

# Workshop: AngularJS

Johannes Lauinger



1. Überblick über Angular
2. Hallo Welt!
  - Plain HTML
  - Reines JavaScript
  - jQuery
  - AngularJS
3. AngularJS ToDo-Liste
4. Vergleich mit Angular 2
5. Fragen

Johannes Lauinger

- Bachelor Informatik im 6. Semester
- IT-Security & Hardware
- Selbstständiger Software-Entwickler und Web-Designer
- Fachschaftsarbeit, Musik selbst machen und hören

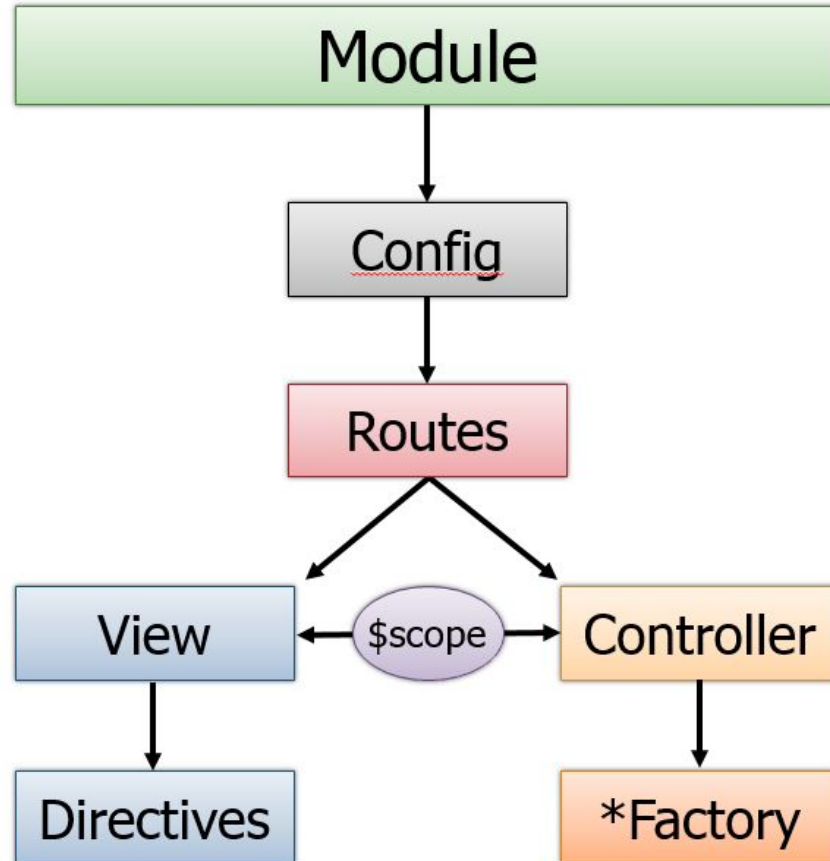
Kontakt:

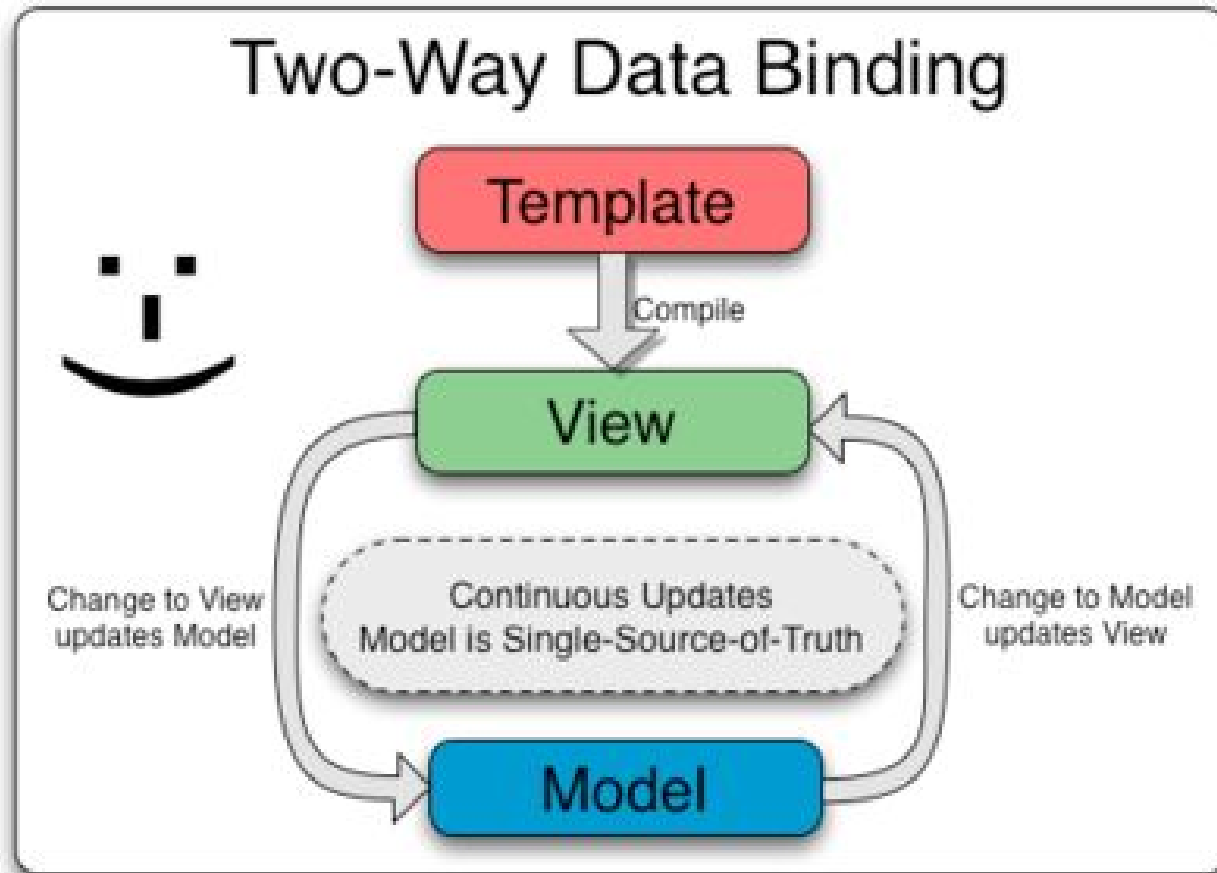
- `johannes@lauinger-it.de`
- `@jlauinger` auf GitHub
- `@realjtl` auf Twitter
- <https://johannes-lauinger.de>

