W3C Working Draft

World Wide Web Consortium

RDF 1.2 N-Triples

A line-based syntax for an RDF graph

W3C Working Draft 23 October 2026

This version:
https://www.w3.org/TR/2026/WD-rdf12-n-triples-20261023/

Latest published version:
https://www.w3.org/TR/rdf12-n-triples/

Latest editor's draft:
https://w3c.github.io/rdf-n-triples/spec/

History:
https://www.w3.org/standards/history/rdf12-n-triples/Commit history

Test suite:
https://w3c.github.io/rdf-tests/rdf/rdf12/rdf-n-triples/

Implementation report:
https://w3c.github.io/rdf-tests/rdf/rdf12/reports/

Latest Recommendation:
https://www.w3.org/TR/n-triples

Editors:
Gregg Kellogg (until 2025-09-06), in memoriam
Dominik Tomaszuk ORCID

Former editors:
Gavin Carothers (RDF 1.1)
Andy Seaborne (RDF 1.1)

Author:
David Beckett

Feedback:
GitHub w3c/rdf-n-triples (pull requests, new issue, open issues) public-rdf-star-wg@w3.org with subject line `[rdf12-n-triples] … message topic …` (archives)

Copyright © 2008-2026 World Wide Web Consortium. W3C® liability, trademark and permissive document license rules apply.

Contents

Abstract . . . 3
Status of This Document . . . 3
Set of Documents . . . 3
1 Introduction . . . 5
2 N-Triples Language . . . 6
2.1 Simple Triples . . . 6
2.2 Version Announcement . . . 6
2.3 Triple Terms . . . 6
2.4 IRIs . . . 7
2.5 RDF Literals . . . 7
2.6 RDF Blank Nodes . . . 8
3 A Canonical form of N-Triples . . . 9
4 Conformance . . . 10
4.1 Media Type and Content Encoding . . . 10
4.1.1 Other Media Types . . . 10
4.2 Future-proofness of N-Triples . . . 10
5 N-Triples Grammar . . . 11
5.1 White Space . . . 11
5.2 Comments . . . 11
5.3 Grammar . . . 11
5.4 Selected Terminal Literal Strings . . . 12
6 Parsing . . . 14
6.1 Parser State . . . 14
6.2 RDF Term Constructors . . . 14
6.3 RDF Graph Construction . . . 15
6.4 Non-conformant Documents . . . 15
A Privacy Considerations . . . 16
B Security Considerations . . . 17
C Internet Media Type and File Extension . . . 18
D Acknowledgments . . . 20
D.1 Acknowledgments for RDF 1.1 . . . 20
D.2 Acknowledgments for RDF 1.2 . . . 20
E Changes between RDF 1.1 and RDF 1.2 . . . 21
F Index . . . 22
F.1 Terms defined by this specification . . . 22
F.2 Terms defined by reference . . . 22
G Issue summary . . . 24
H References . . . 25
H.1 Normative references . . . 25
H.2 Informative references . . . 25
Publication diagnostics . . . 28
I Document provenance . . . 29

Abstract

N-Triples is a line-based, plain text format for encoding an RDF graph.
RDF 1.2 N-Triples introduces triple terms as a fourth kind of RDF term which can be used as the object of another triple, making it possible to make statements about other statements. RDF 1.2 N-Triples also adds support for directional language-tagged strings.

Status of This Document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this technical report can be found in the W3C standards and drafts index.

This document is part of the RDF 1.2 document suite. The N-Triples format is a line-based RDF syntax based on a subset of Turtle [*RDF12-TURTLE*].

To exit the W3C Candidate Recommendation phase, the W3C RDF & SPARQL Working Group requires that at least two independent implementations pass each test of the dedicated test suite.

This document was published by the RDF & SPARQL Working Group as a Working Draft using the Recommendation track.

Publication as a Working Draft does not imply endorsement by W3C and its Members.

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 a work in progress. Future updates to this upcoming recommendation may incorporate new features.

This document was produced by a group operating under the 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 that the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy.

This document is governed by the 18 August 2025 W3C Process Document.

Set of Documents

This document is one of eleven RDF 1.2 and twelve SPARQL 1.2 documents produced by the RDF & SPARQL Working Group.

List of documents

RDF 1.2 Documents:

1. *What's New in RDF 1.2*
2. *RDF 1.2 Concepts and Abstract Data Model*
3. *RDF 1.2 N-Quads*
4. RDF 1.2 N-Triples (this document)
5. *RDF 1.2 Primer*
6. *RDF 1.2 Schema*
7. *RDF 1.2 Semantics*
8. *RDF 1.2 TriG*
9. *RDF 1.2 Turtle*
10. *RDF 1.2 XML Syntax*
11. *RDF 1.2 Interoperability*

SPARQL 1.2 Documents:

1. *What's New in SPARQL 1.2*
2. *SPARQL 1.2 Concepts*

3. *SPARQL 1.2 Query Language*
4. *SPARQL 1.2 Update*
5. *SPARQL 1.2 Service Description*
6. *SPARQL 1.2 Federated Query*
7. *SPARQL 1.2 Query Results JSON Format*
8. *SPARQL 1.2 Query Results CSV and TSV Formats*
9. *SPARQL 1.2 Query Results XML Format*
10. *SPARQL 1.2 Entailment Regimes*
11. *SPARQL 1.2 Protocol*
12. *SPARQL 1.2 Graph Store Protocol*

1 Introduction

This section is non-normative.

This document defines N-Triples, a concrete syntax for RDF [*RDF12-CONCEPTS*]. N-Triples is an easy to parse line-based subset of Turtle [*RDF12-TURTLE*].

