Karlsruhe Institute of Technology

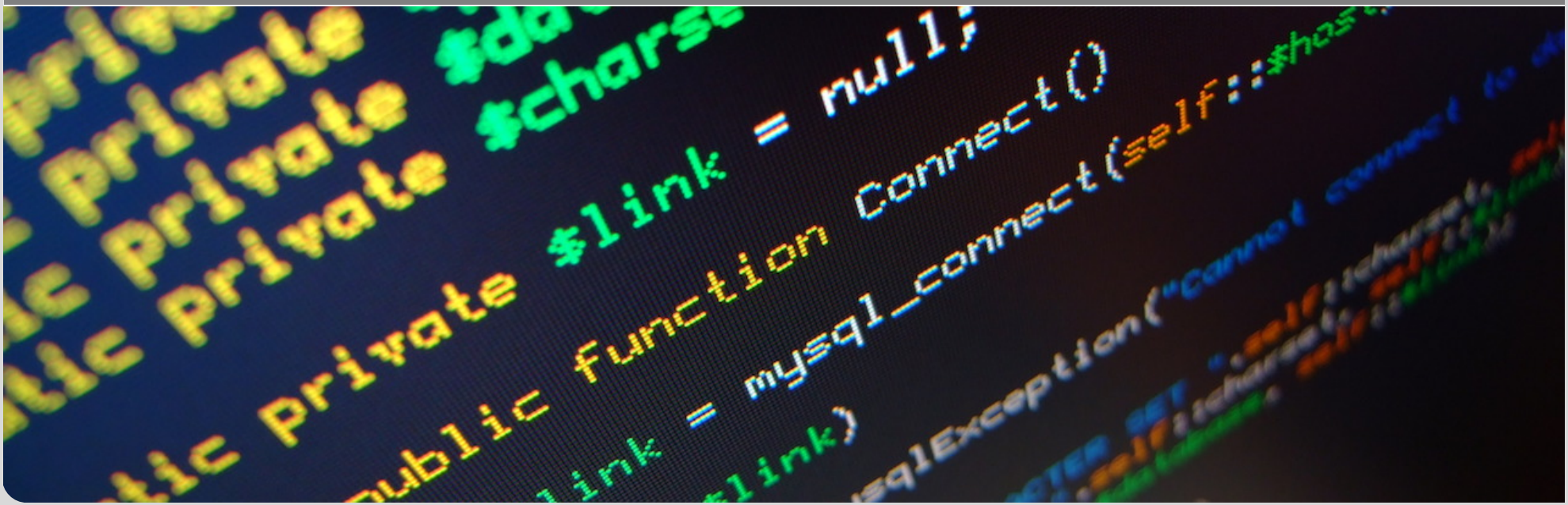
Steinbuch Centre for Computing

rootJS - results of the implementation phase

Software Engineering Practice - Winter Term 2015/16

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart | February 12, 2016

STEINBUCH CENTER FOR COMPUTING



Outline

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)

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)

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)

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)

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)

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) ✓

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

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

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 ✓
- Work OS independent (i.e. support Mac OS X, Linux operating systems)

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 ✓
- Work OS independent (i.e. support Mac OS X, Linux operating systems) ✓

Limiting Criteria

The bindings should not

- Add any extending functionality to the existing ROOT framework
- Necessarily support previous/future ROOT versions

Limiting Criteria

The bindings should not

- Add any extending functionality to the existing ROOT framework ✓
- Necessarily support previous/future ROOT versions

Limiting Criteria

The bindings should not

- Add any extending functionality to the existing ROOT framework ✓
- Necessarily support previous/future ROOT versions ✓

Changes in design

- We added a helper class for asynchronous operations, ’AsyncRunner’
- We added MetaInfo to encapsulate differences between TGlobal and TDataMember