

Rootana : Event Display And Web GUIs

**Thomas Lindner | TRIUMF
MIDAS Seminar – July 2015**

Outline

- Rootana
- X11 GUIs
- Web-based GUIs

Rootana: history / status

- **Rootana** : a package for accessing and analyzing MIDAS data online and offline.
 - Despite name, the classes for accessing MIDAS can be used without ROOT; but normal use does include ROOT.
 - Need to have MIDAS package installed in order to see online MIDAS data, but not to read MIDAS files.
- **Historical note (from Konstantin) (unknown date):**

The origins of this package date back a few years when I wrote some C++ classes to read MIDAS files for the Dragon experiment. Jonty Pearson and Joe Chuma have since improved and added to my work. Then during the Summer of 2006, I wrote some more C++ classes for access to live data and for access to the live ODB, for use by the ALPHA experiment at CERN. This code was then reused for couple of test DAQ stations at TRIUMF. With the addition of ROODY access using the "midas server", ripped out from mana.c, it is now used for the PIENU beam test.
- **Rootana recently moved to bitbucket:**
<https://bitbucket.org/tmidas/rootana/>

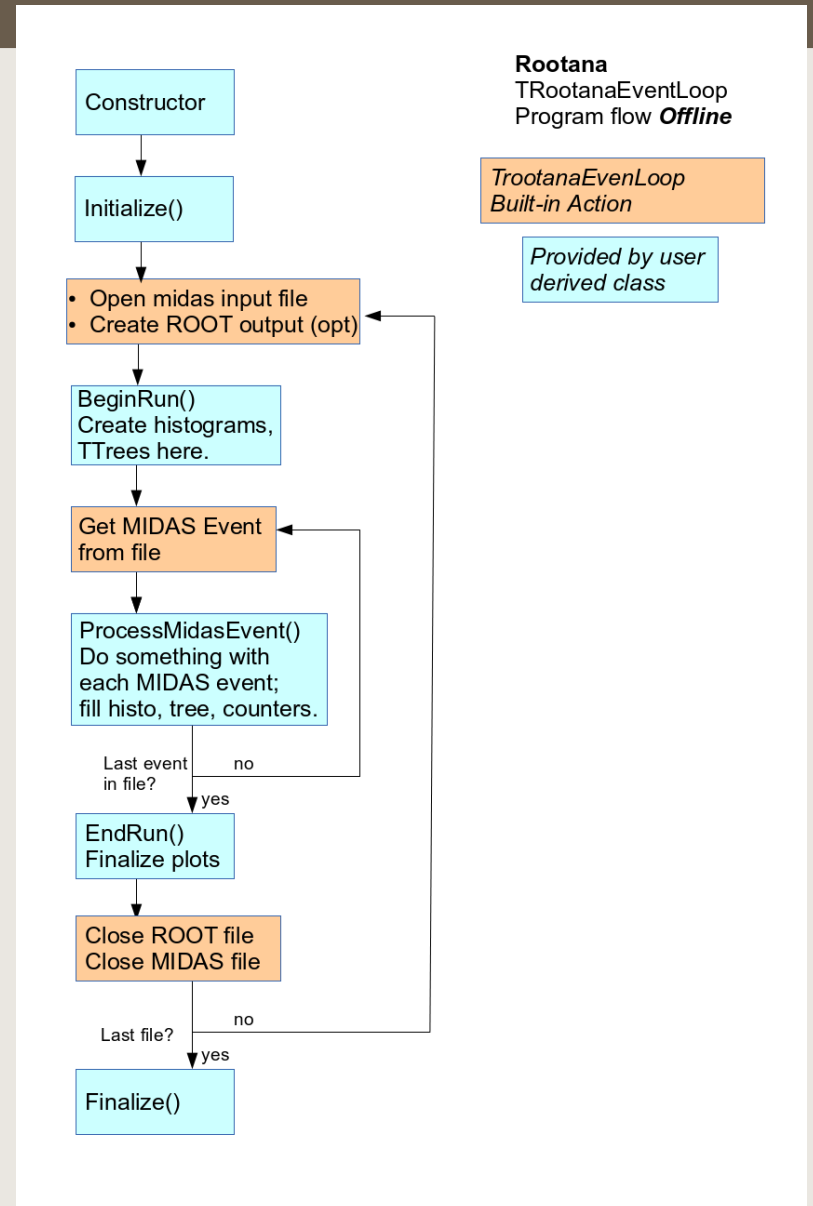
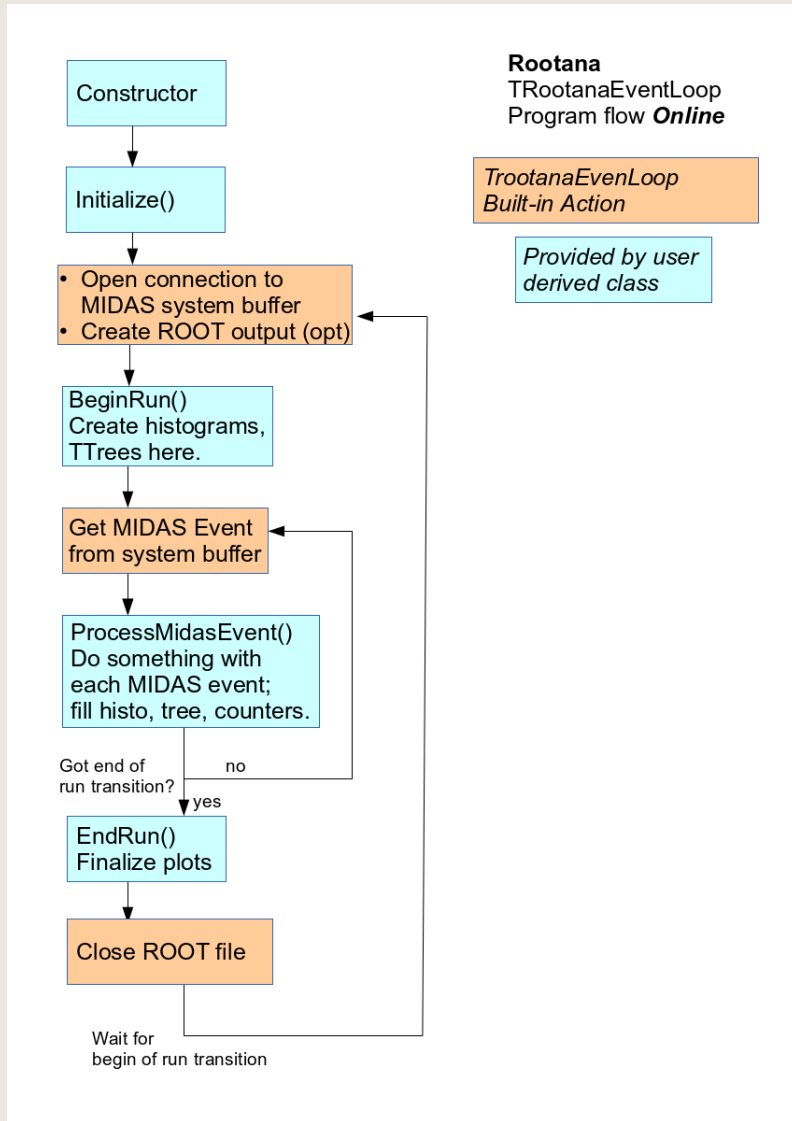
Rootana: overview

