EIC Software Documentation

Wouter Deconinck, William & Mary

May 20-21, 2019
EIC Software Meeting

Supported by the National Science Foundation under Grant Nos. PHY-1405857, PHY-1714792.

What is User-Centered Design? I

User-centered design is a multi-stage problem-solving process that requires designers

- to *analyze and envision the way users are likely to consume a product*,
- to *validate their assumptions* with regard to the user behavior in real world tests.

In this case: how do the users of EIC software want to ‘consume’ the software?

Who Are the Users of EIC Software? I

Who are the scientific computing users?

- Long-term researchers
  - Faculty, staff scientists
  - Experienced in the traditional ways, too busy to learn a new language or software tool (unless we make it easy: it is our task to know the users' attitudes)
- Short-term researchers
  - Undergraduate, graduate researchers, post-doctoral researchers
  - Trained (sometimes) in the latest languages, motivated to learn and apply new skills if empowered and transferable to future opportunities

Some User Stories (fictionalized) I

David, university professor

David is a university professor who is currently not very involved in EIC development, but wishes to evaluate how his favorite production and decay process looks in various interaction region and detector concept designs. David will just use the standard beam parameters and detector geometries. David does not have the time nor inclination to checkout and compile a simulation framework from scratch. David just wants to get the initial results as quickly as possible before deciding whether to invest more time.

Some User Stories (fictionalized) II

Marie, lab researcher

Marie is a researcher at lab A who was assigned to evaluate how well she can study ‘her’ reaction channel with the parameters of a detector system in development at lab B. Although Marie is very familiar with the simulation and analysis code of lab A, there is simply not enough time in a week to also learn the framework of lab B.

Patrina, graduate student

Patrina is a new graduate student and her advisor told her to “get familiar with the EIC software stack before we meet again.” Patrina is still waiting for her computing account to be activated, and all these gitlab and github commands make her head spin. Patrina feels overwhelmed and is worried about the lack of progress.

Some User Stories (fictionalized) III

Markus, developer

Markus is a developer of the software stack of lab J. Markus lives and breathes code. Markus needs no documentation when he has the source code. Markus just does ‘git clone $url 2>&1 && vi INSTALL.md‘. Markus’s needs are typically already satisfied. We will mostly ignore Markus.

User-Center Design Has Outcomes Already I

Containers

- User advantages: shorter time-to-physics, less time compiling and debugging
- Also, of course, technical advantages
- But also a less familiar environment for those steeped in the old ways
- Documentation is at the core of providing an entry point to users

Using Software Carpentry tutorial format I

Mission of The Carpentries, parent of Software Carpentry

> *[...] We collaboratively develop openly-available lessons and deliver these lessons using evidence-based teaching practices. We focus on people conducting and supporting research.*

THE CARPENTRIES

- Also: Data Carpentry, Library Carpentry

Using Software Carpentry tutorial format II

How does it work?

- All documentation hosted in GitHub or GitLab
- Everything is in markdown `md` or restructured text `rst`
- Editing can be done entirely in browser (or `git clone`)
- CI automatically runs jekyll to create html version

Typical module structure

- `Setup` with prerequisites (e.g. link to how to install/run singularity)
- Followed by 10 minute segments
- Each segment constructed around key questions and learning objectives
- Each segment ends with a recap of key points
- Strong focus on active learning: enter commands, do things, not just reading

Using Software Carpentry tutorial format III

Jefferson Lab Singularity Tutorial

- `https://github.com/jeffersonlab/swcarpentry-jlab-singularity/`
- `https://jeffersonlab.github.io/swcarpentry-jlab-singularity/`

Jefferson Lab Jupyter Tutorial

- `https://github.com/jeffersonlab/swcarpentry-jlab-jupyter/`
- `https://jeffersonlab.github.io/swcarpentry-jlab-jupyter/`

Using Software Carpentry tutorial format IV

Home Code of Conduct Setup Episodes Extras License Improve this page Search...

