ROOT I/O IN JAVASCRIPT

Reading ROOT files in any browser

Bertrand Bellenot
CERN, PH-SFT

INTRODUCTION

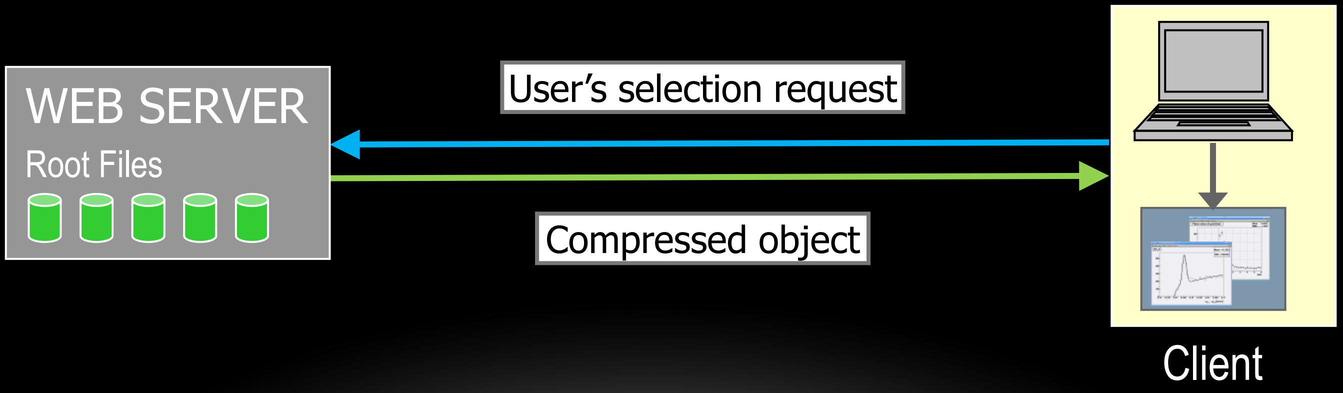
- How to share thousands of histograms on the web, without having to generate picture files (gif, jpg, …)?
- How to easily share a ROOT file?
- How to browse & display the content of a ROOT file from any platform (even from a smartphone or tablet)?
- Online monitoring?
- And obviously, all that without having to install ROOT anywhere?

REQUIREMENTS

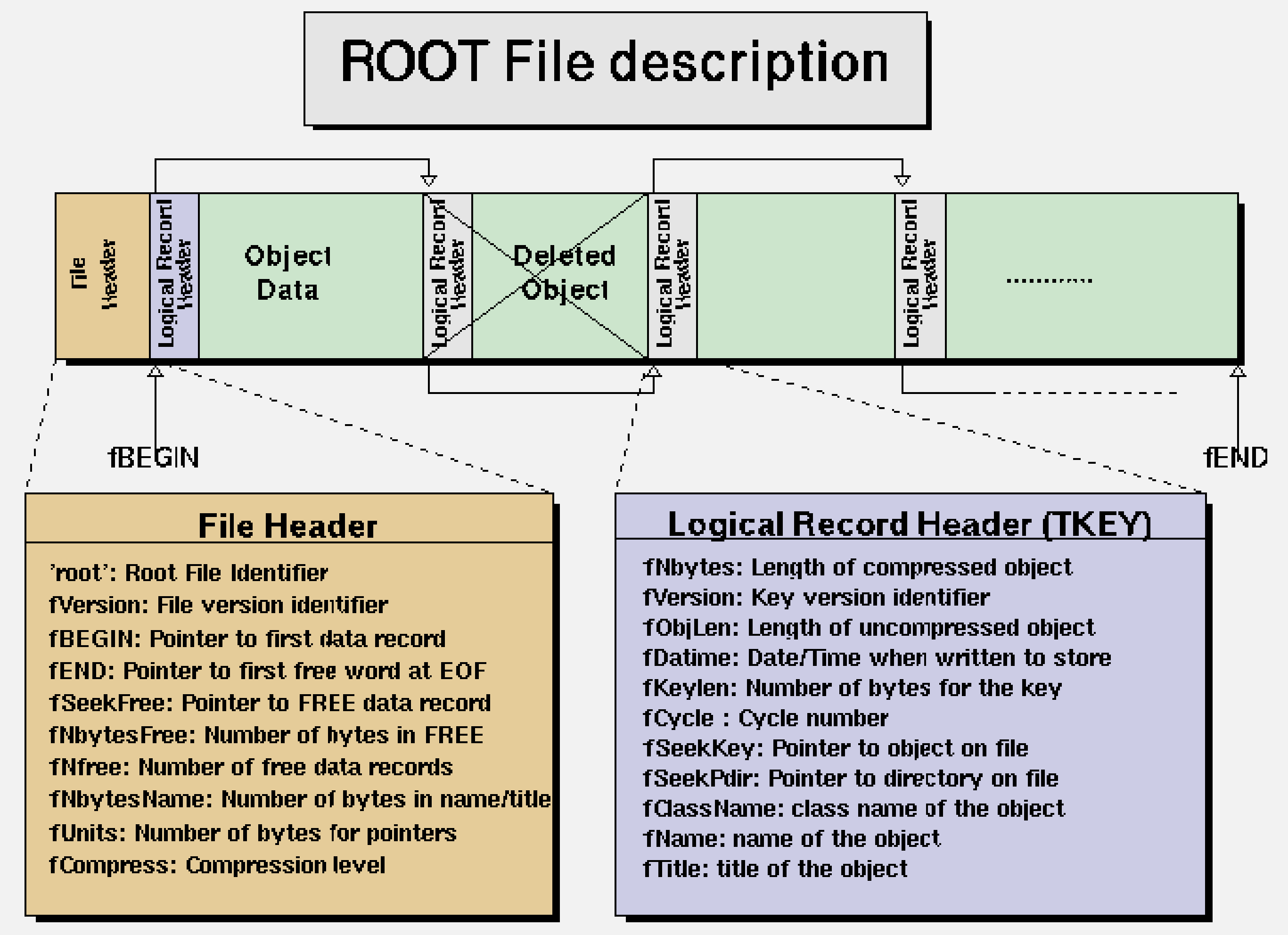
- The solution should be:
  - Portable: use available web browser
  - Lightweight: no library or application to install
  - Easy to use (user side)
  - Easy to extend and maintain (developer side)
  - Fast, with a small memory footprint

