JavaScript: One language to rule them all

Janosch Zbick

janosch.zbick@lnu.se

The Three Ages of JavaScript

- 1st Age (90's)
  - JavaScript was a simple client-side that was mainly used for:
    - Popups
    - Displaying text in the status bar
    - Effects on content (blinking / mouseover ...)
    - No DOM manipulation, no CSS, no support for regular expressions

Janosch Zbick
janosch.zbick@lnu.se

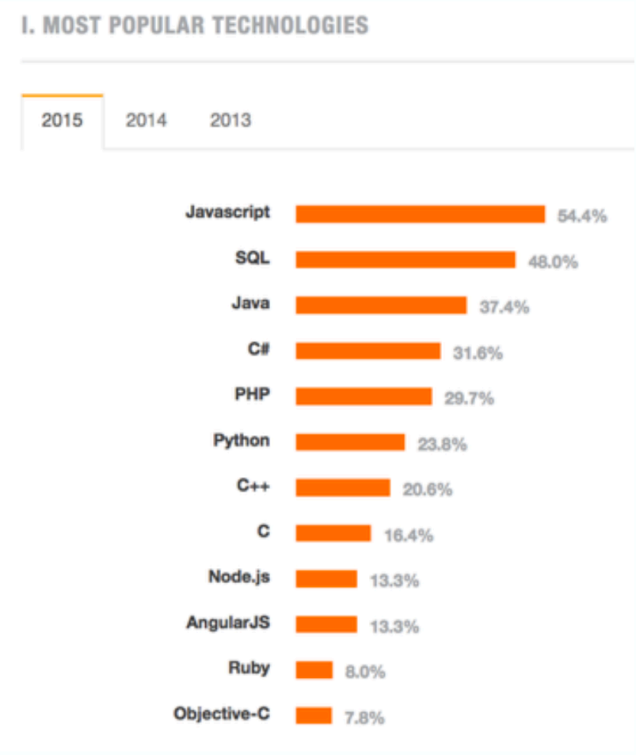
The Three Ages of JavaScrip

- 2nd Age (The 2000's)
  - Libraries like jQuery emerge
  - DOM manipulation gets introduced
  - Evolution of client-side applications
    - Rich Internet Applications like Google products are created (Google Mail, Google Maps, etc.)

Janosch Zbick
janosch.zbick@lnu.se

The Three Ages of JavaScript

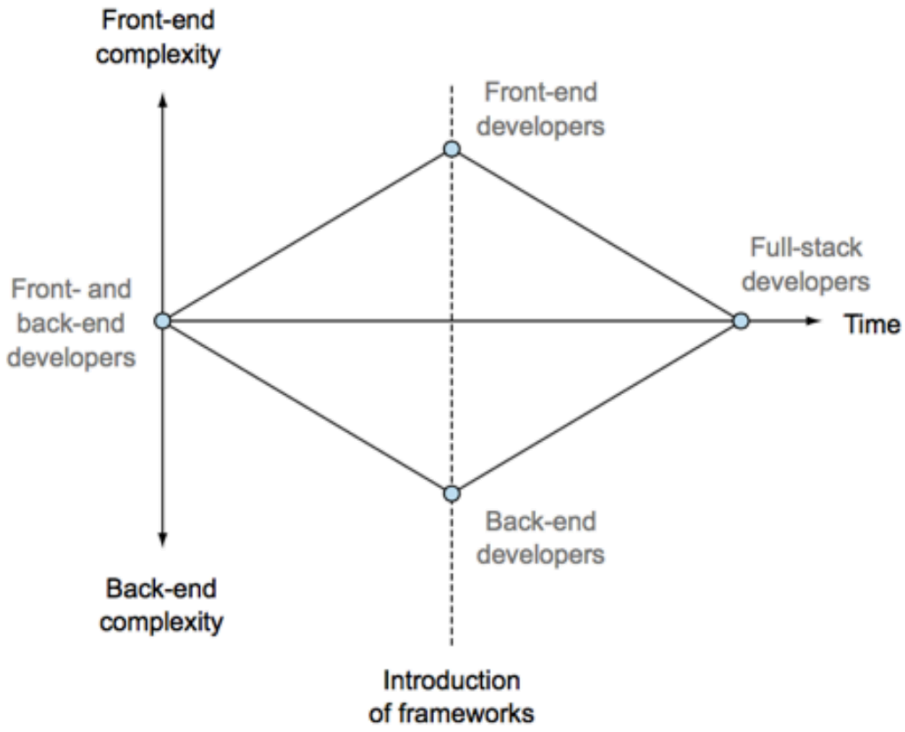
- 3rd Age (The 2010's)
  - Google introduces the very fast JavaScript execution engine (v8)
  - Now it is powerful enough to rival server-side languages like PHP, Ruby, Python, etc.
    - It is used beyond client-side application and server-side applications are now created with JS (i.e. Node.js)
  - The introduction of HTML5 standards like canvas allows JS to build powerful graphical applications



