

How to start g4e with docker

Yuya Ohsumi

github :

<https://github.com/0419YOhsumi>

環境

- 環境 : WSL2 (Ubuntu 20.04 LTS)
WSL だと docker を使えない
既に WSL が入っている場合、それを WSL2 に update することも可能だが、それまでの WSL の環境をそのまま引き継ぐことができるか心配なので、WSL2 を install する方がおすすめ
- docker commands
 - \$ docker ps : ls と同じ + docker の稼働状況確認
 - \$ \$ docker ps -a : 自分が使ったコンテナ一覧 (履歴まで見れる)
 - \$ docker pull ~ : open に提供されているソフト (image) を local の docker に持ってくる
 - \$ docker run ~ : 持ってきた image を docker で展開 (container 作成)

docker を使って g4e の走らせる (1)

- WSL2 で以下の commands をうつ
\$ sudo service docker start (必ず sudo でないとできない)
starting docker と出たら OK
docker が走っているか確認するには、
\$ service docker status
docker is not running と出る場合、windows power shell で
\$ wsl --update
とうって wsl を最新になっているか確認する
また、調子悪いときは再起動するとだいたい直ります
- g4e の image を、EIC User Group からもらってくる
\$ docker pull electronioncollider/epic-gui
(初めて pull するときだけで OK)

docker を使って g4e を走らせる (2)

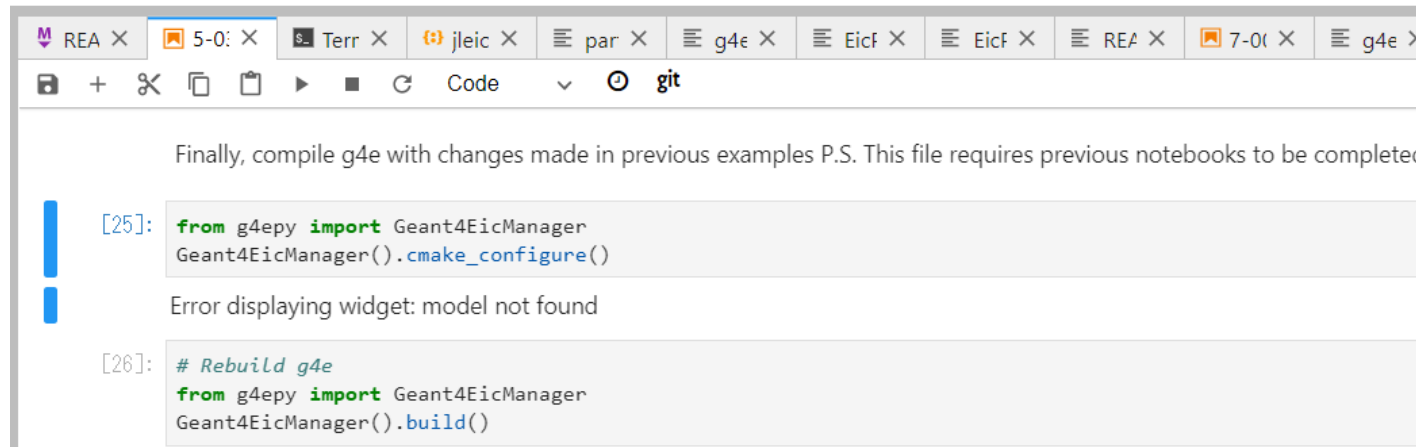
- EIC User Group から pull してきた image を実際に local で走らせる
\$ docker run -it -rm -p8888:8888 -p6080:6080
(rm をつけると、毎回 clean な状態、default の状態から始まる)
(自分の保存した g4e を再開する場合の command は後述)
- ブラウザで、localhost:8888 を開くと、JupyterLab 上で g4e が使える
- 終了するとき
ブラウザで開いている JupyterLab をシャットダウンして、
WSL2 のターミナルで ctrl C

