

Zusammenfassung

Die Dynamik und Innovationsrate der JavaScript-Welt ist enorm: Grund genug für uns, einen Einstieg zu wagen. Wir zeigen in diesem Vortrag, wie wir als erfahrene Java-Entwickler erste Schritte mit den MEAN-Technologien machten: MongoDB, ExpressJs, AngularJs und NodeJs. Der Vortrag bietet einen Überblick über die Architektur einer MEAN-Applikation sowie über die eingesetzten Frameworks und Tools. Zudem zeigen wir auf, wie wir unsere Einarbeitung in die JavaScript-Welt als Mini-Projekt vorbereitet und durchgeführt haben. Das Resultat findet sich unter notes42.herokuapp.com, ein Prototyp einer "geeky" Web-App für Links und Notizen.

Das Ziel

Mini-Projekt basierend auf MEAN Stack



Die Idee

«geeky» Web-App für Links, Notizen

Notes42 all notes recently added notes export all notes all shared queries Raimond Reichert ▾

? #programming

g+ id programming se

Queried Notes (5)

Tags	Notes	
#g+ x #id x #programming x	code.org/educate/20hr K-8 20 Lektionen Einführung ins Programmieren mit Blockly, teilweise in deutsch verfügbar (zB Videos mit Untertiteln). Je nach Altersstufe ("appropriate for kindergartners through 8th graders and beyond") dürfte es aber wohl durchaus etwas anspruchsvoller sein. G+	🔗 🔍 🗑️
#id x #programming x	henrikwarne.com/2012/06/02/why-i-love-coding Summary of Fred Brooks "The Joys of the Craft" in "The Mythical Man-Month": <ul style="list-style-type: none">• The sheer joy of making things.• The pleasure of making things that are useful to other people.• The fascination of fashioning complex puzzle-like objects of interlocking moving parts, and watching them work in subtle cycles, playing out the consequences of principles built in from the beginning.• The joy of always learning, which springs from the nonrepeating nature of the task.• The delight of working in such a tractable medium. The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of imagination. ... yet the program construct, unlike the poet's words, is real in the sense that it moves and works, producing visible outputs separate from the construct itself	🔗 🔍 🗑️







Einfache Query: Notes zu Tag-Kombination

? #mean #grunt

? #mean #grunt

grunt mean

Queried Notes (2)

Tags	Notes	
#grunt x #mean x	www.npmjs.org/package/grunt-focus um dedizierte Watches zu definieren	  
#grunt x #mean x	www.npmjs.org/package/load-grunt-config um Gruntfile.js aufzuteilen	  




Modify: Note hinzufügen

#mean #grunt <http://gruntjs.com/plugins> Grunt Plugins

agile allegro angularjs **ausflug** autor bootstrap cacm coffeescript css edusef **ergon** familie foo g+ ganztag git **grunt** gutwetter halbtage html html5 id inbox java javascript kia kino **log** mean migros mongodb nadine&richi nodejs positioning presentations programming published quovadis regexp restaurant robotics schlechtwetter se testing thomas todo tools upcoming wandern windows

Modified Notes (1)

Cool! We have added the following notes:

Tags	Notes	
#grunt x #mean x	gruntjs.com/plugins Grunt Plugins	  

Piped Command «Export»: Notes exportieren

? #mean #grunt | export

```
? #mean #grunt | export|
```

grunt mean

Exported Notes

```
#grunt #mean https://www.npmjs.org/package/grunt-focus um dedizierte Watches zu definieren  
#grunt #mean https://www.npmjs.org/package/load-grunt-config um Gruntfile.js aufzuteilen  
#grunt #mean http://gruntjs.com/plugins Grunt Plugins
```

Piped Command «Share»: Notes veröffentlichen










? #mean #grunt |share

? #mean #grunt |share|

Share with this URL

grunt mean

Queried Notes (3)

Tags	Notes	
#grunt x #mean x	www.npmjs.org/package/grunt-focus um dedizierte Watches zu definieren	  
#grunt x #mean x	www.npmjs.org/package/load-grunt-config um Gruntfile.js aufzuteilen	  
#grunt x #mean x	gruntjs.com/plugins Grunt Plugins	  

Piped Command «Share»: Veröffentlichte Notes

notes42.herokuapp.com/#!/shared/4c54d060-34cc-4720-9424-80c24c15121d

Raimond Reichert shared 3 notes with you

Tags	Notes
#grunt #mean	www.npmjs.org/package/grunt-focus um dedizierte Watches zu definieren
#grunt #mean	www.npmjs.org/package/load-grunt-config um Gruntfile.js aufzuteilen
#grunt #mean	gruntjs.com/plugins Grunt Plugins

Weitere Query Operatoren und Kombination von Queries und Piped Commands

Notes mit Tag «css» ausschliessen	? #mean -#css
Notes, die in den letzten 7h erstellt oder verändert wurden	? :7h
Sortieren nach zum Beispiel URL	? sort by url asc
Sortiert exportieren	? sort by created desc export
Sortiert veröffentlichen	? sort by urlVisitedCount desc share

Die Vorbereitung Planung

April Ideen prüfen, Stories schreiben

Mai Lesen, lesen, lesen

Juni Setup von VM sowie Entwicklungs- und Produktionsumgebung

Juli 10 Tage Projekt

August Sporadische Weiterentwicklung

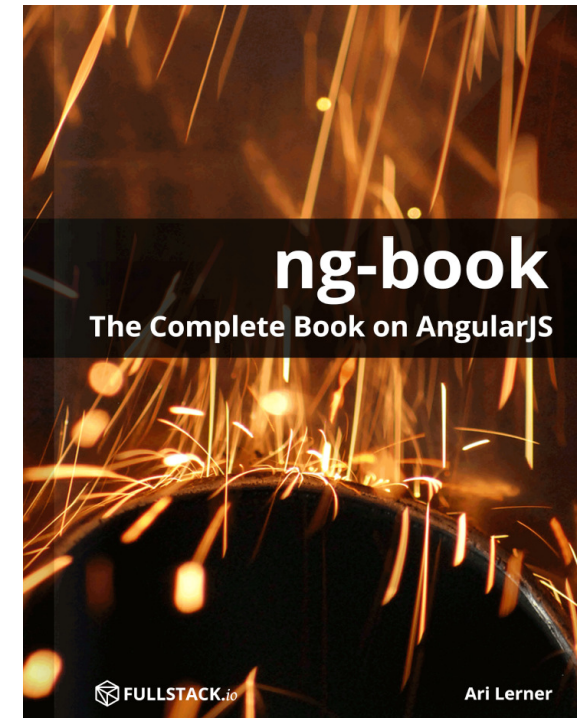
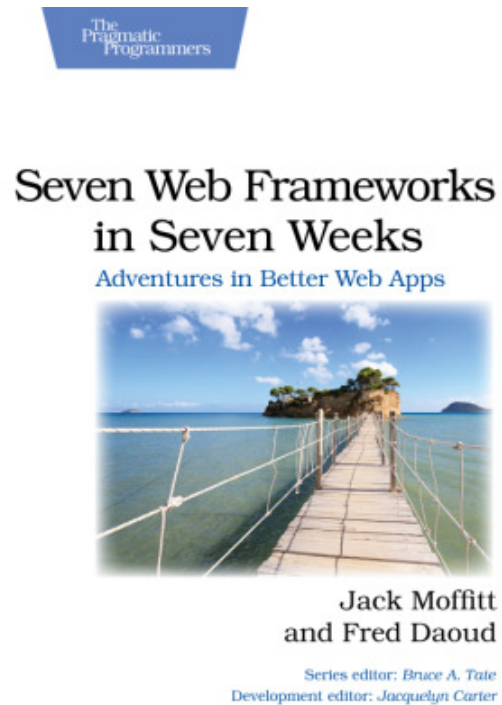
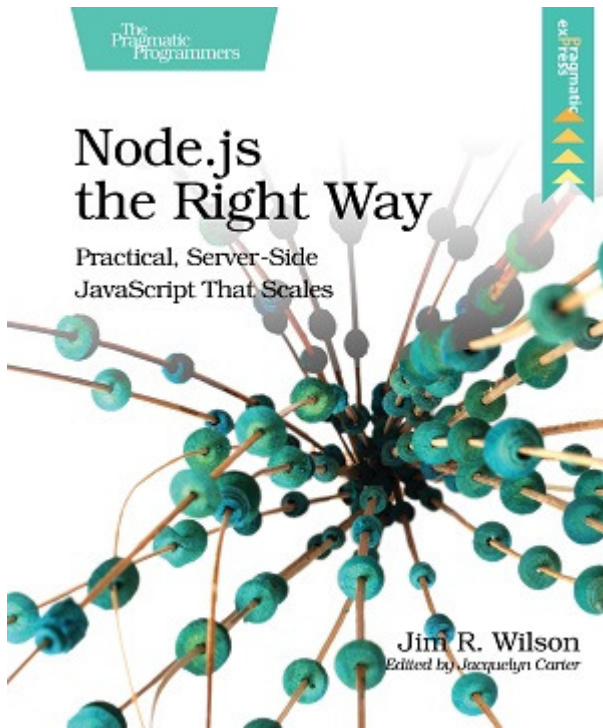
Die Vorbereitung Stories für «minimal viable product» auf github.com

