

ASM-NGS Hackathon 2020

Unit testing
9am to 12pm daily
Dec 2 through Dec 4

Introduction



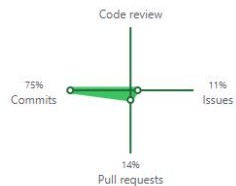
101 contributions in the last year



@microbinfie-hackatho... @bioconda

Activity overview

Contributed to [boasvdp/boasvdp.github.io](#), [microbinfie-hackathon2020/CSIS](#), [ramadatta/CPgeneProfiler](#) and 5 other repositories



448 contributions in the last year

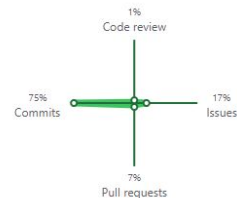
Contribution settings



@microbinfie-hackatho... @CFSAN-Biostatistics @ncezid-biome More

Activity overview

Contributed to [Iskatz/SneakerNet](#), [microbinfie-hackathon2020/CSIS](#), [Iskatz/BookAI](#) and 5 other repositories



So let's say you are coding a project `biotool`

Does my software work on my computer?

- Get `fastq.gz` file
 - run `biotool`
 - It runs
 - Go home happy; have a drink after hours
- Test for more things?
 - Exit code 0
 - Does it make an output file?
 - Is the output file correct?



More on testing biotool

Your colleague says it doesn't work on his or her computer

- Send them the input file
- Ask what they see after they run it
- Solutions if it works here but not there (yet)
 - "Just edit the file and add xyz to the script"
 - "Move the input file to this subfolder you were supposed to make"
 - "You needed to install xyz dependencies"



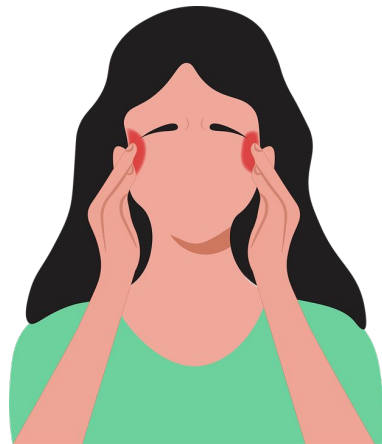
I need to change something on `biotool`

Remember all the tests I made before

Run those tests

Remember how to check the outputs

Oh yeah and does it work on my friend's computer?



The answer: unit tests!

- Create a set of basic tests
 - Not a crazy idea
 - Just a few lines of code
- Examples
 - Exit code 0
 - Check basic output
 - Does --version work?
 - Does --help work?
- Run it here, run it there
 - On your computer
 - On GitHub Actions (on GitHub's remote computer)
 - On your friend's computer (?)



Micro Binfie Podcast



<https://soundcloud.com/microbinfie>



Listen on Apple
iTunes

LISTEN ON



Spotify

RSS

Your Hosts



Lee Katz (CDC)
Andrew Page (QIB)
Nabil Alikhan (QIB)

Further discussion on our podcast

- Episode 25
- Sustainable bioinformatics software
- <https://soundcloud.com/microbinfie/25-sustainable-bioinformatics-software>

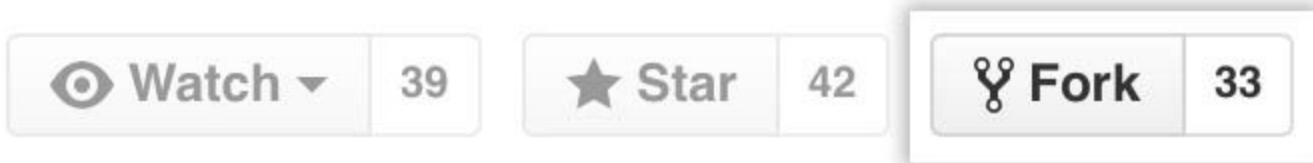
Housekeeping 🧹

- Zoom
 - <https://cdc.zoomgov.com/j/1619445515?pwd=OTZRWmhpdHhVVTV4VFVnV2JvVEd2QT09>
 - ID: 161 944 5515
 - passcode: xwK02Uc\$
- Discord
 - <https://discord.gg/uEzZ9gsnZn>
- GitHub
 - <https://docs.github.com/en/free-pro-team@latest/github/getting-started-with-github/signing-up-for-a-new-github-account>
- Make time! - You get back what you put in
 - 9am-12pm for three days with us
 - 2-3 hours a day with teammates and solo coding
- Camera off, audio off during presentations

GitHub pull requests (PRs)

<https://docs.github.com/en/free-pro-team@latest/github/getting-started-with-github/fork-a-repo#fork-an-example-repository>

You can practice forking on: <https://github.com/octocat/Spoon-Knife>



GitHub pull requests (PRs)

<https://github.com/microbinfie-hackathon2020/CSIS/blob/main/CONTRIBUTING.md>



What to test for

- Does it work?
 - Does it install?
 - Exit code 0
 - Output file created
- Does it do some standard things
 - `biotool --version`
 - `biotool --help`
 - `biotool #` usually similar to `--help`
 - `biotool zero_byte.fastq`
 - `biotool --numcpus two #` should raise an error; should be "2"
- Don't pull your hair out
 - Only test on one or a few test files
 - Don't test every single function in the code
 - Don't unit test on each dependency
 - But *do* test to make sure dependencies are available



