

---

# **OWL Web Ontology Language**

# Contenido

---

- OWL2: Introducción
- Conceptos Básicos de OWL
- Sintaxis – DL, OWL, Manchester
- Lenguaje OWL
  - Clases
  - Propiedades: : Object Properties, Data Properties
  - Individuos
- Clases: Definidas y Primitivas
- Razonamiento

# Clases Primitivas y Definidas en OWL

*Una clase que solo tiene las condiciones necesarias se conoce como clase **primitiva**.*

- $A \subseteq C$  (condiciones necesarias para A)

*Las condiciones necesarias se pueden leer como: "Si algo es miembro de esta clase, entonces es necesario cumplir estas condiciones"*

# Clases Primitivas y Definidas en OWL

*Una clase que tiene al menos un conjunto de condiciones necesarias y suficientes se conoce como clase **definida**.*

- $A \equiv C$  (condiciones necesarias y suficientes para A – definición)

Las condiciones necesarias y suficientes se denominan clases equivalentes

# Clases Primitivas y Definidas en OWL

Active ontology x Entities x Classes x Object properties x Data properties x Annotation

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Annotations:

Annotations

Annotations +

owl:Thing

Description:

Equivalent To +

SubClass Of +

Asserted

# Ejemplo: clases primitivas y definidas

---

Cuál es la diferencia entre usar:

- $\text{CosasCuenca} \equiv \text{localizadoEn.ZonaAustro}$

```
<owl:Class rdf:ID="CosasCuenca">  
  <owl:equivalentClass>  
    <owl:Restriction>  
      <owl:onProperty rdf:resource="#localizadoEn" />  
      <owl:someValuesFrom rdf:resource="#ZonaAustro" />  
    </owl:Restriction>  
  </owl:equivalentClass>  
</owl:Class>
```

- $\text{CosasCuenca} \subseteq \text{localizadoEn.ZonaAustro}$

```
<owl:Class rdf:ID="CosasCuenca">  
  <owl:subClassOf>  
    <owl:Restriction>  
      <owl:onProperty rdf:resource="#localizadoEn" />  
      <owl:someValuesFrom rdf:resource="# ZonaAustro " />  
    </owl:Restriction>  
  </owl:subClassOf>  
</owl:Class>
```

# Ejemplo: clases primitivas y definidas

---

- Cuál es la diferencia entre usar:
  - `owl:subClassOf`
    - Las cosas que se encuentran localizadas en ZonaAustro no son necesariamente cosas de Cuenca
    - $\text{CosasCuenca} \subseteq \text{localizadoEn.ZonaAustro}$ 
      - Expresa condición necesaria
  - `owl:equivalentClass`
    - Si algo está localizado en ZonaAustro, entonces debe estar en la clase CosasCuenca
    - $\text{CosasCuenca} \equiv \text{localizadoEn.ZonaAustro}$ 
      - Expresa una condición necesaria y suficiente

# Restricciones de Propiedad: Sintaxis

---

- Las restricciones de propiedad de OWL se utilizan para describir clases complejas mediante propiedades
- El axioma ***owl:Restriction*** permite describir este tipo de clases
- Tiene un elemento ***owl:onProperty*** y uno o más declaraciones de restricciones
- Ejemplo: Un padre debe tener al menos un hijo

```
:Padre rdfs:subClassOf
  [a owl:Restriction;
    owl:onProperty :tienehijo;
    owl:minCardinality "1"] .
```



# Restricciones de Propiedad

---

- Tipos
  - Restricciones de Valor
    - owl:allValuesFrom: cuantificador universal
    - owl:someValuesFrom: cuantificador existencial
    - owl:hasValue
  - Restricciones de Cardinalidad
    - owl:cardinality
    - owl:minCardinality
    - owl:maxCardinality

# Restricciones Propiedad: allValuesFrom

- Ejemplo: Expresar el conjunto de padres que solo tienen hijas

PadresConSoloHijas  $\sqsubseteq$  Persona  $\sqcap \forall \text{tieneHijo.Mujer}$

PadresConSoloHijas **subClassOf** Persona **and** tieneHijo **only** Mujer

```
<owl:Class rdf:ID="PadresConSoloHijas">
  <rdfs:subClassOf rdf:resource="#Persona" />
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#tieneHijo"/>
      <owl:allValuesFrom rdf:resource="#Mujer"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

# Restricciones de Propiedad: someValuesFrom

- Ejemplo: Expresar que una Madre es una Mujer que tiene un hijo (alguna Persona)

Madre  $\sqsubseteq$  Mujer  $\sqcap \exists \text{tieneHijo. Persona}$

Madre **subclassOf** Mujer **and** tieneHijo **some** Persona