The screenshot shows the GitHub interface for the repository 'samzurcher/eduself'. The page displays a list of issues with the following details:

- Repository: samzurcher / eduself (PRIVATE)
- Search: This repository Search
- Navigation: Explore Gist Blog Help
- User: raimondreichert
- Actions: Unwatch (2), Star (0), Fork (0)
- Issue filters: is:issue is:closed sort:created-asc
- Issue list summary: 10 Open, 47 Closed
- Issue list columns: Author, Labels, Milestones, Assignee, Sort

Issue Title	Labels	Milestones	Comments
Einfache Applikation mit Google Authentisierung	epic: infrastructure, ready, story	Sprint 1	19
Setup DA project	epic: infrastructure, story		1
WebUI: Alle Tags anzeigen	epic: links, ready, story	Sprint 3	1
JSON API: Link erfassen	epic: links, ready, story	Sprint 2	7
WebUI: Link erfassen	epic: links, ready, story	Sprint 2	4
JSON API: Alle Tags abfragen	epic: links, ready, story	Sprint 3	6

Die Vorbereitung angularjs.org, nodejs.org, ...



Die Vorbereitung Scripted Ubuntu VM Setup

```
sudo apt-key adv --keyserver keyserver.ubuntu.com --recv 7F0CEB10
echo 'deb http://downloads-distro.mongodb.org/repo/debian-sysvinit dist 10gen' |
sudo tee /etc/apt/sources.list.d/mongodb.list
sudo apt-get update
sudo apt-get install mongodb-10gen
```

```
sudo apt-get update
sudo apt-get -y install python-software-properties python g++ make
sudo add-apt-repository ppa:chris-lea/node.js
sudo apt-get update
sudo apt-get -y install nodejs
sudo npm install -g grunt-cli bower
```

...



Die Vorbereitung MEAN-Projekt-Scaffolding mit mean.io

- > sudo npm install -g meanio@latest
- > mean init notes42
- > cd notes42 && npm install
- > grunt

... und schon hat man eine einfache CRUD-Applikation inklusive Authentifizierung.

Die Vorbereitung Infrastruktur-Setup abgeschlossen

- ✓ «Setup VM und Entwicklungsumgebung»
- ✓ «Setup Projekt»
- ✓ «Einfache Applikation mit Google Authentisierung»
- ✓ «Setup Produktionsumgebung»

Die Umsetzung 5 Sprints à 2 Tage

April Ideen prüfen, Stories schreiben

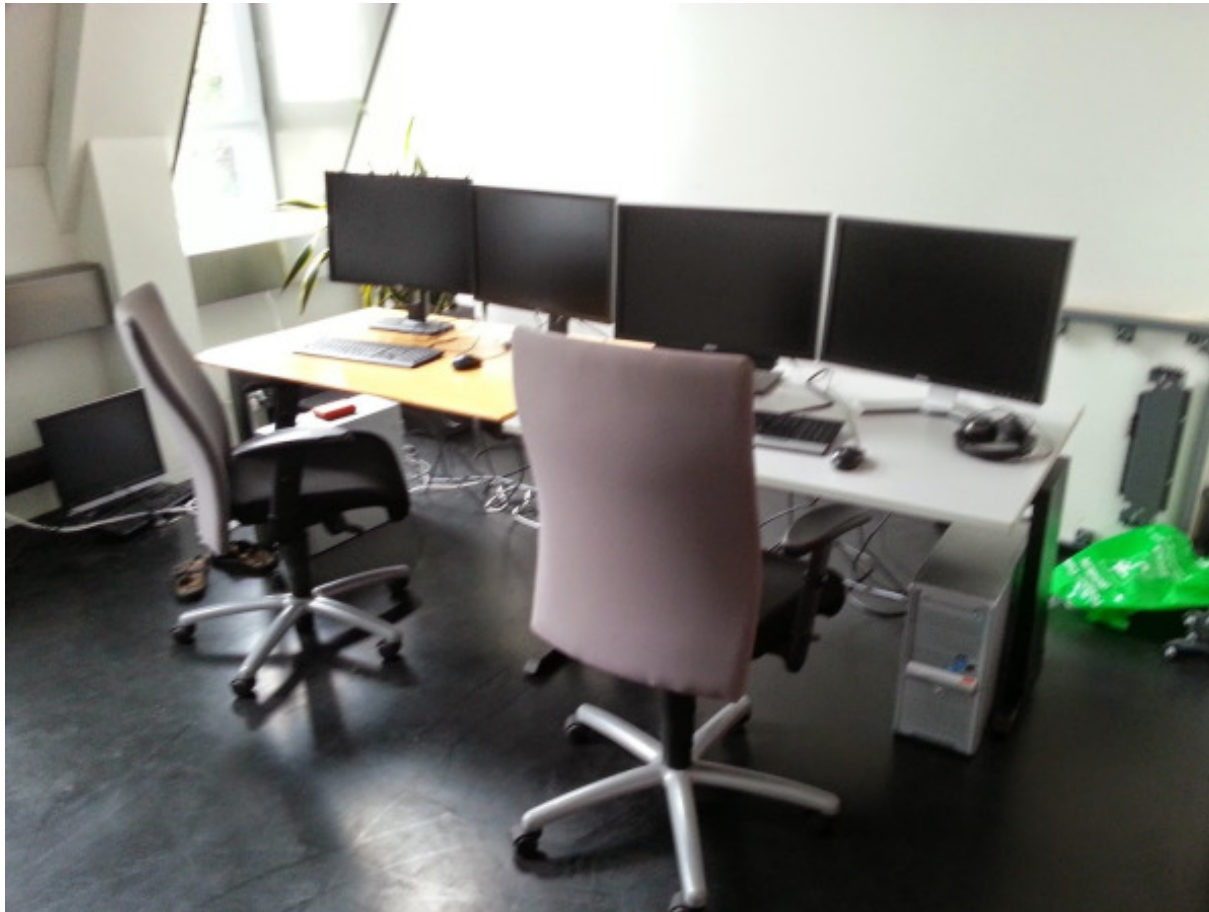
Mai Lesen, lesen, lesen

Juni VM Setup, Development and
Production Environment Setup

Juli 10 Tage Projekt

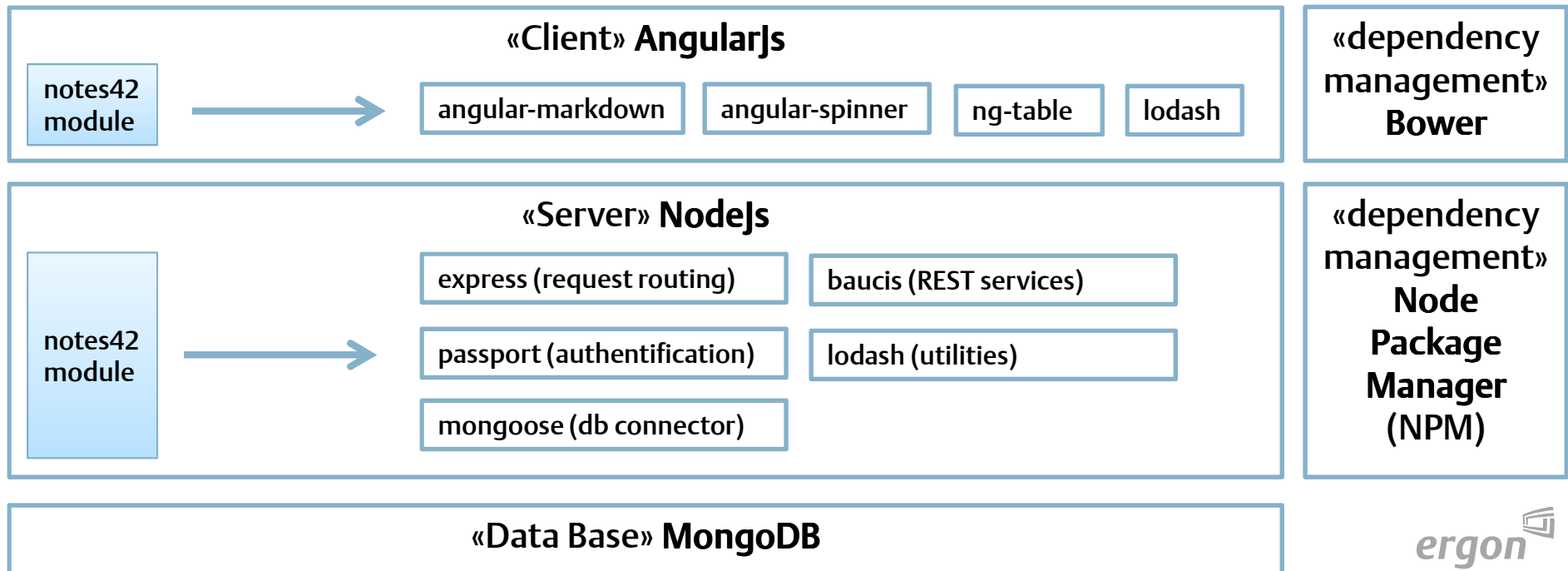
August Sporadische Weiterentwicklung

Der Arbeitsplatz Setup für «Extreme pair programming»