SOLUTION

- HTML & JavaScript: JSROOTIO.js
  - Copy the root file on any plain web server
  - Data is transferred over the web
  - Visualization happens on the client side



ROOT I/O



ROOT File description

File Header

'root': Root File Identifier
fVersion: File version identifier
fBEGIN: Pointer to first data record
fEND: Pointer to first free word at EOF
fSeekFree: Pointer to FREE data record
fNbytesFree: Number of bytes in FREE
fNfree: Number of free data records
fNbytesName: Number of bytes in name/title
fUnits: Number of bytes for pointers
fCompress: Compression level

Logical Record Header (TKEY)

fNbytes: Length of compressed object
fVersion: Key version identifier
fObjLen: Length of uncompressed object
fDatime: Date/Time when written to store
fKeylen: Number of bytes for the key
fCycle : Cycle number
fSeekKey: Pointer to object on file
fSeekPdir: Pointer to directory on file
fClassName: class name of the object
fName: name of the object
fTitle: title of the object

REMINDER: THE STREAMER INFO

- A TStreamerInfo object describes a persistent version of a class.
- A ROOT file contains the list of TStreamerInfo objects for all the class versions written to this file.
- A TStreamerInfo is a list of TStreamerElement objects (one per data member or base class)
- A TStreamerElement describe a data member or a base class (e.g. type, name)

READING THE FILE

- When opening the file:
  - Read the list of streamer info
  - Read the list of keys and display them in a list tree
- When the user select an item in the list tree (and only then)
  - Read the compressed buffer from the file
  - Inflate the buffer
  - Stream the object from the inflated buffer using the streamer info

READING THE FILE

- Use the XMLHttpRequest AJAX API to perform the HTTP HEAD and GET requests
- This API is highly browser dependent:
  - On IE, the binary data is in its responseBody data member (VBScript format), and has to be converted into a JavaScript string
  - On other browsers, the data can be in response, mozResponse, mozResponseArrayBuffer, or responseText...
  - Thanks to Ioannis Charalampidis, who kindly provided a working cross-browser solution

READING PARTIAL ROOT FILE

- Using HTTP byte range (available in HTTP/1.1) to download only a single compressed object when the user wants to read it
- Minimizes data transfer and memory usage
- Some browsers don't support this feature yet (e.g. Opera)

HANDLING BINARY DATA

- Compressed (zipped) objects are in binary format
- JavaScript has very little support for raw binary data
- Avoid to use the “available soon” ArrayBuffers
- Binary data is simply stored in a JavaScript string
- Accessing a single byte is easy:

```
byte = string.charCodeAt(index);
```

READING THE KEYS

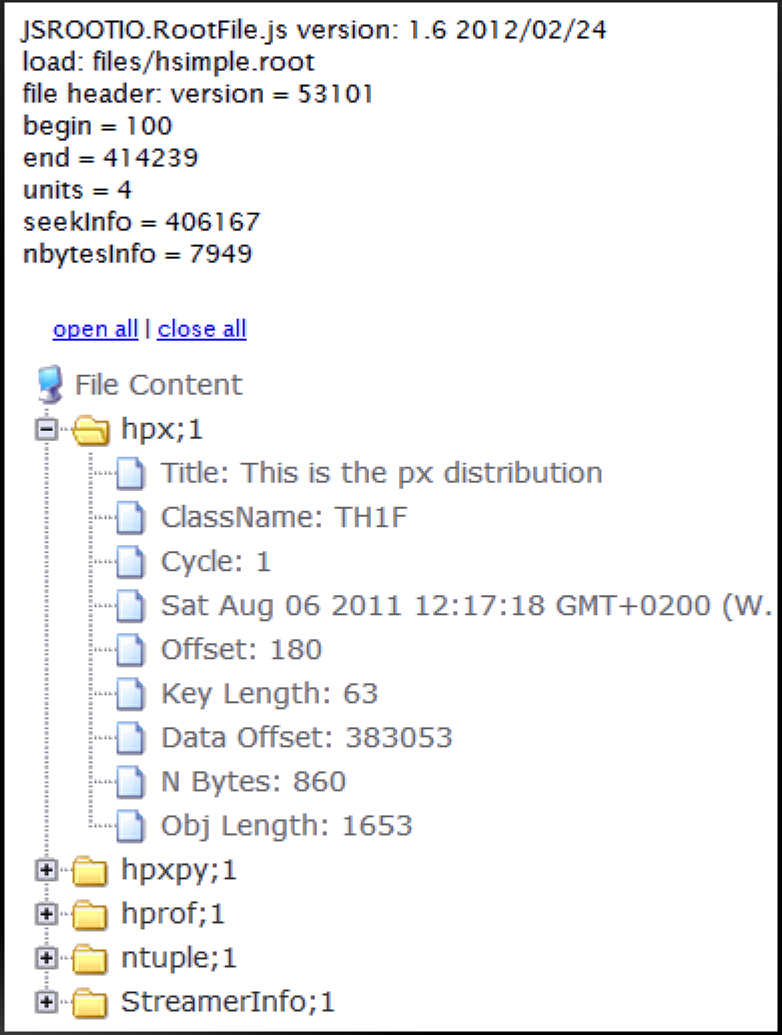
- The keys are not compressed
- They contain basic information on the object they describe, like its name and its type
- First step was quite easy, starting from already working code written by Axel Naumann
- Formatting and displaying the keys is done with a JavaScript tree menu

