.org/vocab#markup>
      """<rect xmlns="http://www.w3.org/2000/svg" width="300"
âheight="100" style="fill:rgb(0,0,255);stroke-width:1; stroke:rgb(0,0,0)"/>
â<rect xmlns="http://www.w3.org/2000/svg" width="50"
âheight="50" style="fill:rgb(255,0,0);stroke-width:2;
âstroke:rgb(0,0,0)"/>"""^^<http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral> .
```

Since the `ex` and `rdf` namespaces are not used in either `rect` element, they are not preserved in the XMLLiteral.

Similarly, compound document elements that reside in different namespaces must have their namespace declarations preserved:

Example 5: Namespace preservation for compound document elements

```
<p xmlns:ex="http://example.org/vocab#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:fb="http://www.facebook.com/2008/fbml">
 This is how you markup a user in FBML:
 <span property="ex:markup" datatype="rdf:XMLLiteral">
â<span><fb:user uid="12345">The User</fb:user></span>
â</span>
</p>
```

The markup above *SHOULD* produce the following triple, which preserves the `fb` namespace in the corresponding triple:

Example 6: Namespace element preservation triple

```
<>
   <http://example.org/vocab#markup>
      """<span xmlns:fb="http://www.facebook.com/2008/fbml">
â<fb:user uid="12345"></fb:user>
â</span>"""^^<http://www.w3.org/1999/02/22-rdf-syntax-ns#XMLLiteral> .
```

3.5 Property Copying

There are times when authors will find that they have many resources on a page that share a common set of properties. For example, several music events may have different performance times, but use the same location, band, and ticket prices. In this particular case, it is beneficial to have a short-hand notation to instruct the RDFa processor to include the location, band, and ticket price information without having to repeat all of the markup that expresses the data.

HTML+RDFa defines a *property copying* mechanism which allows properties associated with a resource to be copied to another resource. This mechanism can be activated by using the `rdfa:copy` predicate. The feature is illustrated in the following two examples:

Example 7: Events with duplicate properties

```
<div vocab="http://schema.org/">
  <p typeof="MusicEvent">
   <link property="image" href="Muse1.jpg"/>
   <link property="image" href="Muse2.jpg"/>
   <link property="image" href="Muse3.jpg"/>
   <span property="name">Muse</span> at the United Center.
   <time property="startDate" datetime="2013-03-03">March 3rd 2013</time>,
   <a property="location" href="#united">United Center, Chicago, Illinois</a>
   ...
  </p>

  <p typeof="MusicEvent">
   <link property="image" href="Muse1.jpg"/>
   <link property="image" href="Muse2.jpg"/>
   <link property="image" href="Muse3.jpg"/>
   <span property="name">Muse</span> at the Target Center.
   <time property="startDate" datetime="2013-03-07">March 7th 2013</time>,
```

```
    <a property="location" href="#target">Target Center, Minneapolis, Minnesota</a>
    ...
  </p>
</div>
```

In this case, two music events are defined with *image*, *name*, *startDate*, and *location* properties. The *image* and *name* values are identical for the two events and are unnecessarily duplicated in the markup. Using RDFa's *property copying* feature, a *pattern* can be declared that expresses the common properties. This pattern can then be copied into other resources within the document:

Example 8: Copying common properties

```
<div vocab="http://schema.org/">
  <div resource="#muse" typeof="rdfa:Pattern">
    <link property="image" href="Muse1.jpg"/>
    <link property="image" href="Muse2.jpg"/>
    <link property="image" href="Muse3.jpg"/>
    <span property="name">Muse</span>
  </div>

  <p typeof="MusicEvent">
    <link property="rdfa:copy" href="#muse"/>
    Muse at the United Center.
    <time property="startDate" datetime="2013-03-03">March 3rd 2013</time>,
    <a property="location" href="#united">United Center, Chicago, Illinois</a>
    ...
  </p>

  <p typeof="MusicEvent">
    <link property="rdfa:copy" href="#muse"/>
    Muse at the Target Center.
    <time property="startDate" datetime="2013-03-07">March 7th 2013</time>,
    <a property="location" href="#target">Target Center, Minneapolis, Minnesota</a>
    ...
  </p>
</div>
```

In this case, the common properties for all of the events are expressed in the first `div`. The common properties are copied into each event resource via the `rdfa:copy` predicate. The output for the previous two examples is the same:

Example 9: Turtle output of property copying example

```
@prefix schema: <http://schema.org/> .
@prefix xsd: http://www.w3.org/2001/XMLSchema#> .

