



Preliminary draft 08:59 29 August 2019

29 August 2019

philip@leindecker.at

Summer Student Report 2019

Philip Leindecker

CERN, CH-1211 Geneva, Switzerland

Keywords: ROOT, JupyterLab, JS Graphics, ROOT GUI, TBrowser, FitPanel

Summary

This work was done during the CERN Summer Student Programme 2019 in the ROOT EP-SFT department, supervised by Enric Tejedor Saavedra and Bertrand Bellenot. The document is split up into two parts. The first part deals with displaying JSROOT Graphics in JupyterLab Notebooks. The second part covers integration of ROOT JS GUI in JupyterLab.

Contents

1	Display JSROOT Graphics in JupyterLab Notebooks	2
1.1	Introduction	2
1.2	Jupyter Interface - Evolution	2
1.3	Bugfix - requirejs	4
1.4	Bugfix - jsUID generation	4
1.5	Result	5
2	Integration of ROOT JS GUI in JupyterLab	5
2.1	Introduction	5
2.2	TBrowser	5
2.3	JupyterLab - Extensions	6
2.4	FitPanel	7
3	Acknowledgements	10

1 Display JSROOT Graphics in JupyterLab Notebooks

1.1 Introduction

The goal was to enable JSROOT graphics in JupyterLab, which requires the `jsroot` on command. These graphics should be displayed in the same way as in the old Jupyter Notebooks with all their interactive features.

1.2 Jupyter Interface - Evolution

Jupyter - “*JupyterLab will eventually replace the classic Jupyter Notebook.*” [1]

JupyterLab is the newer Version of Jupyter Notebooks, with Version 1.0 available since June 2019. The interface evolved from a single-view layout (Fig. 1, 2) to a tab-based all-in-one layout (Fig. 4).

JupyterLab enables you to work with documents and activities such as Jupyter notebooks, text editors, terminals, and custom components in a flexible, integrated, and extensible manner. You can arrange multiple documents and activities side by side in the work area using tabs and splitters[1].