- ROOTANA package includes 5 major components:
 - standalone library for reading and writing data files in the MIDAS .mid format, and for decoding XML dump of MIDAS ODB in MIDAS data file. (libMidasInterface)
 - a C++ interface class for connecting to an active MIDAS experiment, accessing ODB (read and write) and getting event data. (libMidasInterface)
 - **set of C++ classes for exporting ROOT histogram and other objects to an external viewer for interactive visualization of live data, typically using the ROODY histogram viewer or using a standard web browser (experimental feature). (libMidasServer, libNetDirectory, libXmlServer)**
 - simple examples of using these components (a graphical analyzer, an event dump and an event skim programs) (analyzer.cxx, event_dump.cxx, event_skim.cxx)
 - **a full featured framework for graphical data analysis including code to unpack typical VME and CAMAC modules (examples, libAnalyzer, linAnalyzerDisplay)**
- I will mostly concentrate on the use of rootana for creating GUIs for looking at data online; but it also works for writing converters (like midas->root converters).

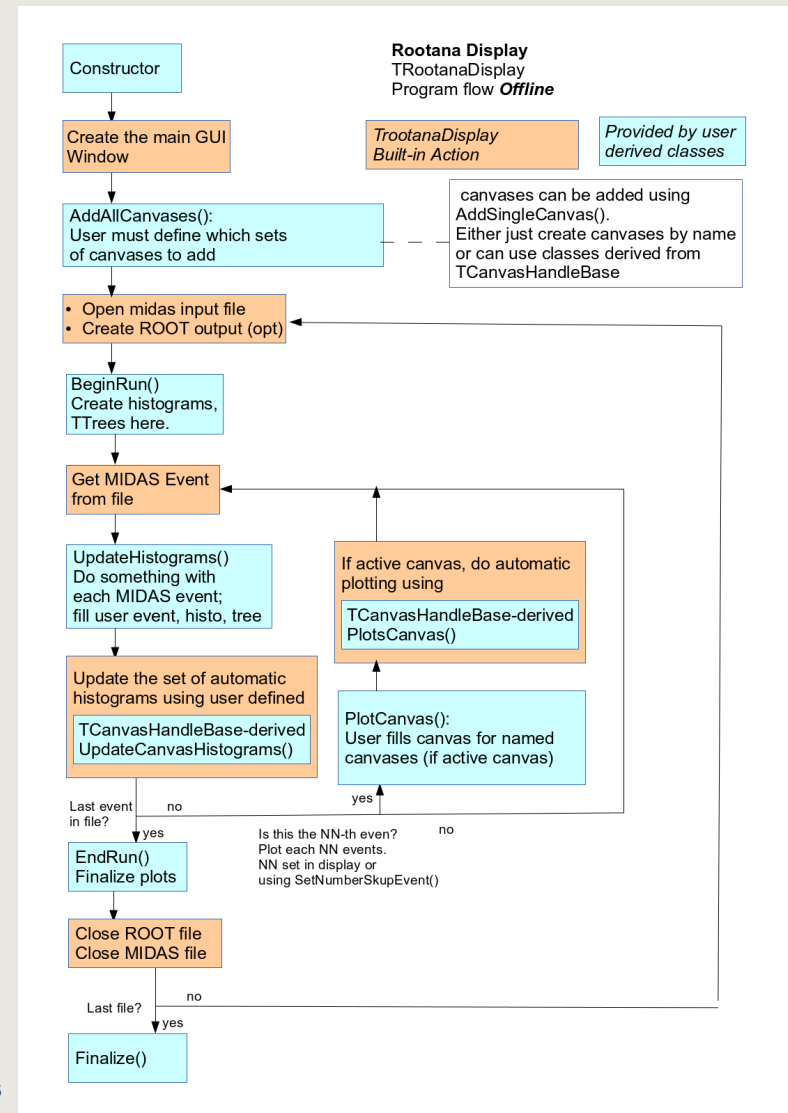
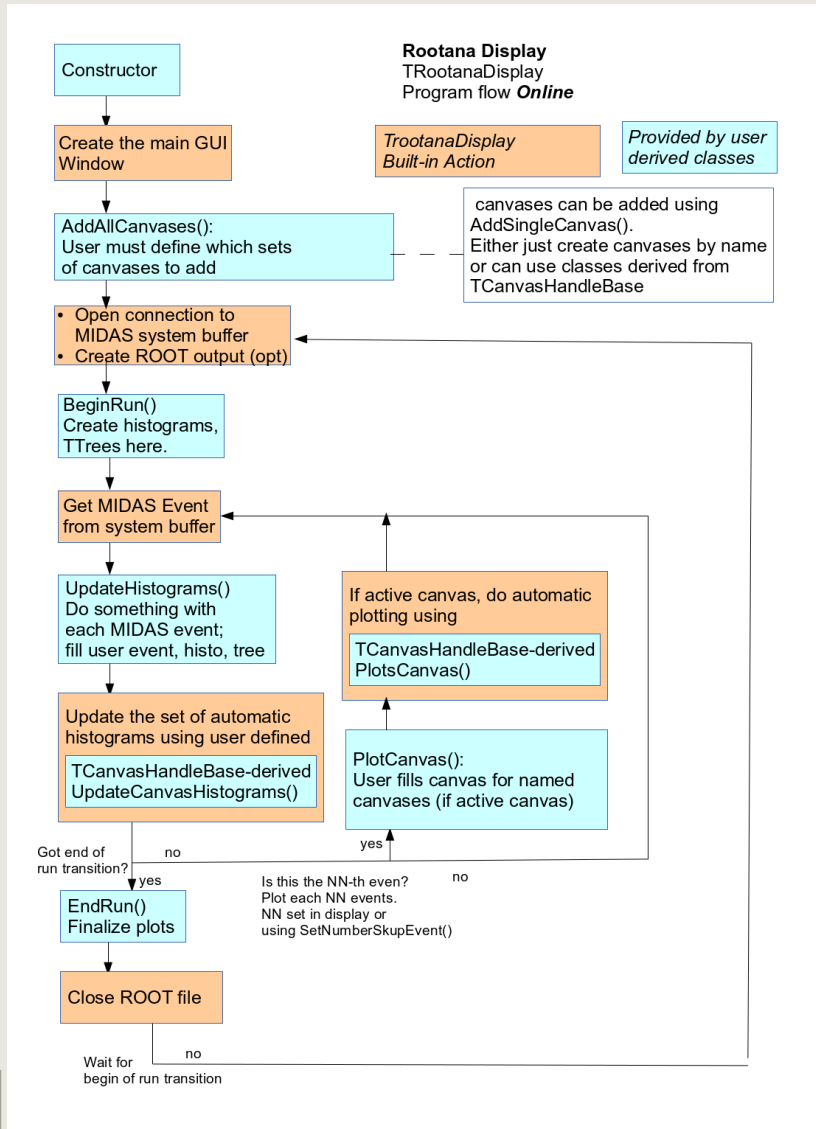
Analyzer Framework

- My attempts to make more object orientated rootana programs (libAnalyzer, libAnalyzerDisplay)
- Have a base class (TRootAnaEventLoop) that executables can inherit from or TRootanaDisplay that GUIs can inherit from.
- Standard set of decoders (mostly for CAEN digitizers) and analyzer manager (TAnaManager).
- Goal to simplify writing analyzer programs; not entirely successful yet; some complaints about too many classes and various bugs.
- More documentation
<http://ladd00.triumf.ca/~lindner/rootana/html/analyzerClass.html>
<http://ladd00.triumf.ca/~lindner/rootana/html/displayClass.html>

