NOTES & DOMINO ENTWICKLERCAMP

assono

ANGULARJS by Google

und

Domino

EntwicklerCamp

28. März 2017

Innovative Software-Lösungen.

www.assono.de

Bernd Hort

Diplom-Informatiker, Universität Hamburg

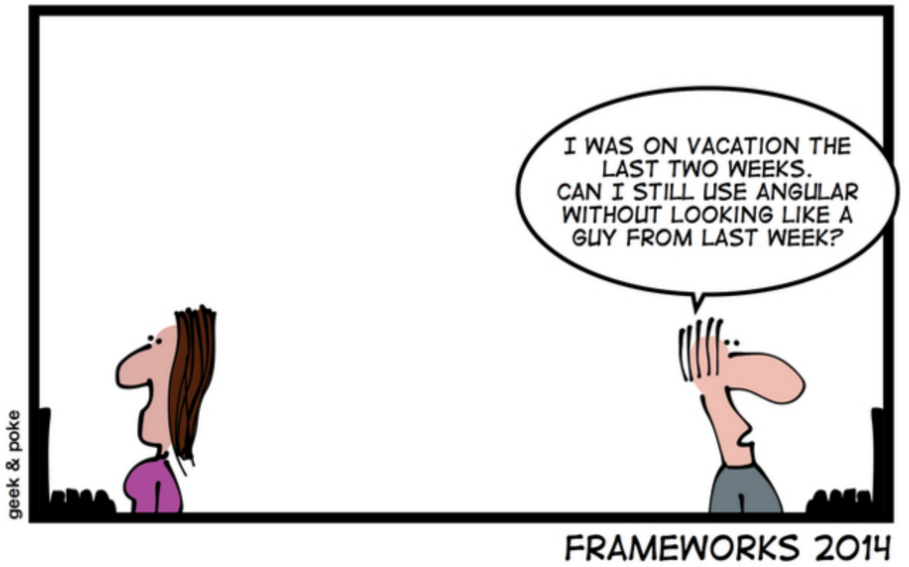
seit 1995 entwickle ich Lotus Notes Anwendungen

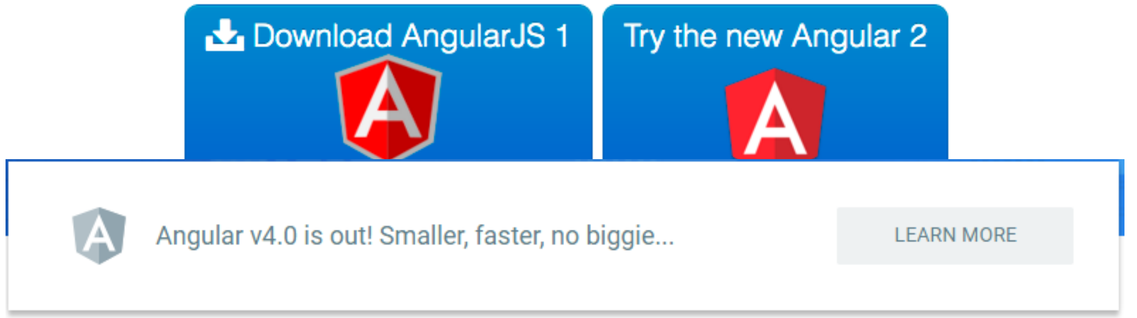
IBM Certified Application Developer, System Administrator & Instructor

Sprecher auf diversen Konferenzen & Lotusphere 2008/IBM Connect 2014/IBM ConnectED 2015/ IBM Connect 2016

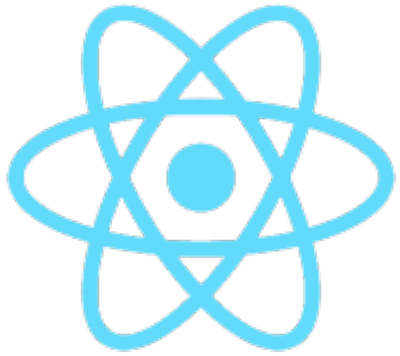
bhort@assono.de
http://www.assono.de/blog
040/73 44 28-315

assono
IT-Consulting & Solutions


I WAS ON VACATION THE LAST TWO WEEKS. CAN I STILL USE ANGULAR WITHOUT LOOKING LIKE A GUY FROM LAST WEEK?
geek & poke
FRAMEWORKS 2014


Download AngularJS 1
Try the new Angular 2
Angular v4.0 is out! Smaller, faster, no biggie...
LEARN MORE

Was ist mit … ?