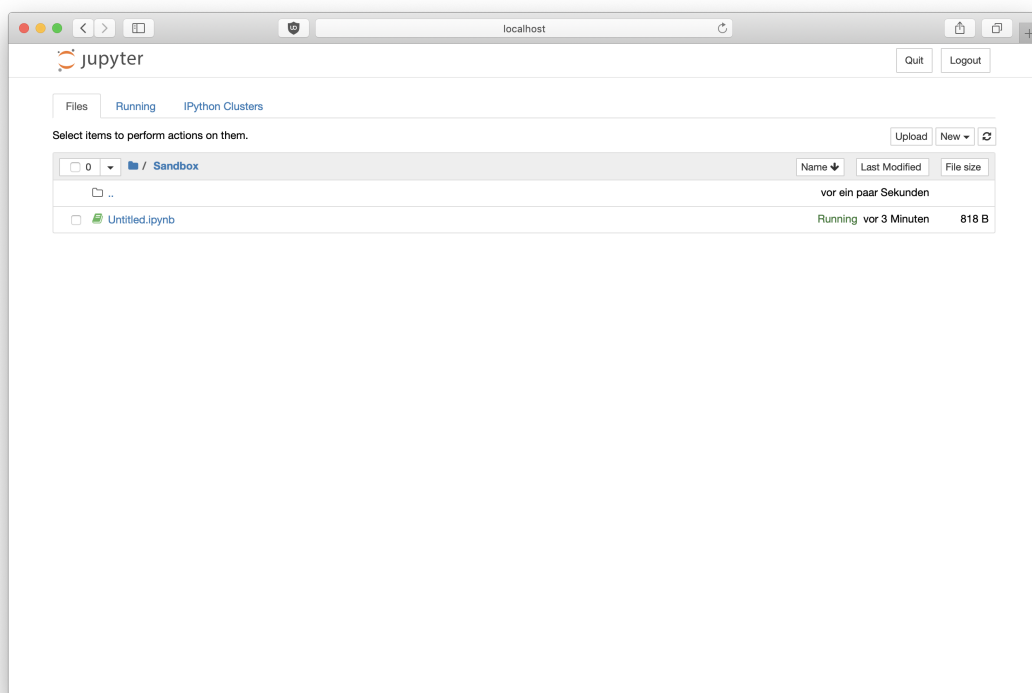


Figure 1: Jupyter Notebook -Filebrowser

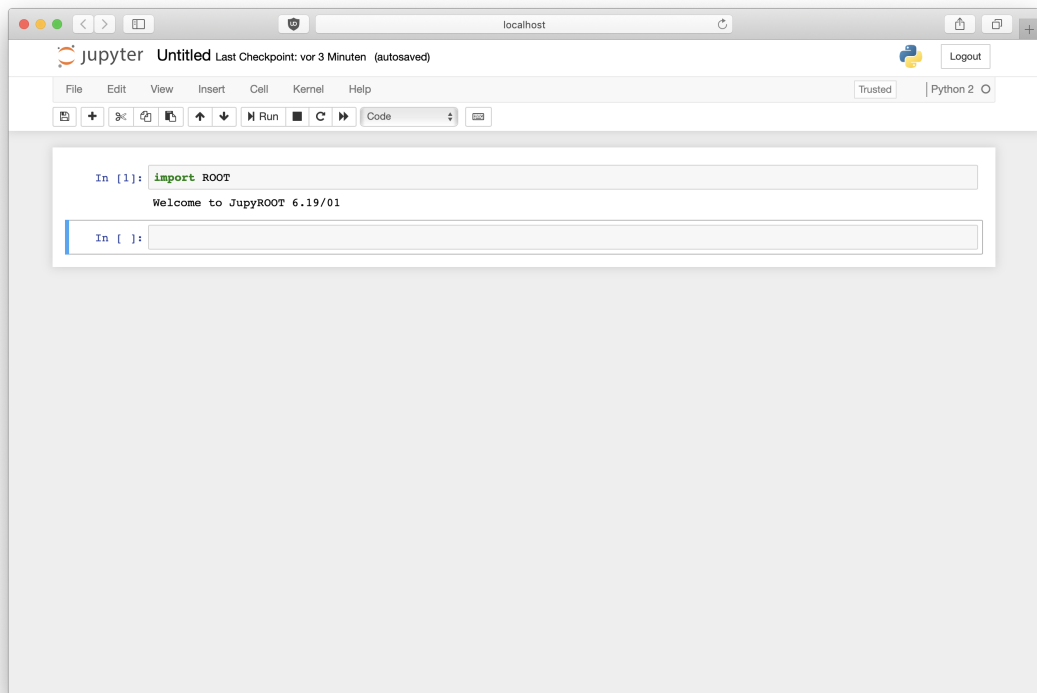


Figure 2: Jupyter Notebook - Python Notebook

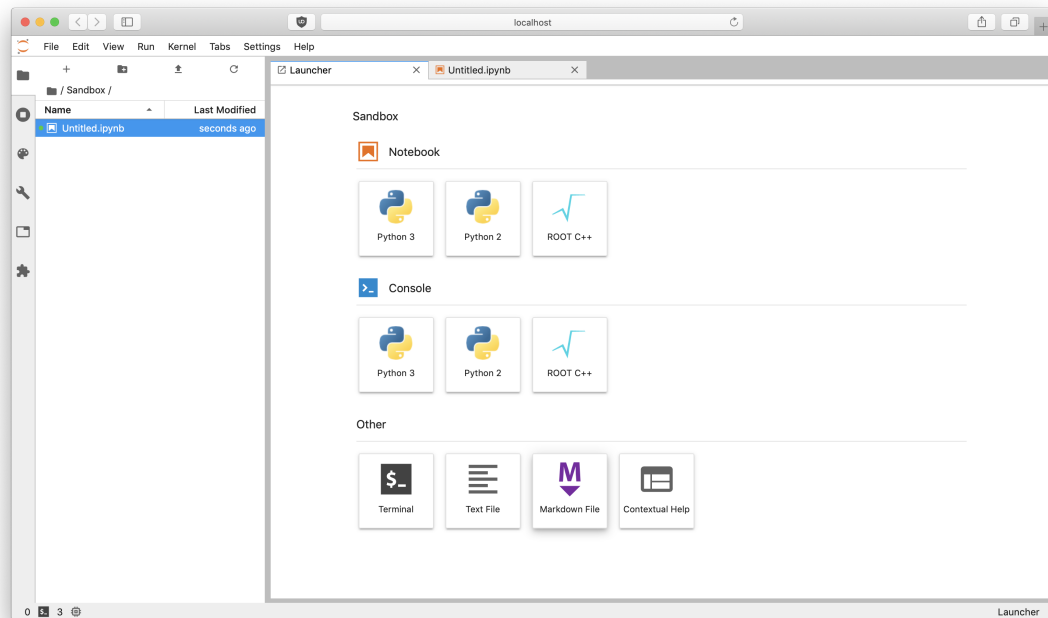


Figure 3: JupyterLab - Interface

1.3 Bugfix - requirejs

The first issue was, that the browser could not find requirejs. After including requirejs in utils.py via the following line only partially solved the issue.

```
1 <script src="/static/components/requirejs/require.js" type="text/
  javascript" charset="utf-8"></script>
```

1.4 Bugfix - jsUID generation

Although the browser could find requirejs now, the interactive graphics were very unstable and most of the time still were not displayed and required further investigation.

One main difference between the old Jupyter Notebook and the new JupyterLab is the tab-based layout in JupyterLab, which allows multiple Notebooks to be open in one single JupyterLab session.

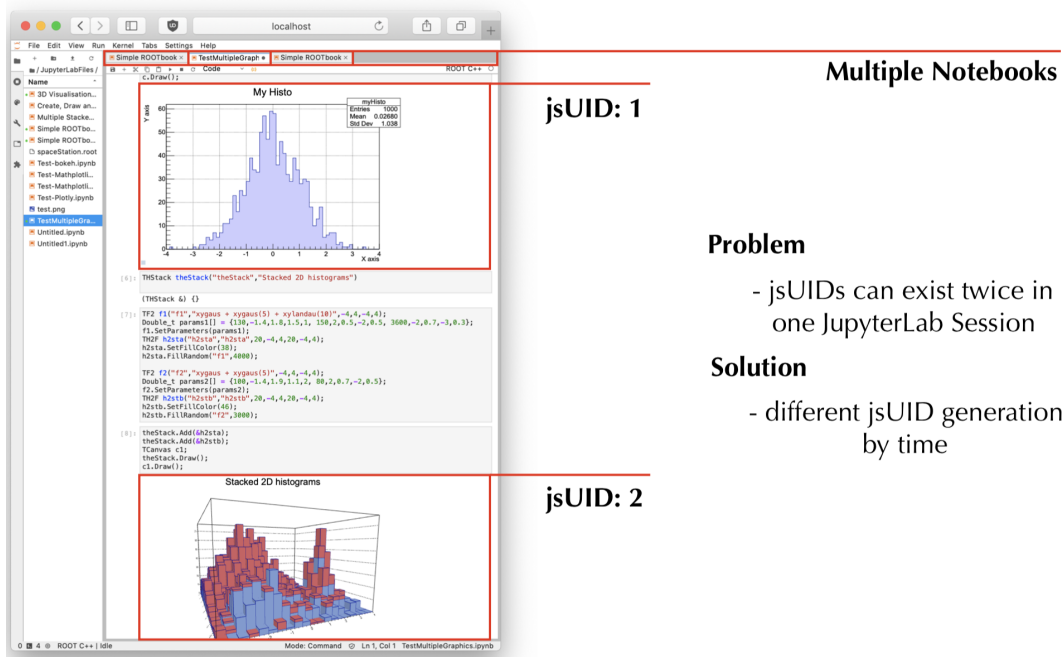


Figure 4: JupyterLab - UUIDs

Every DIV containing a JavaScript snippet (interactive graphic) must be unique in the notebook. This methods provides a unique identifier. With the introduction of JupyterLab, multiple Notebooks can exist simultaneously on the same HTML page. In order to ensure a unique identifier with the UID throughout all open Notebooks the UID is generated as a timestamp. This finally fully solved the issue of the interactive graphics in JupyterLab.

```
1 def _getUID(self):
2     return int(round(time.time() * 1000))
```

1.5 Result

The updates mentioned above are available in github under the pull request:
<https://github.com/root-project/root/pull/4271>.

2 Integration of ROOT JS GUI in JupyterLab

2.1 Introduction

In this second part the goal was to integrate ROOT JS GUI such as the TBrowser (new: RBrowser) or the FitPanel as native elements in JupyterLab.

2.2 TBrowser

A TBrowser (Fig. 5) allows the user to browse all ROOT files.

