Introduction to SWAN

Diogo C., Prasanth K., Enric T.
On behalf of the SWAN team

https://cern.ch/swan

Feb 2nd, 2021
CERN Academic Training
2 Displaying graphics

We can now draw the histogram. We will at first create a `canvas`, the entity which in ROOT holds graphics primitives. Note that thanks to `pyroot`, this is not a static plot but an interactive visualisation. Try to play with it and save it as image when you are satisfied!

```python
In [5]:
c = ROOT.TCanvas()
h.Draw()
c.DrawLine()
```

We’ll try now to beautify the plot a bit, for example filling the histogram with a colour and setting a grid on the canvas.

```python
In [8]:
h.SetFillColor(ROOT.kBlue+1)
c.SetGrid()
h.Draw()
c.DrawLine()
```

```
My Histo
<table>
<thead>
<tr>
<th></th>
<th>Entries</th>
<th>Mean</th>
<th>Std Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1000</td>
<td>0.02680</td>
<td>1.038</td>
</tr>
</tbody>
</table>
```

```
My Histo
<table>
<thead>
<tr>
<th></th>
<th>Entries</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1000</td>
<td>0.02680</td>
</tr>
</tbody>
</table>
```
Integrating (CERN) services

UI/Core
Software
Analysis platforms
Storage
Compute
Infrastructure
The SWAN Team

> 8 people from IT-ST, IT-DB and EP-SFT (all part time)

- Enrico Bocchi
- Riccardo Castellotti
- Diogo Castro
- Prasanth Kothuri
- Jakub Moscicki
- Krishnan Raghavan
- Enric Tejedor
- Omar Zapata
General Architecture
Architecture (soon)

swan.cern.ch

Web portal

Container Scheduler

User 1

User 2

... User n

EOS/CERNBox
(Data/User files)

CERN Resources

CVMFS
(Software)

User 1

User 2

... User n

EOS/CERNBox
(Data/User files)

CERN Resources

CVMFS
(Software)

User 1

User 2

... User n

EOS/CERNBox
(Data/User files)

CERN Resources

CVMFS
(Software)
How to use it?
Have a CERN account

Have a CERNBox space
(Open the service at least once!)

Optional:
Request access to Spark
(Service Now Request)

Personalise the environment? startup script!

Spark: do you want to offload computations to a CERN cluster? Which one?

Platform: what system/compiler?

How much memory? And cores?

Automatically start with the same configuration everytime?
LCG releases:
• Managed by Librarians in EP
• Python 3 or Python Recommended releases are cached
• Bleeding edge might be unstable

http://lcginfo.cern.ch/
CERNBox is your home
Projects

My Projects

<table>
<thead>
<tr>
<th>NAME</th>
<th>SIZE</th>
<th>STATUS</th>
<th>MODIFIED</th>
</tr>
</thead>
<tbody>
<tr>
<td>CERN-CSN-2019</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cinemas Study</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cuno Python</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EOS ops</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New project</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Polish Students Project</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spark</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SWAN User Analysis</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test Examples</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Testing Notebooks</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text reconverted</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work</td>
<td>9 days ago</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SWAN © Copyright CERN 2015-2019. All rights reserved.
Home | Community | Support | Report a bug
Projects

1. Create empty

2. Download from our Gallery, GitHub, CERN GitLab, CERNBox or ROOT website

3. https://cern.ch/swanserver/cgi-bin/go?projurl=<path to your repo>
Open notebooks and other files from the readme!
CERNBox tab

![CERNBox tab](image)

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Status</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documents</td>
<td></td>
<td>a month ago</td>
<td></td>
</tr>
<tr>
<td>Music</td>
<td></td>
<td>a month ago</td>
<td></td>
</tr>
<tr>
<td>Pictures</td>
<td></td>
<td>a month ago</td>
<td></td>
</tr>
<tr>
<td>SHARED</td>
<td></td>
<td>a day ago</td>
<td></td>
</tr>
<tr>
<td>SWAN_projects</td>
<td></td>
<td>21 days ago</td>
<td></td>
</tr>
<tr>
<td>Videos</td>
<td></td>
<td>a month ago</td>
<td></td>
</tr>
<tr>
<td>My file.txt</td>
<td>0 B</td>
<td>2 years ago</td>
<td></td>
</tr>
<tr>
<td>New document.docx</td>
<td>10.9 kb</td>
<td>5 months ago</td>
<td></td>
</tr>
<tr>
<td>New presentation.pptx</td>
<td>29.4 kb</td>
<td>5 months ago</td>
<td></td>
</tr>
</tbody>
</table>
Creating notebooks

