Joint Institute for Nuclear Research

Incorporating Docker into software development process

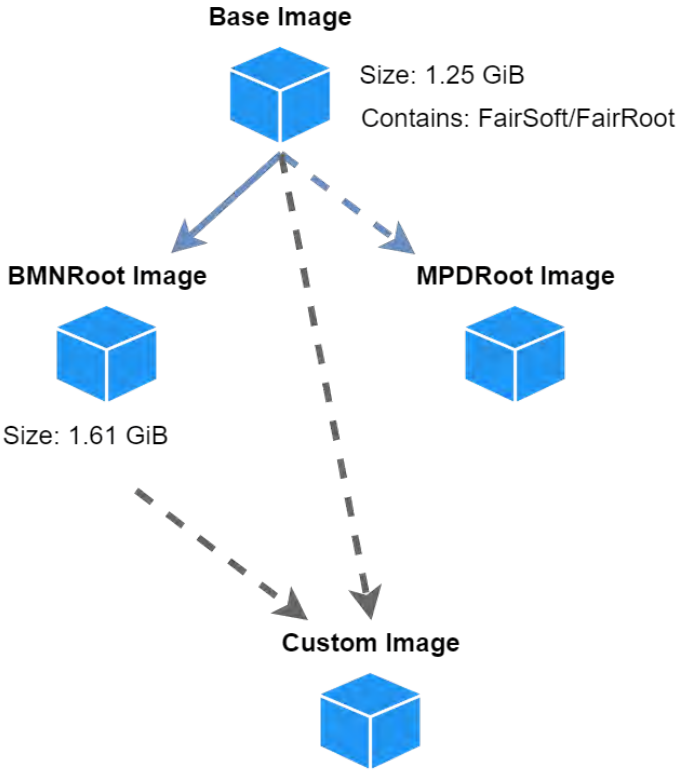
Nikita Balashov

Possible Use-Cases

- User docker containers
  - Users don’t need to install software – just run container
  - Hosting computer can potentially run any operating system
- Containers in CI
  - Simplify CI-infrastructure
  - Quickly add any OS environments to CI pipelines
- Jupyter
  - Everything in a browser (seems exotic?)

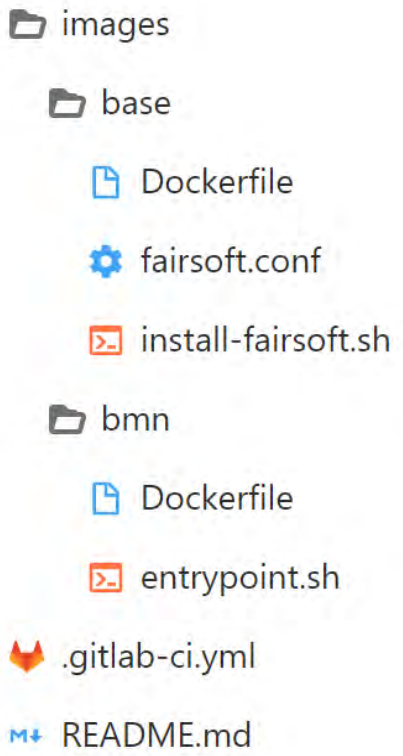
User Containers

- **Base** image: Ubuntu 18.04 (LTS) + FAIRsoft + Fairroot
- A set of images per project, all built on top of the **Base** image: **bmnroot**, **mpdroot**, etc.
- Images can have **tags** similar to git tags
- Automatically built and published with GitLab CI
- Stored in GitLab Container Registry
- Can be used as a base for custom containers
- Use like a virtual machine
- Interactive access via ssh with X-forwarding to get the graphics



User Containers Pipeline

- A dedicated GitLab project under NICA group
- Repository contains all the Dockerfiles, configuration and scripts to build the software and containers
- Pipeline consists of **two** jobs: **build_base** and **build_bmn**
- **build_base** is automatically triggered on any changes in images/base/
- **Build_bmn** is triggered through the API from the main bmnroot pipeline



Containers in CI

- Currently we have a set of VMs dedicated to NICA:
  - 2 VMs to execute tests
  - And 2 VMs to build and publish to CVMFS
- Each VM runs certain OS (SL 6, CentOS 7 and Ubuntu 18.04)
- **Drawbacks** of the current setup:
  - Need to support different OSes and keep environment up-to-date
  - Inability to use Shared runners, which require jobs to run in Docker
- **Pros?** Mainly execution time – no need to load containers first
- **Why Docker:**
  - Dedicated VMs can have same setup
  - Shared runners can be used along with the dedicated ones

Containers in CI

- Image options:
  - **User** images: have all the software installed, but are also big (~1.5 GB)
  - **Standard OS** images (centos:7, ubuntu:18.04, etc): 30-100 MB
- Bind-mount CVMFS to have the FAIRsoft inside standard containers (**Done** on shared runners)
- There should be no issues with tests
- Publishing to CVMFS has some issues:
  - To preserve paths we need to build to /cvmfs which in standard images is bind-mounted in **read-only** mode
  - **User images** can be used to build and publish in CVMFS
  - CI containers are not network accessible: we need to change the **pull** strategy to **push** strategy to deliver builds to CVMFS