```
<owl:Class rdf:ID="Madre">
  <rdfs:subClassOf rdf:resource="#Mujer" />
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#tieneHijo"/>
      <owl:someValuesFrom rdf:resource="#Persona"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

# Restricciones de Propiedad: hasValue

---

- Permite definir clases basadas en la existencia de valores de propiedad particulares
- Ejemplo:

CursoFisica **subClassOf** esTomadoPor value "949352"

```
<owl:Class>
<rdfs:subClassOf>rdf:resource="#CursoFisica"/>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource= "#esTomadoPor"/>
<owl:hasValue rdf:resource= "#949352"/>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

# Restricciones de Propiedad: cardinalidad

---

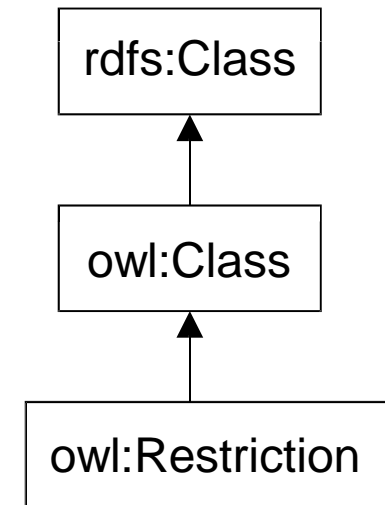
- Definición de **cardinalidad**:
  - el numero de ocurrencias, ya sea **maximo** (`owl:maxCardinality`) o **minimo** (`owl:minCardinality`) o **exacto** (`owl:cardinality`) basado en el contexto en la cual es usada
- Ejemplo. Cursos tomados al menos por dos personas

