Welcome!

This material is the result of a collaborative work. Thanks the Galaxy Training Network and all the contributors!
Visualizations: generic plugins
Questions

- How can visualization plugins benefit science?
Objectives

- Implement a first Galaxy visualization
- Understand the client side vs. server side principle
Why visualizations?

X mean: 54.26, X SD: 16.76
Y mean: 47.83, Y SD: 26.93
Corr: -0.06
Why visualizations?

Bam files - textual
Why visualizations?

Bam files - visualized
Types of visualizations in Galaxy

- **Trackster** - built-in genome browser
- **Display applications**
  - UCSC Genome Browser
  - IGV
- **Galaxy tools**
  - JBrowse
  - Krona
- **Visualization plugins**
  - Charts
  - Generic
- **Interactive Environments**
  - Jupyter/Rstudio
  - IOBIO (bam/vcf visualizations)
  - Phinch (metagenomics visualizations)
Which should I use?

Only available on an external website?  
- Yes → Display Application  
- No  
  Needs to be served?  
    - Yes → Interactive Environment  
    - No  
      Computationally intensive?  
        - Yes → Regular Tool  
        - No  
          Written in JavaScript?  
            - Yes → Charts Plugin  
            - No → Generic Plugin
Plugins

- Visualizations button on history items
- Different options depending on file type
Plugins: Charts

- **Bar diagrams**
  - Regular (NVD3)
  - Stacked (NVD3)
  - Horizontal (NVD3)
  - Stacked horizontal (NVD3)

- **Others**
  - Line with focus (NVD3)
  - Line chart (NVD3)
  - Scatter plot (NVD3)
  - Heatmap (Custom)

- **Area charts**
  - Regular (NVD3)
  - Expanded (NVD3)
  - Stream (NVD3)
  - Pie chart (NVD3)

- **Data processing (requires ‘charts’ tool from Toolshed)**
  - Histogram (NVD3)
  - Discrete Histogram (JaPlot)
  - Box plot (JaPlot)
  - Clustered Heatmap (Custom)
Plugins: Charts

- Allows exporting screenshots

- For more information, see the charts plugin development slides and tutorial
Plugins: Generic

- Elements of visualization plugin:
  - Configuration (name, description, datatype)
  - Template
  - Static resources (js, css, images)

- Project Structure
  - located in $GALAXY_ROOT/config/plugins/$PLUGIN_NAME
Plugins: XML file

$ touch $GALAXY_ROOT/config/plugins/$PLUGIN_NAME/config/$PLUGIN_NAME.xml

- Hooks visualizations into Galaxy

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE visualization SYSTEM "../visualization.dtd">
<visualization name="DrawRNA.js">
  <data_sources>
    <data_source>
      <model_class>HistoryDatasetAssociation</model_class>
      <test type="isinstance" test_attr="datatype" result_type="datatype">sequence.RNADotPlotMatrix</test>
      <test type="isinstance" test_attr="datatype" result_type="datatype">sequence.Fasta</test>
      <test type="isinstance" test_attr="datatype" result_type="datatype">sequence.DotBracket</test>
      <test type="isinstance" test_attr="datatype" result_type="datatype">tabular.ConnectivityTable</test>
      <to_param param_attr="id">dataset_id</to_param>
    </data_source>
  </data_sources>
  <params>
    <param type="dataset" var_name_in_template="hda" required="true">dataset_id</param>
  </params>
</visualization>
```

- Data types must match with class names in $GALAXY_ROOT/lib/galaxy/datatypes/
Plugins: Mako file

- For templates of HTML files:

  ```
  $ touch $GALAXY_ROOT/config/plugins/$PLUGIN_NAME/templates/$PLUGIN_NAME.mako
  ```

- Link to data types and HTML template file
  - Every invocation of visualization: template compiled
Plugins: Mako file

- Access to the history item *(hda)* **pre** template compilation:
  - `$hda.file_name`
  - `$hda.metadata.dbkey`
  - Avoid (unnecessary) copies of whole files
- Access to *hda post* template compilation -> via URLs
  - Download data file by browser (client side)
    - Inconvenient for large files:
      - Indices and query protocols (e.g. DAS protocol)
  - Resolving history uid to hash
  - Correction for extended root URLs:

```
https://bioinf-galaxian.erasmusmc.nl/galaxy/
```
Plugins: Mako syntax

- Access to *hda post* template compilation:
  - Obtain hash value used in urls `hdadict['id']`
  - Essential Python variables:

```python
# Generates hash (hdadict['id']) of history item
hdadict = trans.security.encode_dict_ids( hda.to_dict() )

# Finds the parent directory of galaxy (/, /galaxy, etc.)
root = h.url_for('/')

# Determines the exposed URL of the ./static directory
app_root = root + 'plugins/visualizations/' + visualization_name + '/static/

# Actual file URL:
file_url = root + 'datasets/' + hdadict['id'] + '/display?to_ext=' + hda.ext;
```
Plugins: Javascript and jQuery

- HTML / JS implementation make often use of jQuery
  - Galaxy ships with jQuery

```html
<script type="text/javascript" src="${root}/static/scripts/libs/jquery/jquery.js"></script>
```
Plugins: Generic template

```html
<!DOCTYPE HTML>

```
Plugins: Static files

- External libraries and static files go in ./static dir:

```html
17 | <script type="text/javascript" src="$\{app_root\}/cool_vis_plugin.js" />
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td></td>
<td>Folder</td>
</tr>
<tr>
<td>static</td>
<td></td>
<td>Folder</td>
</tr>
<tr>
<td>cool_vis_plugin.js</td>
<td>187.9 kB</td>
<td>Program</td>
</tr>
<tr>
<td>templates</td>
<td></td>
<td>Folder</td>
</tr>
</tbody>
</table>
Plugins: Summary

- Galaxy visualization plugins:
  - Datatype-specific
  - Written in HTML5 / JS
  - Require minor python / Galaxy ecosystem knowledge
  - Have access to the whole Galaxy system
    - pre-compilation: python
    - post-compilation: JS / API
  - Can be installed without additional configuration
  - Can not be connected to workflows
  - Can not (out of the box) save and share
Plugins: More tips and tricks

- Galaxy has UI guidelines with corresponding CSS
  (https://wiki.galaxyproject.org/VisualizationsRegistry/Cookbook)
- Look at existing plugins:
  - Galaxy mainline
    - https://github.com/bgruening/galaxytools/tree/master/visualisations
- API is now very extensive, everything is accessible with jQuery!
### Key points

- Visualizations require a different way of thinking: server and client side; downloading files rather than system level access
- Interactivity is what makes visualizations different from static tools
- Requires understanding of both the Galaxy ecosystem as well as HTML5/JS
- Performance is more important than for static Galaxy tools
🎉 Congratulations

on successfully completing this tutorial!

Found a typo? Something is wrong in this tutorial?
Edit it on GitHub