It shows in a list on the left side of the window all browsable ROOT classes. Selecting one of the classes displays, in the icon-box on the right side, all objects in the class. Selecting one of the objects in the icon-box, will place all browsable objects in a new list and draws the contents of the selected class in the icon-box[3].

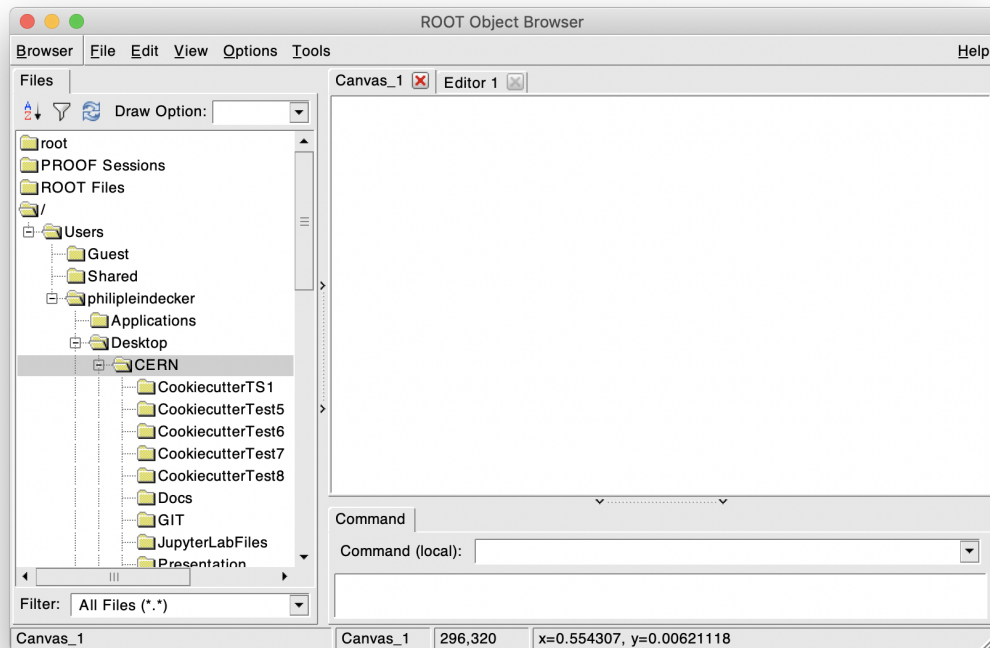


Figure 5: TBrowser - old version

The updated version of the TBrowser is called RBrowser (Fig. 6) and is now implemented in Javascript.

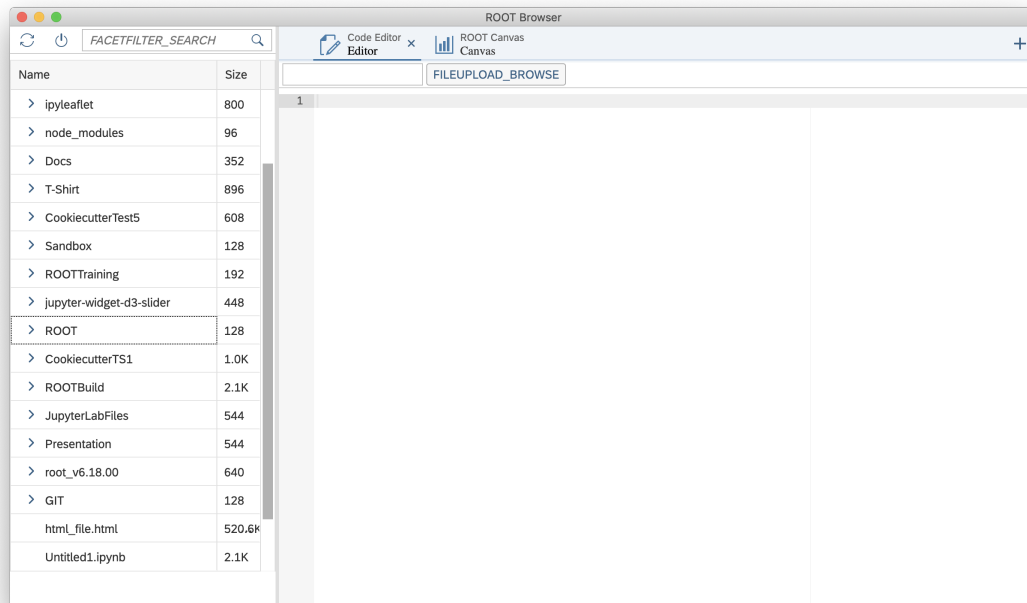


Figure 6: TBrowser - new version

2.3 JupyterLab - Extensions

JupyterLab - *“In fact, the whole of JupyterLab itself is **simply a collection of extensions** that are no more powerful or privileged than any custom extension.”* [2]

