JavaScript GUI

JSROOT reloaded

Basic ideas

- The GUI (widgets) is on the client side (HTML, JavaScript, …).
- The GUI action triggers either a JavaScript function call (on the client side), or pass a formatted C++ method to be called on the server side.
- On the server side (C++) the ROOT THttpServer class is expected to handle the events.
- Users’ GUI could be created using any web GUI toolkit

JavaScript GUI libraries

- Benefits:
  - Large user communities
- Strategy:
  - Select one and use it internally
  - Provide basic support for any GUI library
  - Dedicated add-ons for JSROOT/THttpServer
  - Avoid strong bounding with single one
  - Give complete freedom to the users to use their favourite one

Complex GUI

We need to build complex GUI, like editors and Fit Panel, from the C++ objects (classes) information and define the method and arguments to be processed by the server side from its class members or Getters and Setters. This is very similar to what is currently done in C++.

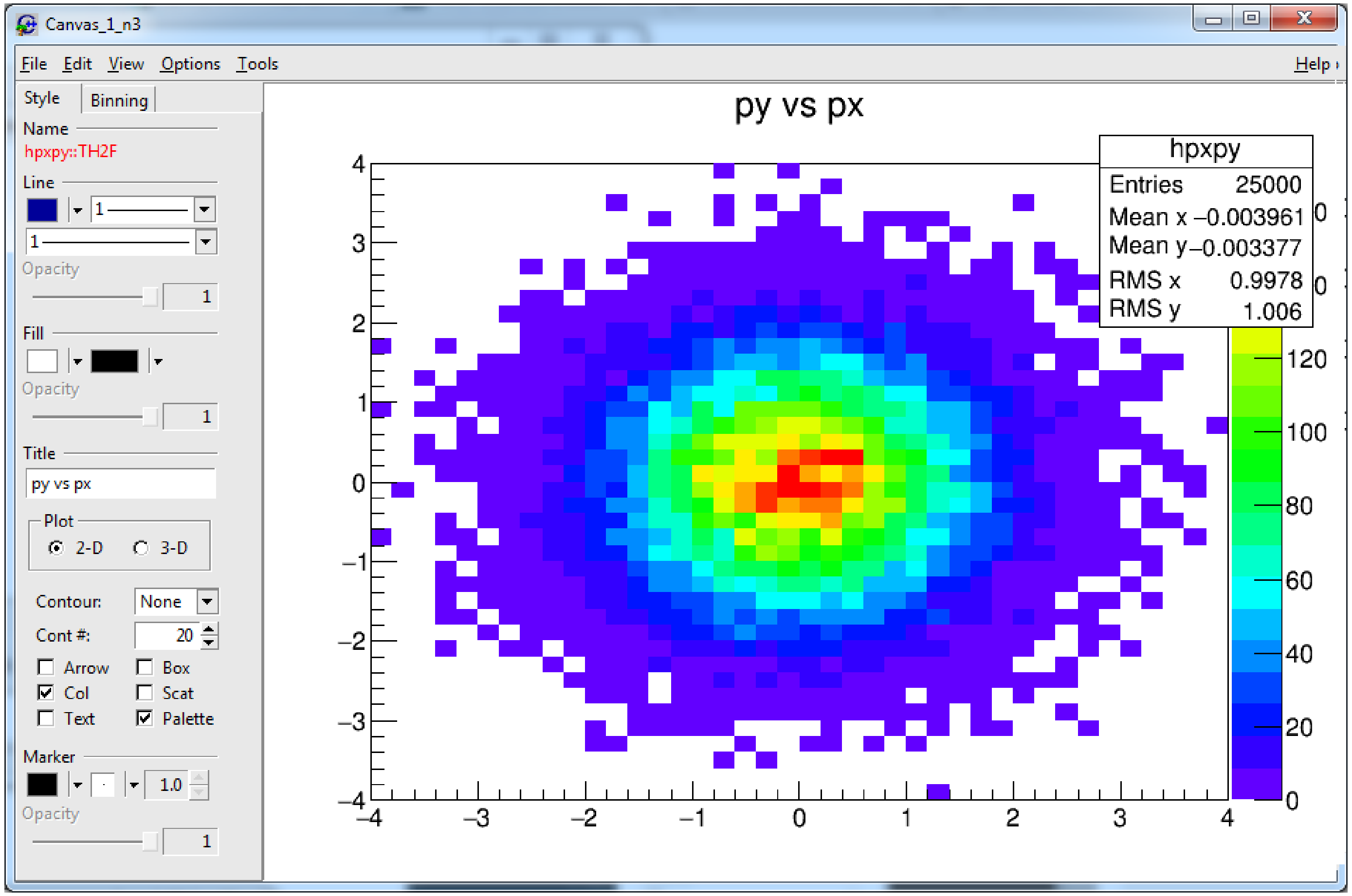
Graphical Editors

Objects like histograms, graphs, axis etc.. displayed in the canvas need local (on the JavaScript side) editors to change their attributes on the client side. Then a validation of the changes will trigger the update to the server side