React

ember

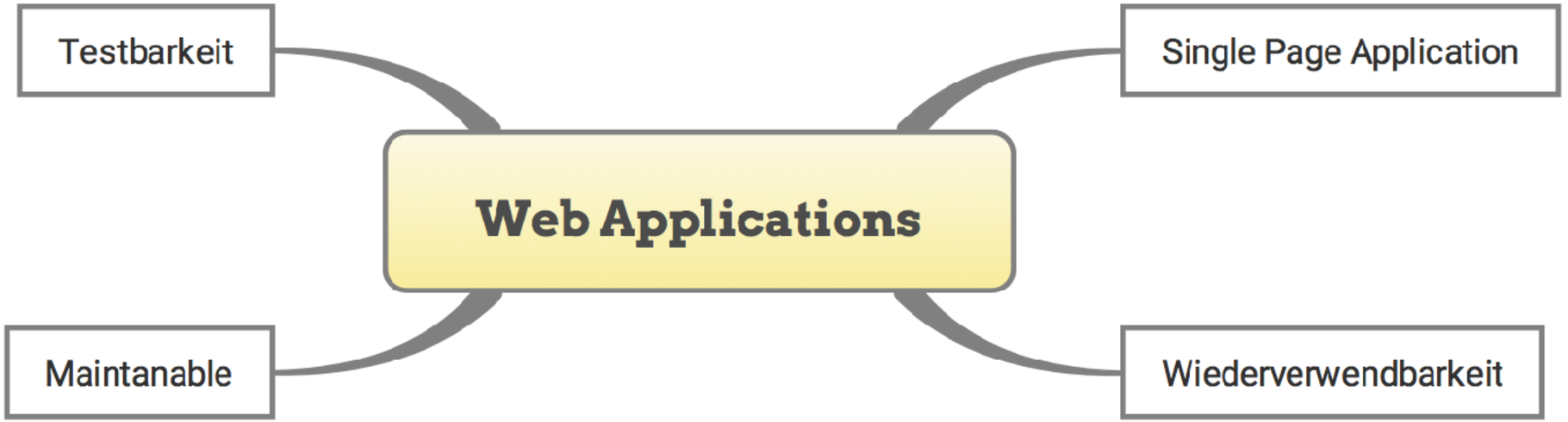
METEOR

5 BEST JAVASCRIPT FRAMEWORKS IN 2017

	AngularJS	Angular 2	ReactJS	Vue.js	Ember.js	Meteor.js
Definition	MVW framework	MVC framework	JavaScript library	MVC framework	MVC framework	JavaScript app platform
1st Release	2009	2016	2013	2014	2011	2012
Homepage	angularjs.org	angular.io	reactjs.net	vuejs.org	emberjs.com	meteor.com
# Contributors on GitHub	1,562	392	912	62	636	328
GitHub Star Rating	54,402	19,832	57,878	39,933	17,420	36,496

https://da-14.com/blog/5-best-javascript-frameworks-2017

Was ist wichtig?