```
[a owl:Restriction;  
  owl:onProperty :esTomadoPor;  
  owl:minCardinality  
  "2"^^xsd:nonNegativeInteger] .
```

# Restricciones de Propiedad: Resumen

---

- Define una Clase usando restricciones LOCALES sobre una Propiedad especifica



- **Restricciones de Propiedad:**
  - allValuesFrom: rdfs:Class (lite/DL owl:Class)
  - hasValue: individuo especifico
  - someValuesFrom: rdfs:Class (lite/DL owl:Class)
  - cardinality: xsd:nonNegativeInteger (in lite {0,1})
  - minCardinality: xsd:nonNegativeInteger (in lite {0,1})
  - maxCardinality: xsd:nonNegativeInteger (in lite {0,1})

---

# **Inicio Tutorial – OWL**

## **Creación de Clases Primitivas, Definidas y Restricciones de Propiedades**

# Ontología sobre Juegos de Video

Supongamos que queremos construir una ontología sobre videojuegos de la siguiente manera.

| Clases  | Clases  | Propiedades   | definiciones   |
|---|---|---|--|
| <ul style="list-style-type: none"><li>- Juego<ul style="list-style-type: none"><li>- JuegoFamoso<ul style="list-style-type: none"><li>- LoL</li><li>- Ajedrez</li><li>- Sudoku</li></ul></li></ul></li><li>- Plataforma<ul style="list-style-type: none"><li>- Windows</li><li>- MacOSX</li><li>- Linux</li></ul></li></ul> | <ul style="list-style-type: none"><li>- TipoJuego<ul style="list-style-type: none"><li>- UnSoloJugador</li><li>- MultiJugador</li><li>- DeRoles</li><li>- Enlinea</li></ul></li><li>- DificultadJuego<ul style="list-style-type: none"><li>- Difícil</li><li>- Normal</li><li>- Fácil</li></ul></li></ul> | <ul style="list-style-type: none"><li>tieneDificultad</li><li>tienePlataforma</li><li>tieneTipo</li></ul> | <ul style="list-style-type: none"><li>JuegoMultiPlataforma</li><li>JuegoDifícil</li><li>JuegoNormal</li><li>JuegoFacil</li><li>JuegoParaLinux</li><li>JuegoParaWindows</li><li>JuegoParaMacOSX</li><li>JuegoMultiJugador</li><li>...</li></ul> |



# Agregando Clases Primitivas y Definidas

---

## Cuáles clases?

- Solamente clases de las siguientes formas
  - $A \subseteq C$  (condiciones necesarias para A)
  - $A \equiv C$  (condiciones necesarias y suficientes para A – definición)
- Por cada subclase de JuegosFamosos se requiere insertar axiomas como:
  - Ajedrez puede ser instalado en cualquier plataforma
  - League of Legends es un juego en línea

# Clases Primitivas – Ejemplo

---

- Especificación en lenguaje natural

Ajedrez puede ser instalado en cualquier plataforma

- Replantear la especificación utilizando el vocabulario de ontología.

Ajedrez tiene plataforma Windows, tiene plataforma MacOSX,  
y tiene plataforma Linux

- Escríbalo usando lógica descriptiva (opcional)

$Ajedrez \sqsubseteq \exists tienePlataforma.Windows$

$Ajedrez \sqsubseteq \exists tienePlataforma.MacOSX$

$Ajedrez \sqsubseteq \exists tienePlataforma.Linux$

- Escríbalo usando sintaxis Manchester (el lado derecho es suficiente)

tienePlataforma some Windows

tienePlataforma some MacOSX

tienePlataforma some Linux

# Agregando axiomas a la Clase “Ajedrez”

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window shows the ontology 'untitled-ontology-20' with a search bar and a breadcrumb path: > Juego > JuegoFamoso > Ajedrez. The 'Active Ontology' tab is selected, showing the 'Class hierarchy' and 'Usage' views. The 'Class hierarchy' view on the left shows a tree structure with 'Ajedrez' highlighted under 'JuegoFamoso'. A red arrow points to 'Ajedrez' in this tree. The 'Usage' view on the right shows 'Found 10 uses of Ajedrez', including 'Ajedrez SubClassOf JuegoFamoso', 'Class: Ajedrez', and 'DisjointClasses: Ajedrez, LeagueOfLegends, Sudoku'. The 'Description: Ajedrez' panel on the right is highlighted with a red box and contains the following information:

- Equivalent To: +
- SubClass Of: + (highlighted with a red arrow) **JuegoFamoso**
- General class axioms: +
- SubClass Of (Anonymous Ancestor):
- Instances: +
- Target for Key: +
- Disjoint With: + **Sudoku, LeagueOfLegends**
- Disjoint Union Of: +

At the bottom right, a status bar indicates: 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences'.

# Agregando axiomas a la Clase “Ajedrez”

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) : [http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)

Search...

Juego > JuegoFamoso > Ajedrez

