ROOT I/O in JavaScript
Browsing ROOT Files on the Web

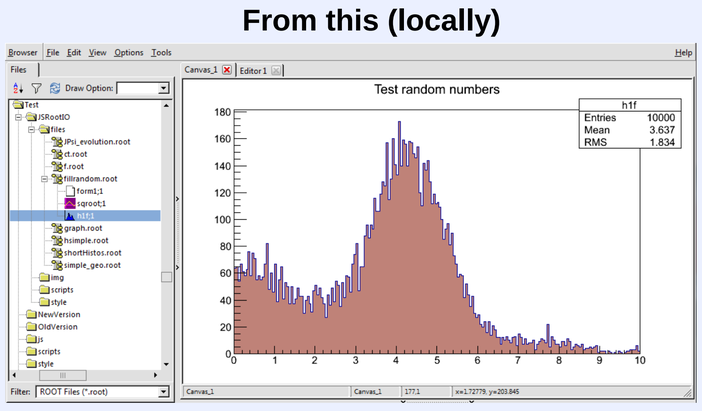
B. Bellenot [1], S. Linev [2]

[1] CERN, PH-SFT – European Organization for Nuclear Research, Geneva, Switzerland

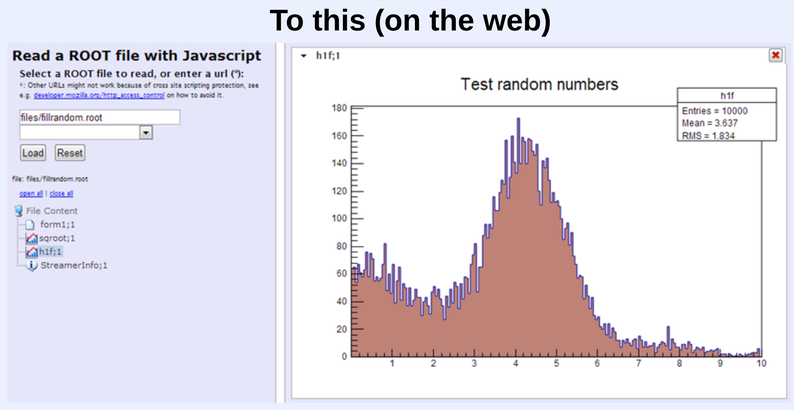
[2] GSI, CS-EE - GSI Helmholtzzentrum für Schwerionenforschung GmbH, Darmstadt, Germany

Browsing ROOT files

From this (locally)



To this (on the web)

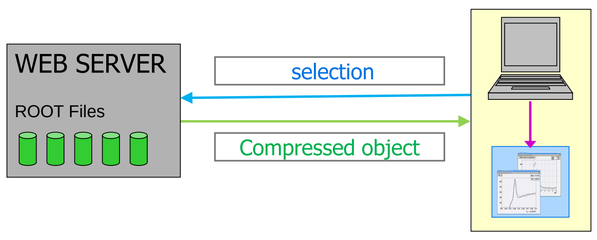


Features
- Uses JavaScript
- Load on demand
- Supported objects:
  - TH1
  - TH2
  - TH3
  - TGraph
  - TCanvas
  - …

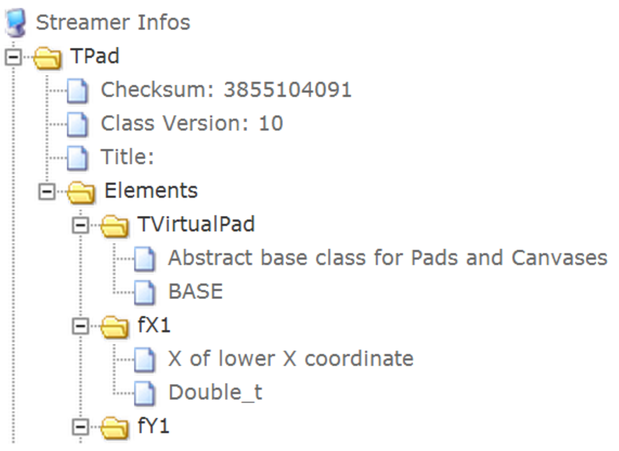
When opening a file:
- Read the list of streamer info
- Read the list of keys and display them in a list tree
- When the user select an item in the list tree (and only then)
- Read the compressed buffer from the file
- Inflate the buffer
- Decode (decipher) the object from the inflated buffer using its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