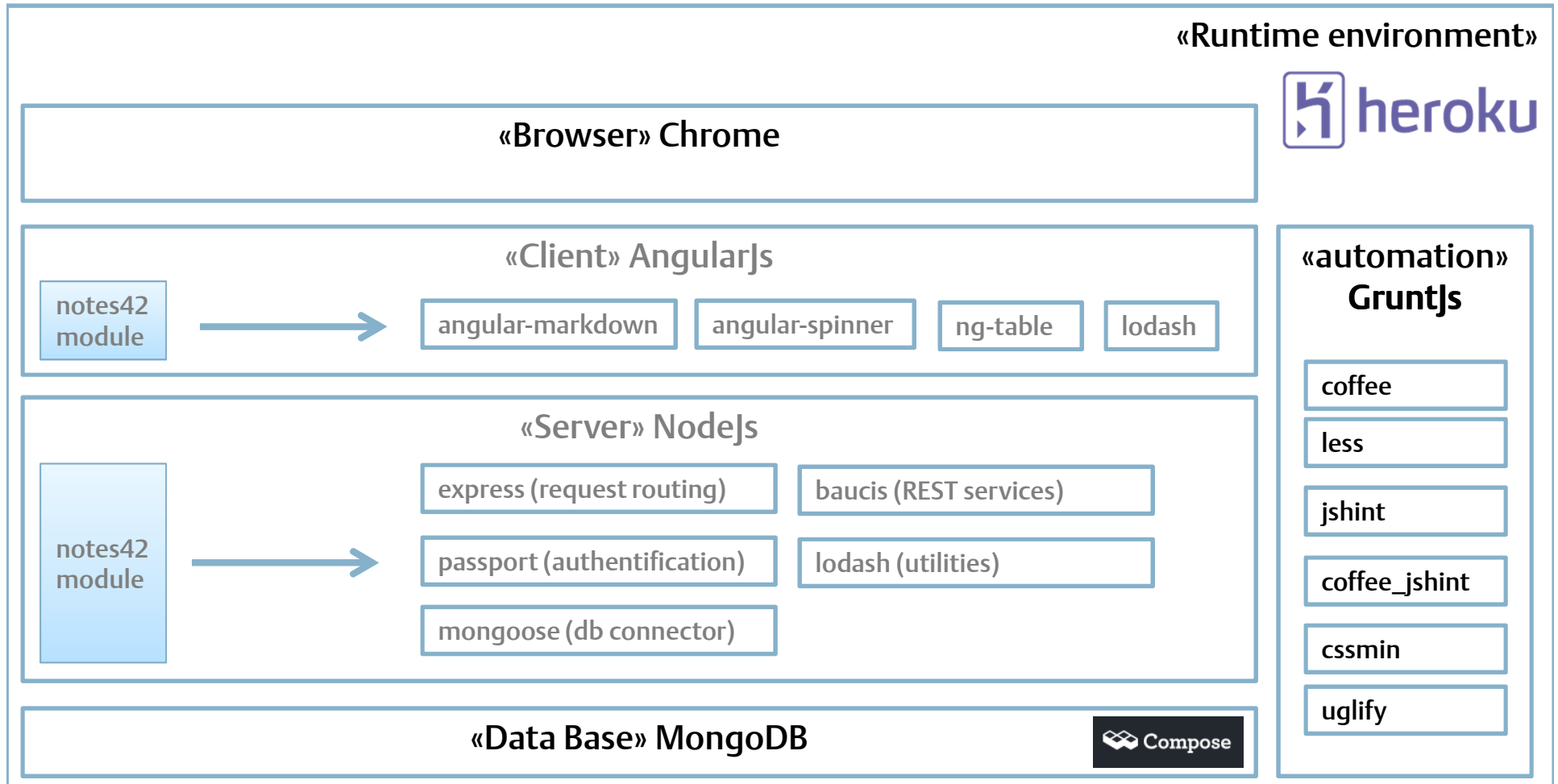


Komponenten der Applikation

Kleine, fokussierte Plugins für so ziemlich alles



Produktionsumgebung: «git push heroku master»



Development Webstorm IDE – empfehlenswert!

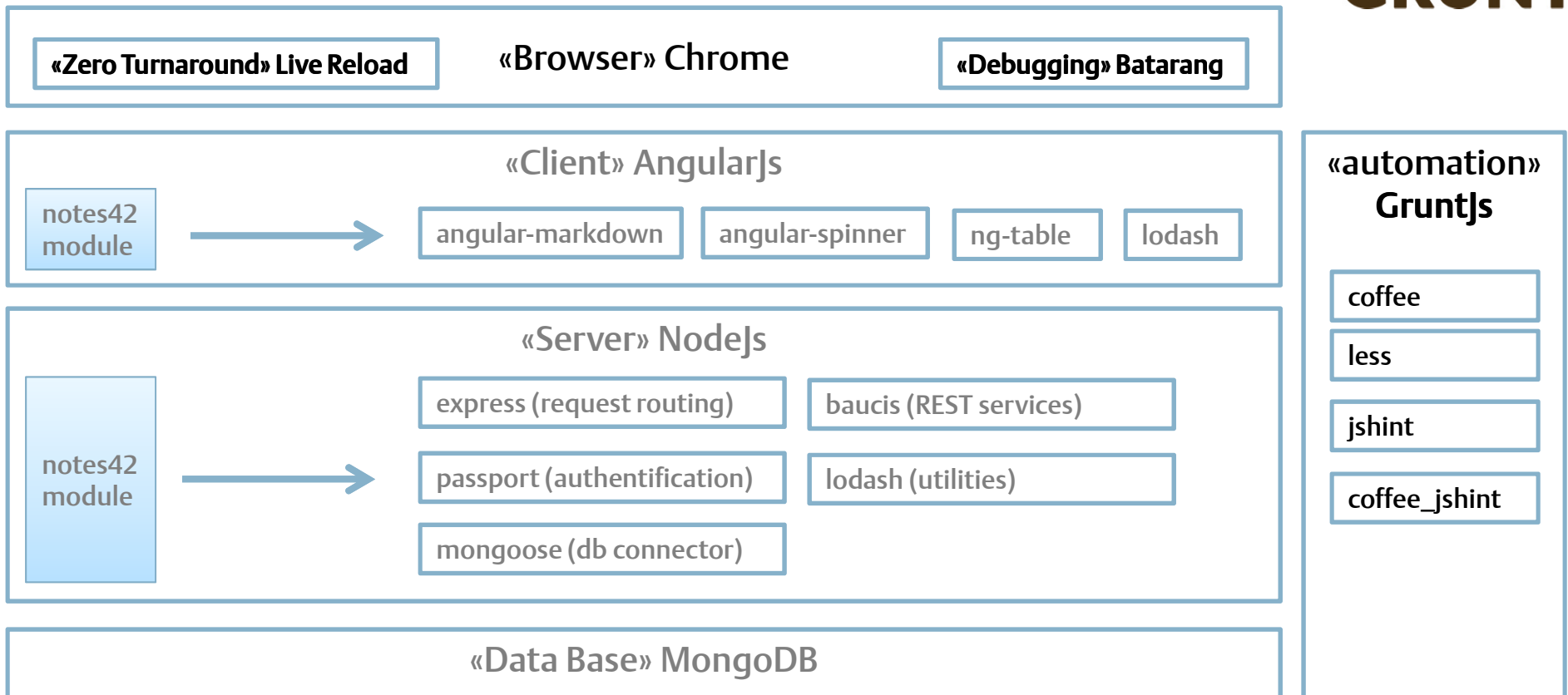


```
1 Command = require('./command')
2 UnvalidatedCommand = require('./unvalidated-command')
3 TimeRangeTransformer = require('./time-range-transformer')
4
5 class UnvalidatedCommandParser
6   @EXCLUDED_TAG_REG_EXP = /^-#[^s#]*/
7   @TAG_REG_EXP = /#[^s#]*/
8   @TIME_RANGE_REG_EXP = /#\d+(ms|my|h|d)?/
9   @HTTP_REG_EXP = /^(https?:\/\/|www\.)[^s]*$/
10  @ID_REG_EXP = /#id:\w+/
11  @PIPED_COMMANDS_REG_EXP = /#(\s*\w+)*$/
12
13  constructor: () ->
14    @timeRangeTransformer = new TimeRangeTransformer()
15
16  parse: (input) ->
17    unvalidatedCommand = new UnvalidatedCommand()
18    if input?
19      input = this._parseMode input, unvalidatedCommand
20
21      lastInputLength = input.length
22      loop
23        input = this._parseTag input, unvalidatedCommand
24        input = this._parseExcludedTag input, unvalidatedCommand
25        input = this._parseTimeRange input, unvalidatedCommand
26        input = this._parseUrl input, unvalidatedCommand if unvalidatedCommand.url.length is 0
27        input = this._parseId input, unvalidatedCommand
28
29        break if lastInputLength is input.length
30        lastInputLength = input.length
31      if unvalidatedCommand.mode is Command.QUERY
32        input = this._parsePipedCommands input, unvalidatedCommand
33      else
34        input = this._parseReplaceTextMode input, unvalidatedCommand
35        unvalidatedCommand.text = input.trim()
36        unvalidatedCommand.timeRanges = @timeRangeTransformer.transform unvalidatedCommand.timeRanges
37        unvalidatedCommand
38
39  _parseMode: (input, command) ->
40    input = input.trim()
```