The syntax is a revised version of N-Triples as originally defined in the RDF Test Cases [*RDF-TESTCASES*] document. Its original intent was for writing test cases, but it has proven to be popular as an exchange format for RDF data. This specification further extends the syntax, as defined in [*N-TRIPLES*], to support the new features introduced by [*RDF12-CONCEPTS*]. This extension is fully backward compatible.

An N-Triples document can contain a single kind of parsing directive for announcing the RDF version of the content. See 2.2 Version Announcement.

N-Triples triples are a sequence of RDF terms representing the subject, predicate, and object of an RDF Triple. These may be separated by white space (`spaces`, and/or `tabs`). This sequence is terminated by a period (`.`), optionally followed by white space and/or a comment, and a new line (optional at the end of a document).

Example 1: Use of comments in N-Triples

```
<http://one.example/subject1> <http://one.example/predicate1> <http://one.example/object1>
 . # comments here
# or on a line by themselves
_:subject1 <http://an.example/predicate1> "object1" .
_:subject2 <http://an.example/predicate2> "object2" .
```

N-Triples triples are also Turtle simple triples, but Turtle includes other representations of RDF terms and abbreviations of RDF Triples. When parsed by a Turtle parser, data in the N-Triples format will produce exactly the same triples as a parser for the N-Triples language.

The RDF graph represented by an N-Triples document contains exactly each triple matching the N-Triples `triple` production.

2 N-Triples Language

This section is non-normative.
An N-Triples document allows writing down an RDF graph in a textual form. An RDF graph is made up of simple triples consisting of a subject, predicate, and object and optional blank lines. Comments may be given after a `#` that is not part of another lexical token and continue to the end of the line.

2.1 Simple Triples

The simplest triple statement is a sequence of (subject, predicate, and object) terms, and terminated by a period (`.`). White space (spaces `U+0020` or tabs `U+0009`) may surround terms, except where significant as noted in the grammar.
Comments are treated as white space, and may be given after a `#` that is not part of another lexical token and continue to the end of the line.
Example 2: Simple Triple

```
<http://example.org/#spiderman> <http://www.perceive.net/schemas/relationship/enemyOf> <ht
tp://example.org/#green-goblin> .
```

2.2 Version Announcement

The N-Triples language has evolved since its origin, and RDF 1.2 adds new syntax. RDF 1.2 N-Triples introduces the `VERSION` directive along with an optional `version` Media Type parameter. When respectively serializing and parsing N-Triples with new features such as initial text directions or triple terms, authors and parsers can announce and detect the use of the new syntax forms using these directives; similarly, HTTP clients and servers can use the `version` Media Type parameter.
In contrast to Turtle, the version declaration is case-sensitive.
Example 3: Version announcement

```
VERSION "1.2"
<http://example.org/#spiderman> <http://www.perceive.net/schemas/relationship/enemyOf> <ht
tp://example.org/#green-goblin> .
```

When transferring content over HTTP, senders and receivers can announce or request the version using the optional `version` Media Type parameter:
Example 4: HTTP version announcement

```
GET /document.nt HTTP/1.1
Host: example.com
Accept: application/n-triples; version=1.2
```

See Version Announcement in [*RDF12-TURTLE*] for more considerations on using the version declaration.

2.3 Triple Terms

A triple term may be the object of an RDF triple.
A triple term is represented as a `tripleTerm` with `subject`, `predicate`, and `object` preceded by `<<(`, and followed by `)>>`. Note that triple terms may be nested.
Example 5: Triple Term

```
VERSION                                                    "1.2"
_:e38  <ex:familyName>                                     "Smith" .
_:anno <http://www.w3.org/1999/02/22-rdf-syntax-ns#reifies> <<( _:e38 <http://example.com/
jobTitle> "Designer" )>> .
_:anno <http://example.com/accordingTo>                    _:e22 .
```

2.4 IRIs

N-Triples does not resolve relative IRI references. IRIs in N-Triples are written as resolved IRIs, as required by RDF 1.2 Concepts.

IRIs are preceded by `<` and followed by `>`, and may contain numeric escape sequences. For example, `<http://example.org/#green-goblin>`.

2.5 RDF Literals

Literals are used to identify values such as strings, numbers, dates.

Literals (Grammar production `Literal`) have a lexical form followed by either a language tag (possibly including an initial text direction), a datatype IRI, or neither.

The representation of the lexical form consists of an initial delimiter `"`, a sequence of permitted characters or numeric escape sequences or string escape sequences, and a final delimiter.

Literals may not contain the characters `"`, `LF`, or `CR` except in their escaped forms. In addition `\` may not appear in any quoted literal except as part of an escape sequence and a `"` character can only be included in a quoted literal using an escape sequence.

The corresponding lexical form is the characters between the delimiters, after processing any escape sequences. If present, the `LANG_DIR` terminal matches the language tag and optionally the initial text direction. The language tag is preceded by an `@`, and, if present, the initial text direction is separated from the language tag by `--`. If there is no language tag, there may be a datatype IRI, preceded by `^^`. If there is no datatype IRI and no language tag, then it is a simple literal and the datatype is `http://www.w3.org/2001/XMLSchema#string`.