Possible options for the placement of the editors:

- On the left space (as in the browser)
- Floating, like this example
- Overlay (something like this)

Left side (current layout)



Name: hpxpy::TH2F

Line: 1, 1

Opacity: 1

Fill

Opacity: 1

Title: py vs px

Plot: [✓ 2-D] 3-D

Contour: None

Cont #: 20

☐ Arrow ☐ Box

☑ Col ☐ Scat

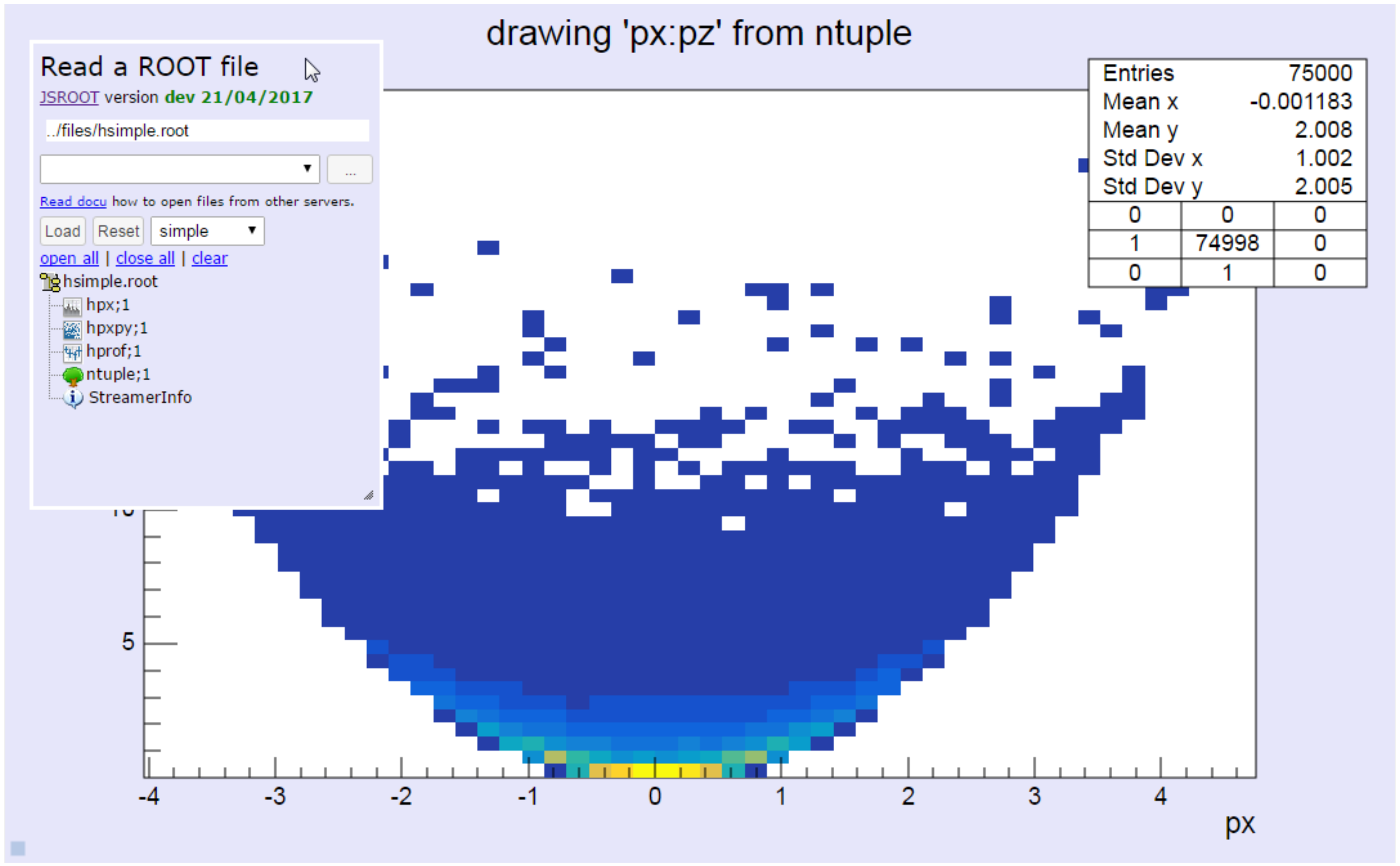
☐ Text ☑ Palette

Marker: 1.0

Opacity: 1

```
hpxpy
Entries      25000
Mean x −0.003961
Mean y−0.003377
RMS x       0.9978
RMS y        1.006
```