- 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
- Compressed (zipped) objects are in binary format
- Binary data is stored in a JavaScript string

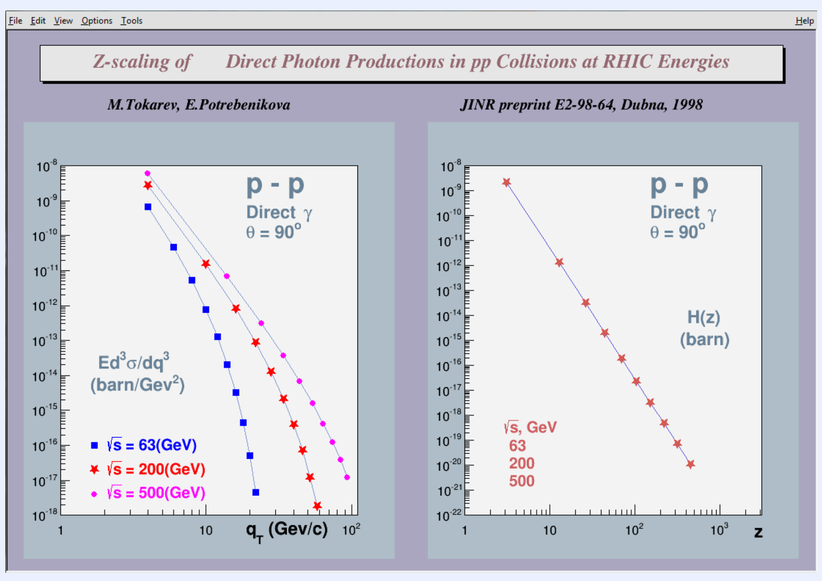
- One very nice feature of JavaScript is the possibility to dynamically (at runtime) create classes
- Allowed to implement dynamic streamers (automatically generated from the streamer info)
- Allows to potentially read any object from a ROOT file, as soon as we can read the streamer info of its class