[] a schema:MusicEvent;
   schema:image <Muse1.jpg>, <Muse2.jpg>, <Muse3.jpg>;
   schema:name "Muse";
   schema:startDate "2013-03-03"^^xsd:date;
   schema:location <#united> .

[] a schema:MusicEvent;
```

```
    schema:image <Muse1.jpg>, <Muse2.jpg>, <Muse3.jpg>;
    schema:name "Muse";
    schema:startDate "2013-03-07"^^xsd:date;
    schema:location <#target> .
```

The copy process is iterative, so that resources may copy other resources that copy other resources. For example:

Example 10: Resources may copy other resources that copy other resources

```
<div vocab="http://schema.org/">
  <div typeof="Person">
    <link property="rdfa:copy" href="#lennon"/>
    <link property="rdfa:copy" href="#band"/>
  </div>
  <p resource="#lennon" typeof="rdfa:Pattern">
    Name: <span property="name">John Lennon</span>
  <p>
  <div resource="#band" typeof="rdfa:Pattern">
    <div property="band" typeof="MusicGroup">
      <link property="rdfa:copy" href="#beatles"/>
    </div>
  </div>
  <div resource="#beatles" typeof="rdfa:Pattern">
    <p>Band: <span property="name">The Beatles</span></p>
    <p>Size: <span property="size">4</span> players</p>
  </div>
</div>
```

In the example above, the properties from `#lennon` and `#band` are copied into the first resource. Then the properties from `#beatles` are copied into `#band`. Subsequently, those properties are again copied into the first resource yielding the following output:

Example 11: Iterative copying example output in Turtle

```
@prefix schema: <http://schema.org/> .

