



rootJS - Specification

PSE - Software Engineering Practice

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart | December 15, 2015

STEINBUCH CENTER FOR COMPUTING



About PSE



Praxis der Softwareentwicklung(PSE) = Software Engineering Practice

- Waterfall model
 - Planing/definition

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart - rootJS

Functional specification



Purpose



Node.js bindings for ROOT

- be able to write ROOT code in Node.js programs
- integrate ROOT into Node.js based web applications

Product Environment Product data Product interface and functions Scenarios Use Cases

Product usage

Required Criteria



The bindings must

- work on Linux
- allow the user to interact with any ROOT class from the Node.js JavaScript interpreter
- accept C++ code for just-in-time compilation
- update dynamically following changes to C++ internals
- provide asynchronous wrappers for common I/O operations (i.e. file and tree access)

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart - rootJS

Optional Criteria



The bindings should

- support the streaming of data in JavaScript Object Notation (JSON) format compatible with JavaScript ROOT
- implement a web server based on Node.js to mimic the function of the ROOT HTTP server

Product Environment Product data Product interface and functions Scenarios Use Cases

 work OS independent (i.e. support Mac OS X, Linux operating systems)



System Mode

Limiting criteria



The bindings should not

add any extending functionality to the existing ROOT framework

Product Environment Product data Product interface and functions Scenarios Use Cases

necessarily support previous/future ROOT versions

Product usage



rootJS will be used to create web-applications that can:

- Expose processed data (that might otherwise be hard to access) and then visualize it locally
- Interact with data both stored somewhere accessible for the server or streamed via remote procedure call (RPC)
- Run on any platform that supports a browser



Audience



Most users of rootJS will be used to working in Linux and with web servers. At the very least, they will be able to install ROOT and also be proficient in programming languages like JavaScript and C++.

- Scientists (e.g. particle physicists)
- Researchers
- Web-developers interested in creating applications based on ROOT

Operating conditions



- rootJS will be used on servers that run ROOT and have access to the required data sources.
- As ROOT 6 currently runs on Linux and OS X only, usage of the bindings is limited to those platforms.

ROOT

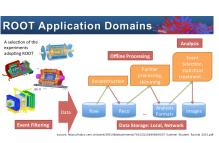
Purpose

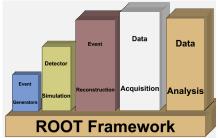


- process and visualize large amounts of scientific data (CERN)
- features a C++ interpreter (CLING) i.e. used for rapid and efficient prototyping

Product Environment Product data Product interface and functions

persistency mechanism for C++ objects







Product usage

System Mode

Node.js



- open source runtime environment
 - develop server side web applications
 - act as a stand alone web server





Node.js



- open source runtime environment
 - develop server side web applications
 - act as a stand alone web server.
- Google V8 engine to execute JavaScript code



Product Environment Product data Product interface and functions



System Mode

Node.js



- open source runtime environment
 - develop server side web applications
 - act as a stand alone web server
- Google V8 engine to execute JavaScript code
- rootJS bindings realized as native Node.js module written in C++



Product Environment Product data Product interface and functions



System Mode

Hardware



<ロ > < 部 > < 差 > < 差 > 差 ≥ f を の の ○

December 15, 2015

System Mode

12/21

PSE Purpose Product usage Product Environment Product data Product interface and functions Scenarios Use Cases

Product data



The following data will be stored by the rootJS bindings

- All ROOT classes and methods as they dynamically mapped to their JavaScript equivalents
- **ROOT** environment state
- Application context is derived from TApplication
- Map of v8::handles 2 identified by the address of ROOT objects

Product interface and functions

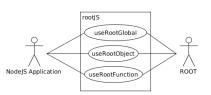


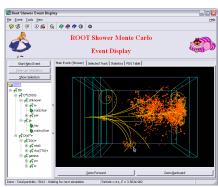


December 15, 2015

Event Viewer









Product usage

Purpose



Initialization



- Expose all
 - Global variables
 - Global functions
 - Classes



PSE Purpose Product usage Product Environment Product data Product interface and functions Scenarios Use Cases

Initialization



- Expose all
 - Global variables
 - Global functions
 - Classes
- Each are bound to corresponding proxy methods
- An object which members are the exposed features is beeing passed to node

Product Environment Product data Product interface and functions Scenarios Use Cases



Initialization



- Expose all
 - Global variables
 - Global functions
 - Classes
- Each are bound to corresponding proxy methods
- An object which members are the exposed features is beeing passed to node

Names

- Functions and classes have the same name as in Root
- Global variables can be called using Get[Variable] and Set[Variable] methods



Call a feature



All features in node are mapped to a proxy method that will be called

Product Environment Product data Product interface and functions Scenarios Use Cases



System Mode

Call a feature



- All features in node are mapped to a proxy method that will be called
- The proxy method will eventually call a root function and pass the result to our ObjectFactory

Call a feature



- All features in node are mapped to a proxy method that will be called
- The proxy method will eventually call a root function and pass the result to our ObjectFactory
- By looking at the object type an corresponding v8::Handle will be generated and returned to node

Product Environment Product data Product interface and functions Scenarios Use Cases

If the result is an object this will be done recursively



Test Cases



< □ > < 圖 > < 불 > < 불 > 필급 외익()

December 15, 2015

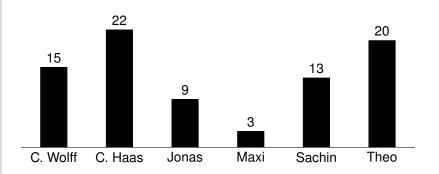
System Mode

19/21

PSE Purpose Product usage Product Environment Product data Product interface and functions Scenarios Use Cases

Merges

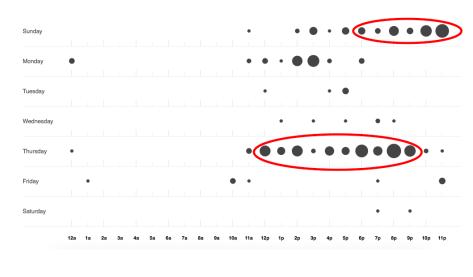






Punchcard





PSE Purpose Product usage Product Environment Product data Product interface and functions Scenarios Use Cases System Mode 00 00

December 15, 2015

References I



- CERN. ROOT application domains. Dec. 2015. URL: https://root.cern.ch/application-domains.
- CERN. ROOT Shower Event Display. Dec. 2015. URL: https://root.cern.ch/rootshower00png.
- exortech. v8 logo. Dec. 2015. URL: https://github.com/exortech/presentations/blob/master/promise_of_node/img/v8.png.
- Node.js logo. Dec. 2015. URL: https://nodejs.org/static/images/logos/nodejs-light.eps.
- Danilo Piparo and Olivier Couet. ROOT Tutorial for Summer Students. Dec. 2015. URL: https://indico.cern.ch/event/395198/attachments/

791523/1084984/ROOT_Summer_Student_Tutorial_2015.pdf.



References II



Boris Vacher. npm logo. Dec. 2015. URL:

https://commons.wikimedia.org/wiki/File:Npm-logo.svg.