OSA ATLAS Web2.0 Interface

Notes on use and demonstration

Monday, 3 September 2012

Stelios Voutsinas

- **Background to Project**

  The initial scope was to build a python web-based tool that allows astronomers to send queries via VO infrastructure (TAP, ADQL, VOTables etc) and view results in table as well as other formats (graphs, scatter plots, histograms, density plots) etc. A major focus was to take advantage and make use of Web2.0 architecture and technologies, so all of what I develop has an interactive, dynamic element to it, which involves alot of client side scripting (using Javascript, AJAX, CSS etc). Overall the idea of this project is to provide a new/interactive web based medium for astronomers to run different queries and allow them to view/filter/transfer the results easily with some neat functionality in the process. (Web-based data exploration could be a tag for it)

  Following this we decided to implement this tool as part of a new OSA Atlas interface, that would maintain the feel and layout of the other WFAU surveys, but would provide new features and a new (Web2.0) look and feel, adding some new features and interactive elements to the survey pages, such as SAMP for transferring results of queries to other applications.

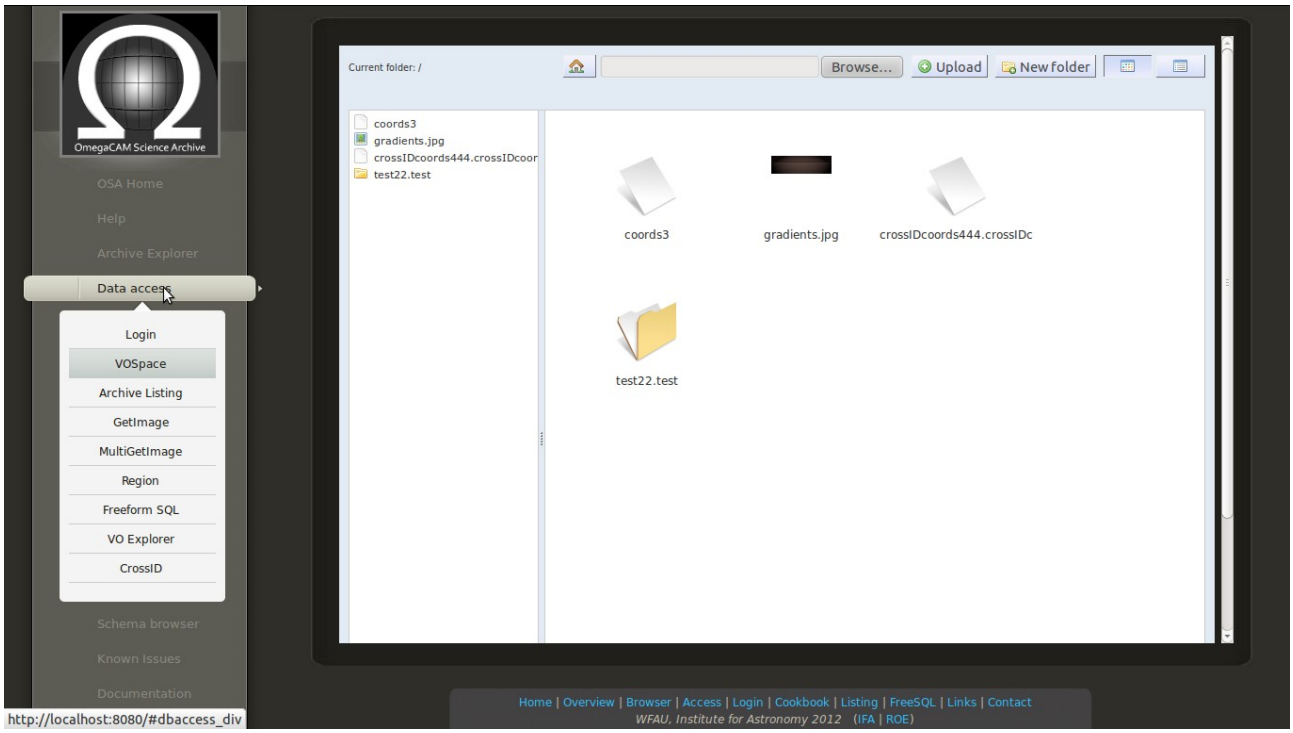
- **Demo Outline and Notes on use**

  Step 1 [General Overview of interface].

