# Dundee Workshop 09/04/2019

The presentation and a PDF version of the workshop are available at
https://downloads.openmicroscopy.org/presentations/2019/Dundee-April/

## Software versions used for this workshop:

- OMERO: 5.4.10
- OMERO.iviewer: 0.7.0a4
- OMERO.figure: 4.0.2
- OMERO.mapr 0.2.3
- OMERO.parade: 0.1.1
- OMERO.FPBioimage: 0.3.0
- OMERO training scripts: 0.5.1
- OMERO training notebooks: 0.4.1
- TrackMate: 3.8.0
- Orbit: 3.0.8
- Bio-Formats: 5.9.2
- Fiji/ImageJ: 2.0.0-rc-68/1.52e
- CellProfiler: 3.1.3

## Content

- Introductory talk about OMERO
- Analysis with 3rd party tools
  - Analysis with Fiji: manual
  - Java API:
    - Using API via Fiji
  - Trackmate example
  - Orbit (machine learning tool)
    - Manual workflow
    - Scripting in Orbit
- Analysis in OMERO (Python)
  - Analysis server side analysis
  - How to write script
  - How to manage script
- API usage
  - Python API
    - in Jupyter notebook - simple FRAP analysis

- ■ Using API via a 3rd party software CellProfiler
  - ○ R API in Jupyter notebook
- ● Data mining using OMERO.parade
- ● Administration of OMERO

# Analysis with Fiji

**Description**

Fiji is a free open-source image processing package based on ImageJ.

We show how to analyze a batch of images stored in OMERO using the scripting facility in Fiji, the results of the analysis will be stored as a CSV and OMERO.table,

If analyzing other users data, the user analyzing the data needs to have admin privileges.

More information about restricted privileges can be found at
https://docs.openmicroscopy.org/latest/omero/sysadmins/restricted-admins.html.

**Setup**
- ● Fiji has been installed on the local machine with the OMERO.insight-ij plugin. The installation instructions can be found at http://help.openmicroscopy.org/imagej.html#plugins.
- ● Installing the OMERO.plugin also adds the dependencies required to connect to OMERO using the Script Editor of Fiji.
- ● The OMERO.script
  https://github.com/ome/training-scripts/blob/v0.5.1/practical/python/server/batch_roi_export_to_table.py has been uploaded to the server.

## Fiji Manual simple workflow: Draw ROI

1. Launch Fiji.
2. Go to *Plugins > OMERO > Connect To OMERO.* This will show a login screen where you can enter the name of the server to connect to, the username and password. The OMERO plugin will allow you to browse your data in a similar manner to the webclient.
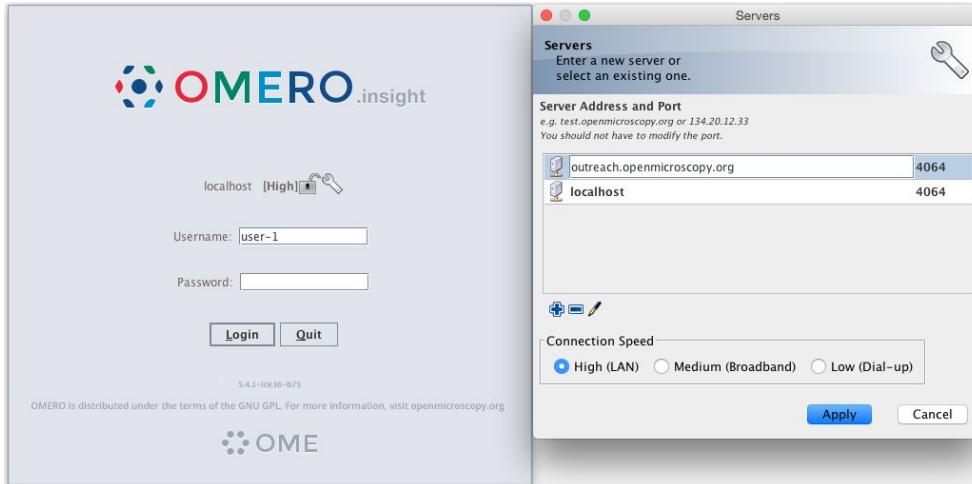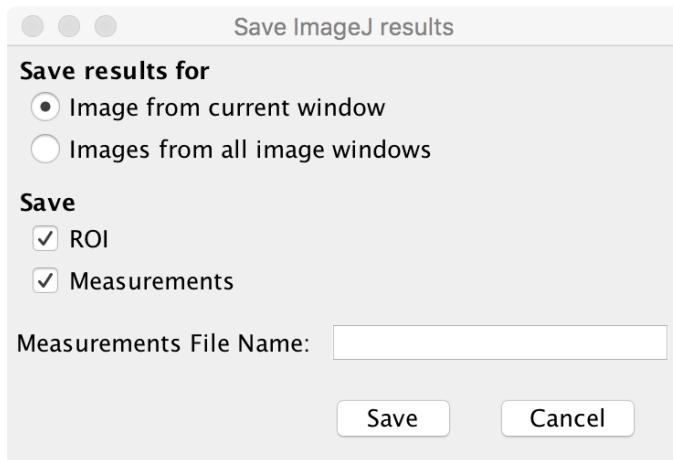
3. Select the **A-Fiji-dataset** Dataset.
4. Double-click on a thumbnail or on an Image in the left-hand tree to open an Image in ImageJ.
5. Go to *Analyze > Tools > ROI Manager…*
6. Draw a shape using for example the Freehand selection tool.
7. In the ROI manager, click the button *Add [t]* to add the shape to the Manager.
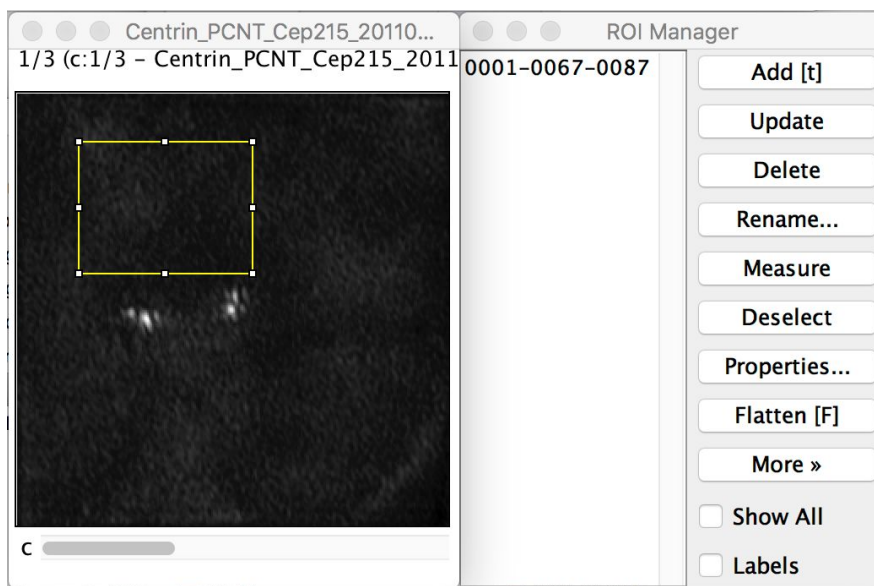


8. Move to another channel, using the *c* slider.
9. Draw other shapes if desired. Add drawing a shape. Click *Add [t]* to add it to the Manager.
10. When done with the drawing, click the button *Measure* in the ROI Manager
11. A dialog with measurements for each shape pops up.
12. To save the ROI and the measurement back to OMERO, go to *Plugins > OMERO > Save ROIs To OMERO.*
13. In the dialog popping up, under the *Save* section select *ROI* and *Measurements*.

14. The measurements are saved as a CSV file.

```
⬤ ⬤ ⬤            Save ImageJ results

Save results for
  ⦿ Image from current window
  ◯ Images from all image windows

Save
  ☑ ROI
  ☑ Measurements

Measurements File Name: [                    ]

              [ Save ]      [ Cancel ]
```

15. Go back to the webclient, in the right-hand panel, check that there is a new CSV file under the *Attachments* harmonica.
16. Open the image in iviewer to the see the ROIs and make sure that you can interact with them.

## Fiji Manual simple workflow: Crop Image

1. This time we will crop an Image stored in OMERO and saved the cropped Image back to OMERO as OME-TIFF.
2. Select the **A-Fiji-dataset** Dataset.
3. Double-click on a thumbnail or on an Image in the left-hand tree to open an Image in ImageJ.
4. Draw a Rectangle on the Image.
5. Select *Image > Crop*
6. A new Image will be displayed in a Fiji window
7. Go to *Plugins > OMERO > Save Image(s) To OMERO*
8. An Import dialog (specific to Fiji plugin) will pop up.
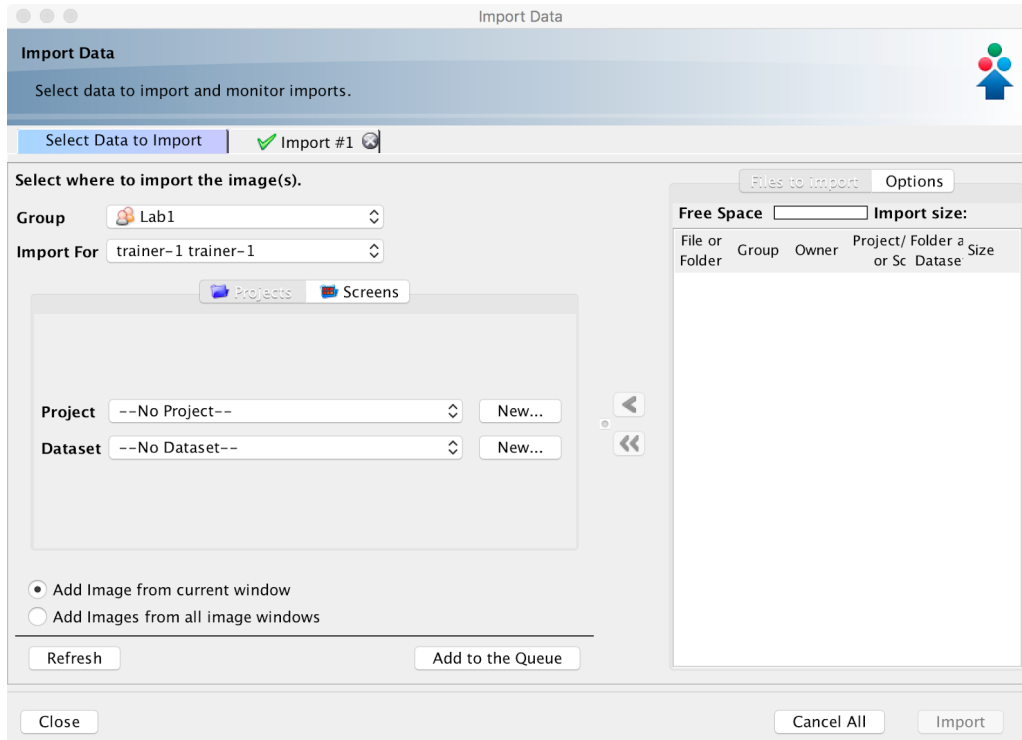
9. Check that the option *Add Image from current window* is selected
10. Select where to import the cropped Image e.g. **A-Fiji-dataset** Dataset.
11. Click *Add to the Queue* button
12. Then click *Import.* The import will start
13. When the import is done, go back to the Tree view in the Fiji plugin or Web client. Refresh. Check the new Image.