Example 6: Literals in N-Triples

```
VERSION     "1.2"
<http://example.org/show/218> <http://www.w3.org/2000/01/rdf-schema#label> "That Seventies
 Show"^^<http://www.w3.org/2001/XMLSchema#string> . # literal with XML Schema string datat
ype
<http://example.org/show/218> <http://www.w3.org/2000/01/rdf-schema#label> "That Seventies
 Show" . # same as above
<http://example.org/show/218> <http://example.org/show/localName> "That Seventies Show"@en
 . # literal with a language tag
<http://example.org/show/218> <http://example.org/show/localName> "That Seventies Show"@en
-ltr . # literal with a language tag and an initial text direction
<http://example.org/show/218> <http://example.org/show/localName> "Cette Série des Années
Septante"@fr-be .  # literal outside of ASCII range with a region subtag
<http://example.org/#spiderman> <http://example.org/text> "This is a multi-line literal wi
th many quotation marks (""""") and two apostrophes ('')." .
<http://en.wikipedia.org/wiki/Helium> <http://example.org/elements/atomicNumber> "2"^^<htt
p://www.w3.org/2001/XMLSchema#integer> . # xsd:integer
<http://en.wikipedia.org/wiki/Helium> <http://example.org/elements/specificGravity> "1.663
E-4"^^<http://www.w3.org/2001/XMLSchema#double> .     # xsd:double
```

2.6 RDF Blank Nodes

RDF blank nodes are expressed as `_:` followed by a blank node label matching the `BLANK_NODE_LABEL` production.

Informally, the first character after `_:` is either a character matched by `PN_CHARS_U` or a digit. Any following characters, if present, are matched by `PN_CHARS` or by `.`, except that `.` is not permitted as the last character.

A fresh RDF blank node is allocated for each unique blank node identifier in a document. Repeated use of the same blank node identifier identifies the same blank node.

Example 7: Blank nodes in N-Triples

```
_:alice <http://xmlns.com/foaf/0.1/knows> _:bob .
_:bob   <http://xmlns.com/foaf/0.1/knows> _:alice .
```

3 A Canonical form of N-Triples

This section defines a canonical form of N-Triples which has a completely specified layout. The grammar for the language is unchanged.
While the N-Triples syntax allows choices for the representation and layout of RDF data, the canonical form of N-Triples provides a unique syntactic representation of any triple. Each code point can be represented by only one of `UCHAR`, `ECHAR`, or unencoded character, where the relevant production allows for a choice in representation. Each triple is represented entirely on a single line with specified white space.
Canonical N-Triples has the following additional constraints on layout:

- White space **MUST NOT** be used except after `subject`, `predicate`, `object`, and the terminal `<<(`, any of which **MUST** be a single `space`.
- A Canonical N-Triples document **MUST NOT** include a `VERSION` directive.
- Literals with the datatype `http://www.w3.org/2001/XMLSchema#string` **MUST NOT** use the datatype IRI part of the `literal`, and are represented using only `STRING_LITERAL_QUOTE`.
- `HEX` **MUST** use only digits (`[0–9]`) and uppercase letters (`[A–F]`).
- Alphabetic characters in `LANG_DIR` **MUST** use only the lowercase letters (`[a–z]`) with any uppercase letters case mapped to lowercase.
- Within `STRING_LITERAL_QUOTE`:
  - Characters `BS`, `HT`, `LF`, `FF`, `CR`, `"`, and `\` **MUST** be encoded using `ECHAR`.
  - Characters in the range from `U+0000` to `U+0007`, `VT`, characters in the range from `U+000E` to `U+001F`, `DEL`, and characters not matching the Char production from [*XML11*] **MUST** be represented by `UCHAR` using a lowercase `\u` with 4 `HEX`es.
  - All characters not required to be represented by `ECHAR` or `UCHAR` **MUST** be represented by their native [*UNICODE*] representation.

- The token `EOL` **MUST** be a single `LF`.
- The final `EOL` **MUST** be provided.

4 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 **MAY**, **MUST**, **MUST NOT**, and **SHOULD** in this document are to be interpreted as described in BCP 14 [*RFC2119*] [*RFC8174*] when, and only when, they appear in all capitals, as shown here.

This specification defines conformance criteria for:

- N-Triples documents
- Canonical N-Triples documents
- N-Triples parsers

A conforming **N-Triples document** is an RDF string that conforms to the grammar and additional constraints defined in 5. N-Triples Grammar, starting with the `ntriplesDoc` production. An N-Triples document serializes an RDF graph.

A conforming **Canonical N-Triples document** is an N-Triples document that follows the additional constraints of Canonical N-Triples.

A conforming **N-Triples parser** is a system capable of reading N-Triples documents on behalf of an application. It makes the serialized RDF graph, as defined in 6. Parsing, available to the application, usually through some form of API.

The IRI that identifies the N-Triples language is: `http://www.w3.org/ns/formats/N-Triples`

4.1 Media Type and Content Encoding

The media type of N-Triples is `application/n-triples`. The content encoding of N-Triples is always UTF-8 [*RFC3629*]. See N-Triples Media Type for the media type registration form.

4.1.1 Other Media Types

N-Triples has been historically provided with other media types. N-Triples may also be provided as `text/plain`. When used in this way N-Triples **MUST** use the escaped form of any character outside US-ASCII. As N-Triples is a subset of Turtle an N-Triples document **MAY** also be provided as `text/turtle`. In both of these cases the document is not an N-Triples document as an N-Triples document is only provided as `application/n-triples`.

4.2 Future-proofness of N-Triples

Any future evolution of the N-Triples format, if it reuses the same media type, **MUST** require the use of the `VERSION` directive, with a version label posterior to `1.2`, for any document that is not strictly compliant with N-Triples 1.2. This will allow parsers to gracefully handle more recent features that they do not support.

