OWL Quick Reference Guide

Authors
Jie Bao, Rensselaer Polytechnic Institute
Elisa F. Kendall, Sandpiper Software, Inc.
Deborah L. McGuinness, Rensselaer Polytechnic Institute
Contributors
Li Ding, Rensselaer Polytechnic Institute
Ankesh Khandelwal, Rensselaer Polytechnic Institute
Peter F. Patel-Schneider, Bell Labs Research, Alcatel-Lucent

Copyright © 2008-2009 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark and document use rules apply.

1 Name Spaces

Standard Namespaces and Prefixes used in OWL 2

Prefix	URI
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#
xsd	http://www.w3.org/2001/XMLSchema#

2 OWL 2 constructs and axioms

Constructs in bold are only available in OWL 2. Each table has columns:

- 1st: construct's name and link to Primer and New Features and Rationale (if applicable, as "(N)")
- 2nd: functional syntax and link to Syntax
- 3rd: RDF syntax with link to Mapping to RDF Graphs

We use the following notation conventions: unless stated otherwise, "C" is an OWL class, "D" is a data range, "P" and "Q" are object properties, "R" and "S" are data properties, "a" is an OWL individual, "u" and "v" are literals, "n" is a non-negative integer, "_:x" is anonymous individual. All names may have subscripts. "(a_1 … a_n)" in the 3rd column stands for a rdf list.

For an OWL 2 DL ontology, there are some global restrictions on axioms.

2.1 Classes

Predefined Classes

all OWL individuals	owl:Thing	owl:Thing
empty class	owl:Nothing	owl:Nothing

Boolean Connectives and Enumeration of Individuals

intersection	ObjectIntersectionOf(C_1 … C_n)	_:x rdf:type owl:Class. _:x owl:intersectionOf (C_1 … C_n).
union	ObjectUnionOf(C_1 … C_n)	_:x rdf:type owl:Class. _:x owl:unionOf (C_1 … C_n).
complement	ObjectComplementOf(C)	_:x rdf:type owl:Class. _:x owl:complementOf C.
enumeration	ObjectOneOf(a_1 … a_n)	_:x rdf:type owl:Class. _:x owl:oneOf (a_1 … a_n).

Object Property Restrictions

Every owl:Restriction is an owl:Class.

universal	ObjectAllValuesFrom(P C)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:allValuesFrom C
existential	ObjectSomeValuesFrom(P C)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:someValuesFrom C
individual value	ObjectHasValue(P a)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:hasValue i.
local reflexivity (N)	ObjectHasSelf(P)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:hasSelf "true"^^xsd:boolean.

Object Property Cardinality Restrictions

C is optional in OWL 2 and not allowed in OWL 1. (N)

exact cardinality	ObjectExactCardinality(n P **C**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:cardinality n. (without C) _:x owl:qualifiedCardinality n. _:x owl:onClass C. (with C)
maximum cardinality	ObjectMaxCardinality(n P **C**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:minCardinality n. (without C) _:x owl:minQualifiedCardinality n. _:x owl:onClass C. (with C)
minimum cardinality	ObjectMinCardinality(n P **C**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. _:x owl:maxCardinality n. (without C) _:x owl:maxQualifiedCardinality n. _:x owl:onClass C. (with C)

Data Property Restrictions

Every owl:Restriction is an owl:Class.

universal	DataAllValuesFrom(R D)	_:x rdf:type owl:Restriction. _:x owl:onProperty R. _:x owl:allValuesFrom D.
existential	DataSomeValuesFrom(R D)	_:x rdf:type owl:Restriction. _:x owl:onProperty R. _:x owl:someValuesFrom D.
individual value	DataHasValue(R u)	_:x rdf:type owl:Restriction. _:x owl:onProperty R. _:x owl:hasValue u.

Data Property Cardinality Restrictions

D is optional in OWL 2 and not allowed in OWL 1. (N)		
exact cardinality	DataExactCardinality(n R **D**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. --- _:x owl:cardinality "n"^^xsd:nonNegativeInteger. --- _:x owl:qualifiedCardinality "n"^^xsd:nonNegativeInteger. (without D) _:x owl:onDataRange D. (with D)
maximum cardinality	DataMaxCardinality(n R **D**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P. --- _:x owl:maxCardinality "n"^^xsd:nonNegativeInteger. (without D) --- _:x owl:maxQualifiedCardinality "n"^^xsd:nonNegativeInteger. _:x owl:onDataRange D. (with D)
minimum cardinality	DataMinCardinality(n R **D**)	_:x rdf:type owl:Restriction. _:x owl:onProperty P.(without D) --- _:x owl:minCardinality "n"^^xsd:nonNegativeInteger. --- _:x owl:minQualifiedCardinality "n"^^xsd:nonNegativeInteger. _:x owl:onDataRange D. (with D)

