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 14, 2016

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)

Fullfilled criteria
○

Changes in design
○○○

Workflow

Statistics

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   3/9

Required criteria

The bindings must:

- Work on Linux $\checkmark$
- 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)

Fullfilled criteria   Changes in design   Workflow   Statistics
○   ○○○
C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   3/9

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)

Fullfilled criteria

Changes in design

Workflow

Statistics

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

February 14, 2016

3/9

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)

Fullfilled criteria
○

Changes in design
○○○

Workflow

Statistics

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

February 14, 2016

3/9

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) ✓

Fullfilled criteria
○

Changes in design
○○○

Workflow

Statistics

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   3/9

AsyncRunner

- We added a helper class for asynchronous operations, 'AsyncRunner'
- Uses libuv internally
- TThread doesn't allow communtication between node threads

Fullfilled criteria
○

Changes in design
●○○

Workflow

Statistics

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   4/9

MetaInfo

- We added a helper class to encapsulate differences between TGlobals and TDataMember
- TGlobal has a getAdress() method to get the absolute adress, TDataMember only getOffset()
- Provide interface for return types

Fullfilled criteria
○

Changes in design
○●○

Workflow

Statistics

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   5/9

Other changes

- Dynamic library loader: ROOT doesn't load enough libraries on its own
- removed get exports

Fullfilled criteria

Changes in design

Workflow

Statistics

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

February 14, 2016

6/9

Workflow

- Test-driven development
- We couldn't use Travis CI, the build of ROOT always timed out
- So we used Jenkins instead
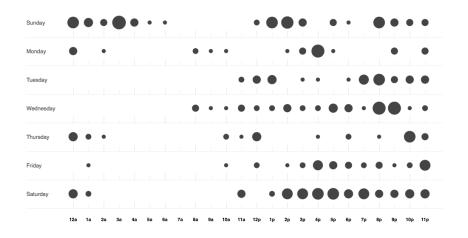- Implement new features in branches first, then merge later

Fullfilled criteria
○

Changes in design
○○○

Workflow

Statistics

C. Wolff, M. Früh, S. Rajgopal, C. Haas, J. Schwabe, T. Beffart – rootJS - implementation   February 14, 2016   7/9

Statistics

- 3200+ LOC
- 400+ commits

Fullfilled criteria
○

Changes in design
○○○

Workflow

Statistics

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

February 14, 2016

8/9

Punch Card

Fullfilled criteria

Changes in design

Workflow

Statistics