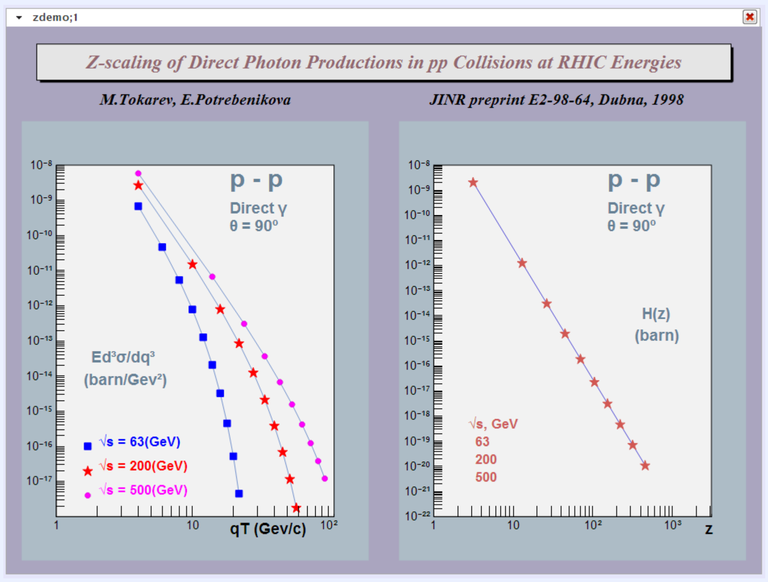
Data flow



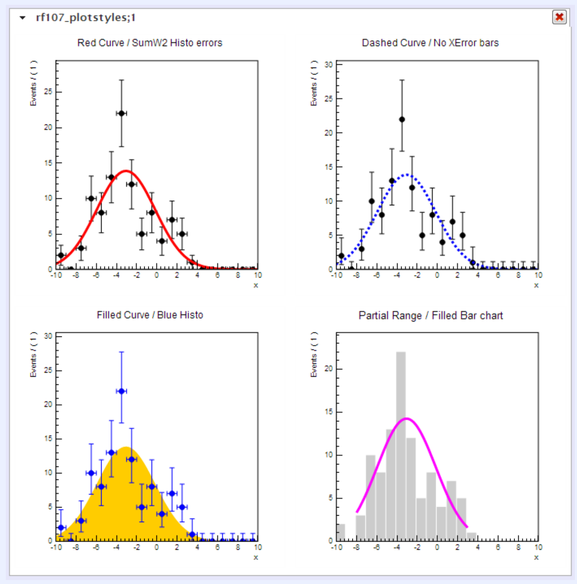
Example of Streamer Info Visualization



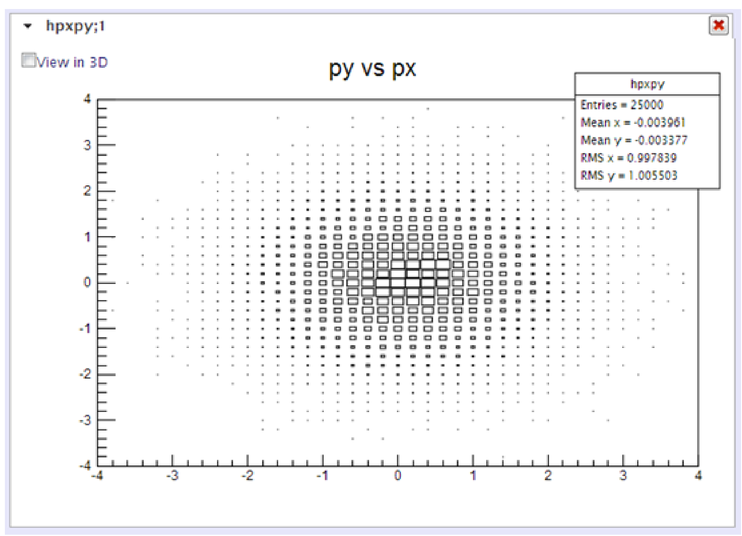
Example of the zdemo.C tutorial of ROOT



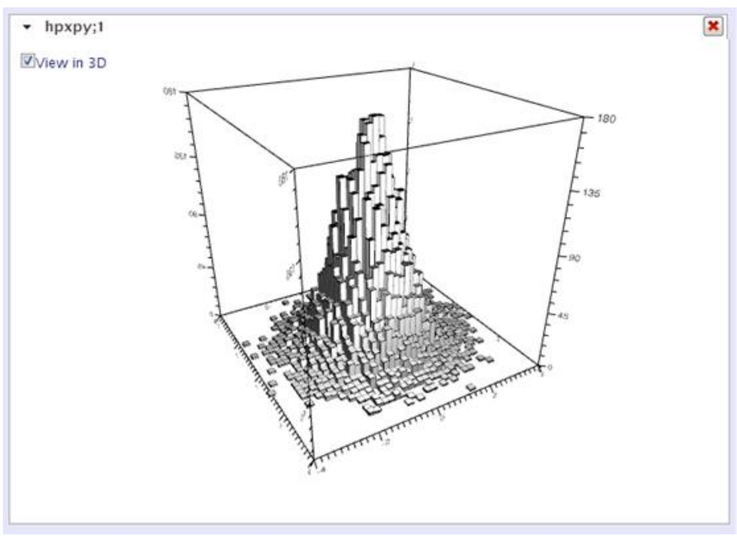
Same example, displayed in a web browser