Restrictions Using n-ary Data Range

"D^n" is a n-ary data range (cf #Data Ranges).

n-ary universal (N)	DataAllValuesFrom(R_1 … R_n D^n)	_:x rdf:type owl:Restriction. _:x owl:onProperties (R_1 … R_n). _:x owl:allValuesFrom D^n.
n-ary existential (N)	DataSomeValuesFrom(R_1 … R_n D^n)	_:x rdf:type owl:Restriction. _:x owl:onProperties (R_1 … R_n). _:x owl:someValuesFrom D^n.

2.1.1 Class Axioms

subclasses	SubClassOf(C_1 C_2)	C_1 rdfs:subClassOf C_2.
equivalent classes	EquivalentClasses(C_1 … C_n)	C_1 owl:equivalentClass C_2. … C_{n-1} owl:equivalentClass C_n.
disjoint classes	DisjointClasses(C_1 C_2)	C_1 owl:disjointWith C_2.
pairwise disjoint classes (N)	DisjointClasses(C_1 … C_n)	_:x rdf:type owl:AllDisjointClasses. _:x owl:members (C_1 … C_n).
disjoint union (N)	DisjointUnionOf(C C_1 … C_n)	C owl:disjointUnionOf (C_1 … C_n).

2.2 Properties

2.2.1 Property Expressions

Object Properties instances of owl:ObjectProperty

universal object property	owl:topObjectProperty	owl:topObjectProperty
bottom object property	owl:bottomObjectProperty	owl:bottomObjectProperty
inverse property	ObjectInverseOf(P)	_:x owl:inverseOf P

Datatype Properties instances of owl:DatatypeProperty

universal datatype property	owl:topDataProperty	owl:topDataProperty
bottom datatype property	owl:bottomDataProperty	owl:topDataProperty

2.2.2 Property Axioms

Object Property Axioms

subproperty	SubObjectPropertyOf(P Q)	P rdfs:subPropertyOf Q.
property chain inclusion (N)	SubObjectPropertyOf(ObjectPropertyChain(P_1 … P_n) Q)	Q owl:propertyChainAxiom (P_1 … P_n).
property domain	ObjectPropertyDomain(P C)	P rdfs:domain C.
property range	ObjectPropertyRange(P C)	P rdfs:range C.
equivalent properties	EquivalentObjectProperties(P_1 … P_n)	P_1 owl:equivalentProperty P_2. … P_{n-1} owl:equivalentProperty P_n.
disjoint properties (N)	DisjointObjectProperties(P_1, P_2)	P_1 owl:propertyDisjointWith P_2.
pairwise disjoint properties (N)	DisjointObjectProperties(P_1 … P_n)	_:x rdf:type owl:AllDisjointProperties. _:x owl:members (P_1 … P_n).
inverse properties	InverseObjectProperties(P Q)	P owl:inverseOf Q.
functional property	FunctionalObjectProperty(P)	P rdf:type owl:FunctionalProperty.
inverse functional property	InverseFunctionalObjectProperty(P)	P rdf:type owl:InverseFunctionalProperty.
reflexive property (N)	ReflexiveObjectProperty(P)	P rdf:type owl:ReflexiveProperty.
irreflexive property (N)	IrreflexiveObjectProperty(P)	P rdf:type owl:IrreflexiveProperty.
symmetric property	SymmetricObjectProperty(P)	P rdf:type owl:SymmetricProperty.
asymmetric property (N)	AsymmetricObjectProperty(P)	P rdf:type owl:AsymmetricProperty.
transitive property	TransitiveObjectProperty(P)	P rdf:type owl:TransitiveProperty.

Datatype Property Axioms

subproperty	SubDataPropertyOf(R S)	R rdfs:subPropertyOf S.
property domain	DataPropertyDomain(R C)	R rdfs:domain C.
property range	DataPropertyRange(R C)	R rdfs:range C.
equivalent properties	EquivalentDataProperties(R_1 … R_n)	R_1 owl:equivalentProperty R_2. … R_{n-1} owl:equivalentProperty R_n.
disjoint properties (N)	DisjointDataProperties(R S)	R owl:propertyDisjointWith S.
pairwise disjoint properties (N)	DisjointDataProperties(R_1 … R_n)	_:x rdf:type owl:AllDisjointProperties. _:x owl:members (R_1 … R_n).
functional property	FunctionalDataProperty(R)	R rdf:type owl:FunctionalProperty.