FILE HEADER AND KEY INFORMATION

Screenshot of the file header and the list of keys contained in hsimple.root

The hpx key is open, showing the information describing the TH1F object in the file

(displayed for debugging purpose only)



```
JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/hsimple.root
file header: version = 53101
begin = 100
end = 414239
units = 4
seekInfo = 406167
nbytesInfo = 7949
```

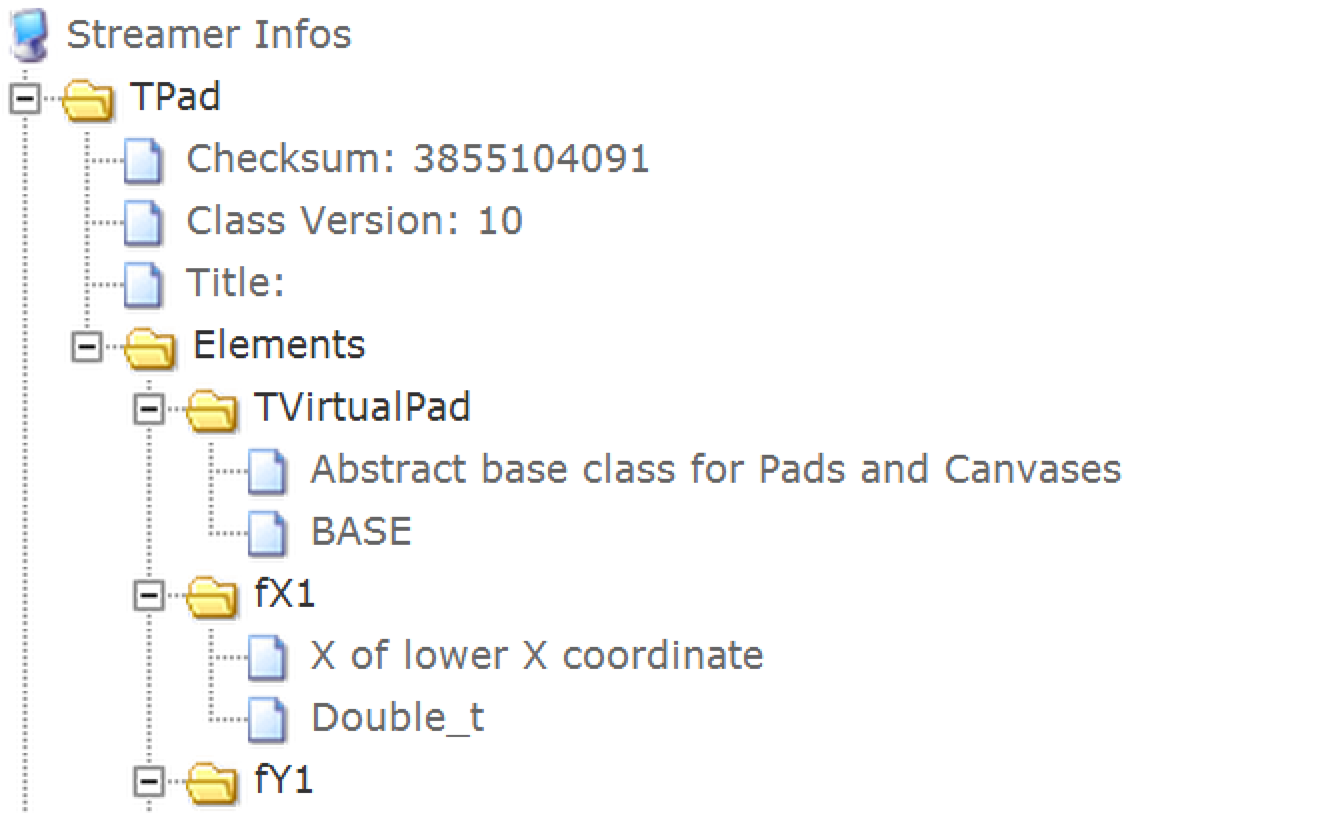
open all | close all

```
File Content
hpx;1
    Title: This is the px distribution
    ClassName: TH1F
    Cycle: 1
    Sat Aug 06 2011 12:17:18 GMT+0200 (W.
    Offset: 180
    Key Length: 63
    Data Offset: 383053
    N Bytes: 860
    Obj Length: 1653
hpxpy;1
hprof;1
ntuple;1
StreamerInfo;1
```

INFLATING & DECODING THE STREAMER INFO

- Inflating (unzipping) the buffers required:
  - A JavaScript implementation of zlib's inflate function from: http://www.onicos.com/staff/iz/amuse/javascript/expert/inflate.txt
- Implementing the streamer info functionality in JavaScript involved:
  - reverse engineering and parallel debugging of C++ and JavaScript
  - valuable help from Philippe Canal
- Streamer info can be displayed for educational / informational purposes

STREAMER INFO VISUALISATION



Streamer Infos
- TPad
  - Checksum: 3855104091
  - Class Version: 10
  - Title:
  - Elements
    - TVirtualPad
      - Abstract base class for Pads and Canvases
      - BASE
    - fX1
      - X of lower X coordinate
      - Double_t
    - fY1

READING OBJECTS, FIRST VERSION

- At the beginning, the classes' streamers were hard-coded. This approach has several issues:
  - Streamers must be updated with every change in the original class
  - Add a new streamer for every new class
  - The library is growing with every new streamer
- The only (partially) supported classes were TH1, TH2, TGraph, and TProfile