[ a schema:Person;
  schema:name "John Lennon" ;
  schema:band [
    a schema:MusicGroup;
    schema:name "The Beatles";
    schema:size "4"
  ]
] .
```

Similar to *Vocabulary Expansion* as defined in [*RDFA-CORE [p.27]*], RDFa Property Copying operates on the output graph after document processing is complete.

3.5.1 Implementing Property Copying

Once the output graph is generated following the processing steps defined in Section 7.5: Sequence of the RDFa Core 1.1 specification [*RDFA-CORE [p.27]*], and the Extensions to the HTML5 Syntax [p.16] defined in this specification, processors *MUST* update the output graph using the following rules:

1. Run the following rule for each `rdfa:copy` statement in the output graph, and for each new `rdfa:copy` statement added as a result of property copying until no new triples are added to the output graph:

Rule name	If output graph contains	then add
pattern-copy	?subject rdfa:copy ?target ?target rdf:type rdfa:Pattern ?target ?predicate ?object	?subject ?predicate ?object

2. Finally, run this rule to remove utilized `rdfa:copy` statements and `rdfa:Pattern` resources from the output graph:

Rule name	If output graph contains	then remove
pattern-clean	?subject rdfa:copy ?target ?target rdf:type rdfa:Pattern ?target ?predicate ?object	?subject rdfa:copy ?target ?subject rdf:type rdfa:Pattern ?target ?predicate ?object

Note

Implementers should be aware that a simplistic implementation of the *pattern-copy* rule may lead to an infinite loop when handling circular dependencies. A processor should cease the *pattern-copy* rule when no unique triples are generated.

Note

Alternate rules may be used to update the output graph as long as the end result is the same.

4. Extensions to the HTML5 Syntax

There are a few attributes that are added as extensions to the HTML5 syntax in order to fully support RDFa:

- If HTML+RDFa Lite document conformance is desired, all RDFa attributes and valid values (including CURIEs), as listed in RDFa Lite 1.1, Section 2: The Attributes, *MUST* be allowed and validate as conforming when used in an HTML4, HTML5 or XHTML5 document. For the avoidance of doubt, the following RDFa attributes are allowed on all elements in the HTML5 content model: `@vocab`, `@typeof`, `@property`, `@resource`, and `@prefix`. All other attributes that RDFa may process, such as `@href` and `@src`, are only allowed when consistent with the content model for that element, as defined in the HTML5 specification.
- If HTML+RDFa document conformance is desired, all RDFa attributes and valid values (including CURIEs), as listed in RDFa Core 1.1, Section 2.1: The RDFa Attributes, *MUST* be allowed and validate as conforming when used in an HTML4, HTML5 or XHTML5 document. For the avoidance of doubt, the following RDFa attributes are allowed on all elements in the HTML5 content model: `@vocab`, `@typeof`, `@property`, `@resource`, `@prefix`, `@content`, `@about`, `@rel`, `@rev`, `@datatype`, and `@inlist`. All other attributes that RDFa may process, such as `@href` and `@src`, are only allowed when consistent with the content model for that element, as defined in the HTML5 specification.
- If the `@property` RDFa attribute is present on the `link` or `meta` elements, they *MUST* be viewed as conforming if used in the `body` of the document. More specifically, when `link` or `meta` elements contain the RDFa `@property` attribute and are used in the `body` of an HTML5 document, they *MUST* be considered flow content.
- If the RDFa `@property` attribute is present on the `link` element, the `@rel` attribute is not required.
- If the RDFa `@resource` attribute is present on the `link` element, the `@href` attribute is not required.
- If the RDFa `@property` attribute is present on the `meta` element, neither the `@name`, `@http-equiv`, nor `@charset` attributes are required and the `@content` attribute *MUST* be specified.

5. Backwards Compatibility

RDFa Core 1.1 deprecates the usage of `@xmlns:` in RDFa 1.1 documents. Web page authors *SHOULD NOT* use `@xmlns:` to express prefix mappings in RDFa 1.1 documents. Web page authors *SHOULD* use the `@prefix` attribute to specify prefix mappings.