Click + and choose language
2 Displaying graphics

We can now draw the histogram. We will at first create a canvas, the entity which in ROOT holds graphics primitives. Note that thanks to JROOT, this is not a static plot but an interactive visualisation. Try to play with it and save it as image when you are satisfied.

```python
In [5]: c = ROOT.TCanvas()
   h.Draw()
   c.Draw()
```

Trusted = execute code on user behalf, including showing the outputs of the cells*

(*) if you execute the cell manually, it will always show it
2 Displaying graphics

We can now draw the histogram. We will first create a canvas, the entity which in ROOT holds graphics primitives. Note that thanks to JSROOT, this is not a static plot but an interactive visualization. Try to play with it and save it as image when you are satisfied.

```
In [5]: c = TCanvas()
h.Draw()
c.Draw()
```

We’ll try now to beautify the plot a bit, for example filling the histogram with a colour and setting a grid on the canvas.

```
In [6]: h.SetFillColor(ROOT.kBrown-6)
c.SetGrid()  
h.Draw()  
c.Draw()
```
Notebook cells

Click to change the current cell type

- Code
- Markdown
- Raw NBConvert
- Heading
Saving notebooks

- Autosave
- Force save + create version (checkpoint)
Rename Notebook

Enter a new notebook name:

**Rename Notebook**

Click the name and enter the new name

Or change in the file view
Keyboard shortcuts

Check the shortcuts in your OS

Shift + Enter to run a cell
Install Python modules:

bash-4.2$ pip install --user mypackage

Startup script

export PYTHONPATH=$CERNBOX_HOME/.local/lib/python3.X/site-packages:$PYTHONPATH

Also possible for R! Check Help.

http://lcginfo.cern.ch/
Search like from CERNBox ("a:" for secondary account)
(Egroups support released later this year)
Sharing

Hover to remove individual users

Remove the share for everyone
### Projects shared with me

<table>
<thead>
<tr>
<th>NAME</th>
<th>SIZE</th>
<th>SHARED BY</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLASIM/VBR</td>
<td>11.0 GB</td>
<td>phoebus</td>
<td>2 years ago</td>
</tr>
<tr>
<td>cambootlog</td>
<td>66.1 kB</td>
<td>gonzalhu</td>
<td>7 months ago</td>
</tr>
<tr>
<td>CondorGisGCttest</td>
<td>33.5 kB</td>
<td>marcelkie</td>
<td>2 years ago</td>
</tr>
<tr>
<td>NewProj</td>
<td>1.21 kB</td>
<td>eisjendr</td>
<td>2 years ago</td>
</tr>
<tr>
<td>Spark-DistROOT</td>
<td>236 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>Super Red Analysis with TOTEM data</td>
<td>3.29 kB</td>
<td>jaykstensen</td>
<td>2 years ago</td>
</tr>
<tr>
<td>SWAN-Spark_NXCALC_Example</td>
<td>185 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TDF Tutorial</td>
<td>608 B</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TestShivans3</td>
<td>4.19 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TOTEM TDataFrame</td>
<td>11.3 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
</tbody>
</table>

### Projects shared by me

<table>
<thead>
<tr>
<th>NAME</th>
<th>SIZE</th>
<th>SHARED BY</th>
<th>DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATLASIM/VBR</td>
<td>11.0 GB</td>
<td>phoebus</td>
<td>2 years ago</td>
</tr>
<tr>
<td>cambootlog</td>
<td>66.1 kB</td>
<td>gonzalhu</td>
<td>7 months ago</td>
</tr>
<tr>
<td>CondorGisGCttest</td>
<td>33.5 kB</td>
<td>marcelkie</td>
<td>2 years ago</td>
</tr>
<tr>
<td>NewProj</td>
<td>1.21 kB</td>
<td>eisjendr</td>
<td>2 years ago</td>
</tr>
<tr>
<td>Spark-DistROOT</td>
<td>236 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>Super Red Analysis with TOTEM data</td>
<td>3.29 kB</td>
<td>jaykstensen</td>
<td>2 years ago</td>
</tr>
<tr>
<td>SWAN-Spark_NXCALC_Example</td>
<td>185 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TDF Tutorial</td>
<td>608 B</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TestShivans3</td>
<td>4.19 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
<tr>
<td>TOTEM TDataFrame</td>
<td>11.3 kB</td>
<td>eisjendr</td>
<td>3 years ago</td>
</tr>
</tbody>
</table>
Share tab