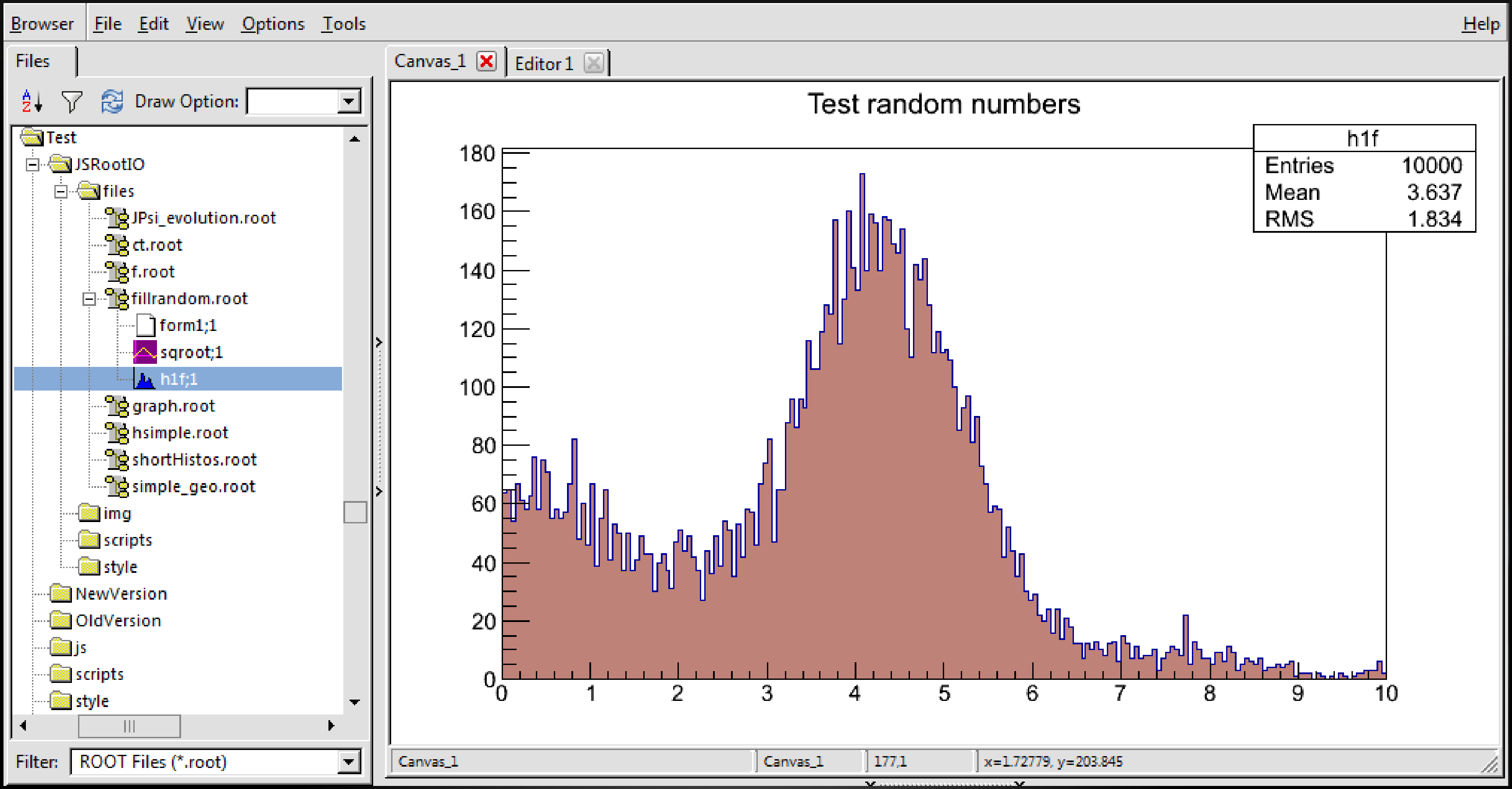
READING OBJECTS, ACTUAL STATUS

- One very nice feature of JavaScript is the possibility to dynamically (at runtime) create classes
- Allowed to implement dynamic streamers (automatically generated from the streamer info)
- Allows to potentially read any object from a ROOT file, as soon as we can read the streamer info of its class
- Navigation in (sub)directories inside a file is fully supported