RootanaEventloop program flow

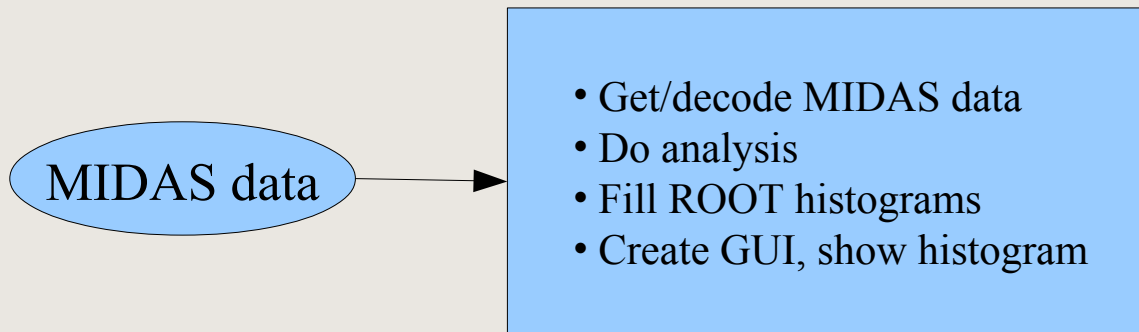


TRootanaDisplay program flow

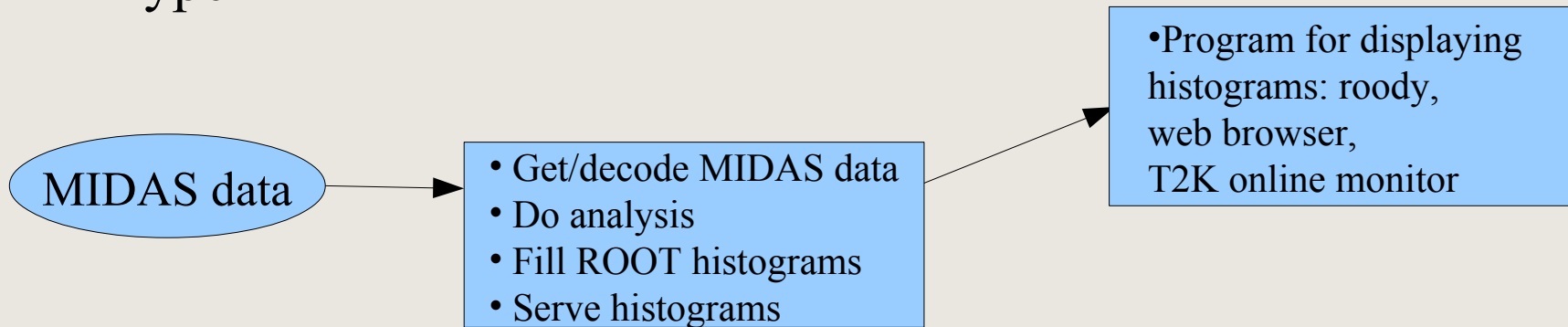


Rootana: 2 types of GUIs

Type 1

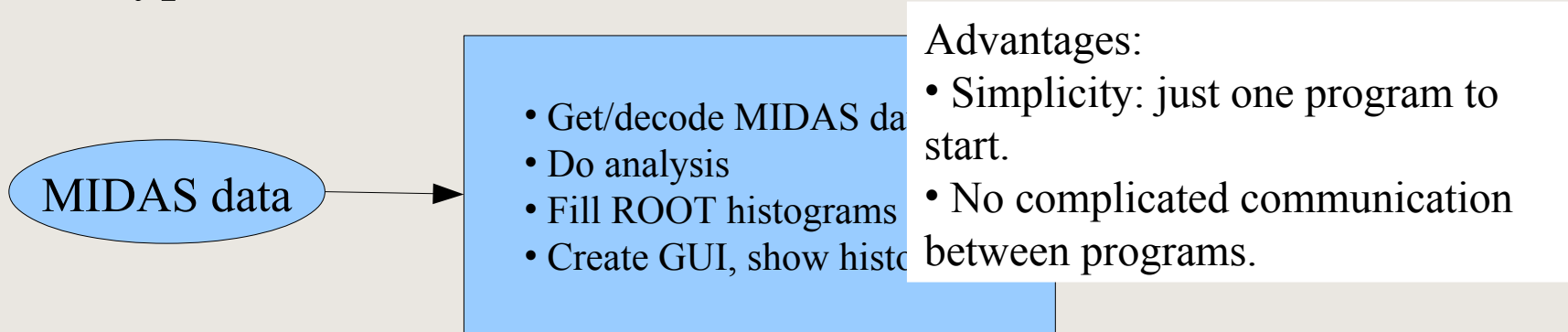


Type 2

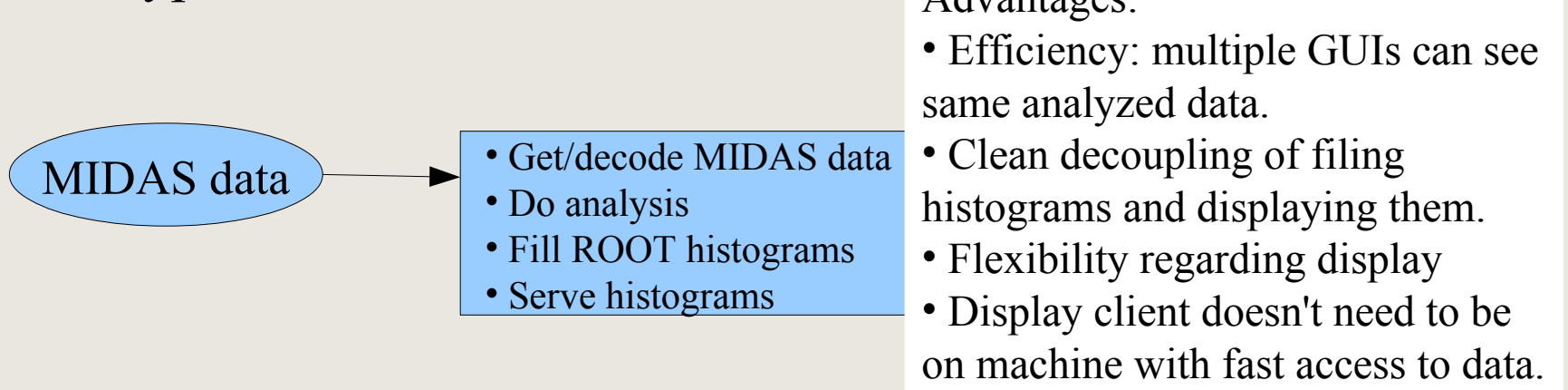


Rootana: 2 types of GUIs

Type 1

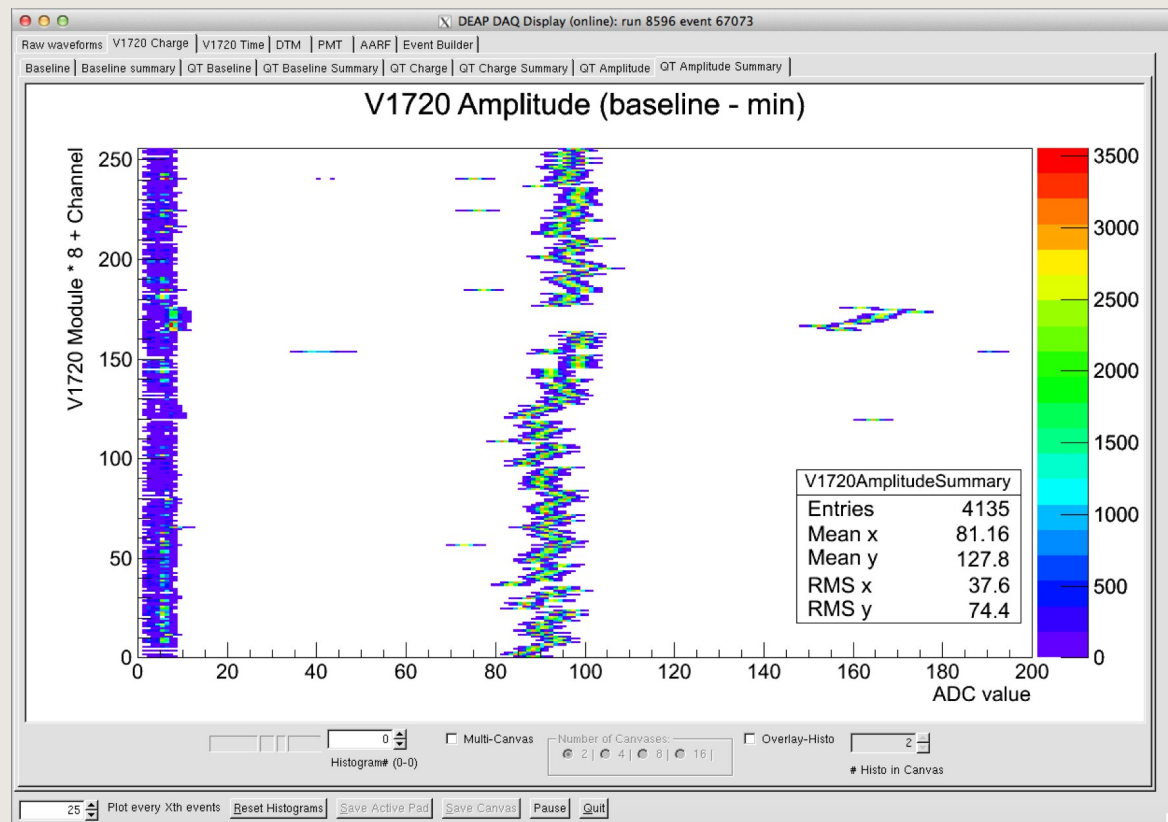


Type 2



Type 1 GUI

- Detailed GUI made for DEAP; ROOT-based GUI.
- Code here
<https://bitbucket.org/ttriumfdaq/deap-daqdisplay>



Type 2 GUI: ROOT http server

- Piece of new ROOT code seems interesting: THttpServer
- THttpServer: http server for arbitrary ROOT programs.