Using Singularity at Jefferson Lab

Software containers are lightweight, stand-alone, executable packages that includes everything needed to run: code, runtime, system tools, system libraries, settings.

Containers are similar to virtual machines, however, the virtualization happens at a different point.

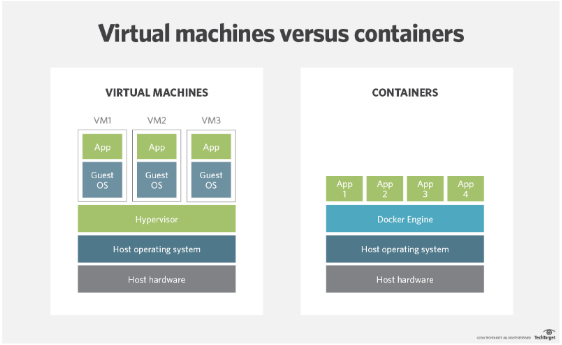
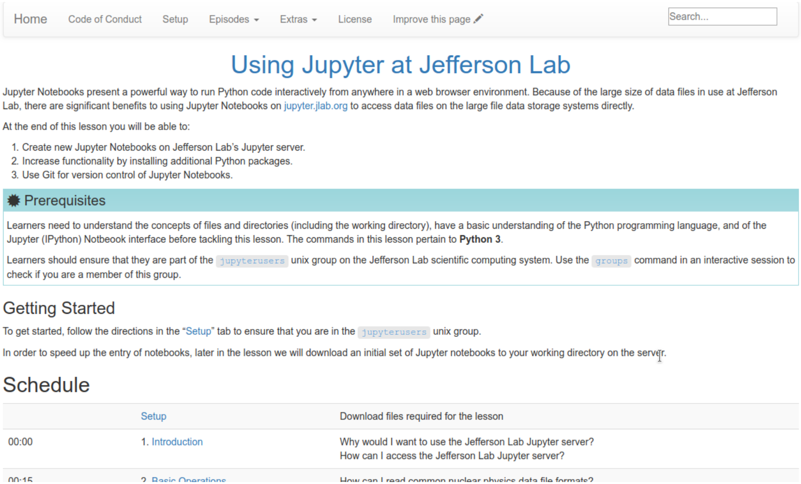


Image source: TechTarget

Using Software Carpentry tutorial format V



Using Jupyter at Jefferson Lab

Jupyter Notebooks present a powerful way to run Python code interactively from anywhere in a web browser environment. Because of the large size of data files in use at Jefferson Lab, there are significant benefits to using Jupyter Notebooks on jupyter.jlab.org to access data files on the large file data storage systems directly.

At the end of this lesson you will be able to:

1. Create new Jupyter Notebooks on Jefferson Lab's Jupyter server.
2. Increase functionality by installing additional Python packages.
3. Use Git for version control of Jupyter Notebooks.

Prerequisites

Learners need to understand the concepts of files and directories (including the working directory), have a basic understanding of the Python programming language, and of the Jupyter (IPython) Notbeook interface before tackling this lesson. The commands in this lesson pertain to **Python 3**.

Learners should ensure that they are part of the `jupyterusers` unix group on the Jefferson Lab scientific computing system. Use the `groups` command in an interactive session to check if you are a member of this group.

Getting Started

To get started, follow the directions in the "Setup" tab to ensure that you are in the `jupyterusers` unix group.

In order to speed up the entry of notebooks, later in the lesson we will download an initial set of Jupyter notebooks to your working directory on the server.

Schedule

	Setup	Download files required for the lesson
00:00	1. Introduction	Why would I want to use the Jefferson Lab Jupyter server? How can I access the Jefferson Lab Jupyter server?
00:15	2. Basic Operations	How can I read common nuclear physics data file formats?

