

Open Web Ontobud



User Manual

This user manual describes all features present in the Web app *Open Web Ontobud*, an Open Source RDF4J Frontend to manipulate and query RDF Graphs.

Here the user can find help and the steps to perform any desired action.

This project was conducted at the University of Minho, as the Master's Thesis in Informatics Engineering (Integrated Master's) for the student Francisco José Moreira Oliveira A78416, under supervision of professor José Carlos Ramalho.

Conteúdo

1	User actions	3
1.1	Topbar / Sidebar	3
1.1.1	Create account	4
1.1.2	Login/Logout	4
1.1.3	Change current Repository	5
1.1.4	Minor settings and links	5
1.1.5	Page navigation	6
1.2	Homepage	7
1.3	Manage Repositories	7
1.3.1	Create Repository	7
1.3.2	Delete Repository	9
1.4	Manage Repository	9
1.4.1	Import Repository	9
1.4.2	Export Repository	10
1.4.3	Clear/Delete Repository	11
1.5	Repository Info	11
1.5.1	Statements	11
1.5.2	Namespaces	12
1.5.3	Classes	13
1.6	SPARQL	14
1.6.1	SPARQL Help	14
1.6.2	Saved Queries	15
1.6.3	Query Editor	16
1.6.4	Save Query	18
1.6.5	Results Table	19
1.7	Resource	20
1.8	Navigation	20
1.9	Feedback	20

1 User actions

This section explains all the actions that can be carried by a normal user.

1.1 Topbar / Sidebar

The Topbar and Sidebar, as the name implies, are positioned on the top and side (left) of the page, respectively. They allow the user quick access to multiple pages and other functionalities.



Figura 1: Topbar

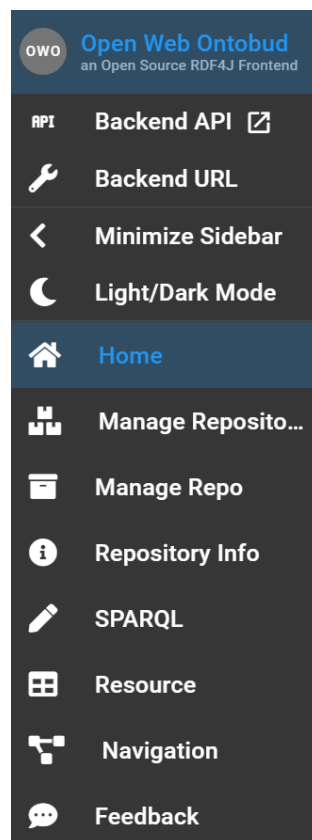


Figura 2: Sidebar

1.1.1 Create account

To create an account the user must click the button shown in the Figure 3 below.



Figura 3: Create account button

Then the account creation form will appear (see Figure 4).

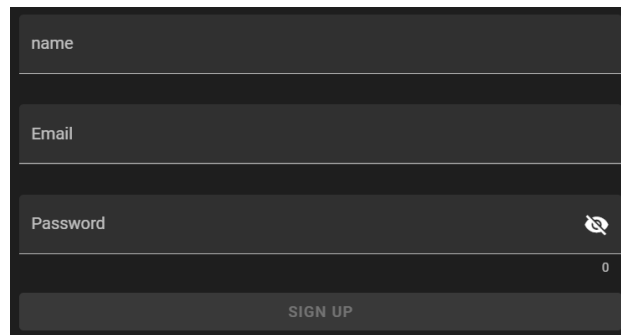
A screenshot of an account creation form. It features three input fields: 'name', 'Email', and 'Password'. The 'Password' field has a toggle icon (an eye with a slash) to its right. Below the fields is a 'SIGN UP' button. The form is set against a dark background.

Figura 4: Sign In form

After completing the form, and if all fields are filled correctly, the "Sign Up" button will turn blue. Pressing this button will then create the account and automatically log in with the new user account.

1.1.2 Login/Logout

If the user doesn't have an account yet, please look at Section 1.1.1 to see how to create an account.

If the user already has an account, then it can login pressing the "Login" button (shown below in Figure 5).



Figura 5: Login button

The user must then enter its credentials (email and password).

A dark-themed login form. It has two input fields: "Email" and "Password". The "Password" field has a toggle icon (an eye) on its right side. Below the fields is a "LOGIN" button.

Figura 6: Login form

The "Login" button will be blue when all fields are inserted. Login success will close the form. Login failure will show a failure message and keep the form open.