  Show main page interface, describe how everything is contained within the main sub-frame, with content being dynamically loaded (without requiring page refreshes) and show javascript based drop-down menu, perhaps describing the different functionality provided by the interface through the menu entries.

  Step 2 [VOSpace file management system]

  The Data Access/VOSpace link will take the user to the *VOSpace file management system* which will allow clients to use the WFAU VOSpace implementation via an interactive Javascript/AJAX page. This back-end code for this is still in progress, however as the front-end is complete this can still be demonstrated in it's current state, which manages local files on the server rather than VOSpace database objects.



  To demo this page, browse through the file hierarchy either using the menu on the left or clicking through folders/files on the main frame. The different features that can be demo'd here include viewing file details by clicking on file, renaming, downloading or deleting a file (by either right clicking on file on the left menu, or clicking on the file which will display these options on the main page), switching between grid/list view (buttons on top right of page), adding a new folder (option on top menu bar), or uploading a new file (first browse to file on local computer, then click upload after selecting it).

  Step 3 [Freeform SQL Query]

  Data Access/Freeform SQL Query will take you to the Freeform SQL query page, which along with the *Table viewer* (step 4), is one of the main parts of the demo/interface. Before

sending a query, you can describe and show some of the available options.

For example:

The *Table Metadata* link, will display the metadata for a given database. This metadata is in the form of an 'accordion' style menu, so clicking on table titles will slide down available columns for a table and then information for each column.

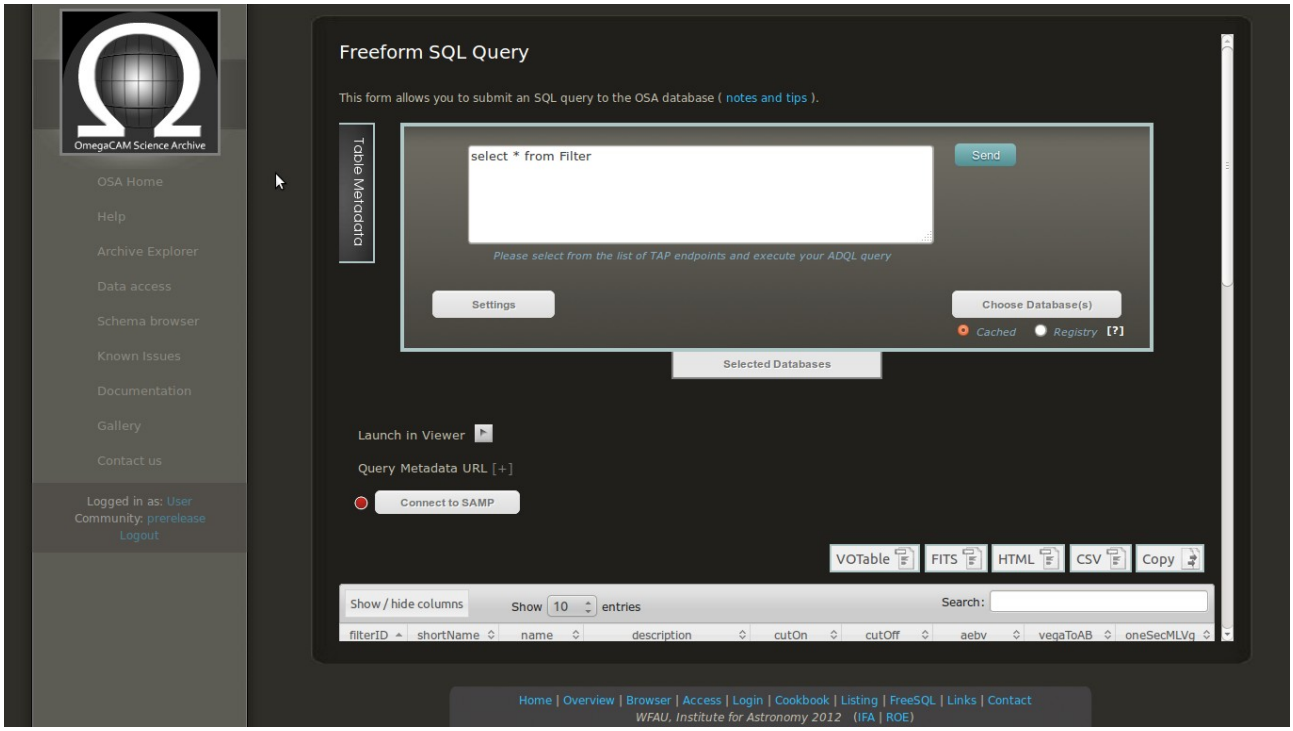
The *Choose Database*(s) button will drop-down a menu of available databases (TAP endpoints since the TAP protocol is used to interact with these databases), so you can either use the pre-selected OSA TAP endpoint or select another database (either local or external). When we launch the interface, the user will be able to select more than one endpoints and send queries directed to any or all of them, however the back-end work for that is still in progress so this is not available yet.