GRAPHICS

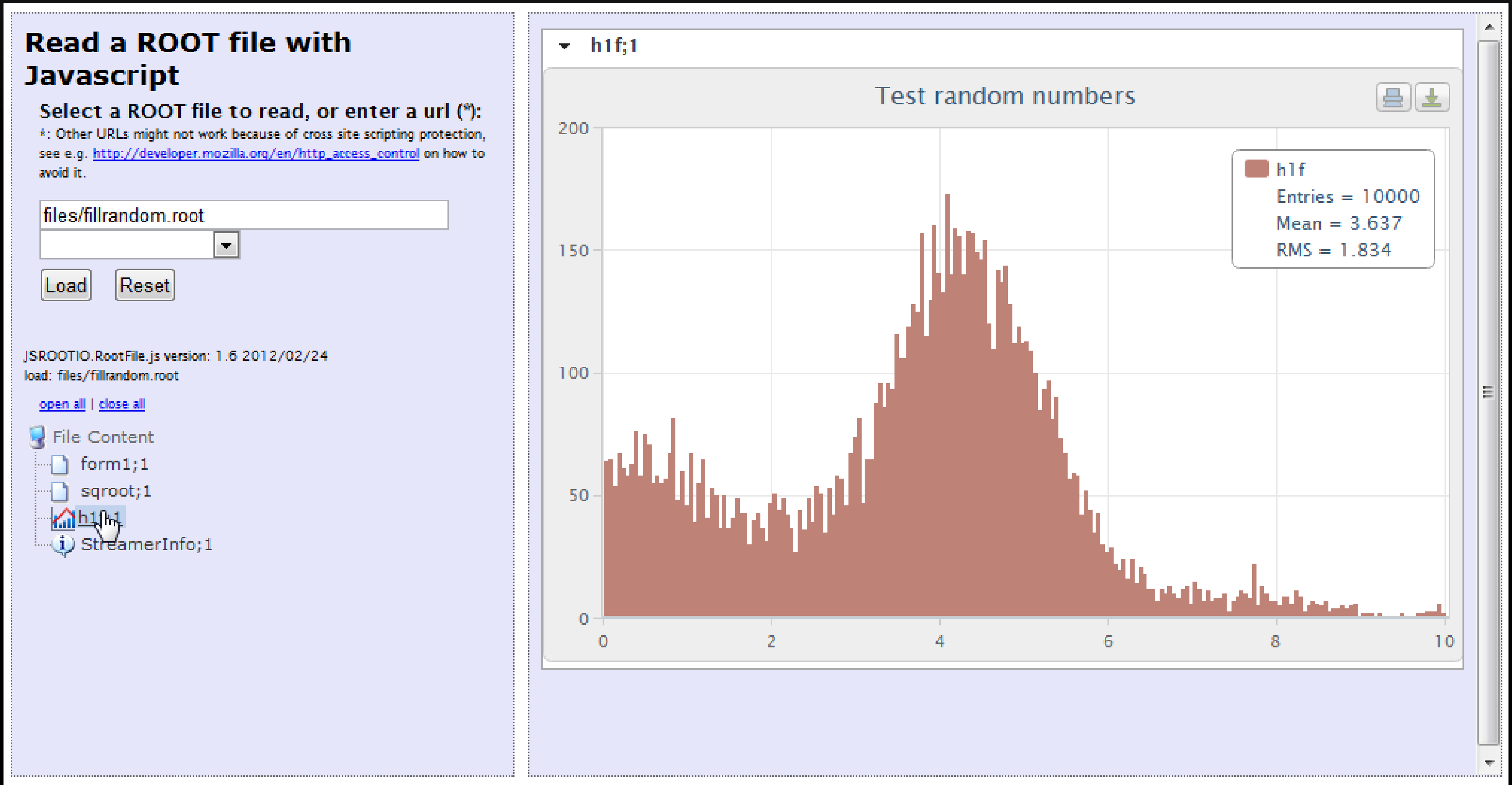
- A JavaScript charting library (HighCharts) is used to display the histograms and graphs (http://www.highcharts.com)
- It is released under the Creative Commons Attribution-Non Commercial 3.0 License, allowing to adapt it to ROOT's needs
- Some missing functionalities (e.g. error bars, lego plots) have to be implemented
- A summer student will start to work on it

DISPLAYING OBJECTS



Traditional visualization of a local ROOT file in the ROOT browser

DISPLAYING OBJECTS (CONT.)



Read a ROOT file with Javascript

Select a ROOT file to read, or enter a url (*):

*: Other URLs might not work because of cross site scripting protection, see e.g. http://developer.mozilla.org/en/http_access_control on how to avoid it.