Janosch Zbick
janosch.zbick@lnu.se

One language to rule them all

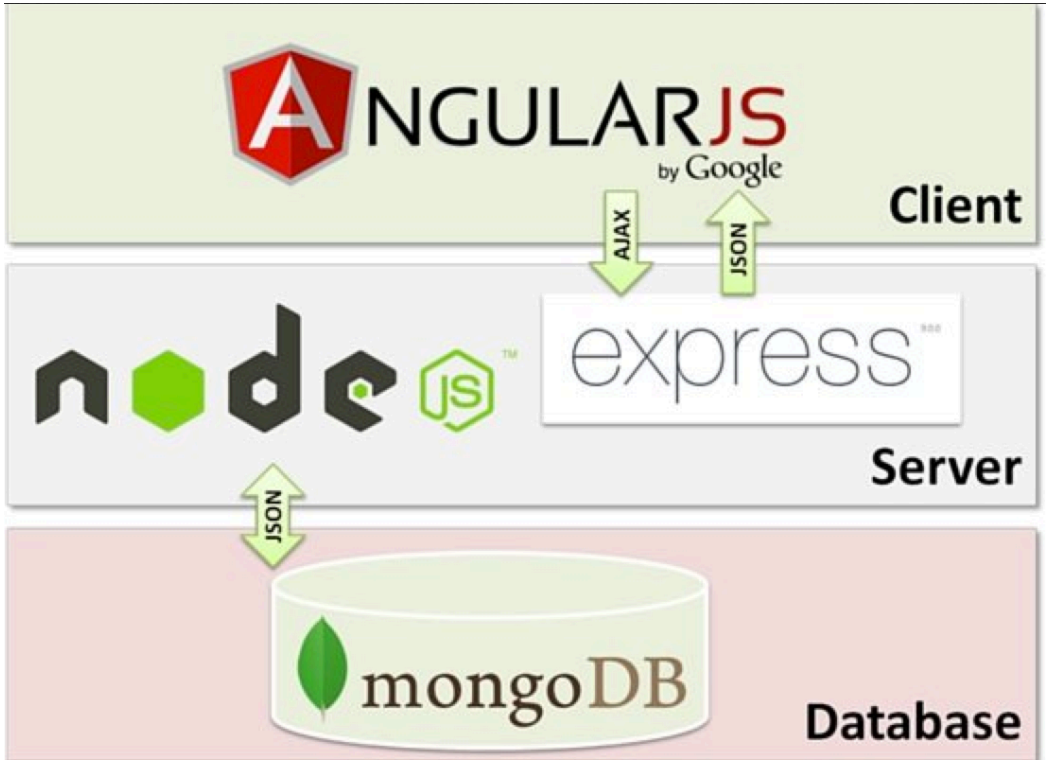
- For web development in all layers it is only necessary to master one programming language, both client and server side.
- Of course, you still need knowledge in HTML and CSS



