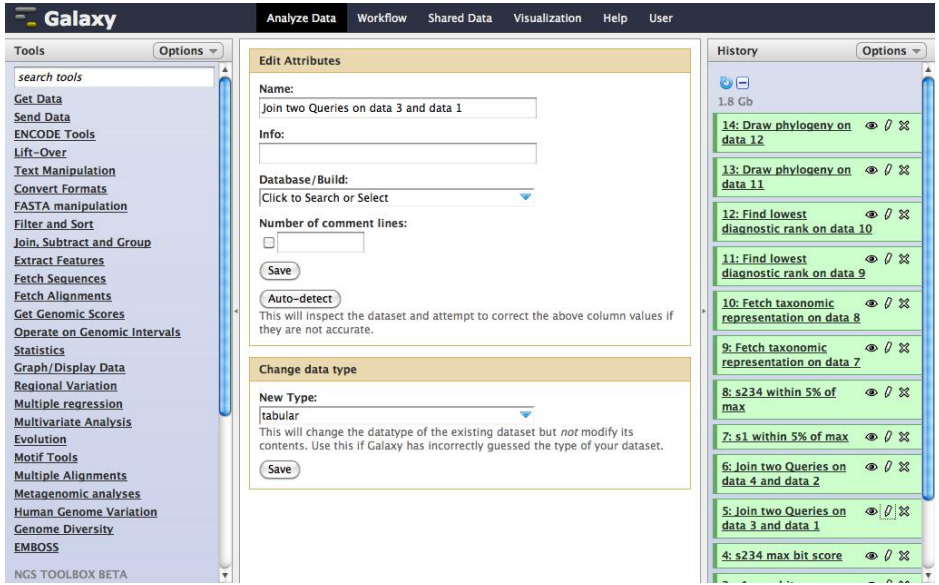
Leibniz Institute on Aging –
Fritz Lipmann Institute

Galaxy Workflow Generator and Galaxy Flavors

Konstantin Riege

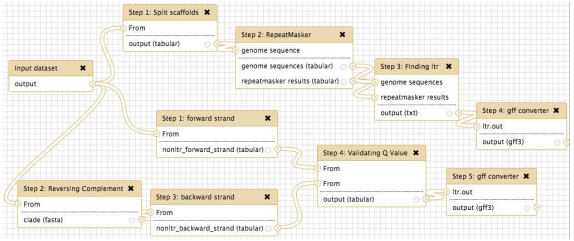
Galaxy Workflow Generator

Tools



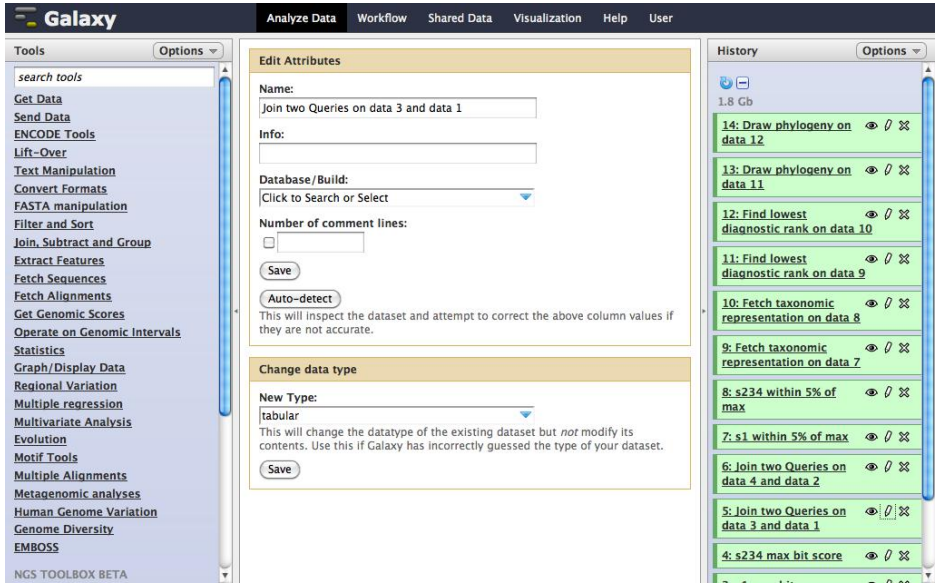
Histories

Galaxy Workflow Generator



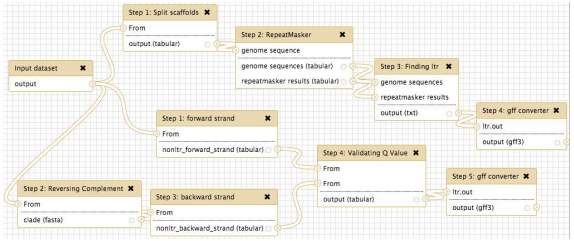
Workflows

Tools

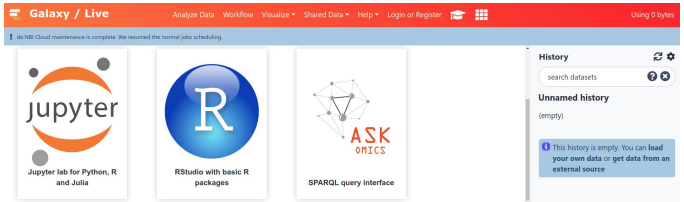


Histories

Galaxy Workflow Generator



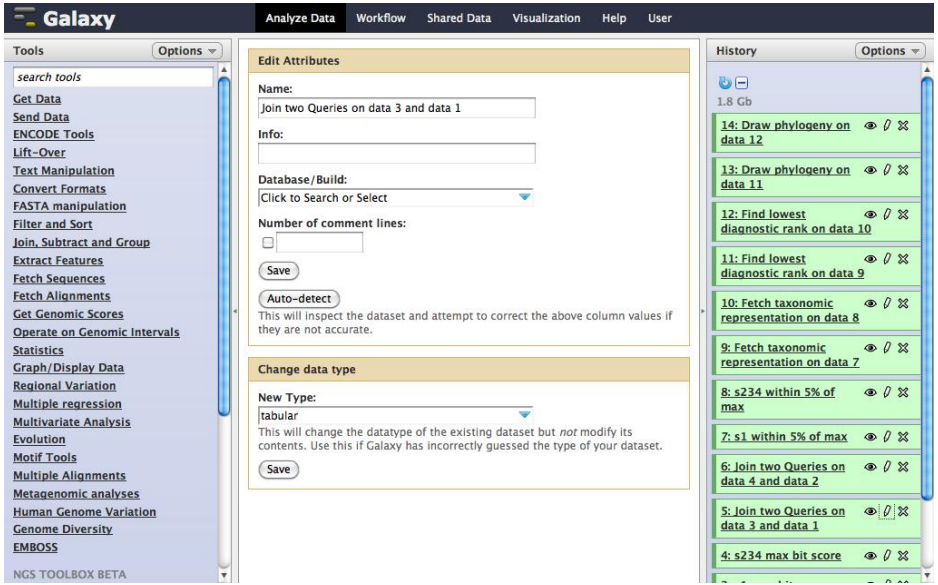
Workflows



Interactive environments

https://live.usegalaxy.eu

Tools

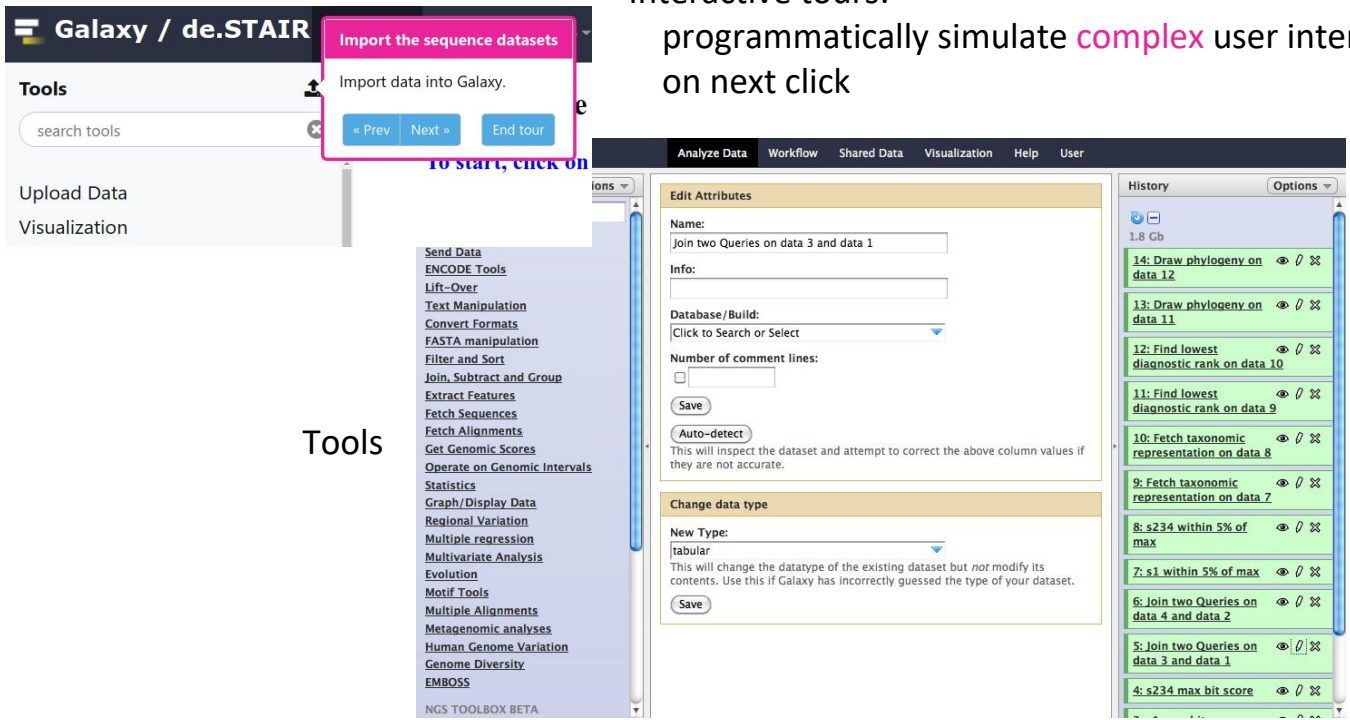


Histories

Galaxy Workflow Generator

Interactive tours:

programmatically simulate complex user interaction on next click

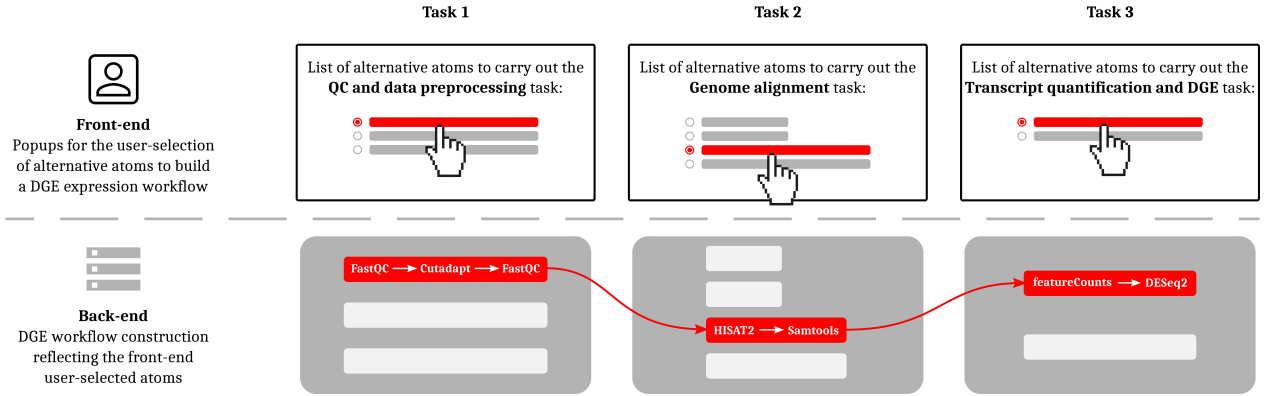


Tools

Histories

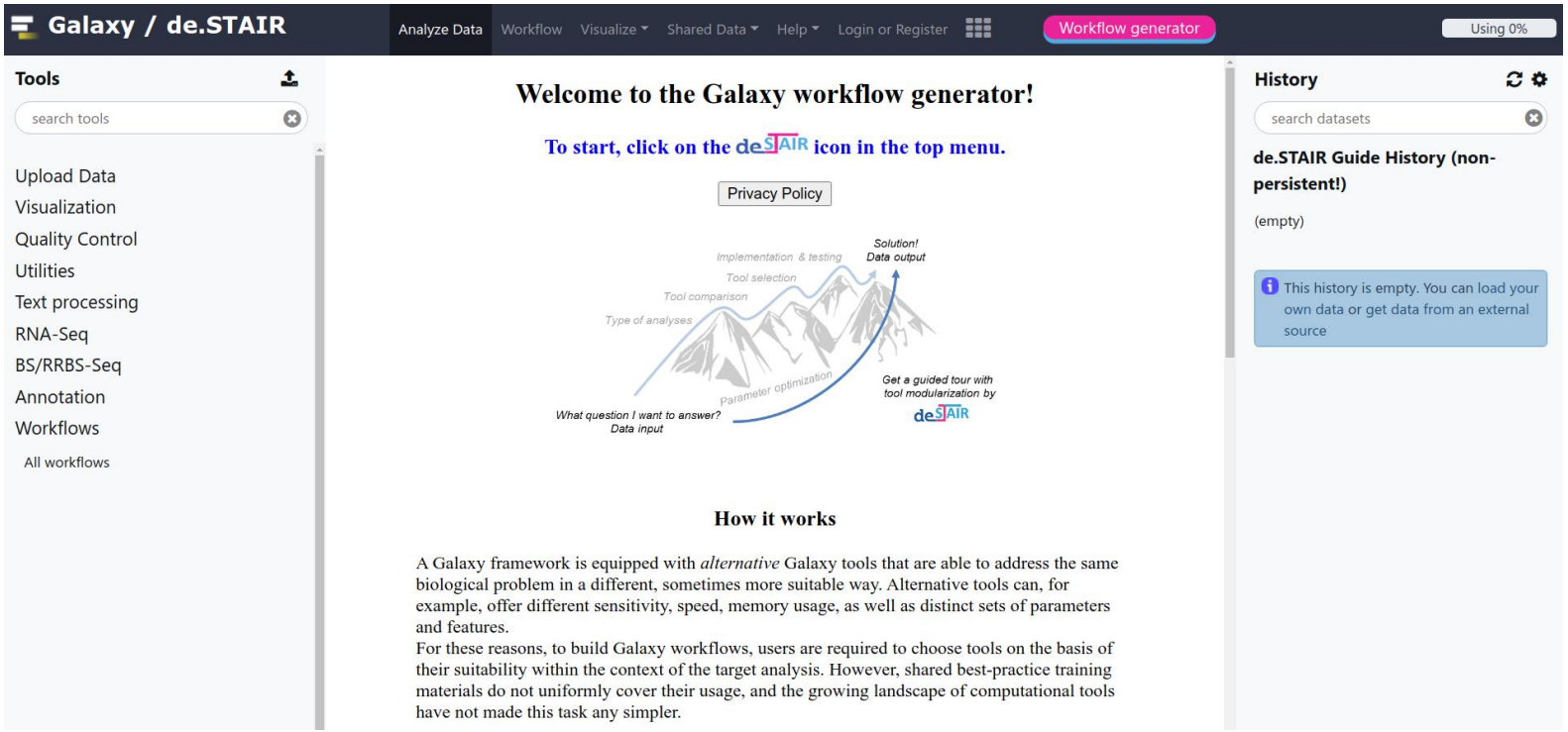
Galaxy Workflow Generator

- de.STAIR Atoms and Galaxy workflow generator plugin
  - improved stability and crafted atomic interactive tours
  - extended functionality to support various HTML elements
  - integrated a new interaction layer as Galaxy plugin to
    - select and execute alternative tools which solve the same data anlysis task (Atoms)
    - guide thorugh parameters
    - extract workflows, citations and commands
    - ensure functionalty without registration



Galaxy Workflow Generator

https://destair.leibniz-fli.de



Galaxy Workflow Generator

https://destair.leibniz-fli.de

But I simply want to know how to setup a Galaxy instance

Technical introduction

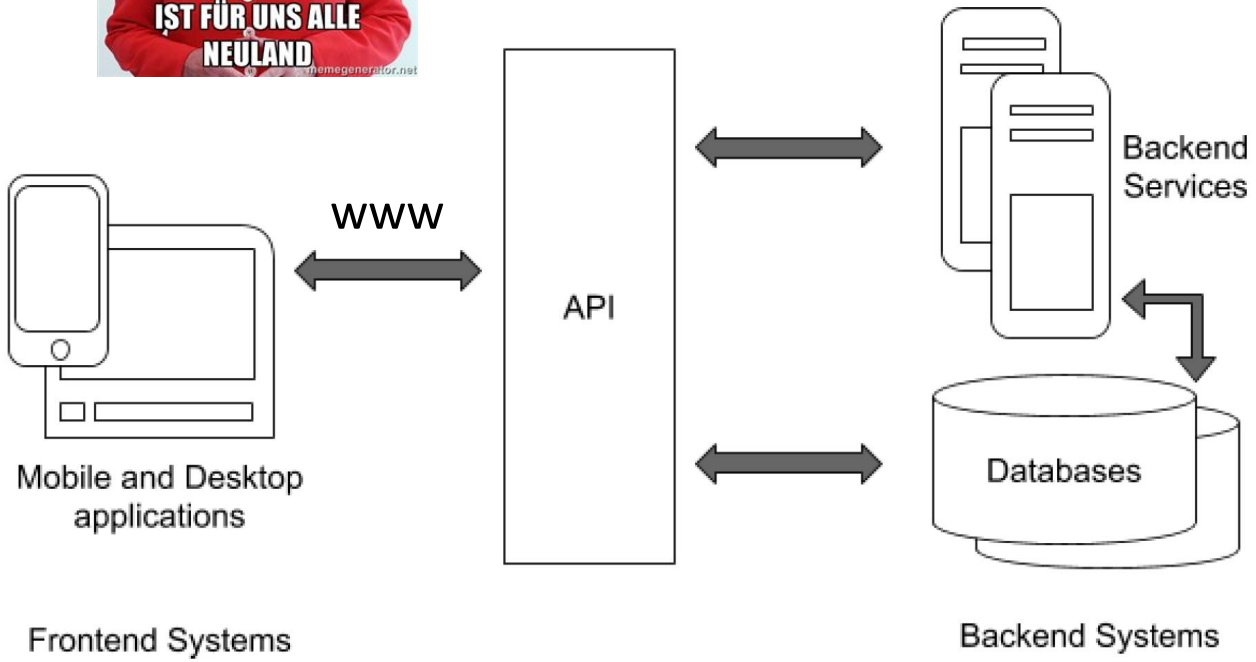
“The internet is unknown territory to all of us”

Merkel 2013, Press briefing topic: NSA monitoring

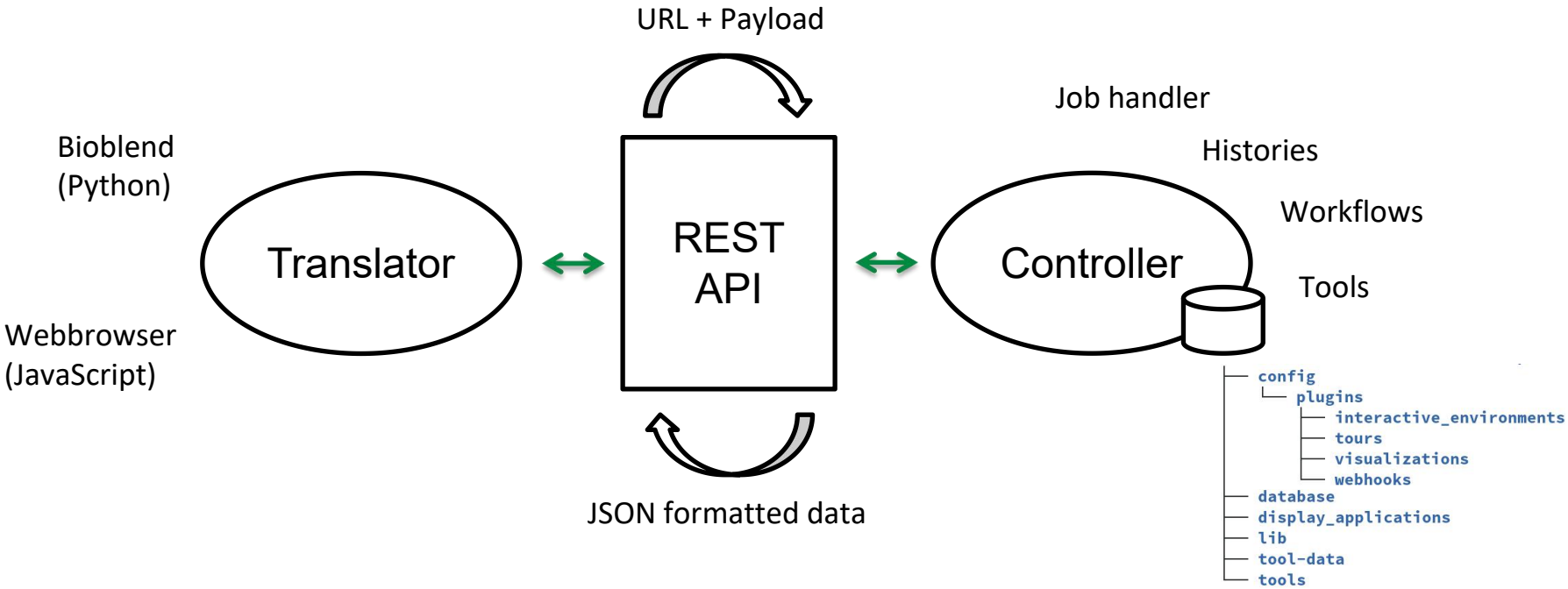
Technical introduction

"The internet is unknown territory to all of us"