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Ajedrez

Class hierarchy: Ajedrez

owl:Thing

- DificultadJuego
  - Dificil
  - Facil
  - Normal
- Juego
  - JuegoFamoso
    - **Ajedrez**
    - LeagueOfLegends
    - Sudoku
  - Plataforma
    - Linux
    - MacOSX
    - Windows
  - TipoJuego
    - EnLinea
    - JuegoDeRoles
    - MultiJugador
    - Rompecabezas
    - UnSoloJugador

Found 10 uses of Ajedrez

- Ajedrez SubC
- Class: Ajedre
- DisjointClas

Description: Ajedrez

Equivalent To +

SubClass Of +

JuegoFamoso

General class axioms +

SubClass Of (Anonymous Ancesto

Instances +

Target for Key +

Disjoint With +

Sudoku, LeagueOf

Disjoint Union Of +

Class expression editor Object restriction creator Data restriction creator Class hierarchy

tienePlataforma some Windows

Help...

Aceptar Cancelar

# Agregando Axiomas a la Clase “Ajedrez”

The screenshot displays the Protégé ontology editor interface. The main window shows the class hierarchy for 'Ajedrez', which is a subclass of 'JuegoFamoso'. A red arrow points to the 'SubClass Of' button in the 'Description: Ajedrez' panel. The 'Ajedrez' dialog box is open, showing the 'Class expression editor' with the following properties:

- Restricted property: `owl:topObjectProperty`
- Restricted property: `tieneTipo`
- Restricted property: `tieneDificultad`
- Restricted property: `tienePlataforma`

The 'Restriction type' is set to 'Some (existential)' and the cardinality is 1. The 'Restriction filler' shows a hierarchy of classes:

- `owl:Thing`
- `DificultadJuego`
- `Juego`
- `Plataforma`
- `Linux`
- `MacOSX`
- `Windows`
- `TipoJuego`

The 'Ajedrez' dialog box also includes tabs for 'Class expression editor', 'Object restriction creator', 'Data restriction creator', and 'Class hierarchy'. The 'Class expression editor' tab is currently active.

# Agregando Axiomas a la Clase “Ajedrez”

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the ontology URL: <http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>. The breadcrumb trail indicates the current location: Juego > JuegoFamoso > Ajedrez.

The main workspace is divided into several panes. On the left, the 'Class hierarchy' pane shows a tree structure of classes. A red arrow points to the 'Ajedrez' class, which is a subclass of 'JuegoFamoso'. The hierarchy includes:

- owl:Thing
  - DificultadJuego
    - Facil
    - Normal
  - Juego
    - JuegoFamoso
      - Ajedrez**
      - LeagueOfLegends
      - Sudoku
  - Plataforma
    - Linux
    - MacOSX
    - Windows
  - TipoJuego
    - EnLinea
    - JuegoDeRoles
    - MultiJugador
    - Rompecabezas
    - UnSoloJugador

The right pane shows the 'Usage: Ajedrez' section, which lists the following uses of the class:

- Ajedrez **SubClassOf** JuegoFamoso
- Class: Ajedrez
- DisjointClasses: Ajedrez, LeagueOfLegends, Sudoku

Below the usage section, the 'Description: Ajedrez' pane is highlighted with a red box. It contains the following axioms:

- Equivalent To: (empty list)
- SubClass Of:
  - JuegoFamoso
  - tieneDificultad **some** Normal
  - tienePlataforma **some** Linux
  - tienePlataforma **some** MacOSX
  - tienePlataforma **some** Windows
  - tieneTipo **some** MultiJugador
  - tieneTipo **some** UnSoloJugador
- General class axioms: (empty list)
- SubClass Of (Anonymous Ancestor): (empty list)
- Instances: (empty list)

The bottom status bar indicates: To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences.

# Mejorando la definición de la Clase “DificultadJuego”

---

Qué se requiere hacer?

- agregar  $DificultadJuego \equiv Dificil \sqcup Normal \sqcup Facil$

Note que las clases *Dificil*, *Normal* y *Facil* son ya disjuntas

- agregar dominio y rango de la propiedad *tieneDificultad*

# Mejorando la definición de la Clase “Ajedrez”

---

## Qué se requiere hacer?

- **Ajedrez tiene dificultad normal de juego**
  - Replantear la especificación utilizando el vocabulario de ontología.  
Ajedrez tiene dificultad Normal
  - Escríbalo usando lógica descriptiva (opcional)  
 $Ajedrez \sqsubseteq \exists tieneDificultad.Normal$
  - Escríbalo usando sintaxis Manchester (el lado derecho es suficiente)  
tieneDificultad some Normal
- **Ajedrez es juego multijugador**
- **Ajedrez es un juego de un solo jugador**



