Galaxy

PROJECT

Introduction to Writing Galaxy Tools & Publishing in Galaxy Tool Shed

Martin Čech, Björn Grüning, Dan Blankenberg, Dave Bouvier, John Chilton, Peter Cock, Eric Rasche, Nicola Soranzo

http://bit.ly/trainhome

The Resources

- the VM can be obtained:
  - travelling USB sticks with the planemo_machine.ova image - requires VirtualBox (https://www.virtualbox.org/)
  - https://images.galaxyproject.org/planemo/latest.ova
- this slidedeck at http://bit.ly/traindeck
- the training repo at http://bit.ly/trainhome
- Wireless: gcc2015 / gcc2015psk

Training Plan

1. Why are we going to do this?
2. What are we going to do?
3. Doing it.
4. Looking back at what we have done.

http://bit.ly/trainhome

The Typical Problems

You have written a Python script to do something cool and you want to share it with command-line averse colleagues.

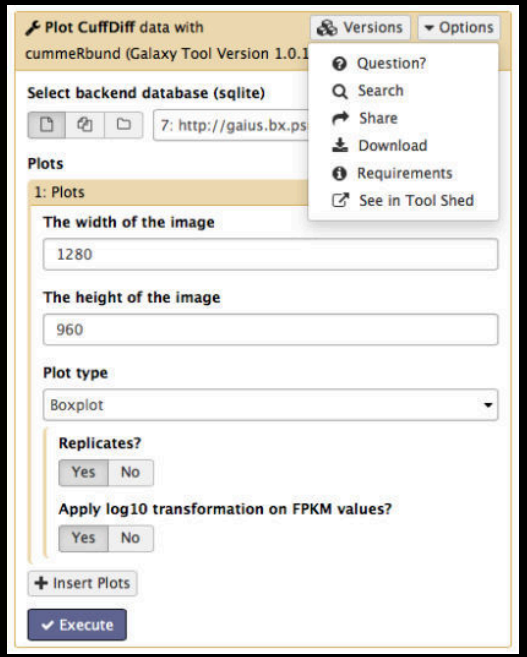
You are missing a tool in Galaxy that would allow you to construct the best workflow.

The Solution

Make a Galaxy Tool.

Why Galaxy Tool?

So people can...

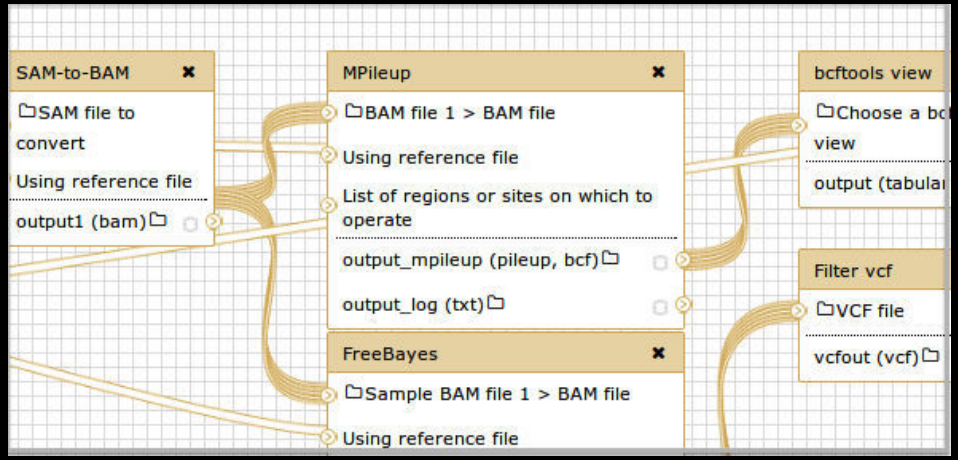


...instead of

http://bit.ly/trainhome

Why Galaxy Tool?

So people can...