Merkel 2013, Press briefing topic: NSA monitoring

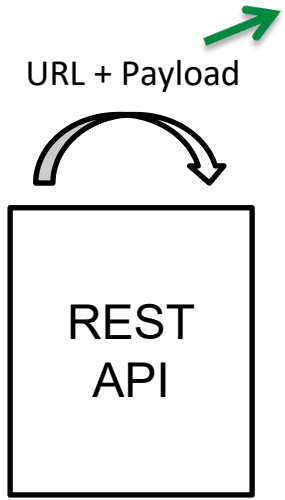


Technical introduction



Technical introduction

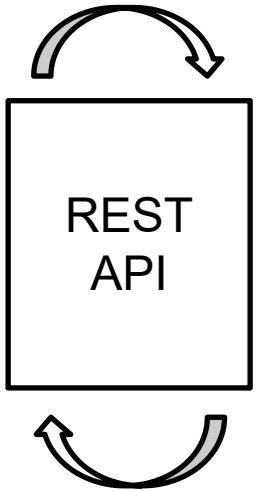
...
https://destair.leibniz-fli.de/api/tours
https://destair.leibniz-fli.de/api/histories
https://destair.leibniz-fli.de/api/workflows
https://destair.leibniz-fli.de/api/tools
...
https://destair.leibniz-fli.de/api/datatypes?extension_only=False



```
[
  {
    "elems": [
      {
        "panel_section_name": "Upload Data",
        "description": "convert Fastq data into unaligned BAM",
        "labels": [

        ],
        "edam_operations": [

        ],
        "form_style": "regular",
        "edam_topics": [

        ],
        "panel_section_id": "getext",
        "version": "2.18.2.1",
        "link": "/tool_runner?tool_id=toolshed.g2.bx.psu.edu%2Frepos%2Fdevteam%2Fpicard%2Fpicard_FastqToSam%2F2.18.2.1",
        "target": "galaxy_main",
        "min_width": -1,
        "model_class": "Tool",
        "id": "toolshed.g2.bx.psu.edu/repos/devteam/picard/picard_FastqToSam/2.18.2.1",
        "tool_shed_repository": {
          "owner": "devteam",
          "changeset_revision": "a1f0b3f4b781",
          "name": "picard",
          "tool_shed": "toolshed.g2.bx.psu.edu"
        },
        "name": "FastqToSam"
      },
      {
        "panel_section_name": "Upload Data",
        "description": "charts the nucleotide distribution per cycle in a SAM or BAM dataset",
        "labels": [
```


...
https://destair.leibniz-fli.de/api/tours
https://destair.leibniz-fli.de/api/histories
https://destair.leibniz-fli.de/api/workflows
https://destair.leibniz-fli.de/api/tools
...
https://destair.leibniz-fli.de/api/datatypes?extension_only=False

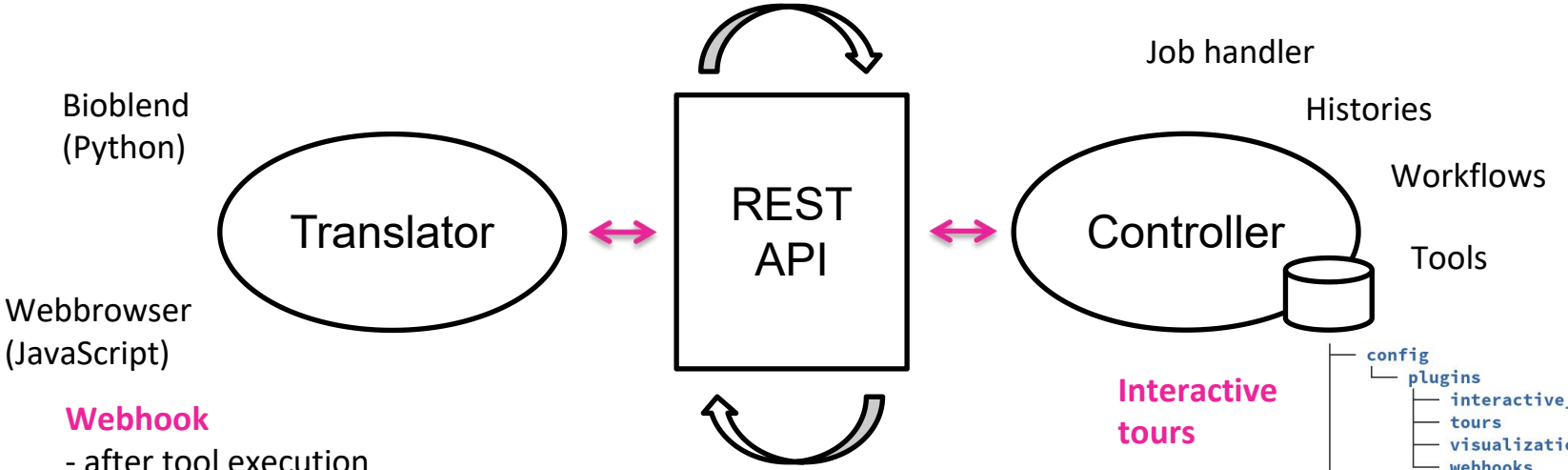
URL + Payload

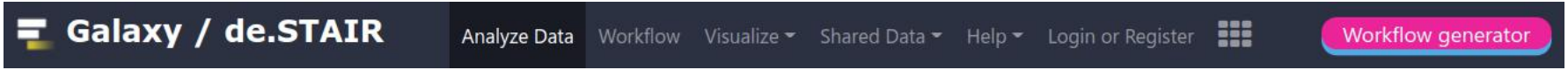


JSON formatted data

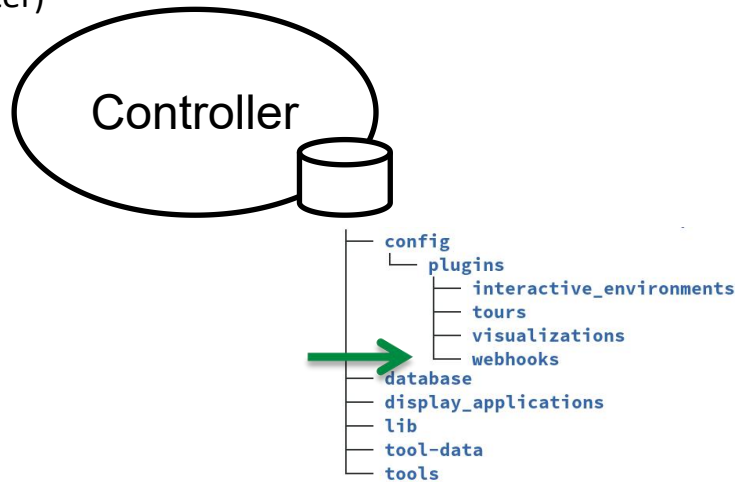
Technical introduction

URL to webhooks API + Payload to trigger new backend functions



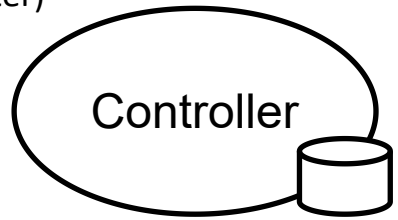


- Webhooks
  - JavaScript based frontend
    - triggers HTTP requests e.g. GET with payload (URL parameter) and receives JSON object string
  - Python based backend fuctions
    - Parse payload and uses its access to the controller objects to perform tasks and return results as JSON object string



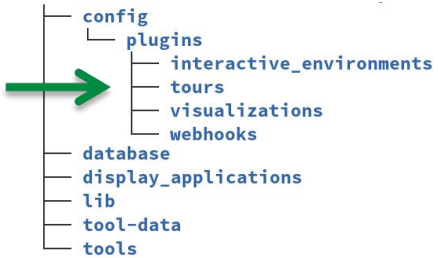
Technical introduction

- Webhooks
  - JavaScript based frontend
    - triggers HTTP requests e.g. GET with payload (URL parameter) and receives JSON object string
  - Python based backend fuctions
    - Parse payload and usses access to controller object to perform tasks and return results as JSON object string



- Tours
  - YAML files

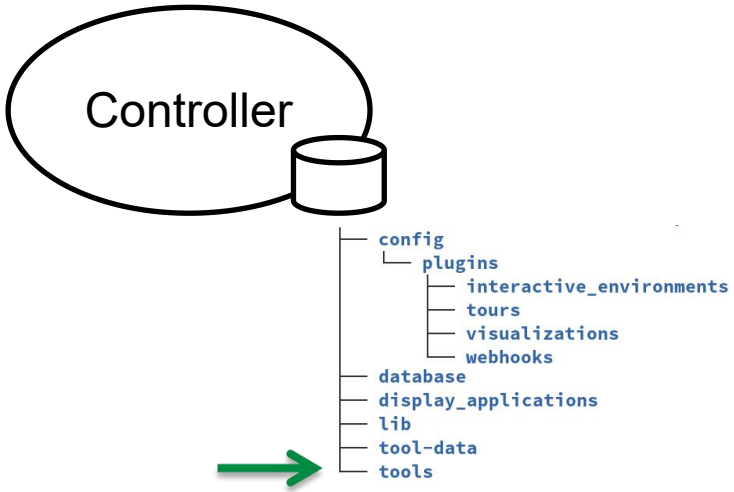
```yaml
id: bs_pe_1a
name: 'BS-Seq analysis (paired-end reads)'
description: 'Data upload'
title_default: 'BS-Seq analysis (paired-end reads)'
steps:
  - title: '<b>Outline</b>'
    element: '#masthead:visible'
    content: 'In this tour we will upload NGS data to be analysed for
methylation levels.<br>
Here, we will leverage on the <b>Data upload</b> tool provided by the Galaxy
framework.'
    placement: bottom

  - title: '<b>Import the sequence datasets</b>'
    element: '.fa.fa-upload:visible'
    content: 'Import data into Galaxy.'
    placement: right
    onnextclick:
      - '.fa.fa-upload:visible'
    onloadwait:
      - element: '.history-size:contains("empty")' # must not check for visibility
        count: 1
```



Technical introduction

- Tools
  - Binaries
  - XML wrapper to render command line options into graphical user interface

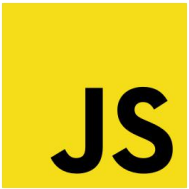
```
<tool id="uniqe_name" name="actual tool name" version="0.0.1">
  <requirements></requirements>
  <stdio><exit_code range="1:" /></stdio>
    <command><![CDATA[
        binary "$input_1" "$input_2" "$output"
    ]]></command>
  <inputs>
     <param type="data" name="input_1" format="csv" label="input table" />
     <param type="integer" name="input_2" value="0" label="input value" />
  </inputs>
  <outputs>
    <data name="output" format="csv" label="output table"/>
  </outputs>
  <tests></tests>
  <help>![CDATA[
    my binary does something
  ]]></help>
</tool>
```



