Learning Perl Through Examples Part 2

L1110@BUMC

2/8/2018

BOSTON UNIVERSITY

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018

Tutorial Resource

Before we start, please take a note - all the codes and supporting documents are accessible through:

- http://rcs.bu.edu/examples/perl/tutorials/

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Sign In Sheet

We prepared sign-in sheet for each one to sign
We do this for internal management and quality control
So please SIGN IN if you haven't done so

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Evaluation

One last piece of information before we start:

- DON'T FORGET TO GO TO:
  - http://rcs.bu.edu/survey/tutorial_evaluation.html

Leave your feedback for this tutorial (both good and bad as long as it is honest are welcome. Thank you)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Today's Topic

- Basics on creating your code
- About Today's Agenda – two tracks (options)
  - Option 1 : hands on experiments on a simple bioinformatical example
    - Fanconi example #1, #2, #3
  - Option 2: code review on a complicated pipeline for PPI detections
    - HuRI pipeline

Please VOTE Your Choice !

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Basics on creating your code

How to combine specs, tools, modules and knowledge.

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018

What is needed

Consider your code/software a '**product**', what will it take to **produce** **it**?

- **User Requirements (domain knowledge, that's very important)**
- Development Environment (Emacs/gedit/Eclipse/etc)
- Third Party Modules/Toolboxes (CPAN)
- Some workman's craft (You/Programmer)
- Help systems (Help documentation/reference books/stackflow/etc)
- Language specification (Perldoc/reference guide)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

User Requirements

Specify what software is expected to do

Can be formal or casual, but better keep records of.

- Formal – User Requirement Documentation (URD)
- Casual – email conversations, scratch paper memos, etc.

Types of Requirements

- M – Mandatory
- D – Desirable
- O – Optional
- E – Enhanceable

Serve as contract – keep project on track

Pitfall – often ignored

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Development Environment

It is like your workshop where you go to work and make your product

How to pick your development tools (mainly editor or IDE)

- Convenient
- Sufficient enough
- Extensible/adaptive
- Personal preference

BOSTON UNIVERSITY

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Development Environment

Some commonly used tools:

1) Editor Only:
   - emacs
   - vim
   - gedit

2) IDE (Integrated Development Environment)
   - Eclipse
   - Padre

You may go to http://perlide.org/poll200910/ for the poll result conducted by a Perl guru for Perl Editors

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

CPAN – Where Third Party Modules Resides

- Perl is a community built software system, enriched by third party contributors. All efforts go to build CPAN open source archive network for Perl.
- Perl's richness and power comes from CPAN and the 3rd party modules and toolkits covering various domains, for example, Finance, BioPerl, Catalyst, DBI, and many others.
- CPAN official site: www.cpan.org
- Two search engine interfaces:

  search.cpan.org (old, traditional)

  metacpan.org (new, modern, provides rich APIs for automation)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Help systems

One significant criteria for a good programming language is its documentation and help system – In this sense, Perl is quite good

Its own:

- Language Specification itself well written
- Organized well (divided by categories)
- Presented well (perldoc utility/man, Internet available)

Online Resource:

- Rich online help, tutorials, and e-books (many for free)

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Language specification

Also called ‘Reference Guide’

Perldoc Official Site: http://perldoc.perl.org

Divided to eight subcategories:

1. Language
2. Functions
3. Operators
4. Special variables
5. Pragmas
6. Utilities
7. Internals
8. Platform Specific

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Workman's Crafts

Hard Part

Takes time to build, but takes no time to start (practice is the best way to learn)

Skills Needed Include:

- Familiarity to language elements
- Software Engineering Methodology
- Algorithm Design
- Code Implementation
- Debugging
- Domain knowledge

Metaphor : How do we acquire skills on natural language

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Open the HuRI pipeline code …

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018

Take a look at main program structure

- Authorship (line 1-5)
- Header (line 7-25)
- Initialization and Configuration (line 27-118)
- Setup Pipeline runtime environment - initialize DB connection(s), etc (line 120-124)
- Call main functions of the pipeline (line 126-131)
- Clean up/reclaim resources – release DB connections, etc. (line 135-138)

Take a look at Main Functions

- Mail Haul function:
  SWIM_pipeline()

  Highlights:
  10 configurable steps to perform a series of tasks at each different stage (pipeline) in the life cycle of a research project.

- Other Maintenance (housekeeping) functions:
  del_pool(), del_pkg(), etc

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

SWIM_pipeline()

10 pipeline steps:

STEP1: record sequence batch information (line 185-209)

STEP2: mapping the plate info with sequence returned (line 211-326)

STEP 3: create pool for each logical batch according to project/wet lab experiment design (line 328-432)

STEP 4 and STEP 5: do seq. alignment and preliminary analysis plate by plate (line 434-463)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

SWIM_pipeline() (continue)

10 pipeline steps:

STEP 6: do IST assembly (line 466-470)

STEP 7: do QC for each plate (line 472-475)

STEP 8: do post analysis accordingly plate by plate, and get summary report (line 477-482)

STEP 9: build analysis package, record analysis parameters, and other related info. (line 484-519)

STEP 10: do Node QC to check the quality of the original clone data, for diagnosis (line 521-524)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Helper functions

These are functions that may be out of the pipeline logic, but serves as building blocks for the pipeline functions. It can be shared among the different pipeline stages or dedicated to certain single pipeline stage, but just for the sake of clarity and modularity, being separated outside the pipeline main structure.