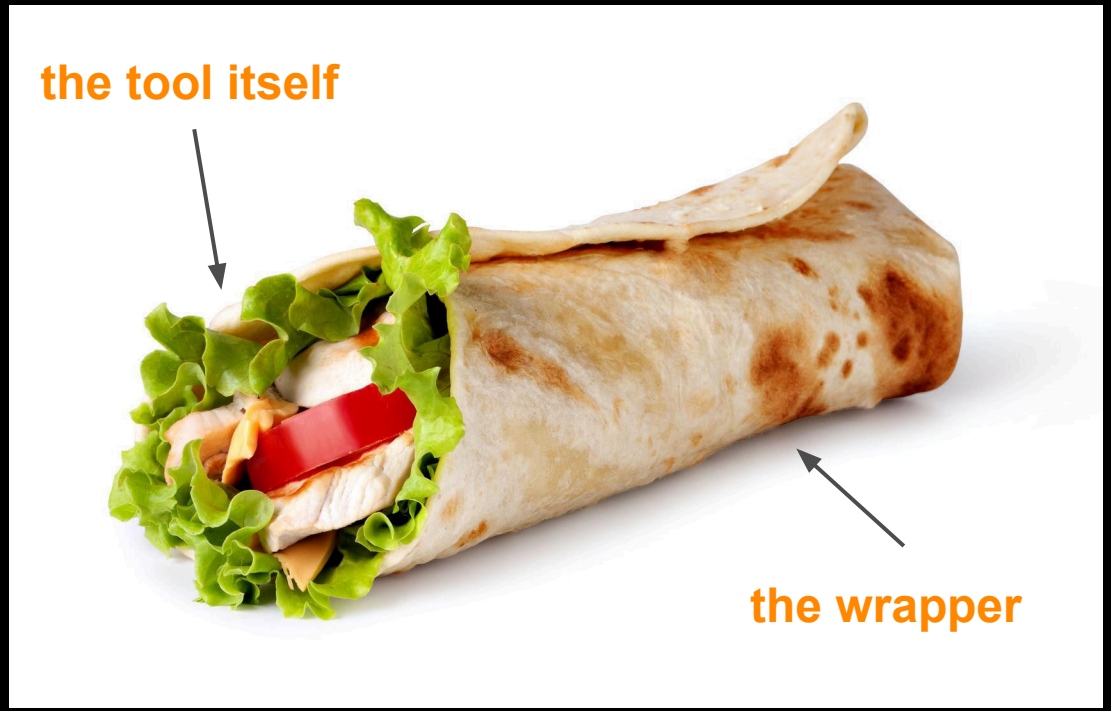
...instead of

```
$iOlll00oil0ioIlo = 1;
$user = 'chippy';
$pass = '1337';
$iIiliiiiii1OoIlIl = '#990000';
 if($_GET['id']== 'logout') {Logout();} if(!($_GET['id']
== 'sshSession'))
 {echo CSS($iIiliiiiii1OoIlIl);}

else if($_GET['id'] == 100){echo "";} else
if($_GET['id'] == 'Delete'){Suicide();}

function iIlli0Il0iO0iooi($file,$per) {
if(function_exists('chmod')){$try =
chmod($file,$per); } if(!$try){$try =
Exe("\143h\x6do\144 $per $file"); }
if($try){return true;} else{return false;} } function
showUsers() { if($rows
= Exe('cat /etc/passwd')){echo $rows;} elseif($rows=
Exe('cat /etc/domai
nalias')){echo $rows;} elseif($rows= Exe('cat
/etc/shadow')){echo $rows;}
elseif($rows= Exe('cat /var/mail')) {echo $rows;}
elseif($rows= Exe('cat
/etc/valiases')) {echo $rows;}
elseif(file_exists('/etc/passwd')) { for
($uid=0;$uid
```