5 N-Triples Grammar

An N-Triples document is an RDF string encoded in UTF-8 [*RFC3629*]. Only Unicode scalar values, in the ranges `U+0000` to `U+D7FF` and `U+E000` to `U+10FFFF`, are allowed. This excludes surrogate code points, range `U+D800` to `U+DFFF`.

5.1 White Space

White space (`spaces`, and/or `tabs`) is allowed outside of terminals. Rule names below in capitals indicate where white space is significant.
White space is significant in the production `STRING_LITERAL_QUOTE`.
A **blank line**, consisting of only white space and/or a comment, may appear wherever a `triple` production is allowed, and are treated as white space.
Note
N-Triples allows only horizontal white space (`spaces` or `tabs`) as compared to Turtle [*RDF12-TURTLE*] which also treats `LFs` and `CRs` as white space.

5.2 Comments

Comments in N-Triples start at `#` outside an `IRIREF` or `STRING_LITERAL_QUOTE`, and continue to the end of line — marked by character `CR` or `LF` — or to the end of file, if there is no end of line after the comment marker. Comments are treated as white space.

5.3 Grammar

The EBNF used here is defined in XML 1.0 [*EBNF-NOTATION*].
Escape sequence rules are the same as Turtle [*RDF12-TURTLE*]. However, as only the `STRING_LITERAL_QUOTE` production is allowed new lines in literals **MUST** be escaped.
The 'VERSION' terminal is in single quotes to indicate that it is case-sensitive.

[1]	ntriplesDoc	::=	statement? (EOL statement)* EOL?
[2]	statement	::=	directive | triple
[3]	directive	::=	versionDirective
[4]	versionDirective	::=	'VERSION' versionSpecifier
[5]	versionSpecifier	::=	STRING_LITERAL_QUOTE
[6]	triple	::=	subject predicate object '.'
[7]	subject	::=	IRIREF | BLANK_NODE_LABEL
[8]	predicate	::=	IRIREF
[9]	object	::=	IRIREF | BLANK_NODE_LABEL | literal | tripleTerm
[10]	literal	::=	STRING_LITERAL_QUOTE (('^^' IRIREF) | LANG_DIR)?
[11]	tripleTerm	::=	'<<(' subject predicate object ')>>'
Productions for terminals			
[13]	IRIREF	::=	'<' ([^#x00-#x20<>"{}|^`\] | UCHAR)* '>'
[14]	BLANK_NODE_LABEL	::=	'_:' (PN_CHARS_U | [0-9]) ((PN_CHARS | '.')* PN_CHARS)?
[15]	LANG_DIR	::=	'@' [a-zA-Z]+ ('-' [a-zA-Z0-9]+)* ('--' [a-zA-Z]+)?

[16]	STRING_LITERAL_QUOTE	::=	'"' ([^#x22#x5C#x0A#x0D] | ECHAR | UCHAR)* '"'
[17]	UCHAR	::=	('\u' HEX HEX HEX HEX) | ('\U' HEX HEX HEX HEX HEX HEX HEX HEX)
[18]	ECHAR	::=	'\' [tbnrf\"']
[19]	PN_CHARS_BASE	::=	[A-Z]
		|	[a-z]
		|	[#xC0-#xD6]
		|	[#xD8-#xF6]
		|	[#xF8-#x02FF]
		|	[#x0370-#x037D]
		|	[#x037F-#x1FFF]
		|	[#x200C-#x200D]
		|	[#x2070-#x218F]
		|	[#x2C00-#x2FEF]
		|	[#x3001-#xD7FF]
		|	[#xF900-#xFDCF]
		|	[#xFDF0-#xFFFD]
		|	[#x00010000-#x000EFFFF]
[20]	PN_CHARS_U	::=	PN_CHARS_BASE | '_'
[21]	PN_CHARS	::=	PN_CHARS_U | '-' | [0-9] | #xB7 | [#x0300-#x036F] | [#x203F-#x2040]
[22]	HEX	::=	[0-9] | [A-F] | [a-f]
[23]	EOL	::=	[#x0D#x0A]+

A text version of this grammar is available here.

5.4 Selected Terminal Literal Strings

This document uses some specific terminal literal strings [*EBNF-NOTATION*]. To clarify the Unicode code points used for these terminal literal strings, the following table describes specific characters and sequences used throughout this document.

Code	Glyph	Description
U+0008	BS	Backspace
U+0009	HT	Horizontal tab
U+000A	LF	Line feed
U+000B	VT	Vertical tab
U+000C	FF	Form feed
U+000D	CR	Carriage return
U+0022	"	Quotation mark
U+0023	#	Number sign

Code	Glyph	Description
U+002D	-	Hyphen
U+002E	.	Full stop
U+0030	0	Digit zero
U+0039	9	Digit nine
U+003B	:	Colon
U+003C	<	Less-than sign
U+003E	>	Greater-than sign
U+0040	@	At sign
U+0041	A	Latin capital letter A
U+0046	F	Latin capital letter F
U+005C	\	Backslash
U+005F	_	Underscore
U+0061	a	Latin small letter A
U+007A	z	Latin small letter Z
U+007F	DEL	Delete
U+00B7	·	Middle dot
U+203F	‿	Undertie
U+2040	⁀	Character tie

Other short terminal literal strings are composed of specific sequences of Unicode characters:

space
: U+0020

<<(
: two concatenated less-than sign characters, each having the code point U+003C, followed by a left parenthesis character, having the code point U+0028

)>>
: a left parenthesis character, having the code point U+0029 followed by two concatenated greater-than sign characters, each having the code point U+003E

^^
: two concatenated circumflex accent characters, each having the code point U+005E

_:
: _ followed by :

--
: two concatenated - characters

6 Parsing

This section maps a string conforming to the grammar in 5.3 Grammar to a stream of triples by mapping strings matching productions and lexical tokens to RDF terms or their components (e.g., language tags, lexical forms of literals). Grammar productions change the parser state and emit triples.