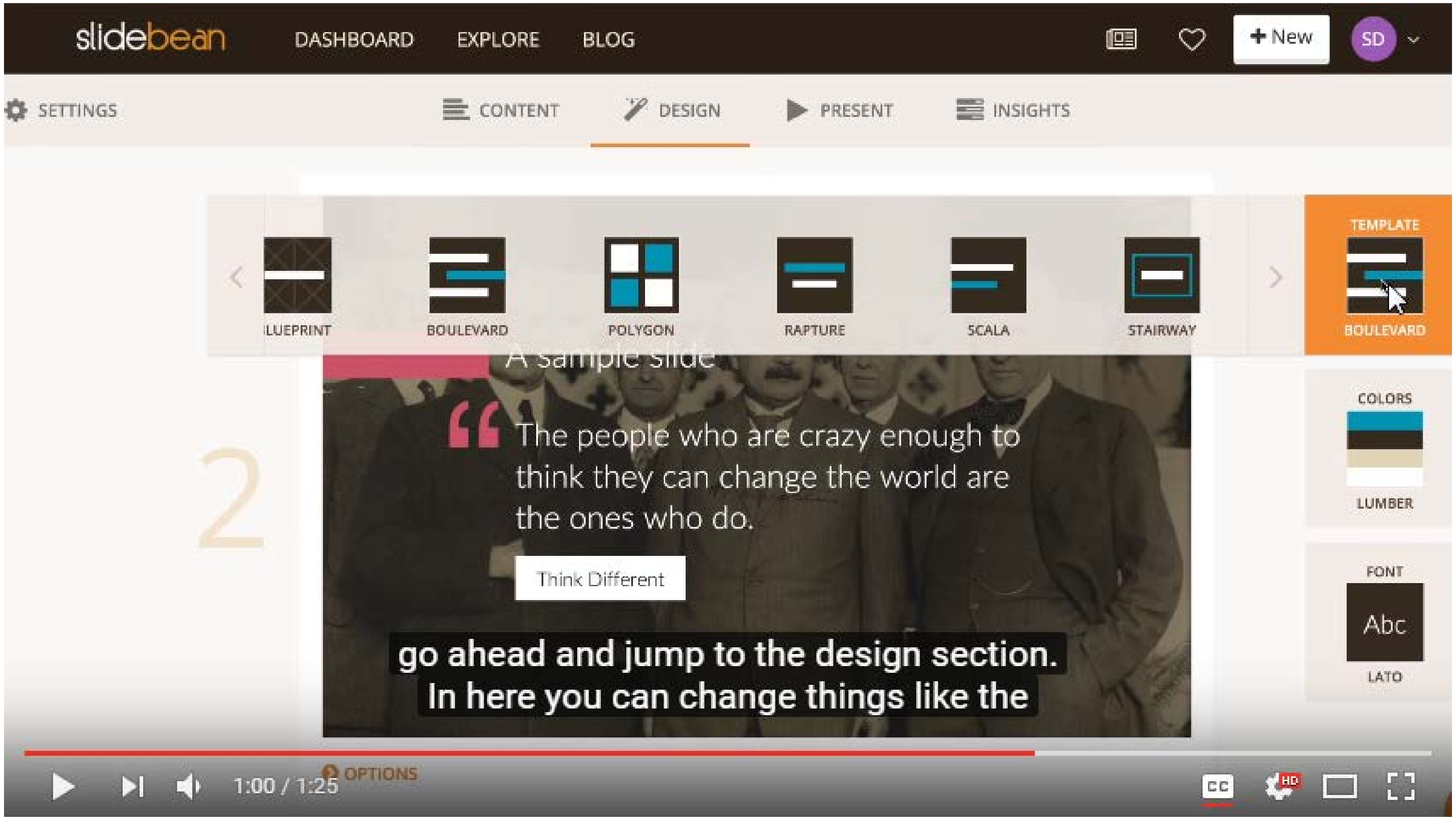
Floating example


drawing 'px:pz' from ntuple
Read a ROOT file
JSROOT version dev 21/04/2017
../files/hsimple.root
Read docu how to open files from other servers.
Load
Reset
simple
open all | close all | clear
hsimple.root
hpx;1
hpxpy;1
hprof;1
ntuple;1
StreamerInfo
10
5
-4
-3
-2
-1
0
1
2
3
4
px

Entries	75000
Mean x	-0.001183
Mean y	2.008
Std Dev x	1.002
Std Dev y	2.005

0	0	0
1	74998	0
0	1	0

Overlay example



The people who are crazy enough to think they can change the world are the ones who do.