Clone the project

Open the files in Read Only mode
The top bar

- **Terminal.** Same environment as the notebooks (LCG)
- **Running notebooks and terminals**
- **Enable/Disable Extensions**
- **Running Processes**
- **Nbextensions configuration**
- **Change configuration**
- **SWAN Gallery**
- **Help**
- **Logout.** Doesn’t kill the session.
  Will stay alive for 6 hours after last interaction via the UI.
- **Kills the session.** Goes to initial configuration menu.
Useful Extensions

- **RISE**
  - Create presentations and execute your cells live

- **Exercise2**
  - Create exercises with hidden solutions

- **2to3 Converter**
  - Convert cells from Py2 to 3

- **ExecuteTime**
  - Log how much time it takes to run a cell

- **Table of Contents**
  - Easily navigate long documents

- **Scratchpad**
  - Run code against current kernel without modifying the notebook

### 3.1 smart exec and fragment

- in general the last cell in a fragment will behave like the end of a slide
- however if what comes after that fragment is visible already
  - like when we show a fragment and move backwards
  - then we allow smart-exec to go to the next cell

```
In [12]: # last cell in fragment
   print("I could go on")
I could go on

In [13]: # move back up once this is displayed
   print("once the next fragment is shown")
   once the next fragment is shown
```

should be red if the css-loading feature works fine

should be blue if the css-loading feature works fine
> Next generation UI
  - Version 3 recently released

> Ongoing work: integration of current extensions
  - SWAN Projects
  - CERNBox sharing integration
  - Spark Connector and Monitor
  - …
Apache Spark and GPU
What is Apache Spark?

> Apache Spark
- An open-source parallel processing framework with expressive development APIs (in multiple languages) that allows for sophisticated analytics, real-time streaming and machine learning on large datasets

> Spark ecosystem
<table>
<thead>
<tr>
<th>Cluster Name</th>
<th>Configuration</th>
<th>Primary Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>analytix</td>
<td>38 nodes (Cores – 1612, Mem – 17.24TB, Storage – 11.65 PB)</td>
<td>General Purpose</td>
</tr>
<tr>
<td>nxcals</td>
<td>30 nodes (Cores – 960, Mem – 11.82TB, Storage – 7.5 PB)</td>
<td>Accelerator logging (NXCALS) project dedicated cluster</td>
</tr>
<tr>
<td>Cloud containers</td>
<td>12 nodes (Cores 224, Mem – 1.6 TB, Storage – EOS)</td>
<td>General Purpose Compute ONLY</td>
</tr>
</tbody>
</table>
Spark Connector – handling the spark configuration complexity

- User is presented with Spark Session (Spark) and Spark Context (sc)
- Ability to bundle configurations specific to user communities
- Ability to specify additional configuration
Spark Monitor

Spark Monitor – Jupyter notebook extension

- For live monitoring of spark jobs spawned from the notebook
- A graph showing number of active tasks & executor cores vs time
- A timeline which shows jobs, stages, and tasks
> **HDFS Browser** – Jupyter notebook extension

- Browsing the Hadoop Distributed File System from notebook
- Useful for selection of the datasets for analysis
Integration of SWAN with Spark clusters

The current setup allows to execute PySpark operations on CERN Hadoop and Spark clusters. This notebook illustrates the use of Spark in SWAN to analyze the monitoring data available on HCFS and plots a heatmap of load across machines in a particular service.

Connect to the cluster

To connect to a cluster, click on the star button on the top and follow the instructions

- The star button only appears if you have selected a SPARK cluster in the configuration
- The star button is active after the notebook kernel is ready

Import necessary spark and python stuff

```python
In [1]: from pyspark.sql.functions import from_unixtime, when, col
from pyspark.sql.types import IntegerType
from pyspark.sql.functions import from_json
import matplotlib
import pandas as pd
```
GPUs for SWAN

> GPUs are available in the pilot instance of SWAN on Kubernetes at https://swan-k8s.cern.ch
  - It will become the default instance at https://swan.cern.ch

> In the pilot instance in CERN cloud, we are offering 5 GPUs (4x Tesla T4 + 1x V100)
  - If there is demand, we will ask for more from CERN cloud

> Software packages from CVMFS
  - The latest release (99 Cuda 10.1 Python3) has Tensorflow 2.3.0 and PyTorch 1.4.0
  - Other releases have older versions Tensorflow 2.1.0 etc

> How are the resources shared?
  - The user gets 1 GPU, 2 cores and 16 GB RAM from the available pool
  - Users are removed after 6 hours of inactivity
ScienceBox

> Packaged deployment of CERN software
  - Includes SWAN, CERNBox, EOS, CVMFS
  - Deployment: kubernetes, docker-compose
  - Created as part of EU funded project UP2U
  - A new version is in the works

> Some successful community installations…
  - AARNet, PSNC, Open Telekom Cloud (Helix Nebula), SURFSARA, Joint Institute for Nuclear Research (JINR)

> Interest from other institutions/future deployments
How to get started?
How to get started?