2.3 Data Ranges

Data ranges are sets of tuples of literals. All datatypes (rdfs:Datatype) are unary data ranges. Complex data ranges are constructed from data types and other data ranges.
n-ary data ranges are not directly provided by OWL 2. However, OWL 2 provides syntactical hooks for applications to add n-ary data range support.

data range complement	DataComplementOf(D)	_:x rdf:type rdfs:Datatype. _:x owl:datatypeComplementOf D.
data range intersection	DataUnionOf(D_1…D_n)	_:x rdf:type rdfs:Datatype. _:x owl:unionOf (D_1…D_n).
data range union	DataIntersectionOf(D_1…D_n)	_:x rdf:type rdfs:Datatype. _:x owl:intersectionOf (D_1…D_n).
literal enumeration	DataOneOf(v_1 … v_n)	_:x rdf:type rdfs:Datatype. _:x owl:oneOf (v_1 … v_n).
datatype restriction (N)	DatatypeRestriction(D f_1 v_1 … f_n v_n) *f_j a constraining facet,* *v_j a restriction value*	_:x rdf:type rdfs:Datatype. _:x owl:onDatatype D. _:x owl:withRestrictions (y1 ... yn). y1 f_1 v_1. ... yn f_n v_n.

2.4 Keys

Keys (N)	HasKey(C (P_1 … P_m) (R_1 … R_n))	C owl:hasKey (P_1 … P_m R_1 … R_n).

2.5 Assertions

individual equality	SameIndividual(a_1,a_2)	a_1 owl:sameAs a_2.
individual equality	SameIndividual(a_1 … a_n)	a_j owl:sameAs a_{j+1}. j=1…n-1
individual inequality	DifferentIndividuals(a_1 a_2)	a_1 owl:differentFrom a_2.
pairwise individual inequality	DifferentIndividuals(a_1 … a_n)	_:x rdf:type owl:AllDifferent. _:x owl:members (a_1 … a_n).
class assertion	ClassAssertion(C a)	i rdf:type C.
positive object property assertion	ObjectPropertyAssertion(P a_1 a_2)	a_1 P a_2.
positive inverse object property assertion	ObjectPropertyAssertion(ObjectInverseOf(P) a_1 a_2)	a_2 P a_1.
positive data property assertion	DataPropertyAssertion(P a v)	a P v.
negative object property assertion (N)	NegativeObjectPropertyAssertion(P a_1 a_2)	_:x rdf:type owl:NegativePropertyAssertion. _:x owl:sourceIndividual a_1. _:x owl:assertionProperty P. _:x owl:targetIndividual a_2
negative datatype property assertion (N)	NegativeDataPropertyAssertion(R a u)	_:x rdf:type owl:NegativePropertyAssertion. _:x owl:sourceIndividual a. _:x owl:assertionProperty R. _:x owl:targetValue u

2.6 Declarations

class (N)	Declaration(Class(x))	x rdf:type owl:Class.
datatype (N)	Declaration(Datatype(x))	x rdf:type rdfs:Datatype.
object property (N)	Declaration(ObjectProperty(x))	x rdf:type owl:ObjectProperty.
datatype property (N)	Declaration(DataProperty(x))	x rdf:type owl:DatatypeProperty.
annotation property (N)	Declaration(AnnotationProperty(x))	x rdf:type owl:AnnotationProperty.
named individual (N)	Declaration(NamedIndividual(x))	x rdf:type owl:NamedIndividual.

2.7 Reification

If an axiom, when removed of all annotations, can be translated in a single RDF triple s p o, it can also be mapped to the following reified form

axioms	s p o	s p o. _:x rdf:type owl:Axiom. _:x owl:subject s. _:x owl:predicate p. _:x owl:object o.	s the subject, p the predicate, o the object of the triple.

Note: 1)for axioms of type EquivalentClasses, EquivalentProperties or SameIndividual, they will first be broken up into several RDF triples of their binary forms, then each triple is reified using the above rule. 2)for a axiom that is translated into a triple with s an anonymous node (e.g., in NegativePropertyAssertion), _:x is same to s.

2.8 Annotations

Annotation of a Resource. y the annotated resource, P annotation property, v a resource

annotations	Annotation(P v)	y P v. or (when the annotation itself has annotations) **y P v.** **_:x rdf:type owl:Annotation.** **_:x owl:subject y.** **_:x owl:predicate P.** **_:x owl:object v.**

Note: in OWL 2, an annotated resource can be an ontology, an ontology entity (in a declaration axiom), an axiom, an anonymous individual, or another annotation.