To logout just press the button shown below in Figure 7.



Figura 7: Logout button

1.1.3 Change current Repository

To change the current repository, simply click the combobox shown below in Figure 8), and select the repository you desire to use from the list shown.



Figura 8: Change Repository

1.1.4 Minor settings and links

The links grant fast access to information and allow to quickly change some settings.

- **User Manual** - This is the button used to open this manual OwO
- **Backend API** - Backend documentation. Useful for more advanced users, or for integration in other platforms.

- **Backend URL** - Allows to see the currently connected backend and change it if desired. Will accept invalid hosts, and as such be careful. If no repositories are detected or queries fail, verify if the backend URL was inserted correctly.
- **Minimize Sidebar** - Minimize Sidebar to save space in smaller screen devices.
- **Light/Dark Mode** - Change the page between light and dark (default) theme.

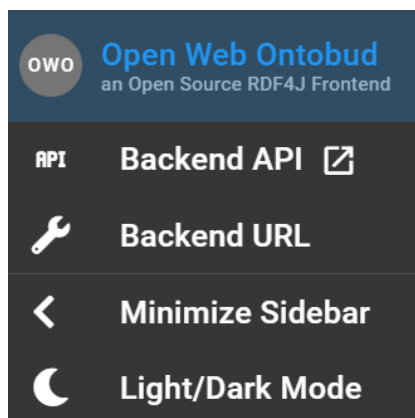


Figura 9: Sidebar - Settings and Links

1.1.5 Page navigation

This area (see Figure 10) contains all the current pages the user can access. Each page allows the user to perform or access different tools, and they will be explained in further detail later in the manual.

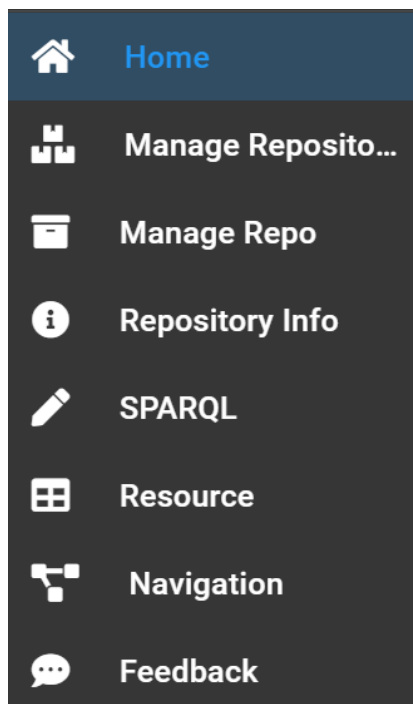


Figura 10: Sidebar - Pages

1.2 Homepage

TODO later

1.3 Manage Repositories

This page is accessible through the sidebar. It allows us to create and delete repositories, each explained below.

1.3.1 Create Repository

To create a new repository the following form must be completed.

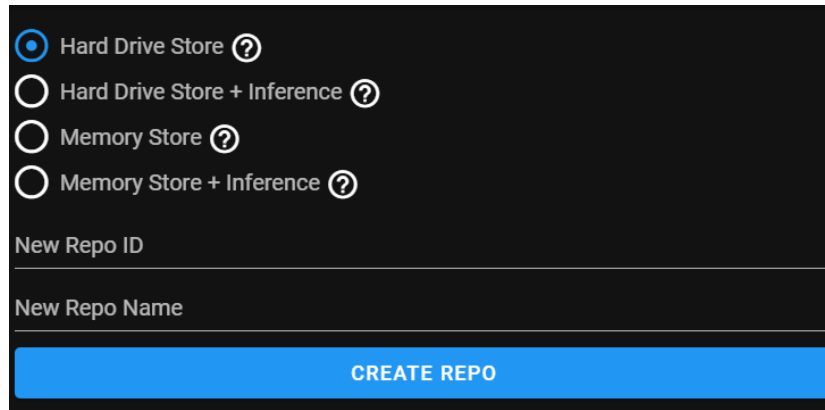
A screenshot of a 'Create Repository' form. It features four radio buttons for selecting the repository type: 'Hard Drive Store' (selected), 'Hard Drive Store + Inference', 'Memory Store', and 'Memory Store + Inference'. Each option has a help icon (question mark). Below the radio buttons are two text input fields labeled 'New Repo ID' and 'New Repo Name'. At the bottom is a large blue button labeled 'CREATE REPO'.