Workshop-Material: <https://github.com/jlauinger/angularjs-workshop>



<https://angularjs.org/>





```
<h1>Hallo!</h1>
```

```
Dein Name: <input type="text" />
```

# Hallo Welt! (Reines JavaScript)

```
<h1>Hallo <span id="label"></span>!</h1>
```

```
Dein Name: <input type="text" id="name" />
```

```
<script>
```

```
    var nameElem = document.getElementById("name");
```

```
    var labelElem = document.getElementById("label");
```

```
    nameElem.addEventListener('keyup', function() {
```

```
        labelElem.innerText = nameElem.value;
```

```
    });
```

```
</script>
```



```
<script src="jquery.min.js"></script>
<script>
    $ ("#name") .on ("keyup", function() {
        $ ("#label") .text ($ ("#name") .val ()) ;
    }) ;
</script>
```

# Hallo Welt! (Mit AngularJS)

```
<!doctype html>

<html ng-app>

<head>...</head>

<body>

    <h1>Hallo {{ name }}!</h1>

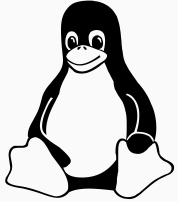
    Dein Name: <input type="text" ng-model="name" />


    <script src="angular.min.js"></script>

</body>

</html>
```

# Exkurs: Entwicklungstools und -umgebung



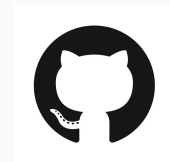
JS



HTML



CSS



- Idee: Grundlegende Konzepte beispielhaft zeigen
- Anschließend kompliziertere Apps bauen
- Funktionen:
  - Liste von Aufgaben
  - Aufgabendetails
  - Aufgaben abhaken
  - Aufgaben hinzufügen und Löschen
  - Daten vom Backend laden

```
{  
  "id": 42,  
  "subject": "Wichtige Aufgabe",  
  "details": "Potenziell unglaublich lange Beschreibung",  
  "done": false  
}
```

```
<body ng-app="todo-app">  
  <div ng-view></div>  
  
  <script src="bower_components/angular/angular.js"></script>  
  ...  
  <script src="app.js"></script>  
  ...  
</body>  
  
var app = angular.module('todo-app', []);
```

```
<ul>  
  <li ng-repeat="item in items">  
    <input type="checkbox" ng-model="item.done" />  
    <a href="#/{{ item.id }}">{{ item.subject }}</a>  
  </li>  
</ul>
```

```
app.config(['$routeProvider', function($routeProvider) {  
    $routeProvider  
        .when('/', {  
            controller: 'listCtrl',  
            templateUrl: 'templates/list.html'  
        })  
        .when('/:id', {  
            controller: 'itemCtrl',  
            templateUrl: 'templates/item.html'  
        });  
}]);
```



```
app.factory('itemsService', ['$http', '$q', function($http, $q) {  
    var items = [];  
    ...  
    return {  
        getItem: getItem,  
        ...  
    };  
}]);
```

GET data/items.json

## Hinzufügen und Löschen: Controller

```
<form>
  <input type="text" ng-model="newSubject" />
  <input type="submit" value="OK" ng-click="addItem(newSubject)" />
</form>
```

```
$scope.addItem = function() {
  if ($scope.newSubject === "") return;
  itemsService.addItem({
    id: itemsService.generateId(), ...
  });
  $scope.newSubject = "";
};
```

# Abstraktion: Eigene Direktiven

```
app.directive('inlineItem', ['itemsService', function(itemsService) {  
    function controller() { ... }  
    return {  
        scope: { item: "=" },  
        link: controller,  
        templateUrl: "templates/inline-item.html"  
    };  
}]);
```

```
<inline-item item="item"></inline-item>
```

## Formatierung: Eigene Filter

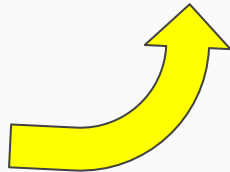
```
return function (input) {  
    input = input || "";  
    for(var i = 0; i < filters.length; i++) {  
        var filter = filters[i];  
        input = input.replace(new RegExp(filter.regex, "g"), filter.replace);  
    }  
    return input;  
};
```

```
<p ng-bind-html="item.details | formatted"></p>
```



<https://angular.io/>

.io vs. .org!



TypeScript

Annotations

Class-based Components

- Statische Typisierung
- Compiler / Transpiler für ES5
- Inline-Compiler verfügbar
- Auto-Complete!

```
let name: string = "Susan",  
    age: number = 25,  
    hasCode: boolean = true;
```

# Vor- und Nachteile von AngularJS 2

## Vorteile:

- (Angeblich) 40x schneller!
- Expliziter Code, etwas bessere Architektur
- TypeScript

## Nachteile:

- Weniger Magie?
- Einstieg weniger intuitiv
- Build-Pipeline

Danke für eure Aufmerksamkeit!

**Fragen?**

<https://github.com/jlauinger/angularjs-workshop>