But I simply want to know how to setup a Galaxy instance

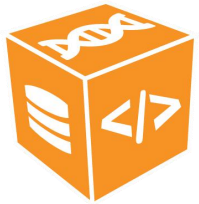
Dockerized Galaxy

HTTP/FTP Server
Job handler
Scheduler plugins



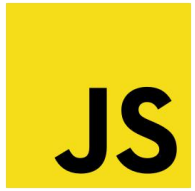
...scratching the surface

Dockerized Galaxy



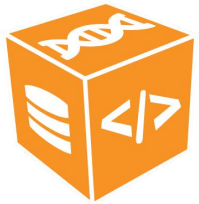
BioContainers

HTTP/FTP Server
Job handler
Scheduler plugins



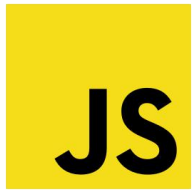
...scratching the surface

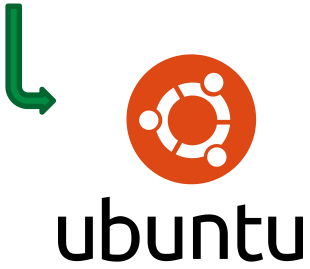
Dockerized Galaxy

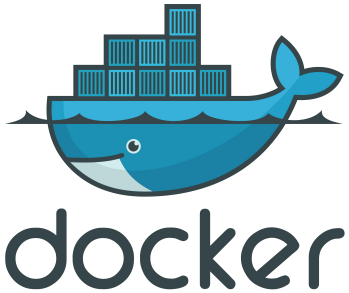


BioContainers

HTTP/FTP Server
Job handler
Scheduler plugins







Dockerized Galaxy

- Fused and stabilized via Docker
  - Dendencies and libraries
  - Necessary environment setting
  - Databases
  - Webservers
  - Software
  - Build systems and job handler
- A universe of techniques to avoid pitfalls in manual pipeline design

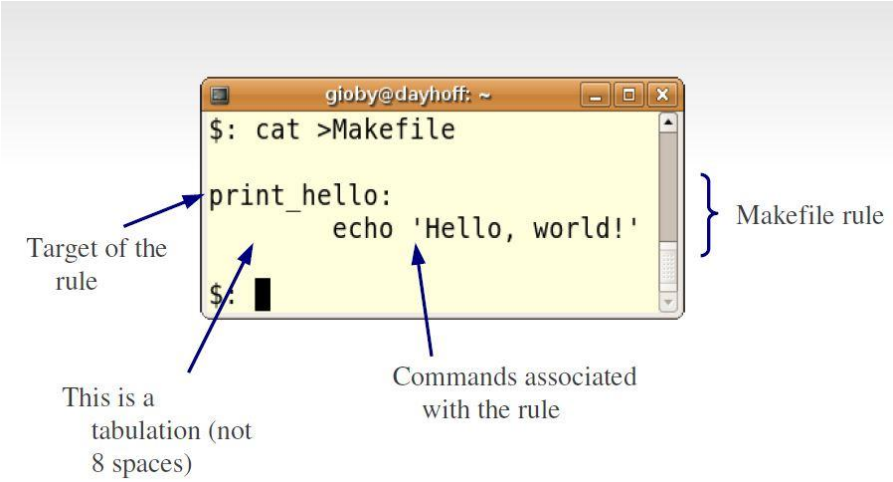
Dockerized Galaxy

- Fused and stabilized via Docker
  - Dendencies and libraries
  - Necessary environment setting
  - Databases
  - Webservers
  - Software
  - Build systems and job handler
- A universe of techniques to avoid pitfalls in manual pipeline design

..lets unravel this!..

Back in the days..

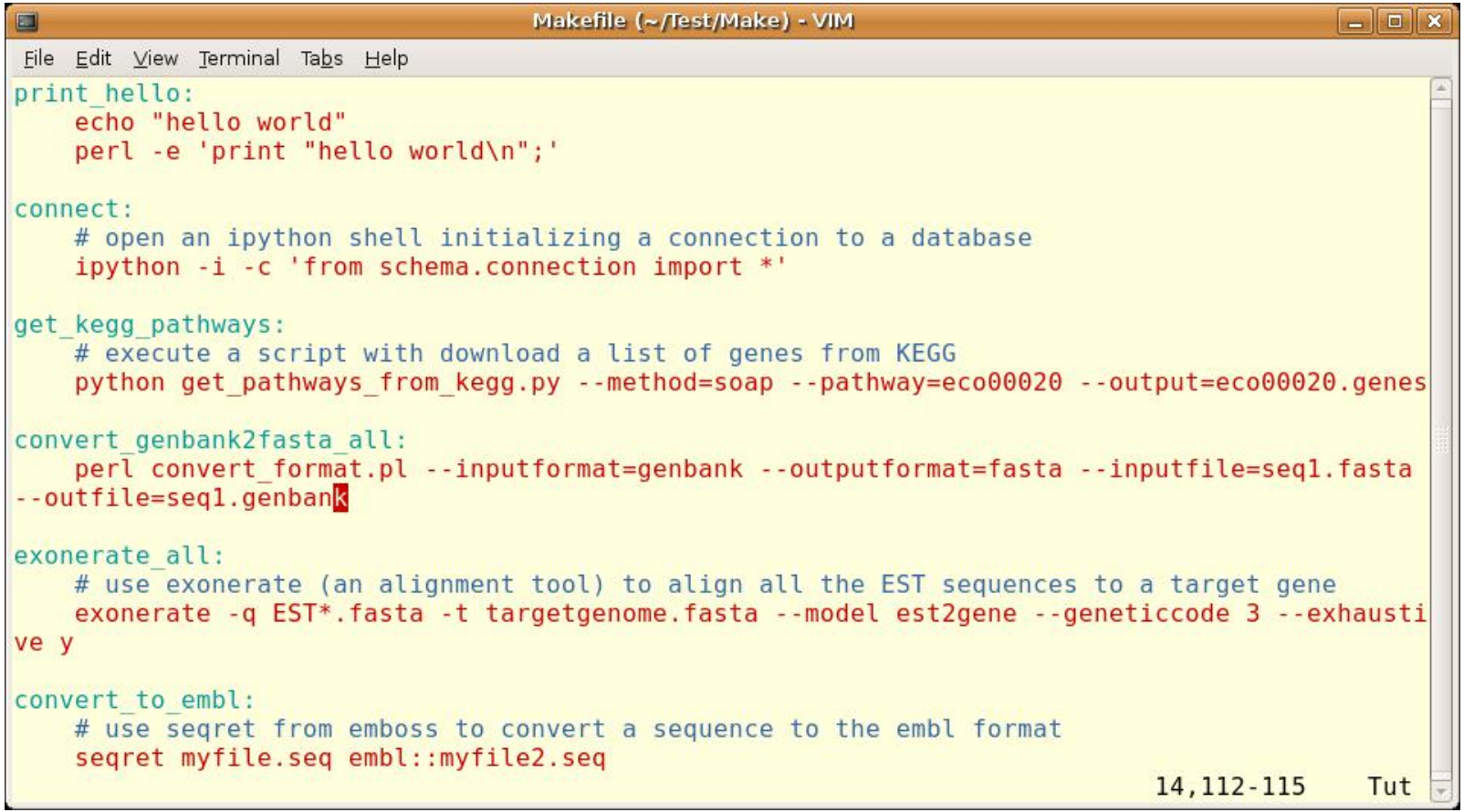
- The make build system
  - make is a tool used by programmers to define how software should be compiled and installed
  - It is also used as a way to store a set of commands to recall them later
  - Makefile is composed of rules to fulfill certain tasks and stores the associated shell commands



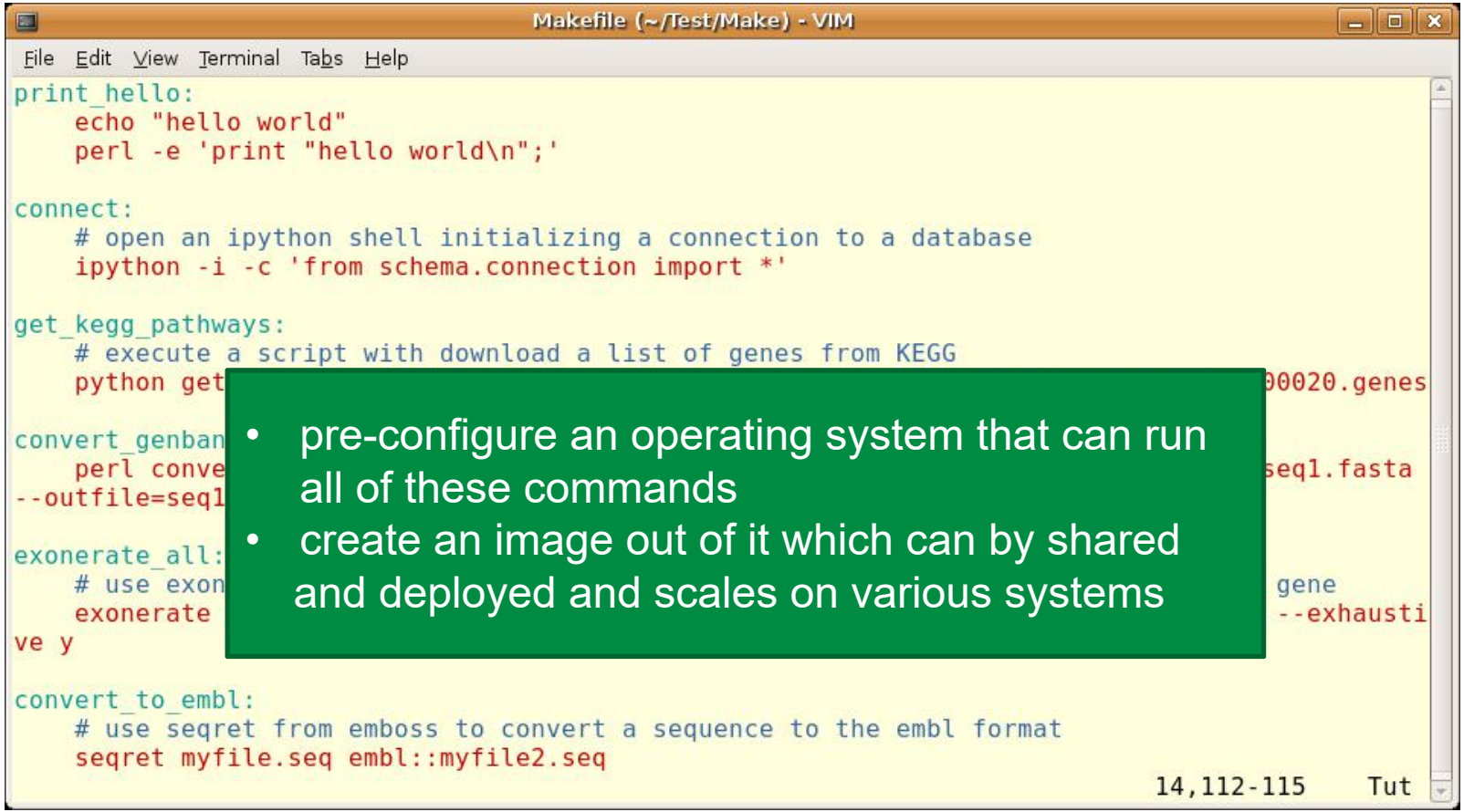
Back in the days..