## Fiji Manual workflow

1. Launch Fiji.
2. To open an image stored in OMERO, Go to *Plugins > OMERO > Connect To OMERO* if necessary.

3. In the OMERO login dialog, click the wrench icon  and then add the server address in the dialog i.e. **outreach.openmicroscopy.org**. Click *Apply*.
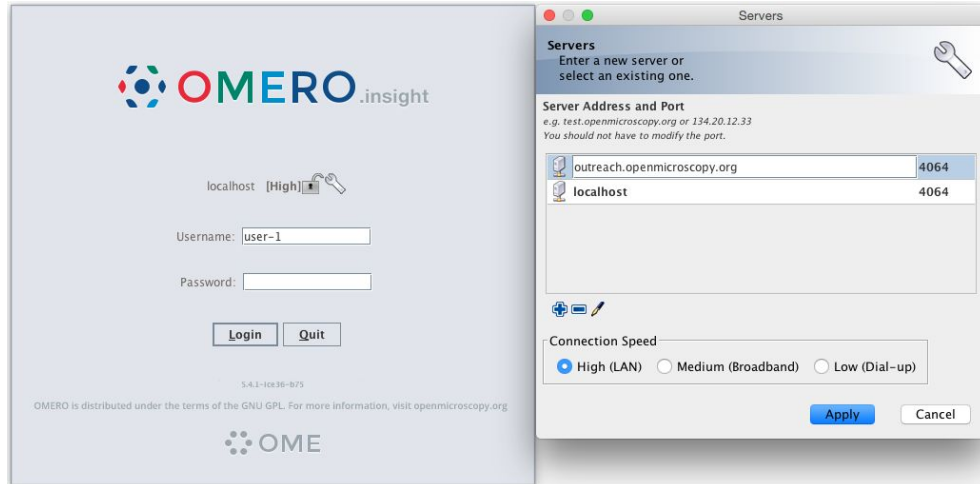
4. Enter your *Username* and *Password* and click *Login*.
5. Browse to the Project **idr0021**, open any Dataset and double-click once on an Image to open it in Fiji. Bio-Formats is used to view the Image.
    ○ Make sure to select *View stack with: Hyperstack* in the *Bio-Formats Import Options* dialog.
    ○ Note that each plane will be transferred from the server to the client machine so this may take a few moments.
6. To open the recorder, go *Plugins > Macros > Record…*, select *JavaScript* to record the actions. The steps will then be used in the Scripting workflow below.
7. Convert floating-point pixel-type to 8-bit using *Image > Type > 8-bit.*
8. Go to *Image > Adjust > Auto Threshold*, to open the *Auto Threshold* dialog:
    ○ Select *MaxEntropy* for the *Method* parameter.
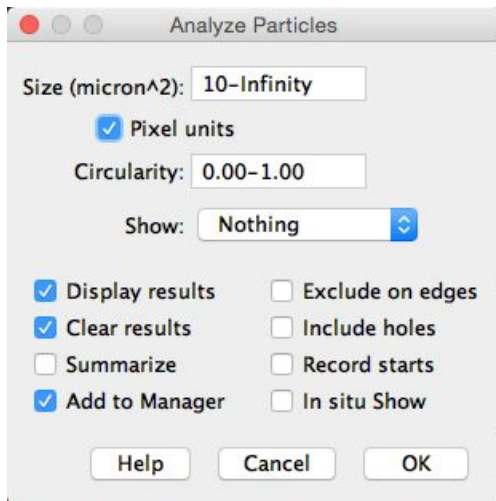    ○ Check the checkbox *Stack.*
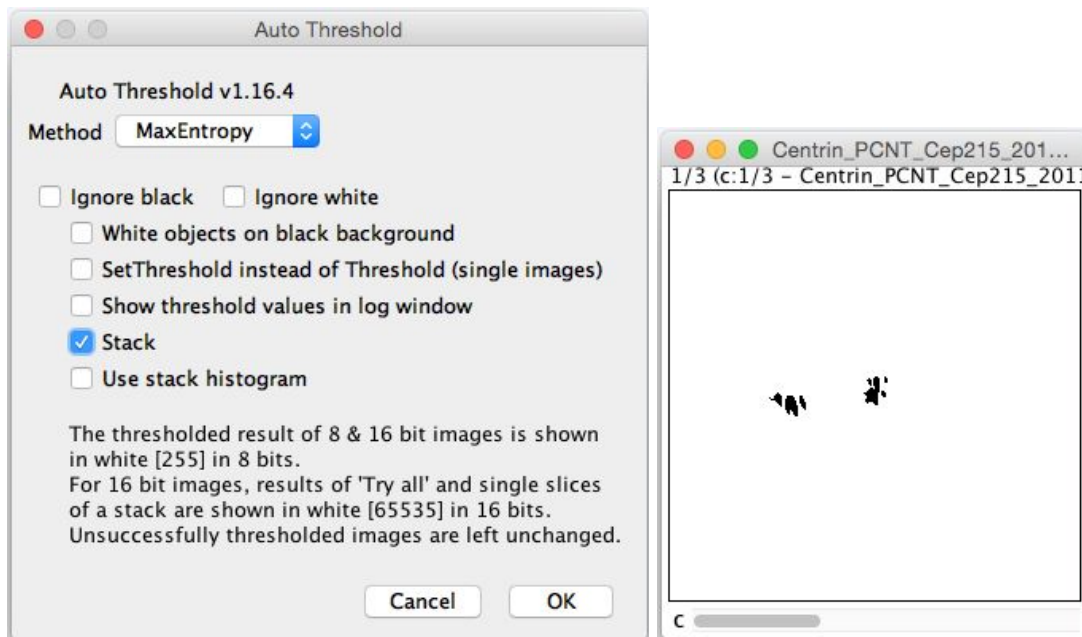


9. Click OK.

10. Then open *Analyze > Analyze Particles….*
11. In the dialog
    ○ Set *Size* to *10-Infinity* and check *Pixel units.*
    ○ Check the following checkboxes:
        i.   *Display results*
        ii.  *Clear results*
        iii. *Add to Manager*
    ○ Click *OK* then *Yes* in the popup dialog indicating asking to *Process all 3 images?*.



12. To save the thresholded Image back to OMERO with the generated ROIs and the measurements:
    ○ Select *Plugins > OMERO > Save Image(s) to OMERO.*
    ○ Create a New Dataset for the image
        i.   Click the *New…* button next to the selection box on the Dataset row.
        ii.  In the dialog that pops up, enter a name and a description (optional).
        iii. Click *Create.*
    ○ The newly created Dataset will automatically be selected.
    ○ Click *Add to the Queue* then *Import.*
    ○ Go to the webclient and check that the measurements have been saved in a CSV file and attached to the Images. The attachment can then be downloaded at any time.

## Fiji Scripting workflow

We will now repeat the manual analysis on a Dataset using the scripting facility available in Fiji.
Let's go over the scripts to understand the logic and see how it matches the UI steps.
This script explores the JAVA API using groovy. See
https://docs.openmicroscopy.org/latest/omero/developers/Java.html,

Now, we will present 3 simple steps (three scripts) to facilitate this analysis.
1.  Go to https://outreach.openmicroscopy.org/.

2. Log in using the credentials provided
3. Make sure you are selecting your own data. Select the Dataset **A-Fiji-dataset.**
4. Launch Fiji. (relaunch if opened).
5. Go to *File > New > Script….*
6. A dialog pops up. In the *Language* menu select *Groovy.*
7. Copy the content of the script
   https://raw.githubusercontent.com/ome/training-scripts/v0.5.1/practical/groovy/analyse_dataset_s
   ave_rois_and_summary_table.groovy into the text script editor of Fiji.
8. Edit the following parameters: dataset_id, USERNAME, PASSWORD.

```groovy
1 // Setup
2 // =====
3
4 // OMERO Server details
5 HOST = "outreach.openmicroscopy.org"
6 PORT = 4064
7
8 //  parameters to edit
9 dataset_id = 2331
10 USERNAME = "username"
11 PASSWORD = "password"
12
13 def connect_to_omero() {
14     "Connect to OMERO"
15
16     credentials = new LoginCredentials()
17     credentials.getServer().setHostname(HOST)
18     credentials.getServer().setPort(PORT)
19     credentials.getUser().setUsername(USERNAME.trim())
20     credentials.getUser().setPassword(PASSWORD.trim())
21     simpleLogger = new SimpleLogger()
22     gateway = new Gateway(simpleLogger)
23     gateway.connect(credentials)
24     return gateway
25 }
26
27 def get_images(gateway, ctx, dataset_id) {
28     "List all images contained in a Dataset"
29
30     browse = gateway.getFacility(BrowseFacility)
31     ids = new ArrayList(1)
32     ids.add(new Long(dataset_id))
33     return browse.getImagesForDatasets(ctx, ids)
34 }
```

Run    Batch    Kill                              Show Errors    Clear

9. The script will process all the Images in the specified Dataset, applying threshold, analyzing particles and saving ROIs back in OMERO i.e. we reproduce in a script the manual steps recorded above. Further, it will create a CSV and OMERO.table to be attached to that Dataset in OMERO. Both are not necessary when analysing data, this is mainly to show how it we
10. Click *Run.*
11. Return to the webclient and open an Image from this Dataset in OMERO.iviewer.
12. Click the *ROIs* tab to see the added ROIs. Note that the ROIs have been assigned a Channel index to indicate which Channel they were derived from.
13. In the *Settings* tab, turning channels on/off will also show/hide ROIs assigned to those channels.

14. Open the image in OMERO.figure for a quick publication by going to *Info* tab in iviewer and

    Open With: OMERO.figure |

    clicking on *OMERO.figure* in the *Open with* line.
15. This script was used previously to run on the whole **idr0021** Project and produce the resulting CSV file. We can now download this file from OMERO and open it in Microsoft Excel.
16. In Excel, select the Dataset column and the column bounding_box and then click on *Insert > X Y*

    *Scatter* [icon] *> Scatter* . Depending on the version of Excel you are using, this option might not be available.
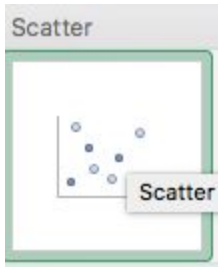
    
    .
17. This will create a scatter plot. Right-click into the Chart now and select *Change chart type > Statistical > Box and Whisker.*


1. Click the *ROIs* tab to see the added ROIs. Note that the ROIs have been assigned a Channel index to indicate which Channel they were derived from.
2. In the *Settings* tab, turning channels on/off will also show/hide ROIs assigned to those channels.
3. Open the image in OMERO.figure for a quick publication by going to *Info* tab in iviewer and

    Open With: OMERO.figure |

    clicking on *OMERO.figure* in the *Open with* line.
    .


## Fiji-TrackMate example

In this example we open all images from a dataset in OMERO in Fiji and use TrackMate plugin of Fiji. For more details about TrackMate, go to https://imagej.net/TrackMate. We first find the tracks of the moving objects. Afterwards, these tracks (polygon ROIs), as well as the segmented objects ROIs will be saved to the same images in OMERO. Also, an OMERO.table will be produced from the measurement results of TrackMate and saved as an attachment to the Project containing the images in OMERO. To determine the parameters suitable  to analyse the images, we will first go through a manual TrackMate workflow, then later we will use the parameters in a scripted TrackMate workflow.


**Setup**
● Fiji has been installed on the local machine with the OMERO.insight-ij plugin. The installation instructions can be found at http://help.openmicroscopy.org/imagej.html#plugins.
● Installing the OMERO plugin also adds the dependencies required to connect to OMERO using the Script Editor of Fiji.

Manual

Each of you can work on your own data. We will use this manual workflow to explain the steps of the script below.