What we are doing and what we are not doing

NOTE

We are unit testing others' software

- Installing dependencies
- Installing target software
- Running the target software

Do not develop for other projects for this hackathon

- Don't fix their software
- Don't debug their software

Seemann GigaScience 2013, 2:15
<http://www.gigasciencejournal.com/content/2/1/15>



COMMENTARY

Open Access

Ten recommendations for creating usable bioinformatics command line software

Torsten Seemann^{1,2}

Abstract

Bioinformatics software varies greatly in quality. In terms of usability, the command line interface is the first experience a user will have of a tool. Unfortunately, this is often also the last time a tool will be used. Here I present ten recommendations for command line software author's tools to follow, which I believe would greatly improve the uptake and usability of their products, waste less user's time, and improve the quality of scientific analyses.

Keywords: Bioinformatics software, Software quality, User interface, Unix, Tools

Background

New bioinformatics tools are released and published every day, most of which are designed for the Unix command line. Ignoring the important issue of algorithmic correctness, the first barrier for community uptake of a bioinformatics tool is the command line interface and usability. It is this author's experience that the majority of these tools fail basic requirements of usability; and thus, a course of action to overcome this would be a list of minimum standards for all command line scientific software that would help software authors and reviewers to improve the average usability of released software tools.

I have used and installed a *lot* of bioinformatics software over the last 12 years, and I have also released a lot of my own software - I try to make it as painless to use as possible. From these experiences, I present ten recommendations for bioinformatics software, using the fictitious "BioTool" project as an example.

% biotool
Please use the --help option to get usage information.



¹ Print something if no parameters are supplied

Some suggested software

- Kraken2
- "Torstyverse"
- FastQC
- MultiQC
- Assemblers - SKESA, Unicycler
- SARS-CoV-2 software

These are only suggestions.



Heads up!

In a few slides, we will ask you to introduce yourselves...


Video encouraged, but not required

- Name
- Where you work or what you do
- What is your role in the hackathon?
- Tell us something interesting about yourself



Hello
my name is

Lee Katz
Enteric Diseases Laboratory Branch
CDC
Stakeholder; will help other groups

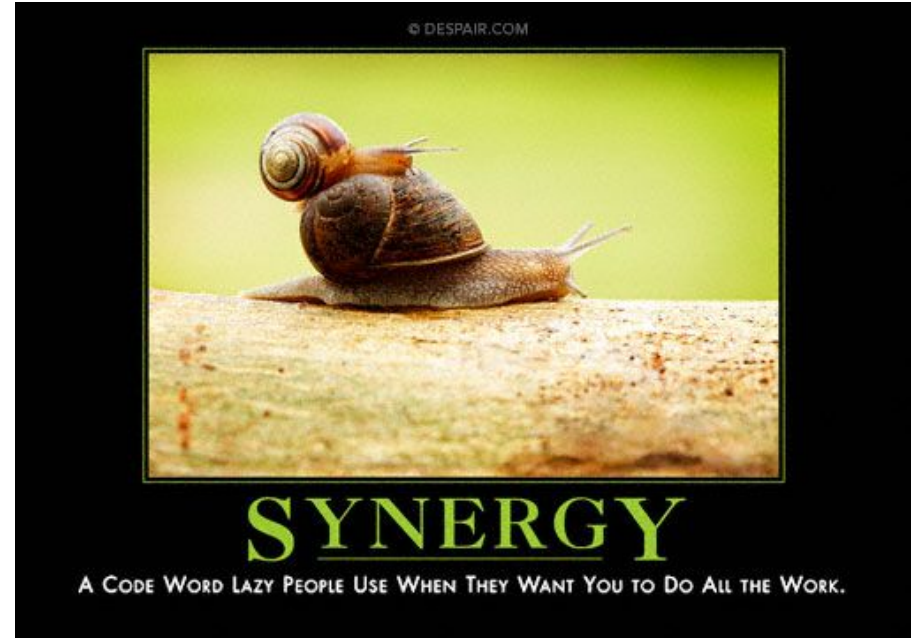


Forming teams

The **stakeholder** - the person who cares and who understands the project. Sometimes called the "**subject matter expert.**"

The **developers** - knowledge of git, yaml, command line

The **data wrangler** - getting data for the team



Purpose

- To show which bioinformatics software works as intended
- To award a green badge to software that works
- To encourage unit testing in target software
- To encourage unit testing in the bioinformatics community
- **Deliverable:** a badge on the CSIS site describing whether software passed or didn't pass

Current software

Software	badge with link to CI	version badge	yaml
This repo			CSIS.yml
Prokka			prokka.yml
Quast			quast.yml

Introductions - video encouraged but not required

- Name
- Where you work or what you do
- What is your role in the hackathon?
- Tell us something interesting about yourself

The **stakeholder** - the person who cares and who understands the project. Sometimes called the "**subject matter expert**."

The **developers** - knowledge of git, yaml, command line

The **data wrangler** - getting data for the team

The rest of the hackathon

Up next: Boas will show us examples in unit testing

After that: we'll form teams and form individual projects

Until tomorrow: your teams will meet and code

Tomorrow: team updates; keynote

Until Friday: teams will meet and code

Friday: wrap up