> A possible way to start using SWAN is by accessing some content (notebooks) that some other user created.

> In order for a user to share content with others, SWAN offers several options:
  - For exploratory work, collaborative editing: use **sharing** feature!
  - For an event (training/course), for attendees to access its content: Open in SWAN
  - To publish something that can be helpful to a broader audience: use **galleries**!
> Galleries of sample notebooks for varied usage of SWAN

- Quick way to be productive
- Also accessible from [https://swan-gallery.web.cern.ch/](https://swan-gallery.web.cern.ch/)
> SWAN users can request the addition of their own gallery to the SWAN galleries site

> How to add a new gallery?
  - Create a (github/gitlab) repository with your notebooks
  - ... plus the images you want to show for each one
  - Make a PR to https://github.com/swan-cern/gallery
  - Detailed instructions can be found here
SWAN common use cases
SWAN Users’ Workshop

> Get together with the users of the service
  ▪ Share use cases and knowledge among the community
  ▪ Allow SWAN admins to understand how the service is used
  ▪ Collect wishes for future improvements

> https://indico.cern.ch/event/834069

<table>
<thead>
<tr>
<th>Morning Session</th>
<th>Afternoon Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; SWAN team</td>
<td>&gt; Beams</td>
</tr>
<tr>
<td>▪ Service overview, new features</td>
<td>▪ NXCALS, machine studies</td>
</tr>
<tr>
<td>&gt; Experiments</td>
<td>&gt; Technology</td>
</tr>
<tr>
<td>▪ Analysis, software configuration, ...</td>
<td>▪ Signal monitoring, ML, ...</td>
</tr>
<tr>
<td>&gt; SWAN on premises</td>
<td>&gt; SWAN for education &amp; outreach</td>
</tr>
<tr>
<td>▪ AARNet's CloudStor SWAN</td>
<td>▪ EduSWAN, open data, documentation</td>
</tr>
<tr>
<td></td>
<td>&gt; And much more!</td>
</tr>
<tr>
<td></td>
<td>▪ System log analysis, Gitlab, ...</td>
</tr>
</tbody>
</table>
Physics analysis

> SWAN can be used for interactive physics analysis
  - Usually medium/small size analysis (input data ~ a few GBs)
  - Last steps of analysis (e.g. nanoAOD, ntuples)

> All the data you need
  - CERNBox: user files
  - EOS: project spaces, experiment data

> All the software you need
  - ROOT
  - Python data science libraries: numpy, pandas, matplotlib, seaborn, ...
  - Python ML libraries: tensorflow, keras, pytorch, ...
  - R kernel
  - Plus anything you can find in an LCG release (~ 500 packages)
SWAN usage at COMPASS - Study of the DVCS process at COMPASS

For all the events candidates, extract the DVCS exact topology

\[ \Delta \phi = \phi^\text{cam} - \phi^\text{spec} \]

\[ \Delta p_T = p_T^\text{cam} - p_T^\text{spec} \]

\[ \Delta z_A = z_A^\text{cam} - z_A^\text{real} \text{ and vertex} \]

\[ M^2_{\text{Undet}} = (k+p-k'q-p)^2 \]

Pros & cons of SWAN

- Well organised analysis in notebooks using sections (use of markdown, latex, etc)
- Flexible enough to make systematic studies: use SWAN to adjust some cuts
- Drawback of interactivity: can be slow with too many events

Spark helps!
SWAN is not restricted to physics analysis only; it is general enough to accommodate other scenarios.

Data: it does not have to be on EOS!
- E.g. it is possible to access databases from SWAN
- or to read data from HDFS

Software: all the data analysis/processing/visualization libraries available in the LCG releases can also be used for non-physics analysis.
**pytimber: simple use**

```python
In [1]:
import pytimber
db=pytimber.LoggingDB()
atlas='ATLAS:LUMI TOT INST'
data=db.get([atlas], '2018-09-10 4:00:00', '2018-09-10 18:00:00')

In [2]:
%matplotlib notebook
import matplotlib.pyplot as plt

t,v=data[atlas]
plt.plot(t,v*1e30*98*1e-27)
plt.ylabel("Collision per second")
plt.xlabel("x axis")
pytimber.set xaxis date()
```
Interactive visualization & monitoring