Selected Databases will display what databases are currently selected.

The Setting button allows users to select if they want server-side or browser-based data processing. While server-side is fine in most cases and preferable for large data sets, browser-based will be faster for small data sets.

For a demo the options could be left default, with a simple explanation sufficing.

Sending a query:
Use a query such as “SELECT TOP 100 * FROM MULTIFRAME WHERE rabase>0 ORDER BY rabase” which is the main testing query that I have been using.
You can mention that the query can be stopped using the stop button, or sent to an email with the according button.

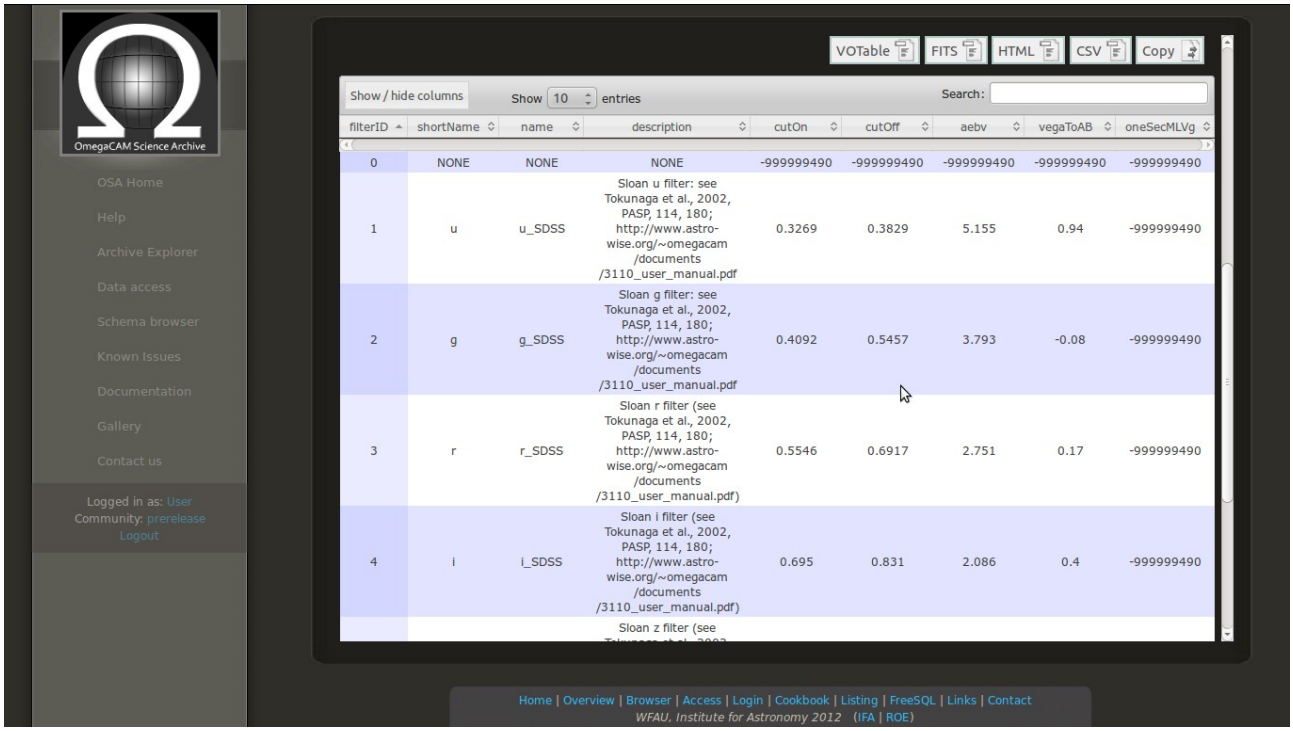


This will return a table with the results, and a set of features for viewing the data, or query metadata.