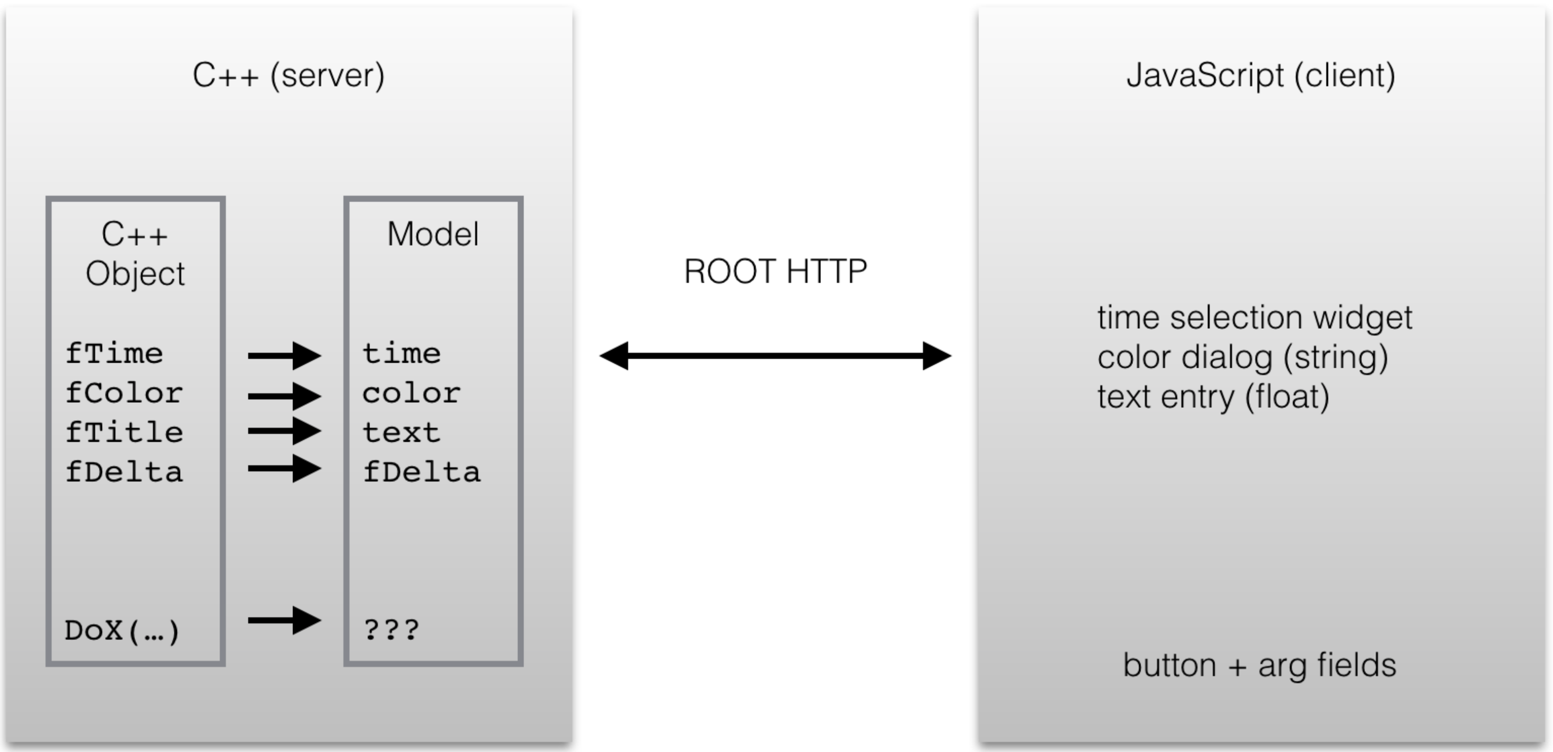
Think Different

go ahead and jump to the design section.
In here you can change things like the

User’s Defined GUI

- To implement their own GUI, users should:
  - Define the interface (model) from their C++ object to be implemented by the JavaScript GUI
  - Register the object in the http server, to make it visible/accessible on the client side
  - Create their GUI using the object's' methods/types