1. Launch Fiji
2. Go to *Plugins > OMERO > Connect to OMERO*
3. Enter the username and password provided for you.
4. Browse the Dataset **artificial-trackmate**
5. Double-click on the **FakeTracks.tif** Image to open it in Fiji. Make sure it is opened using the Hyperstack viewer (This option is dependent on what you did last time when using your Fiji. If you last time used Fiji for running the scripts from Day 1 of this workshop, you should be fine.)
6. Go to *Plugins > Tracking > TrackMate.*
7. Click *Next* in the first dialog that pops up.
8. A new dialog pop ups indicating to select a *detector*. Select the *DoG detector.* Click *Next*.
   a. Set the *Estimated blob diameter* to 5.0
   b. Set the *Threshold* to 5.0
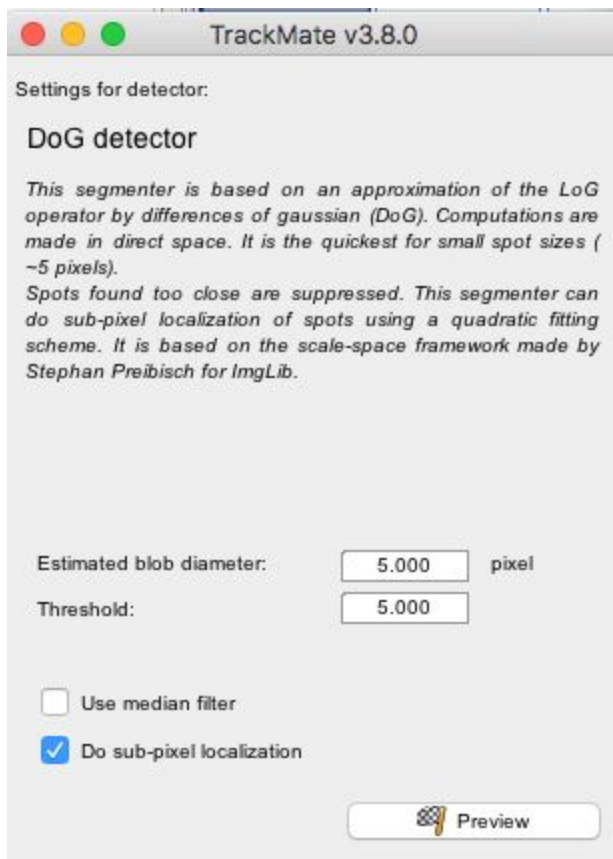


9. Click the *Preview* button to find spots. 4 spots should be found
10. Click the *Next* button few times, until you get to the *Select a view* dialog.
11. Select the *HyperStack Displayer* view
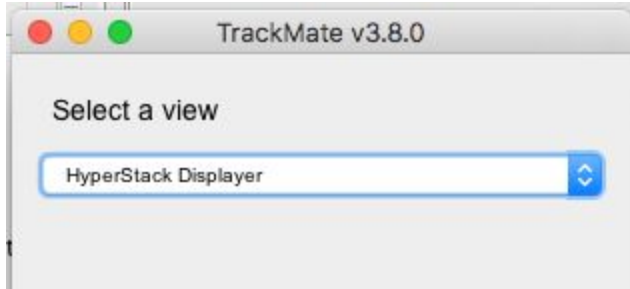
12. Click the *Next* button twice, until you get to the *Select a tracker* window. Select the *LAP Tracker*
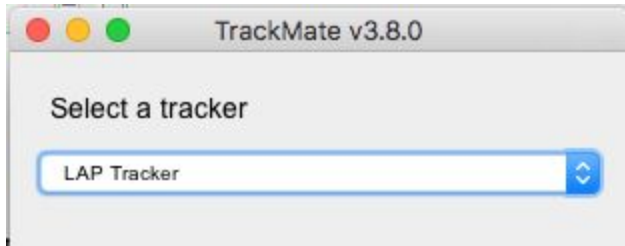


13. Click the *Next* button few times (5), until a dialog with three tabs pops up, see screenshot below.
14. Select the Tracks tab to display the Tracks.
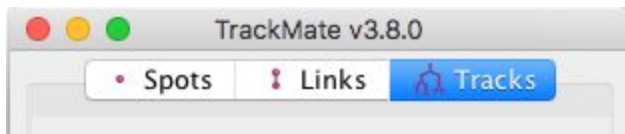


15. Click Next until you get to *Select a action* dialog.
    a. Select the option *Capture overlay*
    b. Click *Execute*.
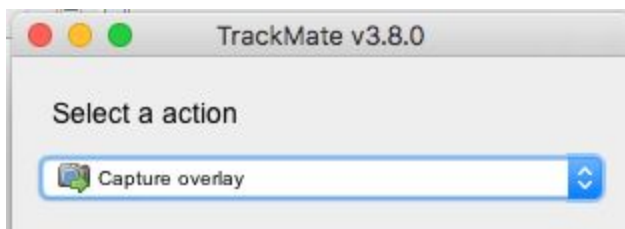


16. Click OK in the following dialog, leaving the defaults (the whole stack will be captured).
17. New image appears. Select it. This new Image can then be imported as on OME-TIFF using the option *Plugins > OMERO > Save Image(s) to OMERO.*


Programatically

This option saves the ROIs to the server, we do not generate a new Image.

1. Go to https://outreach.openmicroscopy.org/webclient/.
2. Enter the username and password provided.
3. Make sure you are working with **your own data**.
4. Create a Project: **trackmate.**

5. Drag and Drop the Dataset **dv-iain-multiTZ** into the Project:  **trackmate.**  Note the ID of the Project.
6. Launch Fiji, open *File > New > Script...* and select *Python* as language.
7. Copy the content of the script https://raw.githubusercontent.com/ome/training-scripts/v0.3.0/practical/jython/tracker.jy  into the script window of Fiji.
8. Edit the credentials to connect to the server and change the ID of the Project in the script window of Fiji, replacing it with the ID of your Project **trackmate.**
9. Study the script step-by-step.
10. Click *Run*.
11. After the script has finished, go to the images in the webclient and open the second image in OMERO.iviewer.
12. Click on the ROI tab and observe that you now have ROIs under which there are Shapes. Each ROI is a collection of shapes. The ROI corresponds to a Track in Trackmate. There is always one polyline shape in each ROI which represents the track. The other, elliptical shapes in the same ROI represent the tracked spots.
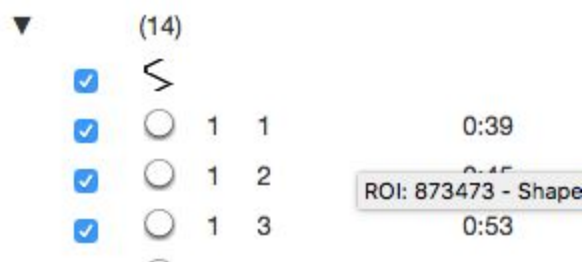


13. Play the timelapse video in OMERO.iviewer.
14. Go to the *Info* tab, and in the *Open with:* line click on *OMERO.figure.* In OMERO.figure, add the Tracks and ellipses to the panel by selecting the appropriate ROIs in the *Labels* tab of OMERO.figure.
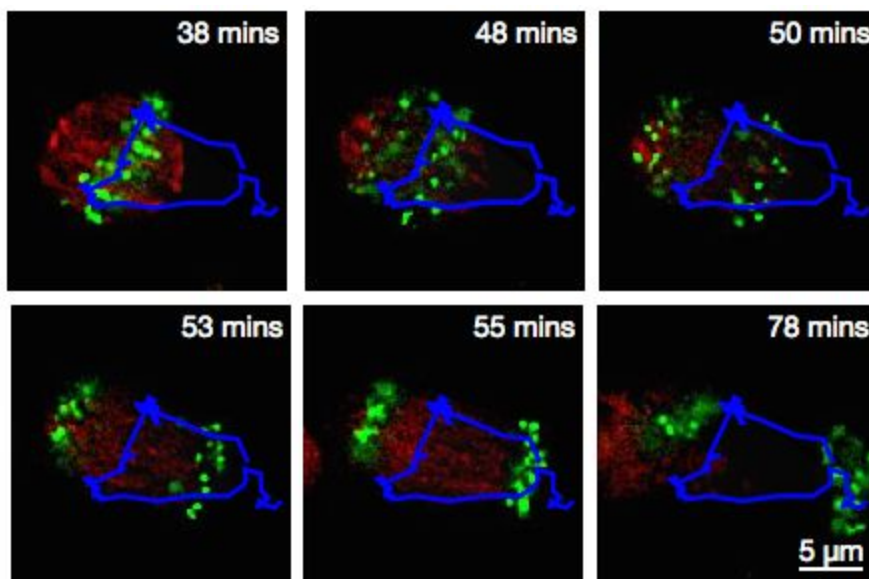
# Orbit workflow: Manual training and segmentation

**Description:**
**Orbit Image Analysis** is a free open-source software with the focus on quantification of big images like whole slide scans. It offers sophisticated image analysis algorithms. Of those, tissue quantification using machine learning techniques, object / cell segmentation, and object classification are the basic ones. For more details, go to http://www.orbit.bio/.
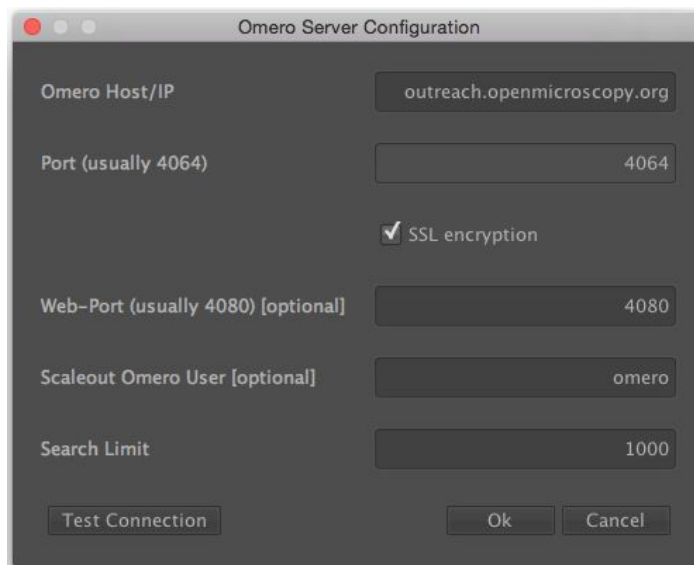
**Setup**
Orbit has been installed on the local machine. See http://www.orbit.bio/ for details.

## Orbit workflow: Login and manual training

1. Launch Orbit, click Yes in the first dialog box:



2. Then enter server details in the next dialog. Only the Host/IP field is essential here and should be set to **outreach.openmicroscopy.org**.

3. Then login to OMERO with the username and password provided:



4. Orbit will show data from OMERO in the left-hand panel. Click *show only my assets* to filter by data you own.
5. Select the group **Lab1**.
6. Datasets not within a Project are listed under *unassigned*:
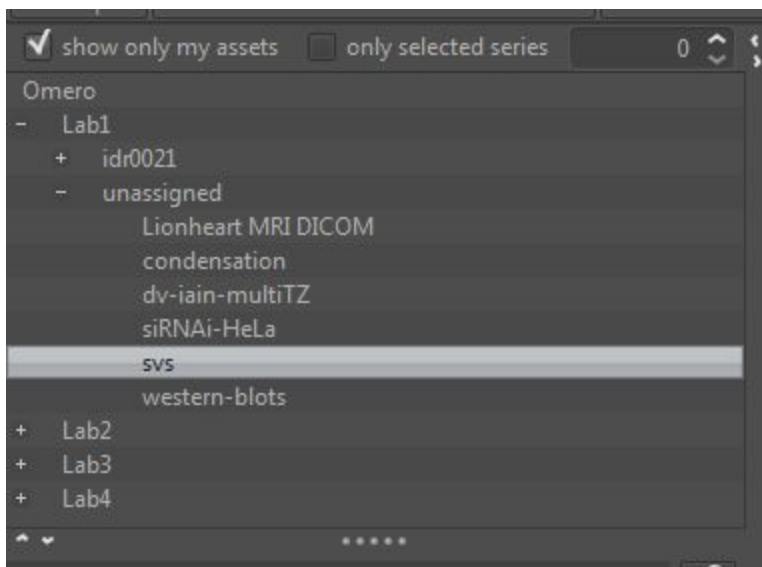


7. Select the **svs** Dataset. Image thumbnails will be shown in the panel below.
8. Double-click on the **77928.svs [Series 1]** thumbnail or drag and drop it into the centre panel and



maximise the viewer window to fill the centre panel.
9. We want to train a model to recognize cell nuclei and use this for segmentation.
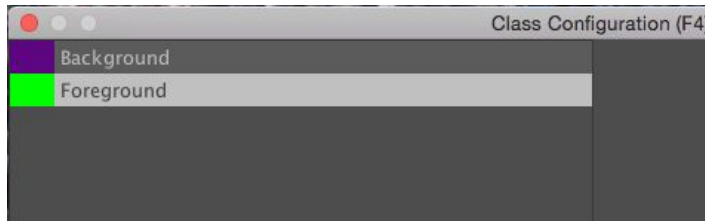
10. Click the *Model* tab and then the *Classes* button 

11. In the dialog, remove the *Celltype 1* class by selecting it and click the *remove class* button.

12. The select the *Celltype 2* class, and rename it by typing *Foreground* in the name field and clicking *rename class*.
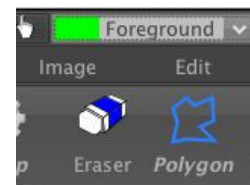
13. You should now have two classes named *Background* and *Foreground*:



14. Click *OK* to close the dialog.

15. Now we will construct the model by defining regions of *Foreground* and *Background* on the Image.

16. Click the *Object Detection* tab, select the *Polygon* tool and choose the *Foreground* class from the



chooser at the top-left of the screen.