- The make build system
  - invoke with the command “make”
  - make looks for a file in the current directory called “Makefile”
  - When we type “make print_hello”, it executes the function (target) called “print_hello” in the Makefile

Back in the days..



Back in the days..

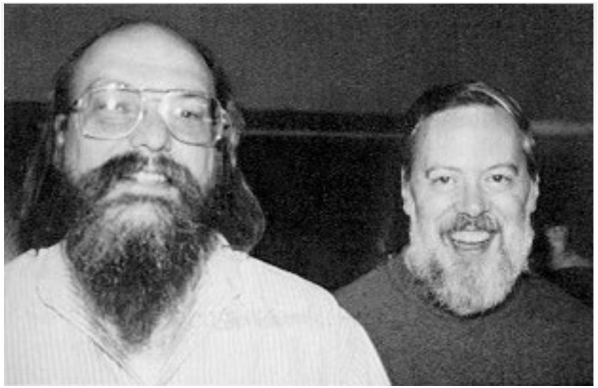


Back in the days..

Back in the days..

- What is an operating system?
  - The set of all the programs needed to use a machine
  - A software collection to coordinate all hardware components (kernel)
  - A software to draw windows (graphical interface)
  - Software to edit texts, browse the internet, install other software, etc..

- Way ahead of Windows: Unix
  - Officialy released 1969
  - ~30 programs/commands (ls, cp, mv,...)
  - Invented shell command structure



Ken Thompson and Dennis Ritchie

Back in the days..

- 1991 GNU Linux implemented Unix kernel concepts as an open source operating system
  - Mac OS
  - Android
  - IoT devices
- Unix was rapidly adopted worldwide, and became the standard operating system, especially in universities due to free licenses
- Thanks to the adoption of GNU/Linux servers, internet grew considerably in 1991 - 1992
- The command interpreter “Unix shell” or “Bourne Again Shell” (Bash) became defacto standard

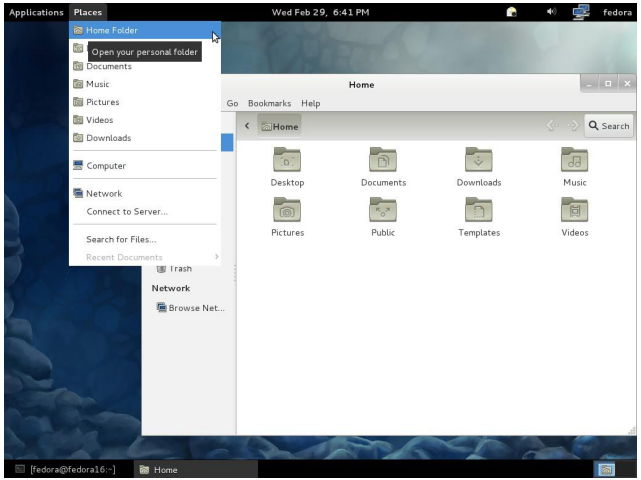
Bash

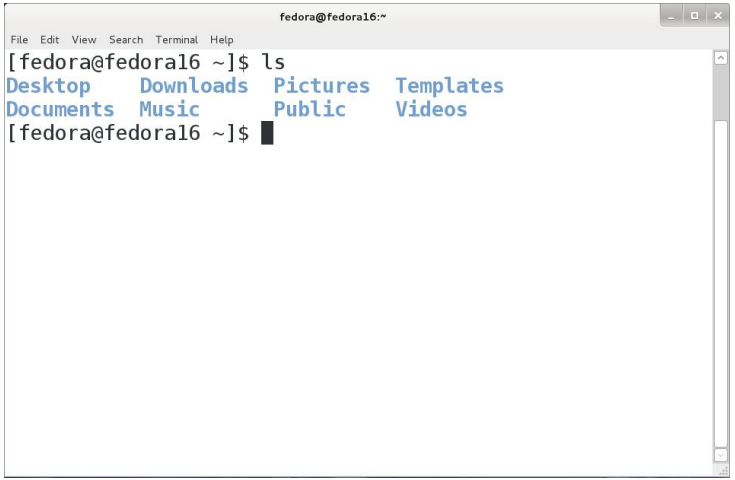

Google
10:14 PM
oliversmith
Anwendungen Orte System
Di, 27. Jun., 10:32
Rechner
Persönlicher Ordner
Papierkorb
Applications Places System
NEBC Homepage
bio-linux 6
User Guide
Introductory Tutorial
Bioinformatics Docs
Sample Data

Bash

"Write programs that do one thing and do it well.
Write programs to work together. Write programs to handle text streams, because that is a universal interface."

(Doug Mcllroy, Unix pipes inventor)





- Command structure
  - tool name: ls, cp, rm, mv, grep, cat, head, tail, sort, paste, join, ...
  - options may or may not start with a minus or double minus (long version)

```
toolname option [argument] [option [argument] ..]
```

  - textual representation enables to chain commands with the “pipe” symbol

```
ls --all | grep -v *.jpeg
```

Bash

- Resource for options
  - toolname -h
  - toolname --help
  - man toolname

```
LS(1)                          User Commands                          LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor --sort.

       Mandatory arguments to long options are  mandatory  for  short  options
       too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..

       --author
              with -l, print the author of each file

       -b, --escape
              print C-style escapes for nongraphic characters

       --block-size=SIZE
              use SIZE-byte blocks.  See SIZE format below

       -B, --ignore-backups
              do not list implied entries ending with ~

       -c     with -lt: sort by, and show, ctime (time of last modification of
              file status information) with -l: show ctime and  sort  by  name
              otherwise: sort by ctime

       -C     list entries by columns

       --color[=WHEN]
              colorize  the  output.   WHEN  defaults  to  `always'  or can be
              `never' or `auto'.  More info below
Manual page ls(1) line 1
```

Conda

- ls, cp, grep, cat, ... are programs shipped with the Linux distribution
- Third party software needs to be installed separately via
  - Linux Package Manager (administrator) - “App-Store”
  - Custom Package Manager (user)
  - Compilation from source (user, needs compiler and dependencies)

    → may by tricky and error prone

Conda

- Conda Is a free, community developed package manager
- Easily install software and dependencies as a user
- Software is retrieved from the online repository “Anaconda cloud”
- ~7500 Bioinformatics tools/libraries can be found in the bioconda channel

https://conda.io

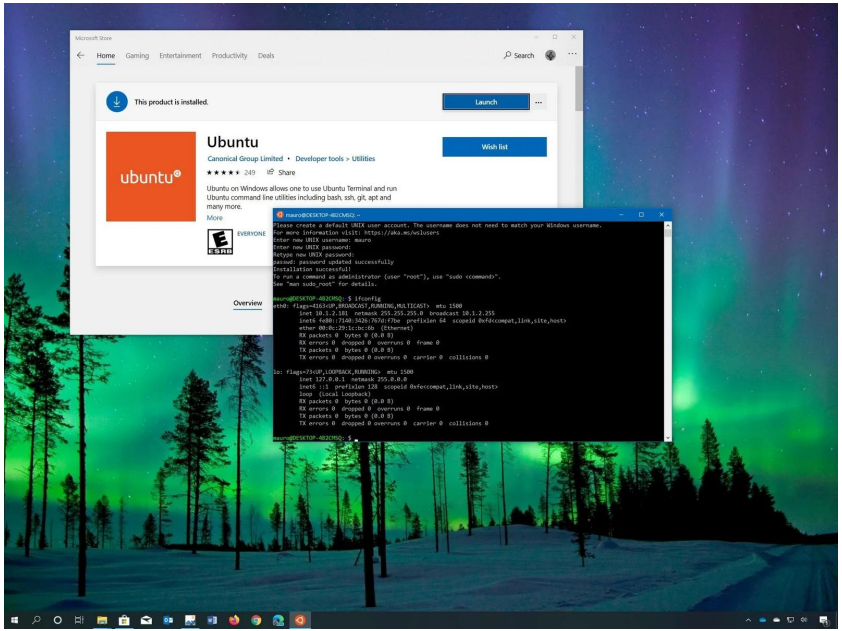
```
bash /path/to/miniconda_setup.sh -b -p /path/to/conda
source /path/to/conda/bin/activate
conda install -c bioconda fastqc samtools trimmomatic bwa

samtools faidx /path/to/genome.fasta
fastqc /path/to/rna-seq.fastq
...
```

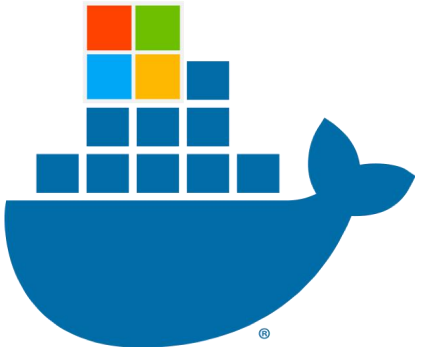
Docker

- Preconfigure the environment
  - As full grown image of an operating system for a virtual machine
  - Orchestrate container as sandbox environments
    - Azure container service
    - Amazon container servervice
    - Portainer
    - OpenShift
    - Kubernetes
    - **Docker**

Docker

- Ubuntu on Windows 10 Subsystem for Linux (WSL) released 2016
- This significant architectural change as it is a full Linux kernel built by Microsoft, allowing Linux containers to run natively