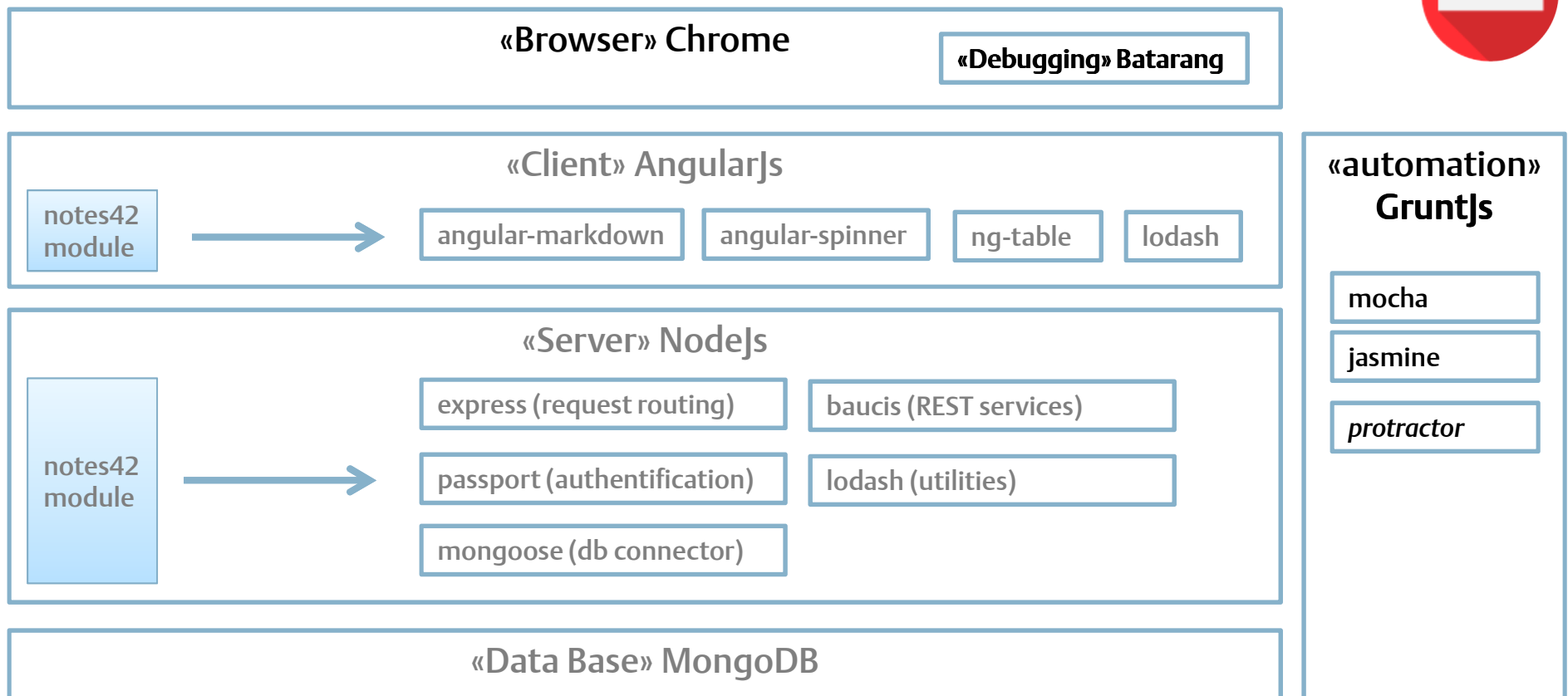
Development

Gruntjs: CoffeeScript und LESS «on save» übersetzen



Development

Gruntjs: Unit Tests «on save» ausführen



Development

JavaScript: Funktionaler Programmierstil

«functions are first class citizens»

Closures und (haufenweise) Callbacks zentral.

Gewöhnungsbedürftig: Kein Compiler, der vor Scope-Problemen in Closures warnt.

```
// returns list of books with at least 'threshold' copies sold
function bestSellingBooks(threshold) {
  return bookList.filter(
    function (book) { return book.sales >= threshold; }
  );
}
```


Development

JavaScript: Prototypen statt Klassen

```
// Prototyp-Definition
var Car = function Car (make, model, year) { // Konstruktor
  this.make = make;
  this.model = model;
  this.year = year;
}

Car.prototype.toString = function() { // Instanz-Methode
  return this.make + ' ' + this.model + ' (' + this.year + ')';
}

console.log(new Car("Eagle", "Talon TSi", 1993).toString());
// prints "Eagle Talon Tsi (1993)"
```

Development

JavaScript: Prototypen statt Klassen

```
// Beliebigen existierenden Objekt-Prototyp erweitern
```

```
String.prototype.dasherize = function() {  
  return this.replace(/_/g, "-");  
};
```

```
console.log("no_more_underscores".dasherize());  
// prints "no-more-underscores"
```

Development

JavaScript: Duck Typing statt Interfaces

- JavaScript ist dynamisch typisiert – Duck Typing statt Interfaces:
«When I see a bird that walks like a duck and swims like a duck and quacks like a duck, I call that bird a duck.»
- Gerade im Zusammenhang mit Callbacks angenehm, da man nicht überall Ein-Methoden-Interfaces definieren muss.
- Nachteile der dynamischen Typisierung:
 - Objekt-Datenstruktur ist im Wesentlichen ein assoziativer Array. Es gibt kein «Design by Contract» in Form von Klassen oder Interfaces, um Objekte verbindlich zu definieren.
 - Automatisierte Refactorings trotz IDE-Unterstützung schwierig.

Development

JavaScript: «Sicher» entwickeln

Software Engineering «Best Practices» werden zur zwingenden Voraussetzung, um in der dynamisch-typisierten JavaScript-Welt sicher entwickeln zu können.

- **Testing: Automatisierte Tests auf allen Ebenen: Unit, Integration, Application – gutes Tooling!**
- **Design: «Law of Demeter» einhalten: «Objekte interagieren nur mit ihrer unmittelbaren Umgebung»**
- **Static Analysis: Tools wie JsHint, «a program that flags suspicious usage in programs written in JavaScript»**



Development

CoffeeScript: «A little language that compiles into JS»

	CoffeeScript	JavaScript
Functions	<code>square = (x) -> x * x</code>	<code>square = function(x) { return x * x; };</code>
Array comprehension	<code>cubes = (math.cube num for num in list)</code>	<code>cubes = (function() { var _i, _len, _results; _results = []; for (_i = 0, _len = list.length; _i < _len; _i++) { num = list[_i]; _results.push(math.cube(num)); } return _results; })();</code>



Development

CoffeeScript: «syntactic sugar» für OOP

```
class Animal
```

```
  constructor: (@name) ->
```

```
    move: (meters) ->
```

```
      alert @name + " moved #{meters}m."
```

```
class Snake extends Animal
```

```
  move: ->
```

```
    alert "Slithering..."
```

```
    super 5
```

```
sammy = new Snake "Sammy the Python"
```

```
sammy.move()
```



Development CoffeeScript: Nachteile

- Build-Infrastruktur wird komplexer, da CoffeeScript kompiliert werden muss.
- JavaScript-Fehlermeldungen zur Laufzeit nicht immer einfach nach CoffeeScript rückzuübersetzen, im Zweifelsfall muss generiertes JavaScript angeschaut werden.
- Tools wie Coffee-JsHint haben noch Kinderkrankheiten wie unbrauchbare Fehlermeldungen.