Using Software Carpentry tutorial format VI

_episodes/01-using-docker-containers.md

```
title: "Using Docker Containers"
teaching: 5
exercises: 5
questions:
- "How can I most quickly download and use the latest J
objectives:
- "Understand container instantiation, versioning and c
keypoints:
- "Docker containers allow you to use the JLEIC simulat
```

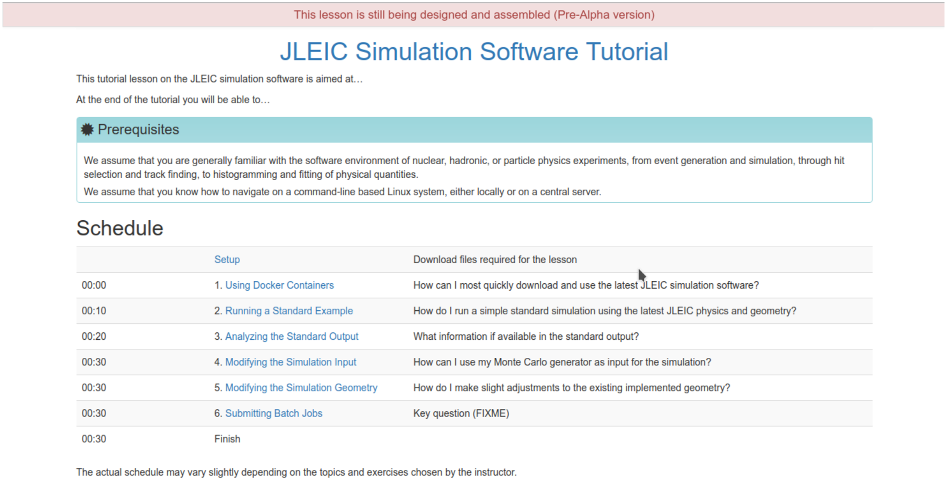
Followed by segment content.

Using Software Carpentry tutorial format VII

First example

- JLEIC simulation quickstart structure
- `https://gitlab.com/ESC/swcarpentry-jleic-simulation/`
- `https://ESC.gitlab.io/swcarpentry-jleic-simulation/`

Using Software Carpentry tutorial format VIII



This lesson is still being designed and assembled (Pre-Alpha version)

JLEIC Simulation Software Tutorial

This tutorial lesson on the JLEIC simulation software is aimed at…

At the end of the tutorial you will be able to…

Prerequisites

We assume that you are generally familiar with the software environment of nuclear, hadronic, or particle physics experiments, from event generation and simulation, through hit selection and track finding, to histogramming and fitting of physical quantities.

We assume that you know how to navigate on a command-line based Linux system, either locally or on a central server.

Schedule

	Setup	Download files required for the lesson
00:00	1. Using Docker Containers	How can I most quickly download and use the latest JLEIC simulation software?
00:10	2. Running a Standard Example	How do I run a simple standard simulation using the latest JLEIC physics and geometry?
00:20	3. Analyzing the Standard Output	What information if available in the standard output?
00:30	4. Modifying the Simulation Input	How can I use my Monte Carlo generator as input for the simulation?
00:30	5. Modifying the Simulation Geometry	How do I make slight adjustments to the existing implemented geometry?
00:30	6. Submitting Batch Jobs	Key question (FIXME)
00:30	Finish	

The actual schedule may vary slightly depending on the topics and exercises chosen by the instructor.

Status and Plans I

Intentions

- Tutorial for each of the major software frameworks
- Co-hosted in similar format
- Intended to demonstrate parallel functionality

Lab	Quickstart	Output	Mod. Input	Mod. Geom.
ANL				
BNL				
JLEIC				
JLab/BNL	✓	✓	✓	✓

Timeline I

Common JLab/BNL Simulation: Demonstration test case

- Simulation tutorial development: in progress, completion by mid-June
- Simulation tutorial testing/feedback: mid-June through Summer 2019 EIC UG meeting

Development of modules for ANL, BNL, JLab specific software

- Fall 2019

EIC Software tutorial sessions

- Unclear as of yet in what form:
  - in-person
  - online synchronous
  - online asynchronous

Discussion

- Who wants to commit to helping develop this entry point for users?
- Who wants to help with testing the documentation for stacks outside of their immediate expertise?