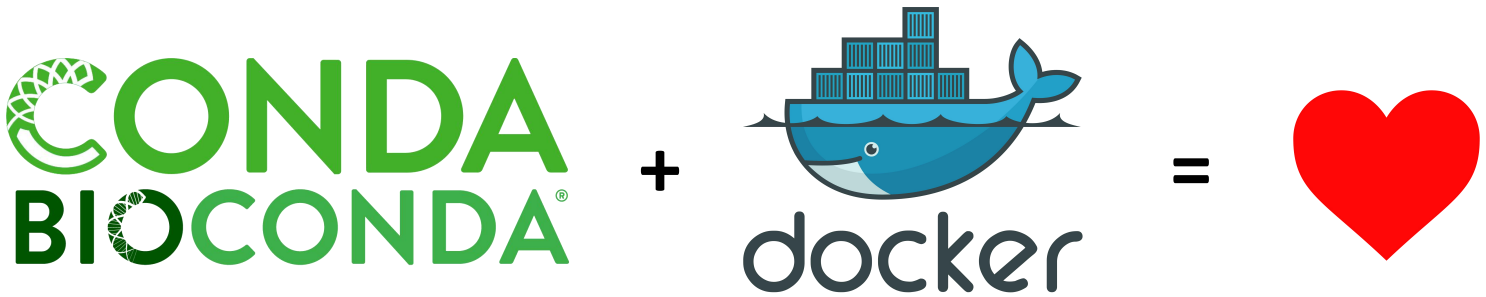




Docker

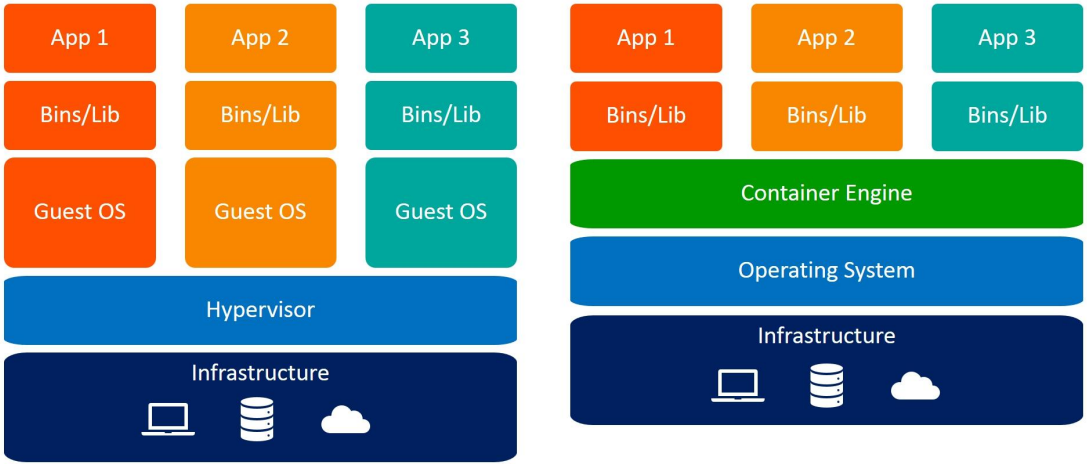
- Docker is a virtualization technique to setup uniform environments in which necessary software can be installed
- The whole environment can be shipped as an “image”
- Docker images can be executes as “containers” on any operating system (Mac, Win, Linux, Unix)

  → Include own applications and run them everywhere



Docker

- Advantages of a container over a virtal machine
  - simple setup of a sandbox environment
  - better performance and lightweight due to shared kernel
  - bind mounts
  - no complex configuration of a hypervisor



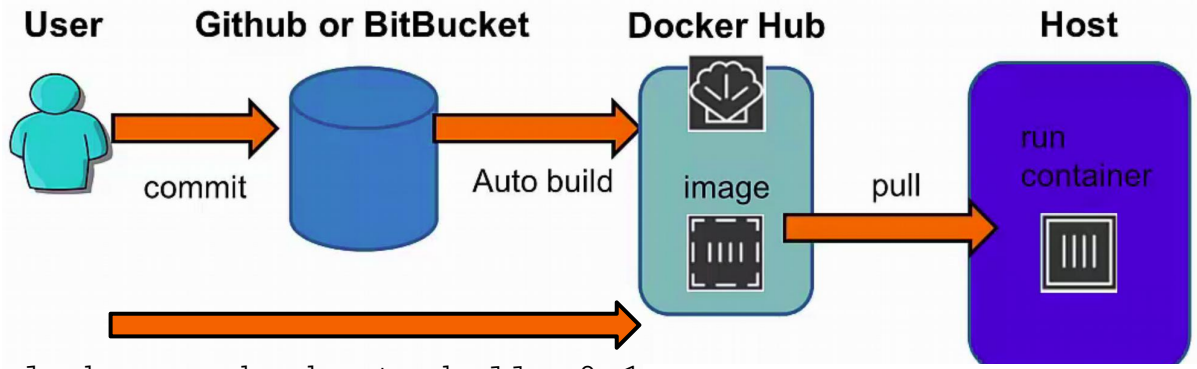
Virtual Machines Containers

Docker

```
# script.sh
echo “hello world” > /bind/mount/directory/hello.txt

# Dockerfile
FROM ubuntu:latest
ENV PATH=$PATH:/path/to/scripts
COPY script.sh /path/to/scripts
```

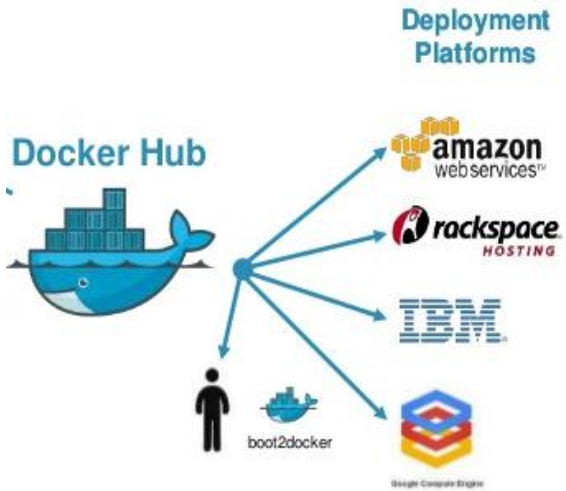
```
docker build -t ubuntu_hello:0.1 /path/to/Dockerfile
docker run --rm -d -v /local/directory:/bind/mount/directory \
  ubuntu_hello:0.1 bash script.sh
cat /local/directory/hello.txt
```

Docker




```
docker push ubuntu_hello:0.1
```

https://hub.docker.com



Docker - Cheat sheet

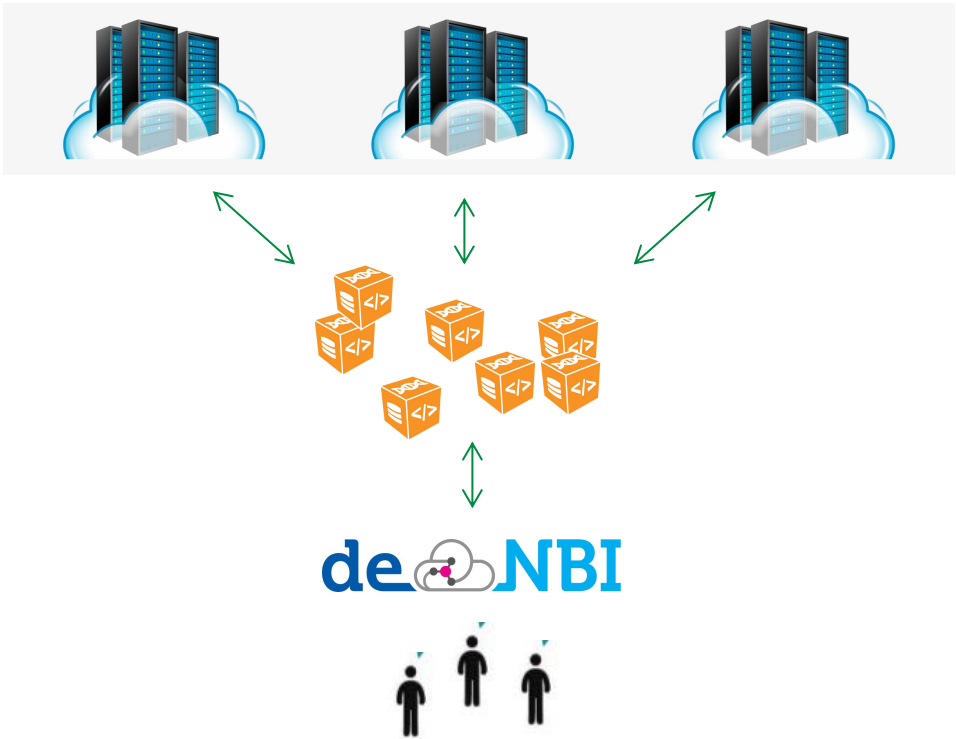
```
docker pull ubuntu:latest                              # pull remote image
docker images                                          # list local images

docker run --rm ubuntu:latest ls -l                    # run container, execute a command and
clean up after exit

docker run -i -t --name ubuntu ubuntu:latest bash      # run and enter an interactive bash
                                                       # detach with ctr + p , ctr + q

docker run -d -i --name ubuntu ubuntu:latest bash      # run interactive container detached
docker ps -a                                           # list running and stoped containers

docker exec ubuntu ls -l                               # execute command in running container
docker exec -i -t ubuntu bash                          # open interactive bash to container

docker stop ubuntu                                     # stop a running container
docker commit ubuntu ubuntu_modified:0.1               # commit as new image to run later
docker run --rm ubuntu_modified:0.1 ls -l

docker start ubuntu                                    # resume a stopped container
docker exec -i -t ubuntu bash                          # re-open interactive bash to container

docker rm -f ubuntu                                    # nuke container
docker rmi -f ubuntu_modified:0.1                      # nuke image
```

Docker

- Docker can be integrated into a pipeline
- BioContainers is a resource for dockerized bioinformatics tools

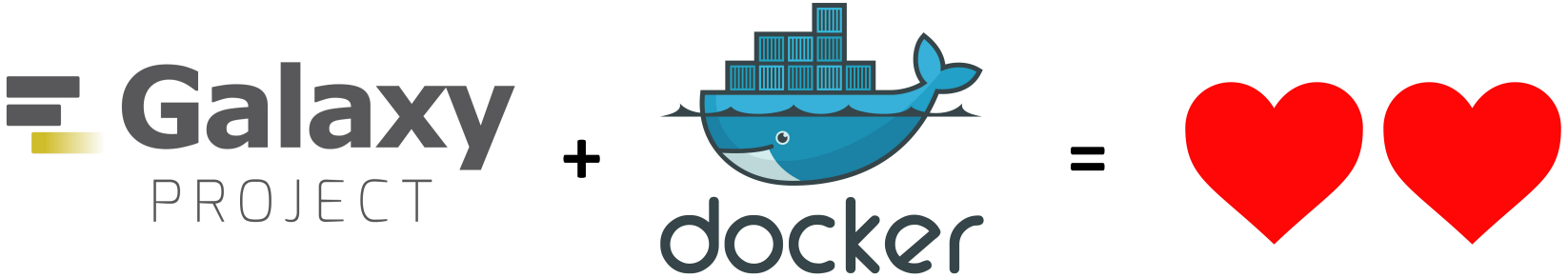
```
docker run --rm quay.io/biocontainers/samtools:1.12 samtools

cat file.sam | docker run --rm -i quay.io/biocontainers/samtools:1.12
samtools view -f 4 /mnt/file.sam | wc -l > number_unmapped.txt

docker run --rm -v /path/to/file.sam:/mnt/file.sam \
quay.io/biocontainers/samtools:1.12 \
samtools view -f 4 /mnt/file.sam | wc -l > number_unmapped.txt

docker run --rm -v /path/to/sam-directory:/mnt \
quay.io/biocontainers/samtools:1.12 \
bash -c 'samtools view -f 4 /mnt/file.sam | wc -l > /mnt/number_unmapped.txt'
```

https://cloud.denbi.de



Thanks, but I want to know how to setup a Galaxy instance

Galaxy Flavors



- Galaxy instance on Ubuntu base image is launching
  - BioContainers or tools from its toolshed mainly setupped via Conda
  - FTP-Server
  - Job and process controller and scheduler
  - Webserver
  - Database
  - User interface
  - Interface for interactive environments