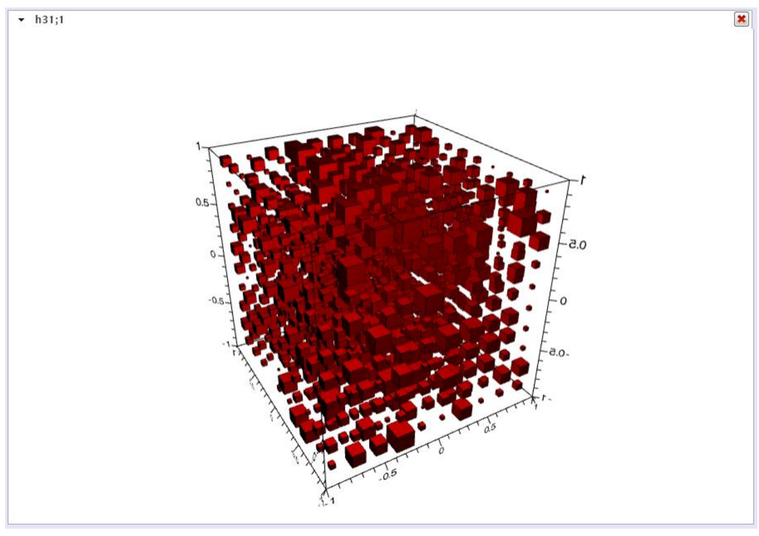
Different roofit plot styles…



2D histogram (TH2), drawn with the BOX option (default)



Same histogram, displayed with 3D option (“LEGO”), using WebGL when available



3D Histogram (TH3), using WebGL when available

- The complete source code is available in git:
  git clone http://root.cern.ch/git/rootjs.git
- 3D graphics uses WebGL technology when available (browser and platform dependent)

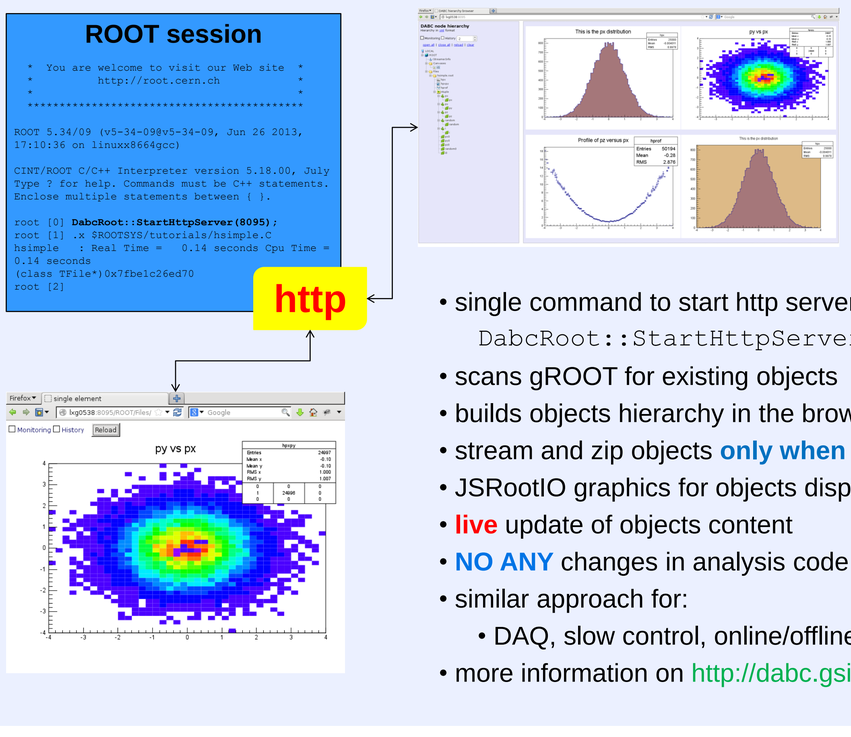
Easy to use!

- 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>
  - 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="http://root.cern.ch/js/style/JSRootInterface.css" />
      <script type="text/javascript" src="http://root.cern.ch/js/scripts/JSRootInterface.js"></script>
   </head>
   <body onload="BuildSimpleGUI()">
      <div id="simpleGUI" files="file_1.root;file_2.root;file_n.root;"></div>
   </body>
</html>
```

  - Including css and js directly from the root web site keeps you up to date with the latest version

Monitoring of running ROOT application



- single command to start http server
  `DabcRoot::StartHttpServer(8095);`
- scans gROOT for existing objects
- builds objects hierarchy in the browser
- stream and zip objects **only when** requested
- JSRootIO graphics for objects display
- **live** update of objects content
- **NO ANY** changes in analysis code
- similar approach for:
  - DAQ, slow control, online/offline analysis
- more information on http://dabc.gsi.de

For more information see: http://root.cern.ch
For any questions please use following address: rootdev@root.cern.ch