- Provides built-in way web-based way to
 - Browse current ROOT objects
 - Access picture of histograms
 - Access JSON dump of histogram data.
- Lots of potential
 - Little MIDAS/rootana application fills histograms and start THttpServer.
 - You write simple html to check your online display through web browser.

<http://indico.cern.ch/event/304944/session/10/contribution/286>

<http://indico.cern.ch/event/304944/session/10/contribution/288>

<https://github.com/linev/jsroot/blob/master/docs/HttpServer.md>

ROOT http server

- Piece of new ROOT code seems interesting: THttpServer
- THttpServer: http server for arbitrary ROOT programs.
- Provides built-in way web-based way to
 - Browse current ROOT objects
 - Access picture of histograms
 - Access JSON dump of histogram data.
- Lots of potential
 - Little MIDAS/rootana application fills histograms and start ThttpsServer.
 - You check your c

GRIFFIN folks implemented a version of this, encoding ROOT histos in JSON a couple years ago. Now baked into ROOT.

<http://indico.cern.ch/event/304944/session/10/contribution/286>

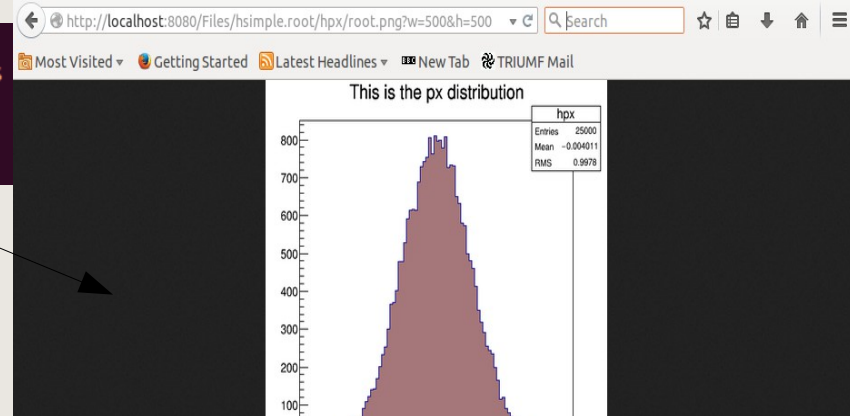
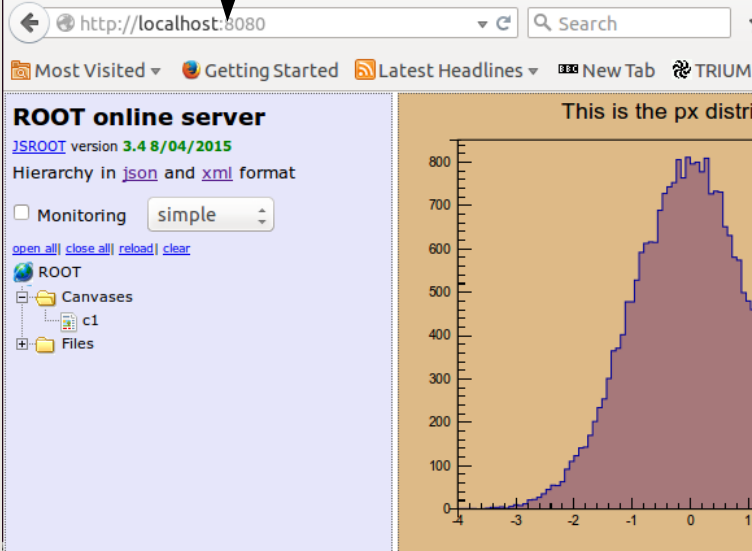
<http://indico.cern.ch/event/304944/session/10/contribution/288>