# Mejorando la definición de la Clase “DificultadJuego”

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window displays the 'DificultadJuego' class hierarchy and its description.

**Class hierarchy: DificultadJuego**

- owl:Thing
  - DificultadJuego**
    - Difícil
    - Facil
    - Normal
  - Juego
    - JuegoFamoso
      - Ajedrez
      - LeagueOfLegends
      - Sudoku
    - Plataforma
      - Linux
      - MacOSX
      - Windows
    - TipoJuego
      - EnLinea
      - JuegoDeRoles
      - MultiJugador
      - Rompecabezas
      - UnSoloJugador

**Usage: DificultadJuego**

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 18 uses of DificultadJuego

- Difícil
  - Difícil SubClassOf DificultadJuego
- DificultadJuego
  - DificultadJuego SubClassOf owl:Thing

**Description: DificultadJuego**

Equivalent To ☒ **owl:Thing**

SubClass Of ☒ **owl:Thing**

General class axioms ☒

SubClass Of (Anonymous Ancestor)

Instances ☒

Target for Key ☒

Disjoint With ☒ **TipoJuego, Plataforma, Juego**

Disjoint Union Of ☒

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

# Mejorando la definición de la Clase “DificultadJuego”

The image shows the Protégé ontology editor interface. The main window displays the class hierarchy for 'DificultadJuego'. A red arrow points to the 'DificultadJuego' class in the hierarchy. The 'DificultadJuego' class is highlighted, and its description is shown in the 'Description: DificultadJuego' panel. The description includes the following axioms:

- Equivalent To: **+** (indicated by a red arrow)
- SubClass Of: **+** (indicated by a red arrow)
- owl:Thing
- General class axioms: **+**
- SubClass Of (Anonymous Ancestor): **+**
- Instances: **+**
- Target for Key: **+**
- Disjoint With: **+**
- TipoJuego, Plataforma, Juego
- Disjoint Union Of: **+**

The 'DificultadJuego' class is defined as a disjoint union of 'Difícil', 'Normal', and 'Facil'. The 'DificultadJuego' class is also a subclass of 'owl:Thing'.

The 'DificultadJuego' class is defined as a disjoint union of 'Difícil', 'Normal', and 'Facil'. The 'DificultadJuego' class is also a subclass of 'owl:Thing'.

The 'DificultadJuego' class is defined as a disjoint union of 'Difícil', 'Normal', and 'Facil'. The 'DificultadJuego' class is also a subclass of 'owl:Thing'.

# Mejorando la definición de la Clase “DificultadJuego”

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the ontology URL: <http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>. The left pane shows the class hierarchy, with 'DificultadJuego' selected and highlighted by a red arrow. The right pane shows the 'Usage' tab for 'DificultadJuego', displaying 18 uses of the class. The 'Description' tab for 'DificultadJuego' is also visible, showing the class definition.

**Class hierarchy:**

- owl:Thing
  - DificultadJuego**
    - Difícil
    - Facil
    - Normal
  - Juego
    - JuegoFamoso
      - Ajedrez
      - LeagueOfLegends
      - Sudoku
    - Plataforma
      - Linux
      - MacOSX
      - Windows
    - TipoJuego
      - EnLinea
      - JuegoDeRoles
      - MultiJugador
      - Rompecabezas
      - UnSoloJugador

**Usage: DificultadJuego**

Found 18 uses of DificultadJuego

- Difícil
  - Difícil SubClassOf DificultadJuego
- DificultadJuego
  - DificultadJuego SubClassOf owl:Thing

**Description: DificultadJuego**

Equivalent To: Difícil or Normal or Facil

SubClass Of: owl:Thing

General class axioms

SubClass Of (Anonymous Ancestor)

Instances

Target for Key

Disjoint With: TipoJuego, Plataforma, Juego

# Mejorando la definición de la Clase “DificultadJuego”

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the ontology URI: <http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>. The left sidebar displays the ontology hierarchy under 'owl:topObjectProperty', with 'tieneDificultad' selected. A red arrow points to this selection. The main panel shows the 'Usage: tieneDificultad' tab, indicating 10 uses of the property. Below this, the 'Description: tieneDificultad' tab is active, showing the 'Characteristics' section with 'Functional' checked. A red arrow points to the 'Functional' checkbox. The 'Description' section shows the property is 'Equivalent To' (empty), 'SubProperty Of' (empty), 'Inverse Of' (empty), 'Domains (intersection)' (Juego), and 'Ranges (intersection)' (DificultadJuego). Red arrows point to the 'Domains (intersection)' and 'Ranges (intersection)' sections. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences'.