The *Query Metadata URL* ([+] toggles down, [-] hides) will display a link to the TAP endpoint where the query was run, if the users wants to find any information for the job run e.g. start/end time

The data can be downloaded in a number of available formats using the buttons above the table.

Within the table of data, users can rearrange and show/hide columns, sort data by any column, search (filter) the data and toggle the number of rows displayed, up to 100 per page.



A SAMP connection button is also included here, however for the demo perhaps it would be better to display this feature as part of the “Table Viewer” page since that includes some additional SAMP options.

Finally a link to the table viewer is included, where additional data exploration features are available, so this would be a logical next step for the demo.

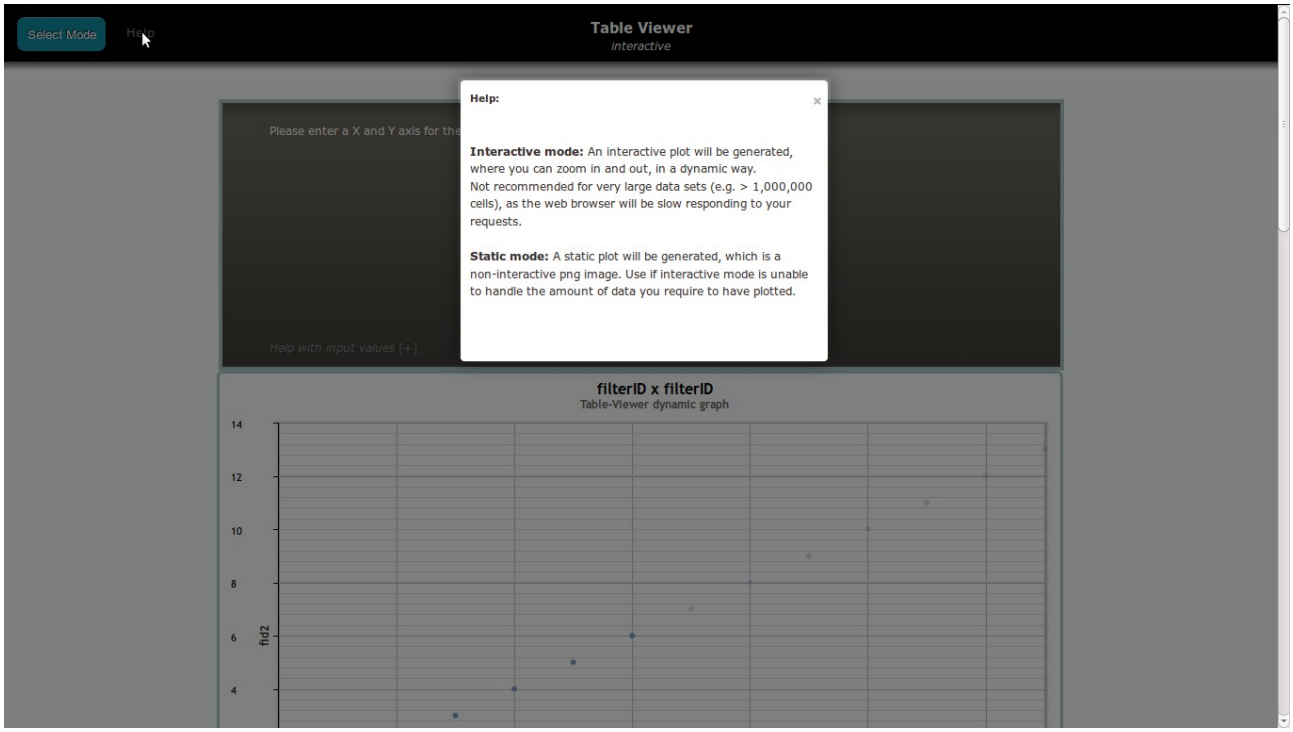
Step 4 [Table Viewer]