17. You can now draw around a number of cell nuclei on the Image. The more accurately you draw and the higher number of objects you define, the more you will improve the performance of the segmentation, but about a dozen should be sufficient.
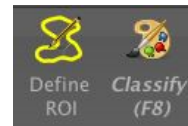
18. Now switch to the *Background* class and draw around several background regions:
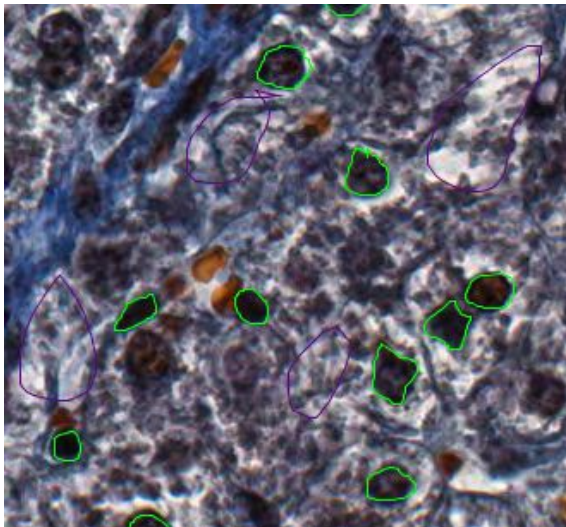


19. We can then train the model by clicking the *Train*  button or press F7. You will see a progress bar in the right-hand panel.

20. To see how this model classifies objects within a region, click the *Define ROI* button and draw around a region of the image. Then click *Classify*. If no ROI is drawn, Orbit will attempt to classify the whole Image which can be very time-consuming.

21. Once the classification is complete, a notification window pops up. Close it and view the results on the Image by dragging the slider below the Image to the right:
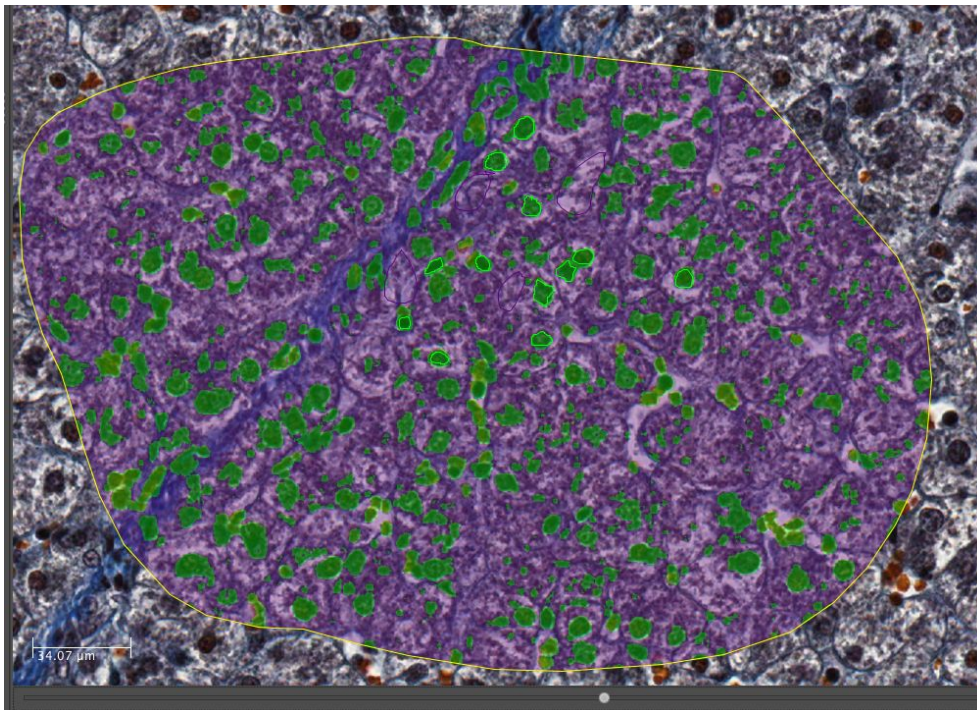


22. To segment the Image using this classification, click *Set Primary Segmentation Model* and then *Object Segmentation*.
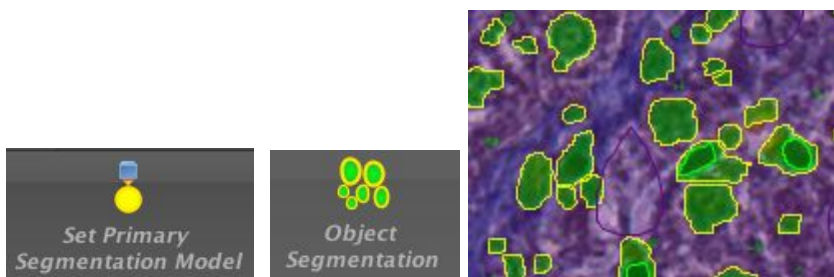


23. Click the *Model* tab and *Save Model On Server,* enter a name to save the model to OMERO. Note that you can also use *Save Model as…* to save the model to your local drive.

## Orbit workflow: Scripted segmentation and saving to OMERO

**Description:** We will use the model created in the last step above to repeat the segmentation, using a script which allows us to save the results back to OMERO. This will use a saved ROI Annotation instead of a temporary ROI as in the manual workflow.

1. Re-open the same image **77928.svs [Series 1]** to clear the ROIs and in the right-hand panel select the *Annotations* tab.
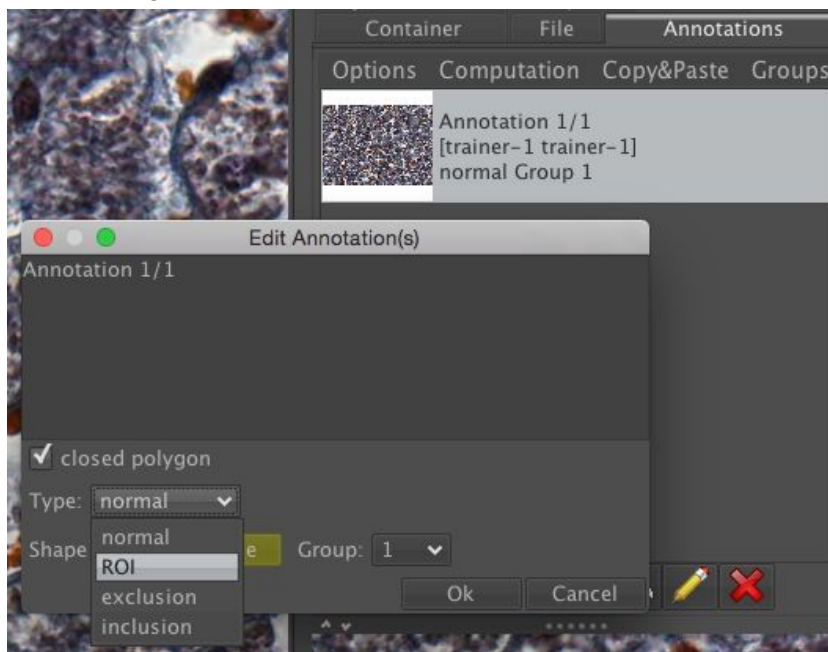2. Pan the Image to a region you wish to analyse, select the *Add Polygon* button and draw around a region. 
3. Select this *Annotation* from the list in the right panel and click *Edit* (*pencil ic*on).
4. In the dialog, set the *Type* to *ROI.*



5. Click *Ok.* This will save the ROI as an annotation on this image in OMERO.
6. Click on *Tools > Script Editor* to open a scripting window.
7. Copy the script from training-scripts: https://raw.githubusercontent.com/ome/training-scripts/v0.4.2/practical/orbit/segmentation.groovy and replace the existing code in the script window.
8. Update the username and password
9. The script will load the Orbit model and the ROI that we saved to OMERO, segment the image within the ROI and save the segmented shapes as Polygons to OMERO.
10. Click *Run*.
11. When complete, you can use OMERO.iviewer to see the ROIs created in OMERO.

## Exploring the OMERO Java API

We can use the Fiji's script editor as a convenient way to explore the OMERO Java API. For convenience, we will use Groovy language.

**Exercise 1**

Using the above example and the documentation available at
https://docs.openmicroscopy.org/latest/omero/developers/Java.html, write a script with the following
steps:
- Load any image stored in OMERO using its ID.
- Open the image using Bio-Formats Import plugin.
- Crop a rectangle using IJ.makeRectangle.
- Save the generated image locally as OME-TIFF using  the Bio-Formats Exporter plugin.
- Create a dataset in OMERO into which the image will be imported.
- Import the generated image to OMERO.

**Exercise 2**

Using the above example and the documentation available at
https://docs.openmicroscopy.org/latest/omero/developers/Java.html, write a script with the following
steps:
- Load an image from **idr0021** using its ID.
- Open the image using Bio-Formats Import plugin.
- Analyse the image using Analyse particles plugin (see script above).
- Count the number of ROIs.
- Create a Map Annotation with the following Key/Value pairs:
  - Analysis: Fiji
  - ROIs: Roi count
  - Antibody: Rabbit
- Link the Map Annotation to the Image.


**Solutions**

Exercise 1
https://raw.githubusercontent.com/ome/training-scripts/v0.5.1/practical/groovy/crop_rectangle_from_ima
ge.groovy

Exercise 2
https://raw.githubusercontent.com/ome/training-scripts/v0.5.1/practical/groovy/analyse_image_map_ann
otation.groovy

# Analysis in OMERO

## Analysis workflow Server side

**Setup**

- The following OMERO.scripts have been uploaded to the server for this workshop:
    - https://github.com/ome/scripts/blob/v5.4.10/omero/export_scripts/Batch_ROI_Export.py
    - https://github.com/ome/training-scripts/blob/v0.5.1/practical/python/server/Kymograph_Analysis.py
    - https://github.com/ome/training-scripts/blob/v0.5.1/practical/python/server/Kymograph.py