untitled-ontology-20 (<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>) : [<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>) Search...

tieneDificultad

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Object property hierarchy: tieneDificultad

Annotations Usage

Usage: tieneDificultad

Show: ☒ this ☒ disjoints

Found 10 uses of tieneDificultad

- Ajedrez
- Ajedrez SubClassOf tieneDificultad some Normal

- tieneDificultad
  - tieneDificultad Range DificultadJuego
  - ObjectProperty: tieneDificultad
  - Functional: tieneDificultad

Characteristics: t ☒ Functional ☐ Inverse functional ☐ Transitive ☐ Symmetric ☐ Asymmetric ☐ Reflexive ☐ Irreflexive

Description: tieneDificultad

Equivalent To +

SubProperty Of +

Inverse Of +

Domains (intersection) +

- Juego

Ranges (intersection) +

- DificultadJuego

Disjoint With +

SuperProperty Of (Chain) +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

# Ontología sobre Juegos de Video

Supongamos que queremos construir una ontología sobre videojuegos de la siguiente manera.

| Clases  | Clases  | Propiedades   | definiciones   |
|---|---|---|--|
| <ul style="list-style-type: none"><li>- Juego<ul style="list-style-type: none"><li>- JuegoFamoso<ul style="list-style-type: none"><li>- LoL</li><li>- Ajedrez</li><li>- Sudoku</li></ul></li></ul></li><li>- Plataforma<ul style="list-style-type: none"><li>- Windows</li><li>- MacOSX</li><li>- Linux</li></ul></li></ul> | <ul style="list-style-type: none"><li>- TipoJuego<ul style="list-style-type: none"><li>- UnSoloJugador</li><li>- MultiJugador</li><li>- DeRoles</li><li>- Enlinea</li></ul></li><li>- DificultadJuego<ul style="list-style-type: none"><li>- Difícil</li><li>- Normal</li><li>- Fácil</li></ul></li></ul> | <ul style="list-style-type: none"><li>tieneDificultad</li><li>tienePlataforma</li><li>tieneTipo</li></ul> | <ul style="list-style-type: none"><li>JuegoMultiPlataforma</li><li>JuegoDifícil</li><li>JuegoNormal</li><li>JuegoFacil</li><li>JuegoParaLinux</li><li>JuegoParaWindows</li><li>JuegoParaMacOSX</li><li>JuegoMultiJugador</li><li>...</li></ul> |

# Agregando la definición de la Clase “JuegoMultiJugador”

---

Qué se requiere hacer?

- agregar la clase *JuegoMultijugador*
- *la cual es una subclase de Juego y que tiene como tipo de juego múltiples jugadores*
- *$JuegoMultiJugador \equiv Juego \sqcap \exists tieneTipo.MultiJugador$*

# Agregando la Clase definible “JuegoMultiJugador”

The screenshot displays the Protégé ontology editor interface for an ontology named 'untitled-ontology-20'. The left pane shows the 'Class hierarchy' with a tree structure. A red arrow points to the 'JuegoMultiJugador' class, which is highlighted in blue. The right pane shows the 'Usage' tab for 'JuegoMultiJugador', indicating it is a subclass of 'Juego'. Below this, the 'Description: JuegoMultiJugador' pane is highlighted with a red border. It shows the class is a 'SubClass Of' 'Juego' and has the property 'tieneTipo some MultiJugador'. A red arrow points to this property. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences' is checked.

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) : [http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) Search...

Juego > JuegoMultiJugador

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: JuegoMultiJugador Usage: JuegoMultiJugador

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 4 uses of JuegoMultiJugador

- JuegoMultiJugador SubClassOf Juego
- Class: JuegoMultiJugador

Description: JuegoMultiJugador

Equivalent To +

SubClass Of +

- Juego
- tieneTipo some MultiJugador

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

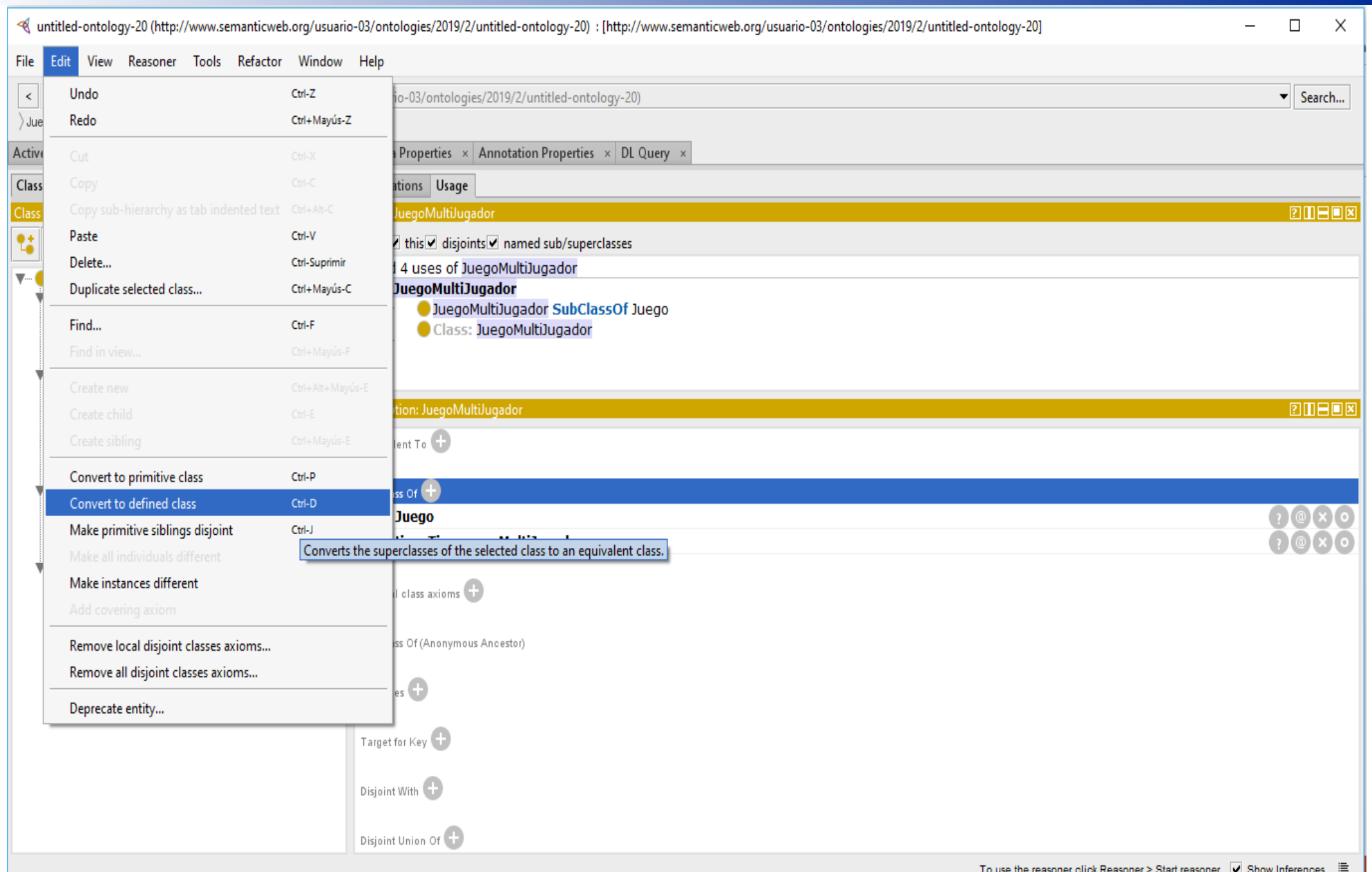
Target for Key +

Disjoint With +

Disjoint Union Of +

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