Galaxy Flavors

- URL ⟷ DNS ⟷ ip
- localhost: 127.0.0.1
- http: port 80
- https: port 443

Type	Domain Name	IP Address	TTL
A	whatismyipaddress.com	104.16.154.36 Cloudflare, Inc. (AS13335)	5 min
A	whatismyipaddress.com	104.16.155.36 Cloudflare, Inc. (AS13335)	5 min

Galaxy Flavors

- URL ⟷ DNS ⟷ ip
- localhost: 127.0.0.1
- http: port 80
- https: port 443

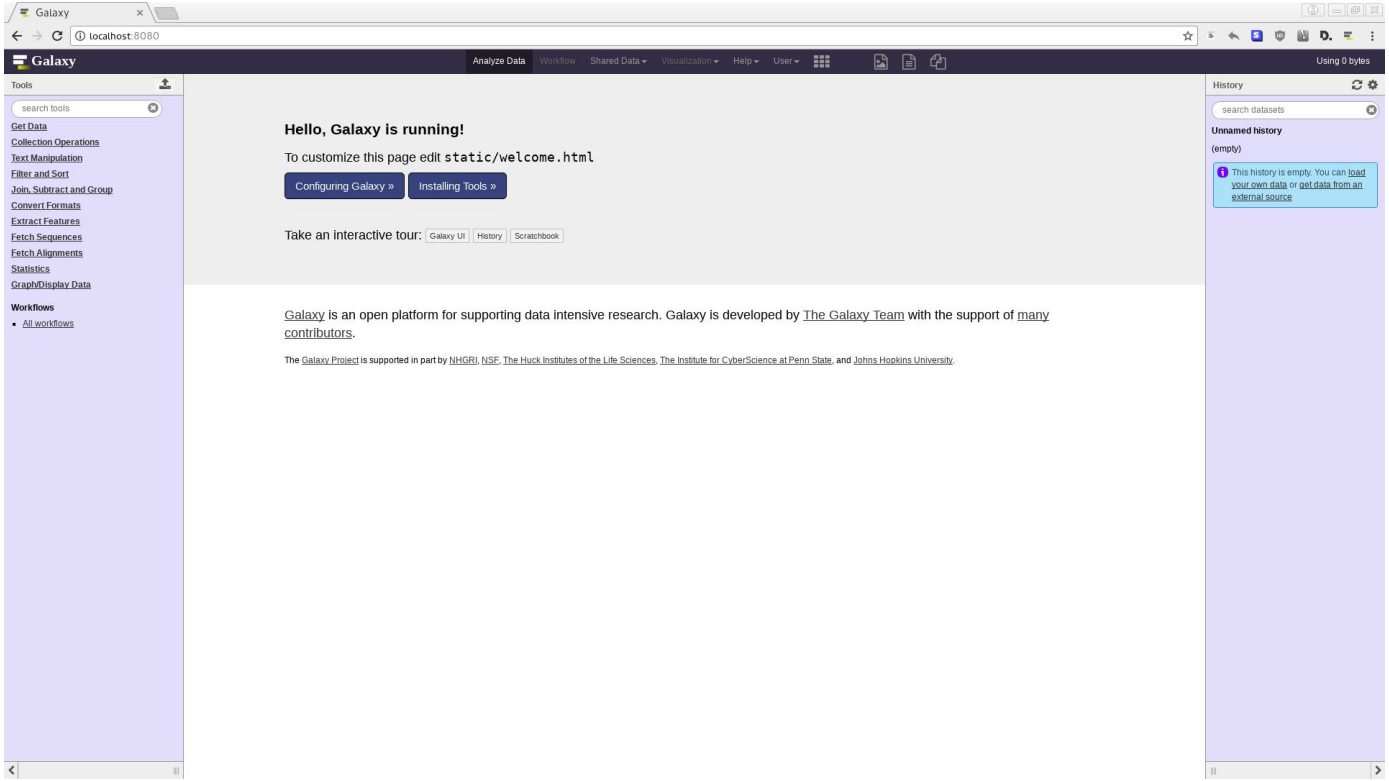
Type	Domain Name	IP Address	TTL
A	whatismyipaddress.com	104.16.154.36 Cloudflare, Inc. (AS13335)	5 min
A	whatismyipaddress.com	104.16.155.36 Cloudflare, Inc. (AS13335)	5 min

run docker container and remap its http server to be accessible via port 8080

```
docker run -d -p 8080:80 \
quay.io/bgruening/galaxy:19.05
```

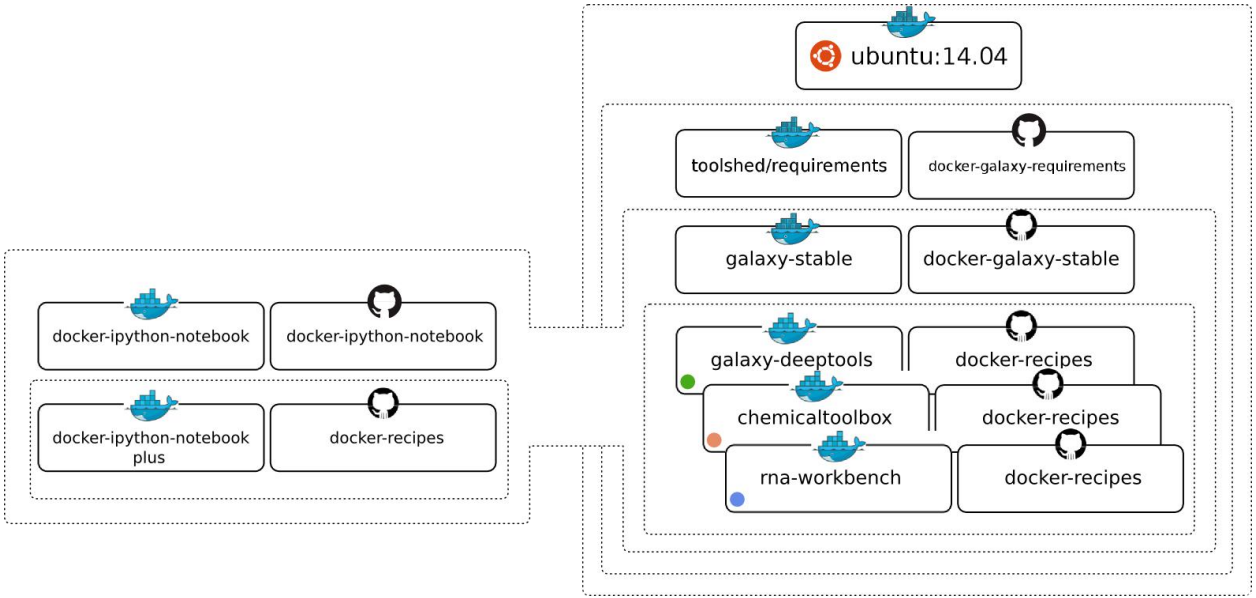
Galaxy Flavors

http://localhost:8080



Galaxy Flavors

- ..not so empty galaxy flavors



ASaiM (Metagenomics)
Galaxy Epigenetics
HiCExplorer
...

Galaxy Flavors

- Its mainly about config files
  - Scheduler configuration
  - List of toolshed tools
  - Galaxy configuration
  - Docker commands

Dockerfile

Tools YAML

Galaxy YAML

Scheduler XML

Where can I find a blueprint with simple, but working config templates

GitHub

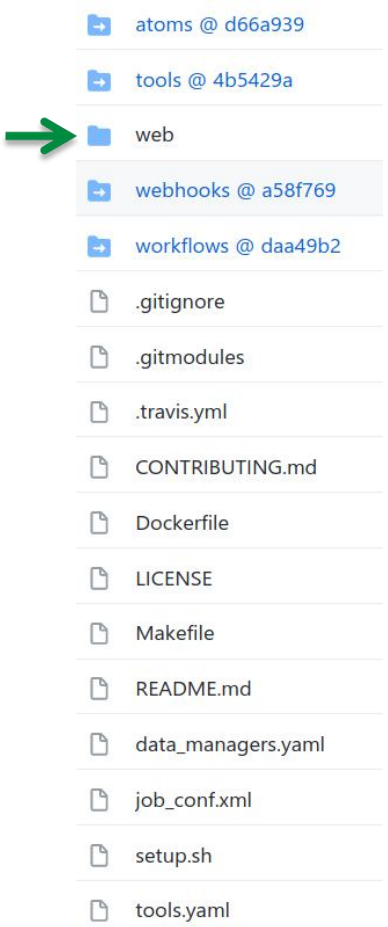
destairdenbi / galaxy-workflow-generator

forked from bgruening/docker-galaxy-ngs-preprocessing

```
git clone --recursive \
https://github.com/destairdenbi/galaxy-workflow-generator
```

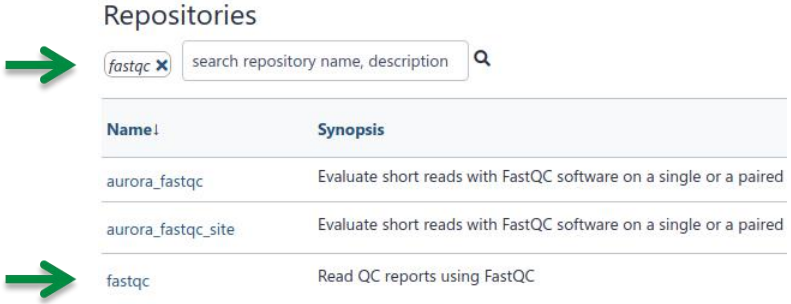
de.STAIR Flavor - Landing page / welcome.html

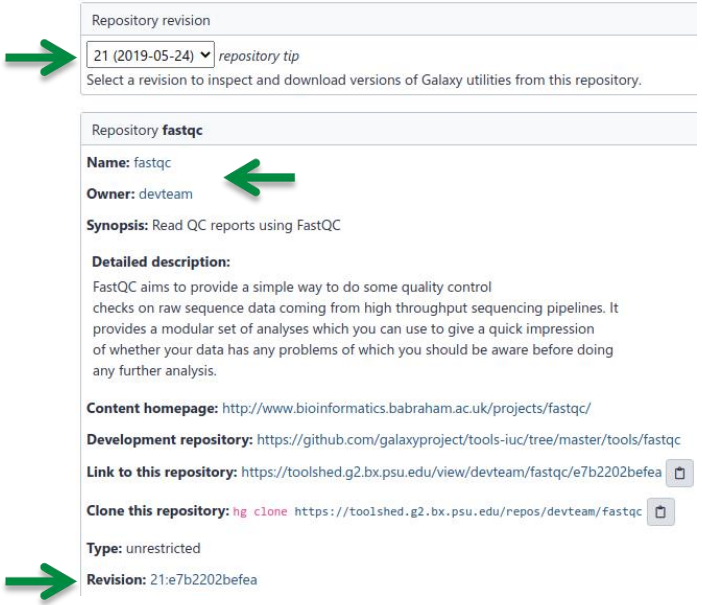
```
<!DOCTYPE html>
<html lang="en">
  <div align="center">
    <img src="./pic.png" alt="pic" width="50%"/>
    <h2>Welcome!</h2>
    <p>
      This is my Galaxy instance.
    </p>
  </div>
</html>
```