However, there are times when XHTML+RDFa 1.0 documents are served by web servers using the `text/html` MIME type. In these instances, the HTML5 specification asserts that the document is processed according to the non-XML mode HTML5 processing rules. In these particular cases, it is important that the prefixes declared via `@xmlns:` are preserved for the RDFa processors to ensure backwards-compatibility with RDFa 1.0 documents. The following sections elaborate upon the backwards compatibility requirements for RDFa processor implementations.

5.1 `@xmlns:`-Prefixed Attributes

The RDFa Core 1.1 [*RDFA-CORE [p.27]*] specification deprecates the use of the `@xmlns:` mechanism to declare CURIE prefix mappings in favor of the `@prefix` attribute. However, there are instances where its use is unavoidable. For example, publishing legacy documents as HTML5 or supporting older XHTML+RDFa 1.0 documents that rely on the `@xmlns:` attribute.

CURIE prefix mappings specified using attributes prepended with `@xmlns:` *MUST* be processed using the algorithm defined in section 4.4.1: Extracting URI Mappings from Infosets [p.20] for infoset-based processors, or section 4.5.1: Extracting URI Mappings from DOMs [p.22] for DOM Level 2-based processors. For CURIE prefix mappings using the `@prefix` attribute, Section 7.5: Sequence, step 3 *MUST* be used to process namespace values.

Since CURIE prefix mappings have been specified using `@xmlns:`, and since HTML attribute names are case-insensitive, CURIE prefix names declared using the `@xmlns:`attribute-name pattern `xmlns:<PREFIX>="<URI>"` *SHOULD* be specified using only lower-case characters. For example, the text "`@xmlns:`" and the text in "`<PREFIX>`" *SHOULD* be lower-case only. This is to ensure that prefix mappings are interpreted in the same way between HTML (case-insensitive attribute names) and XHTML (case-sensitive attribute names) document types.

5.2 Conformance Criteria for `@xmlns:`-Prefixed Attributes

Since RDFa 1.0 documents may contain attributes starting with `@xmlns:` to specify CURIE prefixes, any attribute starting with a case-insensitive match on the text string "`@xmlns:`" *MUST* be preserved in the DOM or other tree-like model that is passed to the RDFa Processor. For documents conforming to this specification, attributes with names that have a case insensitive prefix matching "`@xmlns:`" *MUST* be considered conforming. Conformance checkers *SHOULD* accept attribute names that have a case insensitive prefix matching "`@xmlns:`" as conforming. Conformance checkers *SHOULD* generate warnings noting that the use of `@xmlns:` is deprecated. Conformance checkers *MAY* report the use of xmlns: as an error.

All attributes starting with a case insensitive prefix matching "`@xmlns:`" *MUST* conform to the production rules outlined in Namespaces in XML [*XML-NAMES11 [p.27]*], Section 3: Declaring Namespaces. Documents that contain `@xmlns:` attributes that do not conform to Namespaces in XML *MUST NOT* be accepted as conforming.

5.3 Preserving Namespaces via Coercion to Infoset

RDFa 1.0 documents may contain the `@xmlns:` pattern to declare prefix mappings, it is important that namespace information that is declared in non-XML mode HTML5 documents are mapped to an infoset correctly. In order to ensure this mapping is performed correctly, the "Coercing an HTML DOM into an infoset" rules defined in [*HTML5 [p.27]*] must be extended to include the following rule:

If the XML API is namespace-aware, the tool must ensure that ([namespace name], [local name], [normalized value]) namespace tuples are created when converting the non-XML mode DOM into an infoset. Given a standard `@xmlns:` definition, `xmlns:foo="http://example.org/bar#"`, the [namespace name] is `http://www.w3.org/2000/xmlns/`, the [local name] is `foo`, and the [normalized value] is `http://example.org/bar#`, thus the namespace tuple would be (`http://www.w3.org/2000/xmlns/`, `foo`, `http://example.org/bar#`).

For example, given the following input text:

Example 12