The table viewer is one of the pages that the demo should focus on, as it is generally the area where users of this tool would spend more time exploring the data, creating plots, filter and export data using SAMP etc.

Upon launching the data in the viewer, the original data is immediately available with the same options as the *Freeform* page. Additionally there is an input form where users can enter column names or mathematical expressions using column names, and generate different types of plots with the data.

On the top left side of the page there is a help link (which pops in a frame with help information) as well as a “Select Mode” drop down menu. There are two options available

through this menu which are also described via the help link: Interactive mode, which is used to generate dynamic/interactive plots and the static mode which is used to generate non-interactive plots on the server-side (images).



You may want to briefly demo part's of this interface such as clicking the help link at the top of the page, as well as the help link on the bottom left of the input form (which drops down a post-it image with some help information on the axis input), or hover the select mode menu, just to show that there has been some focus on building a nice visual experience with these non-static elements.

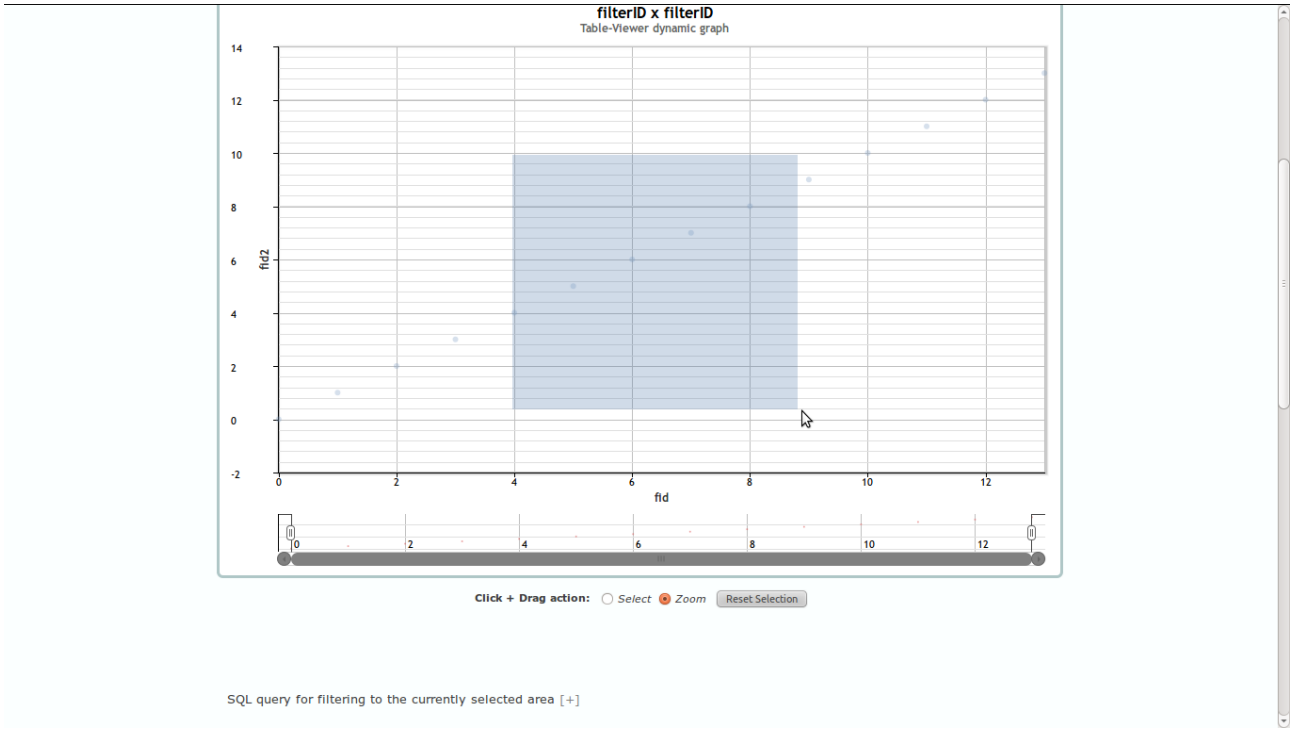
Plotting the data can be done by simply adding input parameters such as: multiframeID as the x-axis and decbase as the y-axis (any values as labels). There are different types of plots that can be created, such as scatter plots, histograms, density and a mixed density/scatter plot which is similar to a standard scatter plot, except that there are bins if more than a certain number of points is contained in a square area. There are some additional options based on the type of plot, such as the number of bins, (higher number = smaller bin width) and inverted axis.