6.1 Parser State

Parsing N-Triples requires a state of two items:

- Map[string -> blank node] `bnodeLabels` — A mapping from string to blank node.
- `xsd:string` *curVersion* – The RDF version used for parsing the document into Triples. If specified as part of a Media Type, the default value for *curVersion* is taken from the `version` parameter. Acceptable values for `version` are defined in 2.1 Version Labels in [*RDF12-CONCEPTS*]. The version announcement is only a hint; this specification does not mandate any parser behavior based on *curVersion*, but a parser **MAY** signal an error or a warning when it encounters a feature that does not match the value of *curVersion*.

6.2 RDF Term Constructors

This table maps productions and lexical tokens to RDF terms or components of RDF terms listed in 6. Parsing:

production	type	procedure
versionSpecifier	literal	The *curVersion* is taken from a literal using the matched RDF string lexical form and `xsd:string` datatype.
BLANK_NODE_LABEL	blank node	The string after `_:`, is a key in bnodeLabels. If there is no corresponding blank node in the map, one is allocated.
IRIREF	IRI	The characters between < and > are taken, with escape sequences unescaped, to form the IRI. The resulting IRI **MUST** comply with the syntactic restrictions of generic IRI syntax, and **SHOULD** conform to section 3.3 of [*RFC3986*] and comply with any narrower restrictions imposed by the corresponding IRI scheme specification.
LANG_DIR	language tag	The characters following the `@` form the language tag and optionally the initial text direction, if the matched characters include `--`. The language tag **MUST** be well-formed according to section 2.2.9 of [*BCP47*]. If present, the initial text direction **MUST** be either `ltr` or `rtl`.
STRING_LITERAL_QUOTE	lexical form	The characters between the outermost quotation marks (`"`) are taken, with escape sequences unescaped, to form the string of a lexical form.
literal	literal	The literal has a lexical form of the first rule argument, `STRING_LITERAL_QUOTE`, and either a language tag with optional initial text direction from `LANG_DIR` or a datatype IRI of `iri`, depending on which rule matched the input. If the `LANG_DIR` rule matched, the language tag and initial text direction are taken from LANG_DIR. If there is no initial text direction, the datatype is `rdf:langString`. If there is an initial text direction, the datatype is `rdf:dirLangString`. If neither `LANG_DIR` nor datatype IRI match, the literal has a datatype of `xsd:string`.
tripleTerm	triple term	The triple term is composed of the terms constructed from the `subject`, `predicate`, and `object` productions.

6.3 RDF Graph Construction

An N-Triples document defines an RDF graph composed of a set of RDF Triples. The `triple` production emits a triple defined by the terms constructed for `subject`, `predicate`, and `object`.

6.4 Non-conformant Documents

If the target of a parsing process does not conform to the grammar specified in 5. N-Triples Grammar, the result is not defined by this specification. Error detection, reporting, and recovery are left to implementors, because the appropriate behavior depends on the application interfaces and protocols within which the parser is used.

A Privacy Considerations

This section is non-normative.

The N-Triples format is used to express arbitrary application data, which may include the expression of personally identifiable information (PII) or other information which could be considered sensitive. Authors publishing such information are advised to carefully consider the needs and use of publishing such information, as well as the applicable regulations for the regions where the data is expected to be consumed and potentially revealed (e.g., GDPR, CCPA, others), particularly whether authorization measures are needed for access to the data.

B Security Considerations

This section is non-normative.

The `STRING_LITERAL_QUOTE` production allows the use of unescaped control characters. Although this specification does not directly expose this content to an end user, it might be presented through a user agent, which may cause the presented text to be obfuscated due to presentation of such characters.

N-Triples is a general-purpose assertion language; applications may evaluate given data to infer more assertions or to dereference IRIs, invoking the security considerations of the scheme for that IRI. Note in particular, the privacy issues in [*RFC3023*] section 10 for HTTP IRIs. Data obtained from an inaccurate or malicious data source may lead to inaccurate or misleading conclusions, as well as the dereferencing of unintended IRIs. Care must be taken to align the trust in consulted resources with the sensitivity of the intended use of the data; inferences of potential medical treatments would likely require different trust than inferences for trip planning.

The N-Triples language is used to express arbitrary application data; security considerations will vary by domain of use. Security tools and protocols applicable to text (for example, PGP encryption, checksum validation, password-protected compression) may also be used on N-Triples documents. Security/privacy protocols must be imposed which reflect the sensitivity of the embedded information.

N-Triples can express data which is presented to the user, such as RDF Schema labels. Applications rendering strings retrieved from untrusted N-Triples documents, or using unescaped characters, **SHOULD** use warnings and other appropriate means to limit the possibility that malignant strings might be used to mislead the reader. The security considerations in the media type registration for XML ([*RFC3023*] section 10) provide additional guidance around the expression of arbitrary data and markup.

N-Triples uses IRIs as term identifiers. Applications interpreting data expressed in N-Triples **SHOULD** address the security issues of *Internationalized Resource Identifiers (IRIs)* [*RFC3987*] Section 8, as well as *Uniform Resource Identifier (URI): Generic Syntax* [*RFC3986*] Section 7.