Development

lodash: Unverzichtbare Utilities

```
_removeTags: (command, tagArray) ->  
  tagsToRemove = _.intersection command.excludedTags, tagArray  
  _.xor tagArray, tagsToRemove  
  
_addTags: (command, tagArray) ->  
  _.union tagArray, command.tags  
  
_getTagStringValue: (notes) ->  
  tagStrings = _.map notes, (note) -> note.tagString  
  sortedTagString = _.sortBy tagStrings, (tagString) -> tagString  
  tagStringValue = _.reduce tagStrings, (result, value) -> result + ' ' + value
```

... und vieles mehr!

Development

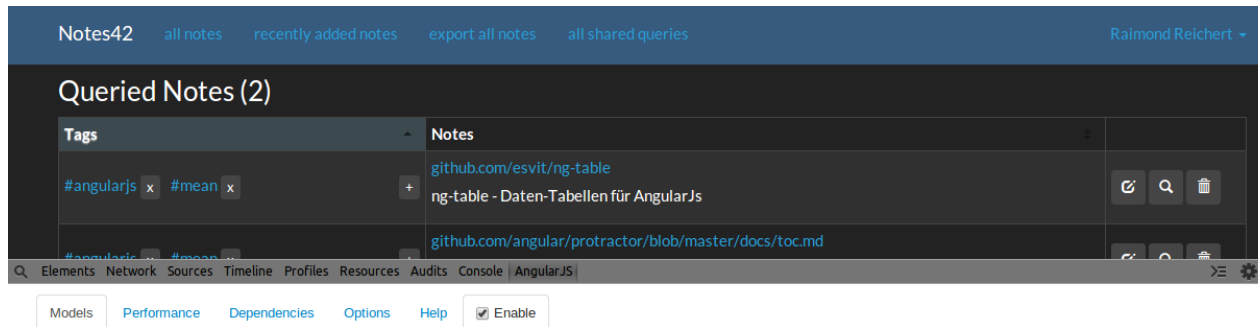
AngularJs : Data Binding – HTML + Expressions + Scopes

```
<html ng-app="phonecatApp"> <!-- head-Element weggelassen -->
<body ng-controller="PhoneListCtrl">
  <ul>
    <li ng-repeat="phone in phones">{{phone.name}}: {{phone.snippet}}</li>
  </ul>
</body>
</html>
```

```
var phonecatApp = angular.module('phonecatApp', []);
phonecatApp.controller('PhoneListCtrl', function ($scope) {
  $scope.phones = [
    {'name': 'Nexus S', 'snippet': 'Fast just got faster with Nexus S.'},
    {'name': 'Motorola XOOM', 'snippet': 'The Next, Next Generation tablet.'} ];
});
```

Development

AngularJS : Hierarchische Scopes – Debugging in Chrome



Scopes

```

< Scope (001)
  < Scope (002)
  < Scope (004)
  < Scope (005)
    < Scope (006)
  < Scope (007)
    < Scope (008)
      < Scope (009)
      < Scope (00A)
      < Scope (00B)
        < Scope (00F)
        < Scope (00G)
        < Scope (00H)
        < Scope (00I)
      < Scope (00C)
        < Scope (00J)
          < Scope (00K)
            < Scope (01D)
              < Scope (01F)
                < Scope (01G)
                < Scope (01H)
                < Scope (01I)
                < Scope (01J)
          < Scope (00N)
            < Scope (00S)
              < Scope (00Y)
            < Scope (00T)
  
```

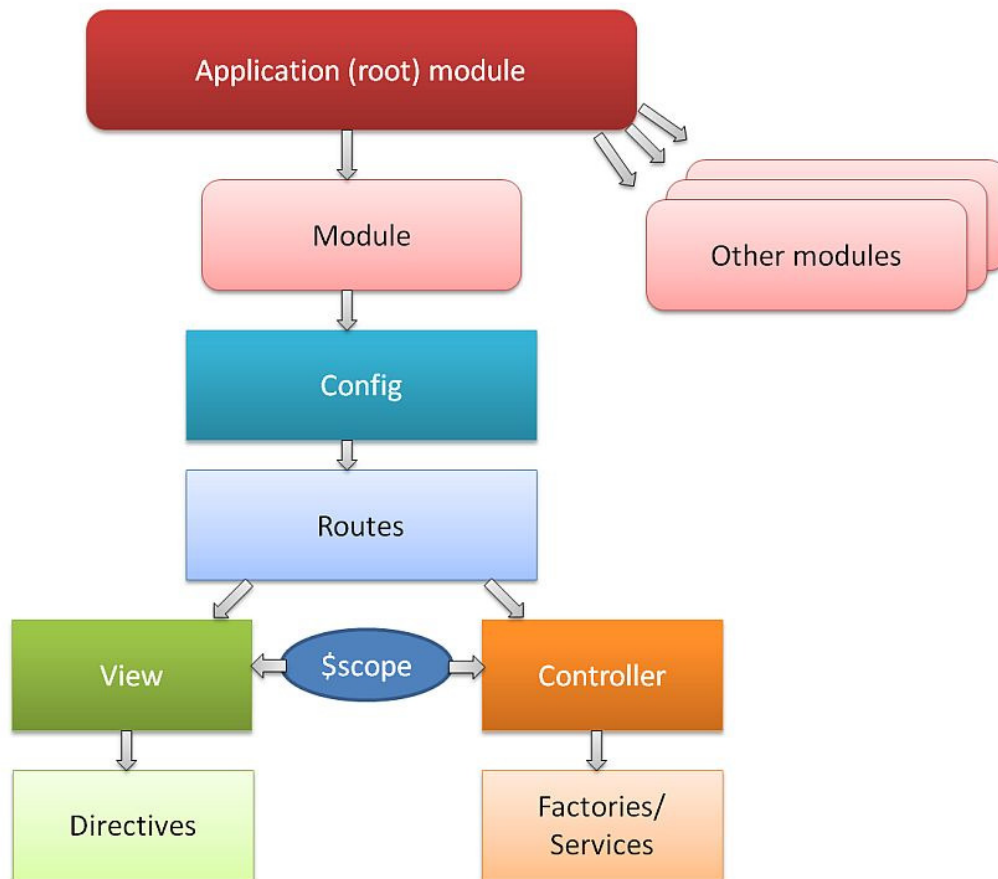
Models for (008)

```

{
  global: {
    user: {
      name: Raimond Reichert
      _id: 53b460ec3db29d00000001f80
      username: raimond.reichert@gmail.com
      roles:
        [ authenticated ]
    }
  }
  authenticated: 1
  isAdmin: 0
}
command: {
  state: notInitial
  value: ? #angularjs
  error:
}
guiMode: {
  value: expert
}
result: {
  value: {
    commandMode: query
    onlyQueryCommand: {
      mode: query
      tags:
        [ angularjs ]
      excludedTags:
  
```

Development

Angularjs : Gibt App-Architektur vor – flache Lernkurve



Konzepte

- Template
- Directive
- Model
- Scope
- Expression
- Compiler
- Filter
- View
- Data Binding
- Controller
- Dependency Injection
- Module
- Service



Development

AngularJs: Google als Sponsor, starke Community

Google als Sponsor

“Misko Hevery [...] managed to recreate a web application that consisted of 17 thousand lines of code and took 6 months to develop in a mere 3 weeks using just GetAngular. Reducing the size of the application to just about 1,000 lines of code convinced Google to start sponsoring the project, turning it into the open-source AngularJS we know today.”

Starke Community

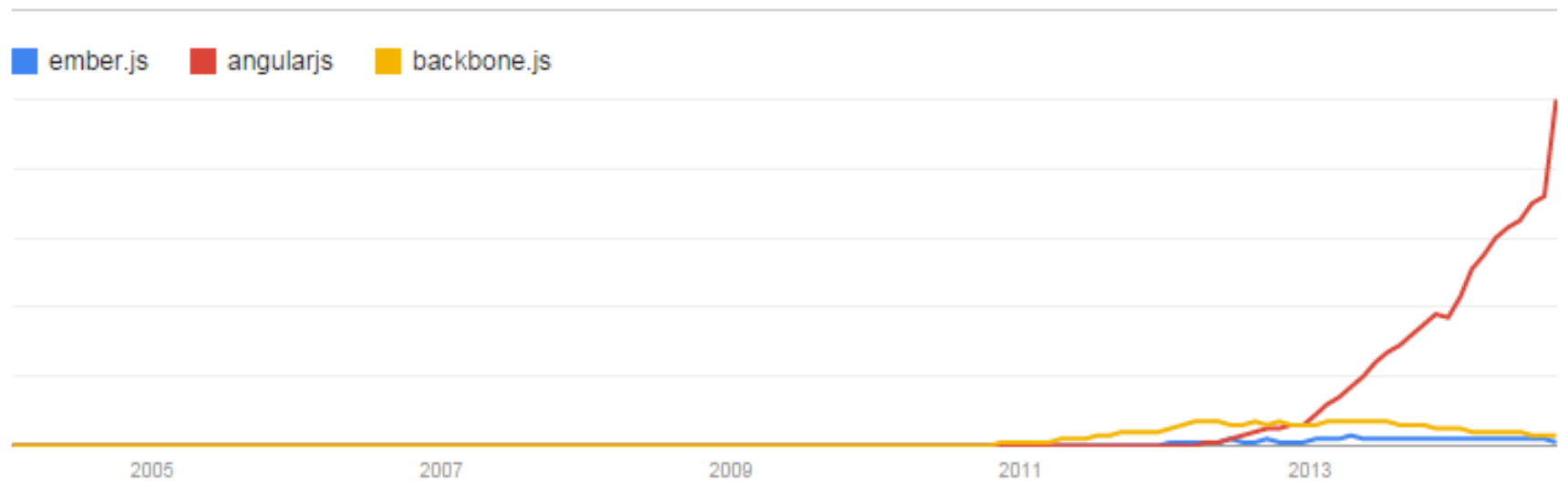
AngularJs hat über 800 Module auf ngmodules.org.