1. We will now analyse the ROIs created in OMERO.iviewer using a server-side script.
2. Go to the **siRNA-HeLa** Dataset and open several images whose name start with **VRAQ…** in OMERO.iviewer.
3. We want to measure the distance between Centromeres, stained with ACA in the 4th Channel. Turn on **ONLY** the 4th channel and open the *ROIs* tab to on the right-hand pane.
4. Draw several lines between the centromeres as indicated on the screenshot below.
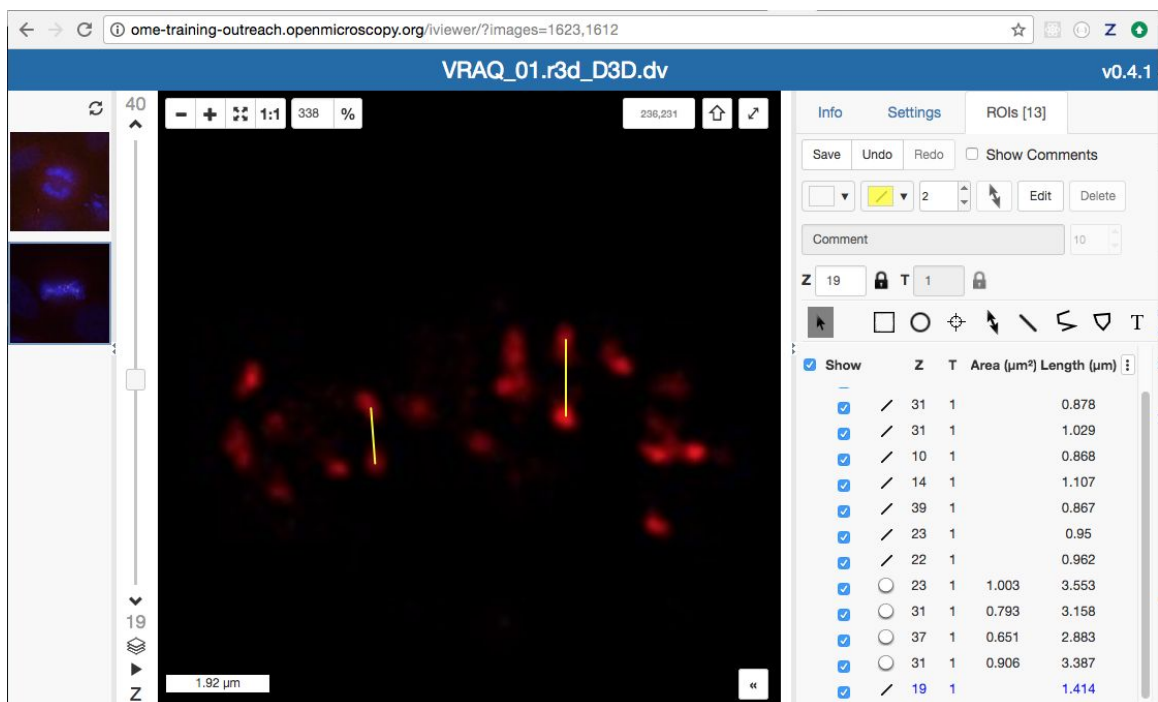


5. Try to identify centromere pairs:
    a. Select the *Line* tool and draw a line between the centres of the centromeres.
    b. In the ROIs table, in the *Comments* column, click the 3 dots in the column header and choose to *Show Area/Length*.
6. Do the same now on several of the Images whose names start with **IN...**, which are in the Metaphase state.
7. Select Dataset **siRNA-HeLa**.

8. Click the *Script* button in the top-right of the page       .
9. Select *export_scripts > Batch_ROI_Export…*
10. In the dialog that pops up, click on *View Script* to view the Python code.
11. Search for "*idr*" to find the code block that selects the filter_channel based on Dataset and Channel names.

12. Scroll to the bottom of the script to see where the input parameters are defined, such as *Data_Type* and *IDs*. Note how these appear in the script dialog and are auto-populated with the currently-selected Datasets or Images.
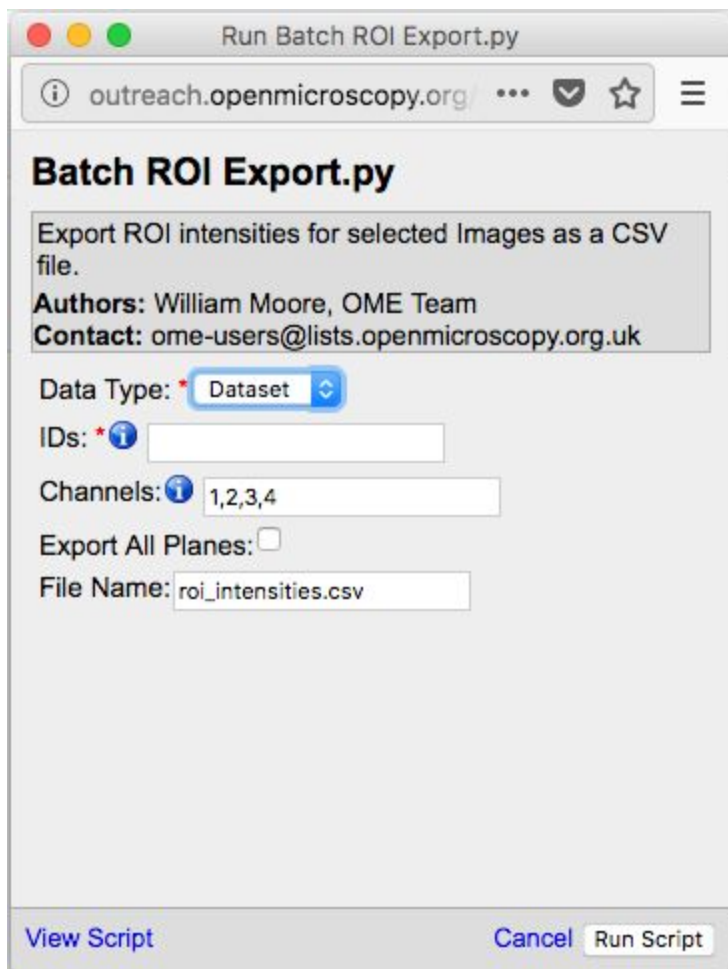


13. The script will process all the Images in the selected Dataset and data can be exported as CSV file, with one table row per Shape/Channel.
14. Click *Run Script.*
15. The status of the processing is displayed in the *Activities* dialog .
16. When the script has completed, it will show in the *Activities* dialog and allow you to download the CSV file. Download this and open it, e.g. in Excel, to see the output data for all the shapes.

We will now use another server-side script for creating a Kymograph from an Image in OMERO. The Image we will use for the Kymograph demonstration has been published in a study investigating protein migration along microtubules http://jcb.rupress.org/content/194/2/187.(Bowen et. al. Journal of Cell Biology 194 (2): 187).

1. Go to the Dataset **Kymograph**.
2. Select the Image inside the Dataset.

3. Double-click to open the Image in OMERO.iviewer and draw one or more lines along microtubules which seems to have the most persistent trafficking of objects along them.
4. Save the line(s).

5. Go back to the webclient. Click the *Script* button in the top-right of the page ⚙ .
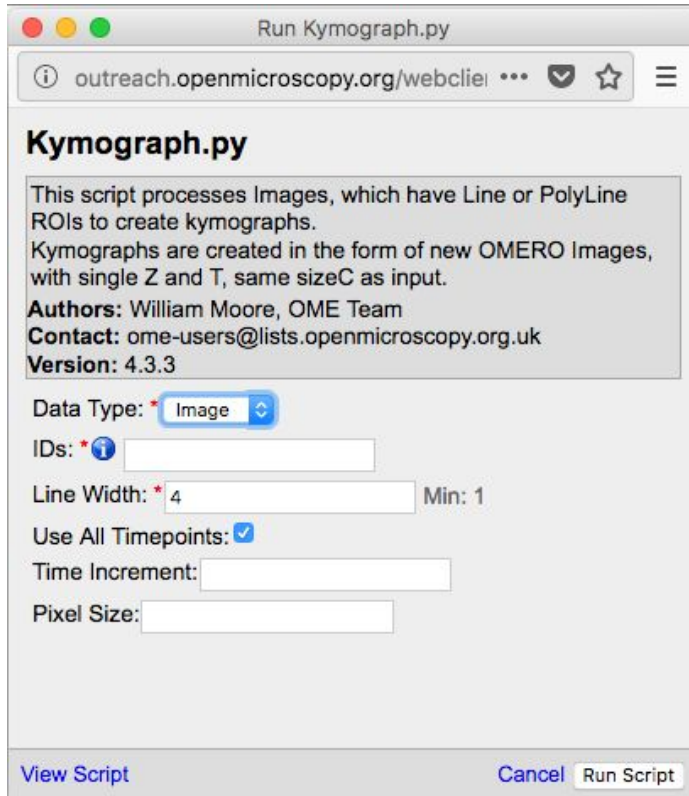


6. Select *workshop_scripts* > Kymograph...
7. The script will create a new image (=Kymograph) where the pixels under the line region you have drawn previously will be collated into this image timepoint by timepoint. The row of pixels from the first timepoint will be on the top of the new Kymograph Image.
8. Note: the direction in which you have drawn the line ROI on the original image matters with respect to the orientation of the stripes composing the Kymograph image. The start of the original line is on the left of the Kymograph Image, the end on the right.
9. Open the new Kymograph image in OMERO.iviewer.
10. Find some tracks (typically red stripes going under angles across the image, see screenshot below).
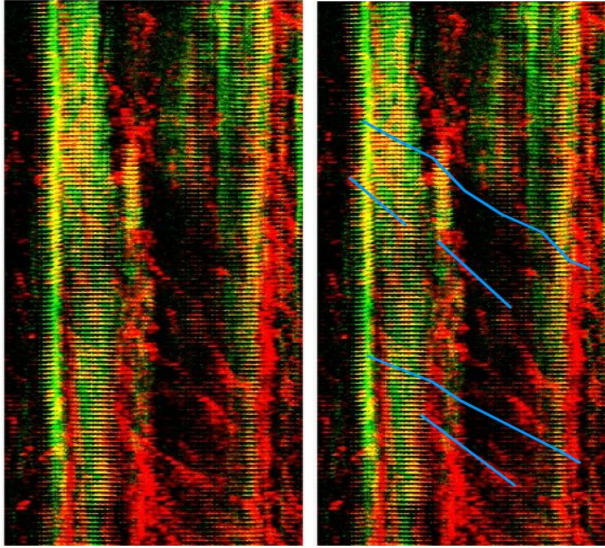
11. Draw some lines over these tracks and save them.
12. Go back to the webclient, select the Kymograph Image and select the script *analysis > Kymograph analysis...*
13. Run this script. The *Kymograph analysis* script will produce a CSV file attachment on the Kymograph Image.
14. Open the CSV in Excel for example and verify the speeds of the observed particles in the original image.

## How to write a server-side script

The server side script follows some simple steps so that a simple UI can be generated automatically. In this section we will show how to write a simple script and upload it to the OMERO.server: The user will specify a Dataset and the script will be uploaded by the OMERO.server administrator (or administrator with restricted privileges) to the OMERO.server. If the script is uploaded as "official", it can be run by all users on the OMERO.server after upload. The script demonstrated here is very simple, it just loads the images contained in a dataset.
See https://raw.githubusercontent.com/ome/training-scripts/v0.5.1/practical/python/server/hello_world.py

1. Click on the link above and copy and paste the script into a text editor of your choice.
2. Study the composition of the script - the script is taking a Dataset ID and produces an output of all the images contained in that dataset. This is of course just a springboard for further work with the images in a more advanced script.
3. (demo only) The script can be immediately uploaded to the OMERO.server in this present state, using

   a. OMERO.insight, the sixth icon from the left, top-left of the UI  (Note that only admins and restricted admins will see this icon in OMERO.insight).
   b. Command line interface (CLI0 using the command.

i. $ bin/omero script upload test-script1.py --official
4. After the demonstrator uploaded the script, you can
   a. Go to OMERO.web and select any dataset in the left-hand tree
   b. Above the central pane, find the "cogs" icon with scripts
   c. Find the newly uploaded script.
   d. Click on the menu item, the script dialog will be already pre-populated with the ID of the selected Dataset.
   e. Click Run.

# How to manage a server-side script

Please refer to https://docs.openmicroscopy.org/omero/5.4.10/developers/scripts/user-guide.html for how to write other simple scripts, execute, edit and delete them.

## OMERO API usage

## OMERO.py scripting - Simple FRAP

**Setup**
- The following OMERO.scripts have been uploaded to the server for this workshop:
  - https://github.com/ome/training-scripts/blob/v0.5.1/practical/python/server/simple_frap.py
  - https://github.com/ome/training-scripts/blob/v0.5.1/practical/python/server/simple_frap_with_figure.py

We will use Jupyter notebooks to introduce the OMERO Python API. See docs for OMERO Python at https://docs.openmicroscopy.org/latest/omero/developers/Python.html.

We will use various code examples from that page to begin a very simple FRAP analysis.