g4e on JupyterLab (1) : directory 構造

- /home/eicuser/epw
 - /01_fast_sim_tutorial
 - /02_full_sim_tutorial (この directory が main の作業 directory)
 - /g4e (この directory は次のスライドのように cmake, build をすると作成される)
 - /1-01_dive_in.ipynb
 - ~ (tutorial の話 とりあえず無視してください)
 - /7_00_shortcut.ipynb (ここに cmake, build の commands も載っているので、便利)
 - /03_explore
 - /04_development
 - /data (ここに generator の source codes とかある herwig とか)
 - /share
 - /img
 - /README.md
- 基本的に 02_full_sim_tutorial 以下の directory で作業する

g4e on JupyterLab (2) : g4e の環境整備

- g4e を cmake, build する : 02_full_sim_tutorial に移動して以下の commands を、cell で実行する



The screenshot shows a JupyterLab window with multiple tabs. The active tab is labeled '5-0:'. Below the tabs is a toolbar with icons for file operations and a 'Code' dropdown menu. The main area displays a code cell with the following content:

```
Finally, compile g4e with changes made in previous examples P.S. This file requires previous notebooks to be completed
```

```
[25]: from g4epy import Geant4EicManager
      Geant4EicManager().cmake_configure()
```

Error displaying widget: model not found

```
[26]: # Rebuild g4e
      from g4epy import Geant4EicManager
      Geant4EicManager().build()
```

(Geant4 のコードを edit するたびにコンパイルするので、自分が作業するタブを作り、そこにこの cell を張っておくと便利です)

- 自分のいる directory に g4e という directory ができる
(一回できたら、あとはその下の directory, file をいじるだけ)

g4e on JupyterLab (3) : 作成された g4e directory の構造

- /home/eicuser/epw/02_full_sim_tutorial
 /g4e (僕の github の 0419YOhsumi/g4e_sim/ にある directory
 は下の directory に対応しています)
 - /bin
 - /cmake
 - /cmake-build-debug
 - /docs
 - /g4e-dev
 - /python (g4e を走らせる、main の python code, g4e.py がある)
 - /resource
 - /src (g4e を走らせる、main の c++ code, g4e.cc や detector
design 等の code が入っている)
 - /test
 - /uicmd_generator
 - /CMakeLists.txt
 - ...
- 基本的に、 g4e.py と src 以下の file しかいじっていない

g4e on JupyterLab (4) : github から file を取ってくる

- ex) home/eicuser/epw/02_full_sim_tutorial/g4e/src/g4e.cc
を、github/0419YOhsumi/g4e_sim/g4e/src/g4e.cc に置き換
える
前提 : github ように、github の working directory を作ってお
くと安全かつ分かりやすい
例えば home/eicuser/epw/ に git_works という directory を
作成、そこに移動
\$ git clone https://github.com/0419YOhsumi/g4e_sim.git
\$ cd g4e/src/
(github 上での、g4e_sim/g4e/src に対応)
\$ cp g4e.cc
home/eicuser/epw/02_full_sim_tutorial/g4e/src/g4e.cc
(original の g4e.cc を、github 上の g4e.cc に書き換えた)
- 場所は自分で指定し、コンパイルしなす必要あり

g4e on JupyterLab (5) : run

- ex) particle_gun.py で particle gun を撃つ
home/eicuser/epw/02_full_sim_tutorial/ に
particle_gun.py を github から持ってくる (前ページ参)
\$ cd /home/eicuser/epw/02_full_sim_tutorial
\$ python particle_gun.py
これで g4e が走ります
- particle_gun.py の中身
 - line 1~3 : g4e.py の実行、
Geant4EIC() 作成
 - line 5 : 関数の名前
 - line 7 : output する filename
 - line 8 : source にする file
(GPS なら、ここはなんでも良い
herwig とかほかの generator を使う
ときは、その file をここに入れる)
 - line 10~12 : 走らせる

```
1 import inspect
2 from os import path
3 from g4epy import Geant4Eic
4
5 g4e = Geant4Eic(detector='jleic', beamline='erhic') \
6
7 g4e .output('pi0_10GeV') \
8     .source('../data/cone_pgun_my.mac') \
9
10 g4e
11
12 g4e.run()
```