www.airpair.com/js/javascript-framework-comparison

Development AngularJS: Verbreitung

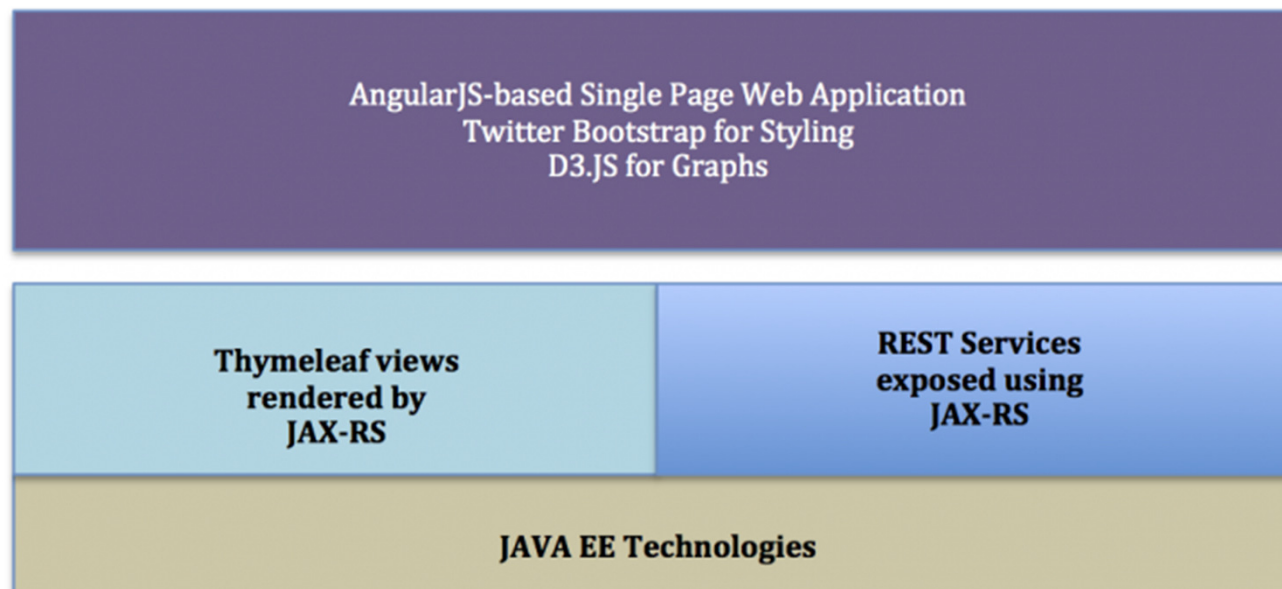


Interest over time. Web Search. Worldwide, 2004 - present.



Development

AngularJS: «Java EE 7 Real World Experience»



...

JEE7 – AngularJS als empfohlene Frontend-Technologie:

blog.arungupta.me/2014/09/

log-your-miles-and-community-runs-java-ee-7-real-world-experience/



Development

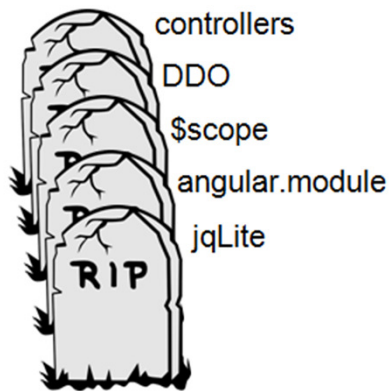
Angularjs: P.S. Browser Kompatibilität...

Unterstützt: Safari, Chrome, Firefox, Opera, IE9/10/11 und Mobile Browser (Android, Chrome Mobile, iOS Safari).

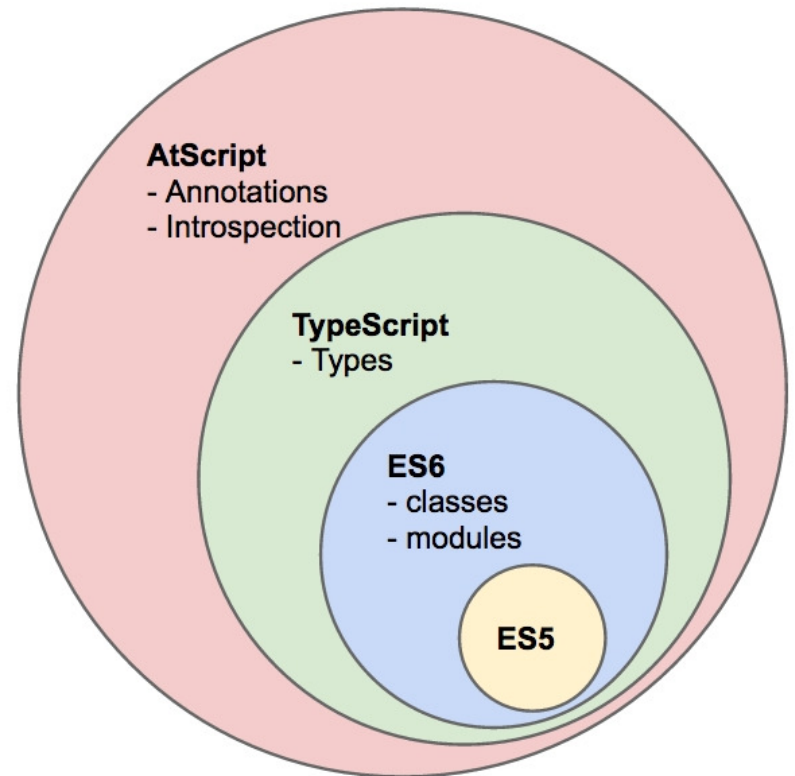
Nicht unterstützt: IE8. Da Microsoft im April 2014 offiziell den Support für Windows XP einstellte und Internet Explorer 8 vor allem darauf lief, unterstützt Angularjs ab Version 1.3 IE 8 nicht mehr.

AngularJs 2

Grundlegende Änderungen – kompletter Rewrite von 1.3



- + generic binding syntax
- + DI query mechanism
- + benchpress
- + WTF instrumentation



Development

Nodejs: Angenehm steile Lernkurve



example.js

```
var http = require('http');
http.createServer(function (req, res) {
  res.writeHead(200, {'Content-Type': 'text/plain'});
  res.end('Hello World\n');
}).listen(1337, '127.0.0.1');
console.log('Server running at http://127.0.0.1:1337/');
```

> node example.js

Server running at http://127.0.0.1:8124/



Development

NodeJs + ExpressJs: Routing einfach gemacht

```
var express = require('express');
```

```
var app = express();
```

```
app.get('/hello.txt', function(req, res) {  
  res.send('Hello World');  
});
```

```
var server = app.listen(3000, function() {  
  console.log('Listening on port %d', server.address().port);  
});
```



Development

NodeJs + ExpressJs: Server-side Rendering mit Jade

```
app.engine('jade', require('jade').__express);  
app.get('/', function (req, res) {  
  res.render('index', {  
    title: 'Hey',  
    message: 'Hello there!'  
  });  
})
```

index.jade

```
html  
  head  
    title!= title  
  body  
    h1!= message
```



Development

NodeJs: Leichtgewichtiger Container, einfach erweiterbar

- Im Vergleich zu JEE-Containern
 - ist Node **extrem leichtgewichtig** und startet in Sekundenbruchteilen,
 - gibt es keine Standards, aber eine sehr aktive Community.
- In NodeJs sind nur eine **überschaubare Anzahl APIs integriert**. Das wiederum hält die Lernkurve steil.
 - Für die Entwicklung von NodeJs-Webapplikationen kommt sehr oft auch **ExpressJs** (Web-App-Framework) zum Einsatz.
- Mit dem **Node Package Manager**. Via **www.npmjs.org** steht eine riesige Anzahl von Modulen zur Auswahl (Stand Sept 2014: 92'000).
 - “The **ecosystem** around JavaScript as a serious application platform continues to evolve. Many interesting new tools for testing, building, and managing dependencies in both server- and client-side JavaScript applications have emerged recently.” (Thoughtworks Tech Radar 2014)