Janosch Zbick
janosch.zbick@lnu.se

Full stack JavaScript

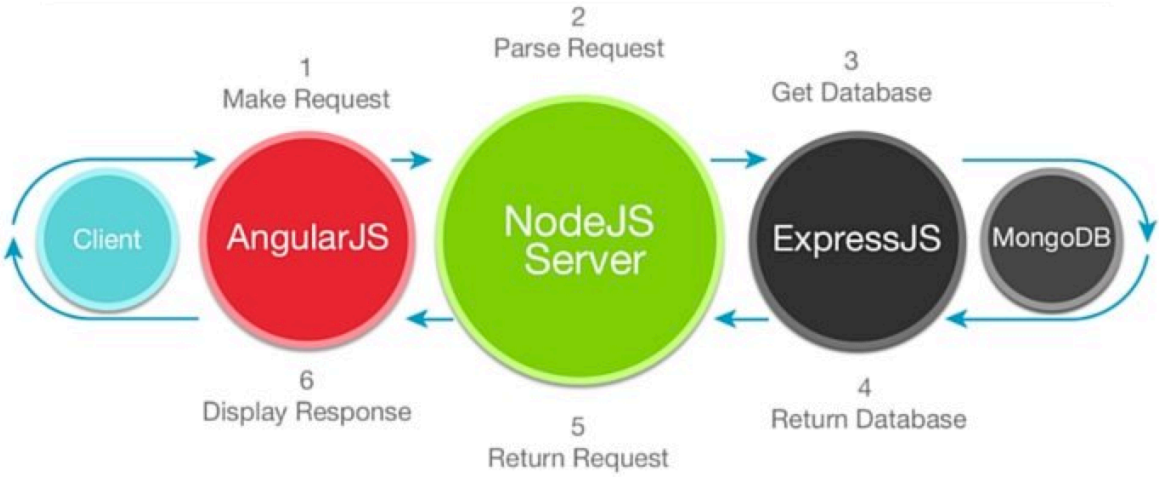
- What does it mean?
  - JavaScript in every layer
    - Front-end (client-side)
    - Back-end
      - Server-side
      - Database



Janosch Zbick
janosch.zbick@lnu.se

MEAN vs. LAMP

LAMP	MEAN
Linux	MongoDB
Apache	ExpressJS
MySQL	AngularJS
PHP (Perl, Python)	Node.js



Janosch Zbick
janosch.zbick@lnu.se

Node.js

- As mentioned before, server-side JavaScript
  - But why? (Besides “one language to rule them all”)
  - It was originally presented as one approach to solve the “C10K” problem
    - Handle more than 10.000 requests simultaneously
    - Node.js is one of many approaches (Nginx, Jetty, ...)

Janosch Zbick
janosch.zbick@lnu.se


Node.js – the C10K problem

- How?
  - Event-driven
    - Application flow is determined by events.
    - If you are familiar with JavaScript you might recall "`element.onclick`"
  - Non-blocking I/O model
  - Single threaded

Janosch Zbick
janosch.zbick@lnu.se


Node.js – single threaded

- What does this mean?
  - In contrast to other popular solutions (LAMP), everything runs in a single thread.
  - Building a web application with Apache means that a thread is created per request
  - The main paradigm while coding with Node.js is using “callbacks”

Janosch Zbick
janosch.zbick@lnu.se


Node.js – NPM

- The package Manager of Node.js: NPM
  - Node.js is just a skeleton, most of the time you want to install external packages that allows you to extend the functionalities (i.e. express)
  - Handles the dependencies of a Node.js project
    - The dependencies are stored and manages in the "`packages.json`" of a Node.js project
    - Install a new packages with "`npm install package-name -save`" (the -save flag adds the package automatically to your package.json)