A demo sample of these plots could be: Create an interactive mixed scatter/density plot with the aforementioned axis values, show some interactions (point selecting, zooming, point click → row highlight, row click → point highlight). Then create an interactive histogram, while finally switching to static to show a simple static scatter plot or density plot with the same axis.

Histograms are pretty straightforward, and the only thing to mention to an audience would be that you can hover over a bar to view the range, value etc.

A small thing to notice is that whenever a plot is generated, there is also a table of plot points generated.

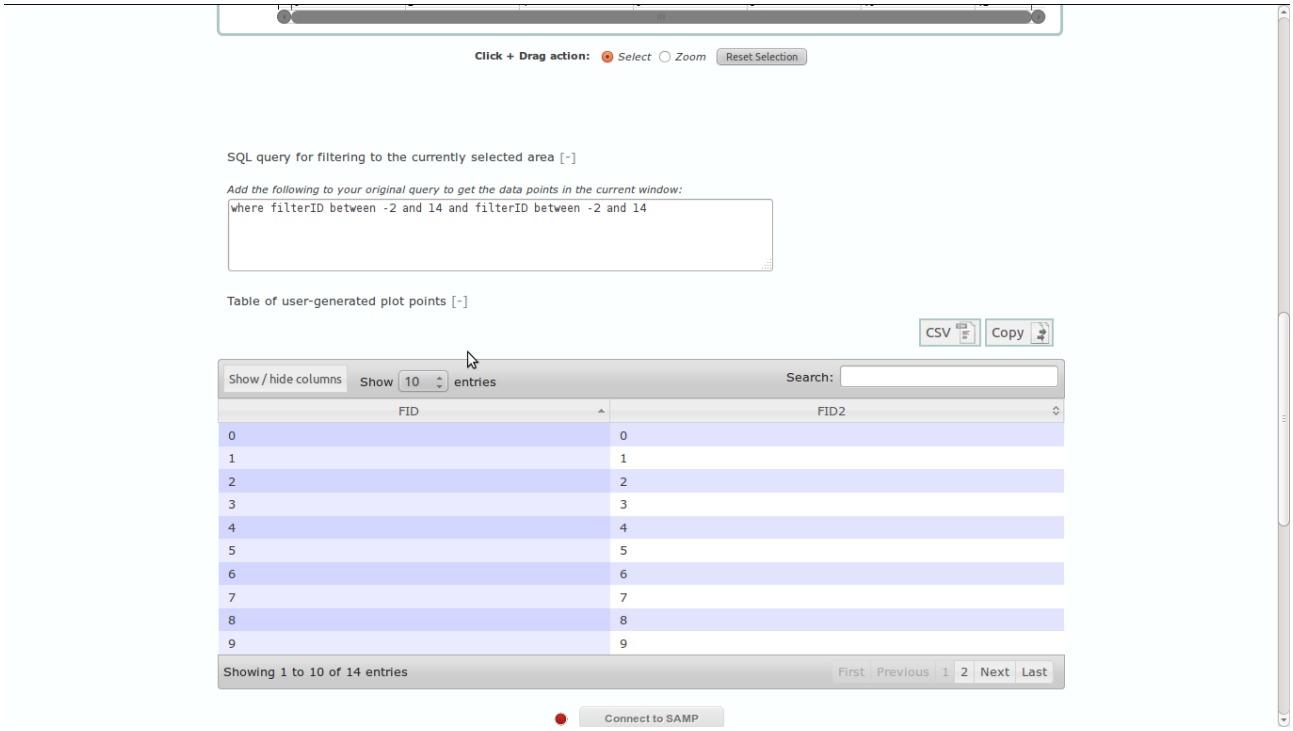
What could be described in more detail would be the mixed scatter/density plot (or the simple density plot as they are quite similar). If you generate a plot with the aforementioned (or other) data try the following: right click in the plot to zoom in.