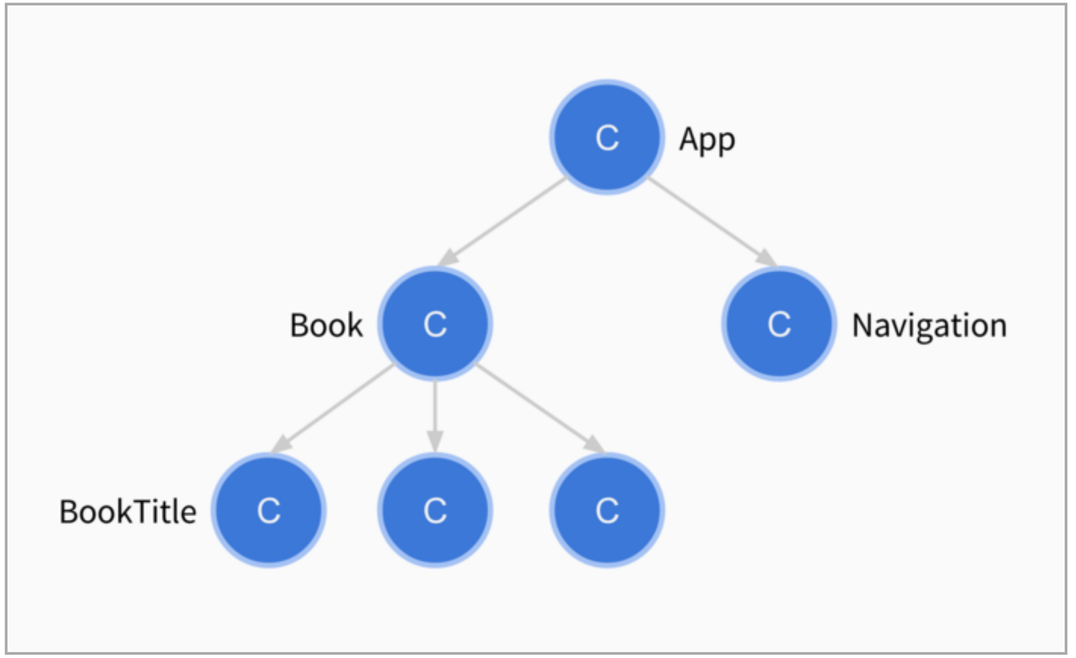


Konzepte, die bleiben …

- Componenten / Templates / Directives
- Trennung zwischen Geschäftsobjekten und Darstellung
- Two-Way Data-binding
- Single Page Application - SPA
- REST
- Dependency Injection

Componenten / Templates / Directives

- Aufteilung in kleinere Blöcke
  - besser testbar
  - höhere Chance auf Wiederverwendbarkeit
- Angular ergänzt HTML um JS-Elemente
  - React schreibt JS, um HTML auszugeben
- Verbindung von HTML und JavaScript, um logische Einheiten zu bilden
  - z.B. Seite einer Single-Page-Application und zugehöriger Controller
  - Eingabe-Block mit Input-Elementen mit Anbindung an Model-Klasse



```
<body ng-app="app">
<div cla="container" ng-controller="  rController">
    <h2>  ber !</h2>
    <p c  ad">Guess the computer generated random number between 1 and 1000.</p>
    <label>Your Guess:  ut type="number" ng-model="guess"/>
    <button ng-click="  " class="btn btn-primary btn-sm">Verify</button>
    <button ng-click="  e()" class="btn btn-warning btn-sm">Restart</button>
    <p>
        <p ng-show="  class="alert alert-warning">Your guess is higher.</p>
        <p ng-show="  class="alert alert-warning">Your guess is lower.</p>
        <p ng-show="deviation===0" class="alert alert-success">Yes! That"s it.</p>
    </p>
    <p class="text-info">No of guesses : <span class="badge">{{noOfTries}}</span><p>
</div>
    <script src="../bower_components/angular/angular.js"></script>
    <script type="text/javascript">
        angular.module('app',[])
        .controller('GuessTheNumberController', GuessTheNumberController);
        function GuessTheNumberController($scope) {
            $scope.verifyGuess = function () {
                $scope.deviation = $scope.original - $scope.guess;
                $scope.noOfTries = $scope.noOfTries + 1;
            }
            $scope.initializeGame=function() {
                $scope.noOfTries = 0;
                $scope.original = Math.floor((Math.random() * 1000) + 1);
                $scope.guess = null;
                $scope.deviation = null;
            }
            $scope.initializeGame();
        }
    </script>
</body>
```