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Helper functions (continue)

In SWIM_pipeline, I have defined total 8 helper functions (but only for step 2-10, why? Will share the reason):

hlp2b_get_pool_data() (line 3376-3441)
hlp2_record_pool() (line 3285-3374)
hlp3_align_seq() (line 2984-3279)
hlp4_analyze_align() (line 2000-2981)
hlp5_get_ist() (line 1521-2077)
hlp6_run_QC() (line 1049-1518)
hlp7_get_plate_summary() (line 809-1038)
hlp8_nodeQC() (line 536-806)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Helper functions (continue)

In SWIM_pipeline, I have defined total 8 helper functions (but only for step 2-10, why? Will share the reason):

hlp2b_get_pool_data() (line 3376-3441)
hlp2_record_pool() (line 3285-3374)
hlp3_align_seq() (line 2984-3279)
hlp4_analyze_align() (line 2000-2981)
hlp5_get_ist() (line 1521-2077)
hlp6_run_QC() (line 1049-1518)
hlp7_get_plate_summary() (line 809-1038)
hlp8_nodeQC() (line 536-806)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Side Helper functions

In SWIM_pipeline, I called those functions serve as the helpers for side functionality of the pipeline ‘SideHelper’. They are important for project and data management, but may not directly connect to the sequence alignment pipeline functionality.

such side helper function defined in the pipeline is:

Sh1_create_pkg() (line 3454-3779)
Sh1_create_pkg2() (line 3781-4119)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Essential subroutines

In SWIM_pipeline, I called those functions serve as essential and repeated used subroutines ‘subroutines’. These functions usually are very simple and single purposed to only achieve one thing. But it is atomic and efficient, and will be called at various places again and again.

such subroutines defined in the pipeline are:
s3_get_ist_table() (line 4134-4179)
s2_get_empty_well_list() (line 4181-4222)
s1_get_well_index() (line 4224-4270)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Utilities

In HURI pipeline, I defined some general purposed utility functions which serve as the tool set to conveniently handle data or manage the databases.

such utilities defined in the pipeline are:
Ordered_hash_ref() (line 4287-4291)
Connect_db() (line 4293-4301)
Disconnect_db() (line 4303-4308)
del_pool() (line 4312-4497)
del_pkg() (line 4502-4559)
prepare_db() (line 4562-4582)

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

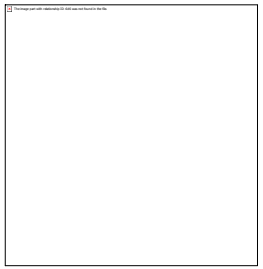
Some Thoughts on Pipeline Design

BOSTON UNIVERSITY

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018

What is Pipeline



www.perl.org

Long and complicate definition - Wikipedia:
https://en.wikipedia.org/wiki/Pipeline_(computing)

Simple and easy way to remember:
connected modules in a tandem line with input/output data flow, each module accomplishes a relatively single purposed task

Recommend article:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5429012/

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Perl as pipeline script tool

Scripts

Scripts, written in Unix shell or other scripting languages such as Perl, can be seen as the most basic form of pipeline framework. Scripting allows variables and conditional logic to be used to build flexible pipelines. However, in terms of 'robustness', as defined by Sussman [2], scripts tend to be quite brittle. In particular, scripts lack two key features necessary for the efficient processing of data: support for 'dependencies' and 'reentrancy'. … … Pipelines often include steps that fail for any number of reasons such as network or disk issues, file corruption or bugs. A pipeline must be able to recover from the nearest checkpoint rather than overwrite or 'clobber' otherwise usable intermediate files. In addition, the introduction of new upstream files, such as samples, in an analysis should not necessitate reprocessing existing samples. (cited from 'A Review of bioinformatic pipeline frameworks')

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

pipeline frameworks from ‘A Review’ paper

Syntax	Paradigm	Interaction	Example	Ease of Development	Ease of Use	Performance
Implicit	Convention	CLI	Snakemake, Nextflow, BigDataScript	★★★½	★★★★½	★★★
Explicit	Convention	CLI	Ruffus, bpipe	★★★★	★★★★	★★★
Explicit	Configuration	CLI	Pegasus	★★½	★★★	★★★★
Explicit	Class	CLI	Queue, Toil	★★½	★★★	★★★★★
Implicit	Class	CLI	Luigi	★★★	★★★½	★★★★★
Explicit	Configuration	Open Source Server Workbench	Galaxy, Taverna	★★★	★★★★½	★★★
Explicit	Configuration	Commercial Cloud Workbench	DNAnexus, SevenBridges	★★½	★★★★★	★★★★½
Explicit	Configuration	Open Source Cloud API	Arvados, Agave	★★★	★★★★	★★★★½

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Q & A

BOSTON UNIVERSITY
Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018

Evaluation Please @

http://scv.bu.edu/survey/tutorial_evaluation.html

Thank You !!

BOSTON UNIVERSITY

Yun Shen, Programmer Analyst
yshen16@bu.edu
IS&T Research Computing Services

Spring 2018