```
<div xmlns:com="https://w3id.org/commerce#">
```

The `div` element above, when coerced from an HTML DOM into an infoset, should contain an attribute in the [namespace attributes] list with a [namespace name] set to "`http://www.w3.org/2000/xmlns/`", a [local name] set to `com`, and a [normalized value] of "`https://w3id.org/commerce#`".

5.4 Infoset-based Processors

While the intent of the RDFa processing instructions is to provide a set of rules that are as language and toolchain independent as possible, for the sake of clarity, detailed methods of extracting RDFa content from processors operating on an XML Information Set are provided below.

5.4.1 Extracting URI Mappings from Infosets

Extracting URI Mappings declared via `@xmlns:` while operating from within an infoset-based RDFa processor can be achieved using the following algorithm:

While processing an element as described in [*RDFA-CORE [p.27]*], Section 7.5: Sequence, Step #2:

1. For each attribute in the [namespace attributes] list that has a [prefix] value, create an [IRI mapping] by storing the [prefix] as the value to be mapped, and the [normalized value] as the value to map.
2. For each attribute in the [attributes] list that has no value for [prefix] and a [local name] that starts with `@xmlns:`, create an [IRI mapping] by storing the [local name] part with the `@xmlns:` characters removed as the value to be mapped, and the [normalized value] as the value to map.
   Note

   This step is unnecessary if the infoset coercion rules preserve namespaces specified in non-XML mode.

For example, assume that the following markup is processed by an infoset-based RDFa processor:

Example 13

```
<div xmlns:ps="https://w3id.org/payswarm#" ...
```

After the markup is processed, there should exist a [URI mapping] in the [local list of URI mappings] that contains a mapping from `ps` to `https://w3id.org/payswarm#`.

5.4.2 Processing RDFa Attributes

There are a number of non-prefixed attributes that are associated with RDFa Processing in HTML5. If an XML Information Set based RDFa processor is used to process these attributes, the following algorithm should be used to detect and extract the values of the attributes.

While processing Infoset Attribute Information Items in Element Information Items as described in [*RDFA-CORE [p.27]*], Section 7.5: Sequence, Step #4 through Step #9:

1. For each Attribute Information Item specific to RDFa in the infoset [attributes] list that has a [prefix] with no value, extract and use the [normalized value].

5.5 DOM Level 1 and Level 2-based Processors

Most DOM-aware RDFa processors are capable of accessing DOM Level 1 [*DOM-LEVEL-1 [p.27]*] methods to process attributes on elements. To discover all `@xmlns:`-specified CURIE prefix mappings, the Node.attributes NamedNodeMap can be iterated over. Each Attr.name that starts with the text string `@xmlns:` specifies a CURIE prefix mapping. The value to be mapped is the string after the `@xmlns:` substring in the Attr.name variable and the value to be mapped is the value of the Attr.value variable.

The intent of the RDFa processing instructions are to provide a set of rules that are as language and toolchain independent as possible. If a developer chooses to not use the DOM1 environment mechanism outlined in the previous paragraph, they may use the following DOM2 [*DOM-LEVEL-2-CORE [p.27]*] environment mechanism.

5.5.1 Extracting URI Mappings via DOM Level 2

Extracting URI Mappings declared via `@xmlns:` while operating from within a DOM Level 2 based RDFa processor can be achieved using the following algorithm:

While processing each DOM2 [Element] as described in [*RDFA-CORE [p.27]*], Section 7.5: Sequence, Step #2:

1. For each [Attr] in the [Node.attributes] list that has a [namespace prefix] value of `@xmlns`, create an [IRI mapping] by storing the [local name] as the value to be mapped, and the [Node.nodeValue] as the value to map.
2. For each [Attr] in the [Node.attributes] list that has a [namespace prefix] value of null and a [local name] that starts with `@xmlns:`, create an [IRI mapping] by storing the [local name] part with the `@xmlns:` characters removed as the value to be mapped, and the [Node.nodeValue] as the value to map.
   Note

   This step is unnecessary if the XML and non-XML mode DOMs are namespace consistent.

For example, assume that the following markup is processed by a DOM2-based RDFa processor:

Example 14