Figura 11: Create Repository

The four radio buttons serve to select the repository type. Here is a brief explanation on the possible memory types and inferences:

- **Hard Drive Store** - Will use hard drive memory, which is abundant, but will be slightly slower than a Memory store.
- **Memory Store** - Will use RAM memory, which is faster compared to Hard drive memory, but is limited so this should be used only for special cases.
- **Inference** - RDFS and Direct Type inference will be used. This allows your repository to infer additional information from the existing knowledge. Will reduce query speed in a varying degree depending on the repository size.
- **No Inference** - If no inference is specified, then it won't be used. Advised for larger repositories to increase performance if inference is not relevant.

“New Repo ID” will be the unique identifier for the repository. No two repositories can have the same identifier. “New Repo Name” will be the repository name. This will be most often used when displaying the current repository.

After completing the form, the user only needs to press the “Create Repo” button and the repository will be created. If the creation fails, the user will receive an alert informing why it failed.

1.3.2 Delete Repository

To delete a repository, simply type the Repository ID into “Repo ID”, and press the “Delete Repo” button. A deleted repository cannot be recovered.

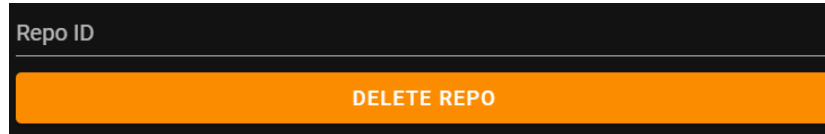
A dark-themed form for deleting a repository. It features a text input field labeled "Repo ID" at the top. Below the input field is a prominent orange button with the text "DELETE REPO" in white capital letters.

Figura 12: Delete Repository

1.4 Manage Repository

1.4.1 Import Repository

A repository can be imported from a file, or from raw text. For a file import, the user must first select the file type (left radio buttons), and whether he wants to replace or add to the ontology (right radio buttons). Finally, the user must select the file and press the "Import Repo (File)" button. Files above 100MB will take some time to import, especially if inference is activated.

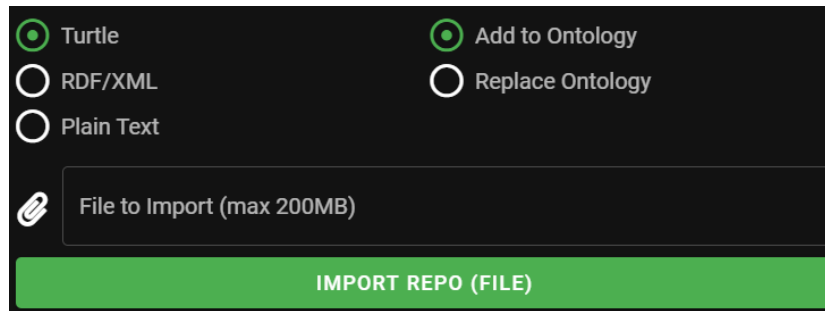
A dark-themed form for importing a repository from a file. On the left, there are three radio buttons for file types: "Turtle" (selected), "RDF/XML", and "Plain Text". On the right, there are two radio buttons for ontology actions: "Add to Ontology" (selected) and "Replace Ontology". Below these is a file selection area with a paperclip icon and the text "File to Import (max 200MB)". At the bottom is a large green button with the text "IMPORT REPO (FILE)" in white capital letters.

Figura 13: Import Repository (File)

For a raw text import, the user only needs to copy and paste the raw text for the ontology it wants to import, and then press the "Import Repo (Input Text)" button. The page might freeze when pasting a large size of information. Because of this, for large amounts of data it is advised to import using a file.

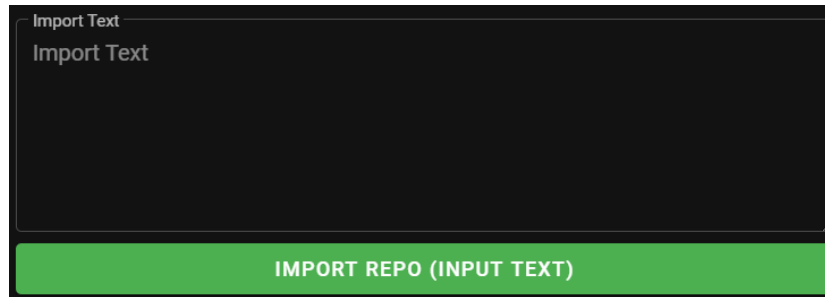
The interface for importing a repository from text. It features a dark gray rectangular area with a thin border, containing the text "Import Text" at the top left. Below this area is a solid green button with the text "IMPORT REPO (INPUT TEXT)" in white, uppercase letters.