files/fillrandom.root

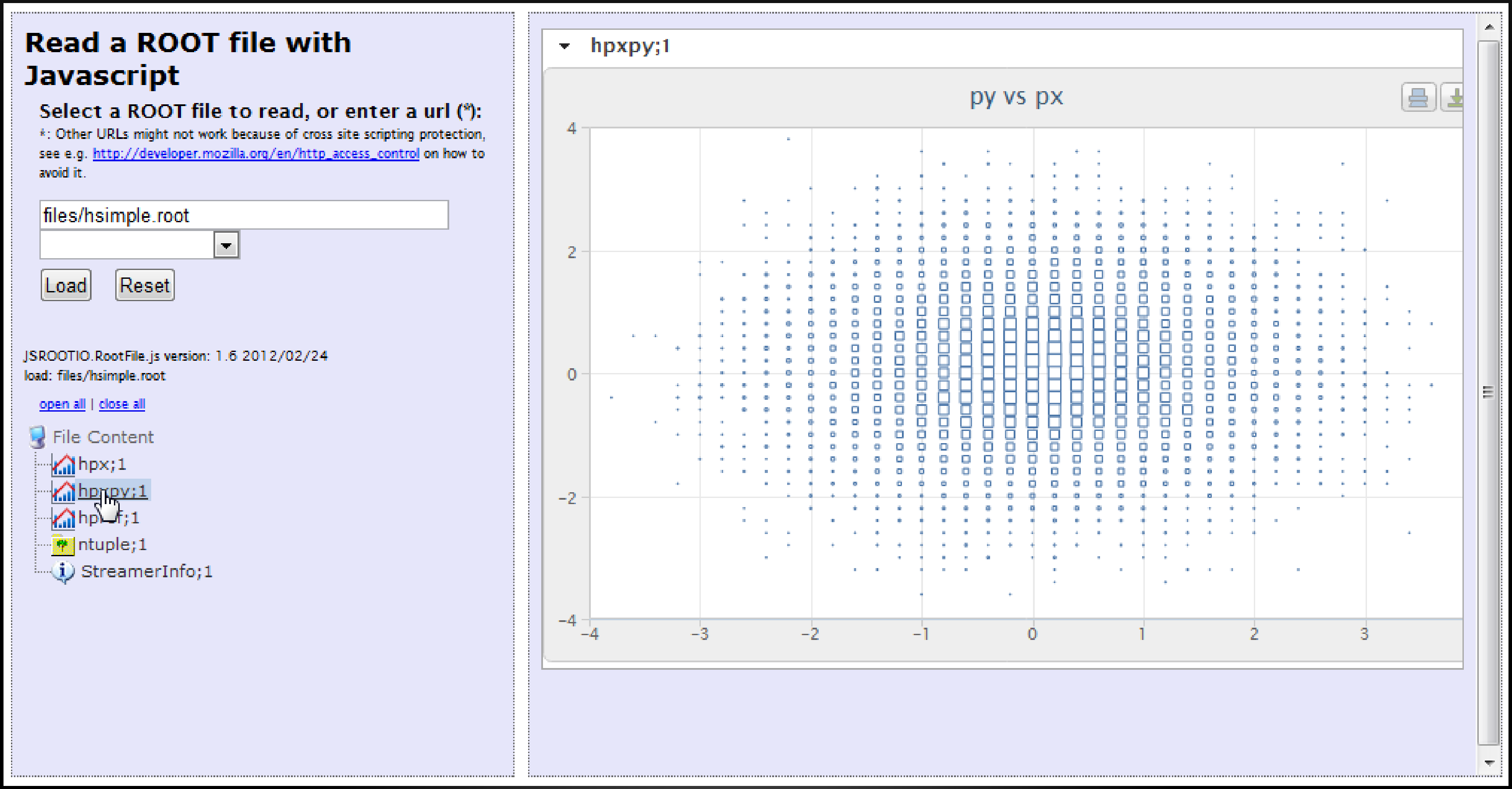
Load Reset

JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/fillrandom.root

open all | close all

JSROOTIO.js visualization of identical histogram

DISPLAYING OBJECTS (CONT.)



Read a ROOT file with Javascript

Select a ROOT file to read, or enter a url (*):

*: Other URLs might not work because of cross site scripting protection, see e.g. http://developer.mozilla.org/en/http_access_control on how to avoid it.

files/hsimple.root

Load Reset

JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/hsimple.root

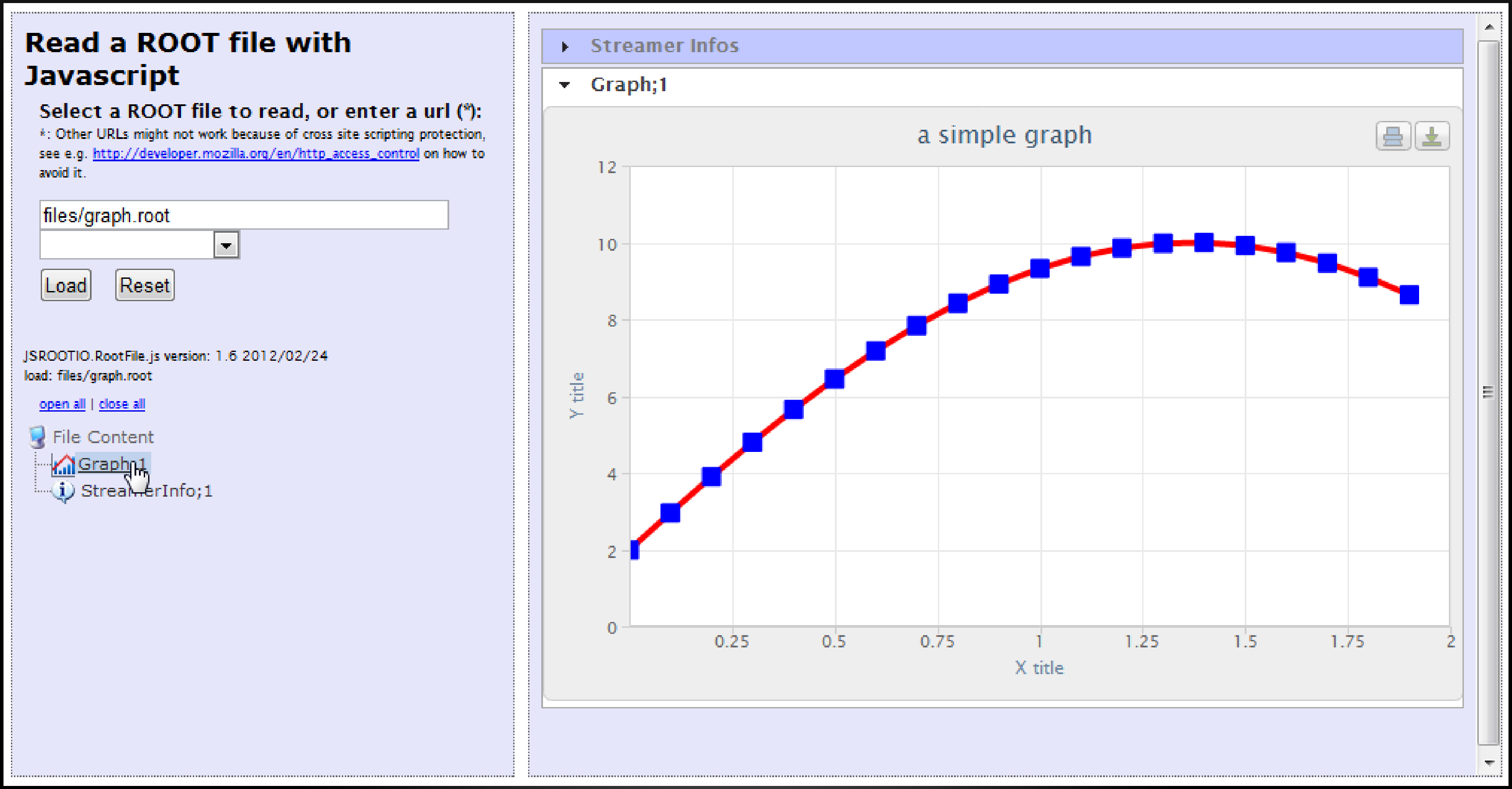
open all | close all

- File Content
  - hpx;1
  - hpxpy;1
  - hprof;1
  - ntuple;1
  - StreamerInfo;1

hpxpy;1

Displaying a TH2F (“BOX” plot only) from hsimple.root

DISPLAYING OBJECTS (CONT.)