Development

NodeJs: Sinnvolle Szenarien für Einsatz von NodeJs

“Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.“ (nodejs.org)

Gut geeignet für ...	Nicht geeignet für ...
<ul style="list-style-type: none">▪ Single-Page-Applikationen Backends / JSON-basierte REST-Services▪ Web-Realtime-Applikationen▪ Streaming	<ul style="list-style-type: none">▪ rechenintensive Anwendungen

heise.de/developer/artikel/2x-Nein-4x-Ja-Szenarien-fuer-Node-js-2111050.html



Development

NodeJs: Produktiv einsetzbar und eingesetzt

- **LinkedIn** hat von Ruby on Rails auf Nodejs umgestellt, siehe etwa cacm.acm.org/magazines/2014/2/171684-node-at-linkedin.
 - Nodejs als Server-seitiger Integrationslayer für diverse Backend-Systeme, die grösstenteils in JEE entwickelt wurden.
 - Beeindruckende Performance: «we discovered that Node was roughly 20 times faster than what we had been using and its memory footprint was smaller as well.»
- Microsoft, VMWare, Ebay, Yahoo und viele andere setzen Nodejs ein.



Development

MongoDB: Dokument-basierte Datenbank

Eigenwerbung von mongodb.org

MongoDB (from "humongous") is an open-source document database, and the leading NoSQL database. Written in C++, MongoDB features:

- **Document-Oriented Storage:** JSON-style documents with dynamic schemas offer simplicity and power.
- **Full Index Support:** Index on any attribute, just like you're used to.
- **Querying:** Rich, document-based queries.
- Map/Reduce, GridFs und vieles mehr



Development

mongoosejs: «elegant mongodb object modeling»

```
mongoose = require 'mongoose'  
Schema = mongoose.Schema  
timestamps = require 'mongoose-  
times'
```

```
NoteSchema = new Schema {  
  text: String  
  tagArray: [  
    type: String  
    index: true  
  ]
```

```
  tagString:  
    type: String  
    default: "  
    trim: true  
  url:  
    type: String  
    default: "  
    index: true  
  ownedBy:  
    type: Schema.Types.ObjectId  
    ref: 'User'  
    index: true  
}  
NoteSchema.plugin timestamps
```




Development

MongoDb + mongoosejs: Aggregation Pipeline-Beispiel

```
class TagsFinder
```

```
  # constructor not shown
```

```
  find: (command, successCallback, failureCallback) ->
```

```
    Note.aggregate([
```

```
      {$match: @matchConditionFactory.queryMatchCondition(command)}
```

```
      {$unwind: '$tagArray'}
```

```
      {$group: {_id: '$tagArray', number: {$sum: 1}}}
```

```
      {$sort: {_id: 1}}
```

```
    ]).exec (err, tags) ->
```

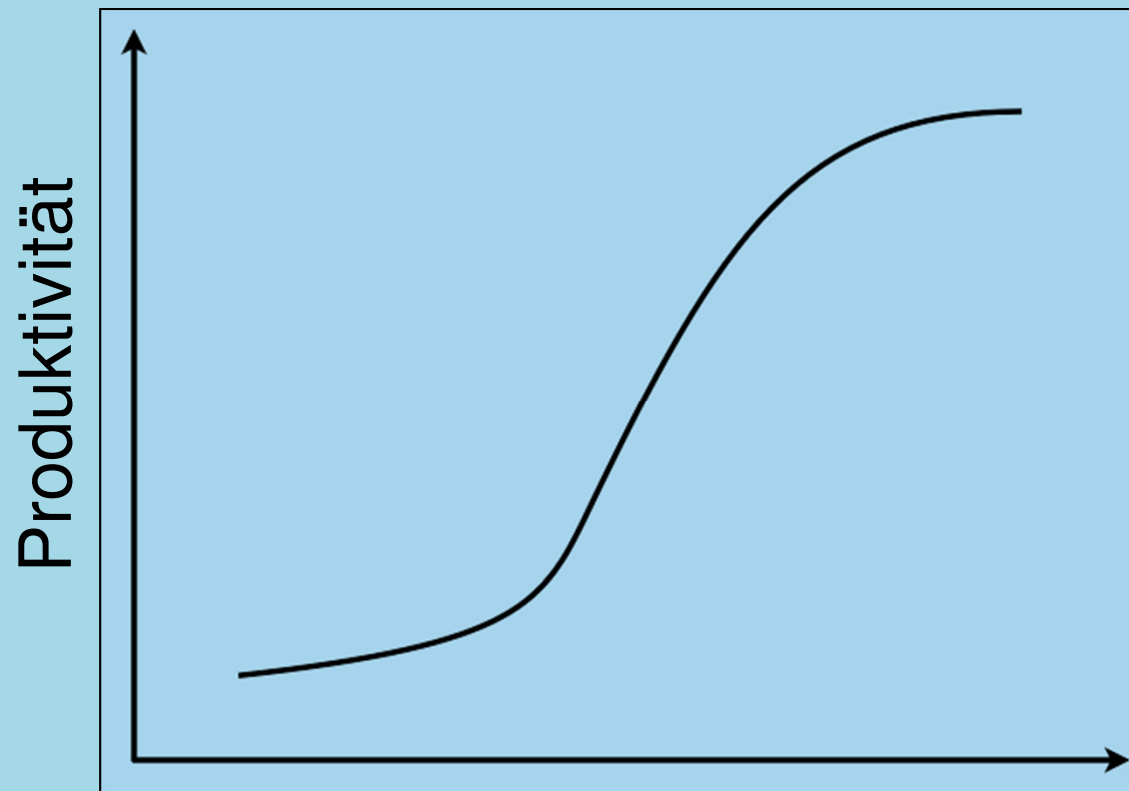
```
      mappedTags = _.map tags, (tag) -> { value: tag._id, count: tag.number }
```

```
      successCallback mappedTags
```

```
  # error handling not shown
```

Der Durchbruch

Nach 7 Tagen gefühlt sehr produktiv



Zeit

Der Rückblick



Lessons Learned

Methodisch

- **Zur Vorbereitung**
 - Unverzichtbar, um in der kurzen Zeit produktiv entwickeln zu können
 - Zeit für Infrastruktur-Aufbau: Der Aufwand zahlt sich rasch wieder aus
 - Einlesen in die Technologien: Spezialisierung hilfreich
 - Stories auf Github halfen, Projekt zu strukturieren und uns zu fokussieren
- **Zur Durchführung**
 - «Extrem pair programming»: Deutlich effizienter gelernt als alleine
 - Kurze Feedback-Zyklen: abends «lessons learned» notieren, morgens umsetzen
 - Time boxing: Manchmal müssen pragmatische Hacks genügen
 - «Continuous Deployment»: Von Anfang an regelmässige Verifikation auf herokuapp.com
 - Das Erfolgserlebnis: Nach 7 Tagen auch individuell produktiv

Lessons Learned

Was noch fehlt für «produktives Projekt»

- **Continuous Integration**
 - Zum Beispiel mit Jenkins: blog.dylants.com/2013/06/21/jenkins-and-node
 - Strider (stridercd.com) für Selbstbetrieb; auch als Hosted Service zu haben
 - Travis-Ci (travis-ci.org) als Hosted Service für Github-Projekte
 - CodeShip (www.codeship.io) als Hosted Service
- **Betrieb von NodeJs-Server**
 - Einzelne NodeJs-Server: Sehr einfach mit Forever (github.com/nodejitsu/forever)
 - Clusters zum Beispiel mit Strongloop (strongloop.com/node-js/controller)

Lessons Learned

Was noch fehlt für «produktives Projekt»

- **Code Coverage Tools**

- Istanbul (github.com/gotwarlost/istanbul) wird von Karma Test Runner verwendet
- BlanketJs (blanketjs.org)
- JsCover (tntim96.github.io/JScover)

- **Performanz-Analyse-Tools**

- Strongloop (strongloop.com/node-js/monitoring/)
- Weitere Möglichkeiten siehe www.clock.co.uk/blog/easy-cpu-profiling-in-node-js