Directive

Directive

Directive

Directive

AngularJS 1.x: Direktiven / Directives

- HTML-Elemente, die von AngularJS mit zusätzlichen Fähigkeiten ausgestattet werden
- Direktiven können Tagnamen, Attribute, Kommentare oder CSS-Klassennamen sein
- Bevorzugt werden Tagnamen und Attribute

```
<my-dir></my-dir>
<span my-dir="exp"></span>
<!-- directive: my-dir exp -->
<span class="my-dir: exp;"></span>
```

- Standard oder selbst definierte Direktriven
- Ablösung in Angular 2.x etc. durch Components

```
<body ng-app="app">
<div class="container  ng-controller=  GuessTheNumberController">
    <h2>Guess the Number !</h2>
    <p class="well lead">Guess the computer generated random number between 1 and 1000.</p>
    <label>Your Guess: </label><input type="number" ng-model="guess"/>
    <button ng-click="verifyGuess()" class="btn btn-primary btn-sm">Verify</button>
    <button ng-click="initializeGame()" class="btn btn-warning btn-sm">Restart</button>
    <p>
        <p ng-show="deviation<0" class="alert alert-warning">Your guess is higher.</p>
        <p ng-show="deviation>0" class="alert alert-warning">Your guess is lower.</p>
        <p ng-show="deviation===0" class="alert alert-success">Yes! That"s it.</p>
    </p>
    <p class="text-info">No of guesses : <span class="badge">{{noOfTries}}</span><p>
</div>
    <script src="../bower_components/angular/angular.js"></script>
    <script type="text/javascript"
        angular.module('app',[])
        .controller('GuessTheNumbe                    berController);
        function GuessTheNumberController($scope) {
            $scope.verifyGuess = function () {
                $scope.deviation = $scope.original - $scope.guess;
                $scope.noOfTries = $scope.noOfTries + 1;
            }
            $scope.initializeGame=function() {
                $scope.noOfTries = 0;
                $scope.original = Math.floor((Math.random() * 1000) + 1);
                $scope.guess = null;
                $scope.deviation = null;
            }
            $scope.initializeGame();
        }
    </script>
</body>
```