Figura 14: Import Repository (Text)

1.4.2 Export Repository

A repository can be exported into a file, or to raw text. For a file export, the user must first select the output file type (radio buttons), and whether he wants inferred triples included in the export. Then, press the "Export Repo (Download File)" button. The file will be automatically downloaded as "exportRepository".

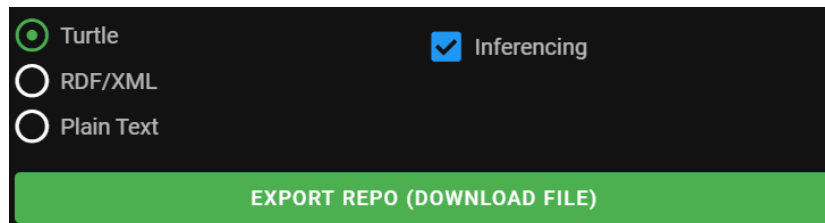
The interface for exporting a repository to a file. It shows three radio button options: "Turtle" (selected with a green dot), "RDF/XML", and "Plain Text". To the right, there is a checked checkbox labeled "Inferencing". Below these options is a solid green button with the text "EXPORT REPO (DOWNLOAD FILE)" in white, uppercase letters.

Figura 15: Export Repository (File)

For a raw text export the user only needs to press the "Export Repo (In-screen Text)" button. The export will appear in the textbox. The page might freeze for large repositories due to the large size of information. Because of this, for large amounts of data it is advised to export to file.

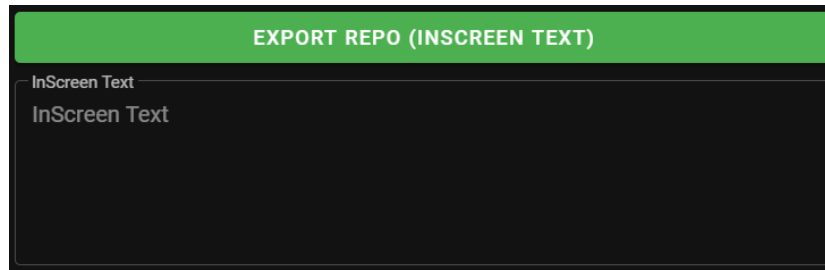


Figura 16: Export Repository (Text)

1.4.3 Clear/Delete Repository

"Clear All Repo Statements" will delete all triples from the current repository. This does not include used namespaces.

"Delete Current Repo" will firstly ask for confirmation, by typing the Repository ID, and then delete the entire repository.

Alerts will notify the user about the action success or failure for both cases.

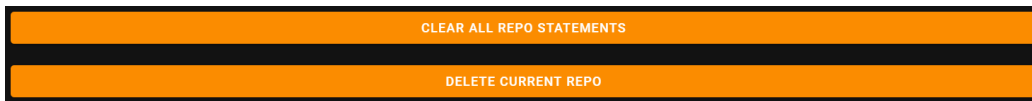


Figura 17: Clear/Delete Repository

1.5 Repository Info

This page provides an overview on the current repository, by presenting information such as number of statements, namespaces used, and classes created in this repository.

1.5.1 Statements

Provides the number of triples in the repository, and how many are explicit or implicit. This allows us to also calculate the expansion ratio, which can give us an idea on how much information is being gained through inference. When inference is disabled the implicit statements will be zero.

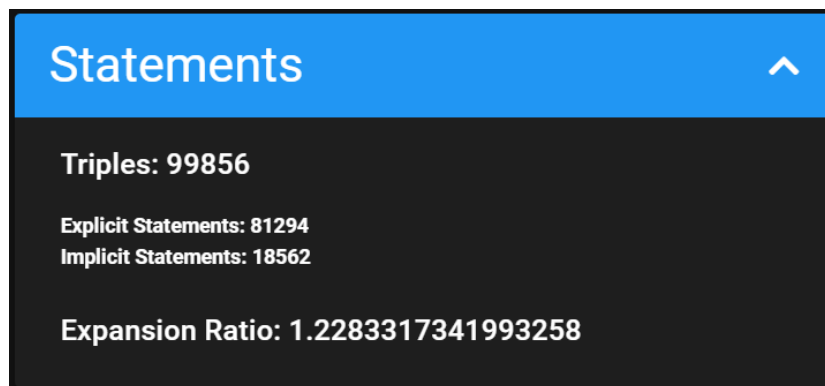


Figura 18: Statements

1.5.2 Namespaces

List of Namespaces used in this repository. Can be used to detect if the repository is a simple RDF graph or an Ontology. Also can give a broad idea of what other resources where used.