Notice that if you zoom in an area that contains a bin after a certain zoom percentage the bin will turn to individual points. Also notice that there are some options below the plot, which specifically determine what the left mouse click + drag action will do. Zoom out and change this option to “Select” and then try dragging around an area with points and/or bins. You will notice that these points are highlighted in the plot, but also the table is filtered down to these values. An important point to describe now is that through this way you can filter your data set down to any desired selection and then export or download this dataset, as the “save as” options save the current filtered selection. Additionally, instead of click and dragging, you may simply click on points which will also filter the table to those values, or click on individual bins which will display a notification with the density of that bin.

Finally the inverse behavior is also available, specifically clicking on rows which will highlight a point, or a bin if the point is contained within one.

An additional feature of the scatter and mixed scatter/density plot is that there a link below the plot which when clicked will display SQL for filtering to the currently viewing area which depends on the zoom percentage (e.g. where multiframeID between 0 and 55000 and decbase between -35 and 5).

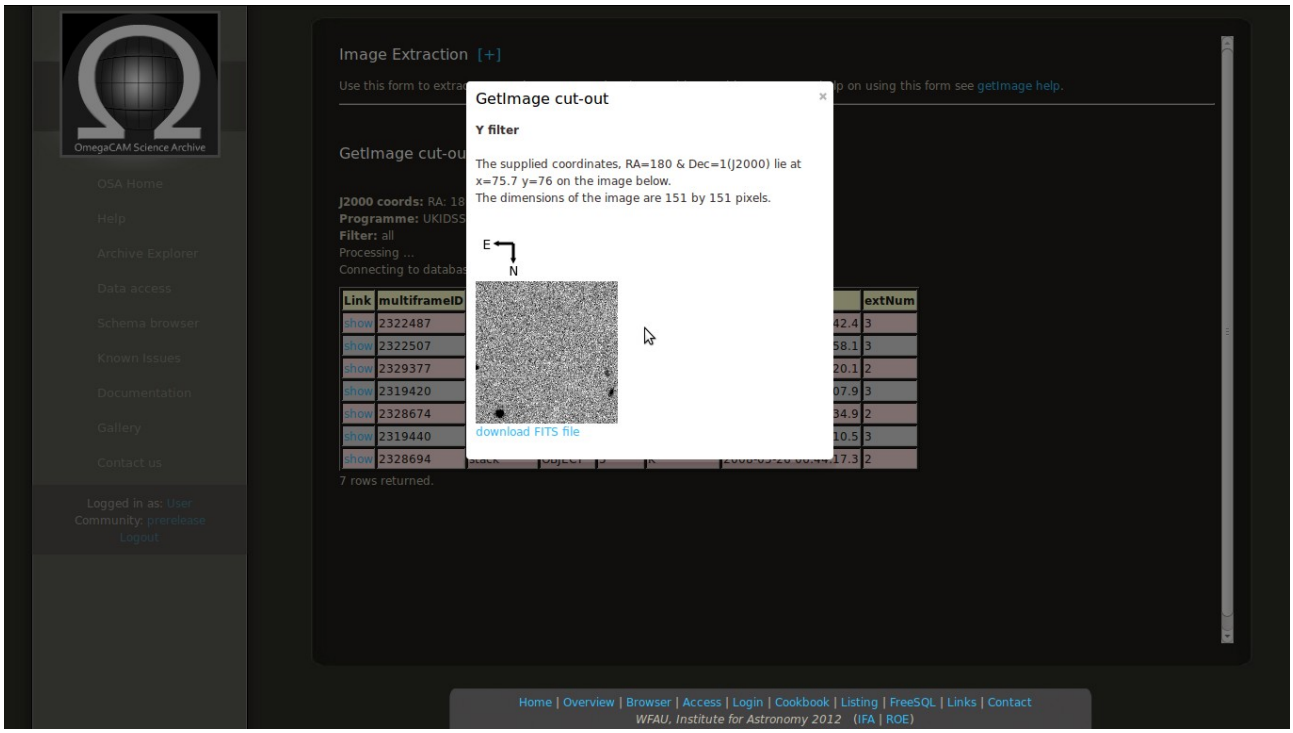


The last part of a demo could be talking about and showing a sample of the SAMP functionality. More precisely, there is a *Connect to SAMP* button below the plot, which when clicked connects to any available SAMP hubs (e.g. *Topcat*). Once connected, there will be a notification icon and green light to show the connection status, and users can at any time broadcast a table to the other SAMP clients. Note that this broadcast function takes whatever values are currently in the main (original) data table, so if the user has filtered down by selecting a number of points, this filtered selection will be broadcast, not the whole table.