```
<div xmlns:com="https://w3id.org/commerce#" ...
```

After the markup is processed, there should exist a [URI mapping] in the [local list of URI mappings] that contains a mapping from `com` to `https://w3id.org/commerce#`.

5.5.2 Processing RDFa Attributes

There are a number of non-prefixed attributes that are associated with RDFa processing in HTML5. If an DOM2-based RDFa processor is used to process these attributes, the following algorithm should be used to detect and extract the values of the attributes.

While processing an element as described in [*RDFA-CORE [p.27]*], Section 5.5: Sequence, Step #3 through Step #9:

1. For each RDFa attribute in the [Node.attributes] list that has a [namespace prefix] that is null, extract and use [Node.nodeValue] as the value.

Note

When extracting values from `@href` and `@src`, web authors and developers should note that certain values *MAY* be transformed if accessed via the DOM versus a non-DOM processor. The rules for modification of URL values can be found in the main HTML5 specification under Section 2.5: URLs.

A. About this Document

A.1 History

This section is non-normative.

In early 2004, Mark Birbeck published a document named "RDF in XHTML" via the XHTML2 Working Group wherein he laid the groundwork for what would eventually become RDFa (The Resource Description Framework in Attributes).

In 2006, the work was co-sponsored by the Semantic Web Deployment Working Group, which began to formalize a technology to express semantic data in XHTML. This technology was successfully developed and reached consensus at the W3C, later published as an official W3C Recommendation. While HTML provides a mechanism to express the structure of a document (title, paragraphs, links), RDFa provides a mechanism to express the meaning in a document (people, places, events).

The document, titled "RDF in XHTML: Syntax and Processing" [XHTML-RDFA], defined a set of attributes and rules for processing those attributes that resulted in the output of machine-readable semantic data. While the document applied to XHTML, the attributes and rules were always intended to operate across any tree-based structure containing attributes on tree nodes (such as HTML4, SVG and ODF).

While RDFa was initially specified for use in XHTML, adoption by a number of large organizations on the web spurred RDFa's use in non-XHTML languages. Its use in HTML4, before an official specification was developed for those languages, caused concern regarding document conformance.

Over the years, the members of the RDFa Community had discussed the possibility of applying the same attributes and processing rules outlined in the XHTML+RDFa specification to all HTML family documents. By design, the possibility of a unified semantic data expression mechanism between all HTML and XHTML family documents was squarely in the realm of possibility.

An RDFa Working Group was created in 2010 to address the issues concerning multiple language implementations of RDFa. The XHTML+RDFa document was split into a base specification, called RDFa Core 1.1 [*RDFA-CORE [p.27]*], and *thin* specifications that layer above RDFa Core 1.1. The XHTML+RDFa 1.1 specification [*XHTML-RDFA [p.28]*] is an example of such a *thin* specification. This document, also a *thin* specification, is targeted at HTML4, HTML5 and XHTML5.

This document describes the extensions to the RDFa Core 1.1 specification that permits the use of RDFa in all HTML family documents. By using the attributes and processing rules described in the RDFa Core 1.1 specification and heeding the minor changes in this document, authors can generate markup that produces the same semantic data output in HTML4, HTML5 and XHTML5.

A.2 Change History

This section is non-normative.

2009-10-15: First version of the RDFa for HTML4, HTML5 and XHTML5.

2010-03-04: Updated HTML5 coercion to infoset rules, preservation of namespaces in infoset and DOM2-based processors, clarifying how to extract RDFa attributes via infoset, how to extract RDFa attributes via DOM2.

2010-05-02: Inheritance of basic processing rules from RDFa 1.1 [*RDFA-CORE [p.27]*], instead of XHTML+RDFa 1.0 [*RDFA-SYNTAX [p.27]*], inclusion of the HTML Default Vocabulary Terms, inclusion of a HTML 4.01 + RDFa 1.1 DTD for validation purposes.

2010-06-24: Inheritance of basic processing rules from RDFa 1.1 [*RDFA-CORE [p.27]*], instead of XHTML+RDFa 1.0 [*RDFA-SYNTAX [p.27]*], inclusion of the HTML Default Vocabulary Terms, added HTML 4.01 + RDFa 1.1 DTD for validation purposes, added normative definition of @version attribute.

2010-10-19: Removal of @version attribute, migrated HTML Vocabulary Terms to RDFa Profile document, added statement to send comments to the HTML WG bug tracker.

2011-01-11: Removed decentralized extensibility issue markers, added DOM Level 1 prefix mapping extraction algorithm.

2011-04-05: Moved all xmlns: rules into a section titled Backwards Compatibility and brought spec in-line with latest RDFa Core 1.1 spec.

2011-05-12: Generated Last Call document, no substantive changes.

2011-12-30: Addition of normative dependency for RDFa Lite 1.1. Addition of rules to allow `meta` and `link` elements in flow and phrasing content as long as they contain at least one RDFa-specific attribute. Added support for `@datetime` and `value` processing.