Multiple IRIs may have the same appearance. Characters in different scripts may look similar (for instance, a Cyrillic "o" may appear similar to a Latin "o"). A character followed by combining characters may have the same visual representation as another character (for example, LATIN SMALL LETTER "E" followed by COMBINING ACUTE ACCENT has the same visual representation as LATIN SMALL LETTER "E" WITH ACUTE). Any person or application that is writing or interpreting data in N-Triples must take care to use the IRI that matches the intended semantics, and avoid IRIs that may look similar. Further information about matching visually similar characters can be found in *Unicode Security Considerations* [*UNICODE-SECURITY*] and *Internationalized Resource Identifiers (IRIs)* [*RFC3987*] Section 8.

C Internet Media Type and File Extension

The Internet Media Type (formerly known as MIME Type) for N-Triples is "`application/n-triples`".
The information that follows has been submitted to the Internet Engineering Steering Group (IESG) for review, approval, and registration with IANA.

Type name:
application

Subtype name:
n-triples

Required parameters:
None

Optional parameters:

`version`
This parameter is optional. If present, acceptable values of `version` are defined in 2.1 Version Labels in [*RDF12-CONCEPTS*] or later specifications.

`profile`
This parameter is optional and is used to include additional information. It does not change the semantics of the resource representation when processed without knowledge of the profile. The value of a profile parameter is a non-empty list of space-separated URIs. For more information and background, please refer to [*RFC6906*].

Encoding considerations:
The syntax of N-Triples is expressed over code points in Unicode [*UNICODE*]. The encoding is always UTF-8 [*UTF-8*].
Unicode code points may also be expressed using an \uXXXX (U+0 to U+FFFF) or \UXXXXXXXX syntax (for U+10000 onwards) where X is a hexadecimal digit [0-9A-F]

Security considerations:
See B. Security Considerations.

Interoperability considerations:
There are no known interoperability issues.

Published specification:
This specification.

Applications which use this media type:
N-Triples is used widely for representing RDF data. There are implementations available in most common programming languages.

Additional information:

Magic number(s):
None.

File extension(s):
.nt;

Person & email address to contact for further information:
RDF & SPARQL Working Group <public-rdf-star-wg@w3.org>

Intended usage:
Common

Restrictions on usage:
None

Author(s):
The N-Triples specification is the product of the RDF & SPARQL WG. The W3C reserves change control over this specification.

Note
The `profile` parameter may be used by clients to express their preferences in the content negotiation process and by a server to indicate additional information about the response.
If the `profile` parameter is given by a client, a server should return a document that honors all the profiles in the list that are recognized by the server. A server should not respond with an error solely based upon the profile value.
If the `profile` parameter is given by a server, a client may choose to ignore it.

Note
It is recommended that profile URIs be dereferenceable and provide useful documentation at that URI.

Note
When used as a media type parameter [*RFC4288*] in an HTTP Content-Type header or an HTTP Accept header [*RFC7231*] the value of the `profile` parameter will need to be enclosed in quotes (ASCII `"`) if it contains special characters such as white space, including any space used to separate multiple profile URIs.
It is important to note that the value of the `profile` parameter contains one or more URIs and not IRIs. It might therefore be necessary to convert between IRIs and URIs, as specified in section 3 Relationship between IRIs and URIs of [*RFC3987*].

D Acknowledgments

This section is non-normative.

D.1 Acknowledgments for RDF 1.1

This section is non-normative.

The editor of the RDF 1.1 edition acknowledges valuable contributions from Gregg Kellogg, Eric Prud'hommeaux, Dave Beckett, David Robillard, Gregory Williams, Pat Hayes, Richard Cyganiak, Henry S. Thompson, Peter Ansell, Evan Patton and David Booth.

This specification is a product of extended deliberations by the members of the RDF Working Group. It draws upon the earlier specification in [*RDF-TESTCASES*], edited by Dave Beckett.

D.2 Acknowledgments for RDF 1.2

This section is non-normative.

The editors of the RDF 1.2 edition acknowledge valuable contributions from Andy Seaborne.

In addition to the editors, the following people have contributed to this specification: Andy Seaborne, Denis Ah-Kang, Gregory Todd Williams, james anderson, Niklas Lindström, Peter F. Patel-Schneider, Pierre-Antoine Champin, Sarven Capadisli, Ted Thibodeau Jr, and Thomas Tanon

Members of the RDF & SPARQL Working Group included the following:

James Anderson, Dörthe Arndt, Jerven Bolleman, Erich Bremer, Pierre-Antoine Champin, Souripriya Das, Enrico Franconi, Adrian Gschwend, Olaf Hartig, Gregg Kellogg†, Ora Lassila, Niklas Lindström, Thomas Lörtsch, Peter Patel-Schneider, Dave Raggett, Felix Sasaki, Andy Seaborne, Ruben Taelman, Thomas Pellissier Tanon, Ted Thibodeau Jr, Dominik Tomaszuk, Gregory Williams, William Van Woensel, and Antoine Zimmermann

† Gregg Kellogg passed away in September 2025. His vast contributions to RDF and the broader ecosystem of related standards are gratefully acknowledged.

Editor's note

Recognize members of the Task Force? Not an easy to find list of contributors.

E Changes between RDF 1.1 and RDF 1.2

This section is non-normative.

This specification extends the original N-Triples syntax, as defined in [*N-TRIPLES*], to support the new features introduced by [*RDF12-CONCEPTS*]. This extension is fully backward compatible: any document complying with the old version complies with the new version, and parses to the same graph. Furthermore, any document complying with the new version and containing only RDF 1.1 features is also compliant with the older version (with the exception of the `VERSION` directive; see 2.2 Version Announcement). Finally, none of the new syntactic constructs are valid in the old syntax. This means that any N-Triples document using RDF 1.2 features does not comply with the previous version of this specification and cannot be interpreted as a different graph under it.