Janosch Zbick
janosch.zbick@lnu.se

MongoDB

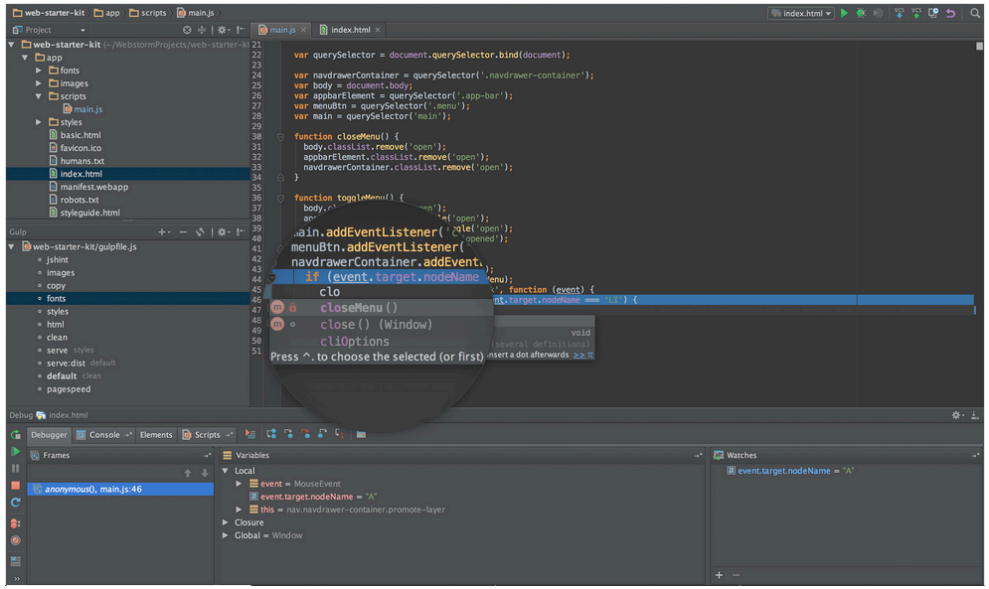
- Why MongoDB?
  - MongoDB is a non-relational database system that stores JSON-style documents
  - JSON documents are indexable and searchable by MongoDB
  - JSON is the natural communication object of JavaScript
  - Among XML, JSON is the most common interface “language” to communicate with other systems

  - Non-relational databases scale better than relational databases, therefore they make a good fit for web-applications

Janosch Zbick
janosch.zbick@lnu.se


Developing Environment

- Suggestion: WebStorm: https://www.jetbrains.com/webstorm/
  - Specialized on Web Applications based on JavaScript
  - Good integration of Node.js
  - Supports latest technologies
  - AngularJS / Node.js / ...
  - Debugger / Terminal / VCS
  - Cross-platform



  - Free for students:
  - https://www.jetbrains.com/student/

Janosch Zbick
janosch.zbick@lnu.se

Example

- Let’s have a look, how we easily create a well structured Node.js application with Express
  - Then, let’s have a look at an API created with Node.js, MongoDB and Express

Janosch Zbick
janosch.zbick@lnu.se

Client-side JavaScript

- As mentioned before, there are many JavaScript libraries available that helps creating client-side applications (jQuery, ...)
  - In recent the times, the most important one is: AngularJS
    - It offers support in developing client-side applications
    - Main features
      - Model-View-Controller structure on client-side
      - Two way data-binding
        - Through directives it is possible to bind input data from HTML5 to the controller and the other way around
    - Besides the structuring and two-way data binding, it is suppose to be faster than jQuery
    - Higher learning curve (Angular 2 is about to be launched)

Janosch Zbick
janosch.zbick@lnu.se


AngularJS – example

- Let's have a look at a simple AngularJS example
  - It makes use of AngularUI router for flexible routing with nested views in AngularJS
  - AngularUI offers a set of good extensions for AngularJS (Router, Bootstrap (withouth jQuery), Google Maps, etc.)

Janosch Zbick
janosch.zbick@lnu.se


AngularJS – mobile example

- Ionic makes use of AngularJS and CSS to design mobile applications
  - It also provides the structure and support for Cordova to allow the port of web-applications to native applications
  - Let's see how we can set-up a tab-example
  - And let's have a look at a simple example with the combination of Cordova

Janosch Zbick
janosch.zbick@lnu.se


The end

Thank you for your attention

I hope you saw how useful JavaScript can be

Now is the time for questions

Janosch Zbick
janosch.zbick@lnu.se