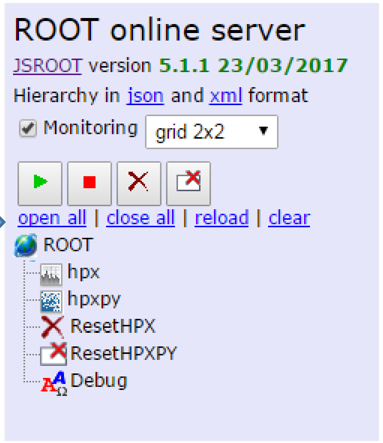
Interface (model) Definition



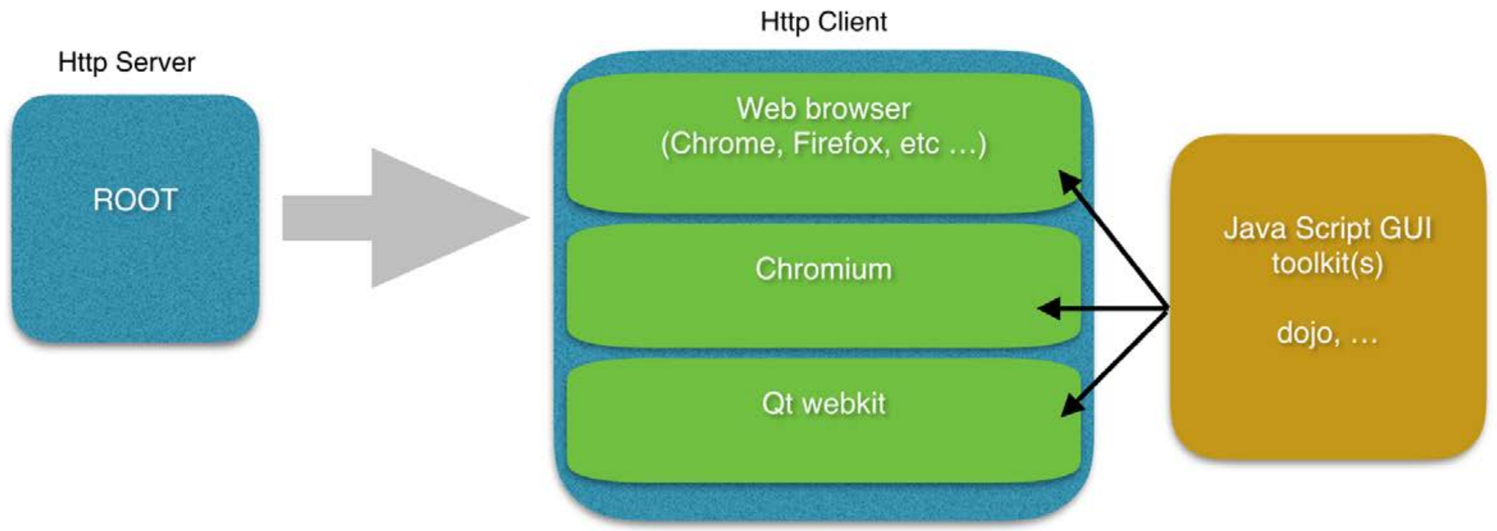
Current Status

- The TRootSniffer class gives full description of the ROOT objects hierarchy used with THttpServer
- It is already possible to add objects and actions to THttpServer:

```
// register histograms
serv->Register("/", hpx);

// register simple start/stop commands
serv->RegisterCommand("/Start", "bFillHist=kTRUE;",
                      "button;rootsys/icons/ed_execute.png");

// register commands, invoking object methods
serv->RegisterCommand("/ResetHPX","/hpx/->Reset()",
                      "button;rootsys/icons/ed_delete.png");
```


ROOT online server
JSROOT version 5.1.1 23/03/2017
Hierarchy in json and xml format

Client-Server architecture



HTTP Client

- Any Web browser like Firefox, Chrome, etc... Allowing to render and interact with the graphics locally or remotely
- Local client allowing to render and interact with the graphics locally only. Examples of such tool are libchrome (from the Chromium project) and the Qt Webkit
- libchrome need investigations
- Some preliminary tests have been done with WebKit