Jupyter Notebooks

- Web-based interactive development environment
- Has support for a large list of programming languages, including ROOT
- Can be run as individual servers or as a multi-user environment via JupyterHub
- Test setup jupyter.jinr.ru:
  - JINR SSO account for web access
  - JINR Kerberos authentication available via command-line
  - Has CVMFS mounted
  - Has EOS mounted (that’s why Kerberos is needed)
  - A set of “Core Stacks” containers available to play with

FAIRsoft in Jupyter

- A Jupyter container can be easily added to the jupyter.jinr.ru
- NICA environment can be initialized from CVMFS, but a ROOT kernel needs to be additionally added to the container
- FAIRsoft needs to be build with python bindings (probably)
- I did some tests – at least ROOT kernel from FAIRsoft is working

```
In [20]: .x /root/bmnroot/macro/run/run_sim_bmn.C("/root/srcsim.root","/root/bmnsim.root")
   Processed MC points  : 1
 BmnCSCDigitizer::Exec() finished

-I- BmnFD: 6 points registered in this event.
-I- BmnMwpc: 0 points registered in this event.
-I- BmnBd: 13 points registered in this event.
-I- BmnSilicon: 29 points registered in this event.
-I- CbmSts: 36 points registered in this event.
-I- BmnCSC: 1 points registered in this event.
-I- BmnTOF1: 1 points registered in this event.
-I- BmnDch: 58 points registered in this event.
-I- BmnTOF: 3 points registered in this event.
-I- BmnEcal: 178 points registered in this event.
-I- BmnZdc: 0 points registered in this event.
Work time of the Silicon digitizer: 0.6400
Work time of the GEM digitizer: 2.930
Work time of the CSC digitizer: 0.06000
RealTime=7.363219 seconds, CpuTime=6.720000 seconds
Macro finished successfully.
```

ROOT in Jupyter Example

Home Page - Select or create | Untitled2 - Jupyter Notebook

10.93.221.198:8888/notebooks/Untitled2.ipynb

jupyter Untitled2 Last Checkpoint: Вчера в 19:11 (autosaved) Logout

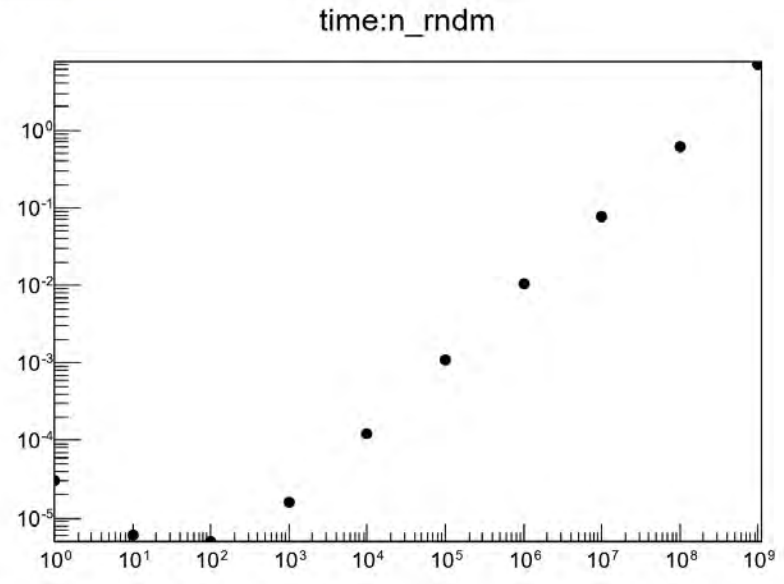
File Edit View Insert Cell Kernel Widgets Help Trusted ROOT C++

Run Code

```
1 :             0.0309944 ms.
10 :            0.00596046 ms.
100 :           0.00500679 ms.
1000 :          0.015974 ms.
10000 :         0.121117 ms.
100000 :        1.08004 ms.
1000000 :       10.3478 ms.
10000000 :      76.658 ms.
100000000 :     613.551 ms.
1000000000 :    7039.4 ms.
```

In [10]:
```
TCanvas c("c");
```

In [11]:
```
%jsroot on
c.cd();
nt->SetMarkerStyle(20);
nt->Draw("time:n_rndm>>h1");
c.SetLogx(true);
c.SetLogy(true);
c.Draw();
```



Conclusions

- User Docker containers wait for its **testers**
- Almost everything technically is ready for migrating CI pipeline to containers
- Dedicated runners are shared with MPD and SPD – migration needs to be done simultaneously
- If there is any interest in using Jupyter – let me know
- Release docker containers (anybody needs them?)

Thanks!