Once a table has been broadcast users will be able to click on rows to highlight them in other clients such as Topcat, or select points/bins to highlight one or more rows in these clients. The demo could include this in a simple row click and then a bin drag + select, showing how these highlight rows in Topcat for example.

Step 5 [GetImage]

Some of the data access pages such as the '*GetImage*' page use data from other surveys (UKIDSS). However it could be useful to demonstrate this page to show how the data access forms are, sending a simple submission such as 180 RA and 1 Dec, and then clicking on the link to the image which fades out the background and displays a frame with the according image.



Step 6 [Schema Browser]

The Schema browser menu link launches the interactive schema browser which has the same structure as the browsers of other WFAU but is dynamic so the menu tree as well as menu entries can be toggled on/off.

If you wish to display any other parts of the OSA Interface, the interactions and the data access forms should be self-explanatory.

- **Notes on system architecture**
  This OSA interface is build using web.py, a Python-based web framework as the backbone. The way web.py allows developers to structure websites, allows for easy maintenance and extensions, using templating and clear and concise URL handling through Python classes. The interactivity is brought through the use of Javascript/AJAX and use of new technologies such as JQuery. The combination of AJAX with webpy allows easy interactions and communication between the client and server, enabling the interface to be non-static and thus a richer experience for the user.
  The interactions with the data archives (databases) is achieved using VO standards (TAP, VOTable, and SAMP for client interactions with desktop apps) and in some cases using HTTP requests to Java servlets that were already in place.

- **Missing [Incomplete] Features**
  - Email to user. While this function works, it is not configured on the server yet, as we need to have an SMTP user account/password setup.
  - GetImage, MultiGetImage and CrossID do not use ATLAS data yet, as we are waiting for the Java servlets to be in place, and currently use UKIDSS data.