Karlsruhe Institute of Technology

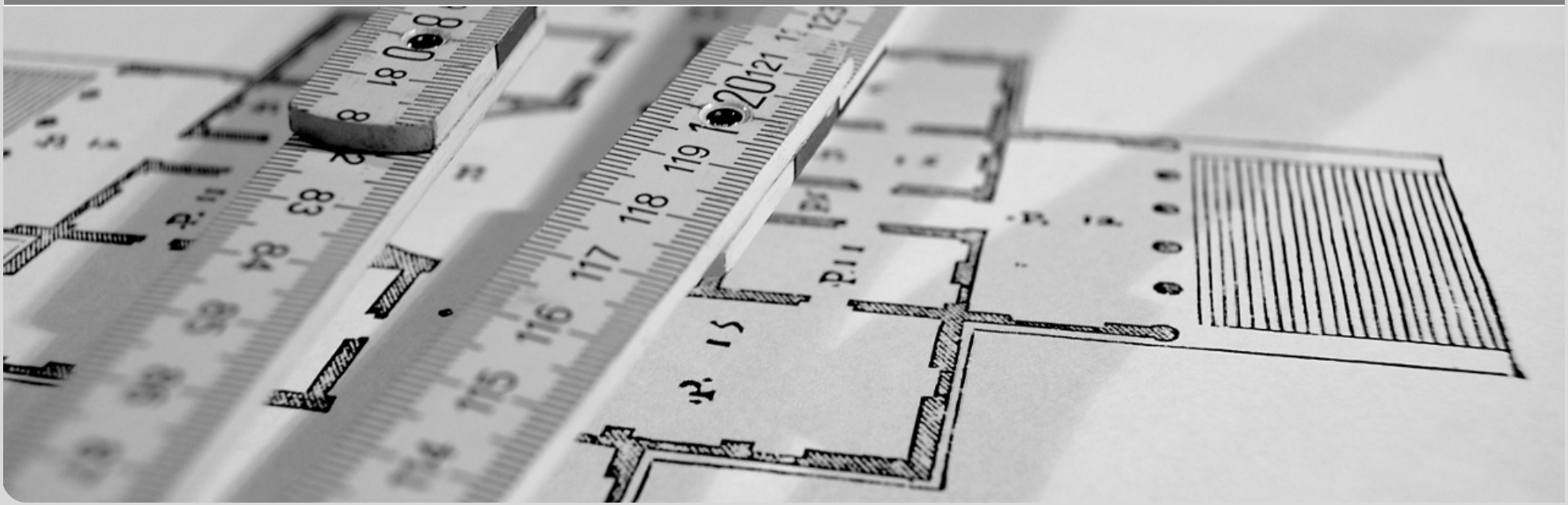
Software Design and Quality

# rootJS - Specification

PSE - Software Engineering Practice

Christoph Wolff, Maxi Früh, Sachin Rajgopal,
Christoph Haas, Jonas Schwabe, Theo Beffart | December 13, 2015

STEINBRUCH CENTER FOR COMPUTING



# Outline/Gliederung

# Software

## ROOT

- 
- 

## Node.js

- 
- 

# Software

## ROOT

- 
- 

## Node.js

- 
- 

# Hardware

# 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

# 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

# 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
- 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
  - If the result is an object this will be done recursively

# 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
  - If the result is an object this will be done recursively

# 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
  - If the result is an object this will be done recursively

# Example slide C

## Example 1

- Bullet point 1
- Bullet point 2
- ...

# Example slide C

## Example 1

- Bullet point 1
- Bullet point 2
- ...

# Example slide D

## Alert 1

- Bullet point 1
- Bullet point 2
- ...

# Example slide D

## Alert 1

- Bullet point 1
- Bullet point 2
- ...

# References I