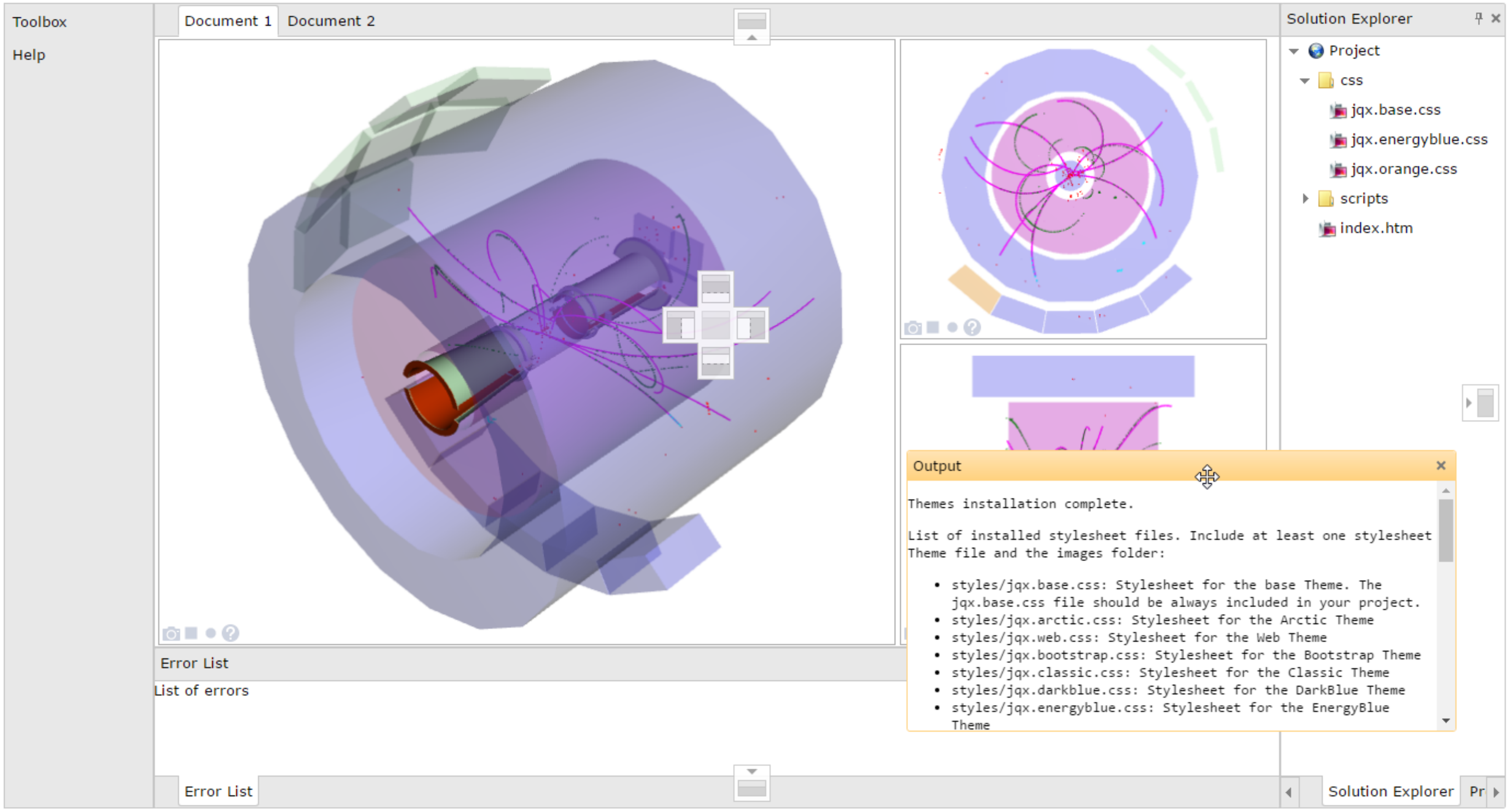
WebKit - Preliminary Tests

- Qt4 and Qt5 implements QWebView widget, based on WebKit.
- Such widget can be integrated into any qt-based GUI as any other normal QWidget.
- Some preliminary tests with the "fancybrowser" example from both qt4 and qt5 show that JSROOT graphics in general works! One could display SVG and WebGL graphics smoothly.