# Agregando la Clase definible “JuegoMultiJugador”





# Agregando la Clase definible “JuegoMultiJugador”

The screenshot shows the Protégé ontology editor interface. The left pane displays the class hierarchy, with 'JuegoMultiJugador' highlighted under the 'Juego' class. A red arrow points to this class in the hierarchy. The right pane shows the 'Usage' tab for 'JuegoMultiJugador', indicating it is a subclass of 'Juego'. Below this, the 'Description' tab for 'JuegoMultiJugador' is shown, containing the equivalent class axiom:  $\text{Juego} \text{ and } (\text{tieneTipo some MultiJugador})$ . A red arrow points to this axiom. To the right of the description pane, the logical expression  $\text{JuegoMultiJugador} \equiv \text{Juego} \sqcap \exists \text{tieneTipo. MultiJugador}$  is written in red. The bottom status bar indicates the reasoner is active and inferences are shown.

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) : [http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)

Juego > JuegoMultiJugador

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: JuegoMultiJugador Usage: JuegoMultiJugador

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 4 uses of JuegoMultiJugador

Usage: JuegoMultiJugador

- JuegoMultiJugador SubClassOf Juego
- Class: JuegoMultiJugador

Description: JuegoMultiJugador

Equivalent To  $\text{Juego} \text{ and } (\text{tieneTipo some MultiJugador})$

SubClass Of

General class axioms  $\text{JuegoMultiJugador} \equiv \text{Juego} \sqcap \exists \text{tieneTipo. MultiJugador}$

SubClass Of (Anonymous Ancestor)

Instances

Target for Key

Disjoint With

Disjoint Union Of

To use the reasoner click Reasoner > Start reasoner ☒ Show Inferences

# Agregar más definiciones

---

- Agregue las definiciones de las siguientes clases:
- JuegoParaLinux
- JuegoParaMacOSx
- JuegoParaWindows
- JuegoMultiPlataforma
- JuegoDeRoles
- JuegoUnSoloJugador
- JuegoEnLinea
- JuegoDifícil
- JuegoFácil
- JuegoNormal

---

**Fin Tutorial – OWL**  
**Creación de Clases Primitivas,**  
**Definidas y Restricciones de**  
**Propiedades**

---

# **Inicio Tutorial – OWL Inferencia**

# Ontología sobre Juegos de Video

Supongamos que queremos construir una ontología sobre videojuegos de la siguiente manera.

| Clases  | Clases   | Propiedades   | definiciones   |
|---|--|---|--|
| <ul style="list-style-type: none"><li>- Juego<ul style="list-style-type: none"><li>- JuegoFamoso<ul style="list-style-type: none"><li>- LoL</li><li>- Ajedrez</li><li>- Sudoku</li></ul></li></ul></li><li>- Plataforma<ul style="list-style-type: none"><li>- Windows</li><li>- MacOSX</li><li>- Linux</li></ul></li></ul> | <ul style="list-style-type: none"><li>- TipoJuego<ul style="list-style-type: none"><li>- UnSoloJugador</li><li>- MultiJugador</li><li>- DeRoles</li><li>- Elinea</li></ul></li><li>- DificultadJuego<ul style="list-style-type: none"><li>- Difícil</li><li>- Normal</li><li>- Fácil</li></ul></li></ul> | <ul style="list-style-type: none"><li>tieneDificultad</li><li>tienePlataforma</li><li>tieneTipo</li></ul> | <ul style="list-style-type: none"><li>JuegoMultiPlataforma</li><li>JuegoDifícil</li><li>JuegoNormal</li><li>JuegoFacil</li><li>JuegoParaLinux</li><li>JuegoParaWindows</li><li>JuegoParaMacOSX</li><li>...</li></ul> |

# Razonador

---

Protégé puede ser usado para tareas de razonamiento como la clasificación

- configure el razonador

Reasoner → Configure... (para este tutorial, seleccione todo bajo *Class inferences* y *Object property inferences*)

- seleccione un razonador

por ejemplo, Reasoner → HermiT (otros razonadores pueden ser agregados, cuál utilizar depende de varios factores, como la expresividad de la ontología)

- finalmente, Reasoner → Start reasoner

# Ejemplo Razonamiento

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the ontology URL: <http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>. The main toolbar contains icons for navigating between different views: Class hierarchy, Class hierarchy (inferred), Annotations, and Usage. The 'Class hierarchy' tab is active, showing a tree structure of classes. A red arrow points to the 'JuegoParaLinux' class in the hierarchy. The 'Annotations' tab is also visible, showing the description of the 'JuegoParaLinux' class: **Juego and (tienePlataforma some Linux)**. The 'Reasoner' tab is active, showing the state of the ontology. The status bar at the bottom indicates 'Reasoner state out of sync with active ontology' and 'Show Inferences'.