Annotation Properties all annotation properties are instances of owl:AnnotationProperty

human-readable name	Label	rdfs:label
human-readable description	Comment	rdfs:comment
additional information		rdfs:seeAlso
defining agent		rdfs:isDefinedBy
deprecation	Deprecated	owl:deprecated

Annotation Axioms
s the annotation subject, P annotation property, v a resource; P,Q annotation properties, U a URI.

annotation assertions	AnnotationAssertion(P s v)	s P v. or (if the assertion itself has annotation) **s P v.** **_:x rdf:type owl:Annotation.** **_:x owl:subject s.** **_:x owl:predicate P.** **_:x owl:object v.**

annotation subproperties	SubAnnotationPropertyOf(P Q)	P rdfs:subPropertyOf Q.
annotation property domain	AnnotationPropertyDomain(P U)	P rdfs:domain U.
annotation property range	AnnotationPropertyRange(P U)	P rdfs:range U.

Deprecation

C an OWL class or a datatype, P an object property, datatype property or annotation property.

deprecated class	C Deprecated	C rdf:type owl:DeprecatedClass.
deprecated property	P Deprecated	P rdf:type owl:DeprecatedProperty.

Note: "Deprecated" is the short for *owl:deprecated "true"^^xsd:boolean*

2.9 OWL Ontologies

O an ontology, U an ontology URI, V a URI.

OWL ontology	Ontology(O V) *(V is optional)*	O rdf:type owl:Ontology. O owl:versionInfo V.
Ontology Properties owl:OntologyProperty		
importing	O Import(U)	O owl:imports U.
backwards compatibility		O owl:backwardCompatibleWith U.
incompatibility		O owl:incompatibleWith U.
prior version		O owl:priorVersion U.

2.10 Deprecated Vocabulary in OWL 2

owl:DataRange	replaced by rdfs:Datatype
owl:distinctMembers	replaced by owl:members

3 Built-in Datatypes and Facets

3.1 Built-in Datatypes

The *value space* is a set determining the set of values of the datatype. A literal value "abc" of the datatype DT can be given in the form "abc"^^DT.

Numeric DataTypes

OWL Numeric Datetypes:

owl:rational	rational numbers
owl:real (N)	real numbers

XSD Numeric Datatypes

xsd:double	xsd:nonNegativeInteger	xsd:long	xsd:unsignedLong
xsd:float	xsd:nonPositiveInteger	xsd:int	xsd:unsignedInt
xsd:decimal	xsd:positiveInteger	xsd:short	xsd:unsignedShort
xsd:integer	xsd:negativeInteger	xsd:byte	xsd:unsignedByte

Strings: *value space* is of the form <"abc", tag>

Strings with a Language Tag: tag is either an empty string or a lowercase language tag

rdf:text	internationalized strings

Strings without a Language Tag: tag is an empty string

xsd:string	xsd:NCName
xsd:normalizedString	xsd:NMTOKEN
xsd:token	xsd:language
xsd:Name	

Boolean Values

xsd:Boolean	value space has only two values: *true* and *false*

Binary Data

xsd:base64Binary
xsd:hexBinary

IRIs

xsd:anyURI	IRIs as defined in XML Schema Datatypes

Time Instants

xsd:dateTime	time instants with time zone offset
xsd:dateTimeStamp	time instants without time zone offset

XML Literals

rdf:XMLLiteral	Note: at risk in OWL 2

3.2 Facets

The *facet space* is a set of pairs of the form < *f v* >, where *f* is a URI called a constraining facet, and *v* is a value. Each such pair is mapped to a subset of the value space of the datatype.

Notations: Numeric Datatype, String Datatype and Binary datatype refer to a set of datatypes based on the classification done in the prior section.

Facet(N) *f*	**Datatype**	**Value** *v*	**Explanation**
xsd:minInclusive, **xsd:maxInclusive,** **xsd:minExclusive,** **xsd:maxExclusive**	Numeric Datatype DT, Time instant DT	Literal in DT	Restricts the value-space to greater than (equal to) or lesser than (equal to) a value
xsd:minLength, **xsd:maxLength,** **xsd:length**	String Datatype, Binary Datatype, xsd:anyURI	Nonnegative integer	Restricts the value-space based on the lengths of the literals
xsd:pattern	String Datatype, xsd:anyURI	xsd:string literal whose value is a regular expression	Restricts the value space to literals that match the regular expression
rdf:langRange	rdf:text	xsd:string literal whose value is a regular expression	Restricts the value space to literals with language tags that match the regular expression