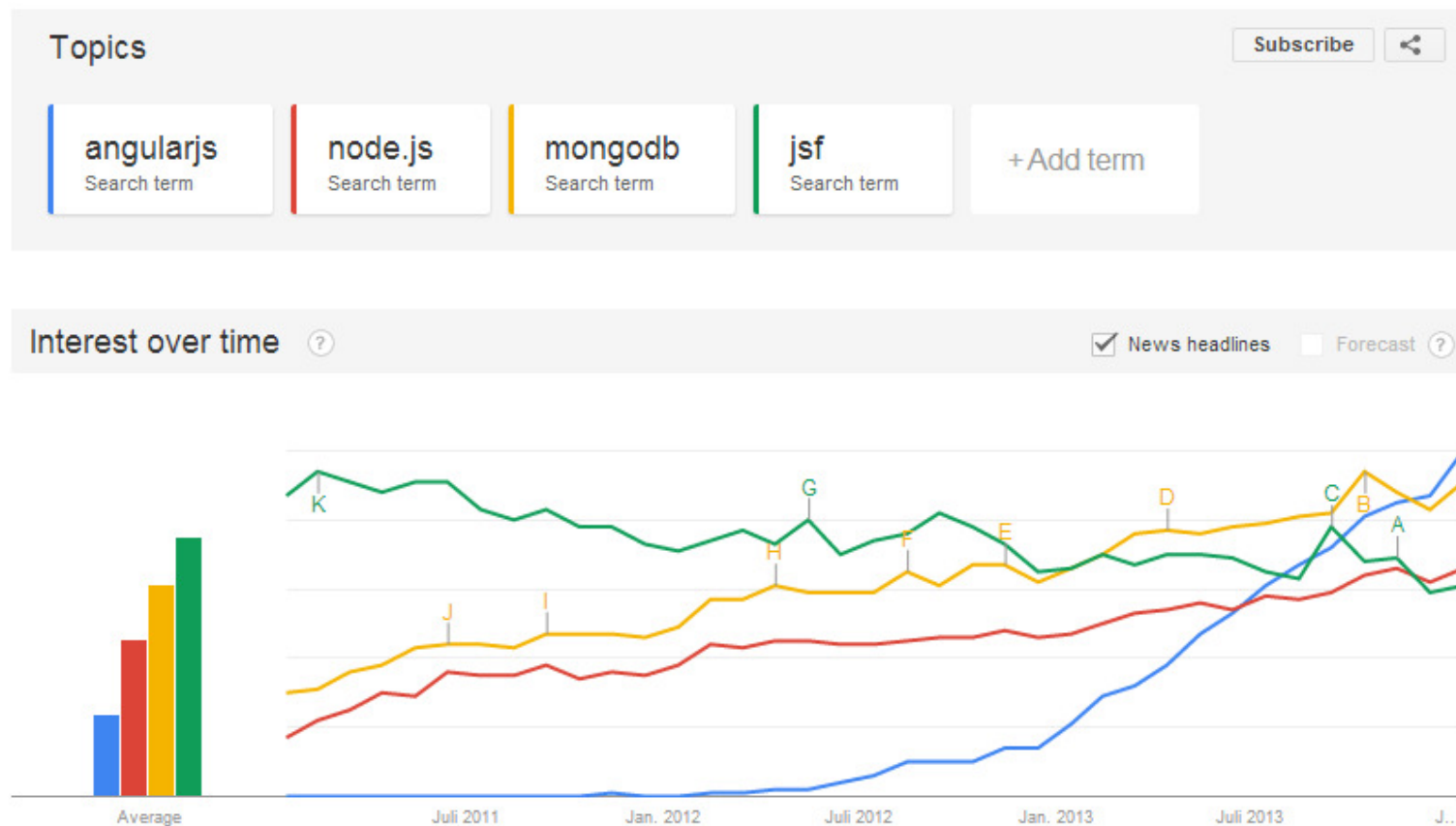
The Future

NEXT EXIT



Lessons Learned

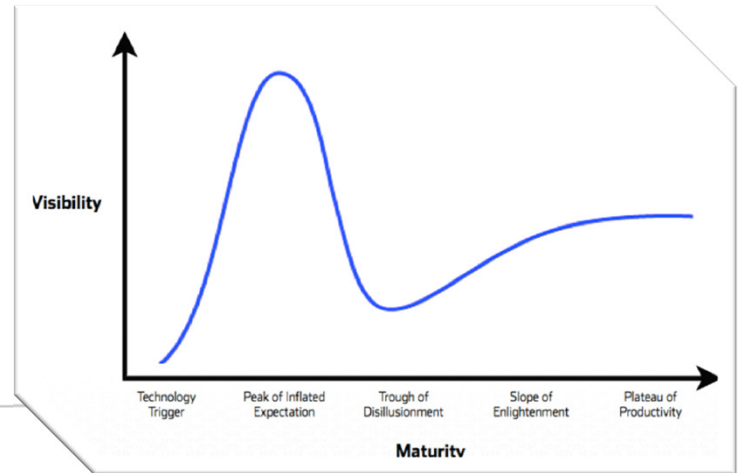
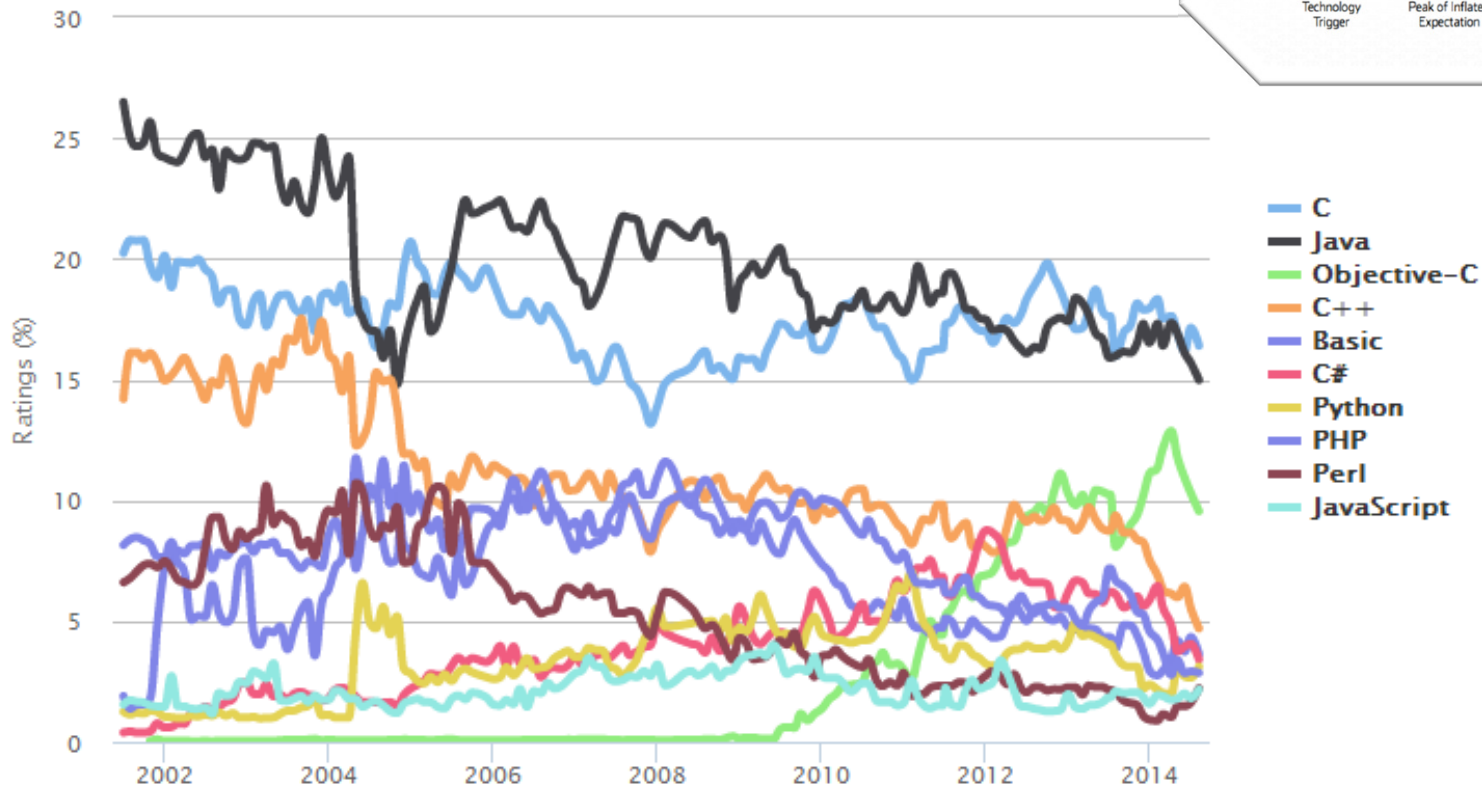
JavaScript macht Spass und ist voll im Trend ...



Lessons Learned ... der noch sehr jung ist!

TIOBE Programming Community Index

Source: www.tiobe.com





notes42.herokuapp.com

«notes for two» oder «notes fortytwo»?

**Raimond Reichert, Samuel Zürcher
Ergon Informatik AG**