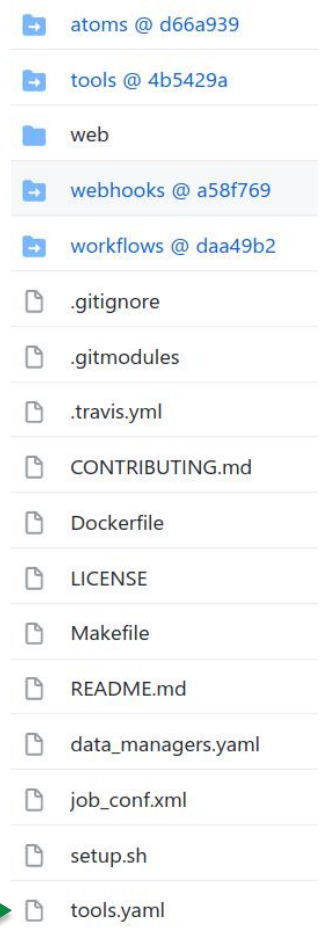
de.STAIR Flavor - Tools

https://toolshed.g2.bx.psu.edu






```
11  - name: fastqc
12    owner: devteam
13    tool_panel_section_label: "Quality Control"
14    revisions:
15    - 'e7b2202befea'
```



de.STAIR Flavor - Scheduler

https://docs.galaxyproject.org/en/latest/admin/jobs.html

- Multithreaded tools should be assigned to “slurmthreaded”

```
32        <destination id="slurm" runner="slurm">
33            <!-- <env file="/galaxy_venv/bin/activate"/> -->
34            <param id="nativeSpecification">--ntasks=1</param>
35        </destination>
36        <destination id="slurmthreaded" runner="slurm">
37            <!-- <env file="/galaxy_venv/bin/activate"/> -->
38            <param id="nativeSpecification" from_environ="GALAXY_CONFIG_PARALLEL_SLURM_PARAMS">--ntasks=2</param>
39        </destination>

42    <tools>
43        <tool id="cutadapt" destination="slurmthreaded"/>
44        <tool id="trim_galore" destination="slurmthreaded"/>
45        <tool id="trimmomatic" destination="slurmthreaded"/>
```

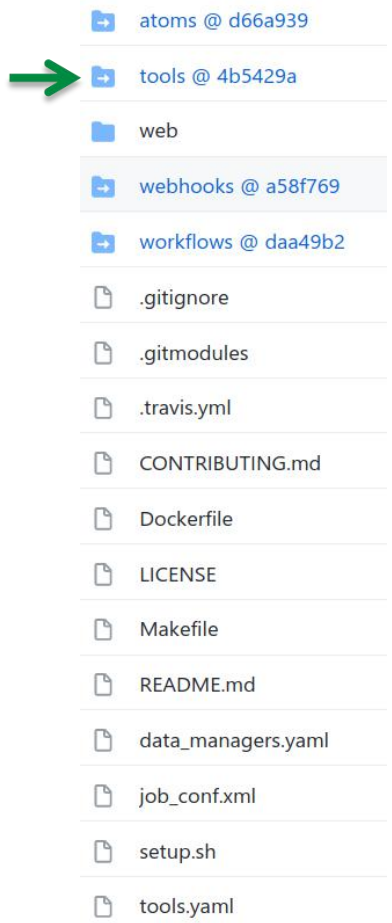
```
<limits>
    <limit type="walltime">999999:00:00</limit>
    <limit type="destination_user_concurrent_jobs" id="slurm">999999</limit>
    <limit type="destination_user_concurrent_jobs" id="slurmthreaded">999999</limit>
    <limit type="destination_user_concurrent_jobs" id="local">999999</limit>
    <limit type="destination_user_concurrent_jobs" id="localthreaded">999999</limit>
</limits>
```

- atoms @ d66a939
- tools @ 4b5429a
- web
- webhooks @ a58f769
- workflows @ daa49b2
- .gitignore
- .gitmodules
- .travis.yml
- CONTRIBUTING.md
- Dockerfile
- LICENSE
- Makefile
- README.md
- data_managers.yaml
- job_conf.xml
- setup.sh
- tools.yaml

de.STAIR Flavor - Custom tools

- create a sub-directory with unique name
  - place in the binary
  - create a .xml file named like the directory itself
  - for data types and more see https://planemo.readthedocs.io

```
<tool id="uniqe_name" name="actual tool name" version="0.0.1">
  <requirements></requirements>
  <stdio><exit_code range="1:" /></stdio>
    <command><![CDATA[
        binary "$input_1" "$input_2" "$output"
    ]]></command>
  <inputs>
     <param type="data" name="input_1" format="csv" label="input table" />
     <param type="integer" name="input_2" value="0" label="input value" />
  </inputs>
  <outputs>
    <data name="output" format="csv" label="output table"/>
  </outputs>
  <tests></tests>
  <help>![CDATA[
    my binary does something
  ]]></help>
</tool>
```



de.STAIR Flavor

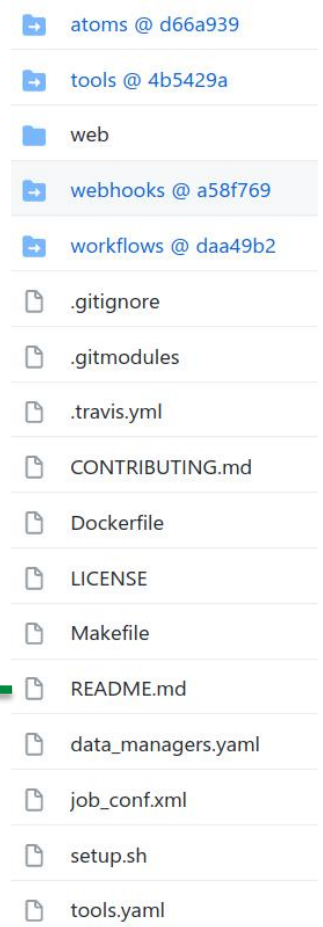
```
docker build -t my_galaxy:0.1 /path/to/Dockerfile
docker run -d -p 8080:80 --name galaxy my_galaxy:0.1
```

enable multithreading

```
[...] -e "GALAXY_CONFIG_PARALLEL_SLURM_PARAMS=--ntasks=8"
  -e "GALAXY_CONFIG_PARALLEL_LOCAL_NTASKS=8" [...]
```

bind mount location to persistent database

```
[...] -v /path/to/local/directory:/export [...]
```



de.STAIR Flavor - Galaxy config file

```
docker exec galaxy bash -c 'echo $GALAXY_CONFIG_FILE'
docker cp galaxy:/etc/galaxy/galaxy.yml /path/to/galaxy.yml
```

edit galaxy.yml - e.g. set require_login to false

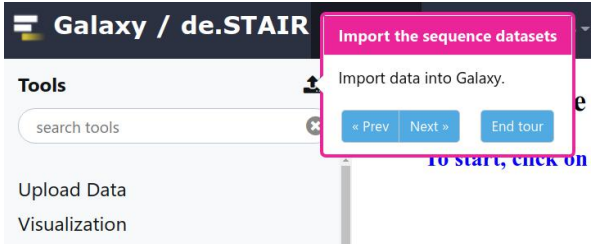
```
docker cp /path/to/galaxy.yml galaxy:/etc/galaxy/galaxy.yml
docker exec galaxy supervisorctl restart galaxy:
```

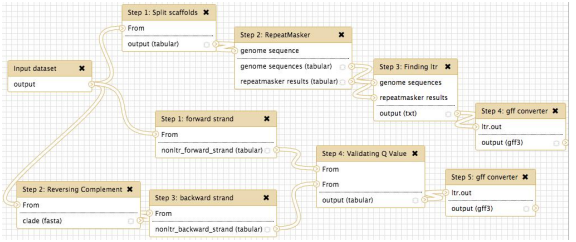
refresh browser

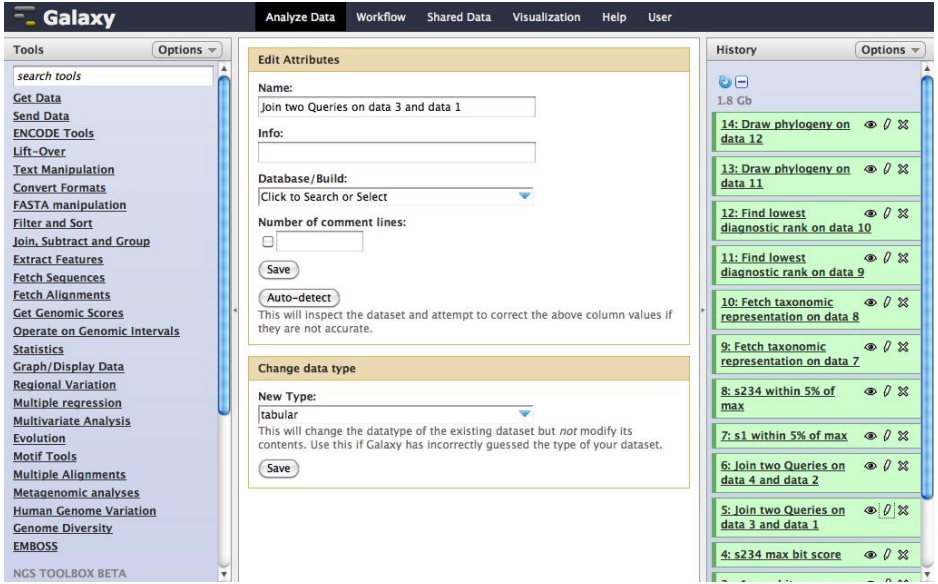
sometimes it is necessary to delete cookies and cache

de.STAIR Flavor - Have fun!

deSTAIR







Questions?

Leibniz Institute on Aging – Fritz Lipmann Institute

Universität Rostock Traditio et Innovatio

GERMAN NETWORK FOR BIOINFORMATICS INFRASTRUCTURE

Bundesministerium für Bildung und Forschung

We thank the Freiburg Galaxy Team and all Galaxy community trainers and developers for their great work.
We also thank the BMBF for the de.NBI-Partner program (Grant 031L0106) and the European Social Fund (ESF/14-BM-A55-0027/18).