Read a ROOT file with Javascript

Select a ROOT file to read, or enter a url (*):

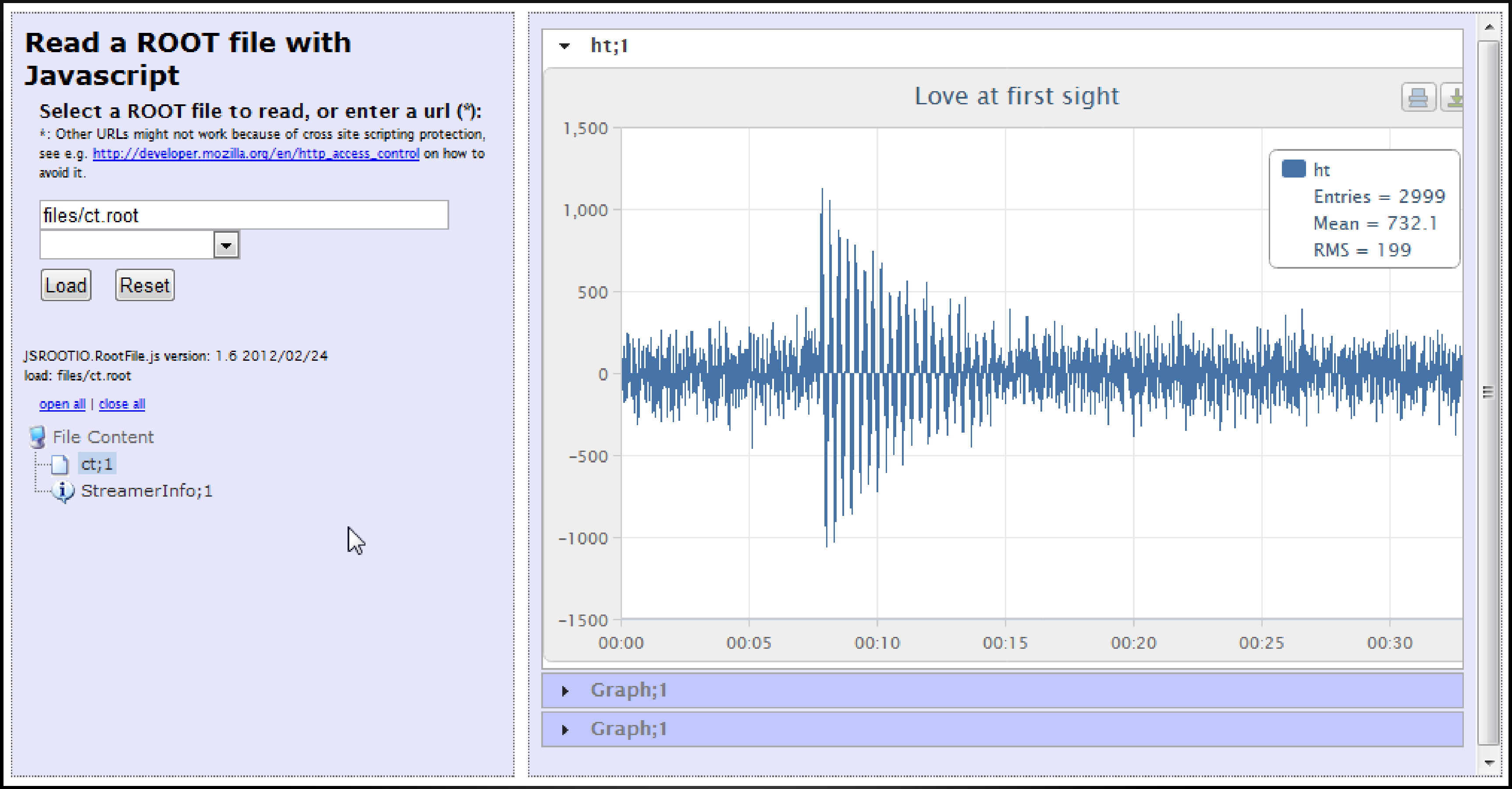
*: Other URLs might not work because of cross site scripting protection, see e.g. http://developer.mozilla.org/en/http_access_control on how to avoid it.

files/graph.root

JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/graph.root

Displaying a simple TGraph

DISPLAYING OBJECTS (CONT.)



Read a ROOT file with Javascript

Select a ROOT file to read, or enter a url (*):