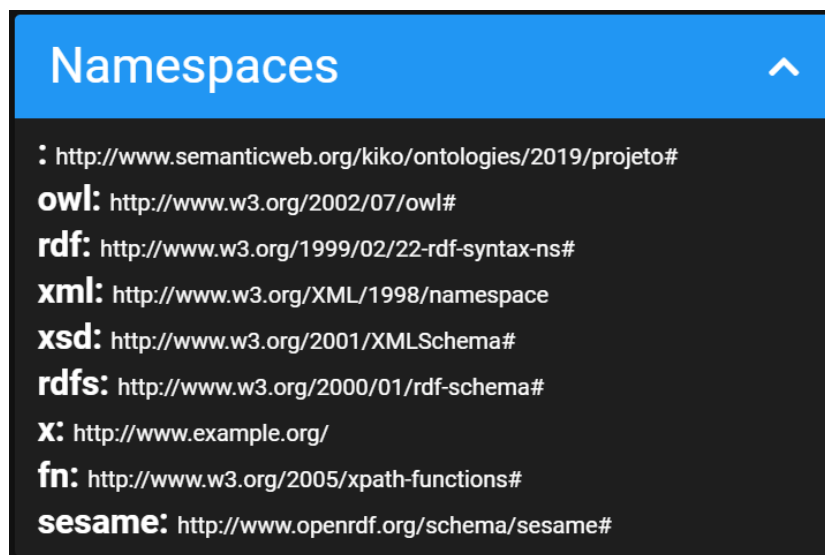
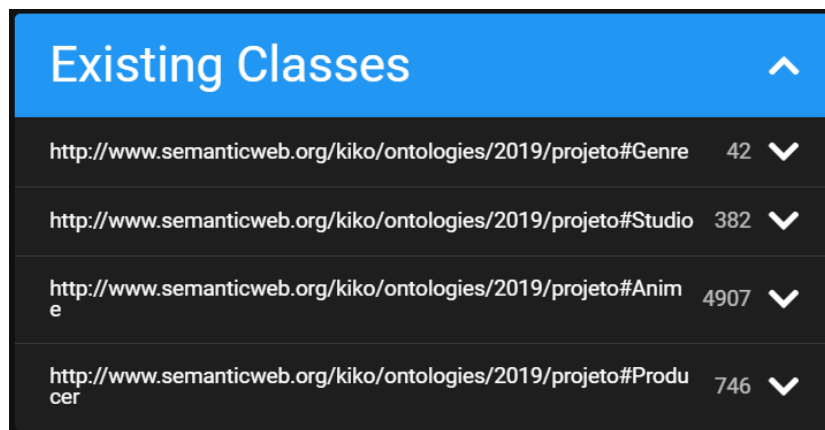


Figura 19: Namespaces

1.5.3 Classes

Here we have a list with the Classes created in this repository. This does not include classes imported from other namespaces. This allows us to quickly see the main focus of the repository by looking at its main classes. Each class shows its count on the right, giving an understanding on which classes are most used and their importance. In the Figure 20 we can see the class "Anime" has the most elements and is likely very important in this repository.



Existing Classes		^
http://www.semanticweb.org/kiko/ontologies/2019/projeto#Genre	42	▼
http://www.semanticweb.org/kiko/ontologies/2019/projeto#Studio	382	▼
http://www.semanticweb.org/kiko/ontologies/2019/projeto#Anime	4907	▼
http://www.semanticweb.org/kiko/ontologies/2019/projeto#Producer	746	▼

Figura 20: Classes

Each class can be clicked to expand its tab. In Figure 21 we can see the class "Studio" expanded. Expanding a class shows some elements that belong to that class. More can be seen by clicking "Show More". Each element can also be clicked to see information about that specific resource, using the Resource page that will be talked in Section 1.7.