> Notebooks are a good interface for displaying interactive graphics
  - matplotlib, seaborn

> It can be used as a complement to monitoring
  - For debugging, exploration, posterior analysis
  - Does not replace live monitoring tools (e.g. Grafana)
Use cases for SWAN

Two types of debugging:

1. live! There’s an issue NOW. Probably just need to load in a small amount of recent “bad” data, maybe some previous “good” data and investigate as many relevant variables as possible

   Here, the interactive visualization is the best part – can see ALL the possibilities in one place, with the exact code that made them – discuss with others, etc.

   https://atlas.cern/updatessymphory-atlas

2. post-run: Something looked a little off last run. Did it look off the run before? When did it start? What else changed? Could need to load a large amount of data, overlay many runs…

   Only need to load data once – and can then play around with many plots, without having to wait for the data to load again

And then another use, for post-mortem of new changes: want to document effects of updates on the system and verify that changes had the expected outcome

   e.g. new trigger had the expected rate and the CPU changed accordingly

---

SWAN in ATLAS (with a focus on operations) – Heather Russell (EP)
Entry point to computational resources

> Notebooks are meant for interactive computing
  - Usually lightweight computations

> A SWAN session gives access to limited (shared) resources
  - CPU, memory
  - Not intended for heavy, long-running computations

> Heavy computations should be **offloaded**
  - SWAN can be used as the entry point to access computational resources
  - Examples: the Grid, Spark clusters, GPUs, HPC clusters (soon)
Data Analysis Procedure in ALICE with SWAN

User’s Side

- **Analysis task (code) development:**
  - Develop new analysis code (C++) for one’s physics motivation.
  - Can use pre-developed framework with different configuration.
  - Need to merge into Official Git Repository (AliPhysics)

- **Compute (Run) on the grid:**
  - Run the analysis code on the grid (AliEn) for experimental dataset.
  - Submit to AliEn directly (small quota) or using central LEGO train (big quota)

- **Post-process for the plotting:**
  - Download merged output from AliEn
  - Run post-processing macro to generate final plots

ALICE Data Analysis without wired PC feat. SWAN – Bong-Hwi Lim (EP)
Education is one of the use cases where SWAN shines the most

- Notebooks as teaching material
- Code, text with explanations and interactive execution, all in the same place!

Many courses already given using SWAN as a platform

- ROOT trainings, ML, CERN School of Computing, …
- Repositories with teaching material easily opened in SWAN
- Files are cloned into your CERNBox, software provided by CVMFS

Open data access and analysis also a perfect match

- SWAN’s central instance targeted for CERN users
- But there is ScienceBox!
Example: $H \rightarrow \tau\tau$ analysis with CMS Open Data

- Providing ready-to-run examples for more realistic analyses is currently very challenging
  - HEP is inherently large scale
  - Binder-like services and internet connection of individuals provide only limited bandwidth

- Scale of an example minimal $H \rightarrow \tau\tau$ analysis
  - "University-level" example
  - 9 derived datasets, in total about 60 GB
  - Stored in ROOT files / NanoAOD-like format
  - Computation mainly IO bound
    (skimming and making histograms)

- Runs smoothly on SWAN due to the existing software stacks (CVMFS and LCG stacks) and excellent connectivity to the CERN eco-system (EOS)
Getting help & contact
> **SWAN Community**
- [https://cern.ch/swan-community](https://cern.ch/swan-community)
- Find solution to the commonly encountered issues / questions on the usage of Jupyter notebooks, LCG releases, storage and Spark
- Report improvements / new features to the service
- E.g: How to install custom user packages

> **Service Now**
- Report issues to the service
- E.g: Unable to start a session

> **Help** on various features of the tool

---

**Help**

1. **Introduction**
   - What is SWAN
   - Jupyter notebooks
   - Cloud storage: CERNBox and EOS
   - Software: CVMFS

2. **Create and manage a SWAN session**
   - Select a configuration
   - Set a configuration as default
   - Switch to a new configuration
   - Terminate a session

3. **Working with SWAN**
   - Create a Project
   - Create a Notebook
   - Create a Folder
   - Open a Terminal
Where to find us

> Contacts
  - swan-admins@cern.ch
  - http://cern.ch/swan

> Repository
  - https://github.com/swan-cern

> ScienceBox
  - https://cern.ch/sciencebox
Introduction to SWAN

Thank you

Diogo Castro (diogo.castro@cern.ch)
Enric Tejedor (etejedor@cern.ch)
Prasanth Kothuri (prasanth.kothuri@cern.ch)