http://bit.ly/trainhome

Galaxy Tool Concepts

wrap tool's inputs and outputs in the wrapper



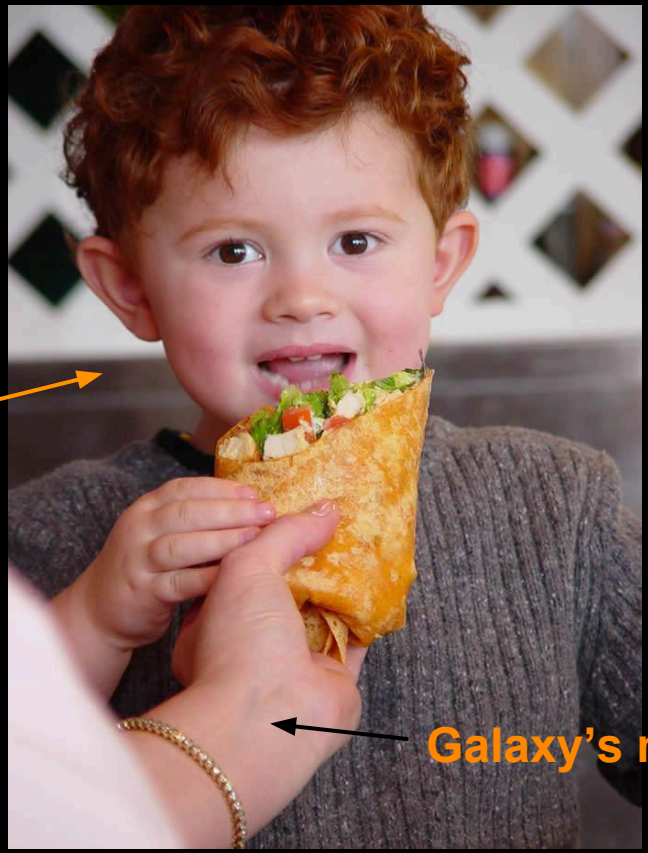
http://bit.ly/trainhome

picture from http://www.mymealchoice.com.au/wp-content/uploads/2014/09/32149501_ml.jpg

Galaxy Tool Concepts

Galaxy will consume the whole thing and make use of the tool.

this fella is Galaxy



http://bit.ly/trainhome

picture from http://www.snakesandslides.co.uk/wp-content/uploads/2011/03/18042_5487_Child-eating-wrap.jpg

Tooldev Jargon

these are used interchangeably

Galaxy Tool ~= wrapper == tool definition == xml file
Tool Shed == shed == TS

Main Tool Shed == MTS == toolshed.g2.bx.psu.edu
Test Tool Shed == TTS == testtoolshed.g2.bx.psu.edu

the tool for making Galaxy Tools

http://bit.ly/trainhome

Turn on your engines!

- start the VM
  - install https://www.virtualbox.org/
  - dblclick the .ova image to load
  - run from the Virtual Box interface

- or run a Planemo appliance by following http://planemo.readthedocs.org/en/latest/appliance.html

Navigating around the VM

- username/password = ubuntu/ubuntu
- clipboard is shared in between VM and your PC
- Terminal
  - ctrl+shift+C = copy; +V = paste
- Firefox, Atom
  - ctrl+C = copy; +V = paste
- Galaxy running at http://localhost:80
  - folder with tools at /opt/galaxy/tools is autoloaded
- Tool Shed running at http://localhost:9009
- various useful links on virtual machine's desktop