In order to integrate the TBrowser in JupyterLab a JupyterLab Extension was implemented, which allowed to open the TBrowser natively in a new Tab in JupyterLab.

JupyterLab extensions can customize or enhance any part of JupyterLab. They can provide new themes, file viewers and editors[2] ... and many more.

Using the cookiecutter template available on github (<https://github.com/jupyterlab/extension-cookiecutter-ts>) allowed a quick basic setup of the extension for the TBrowser. The existing javascript UI of the TBrowser is then displayed in an iFrame in the extension (Fig. 7).

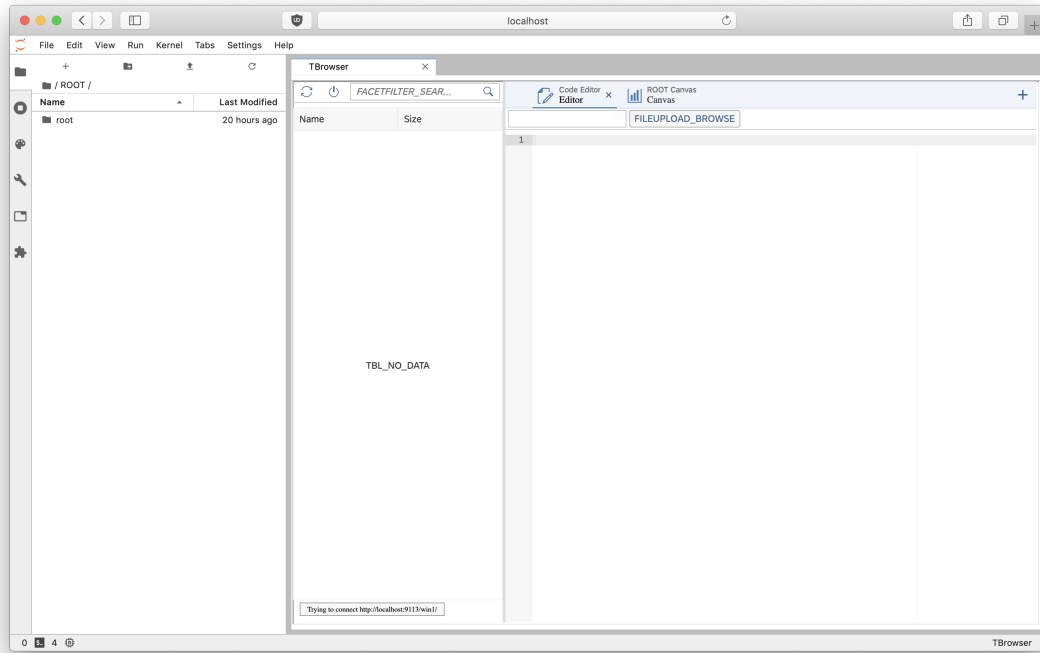


Figure 7: TBrowser in JupyterLab

The list of files in the left view of the TBrowser is not showing in JupyterLab and still has to be fixed. Additionally, while testing the TBrowser remotely, it also did not show the list of files. This problem is currently topic of investigation.

2.4 FitPanel

The FitPanel allows the user to interactively operate on Graphs such as Histograms. In contrast to the old version of the FitPanel (Fig. 8) which opens in an external window, the new FitPanel (Fig. 9) should be again displayed in another Tab in JupyterLab.

The main difference between the FitPanel and the TBrowser before is the communication with the Kernel. The FitPanel has to communicate directly with the JupyterLab Kernel for the Notebook in which the Histogram is being displayed. For such a kind of communication the widgets are designed for. The Widgets are a special kind of extension and rely on the ipywidgets library (Fig. 10).