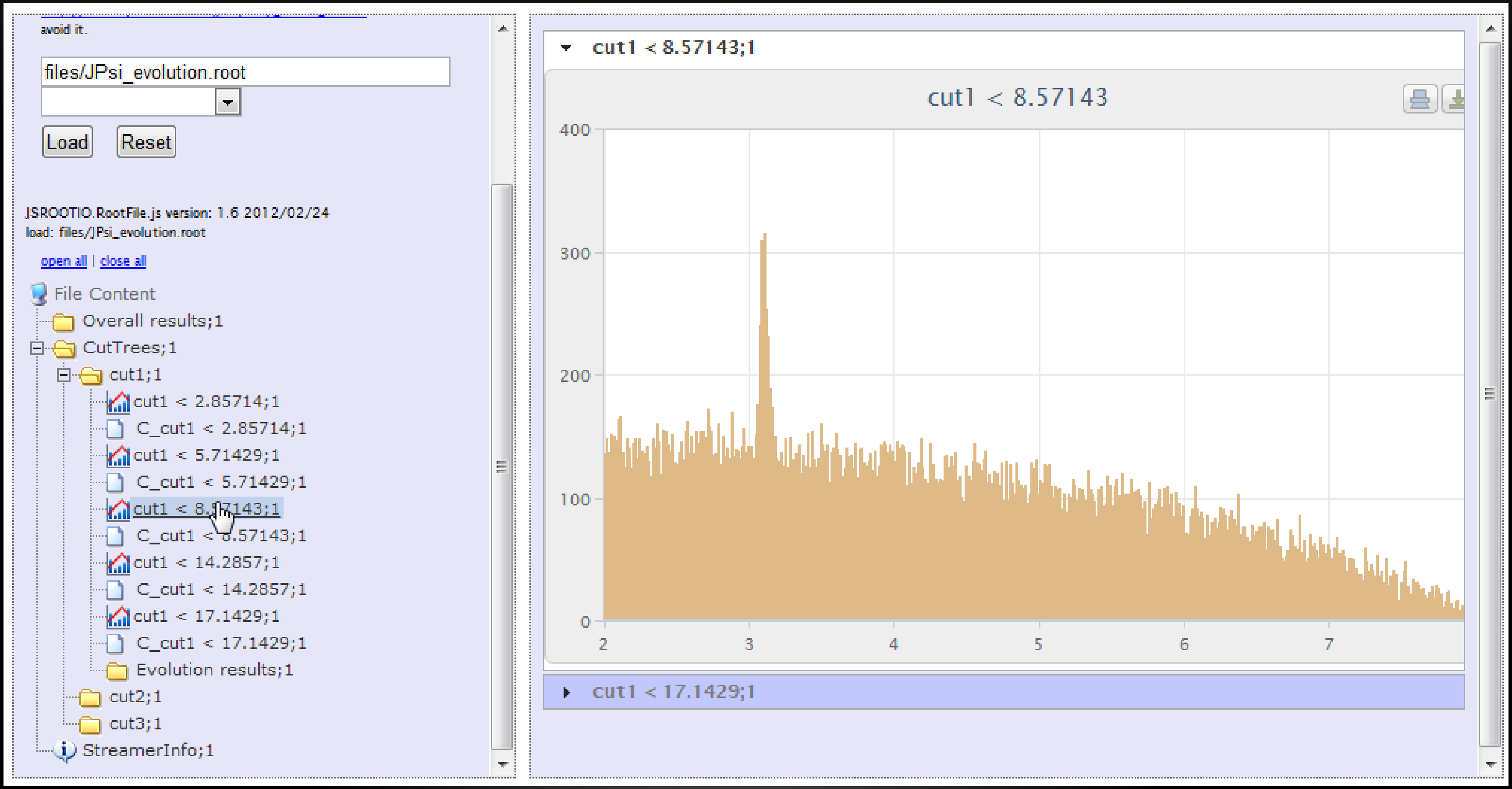
*: Other URLs might not work because of cross site scripting protection, see e.g. http://developer.mozilla.org/en/http_access_control on how to avoid it.

files/ct.root

JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/ct.root

Displaying the content of a TCanvas

DIRECTORY NAVIGATION



avoid it.

files/JPsi_evolution.root

JSROOTIO.RootFile.js version: 1.6 2012/02/24
load: files/JPsi_evolution.root

HOW TO USE IT?

- Simply copy the ROOT file(s) anywhere on the web
- Create a simple html page next to the files
  - Only two lines have to be added in the <head>

```
<head>
   <title>Read a ROOT file in JavaScript (Demonstration)</title>
   <link rel="stylesheet" type="text/css" href="style/JSRootInterface.css" />
   <script type="text/javascript" src="scripts/JSRootInterface.js"></script>
</head>
```

SIMPLE HTML EXAMPLE

- And a few lines in the <body>. Here is a complete example:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
   <head>
      <title>Read a ROOT file in Javascript (Demonstration)</title>
      <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
      <link rel="stylesheet" type="text/css" href="style/JSRootInterface.css" />
      <script type="text/javascript" src="scripts/JSRootInterface.js"></script>
   </head>
   <body onload="BuildSimpleGUI()">
      <div id="simpleGUI" files="file_1.root;file_2.root;file_n.root;"></div>
   </body>
</html>
```

JAVASCRIPT API

- Offering a very simple API:

```
f = new JSROOTIO.RootFile(url);
histo = f.ReadHistogram(histo_name);
if (typeof(histo) != 'undefined')
   displayHistogram(histo);
```

- But could be internally complex:

```
clRef = streamerInfo.ReadClass(str, o);
histo = eval('new JSROOTIO.' + clRef['name'] + '()');
if (typeof histo != 'undefined' &&
    typeof histo.Streamer == 'function')
   histo.Streamer(str, o);
```

AVAILABILITY

- The source code is available in svn: http://root.cern.ch/svn/root/branches/dev/bellenot/JSRootIO
- Remaining issues (non exhaustive list)
  - Doesn’t work on Android prior to version 4.0 (doesn't allow byte range HTTP requests)
  - Reading ROOT geometries is not implemented
  - Missing option to superimpose (overlay) histograms
  - lzma compression is not supported

WHAT NEXT

- Finalize the automatic streaming
- Implement custom streamers
- Investigate lzma decompression
- Implement missing parts of graphics in HighCharts and/or implement them in our own graphic library
- Possibly use WebGL for displaying 2D/3D objects (e.g. TH2, TH3, ROOT geometries)

CONCLUSION

- We are still in an early stage, and there is still quite some work in front of us
- Already working and usable, thanks to very valuable feedback from early users
- More features are coming
- Feel free to try and to send feedback & requests

And don’t forget to pass by our (CERN PH-SFT) booth, located at the 4th floor, here in Kimmel, to see a live demo or any other amazing development done in our group!