Planemo Basics

show me all the commands

```
$ planemo --help
```

show me full help for one command

```
$ planemo <command> --help
```

the head tool

given file.txt looks like this:

```
chr7 56632 56652 310 +
chr8 56736 56756 354 +
chr9 56761 56781 220 +
chr10 56772 56792 372 +
chr11 56775 56795 207 +
```

```
$ head --lines 2 file.txt
chr7 56632 56652 310 +
chr8 56736 56756 354 +
```

```
<tool id="tp_head_tool" name="Select first" version="0.1">
    <description>lines from a dataset (head)</description>
    <command>
    <![CDATA[
            head
                --lines
                $complement$count
                '${infile}'
                > '${outfile}'
    ]]>
    </command>
    <inputs>
        <param name="infile" type="data" format="txt" label="File to select" />
        <param name="complement" type="select" label="Operation">
            <option value="">Keep first lines</option>
            <option value="-">Remove last lines</option>
        </param>
        <param name="count" type="integer" size="5" value="10"
            label="Number of lines" help="These will be kept/discarded depending on 'operation'. (--lines)" />
    </inputs>
    <outputs>
        <data name="outfile" format_source="infile" metadata_source="infile"/>
    </outputs>
    <help>
<![CDATA[
**What it does**
This tool outputs specified number of lines from the **beginning** of a dataset
]]>
    </help>
</tool>
```

Adding a tool to Galaxy

1. create an empty file named `head.xml` in your /opt/galaxy/tools directory on VM
2. grab contents of `head.xml` from http://bit.ly/trainhome
3. copy them to your new file
4. open browser and go to `localhost` in the address bar

Let's wrap seqtk!

seqtk is written by Heng Li https://github.com/lh3/seqtk

"processing sequences in the FASTA or FASTQ format"

```
$ seqtk seq -a 2.fastq > 2.fasta
```

Obtaining seqtk

```
$ brew install homebrew/science/seqtk

$ seqtk
Usage:   seqtk <command> <arguments>
Version: 1.0-r77-dirty

Command: seq       common transformation of FASTA/Q
         comp      get the nucleotide composition of FASTA/Q
         sample    subsample sequences
         subseq    extract subsequences from FASTA/Q
         ...
```

Wrapping seqtk

Subset to wrap: Convert FASTQ to FASTA

```
$ seqtk seq -a 2.fastq > 2.fasta
```

You can grab 2.fastq at http://bit.ly/trainhome or grab any other fastq file you want.

Make sure to run the above command at least once so you have test data to use further.

Initializing seqtk wrapper

Using Planemo we can avoid writing boilerplate code

```
$ planemo tool_init
```

from now on we will follow the official docs starting at http://bit.ly/planemowrite

Publishing to the Tool Shed

We will publish your seqtk to a your local Tool Shed at http://localhost:9009 in your VM

For that we need its dependency - the seqtk package

Publishing to the Tool Shed

```
$ planemo project_init \
--template package_seqtk_1_0_r75 \
package_seqtk_1_0_r75
```

Now we will follow instructions at http://bit.ly/planemoshed

Extra tasks for the quick

- Wrap another function (subseq, trimfq...) of seqtk
- Update the description of your Tool Shed repository using shed_update
- Publish your seqtk wrapper on GitHub (see https://github.com/galaxyproject/tools-iuc/tree/master/tools/seqtk as example)
- Compare your wrapper with the one by Eric Rasche above
- Write another test case for your wrapper

Extra Slides

How to Write a Galaxy Tool

~~the simple case~~ the best practice

1. understand what the tool does
2. use Planemo
3. wrap inputs and outputs using tool definition file
4. try it in Galaxy
5. write tests

How to Publish in the Tool Shed

the complex case

- ➔ create repository on web for tool and every dependency
- ➔ fill information
- ➔ upload every dependency
- ➔ upload your tool
- ➔

Buying Your Own Planemo

via homebrew/linuxbrew:

```
$ brew install galaxyproject/tap/planemo
```

via pip

```
$ pip install planemo
```

The Vagrant Training Setup 1/2

training home: http://bit.ly/trainhome

1. install Vagrant on your computer
2. get https://images.galaxyproject.org/planemo/latest.box
3. initialize by following http://bit.ly/trainmachine
4. obtain cli by visiting localhost:8010/ide/

The Vagrant Training Setup 2/2

now you have:

1. running local Galaxy
2. running local Tool Shed
3. cli to run Planemo
4. sync'd folder to your computer (you can use your editor)