AngularJS-Applikation

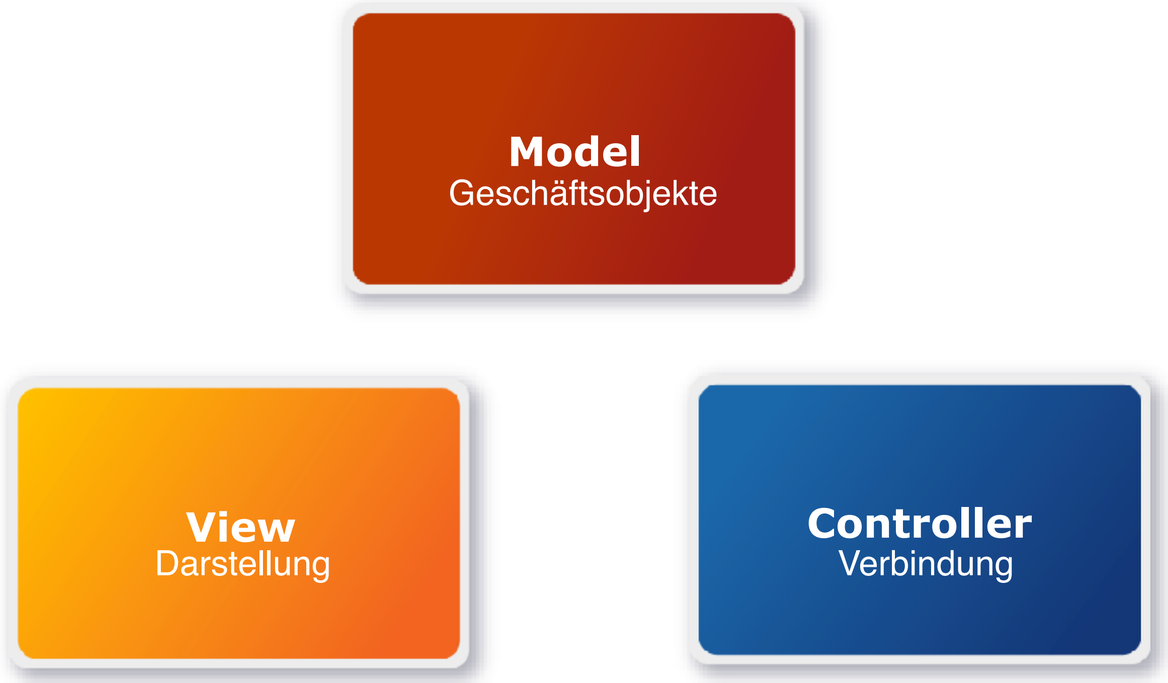
AngularJS-Applikation

AngularJS-Application

- Die Direktive ng-app="{appname}" definiert eine AngularJS-Application.
- Durch die Bindung an ein HTML-Element wird der Gültigkeitsbereich definiert.
- `angular.module('app',[{Dependency Injection}]);`
  definiert das zugehörige Module.
- Über Dependency Injection werden Abhängigkeiten zu anderen Modulen definiert.

Model-View-Controller / Model-View-*Whatever*

- Trennung von Geschäftsobjekten und Darstellung



```
<body ng-app="app">
<div class="container" ng-controller="GuessTheNumberController">
    <h2>Guess the Number !</h2>
    <p class="well lead">Guess the computer generated random number between 1 and 1000.</p>
    <label>Your Guess: </label><input type="number" ng-model="guess"/>
    <button ng-click="verifyGuess()" class="btn btn-primary btn-sm">Verify</button>
    <button ng-click="initializeGame()" class="btn btn-warning btn-sm">Restart</button>
    <p>
        <p ng-show="deviation<0" class="alert alert-warning">Your guess is higher.</p>
        <p ng-show="deviation>0" class="alert alert-warning">Your guess is lower.</p>
        <p ng-show="deviation===0" class="alert alert-success">Yes! That"s it.</p>
    </p>
    <p class="text-info">No of guesses : <span class="badge">{{noOfTries}}</span><p>
</div>
    <script src="../bower_components/angular/angular.js"></script>
    <script type="text/javascript">
        angular.module('app',[])
        .controller('GuessTheNumberController', GuessTheNumberController);
        function GuessTheNumberController($scope) {
            $scope.verifyGuess = function () {
                $scope.deviation = $scope.original - $scope.guess;
                $scope.noOfTries = $scope.noOfTries + 1;
            }
            $scope.initializeGame=function() {
                $scope.noOfTries = 0;
                $scope.original = Math.floor((Math.random() * 1000) + 1);
                $scope.guess = null;
                $scope.deviation = null;
            }
            $scope.initializeGame();
        }
    </script>
</body>
```