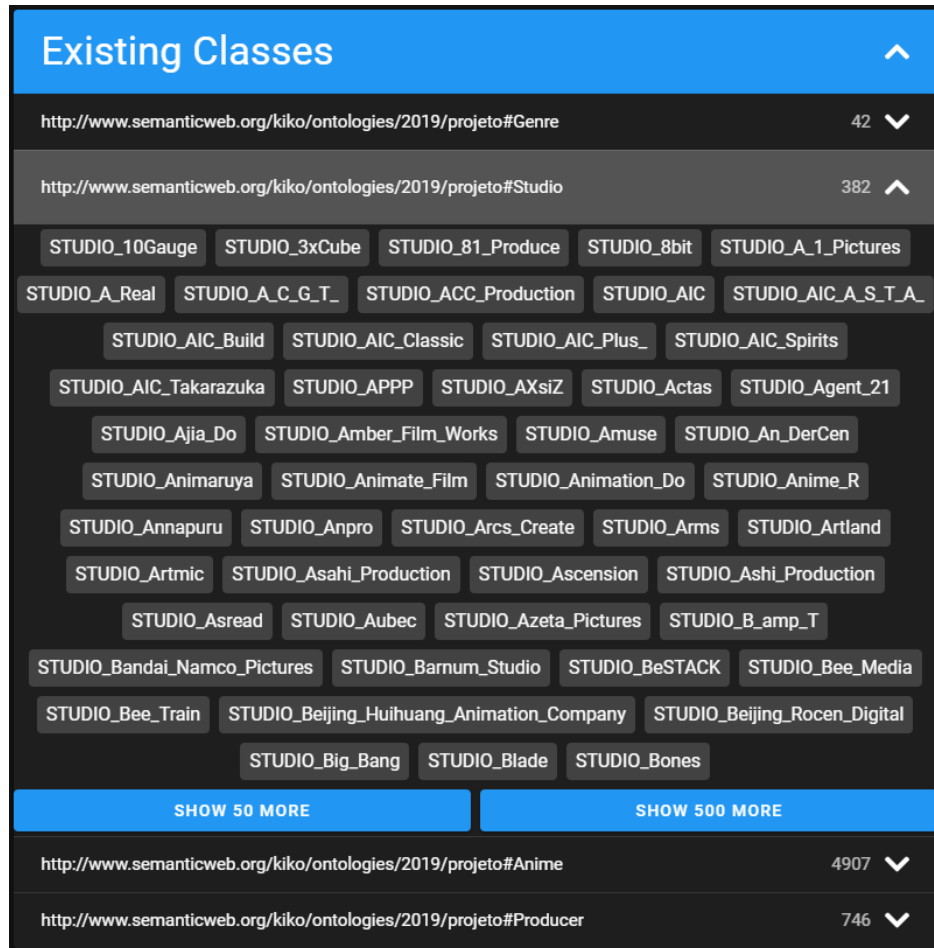


Figura 21: Class “Studio” Expanded

1.6 SPARQL

The SPARQL page allows the user to query the repository and see the results. It also allows the logged in user to save queries to reuse later.

1.6.1 SPARQL Help

This area contains links directing the user to information about SPARQL. The first link is the SPARQL 1.1 Specification by W3C, which contains vast and detailed information about SPARQL and should be capable to help the

user get a grasp and just about anything SPARQL can do. The second link takes the user to a video tutorial while the third link contains the same tutorial but as a text tutorial. Both serve to give a first introduction for users new to SPARQL, providing the most basic and important points on how to use this query language.

Note that this area can be shown/hidden by clicking the "Show SPARQL Help"/"Hide SPARQL Help" button, respectively. This allows the use to hide the information, saving space when not being used. Hidden by default.

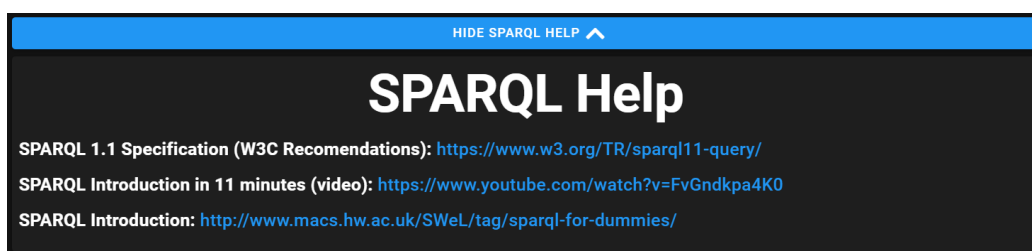


Figura 22: Sparql - Help

1.6.2 Saved Queries

Here we can see all the user saved queries. If the user is not logged in an alert will appear telling the user he must be logged in to see saved queries.