docker を使って g4e を保存、再開する

- Ctrl C で JupyterLab を閉じた後、\$ docker ps とつつ

- まだ g4e
(自分の場合は
3ceec6662313
という

container 名)
が稼働中



\$ docker stop ~
で終了

- 再開するとき
\$ docker start -a
(container 名)

```
yuya@LAPTOP-ILM3Q2O4: ~  
[sudo] password for yuya:  
* Starting Docker: docker  
yuya@LAPTOP-ILM3Q2O4: ~$ docker start -a 3ceec6662313  
VNC PASSWORD in ~/.vnc/passwd already set to: 123456  
expr: syntax error  
Launching noVNC on port 6080 ...  
  
Point your HTML5 enabled browser on the host to:  
    http://localhost:6080  
  
Alternatively, point your VNC client to localhost:  
    VNC PASSWORD: 123456  
  
[W 06:58:46.881 LabApp] All authentication is disabled. Anyone who can connect to this server will be able to run code.  
[I 06:58:46.949 LabApp] Loading IPython parallel extension  
[I 06:58:47.224 LabApp] JupyterLab extension loaded from /usr/local/lib/python3.7/site-packages/jupyterlab  
[I 06:58:47.225 LabApp] JupyterLab application directory is /usr/local/share/jupyter/lab  
[I 06:58:48.005 LabApp] Setting Pyjano nb server extension  
/home/eicuser/epw  
=====
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3ceec6662313	electronioncollider/epic-gui	/tini -- /bin/sh -c..."	6 months ago	Up 29 minutes	0.0.0.0:6080->6080/tcp, 5900/tcp, 0.0.0.0:8888->8888/tcp	great_mccarth

```
yuya@LAPTOP-ILM3Q2O4: ~$ .
```

g4e (on JupyterLab) での git への access

- JupyterLab でターミナルを開く

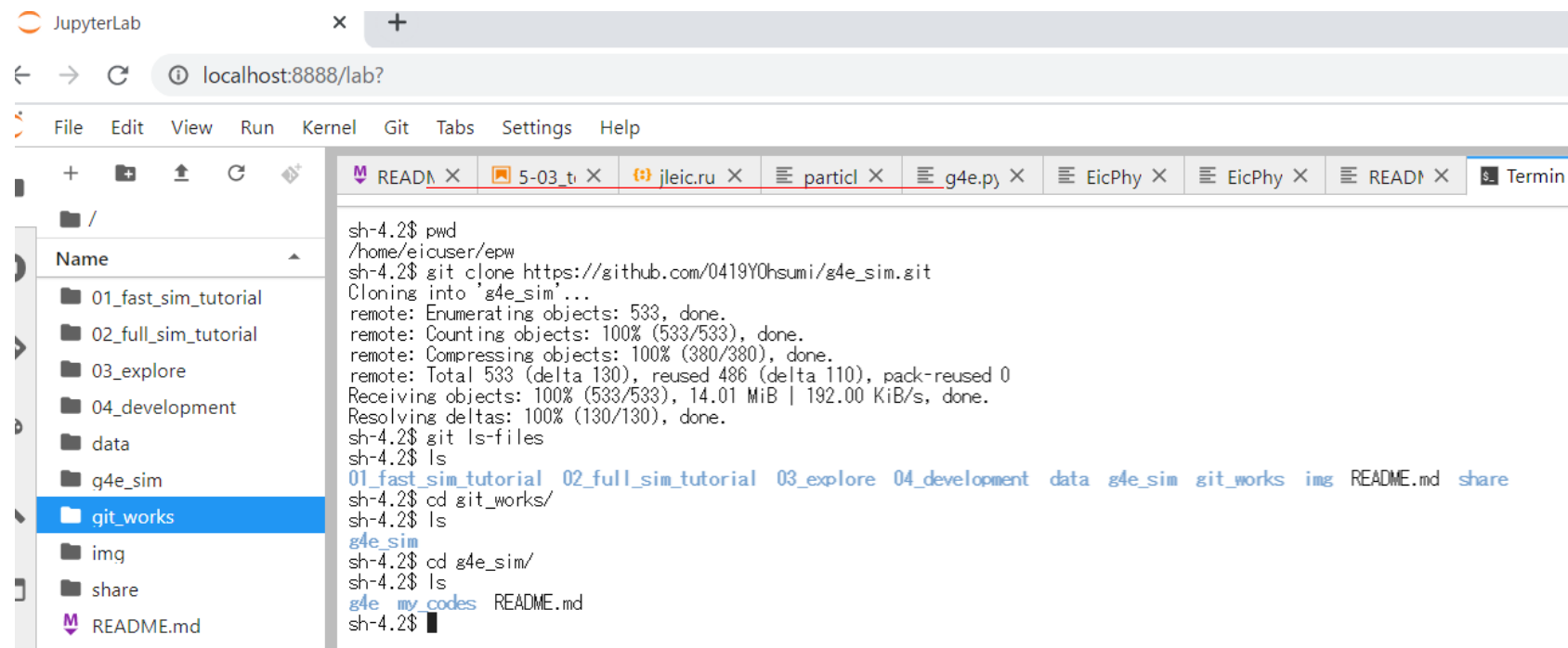


git 用の directory を作成



\$ git clone (github のリンク).git

とっただけでその
github 上に
access 完了



The screenshot shows the JupyterLab web interface in a browser window. The address bar shows 'localhost:8888/lab?'. The top menu bar includes File, Edit, View, Run, Kernel, Git, Tabs, Settings, and Help. Below the menu is a toolbar with icons for file operations. On the left is a file explorer sidebar showing a directory tree with folders like '01_fast_sim_tutorial', '02_full_sim_tutorial', '03_explore', '04_development', 'data', 'g4e_sim', 'git_works' (highlighted in blue), 'img', 'share', and a 'README.md' file. The main area on the right is a terminal window with the following output:

```
sh-4.2$ pwd
/home/eicuser/epw
sh-4.2$ git clone https://github.com/0419Y0hsumi/g4e_sim.git
Cloning into 'g4e_sim'...
remote: Enumerating objects: 533, done.
remote: Counting objects: 100% (533/533), done.
remote: Compressing objects: 100% (380/380), done.
remote: Total 533 (delta 130), reused 486 (delta 110), pack-reused 0
Receiving objects: 100% (533/533), 14.01 MiB | 192.00 KiB/s, done.
Resolving deltas: 100% (130/130), done.
sh-4.2$ ls -ls
total 12
drwxr-xr-x 2 root root 4096 Sep  1 10:00 .
drwxr-xr-x 1 root root 4096 Sep  1 10:00 ..
-rw-r--r-- 1 root root  120 Sep  1 10:00 .gitignore
-rw-r--r-- 1 root root  120 Sep  1 10:00 README.md
sh-4.2$ cd git_works/
sh-4.2$ ls
g4e_sim
sh-4.2$ cd g4e_sim/
sh-4.2$ ls
g4e  my_codes  README.md
sh-4.2$
```

Back up (1) : github のはなし (master の場合)

- Local の環境に公開されている git のコードをとってくるとき
\$ git clone (access したい git への URL).git
ex) \$ git clone https://github.com/0419YOhsumi/g4e_sim.git
- コードを編集して、それを git に反映させる
\$ git add (編集したコード or ファイル)
 - local index に追加
(git add . とうつと変更された内容すべて指定、となる)
\$ git commit -m “ 何かしらのコメント “
 - github にあげるための準備 どのような変更をしたか、という簡潔なもの
\$ git remote add (access したい git への URL).git
 - remote repository へ access
\$ git push -u origin master
 - remote repository に編集内容を反映
- ただし、authority を持った人でないと、編集内容を git に反映できない

Back up (2) : install docker

- Docker install

docker を WSL2 に install する手順 commands

```
$ sudo apt update ( update してからでないと install できない可能性あり )
```

```
$ sudo apt install apt-transport-https  
ca-certificates curl software-properties-common  
( docker install に必要な package )
```

(環境によっては、あとで足りないといわれる package があるかもしれないが、その時は言われたものを sudo apt install) Docker
install on WSL2 手順 commands 続き

```
$ curl -fsSL
```

```
https://download.docker.com/linux/ubuntu/gpg
```

```
| sudo apt-key add -
```

```
$ sudo add-apt-repository "deb [arch=amd64]
```

```
https://download.docker.com/linux/ubuntu focal stable"
```

```
$ sudo apt update
```

```
$ sudo apt install docker-ce
```