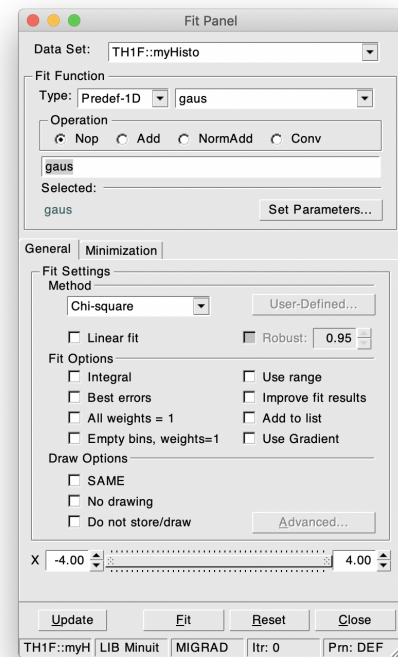
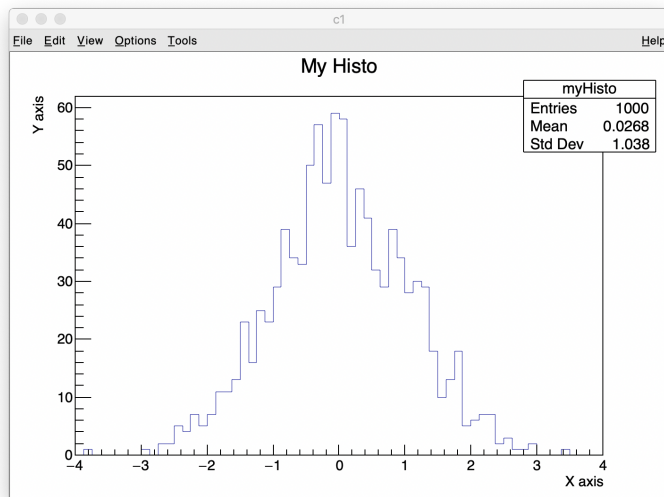


Figure 8: FitPanel - old version

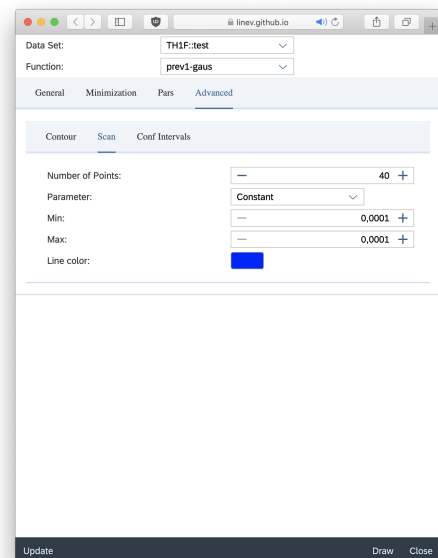
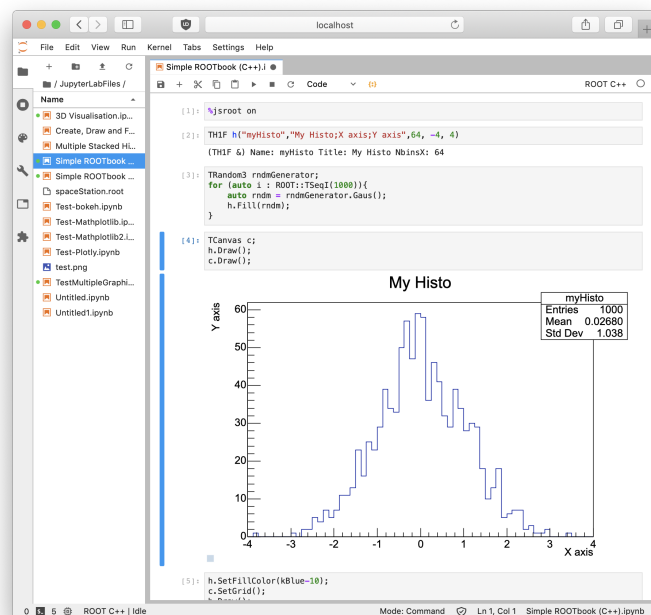