More specifically, the following changes have been made:

- Better align the use of white space and comments with [*RDF12-TURTLE*].
- Removed language about white space use between terminals that would otherwise be (mis-)recognized, as this can't happen in N-Triples.
- Clarify the use of blank lines, including those composed of white space and/or comments. Comments can appear at the end of a triple before the newline as was already evident from Example 1.
- Create separate subsections of 5. N-Triples Grammar for White space and Comments, better mirroring [*RDF12-TURTLE*].
- Updated the `PN_CHARS_U` grammar production to be consistent with Turtle.
- Adds support for triple terms as described in 2.3 Triple Terms with updates to 6.2 RDF Term Constructors.
- Separated B. Security Considerations from C. Internet Media Type and File Extension and updated language.
- Changes the `LANGTAG` terminal production to `LANG_DIR` to include an optional initial text direction.
- Changes 3. A Canonical form of N-Triples to clarify use of datatype IRIs and expand the use of escapes in literals and to require that the characters in `LANG_DIR` be in lowercase.
- Clarify that Unicode surrogates are not legal in N-Triples documents, nor as the value of a Unicode escape sequence.
- Added 2.2 Version Announcement and parser state and productions to announce the RDF version associated with the input document.
- Added a `profile` parameter to the media type in C. Internet Media Type and File Extension.
- Added a normative requirement for processors to signal input errors (as errors or warnings).

F Index

This section is non-normative.

F.1 Terms defined by this specification

- blank line §5.1
- Canonical N-Triples document §4.
- N-Triples document §4.
- N-Triples parser §4.

F.2 Terms defined by reference

- [*BCP47*] defines the following:
  - section 2.2.9
- [*I18N-GLOSSARY*] defines the following:
  - case mapped
  - surrogate code points
  - Unicode scalar values
- [*RDF12-CONCEPTS*] defines the following:
  - 2.1 Version Labels
  - blank node identifier
  - datatype IRI
  - directional language-tagged strings
  - initial text directions
  - IRIs
  - language tag
  - lexical form
  - Literals
  - object type
  - predicate
  - RDF 1.2 Concepts
  - RDF blank nodes
  - RDF graph
  - RDF string
  - RDF term
  - simple literal
  - subject
  - triple
  - triple terms

- [*RDF12-TURTLE*] defines the following:
  - abbreviations of RDF Triples
  - numeric escape sequences
  - string escape sequences
  - Version Announcement

- [*RFC3986*] defines the following:
  - relative IRI references
  - resolved
  - section 3.3

- [*RFC3987*] defines the following:
  - syntactic restrictions

- [*XML11*] defines the following:
  - Char

G Issue summary

This section is non-normative.
There are no issues listed in this specification.

H References

H.1 Normative references

[BCP47]
Tags for Identifying Languages. Addison Phillips; Mark Davis. IETF. September 2009. Best Current Practice. URL: https://www.rfc-editor.org/info/bcp47

[EBNF-NOTATION]
EBNF Notation. Tim Bray; Jean Paoli; Michael Sperberg-McQueen; Eve Maler; François Yergeau et al. W3C. W3C Recommendation. URL: https://www.w3.org/TR/xml/#sec-notation

[I18N-GLOSSARY]
Internationalization Glossary. Richard Ishida; Addison Phillips. W3C. 17 October 2024. W3C Working Group Note. URL: https://www.w3.org/TR/i18n-glossary/

[RDF12-CONCEPTS]
RDF 1.2 Concepts and Abstract Data Model. Andy Seaborne; Gregg Kellogg; Olaf Hartig; Pierre-Antoine Champin. W3C. 7 April 2026. W3C Candidate Recommendation. URL: https://www.w3.org/TR/rdf12-concepts/

[RDF12-TURTLE]
RDF 1.2 Turtle. Gregg Kellogg; Andy Seaborne; Dominik Tomaszuk. W3C. 7 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/rdf12-turtle/

[RFC2119]
Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997. Best Current Practice. URL: https://www.rfc-editor.org/info/rfc2119/

[RFC3629]
UTF-8, a transformation format of ISO 10646. F. Yergeau. IETF. November 2003. Internet Standard. URL: https://www.rfc-editor.org/info/rfc3629/

[RFC3986]
Uniform Resource Identifier (URI): Generic Syntax. T. Berners-Lee; R. Fielding; L. Masinter. IETF. January 2005. Internet Standard. URL: https://www.rfc-editor.org/info/rfc3986/

[RFC3987]
Internationalized Resource Identifiers (IRIs). M. Duerst; M. Suignard. IETF. January 2005. Proposed Standard. URL: https://www.rfc-editor.org/info/rfc3987/

[RFC6906]
The 'profile' Link Relation Type. E. Wilde. IETF. March 2013. Informational. URL: https://www.rfc-editor.org/info/rfc6906/

[RFC8174]
Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017. Best Current Practice. URL: https://www.rfc-editor.org/info/rfc8174/

[UNICODE]
The Unicode Standard. Unicode Consortium. URL: https://www.unicode.org/versions/latest/

[XML11]
Extensible Markup Language (XML) 1.1 (Second Edition). Tim Bray; Jean Paoli; Michael Sperberg-McQueen; Eve Maler; François Yergeau; John Cowan et al. W3C. 16 August 2006. W3C Recommendation. URL: https://www.w3.org/TR/xml11/