All the examples use the BlitzGateway connection wrapper "conn" to access data in OMERO.
- Go to https://idr-analysis.openmicroscopy.org/training
- Load the notebook  *Files > notebooks > Python > OMEROHelloWorldNotebook.ipynb*. The first step is already loaded for you:
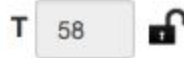
```
from omero.gateway import BlitzGateway
conn = BlitzGateway(raw_input("Username: "), getpass("OMERO Password: "),
host="outreach.openmicroscopy.org", port=4064)
conn.connect()
```

1. Go to webclient, and make sure you are working on your own data.
2. Find the Dataset **FRAP.**
3. Note the ID of the first image in this Dataset.

4. Double-click on the first Image from this Dataset to open it in OMERO.iviewer.
5. Play the timelapse in OMERO.iviewer.
6. Draw in OMERO.iviewer an ellipse ROI on the spot which was bleached.
7. We will use an Ellipse saved on the Image in OMERO.iviewer **without T index set** (spans all T).

<div align="center">T 58 🔓</div>

8. Click on the lock next to the T Box to unlock      .
9. Save the ROI.
10. Go back to the Jupyter notebook. First list the ROIs and Shapes saved on the Image, using the `image_id` of the image you just worked with in iviewer.
11. Enter your username and password when requested.
12. You may with to 'Cut' (remove) the remaining steps from the *HelloWorld* example before adding your own.
13. Add a new step with this code, setting the image_id to the Image you just edited.

```
image_id = 123
roi_service = conn.getRoiService()
result = roi_service.findByImage(image_id, None)
shape_id = None
for roi in result.rois:
    for s in roi.copyShapes():
        shape_id = s.id.val
print shape_id
```

14. With the `shape_id`, we can get pixel intensity stats for the first channel across all time points and make a list of mean intensity values. Add a new step with this code:

```
the_c = 0
image = conn.getObject('Image', image_id)
size_t = image.getSizeT()
meanvalues = []
for t in range(size_t):
    stats = roi_service.getShapeStatsRestricted([shape_id],0, t, [the_c])
    meanvalues.append(stats[0].mean[the_c])
print meanvalues
```

15. We can add these values as a Map Annotation (Key-Value pairs) on the image:

```
import omero
key_value_data = [[str(t), str(meanvalues[t])] for t in range(size_t)]
map_ann = omero.gateway.MapAnnotationWrapper(conn)
namespace = "demo.simple_frap_data"
map_ann.setNs(namespace)
```

```
map_ann.setValue(key_value_data)
map_ann.save()
image.linkAnnotation(map_ann)
```

16. Plot values with Matplotlib:

```
from PIL import Image
import matplotlib
matplotlib.use('Agg')
from matplotlib import pyplot as plt
fig = plt.figure()
plt.subplot(111)
plt.plot(meanvalues)
fig.canvas.draw()
fig.savefig('plot.png')
pil_img = Image.open('plot.png')
pil_img.show()
```

This will create a *plot.png* in the current folder. Click to view it.
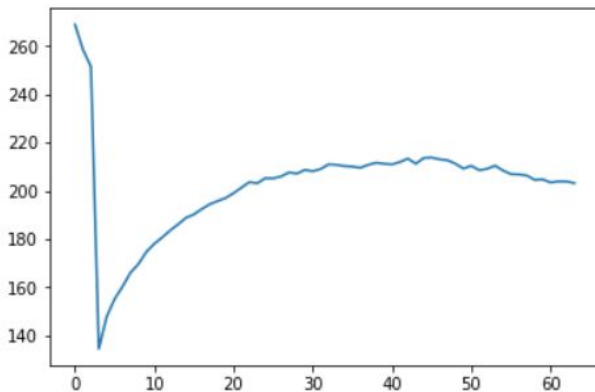


17. Go to the webclient, select the Image you have drawn the ROI on.
18. Note the new Map Annotation (Key-Value pairs).
19. If desired, draw an Ellipse as above (T unset) on additional FRAP Images to analyse.
20. Select the image(s) and Open the script *workshop_scripts > simple frap...*
21. Click on the *View script* link in the bottom-left corner and inspect the script - basically, the script does what you have done in the Jupyter notebook up till now.
22. Run the script, then observe that a new Map Annotation appeared on the image with the FRAP values.
23. Now select the script *workshop_scripts > simple frap with figure...*
24. Inspect this script as well, it is basically an extension of the *simple frap* script with the creation of OMERO.figure at the end.
25. Run the script on the image. Note in *Activities*, the ID of the newly created figure.

26. Open OMERO.figure, and go to *File > Open…* to open the newly created figure (named **FRAP Figure from script**).
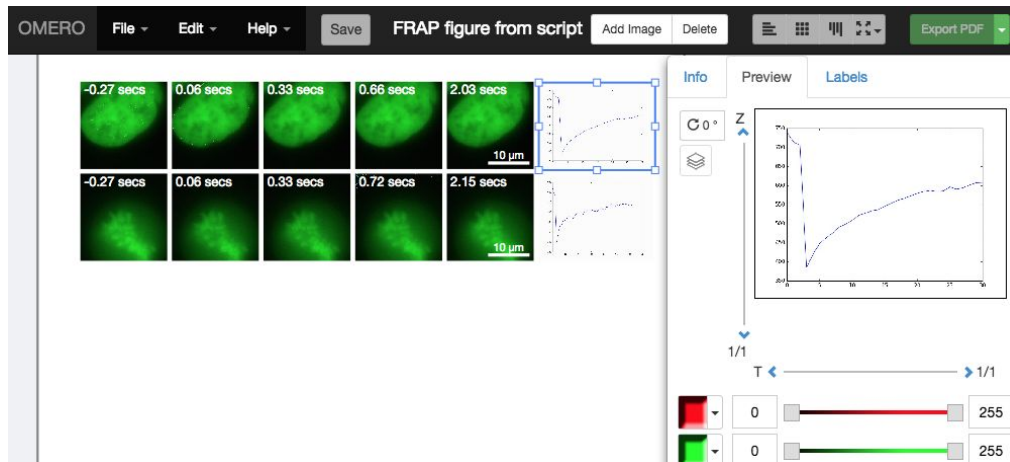


# CellProfiler

**CellProfiler** is a free open-source software for quantitative analysis of biological images. For more details, go to http://cellprofiler.org/.

**Setup:**
CellProfiler **version 3.1.3** has been installed in a Jupyter notebook using docker as described at https://github.com/ome/training-notebooks/releases/tag/v0.4.1. We will use the Python API only. To use CellProfiler using the Python API,  you can see how to set up the environment https://github.com/ome/training-notebooks/blob/v0.4.1/docker/environment-python2-cellprofiler.yml and https://github.com/ome/training-notebooks/blob/v0.4.1/Dockerfile

1.  We will use CellProfiler to analyse a Plate of Images in OMERO. We will run CellProfiler version 3.1.3 "headless" via a Jupyter notebook.
2.  We will use the example pipeline from http://cellprofiler.org/examples/#PercentPositive to analyse RNAi screening data from IDR https://idr.openmicroscopy.org/webclient/?show=screen-102
3.  First, open the webclient and find the Plate belonging to *trainer-1* named *plate1_1_013*.

4.  Go to https://idr-analysis.openmicroscopy.org/training and look under *Notebooks > CellProfiler* for *idr0002.ipynb.*
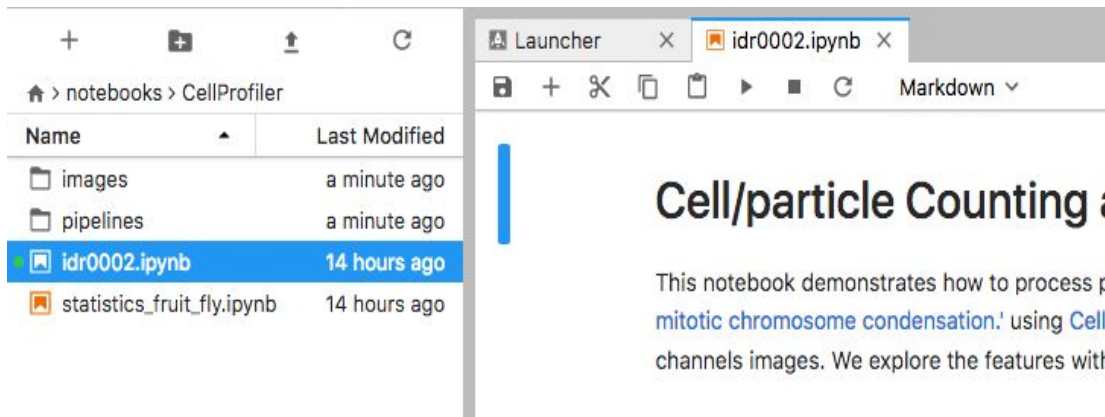


5.  Select the first Step and click on the *Run* button to execute each step in turn.
6.  For the connection to OMERO, you will be ask to enter your login details when running the OMERO credentials cell.
7.  Select the plate in the webclient, find the Plate ID in the right-hand panel and copy this into the *plate_id* variable in the next step of the notebook.
8.  The following step loads the example pipeline and modifies it to remove the modules that are normally used for loading images from disk.
9.  These modules are replaced by the *InjectImage* modules, using numpy planes loaded from OMERO Images.
10. The pipeline is run on a 2-Channel image from each Well in the 96-well plate, generating a CSV file containing rows for different objects identified in the image and columns for various parameters measured.
11. Note that to save time during the workshop, we can run on a subset of all Wells in the plate. To use the first 5 wells, add this line `wells = wells[0:5]` as shown:

```
# create list from generator
wells = list(plate.listChildren())
wells = wells[0:5]
well_count = len(wells)
```

12. CSV files are read into a Dataframe for each image. We add the Image ID and Well ID, as well as the total number of Objects, Cell_Count, to each Dataframe.
13. All the Dataframes are then concatenated into a single Dataframe.

14. We can visualise the data as histograms for each column with `df.hist()`
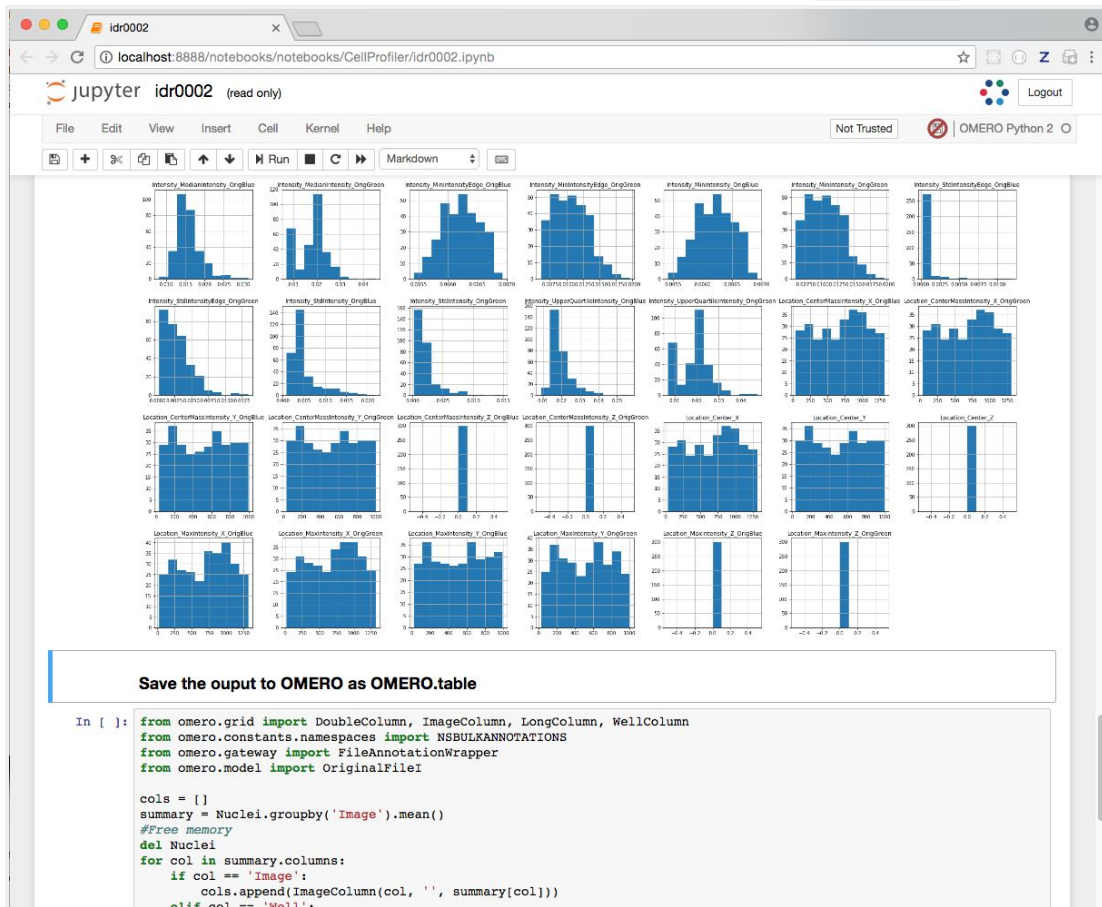


15. Finally, the Dataframe rows are grouped by Image to give an average value per Image of each parameter (column) in the table.
16. This data is saved back to OMERO as an HDF5-based table attached to the Plate, which can be read by other clients.
17. Return to the webclient and select the Plate named *plate1_1_013_previously_analysed*.
18. Select a Well and open the *Tables* pane in the *General* tab in the right-hand panel. This will show all the CellProfiler values for this Well.
19. In the *Thumbnails* chooser at the top-right of the centre panel, select the *Parade* plugin.
20. At the top-left of the centre panel choose *Add filter… -> Table* to filter Wells by the data from CellProfiler.
21. Change the filter from *ImageNumber* to *Cell_Count* (at the bottom of the list).