2012-03-10: Clarification of where each RDFa attribute is allowed to be used. Feature at risk warning for HTML4+RDFa DTD-based validation.

2012-09-10: Publishing control of the HTML+RDFa document is handed over from the HTML WG to the newly re-chartered RDFa WG. DTD-based validation is removed from the specification.

2012-12-13: Addition of new HTML-specific processing rules for dealing with XHTML5 vs. HTML5 documents, xml:base, HTML Literals, property and rel/rev on the same element, and more types for the datetime attribute.

2012-12-27: Added Property Copying section and special processing for head and body.

2013-01-19: Removed @value processing, added @content overriding @datetime if present, cleanup and prep for Last Call publication in RDFa WG.

2013-06-06: Applied all Last Call comments and editorial fixes in preparation for Proposed Recommendation phase.

2013-08-07: Fixed invalid dates, bad grammar, updated status of document for Recommendation publication.

A.3 Acknowledgments

This section is non-normative.

At the time of publication, the members of the RDFa Working Group were:

Ivan Herman (staff contact), Shane McCarron, Gregg Kellogg, Niklas LindstrÃ¶m, Steven Pemberton, Manu Sporny (chair), Ted Thibodeau, and StÃ©phane Corlosquet.

A great deal of thanks to everyone that provided feedback on the specification (most of whom are listed below):

Adam Powell, Alex Milowski, Andy Seaborne, Arto Bendiken, Austin William, BAI Xi, Benjamin Adrian, Benjamin Nowack, Bjoern Hoehrmann, Christian Langanke, Christoph Lange, Cindy Lewis, Corey Mwamba, Crisfer Inmobiliaria, Dan Brickley, Daniel Friesen, Dave Beckett, David Wood, D. Grant, Dominik Tomaszuk, Dominique Hazael-Massieux, Doug Schepers, Dr. Olaf , Edward O'Connor, Faye Harris, Felix Sasaki, Gavin Carothers, Grant Robertson, Guus Schreiber, Harry Halpin, Michael Hausenblas, Henri Bergius, Henri Sivonen, Henry Story, Holger Knublauch, Ian Hickson, Irene Celino, Alexander Kroener, Knud MÃ¶ller, Philip JÃ¤genstedt, Reto Bachmann-GmÃ¼r, Ivan Mikhailov, James Leigh, Jeff Sonstein, Jeni Tennison, Jens Haupert, Jochen Rau, John Breslin, John Cowan, John O'Donovan, Jonathan Rees, Julian Reschke, KANZAKI Masahide, Kingsley Idehen, Knud Hinnerk, Landong Zuo, Leif Halvard Silli, Liam R., Lin Clark, Maciej Stachowiak, Mark Nottingham, Markus Gylling, Martin Hepp, Martin McEvoy, Matthias Tylkowski, Darin McBeath, Melvin Carvalho, Michael Chan, Michael Hausenblas, Michael Steidl, Michaelâ¢ Smith, Mischa Tuffield, Misha Wolf, Nathan Rixham, Nathan Yergler, Nicholas Stimpson, Noah Mendelsohn, Paul Cotton, Paul Sparrow, Pete Cordell, Peter Frederick, Peter Mika, Peter Occil, Phil Archer, Reece Dunn, Richard Cyganiak, Robert Leif, Robert Weir, Ramanathan V. Guha, Sami Korhonen, Sam Ruby, Sandro Hawke, Sebastian Germesin, Sebastian Heath, Shelley Powers, Simon Grant, Simon Reinhardt, Stefan Schumacher, Tab Atkins Jr., Thomas Adamich, Thomas Baker, Thomas Roessler, Thomas Steiner, Tim Berners-Lee, Toby Inkster, Tom Adamich, Tantek Ãelik, Ville SkyttÃ¤, Wayne Smith, and Will Clark

B. References

B.1 Normative references

[DOM-LEVEL-1]
Scott Isaacson; Steven B Byrne; Mike Champion; Ian Jacobs; Arnaud Le Hors; Gavin Nicol; Jonathan Robie; Robert S Sutor; Chris Wilson; Lauren Wood et al. *Document Object Model (DOM) Level 1*. 1 October 1998. W3C Recommendation. URL: http://www.w3.org/TR/DOM-Level-1/

[DOM-LEVEL-2-CORE]
Arnaud Le Hors; Philippe Le HÃ©garet; Lauren Wood; Gavin Nicol; Jonathan Robie; Mike Champion; Steven B Byrne et al. *Document Object Model (DOM) Level 2 Core Specification*. 13 November 2000. W3C Recommendation. URL: http://www.w3.org/TR/DOM-Level-2-Core/

[HTML5]
Robin Berjon et al. HTML5. 6 August 2013. W3C Candidate Recommendation. URL: http://www.w3.org/TR/html5/

[RDFA-CORE]
Ben Adida; Mark Birbeck; Shane McCarron; Ivan Herman et al. RDFa Core 1.1 - Second Edition: Syntax and processing rules for embedding RDF through attributes. 22 August 2013. W3C Recommendation. URL: http://www.w3.org/TR/rdfa-core/

[RDFA-LITE]
Manu Sporny. *RDFa Lite 1.1*. 7 June 2012. W3C Recommendation. URL: http://www.w3.org/TR/rdfa-lite/

[RFC2119]
S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels.* March 1997. Internet RFC 2119. URL: http://www.ietf.org/rfc/rfc2119.txt

[XML-NAMES11]
Tim Bray; Dave Hollander; Andrew Layman; Richard Tobin et al. *Namespaces in XML 1.1 (Second Edition)*. 16 August 2006. W3C Recommendation. URL: http://www.w3.org/TR/xml-names11/

B.2 Informative references

[RDF-CONCEPTS]
Richard Cyganiak, David Wood, Editors. *RDF 1.1 Concepts and Abstract Syntax* World Wide Web Consortium (work in progress). 23 July 2013. Last Call Working Draft.

[RDFA-SYNTAX]
Ben Adida; Mark Birbeck; Shane McCarron; Steven Pemberton et al. RDFa in XHTML: Syntax and Processing. 14 October 2008. W3C Recommendation. URL: http://www.w3.org/TR/2008/REC-rdfa-syntax-20081014

[TURTLE]
Eric Prud'hommeaux, Gavin Carothers. *Turtle: Terse RDF Triple Language.* 19 February 2013. W3C Candidate Recommendation. URL: http://www.w3.org/TR/turtle/

[XHTML-RDFA]
Shane McCarron. XHTML+RDFa 1.1 - Second Edition. 22 August 2013. W3C Recommendation. URL: http://www.w3.org/TR/xhtml-rdfa/