<https://github.com/linev/jsroot/blob/master/docs/HttpServer.md>

ROOT http server

```

root [0] ./x $ROOTSYS/tutorials/hsimple.C
hsimple : Real Time = 0.81 seconds Cpu Time = 0.30 seconds
(class TFile*)0x2ef7f30
root [1] serv = new THttpServer("http:8080");
Info in <TCivetweb::Create>: Starting HTTP server on port 8080
    
```

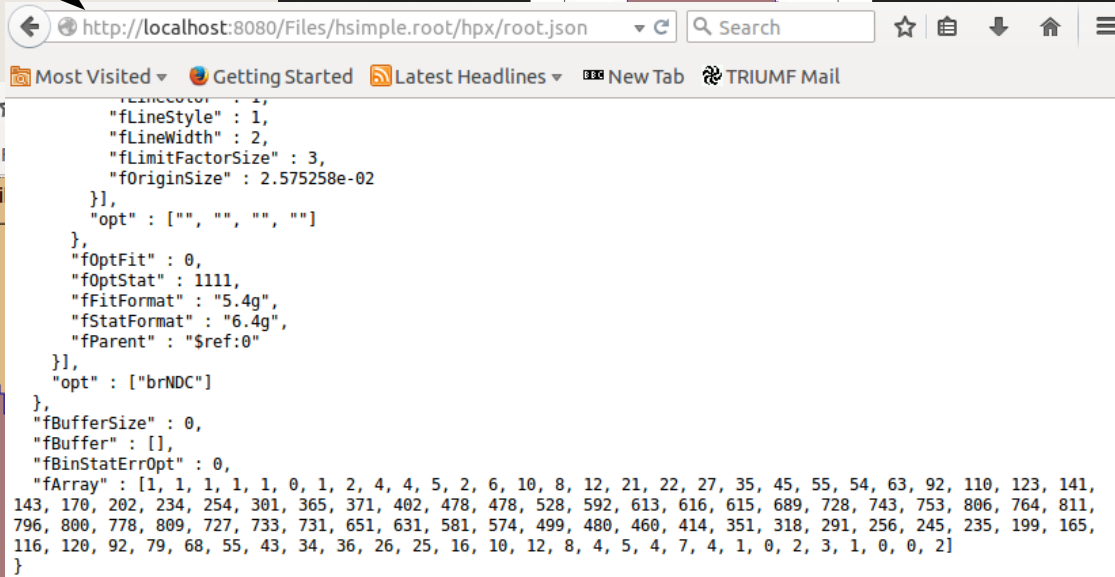
Browser address: `http://localhost:8080`

ROOT online server
 JSROOT version 3.4.8/04/2015
 Hierarchy in [json](#) and [xml](#) format

Monitoring

[open all](#) [close all](#) [reload](#) [clear](#)