22. Now you can use a slider to filter Wells by Cell Count.
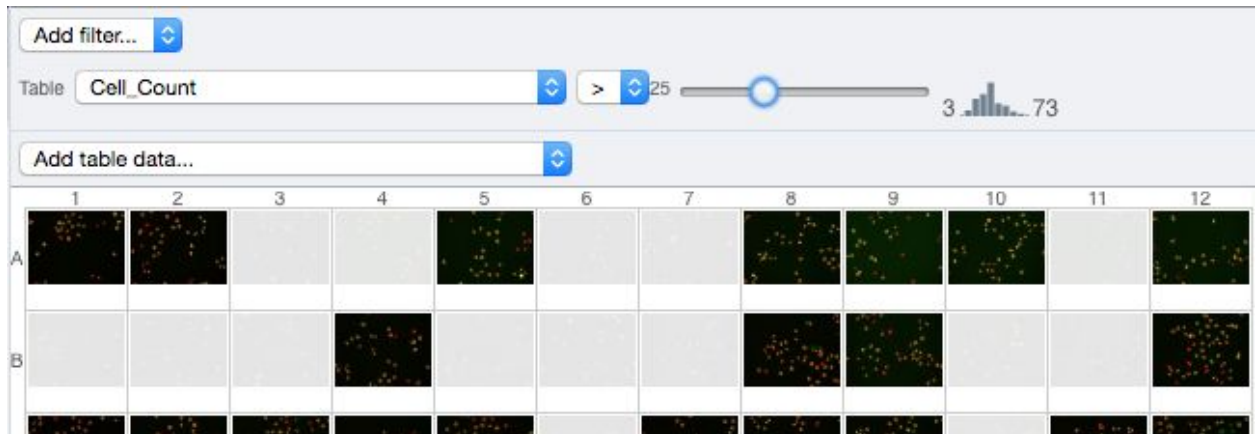


# Analyse data using R

For this section we use R as part of a Jupyter notebook. In order to be able to connect to OMERO from within R we need the romero.gateway and rJava packages. rJava can be difficult to set up, depending on the operating system you use. On the Jupyter server, however, everything has already been installed and we can simply access the notebook in the web browser. Everything we are going to do there can be done in Rstudio as well, if the romero.gateway package and its dependencies are available on your system. For installation instructions go to

romero.gateway github page.

In the notebook we are going to use the idr0021 data. These are super resolution microscopy images showing certain proteins around the centrioles. With these images the authors of the article Subdiffraction imaging of centrosomes reveals higher-order organizational features of pericentriolar material showed that the area around the centrioles has a specific structure. By measuring the shape and diameter of these proteins and comparing them to each other they could show that each protein builds a specific part of this structure, see Figure 1. Instead of manually measuring each centriole ring like the authors did, we are going to try to do this in an automated way in order to show that there is a significant difference between these proteins and create a plot similar to the one seen on Figure 1.

In order to do that we need to…
1. Fetch the images from OMERO. (romero.gateway)
2. Perform image segmentation to identify the protein rings around the centrioles. (EBImage)
3. Calculate the properties (features) of the identified objects (shape, diameter, etc.).  (EBImage)
4. Store these features as R dataframe. (in R)
5. The segmentation and feature calculation is a rather costly process. You don't want to do this all over again. Therefore:

     a.   Save the identified objects (ROIs, region of interests) back to OMERO. (romero.gateway)
     b.   Save the properties back to OMERO. (romero.gateway)
6.  Perform statistical analysis on that data. (in R)

Note: We are going to do the image segmentation and feature calculation on a small subset. Everyone has a dataset called 'R-dataset' which contains four images of the idr0021 data. Use this dataset for learning the segmentation tasks. The full analysis of the idr0021 project has already been done and the result saved as 'Summary from R' table. We are going to load this table in order to do the statistical analysis at the end of the notebook.

- Go to https://idr-analysis.openmicroscopy.org/training .
- In the Files > notebooks > R folder, select the notebook idr0021_Segmentation.ipynb.
- Follow the steps in the notebook.

Further exercise: At the end of the notebook you will have created a plot which is similar to the one shown in Figure 1 of the article. It won't be exactly the same. Can you explain why it is different? One dataset in particular has extreme outliers. Use OMERO.parade to find the images which cause the outliers. What went wrong in these cases? Note: OMERO.parade uses the 'Summary from R' table which is attached to the project, in order to provide enhanced filtering and plotting features.

## Analyze data using MATLAB

Here we demonstrate how to analyze a batch of images associated to the paper Subdiffraction imaging of centrosomes reveals higher-order organizational features of pericentriolar material.
This will show how to connect to OMERO, load data, analyse and save the output back to the server. We will use https://docs.openmicroscopy.org/latest/omero/developers/Matlab.html as a reference.

**Setup**
1.  The OMERO.matlab toolbox and the Image Processing toolbox have been installed.
2.  Make sure that the OMERO.matlab toolbox is on the MATLAB path. To add it to the path, you can
     a.   Launch MATLAB
     b.   Under the *HOME* tab, click on *Set Path* (middle of the top task bar).
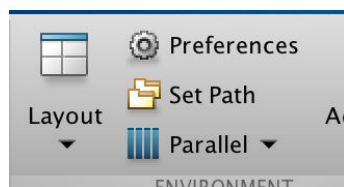


     c.   A Set Path dialog pops up.
     d.   Click on the button *Add with Subfolders...*
     e.   Select the OMERO.matlab toolbox, Click *Open*
     f.   Close the Set Path dialog, you can either save the path for future use or not.

For the exercises below, we will use code examples from the following page
https://docs.openmicroscopy.org/latest/omero/developers/Matlab.html

- The parameter `servername` is outreach.openmicroscopy.org
- The parameter `user` is the username provided i.e. user-x
- The parameter `password` is the password provided

## Exercise

The aim of this exercise is to show how to connect to OMERO and load images in a dataset.
1. Launch MATLAB
2. Create a new Script, name it for example `analyse_dataset.m`.
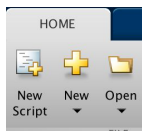


3. When saved, click on the *EDITOR* tab. Note: If you do not have an EDITOR tab in your Matlab, you are probably in a new window after you clicked on *New* button in the taskbar (step above). In such case, you can click on the small blue downward arrow on top right of that window  and select *Dock* to get the EDITOR tab into your workspace. Once done, you can create as many short script snippets as you like in this environment by clicking on a plus to open a new tab in your EDITOR.
4. Create a connection
5. Select the Dataset **CDK5RAP2-C** in the left-hand tree. Note its ID displayed in the right-hand panel.
6. In the same script, load the dataset with all its images using the `getDatasets` function
7. Print out the name of the dataset
8. Iterate through the Images in the dataset. A Java list is returned, you may want to convert it to a Matlab list.
1. For each Image
     a. Retrieve its name and its ID.

Solution:
First part of
https://raw.githubusercontent.com/ome/training-scripts/v0.5.1/practical/matlab/analyze_dataset_save_rois_and_attachments.m

# Analyze metadata using OMERO.parade

First let's look at the analytical results generated using the *Analyze particles* plugin in Fiji

1. Select the Project **idr0021**.
2. Choose the *Parade* option in the centre panel dropdown menu.
3. Expand all Datasets by clicking on the *Open All* button.
4. In the *Add filter...* selection box, select the *Table* item so we can find using the analytical results generated previously:. Choose the *Bounding_Box* item and drag the slider to filter the Images. Note that *PCNT* has the largest number of Images with large ROIs.

        i.      Table  Bounding_Box       &gt;

           3.128400072455406      268.884015083313
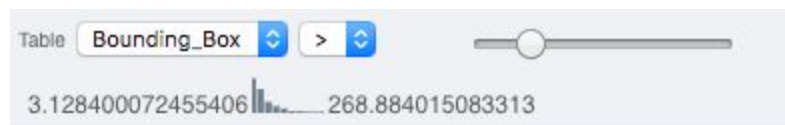
*5.*

        i.

6. Switch to Table layout (middle button) .
7. In the selection box *Add table data*…, select
    a. *Table_Bounding_Box*
    b. *Table_Total_Area*
    c. *Table_Image*
    d. Note that it is currently not possible to remove a column.
8. Click on the name of a column to sort it.
9. Uncheck *Show Datasets* to sort all Images together e.g. by ROI count.

10.    Show Datasets ☐

11. Check the checkbox in each column to show the *Heatmap*. Note the corresponding pattern in the Heatmap.
12. Switch now to the Plot Layout (third button)
13. It takes the table data loaded and plot the values.
14. Filters can be added to plot the relevant results.
15. Try plotting by different Axis values.
16. Closing a Dataset in the left-hand tree removes the values from the plot.
17. Drag to select several outliers.
18. Note that you can use the selected images in right panel to annotate or *Open with....*
19. Choose *Open with Figure…*

Let's now look at the results generated by CellProfiler

1. Return to the webclient and select the Plate named **plate1_1_013**.
2. Select a Well and open the *Tables* pane in the General tab in the right-hand panel. This will show all the CellProfiler values for this Well.
3. In the *Thumbnails* chooser at the top-right of the centre panel, select the *Parade* plugin.
4. At the top-left of the centre panel choose *Add filter… -> Table* to filter Wells by the data from CellProfiler.
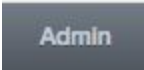5. Change the filter from *ImageNumber* to *Cell_Count* (at the bottom of the list).
6. Now you can use a slider to filter Wells by Cell Count.
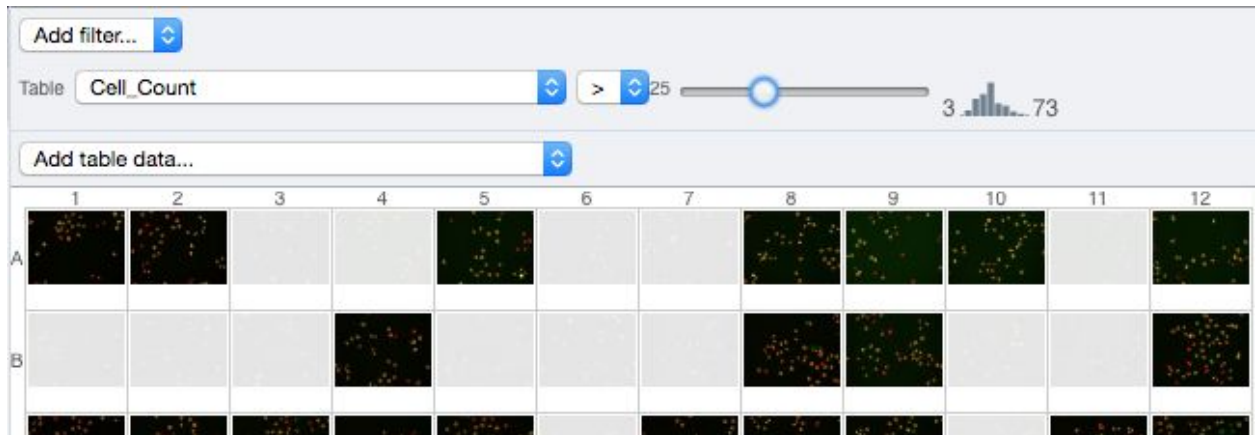