The saved queries list can also be shown/hidden by clicking the button. Shown by default.

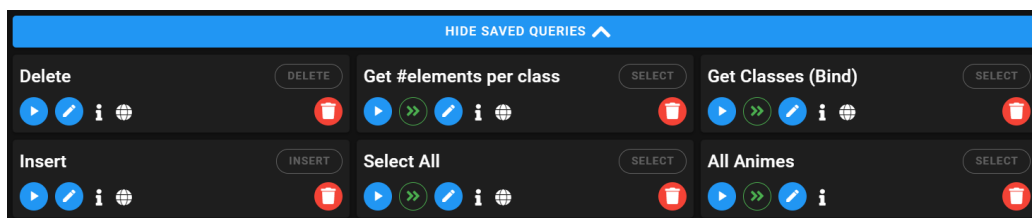


Figura 23: Saved Queries

Each saved query contains multiple buttons and information for the user to see and interact. Below we describe what each component is and does for the user.

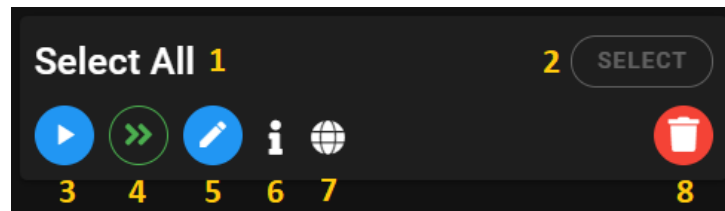


Figura 24: Saved Query - Component Labeling

1. [**Query name**] - This is the query name. Used to identify queries.
2. [**Query type**] - This is the query type. Useful to easily identify the query type when the name might not be obvious.
3. [**Run query**] - Runs saved query when clicks. Results are shown below in the results table (Section 1.6.5).
4. [**Inference toggle**] - Default ON. Selects whether the saved query returns inferred triples in the results when executed. This will not appear in SPARQL Update queries.
5. [**Edit query**] - Opens a floating panel where the user can edit the saved query. Query name cannot be changed, only the query itself can be changed.
6. [**Query info**] - Shows saved query when hovered. Useful to quickly check the query without opening the edit panel.
7. [**Global query**] - When the globe is present, it means that saved query is global and usable in any repository. If it is missing then the saved query is local and only usable for the current repository.
8. [**Delete query**] - Delete query button. Open a confirmation button when clicked. Clicking the confirmation button will delete the query permanently.

1.6.3 Query Editor

The query editor is where the user types and runs any custom queries. This is also where queries are saved.

Many buttons here have tooltips to make sure the user will not be confused about what each button does. Even so, below we identify each button and it does for the user.

