# THttpServer and JavaScript in ROOT*

*J. Adamczewski-Musch*[1], *B. Bellenot*[2], *S. Linev*[1], *and the FAIR@GSI project*[1]

[1]GSI, Darmstadt, Germany, [2]CERN, Geneva, Switzerland

## Introduction

The ROOT [1] software framework is widely used in most HEP experiments. ROOT is also often applied for monitoring and control at different stages of data taking and online analysis. Many such online tasks could be solved with web technologies – http protocols for data exchange, and powerful JavaScript/HTML graphics to implement user interfaces.

## JavaScript ROOT

The prominent functionality of JavaScript ROOT (JSROOT) library is the ability to read binary ROOT files and provide interactive ROOT-like graphics in web browsers (Figure 1). Many significant changes and improvements have been implemented in 2014 compared to original JSRootIO [2] code.

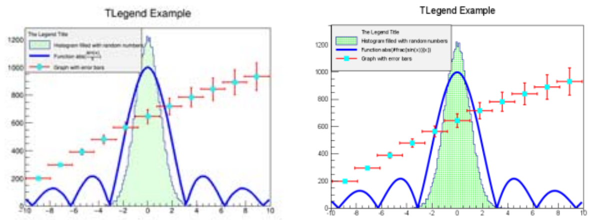

Figure 1: Same canvas, displayed with native ROOT graphics (left) and JSROOT (right)

The I/O part of JSROOT has been organized modular and structurally similar to native ROOT I/O functionality. Most objects stored in binary ROOT files now can be read from the web-browser, using meta-information provided within ROOT files (so-called “streamer-infos”). Several special cases of custom streamers (like TCanvas or TList) are treated in a central place, making it much easier to maintain and extend functionality in the future. It is possible to display and overlay data from different ROOT files in the same browser window.

The graphical part of JSROOT code has been fully rewritten and decoupled from I/O part of the JSROOT library. The main focus was put on flexibility – now JSROOT graphics can be inserted in any webpage, and one can update graphics interactively. Many new features have been implemented, e.g. context menu, statbox update, comfort zooming.

JSROOT code, documentation and examples are available online [3] and with ROOT distribution.

## THttpServer class

The new THttpServer class of ROOT implements http protocol by means of embeddable Civetweb [4] server. With simple http requests syntax one could obtain object data from application in different form. (binary, xml, json, png). The TBufferJSON class has been developed to convert ROOT objects into JSON (JavaScript Object Notation), which can be directly evaluated in web browsers.

The user interface for THttpServer has been implemented with JSROOT (Figure 2). One could browse, display, and monitor objects registered to the server. In addition one could execute registered commands (like start/stop analysis or clear histograms) in the server application. There is also the possibility to perform TTree::Draw() on the server and display results in the browser.

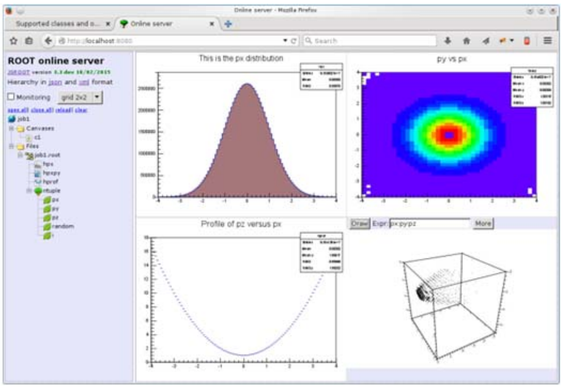

Figure 2. Browser with objects available from httpserver.C macro. The objects hierarchy is on the left side, several displayed objects are on the right. Also tree draw functionality is shown.

Source code, documentation, and several examples for THttpServer are included into the ROOT distribution.

## Conclusions and outlook

THttpServer together with JSROOT provide a powerful tool set to create ROOT-based online applications. JSROOT functionality is also used to implement web interface for DABC, MBS and Go4 frameworks [5].

## References

[1] ROOT project homepage http://root.cern.ch

[2] B.Bellenot, S.Linev, “ROOT I/O in JavaScript”, Journal of .Physics: Conference Series, vol 513, Proceedings of CHEP 2013, http://iopscience.iop.org/1742-6596/513

[3] JSROOT homepage https://root.cern.ch/js/jsroot.html

[4] Civetweb embeddable webserver, project homepage https://github.com/bel2125/civetweb

[5] S. Linev, “Web interface in DABC”, GSI scientific report 2013, doi:10.15120/GR-2014-1-FG-CS-12

* PSP codes: SFRS 2.4, ILIMA 1.2.2.4, R3B 1.2.5.1.4, 1.2.6.6, 1.2.5.1.2, HADES 1.1.2.4

DOI:10.15120/GR-2015-1-FG-CS-04