# Administrate Groups and Users

Administrate using the Web Interface

Most of following tasks below can only be done by users with some administrator privileges. The following steps will be done by the trainer.  Below are some useful links to know more about permissions and administrator privileges:

- https://docs.openmicroscopy.org/latest/omero/sysadmins/server-permissions.html
- https://docs.openmicroscopy.org/latest/omero/sysadmins/restricted-admins.html

1. Go to https://outreach.openmicroscopy.org/
2. Login using the username and password

3. In the toolbar, click the *Admin* button    **Admin**    .
4. Managing Groups:
   a.  Click on the *Groups* tab.
   b.  You can search for groups if desired
   c.  To create a new Group:

   i. Click on the *Add new Group* button

   ii. The *Name* and *Permissions* fields are mandatory.

   iii. Click *Save*

  d. To edit a Group:

   i. Click on the Pencil button ✏

   ii. You can add or remove members or group's owners. Change permissions etc..

   iii. Make sure that the data owned by a user is moved or transferred to another user before removing the user from the group

   iv. Click *Save*

5. Managing Users:

  a. Click on the *Users* tab

  b. You can search for users if desired

  c. How to identify users:

   i. Users with administrators privileges with ✂

   ii. Active users with 🔳

   iii. Inactive users with 🔒

   iv. LDAP users with 🛑

  d. To create a new User:

   i. Click on the *Add new User* button

   ii. Mandatory fields are highlighted in red

   iii. You can select the role of the user

   iv. Click *Save*

  e. To edit a User:

   i. Click on the Pencil button ✏

   ii. You can add or remove the user to/from a group. Modify the roles etc..

   iii. Click *Save*

  f. Creating an administrator privileges allow to give some limited rights to some trusted users e.g. to allow a facility manager to import data for other users. It is currently preferable to create users with such role via the Web Interface.

6. Note that a user can manage his/her settings.

  a. In the top-right corner of the webclient, click on your name

  b. In the dropdown menu, click on *User settings*

  c. In the dialog that pops up, the user can change password, default group i.e. the group he/she will log in by default etc.

## Administrate using the Command Line Interface

1. Managing Groups:

  a. By default when creating a group, its permissions level is set to private. To create a new **read-annotate** group *Basel*, run:

```
$ bin/omero group add Basel --type=read-annotate
```

Or

```
$ bin/omero group add Basel --perms='rwra--'
```

b. To list all the groups and save the output for example in a CSV file:

```
$ bin/omero group list --style csv > groups.csv
```

c. To add an existing user *user-1* to the *Basel* group and make him/her a group owner (the option `--as-owner` is not needed when adding a member), run:

```
$ bin/omero group adduser user-1 --name=Basel --as-owner
```

d. Let's add *trainer-1* as an owner of the group too:

```
$ bin/omero group adduser trainer-1 --name=Basel --as-owner
```

e. To remove *user-1* from the list of owners (*user-1* will still be a member of the *Basel* group):

```
$ bin/omero user leavegroup Basel --name=user-1 --as-owner
```

f. To remove *user-1* from the *Basel* group, run:

```
$ bin/omero group removeuser user-1 --name=Basel
```

g. To edit the *Basel* group:

    i. First determine its ID:

```
$ bin/omero group info --group-name Basel
```

```
 id  | name  | perms  | ldap  | # of owners | # of members
-----+-------+--------+-------+-------------+--------------
 653 | Basel | rwra-- | False | 0           | 0
```

    ii. Change its name:

```
$ bin/omero obj update ExperimenterGroup:653 name='Basel-1'
```

    iii. Let's reset the name back to "Basel" to simplify the rest of the workflow

    iv. Change groups permissions to **read-write**:

```
$ bin/omero group perms --perms='rwrw--' --name='Basel'
```

2. Managing Users:

a. Create a new user *user-basel* and add the user to the *Basel* group:

```
$ bin/omero user add user-basel Jan Purkyne --group-name Basel
```

b. Let's now add the user to another group:

```
$ bin/omero user joingroup Lab1 --name=user-basel
```

c. To edit the user and for example add an email address

    i. First determine the user's ID:

```
$ bin/omero user info --user-name user-basel
```

    ii. Add an email address:

```
$ bin/omero obj update Experimenter:123 email=jpurkyne@demo.co.uk
```

d. Make a user inactive:

    i. User cannot be deleted but it is possible to prevent a user from logging in. For that we need to remove the user from the *user* group (internal OMERO group).

```
$ bin/omero user leavegroup user --name=user-basel
```

e. To reactivate the user:

```
$ bin/omero user joingroup user --name=user-basel
```

       f.   LDAP authentication, for information only:
- i.   It is possible to convert non LDAP users to LDAP authentication using the command `bin/omero ldap setdn`
- ii.   When using LDAP as an authentication backend, user when they log in will be added to the internal OMERO group called `default` unless they have already been added to a given group. To add a user before they have ever logged in to OMERO, run:
  1. First create the user
     ```
     $ bin/omero ldap create enoether
     ```
  2. Then add the user to the *Basel* group
     ```
     $ bin/omero group adduser enoether --name=Basel
     ```

## Search data

In order to move data, change ownership, etc., the first step is to find the data. Depending on the complexity of the search, this can be achieved either one of the commands:

- `$ bin/omero search`
- `$ bin/omero hql`

As an administrator, let's find the Projects starting with **idr** and retrieve only their IDs:

```
$ bin/omero search Project "idr*" --ids-only
```

The `bin/omero hql` command is usually used for more complex query. The number of items is by default set to 25 to increase the number, the option ``--limit`` must be specified.

In an institution, administrators with restricted privileges (restricted administrators) will typically be imaging facility managers, image analysts, or anybody who needs to organize users and data of others in OMERO.
The following commands will be run by a restricted administrator *trainer-1* for another user.

## Create data

For more information http://docs.openmicroscopy.org/latest/omero/users/cli/containers-annotations.html

1. The user `trainer-1` now logs in as `user-basel,` the data will be owned by `user-basel`:
   ```
   $ bin/omero --sudo trainer-1 -u user-basel login
   ```
2. Create a Project:
   `$ bin/omero obj new Project name="Training-Project"`
3. Create a new Dataset:
   `$ bin/omero obj new Dataset name="Training-Dataset"`
4. To update the Dataset:
   ```
   $ bin/omero obj update Dataset:501 description="Basel Day 2"
   ```

5. Link the Project and the Dataset:
   `$ bin/omero obj new ProjectDatasetLink parent=Project:101 child=Dataset:501`
6. Create a Map Annotation:
   - `$ bin/omero obj new MapAnnotation name=used-antibodies`
   - `$ bin/omero obj map-set MapAnnotation:301 mapValue host rabbit`
7. Link the Map Annotation to the Dataset:
   `$ bin/omero obj new DatasetAnnotationLink parent=Dataset:501 child=MapAnnotation:301`

## Delete data

For more information http://docs.openmicroscopy.org/latest/omero/users/cli/delete.html

1. The `delete` command removes **entire graphs of data** based on the ID of the top node.
2. Run the command as *trainer-1.*
3. It is recommended to first run the command with the `--dry-run` option and `--report` to list the objects that will be deleted.
4. To delete the Project/Dataset graph:
   `$ bin/omero delete Project:101 --report --dry-run`
5. To delete the graph and not delete the map annotation:
   `$ bin/omero delete Project:101 --exclude MapAnnotation --report --dry-run`

## Move data between groups

The owner of the data will usually use OMERO.web or OMERO.insight to move data between groups. This is not covered by this workshop and information can be found at https://help.openmicroscopy.org/sharing-data#moving

For more information https://docs.openmicroscopy.org/latest/omero/users/cli/chgrp.html

1. The `chgrp` command moves **entire graphs of data** based on the ID of the top node.
2. Run the command as *trainer-1.*
3. It is recommended to first run the command with the `--dry-run` option and `--report` to list the objects that will be moved.
4. First determine the ID of the target group. We want to move data from the *Lab1* group to the *Basel* group:
   `$ bin/omero group info --group-name Basel`
5. To move the Project/Dataset/Image graph to *Basel*::
   `$ bin/omero chgrp Group:123 Project:123 --report --dry-run`
6. To move the graph and not move the map annotation for example:
   `$ bin/omero chgrp Group:603 Project:101 --exclude MapAnnotation --report --dry-run`

7. To move several containers at once:

```
$ bin/omero chgrp Group:603 Project:101,103,104 --report --dry-run
```

## Change data ownership

For more information https://docs.openmicroscopy.org/latest/omero/users/cli/chown.html
Change ownership is currently not available neither in OMERO.web nor OMERO.insight.
The command can be run by administrators, restricted administrators or group owners.

1. The chown command transfer ownership of **entire graphs of data** based on the ID of the top node.
2. Run the command as *trainer-1.*
3. To transfer the ownership of the Project/Dataset graph to *trainer-1*:

```
$ bin/omero chown trainer-1 Project:101 --report --dry-run
```

## Upload data

1. It is possible to upload local files to the OMERO server and link them to an Object. This could be for example the output of an analysis.
   a. Upload, for example, a CSV file and attach to a Project:

   ```
   $ bin/omero upload analysis.csv
   ```

   b. The ID of the original file is returned, create a FileAnnotation:

   ```
   $ bin/omero obj new FileAnnotation file=OriginalFile:123
   ```

   c. Link it to the Project:

   ```
   $ bin/omero obj new ProjectAnnotationLink parent=Project:101
   child=FileAnnotation:123
   ```

2. Administrators or restricted administrators can also upload a script to the server so it can be used by all the users. It is important to be in the correct directory when uploading so that the script is uploaded with the path that will show up in the UI e.g. basel/Channel_Offsets.py
   a. Go to the parent directory of the basel folder. Then run:

   ```
   $ bin/omero script upload basel/Channel_Offsets.py --official
   ```

**Exercise:**
Use bash to automate the workflows. Include the following steps:
- Log in as a restricted administrator and perform task for another user.
- Create a dataset.
- Import an image into the dataset.
- Upload a file.
- Create a file annotation and link it to the imported Image.

One solution:
https://github.com/ome/training-scripts/blob/v0.3.0/practical/bash/import_as_create_annotation.sh

## Server Installation

1. System requirements:
   a. https://docs.openmicroscopy.org/latest/omero/sysadmins/system-requirements.html
2. Ansible way. CentOS 7 only
   a. Playbook example
      https://github.com/ome/ansible-examples-omero/blob/master/public-user/playbook.yml
3. "Standard" way
   a. https://docs.openmicroscopy.org/latest/omero/sysadmins/unix/server-installation.html
   b. https://docs.openmicroscopy.org/latest/omero/sysadmins/unix/install-web/web-deployment.html
4. CLI-plugin
   a. https://pypi.org/project/omero-cli-render/
5. Web apps
   a. https://pypi.org/project/omero-figure/
   b. https://pypi.org/project/omero-iviewer/
   c. https://pypi.org/project/omero-fpbioimage/
   d. https://pypi.org/project/omero-parade/
6. LDAP
   a. https://docs.openmicroscopy.org/latest/omero/sysadmins/server-ldap.html
   b. https://docs.openmicroscopy.org/latest/omero/developers/Server/Ldap.html