Figure 9: FitPanel - new version

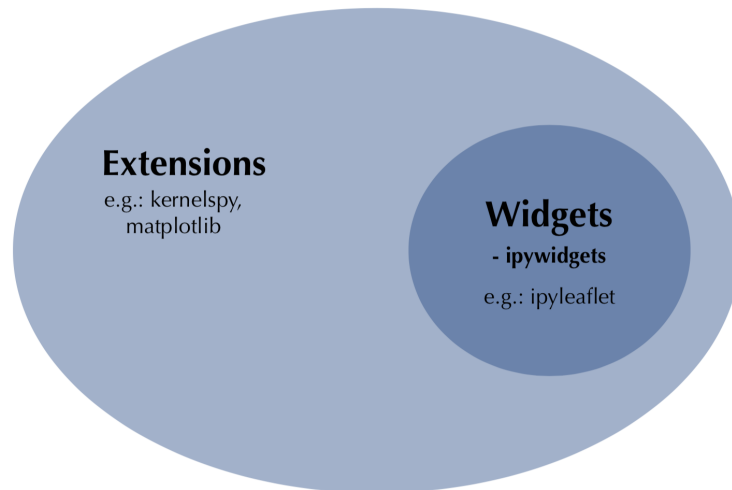


Figure 10: JupyterLab Extension Overview

These widgets are a more high level extension and allow a synced communication between the Frontend (Javascript) and the Python Kernel with a MVC architecture.

The proposed communication for the FitPanel is displayed in Fig. 11. It consists of the existing HTML/Javascript Structure of the Fitpanel and a proposed communication layer (JS ROOT) between this structure and the ipywidgets.

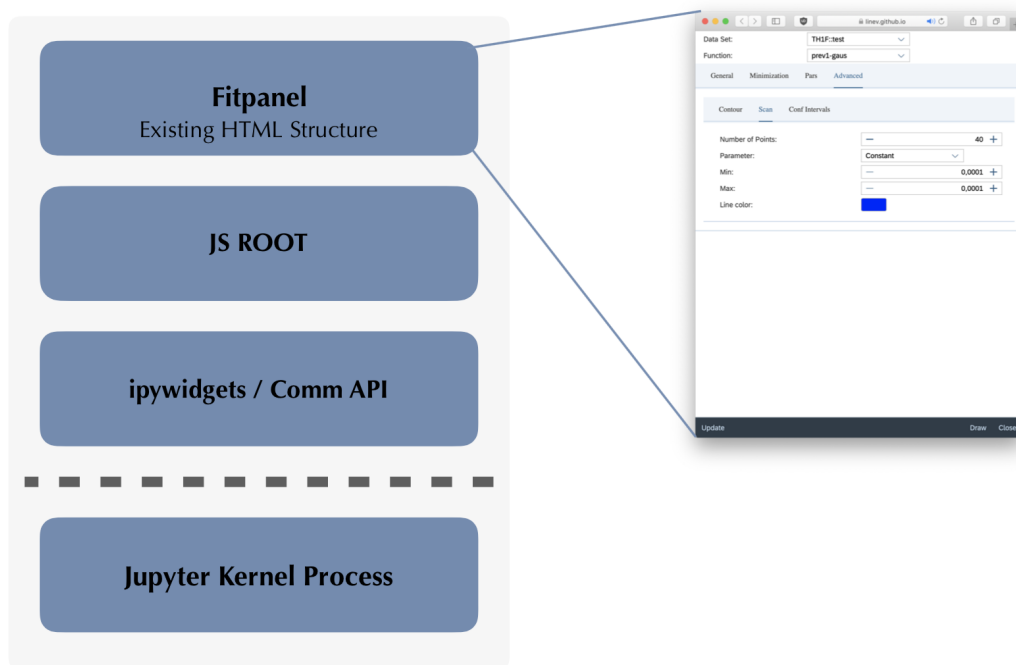


Figure 11: FitPanel in JupyterLab - possible communication

3 Acknowledgements

I would like to thank my supervisors Enric Tejedor Saavedra and Bertrand Bellenot for their support and help during the whole programme. I also want to thank all the members of the ROOT Team, together with the Summer Student Team and the other Summer Students who made my experience here at CERN really unforgettable and I am grateful to had the opportunity to be part of CERN.

References

- [1] JupyterLab. https://jupyterlab.readthedocs.io/en/stable/getting_started/overview.html.
- [2] JupyterLab. <https://jupyterlab.readthedocs.io/en/stable/user/extensions.html>.
- [3] ROOT. <https://root.cern.ch/doc/master/classTBrowser.html>.