Controller

Controller

Controller

- Die Direktive ng-controller="{ControllerName}" definiert den Controller.
- Durch die Bindung an ein HTML-Element wird der Gültigkeitsbereich definiert.
- `angular.module('app',[{Dependency Injection}]).controller('GuessTheNumberController', function($scope) {});`
  registriert den Controller im Module.

```
<body ng-app="app">
<div class="container" ng-controller="GuessTheNumberController">
    <h2>Guess the Number !</h2>
    <p class="well lead">Guess the computer generated random number between 1 and 1000.</p>
    <label>Your Guess: </label><input type="number" ng-model="guess"/>
    <button ng-click="verifyGuess()" class="btn btn-p                    m">Verify</button>
    <button ng-click="initializeGame()" class="btn bt                    n-sm">Restart</button>
    <p>
        <p ng-show="deviation<0"                    alert-warning">Your guess is higher.</p>
        <p ng-show="deviation>0"                    alert-warning">Your guess is lower.</p>
        <p ng-show="deviation===0" class="alert alert-success">Yes! That"s it.</p>
    </p>
    <p class="text-info">No of guesses : <span class="badge">{{noOfTries}}</
</div>
    <script src="../bower_components/angular/angular.js"></script>
    <script type="text/javascript">
        angular.module('app',[])
        .controller('GuessTheNumberController', GuessTheNumberController);
        function GuessTheNumberController($scope) {
            $scope.verifyGuess = function () {
                $scope.deviation = $scope.original - $scope.guess;
                $scope.noOfTries = $scope.noOfTries + 1;
            }
            $scope.initializeGame=function() {
                $scope.noOfTries = 0;
                $scope.original = Math.floor((Math.random() * 1000) + 1);
                $scope.guess = null;
                $scope.deviation = null;
            }
            $scope.initializeGame();
        }
    </script>
</body>
```

Model

Model

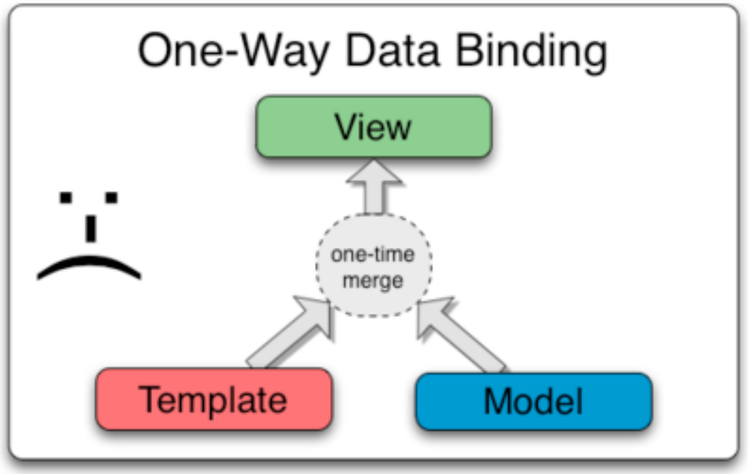
Model

Model

- Das Model wird über die `$scope`-Variable des Controllers zugänglich gemacht.
- Input-Elemente werden mit der Direktive
  `ng-model="{Variablenname}"`
  an das Model gebunden.
- Angular verwendet ein Two-Way Datebinding.

One-Way Data Binding

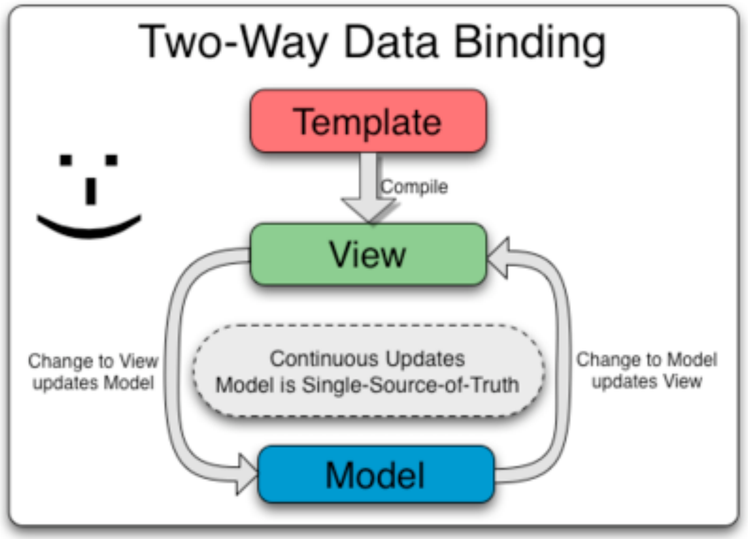
- Andere JavaScript Frameworks bringen zum Zeitpunkt des Renderings die Daten aus dem Model mit einer Vorlage zusammen.
- Änderungen aus der View (HTML-Seite) haben keine direkte Auswirkung auf das Model.