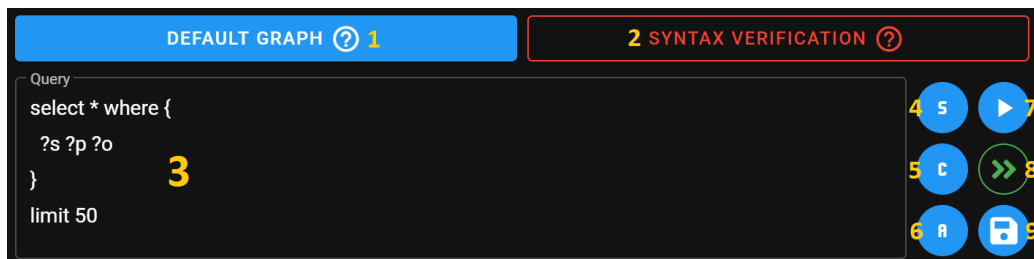


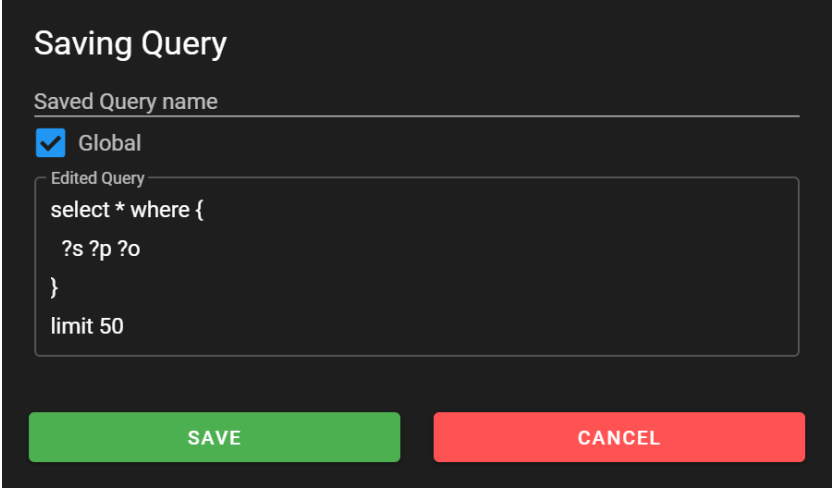
Figura 25: Query editor

1. **[Default Graph]** - This button will run a query that return all triples where the predicate is a "rdf:type" and the object belongs to the default namespace (was defined on this ontology). This returns mostly classes defined by the current ontology and how they relate to other resources imported from other namespaces.
2. **[Syntax verification]** - Default OFF. Toggles query syntax verification ON/OFF. Verification can aid less experienced users to detect possible errors in their queries even before running them. Warning: this feature has not been extensively tested and can have some bugs. There can be cases where the verification detects an error that does not exist. If you are experienced or believe your query is correct disable the verification and run the query normally.
3. **[Query editor]** - This is where the user will write the query to be executed.
4. **[Select query template]** - This button replaces the current query with a Select query template. Useful for beginners struggling to remember the query base structure.
5. **[Construct query template]** - This button replaces the current query with a Construct query template. Useful for beginners struggling to remember the query base structure.

6. **[Ask query template]** - This button replaces the current query with a Ask query template. Useful for beginners struggling to remember the query base structure.
7. **[Run query]** - This runs the query. Results are shown below in the results table (Section 1.6.5). If syntax verification is ON, the run button will be disabled if the user enters an invalid query.
8. **[Inference toggle]** - Default ON. Toggles whether results contain inferred triples or not.
9. **[Save query]** - This button opens a floating panel where the user can save a query. More detailed information on Section 1.6.4). If syntax verification is ON, the save button will be disabled if the user enters an invalid query.

1.6.4 Save Query

After pressing the save query button near the query editor, the following floating panel will open.



Saving Query

Saved Query name _____

☒ Global

Edited Query

```
select * where {  
  ?s ?p ?o  
}  
limit 50
```

SAVE **CANCEL**

Figura 26: Save Query

Firstly the user must give the query a name. This will be displayed later in the saved query list and can give an idea of what the query does by choosing

a good name. Secondly the user must tell whether the query is global or not. Global queries are usable in any repository while non global queries are only usable in the repository they were created on. Generic queries will often be global queries, while queries that look into specific classes or attributes tend to be non global. Finally the user must enter the query to be saved. This can be later edited if any mistakes happen. It is advised to preemptively test any queries the user wishes to save, as no verification is made before saving it, meaning an invalid query can be saved.

1.6.5 Results Table

Results table shows the results returned from executed queries.

Results can be filtered with a text search which can be case sensitive or not. The text search is executed over the values shown in the table. These values can be changed slightly by toggling ON/OFF the namespace and/or prefix. With this the namespace can be replaced by the shorter prefix ("use prefix" ON), or removed entirely ("Show namespaces" OFF).

Warning: Currently sometimes the results will not correctly follow the current toggle restrictions (ex: "Show namespaces" OFF, and namespaces appear in the table). This is a small startup problem usually caused by a big query result and the browser not handling the full load correctly. When this happens simply toggle whatever is failing twice and it will be fixed.

Results exportation supports multiple file types. Just select the desired export file extension and press "Export results". A file will be download with the "exportQueryResults" + the desired file extension.

Finally some elements in the results will be underlined. This are elements associated to an URI (Universal Resource Identifier). Clicking an underlined element will take the user to its resource page (1.7) where information related to that specific resource can be more easily accessed. This also gives us quick access to the navigation page to visualize the graph centered on that resource.

s	p	o
:Anime	rdf:type	owl:Class
:Genre	rdf:type	owl:Class
:Producer	rdf:type	owl:Class
:Studio	rdf:type	owl:Class

Figura 27: Results table

1.7 Resource

1.8 Navigation

TODO later

1.9 Feedback

Here the user can provide feedback to help improve the platform. Sliders allow for fast and effortless feedback, and the textbox can be used to describe ideas or bugs found in text.

Lastly feedback can also be sent by email, for those who desire greater connection with the development team.