JavaScript GUI libraries

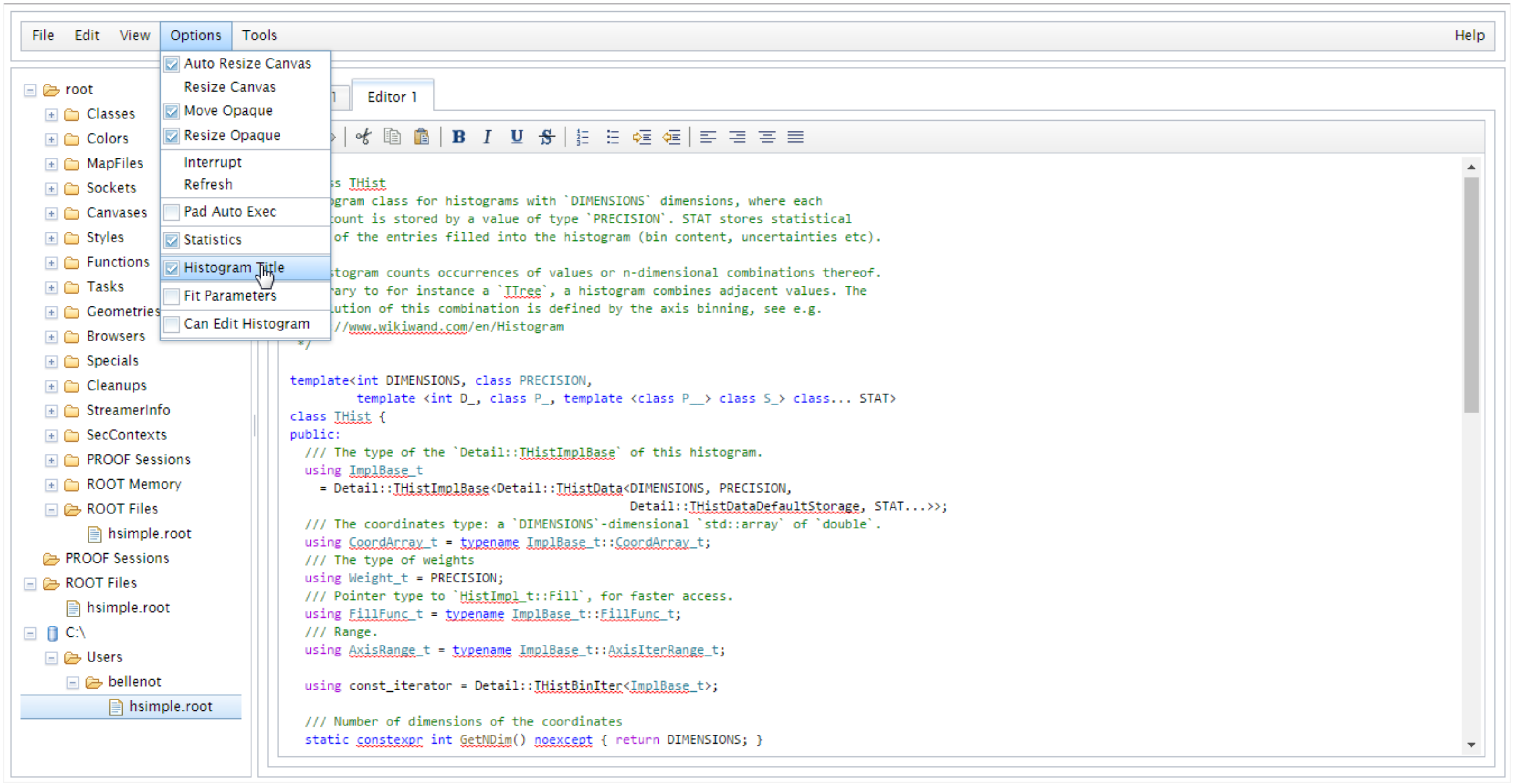
- There are many of them
  - jQWidgets http://www.jqwidgets.com (Creative Commons Attribution-NonCommercial 3.0)
  - Dijit (based on dojo) https://dojotoolkit.org (New BSD or Academic Free License version 2.1)
  - Webix http://webix.com (GPL v3)
  - OpenUI5 http://openui5.org (Apache-2.0)
  - ...
- See also
  - http://stackoverflow.com/questions/200284
  - https://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks

jQWidgets example



```
Themes installation complete.

List of installed stylesheet files. Include at least one stylesheet
Theme file and the images folder:

  • styles/jqx.base.css: Stylesheet for the base Theme. The
    jqx.base.css file should be always included in your project.
  • styles/jqx.arctic.css: Stylesheet for the Arctic Theme
  • styles/jqx.web.css: Stylesheet for the Web Theme
  • styles/jqx.bootstrap.css: Stylesheet for the Bootstrap Theme
  • styles/jqx.classic.css: Stylesheet for the Classic Theme
  • styles/jqx.darkblue.css: Stylesheet for the DarkBlue Theme
  • styles/jqx.energyblue.css: Stylesheet for the EnergyBlue
    Theme
```