untitled-ontology-20 (<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>) : [<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (<http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>) Search...

Juego > JuegoParaLinux

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: JuegoParaLinux

Annotations: JuegoParaLinux

Annotations +

Description: JuegoParaLinux

Equivalent To +

Juego and (tienePlataforma some Linux)

SubClass Of +

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

Disjoint Union Of +

Reasoner state out of sync with active ontology ☒ Show Inferences

# Ejemplo Razonamiento

The screenshot displays a Semantic Web editor interface with the following components:

- Browser Address Bar:** `untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)`
- Menu Bar:** File, Edit, View, Reasoner, Tools, Refactor, Window, Help
- Tab Bar:** untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)
- Breadcrumb:** > Juego > JuegoFamoso > Ajedrez
- Active Ontology:** untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)
- Tab Bar (Left):** Active Ontology, Entities, Classes, Object Properties, Data Properties, Annotation Properties, DL Query
- Class hierarchy:** Class hierarchy (inferred)
- Class hierarchy: Ajedrez:** A tree view showing the hierarchy of classes. The 'Ajedrez' class is highlighted. A red arrow points to the 'Asserted' tab.
- Annotations: Ajedrez:** A table showing annotations for the 'Ajedrez' class. The table has columns for the property, the value, and a set of icons. The values are: `tieneDificultad some Normal`, `tienePlataforma some Linux`, `tienePlataforma some MacOSX`, `tienePlataforma some Windows`, `tieneTipo some MultiJugador`, and `tieneTipo some UnSoloJugador`. The word 'Inferido' is written in red above this table, with a red arrow pointing to it.
- Description: Ajedrez:** A table showing the description of the 'Ajedrez' class. The table has columns for the property, the value, and a set of icons. The values are: `JuegoMultiJugador`, `JuegoMultiplataforma`, `JuegoNormal`, and `JuegoUnSoloJugador`. The word 'Inferido' is written in red above this table, with a red arrow pointing to it.
- General class axioms:** A table showing general class axioms. The table has columns for the axiom, the value, and a set of icons. The axioms are: `Juego and (tieneTipo some MultiJugador)`, `Juego and (tieneDificultad some Normal)`, `Juego and (tieneTipo some UnSoloJugador)`, and `Juego and (tienePlataforma some Linux) and (tienePlataforma some MacOSX) and (tienePlataforma some Windows)`. The word 'Inferido' is written in blue above this table, with a red arrow pointing to it.
- SubClass Of (Anonymous Ancestor):** A table showing sub-classes of the 'Ajedrez' class. The table has columns for the sub-class, the value, and a set of icons. The sub-classes are: `JuegoMultiJugador`, `JuegoMultiplataforma`, `JuegoNormal`, and `JuegoUnSoloJugador`.
- Instances:** A table showing instances of the 'Ajedrez' class. The table has columns for the instance, the value, and a set of icons. The instances are: `LeagueOfLegends` and `Sudoku`.



# Explicar las clases inferidas

---

Revise y verifique que las clases hayan sido clasificadas correctamente

Qué fue inferido para la Clase Ajedrez?

---

# **Fin Tutorial – OWL Inferencia**

# OWL 2 RESUMEN

## Class Expressions

- Class names  $A, B$
- Conjunction  $C \sqcap D$
- Disjunction  $C \sqcup D$
- Negation  $\neg C$
- Exist. property restriction  $\exists R.C$
- Univ. property restriction  $\forall R.C$
- Self  $\exists S.\text{Self}$
- Greater-than  $\geq n \ S.C$
- Less-than  $\leq n \ S.C$
- Enumerated classes  $\{a\}$

## Properties

- Property names  $R, S, T$
- Simple properties  $S, T$
- Inverse properties  $R^{-}$
- Universal property  $U$

## Tbox (Class axioms)

- Inclusion  $C \sqsubseteq D$
- Equivalence  $C \equiv D$

## Rbox (Property Axioms)

- Inclusion  $R_1 \sqsubseteq R_2$
- General Inclusion  $R^{(-)}_1 \circ R^{(-)}_2 \circ \dots \circ R^{(-)}_n \sqsubseteq R$
- Transitivity
- Symmetry
- Reflexivity
- Irreflexivity
- Disjointness

## Abox (Facts)

- Class membership  $C(a)$
- Property relation  $R(a, b)$
- Negated property relation  $\neg S(a, b)$
- Equality  $a=b$
- Inequality  $a \neq b$



---

# **Práctica OWL**

## **Creación ontología en**

### **Protege**