- ROOT
 - Canvases
 - c1
 - Files



Browser address: `http://localhost:8080/Files/hsimple.root/hpx/root.json`

```

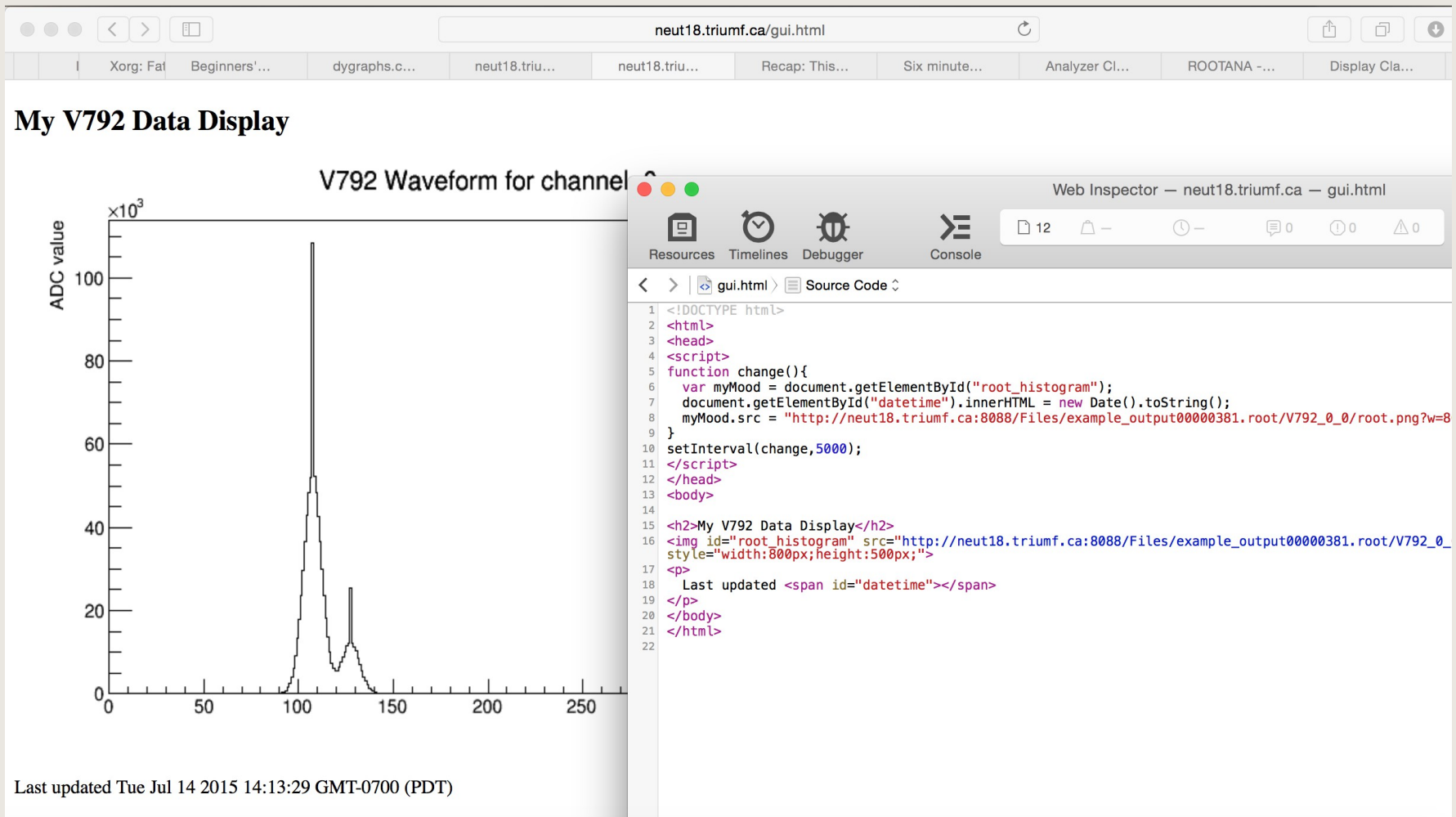
{
  "fLineColor" : 1,
  "fLineStyle" : 1,
  "fLineWidth" : 2,
  "fLimitFactorSize" : 3,
  "fOriginSize" : 2.575258e-02
},
"opt" : ["", "", "", ""]
},
"fOptFit" : 0,
"fOptStat" : 1111,
"fFitFormat" : "5.4g",
"fStatFormat" : "6.4g",
"fParent" : "$ref:0"
}],
"opt" : ["brNDC"]
},
"fBufferSize" : 0,
"fBuffer" : [],
"fBinStatErrOpt" : 0,
"fArray" : [1, 1, 1, 1, 1, 0, 1, 2, 4, 4, 5, 2, 6, 10, 8, 12, 21, 22, 27, 35, 45, 55, 54, 63, 92, 110, 123, 141, 143, 170, 202, 234, 254, 301, 365, 371, 402, 478, 528, 592, 613, 616, 615, 689, 728, 743, 753, 806, 764, 811, 796, 800, 778, 809, 727, 733, 731, 651, 631, 581, 574, 499, 480, 460, 414, 351, 318, 291, 256, 245, 235, 199, 165, 116, 120, 92, 79, 68, 55, 43, 34, 36, 26, 25, 16, 10, 12, 8, 4, 5, 4, 7, 4, 1, 0, 2, 3, 1, 0, 0, 2]
}
    
```

ROOT http server in rootana

- Added option to TRootAnaEventLoop to start THttpServer.
- Example
`./anaDisplay.exe -Hneut14:7071 -r8088`
(connect online to experiment on neut14, start ThttpServer on port 8088).
 - Note: change to rootana was trivial; 2-3 line change. This is easy functionality for you to add to non-rootana programs.
- There may be some issue with sequencing: ThttpServer developer said that should be careful to make sure that requests for histograms were not made when histograms were being filled.
 - Haven't seen problems yet, but not high data rate tests.

Rootana : ThttpServer example 1

- Can embed ROOT png directly in webpages
<http://neut18.triumf.ca/gui.html>



The screenshot shows a web browser window displaying a ROOT histogram titled "V792 Waveform for channel". The histogram plots "ADC value" (scaled by $\times 10^3$) on the y-axis against an unlabeled x-axis. The y-axis ranges from 0 to 100, and the x-axis ranges from 0 to 250. The histogram shows a prominent peak at approximately x=105, with a smaller peak at approximately x=135.

Below the histogram, the text "Last updated Tue Jul 14 2015 14:13:29 GMT-0700 (PDT)" is displayed.

Overlaid on the right side of the browser window is the Web Inspector, showing the source code of the page (gui.html). The code includes a JavaScript function that updates the histogram and the last updated date every 5000 milliseconds.