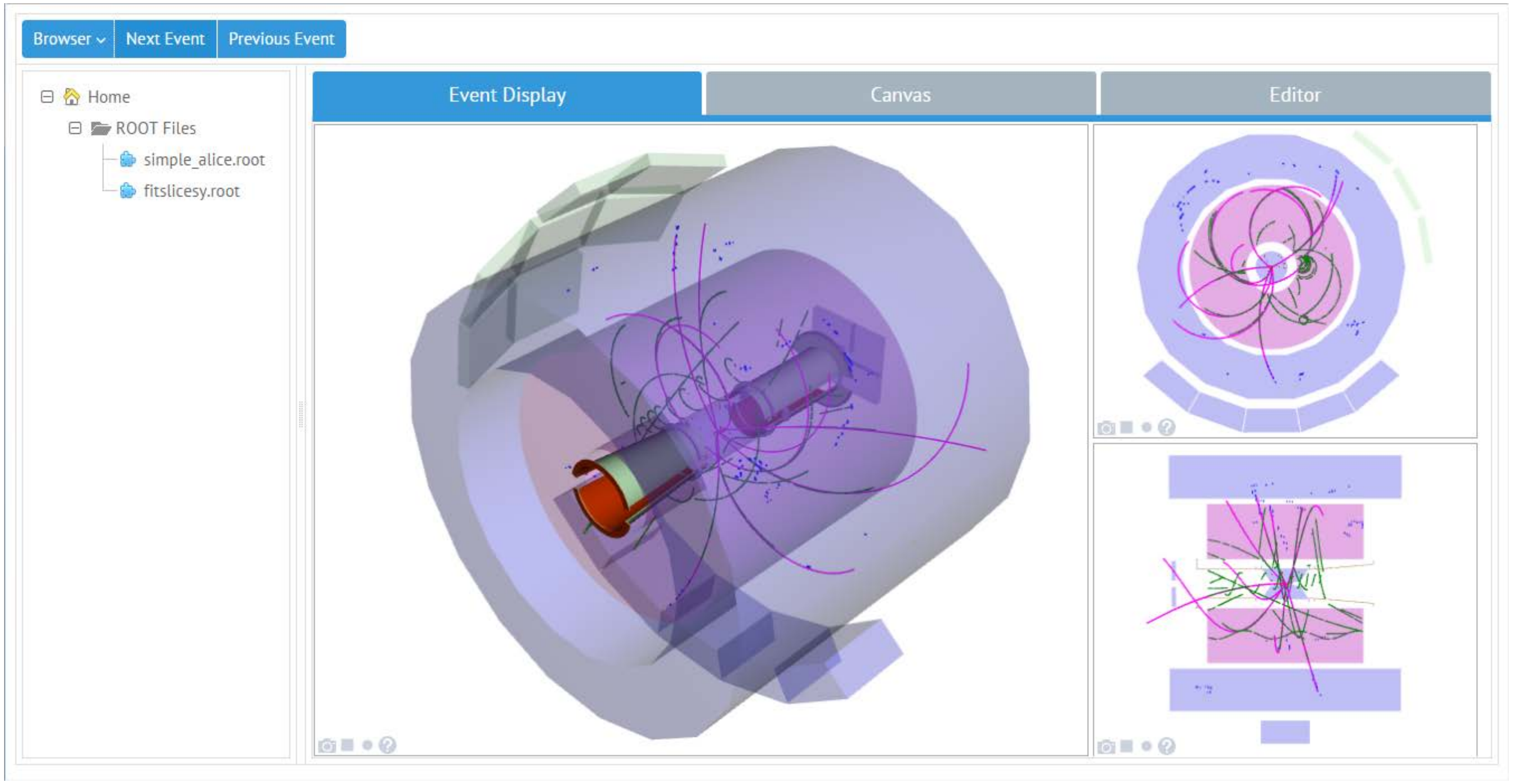
Dijit Example



- ✓ Auto Resize Canvas
- Resize Canvas
- ✓ Move Opaque
- ✓ Resize Opaque
- Interrupt
- Refresh
- ☐ Pad Auto Exec
- ✓ Statistics
- ✓ Histogram Title
- ☐ Fit Parameters
- ☐ Can Edit Histogram

```
s THist
gram class for histograms with `DIMENSIONS` dimensions, where each
ount is stored by a value of type `PRECISION`. STAT stores statistical
 of the entries filled into the histogram (bin content, uncertainties etc).

togram counts occurrences of values or n-dimensional combinations thereof.
ary to for instance a `TTree`, a histogram combines adjacent values. The
ution of this combination is defined by the axis binning, see e.g.
 //www.wikiwand.com/en/Histogram
 */

template<int DIMENSIONS, class PRECISION,
         template <int D_, class P_, template <class P__> class S_> class... STAT>
class THist {
public:
  /// The type of the `Detail::THistImplBase` of this histogram.
  using ImplBase_t
    = Detail::THistImplBase<Detail::THistData<DIMENSIONS, PRECISION,
                                              Detail::THistDataDefaultStorage, STAT...>>;
  /// The coordinates type: a `DIMENSIONS`-dimensional `std::array` of `double`.
  using CoordArray_t = typename ImplBase_t::CoordArray_t;
  /// The type of weights
  using Weight_t = PRECISION;
  /// Pointer type to `HistImpl_t::Fill`, for faster access.
  using FillFunc_t = typename ImplBase_t::FillFunc_t;
  /// Range.
  using AxisRange_t = typename ImplBase_t::AxisIterRange_t;

  using const_iterator = Detail::THistBinIter<ImplBase_t>;

  /// Number of dimensions of the coordinates
  static constexpr int GetNDim() noexcept { return DIMENSIONS; }
```

Webix Example



What Next

- Select a JavaScript GUI library
- Implement wrappers (C++ and JavaScript)
- Implement a JavaScript Tree Viewer