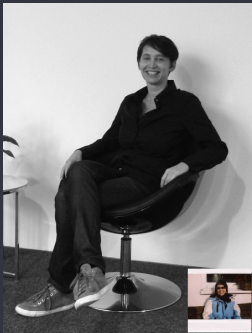# NoSQL web apps

w/ MongoDB, Node.js, AngularJS

Dr. Gerd Jungbluth, NoSQL UG Cologne, 4.9.2013

About us

- Passionate (web) dev. since fallen in love with Sinclair ZX Spectrum
- Academic background in natural sciences
- Entrepreneur
- Always moving forward

- Passionate web user / worker since a long time
- Academic background in psychology / social sciences
- Entrepreneur
- Be open minded

Agenda

press Esc

SlideCaptain WebApp

Online (HTML5) Presentation Tool

+

NEUER FLOW

NoSQL web apps

w/ MongoDB, Node.js, AngularJS

Dr. Gerd Jungbluth, NoSQL UG Cologne, 4.9.2013

NOSQL WEB APPS WITH MONGODB, …
🏷 NoSQL, web apps, MongoDB, Node.js, A…
📅 Gestern um 16:34 Uhr

Floating images

FLOATING IMAGES
🏷 image, editing
📅 09.08.2013

CREATE AND SHARE MODERN PRESENTATIONS

Your presentations are fully responsive with touch device support.
Show them on desktops, laptops, tablets or phones.

SLIDECAPTAIN
🏷 product page
📅 26.07.2013

AngularJS and NodeJS for a modern web application

SlideCaptain

SLIDECAPTAIN UND ANGULARJS
🏷 slidecaptain, angularJS, module, app
📅 23.07.2013

Periodensystem der Elemente

Abbildung: http://www.periodni.com/download/periodinsystem-farbe.png

PERIODENSYSTEM DER ELEMENTE
🏷 Chemie, Periodensystem, Naturwissens…
📅 21.07.2013

Embed video

Feature

EMBED VIDEOS
🏷 videos
📅 18.07.2013

Features

PDF and ZIP Export

PDF EXPORT
🏷 pdf
📅 16.07.2013

MathJax Support

MATHJAX SUPPORT
🏷 mathjax, equations, latex
📅 16.07.2013

NEW FEATURES

(BACKGROUND) IMAGES and COLORS

BACKGROUND IMAGES AND COLORS
🏷 background, images, style, colors
📅 09.07.2013

we are

Engawa

Our passion is creating software, that people love to use.

powered by SlideCaptain

ENGAWA TRAILER
🏷 engawa, webpage, products
📅 20.06.2013

Formatting Elements

Elements and Styles

FORMATTING ELEMENTS
🏷 format, editing, style
📅 14.05.2013

NoSQL web apps

w/ MongoDB, Node.js, AngularJS

Dr. Gerd Jungbluth, NoSQL UG Cologne, 4.9.2013

Three tiers

AngularJS
(HTML5 / CSS3 / JavaScript / XHR / JSON)

Node.js
(JavaScript / JSON / REST)

MongoDB
(JSON / BSON)

What is **MongoDB**?

*MongoDB (from "humongous") is an open-source **document database**, and the leading NoSQL database.*

- **document oriented** database w/ flexible schema
- secondary **indexes** support
- rich ad-hoc **query language** (JSON based)
- **replication** and **sharding** built-in
- supports **map / reduce**
- **aggregation** framework

Why we chose MongoDB

- bored of the SQL / ORM overhead
- when we use JSON in the browser, why not use it in the database as well
- agile / flexible
- easy to get started
- ad hoc queries
- excellent support for Node.js

One data format

JSON *is a* **syntax** *for* **serializing** *objects, arrays, numbers, strings, booleans, and null*

Example: first page of this presentation in JSON

```
{
  "_id": "5224a1eb603315b210000006",
  "centered": true,
  "paragraphs": [
    {
      "content": "<h1>NoSQL web apps</h1>",
      "_id": "5225d14002f1be8214000460",
      "parallax": {
        "active": false
      },
      "fragment": false
    },
    {
      "content": "<h3>w/ MongoDB, Node.js, AngularJS</h3>",
      "_id": "5225d14002f1be821400045f",
      "parallax": {
        "active": false
```

What is **Node.js**?

> *Node.js is a **platform** built on Chrome's JavaScript runtime for easily **building** fast, scalable network **applications.***

- based on **V8** (Google's JavaScript engine) for JavaScript execution
- based on **libuv** (a high performance evented, non-blocking I/O library) for OS interaction
- the **core** (written in C / C++ / JavaScript) is intentionally kept **small**
- user contributed **modules** available via **npm** ("node package manager")
- executed in a **single thread**, demanding **asynchronous** programming

Why we chose Node.js

- bored of the JEE overhead
- when we use JavaScript in the browser, why not use it on the server as well
- easy to get started
- perfect for REST APIs
- excellent support for MongoDB
- excellent ODM (Mongoose)
- huge (~40.000) number of modules available

One paradigma

In **event-driven** programming, an application expresses **interest** in certain events and responds to them *when they occur.*

Example: synchronous (mongo Shell) vs. asynchronous (Node.js) database query

```javascript
var p1, p2, query;
query = {"sections.pages.headline" : "One paradigma"};

// synchronous, blocking (mongo Shell)
p1 = db.presentations.find(query);
console.log(typeof p1); // "object"

// asynchronous, non-blocking (MongoDB native Node.js driver)
db.collection("presentations").find(query, function(error, result) {
 if (error) { throw new Error(error); }
  p2 = result;
});
console.log(typeof p2); // undefined
```

What is **AngularJS**?

*AngularJS is a JavaScript **framework** for building **web applications** with HTML, JavaScript and CSS. It offers powerful **data-binding**, **dependency injection**, guidelines for structuring your app and other useful goodies to make your webapp testable and maintainable.*

- written from scratch at Google
- highly modular
- **model** objects are plain JavaScript objects
- **view** templates are plain HTML with directives and expressions
- **controller** mediates between model and view
- **services** provide additional functionality (e.g. REST, client side routing, l10n, local storage)

Why we chose AngularJS

To be honest: there are too many reasons to mention them here :-)

Example: just a few snippets (HTML, directives, expressions, filters, data binding)

```html
<!-- plain HTML with directives and expressions -->
<ul>
  <li ng-repeat="page in pages | limitTo:5" ng-click="selectPage(page)">
    {{page.headline}} - {{page.paragraphs.length}}
  </li>
</ul>
<!-- custom directives to build reusable components -->
<div sc-panel data-name="p1" data-headline="My Panel">
  <!-- custom content -->
</div>
<!-- changes in the model are automatically applied to the view -->
<form>
  <input type="text" ng-model="user.name.first" required />
  Hello {{user.name.first}}
</form>
```

One language to rule them all

JavaScript is a lightweight, interpreted, object-oriented programming language with first-class functions

Benefits of the M(E)AN stack

Technical

1. One data format (**JSON**)

2. One programming paradigma (**asynchronous** / event driven)

3. One programming language (**JavaScript**)

Commercial

1. **Scalability** built in from the beginning

2. Perfect for the **cloud**

3. Open Source

4. Incredibly active **community**

5. Strong **commercial** support

Thank you

This presentation was built with SlideCaptain