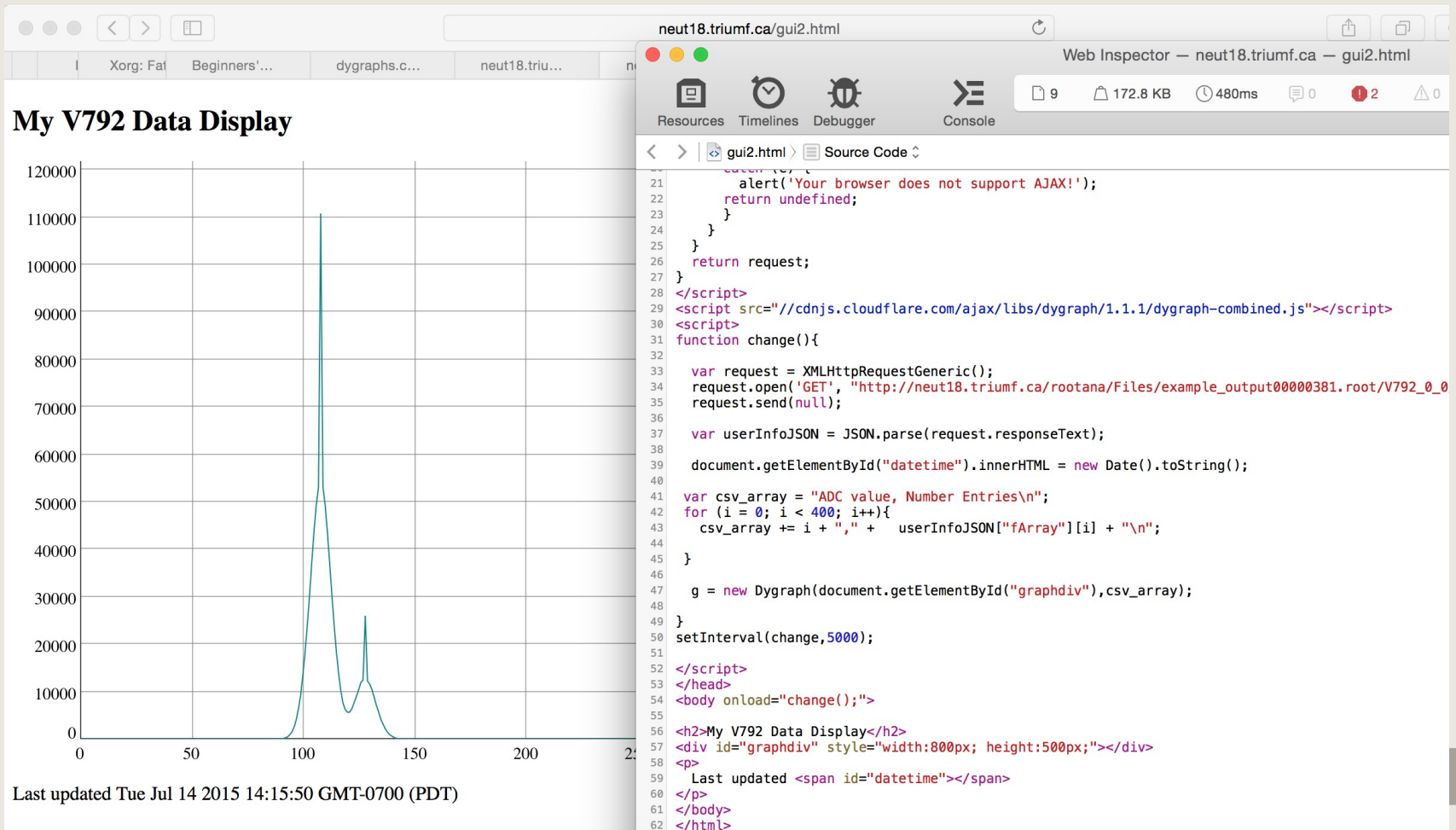
```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <script>
5 function change(){
6   var myMood = document.getElementById("root_histogram");
7   document.getElementById("datetime").innerHTML = new Date().toString();
8   myMood.src = "http://neut18.triumf.ca:8088/Files/example_output0000381.root/V792_0_0/root.png?w=8
9 }
10 setInterval(change,5000);
11 </script>
12 </head>
13 <body>
14
15 <h2>My V792 Data Display</h2>
16 
18 <p>
19   Last updated <span id="datetime"></span>
20 </p>
21 </body>
22 </html>

```

Rootana : ThttpServer example 2

- Or can use json data and plot it using modern plotting package:
<http://neut18.triumf.ca/gui2.html>
 (this example uses dygraphs, from Bryerton GRIF-16 display)



The image shows a browser window displaying a web page titled "My V792 Data Display". The page features a line graph with a y-axis ranging from 0 to 120,000 and an x-axis ranging from 0 to 200. The graph shows a sharp peak at approximately x=110 with a value of about 110,000, and a smaller peak at approximately x=130 with a value of about 25,000. The browser's developer tools are open, showing the source code of the page. The code includes a JavaScript function named `change()` that uses the `XMLHttpRequest` object to fetch data from a server endpoint and then uses the `dygraph` library to plot the data. The code also includes a `setInterval` call to update the graph every 5000 milliseconds.

```

21     alert('Your browser does not support AJAX!');
22     return undefined;
23   }
24 }
25 }
26 return request;
27 }
28 </script>
29 <script src="//cdnjs.cloudflare.com/ajax/libs/dygraph/1.1.1/dygraph-combined.js"></script>
30 <script>
31 function change(){
32
33   var request = XMLHttpRequestGeneric();
34   request.open('GET', "http://neut18.triumf.ca/rootana/Files/example_output0000381.root/V792_0_0");
35   request.send(null);
36
37   var userInfoJSON = JSON.parse(request.responseText);
38
39   document.getElementById("datetime").innerHTML = new Date().toString();
40
41   var csv_array = "ADC value, Number Entries\n";
42   for (i = 0; i < 400; i++){
43     csv_array += i + "," + userInfoJSON["fArray"][i] + "\n";
44   }
45
46   g = new Dygraph(document.getElementById("graphdiv"), csv_array);
47
48
49 }
50 setInterval(change, 5000);
51 </script>
52 </head>
53 <body onload="change();">
54
55
56 <h2>My V792 Data Display</h2>
57 <div id="graphdiv" style="width:800px; height:500px;"></div>
58 <p>
59   Last updated <span id="datetime"></span>
60 </p>
61 </body>
62 </html>

```

Last updated Tue Jul 14 2015 14:15:50 GMT-0700 (PDT)

Conclusion

- Rootana has been used for providing GUI for user feedback for small/medium sized experiments.
- Currently providing support for both single-program and separate-program GUIs.
- May be a place for both solutions long term.
- But I imagine in future that more complicated experiments would prefer web-based displays.

Backup