H.2 Informative references

[N-TRIPLES]
RDF 1.1 N-Triples. Gavin Carothers; Andy Seaborne. W3C. 25 February 2014. W3C Recommendation. URL: https://www.w3.org/TR/n-triples/

[RDF-TESTCASES]
RDF Test Cases. jan grant; Dave Beckett. W3C. 10 February 2004. W3C Recommendation. URL: https://www.w3.org/TR/rdf-testcases/

[RDF12-INTEROP]
RDF 1.2 Interoperability. Pierre-Antoine Champin. W3C. W3C Editor's Draft. URL: https://w3c.github.io/rdf-interop/spec/

[RDF12-N-QUADS]
RDF 1.2 N-Quads. Gregg Kellogg; Dominik Tomaszuk. W3C. 10 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/rdf12-n-quads/

[RDF12-NEW]
What's New in RDF 1.2. The W3C RDF & SPARQL Working Group. W3C. W3C Editor's Draft. URL: https://w3c.github.io/rdf-new/spec/

[RDF12-PRIMER]
RDF 1.2 Primer. Pierre-Antoine Champin; Niklas Lindström. W3C. 17 September 2026. DNOTE. URL: https://www.w3.org/TR/rdf12-primer/

[RDF12-SCHEMA]
RDF 1.2 Schema. Dominik Tomaszuk. W3C. 22 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/rdf12-schema/

[RDF12-SEMANTICS]
RDF 1.2 Semantics. Peter Patel-Schneider; Dörthe Arndt; Enrico Franconi. W3C. 24 September 2026. CRD. URL: https://www.w3.org/TR/rdf12-semantics/

[RDF12-TRIG]
RDF 1.2 TriG. Gregg Kellogg; Dominik Tomaszuk. W3C. 7 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/rdf12-trig/

[RDF12-XML]
RDF 1.2 XML Syntax. Gregg Kellogg; Jerven Bolleman. W3C. 9 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/rdf12-xml/

[RFC3023]
XML Media Types. M. Murata; S. St. Laurent; D. Kohn. IETF. January 2001. Proposed Standard. URL: https://www.rfc-editor.org/info/rfc3023/

[RFC4288]
Media Type Specifications and Registration Procedures. N. Freed; J. Klensin. IETF. December 2005. Best Current Practice. URL: https://www.rfc-editor.org/info/rfc4288/

[RFC7231]
Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. R. Fielding, Ed.; J. Reschke, Ed. IETF. June 2014. Proposed Standard. URL: https://httpwg.org/specs/rfc7231.html

[SPARQL12-CONCEPTS]
SPARQL 1.2 Concepts. The W3C RDF & SPARQL Working Group. W3C. W3C Editor's Draft. URL: https://w3c.github.io/sparql-concepts/spec/

[SPARQL12-ENTAILMENT]
SPARQL 1.2 Entailment Regimes. Peter Patel-Schneider. W3C. 9 April 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-entailment/

[SPARQL12-FEDERATED-QUERY]
SPARQL 1.2 Federated Query. Ruben Taelman; Gregory Williams. W3C. 23 April 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-federated-query/

[SPARQL12-GRAPH-STORE-PROTOCOL]
SPARQL 1.2 Graph Store Protocol. Andy Seaborne; Thomas Pellissier Tanon. W3C. 24 September 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-graph-store-protocol/

[SPARQL12-NEW]
What's New in SPARQL 1.2. The W3C RDF & SPARQL Working Group. W3C. W3C Editor's Draft. URL: https://w3c.github.io/sparql-new/spec/

[SPARQL12-PROTOCOL]
SPARQL 1.2 Protocol. Andy Seaborne; Ruben Taelman; Gregory Williams; Thomas Pellissier Tanon. W3C. 23 July 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-protocol/

[SPARQL12-QUERY]
SPARQL 1.2 Query Language. Olaf Hartig; Andy Seaborne; Ruben Taelman; Gregory Williams; Thomas Pellissier Tanon. W3C. 11 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-query/

[SPARQL12-RESULTS-CSV-TSV]
SPARQL 1.2 Query Results CSV and TSV Formats. Ruben Taelman; Gregory Williams; Thomas Pellissier Tanon. W3C. 22 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-results-csv-tsv/

[SPARQL12-RESULTS-JSON]
SPARQL 1.2 Query Results JSON Format. Andy Seaborne; Ruben Taelman; Gregory Williams; Thomas Pellissier Tanon. W3C. 13 August 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-results-json/

[SPARQL12-RESULTS-XML]
SPARQL 1.2 Query Results XML Format. Ruben Taelman; Dominik Tomaszuk; Thomas Pellissier Tanon. W3C. 27 December 2024. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-results-xml/

[SPARQL12-SERVICE-DESCRIPTION]
SPARQL 1.2 Service Description. Ruben Taelman; Gregory Williams. W3C. 23 April 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-service-description/

[SPARQL12-UPDATE]
SPARQL 1.2 Update. Ruben Taelman; Andy Seaborne; Thomas Pellissier Tanon. W3C. 12 October 2026. W3C Working Draft. URL: https://www.w3.org/TR/sparql12-update/

[UNICODE-SECURITY]
Unicode Security Considerations. Mark Davis; Michel Suignard. Unicode Consortium. 19 September 2014. Unicode Technical Report #36. URL: https://www.unicode.org/reports/tr36/tr36-15.html

Publication diagnostics

unresolved-native-reference
No selected content defines the anchor #must-signal-errors (#must-signal-errors)

I Document provenance

This record identifies the generator and source revisions collected for this publication.

Generator	GraphArch 0.2.29 (build 412)
Generator commit	`8908a9610d77`

No Git repository revision was recorded for these sources.