Grafik dem AngularJS Tutorial entnommen: https://docs.angularjs.org/guide/databinding

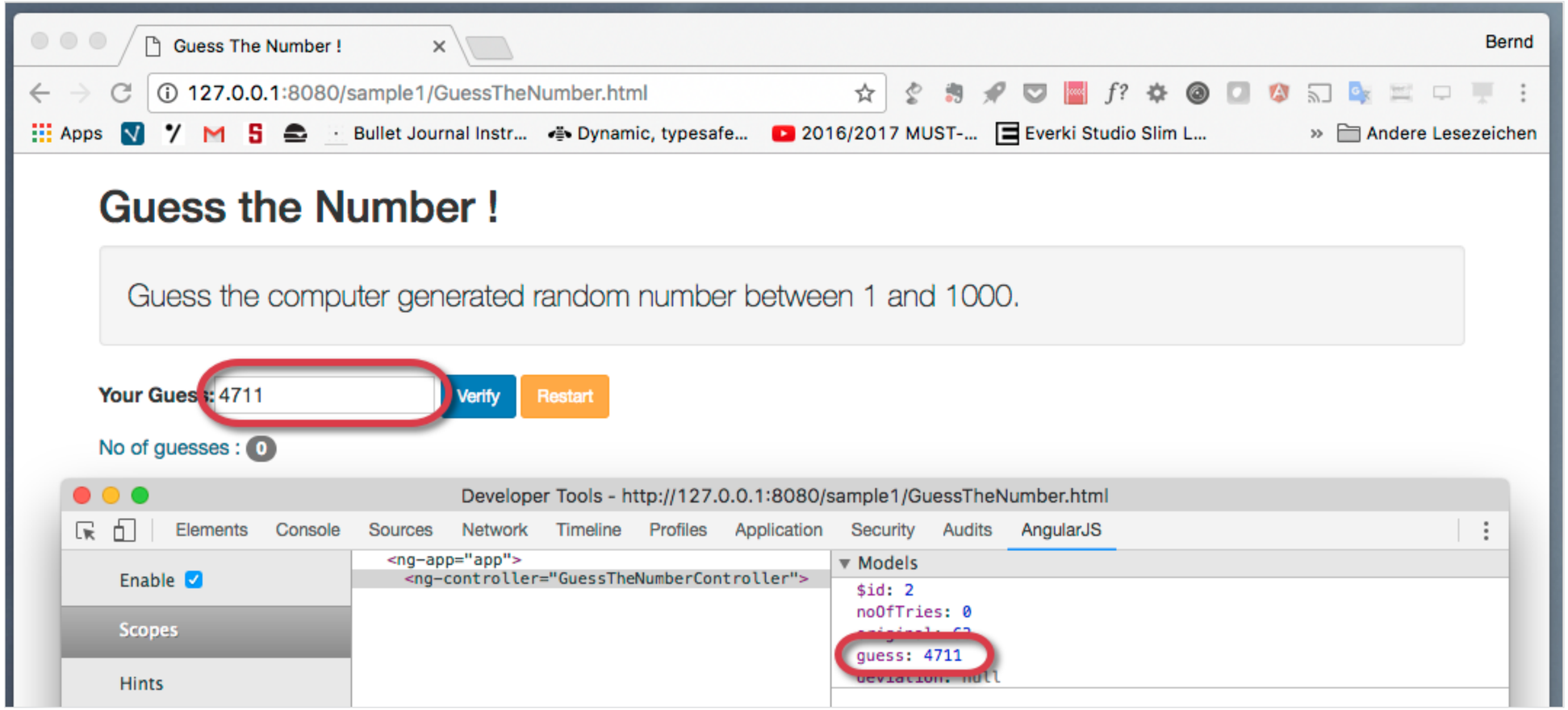
Two-Way Data Binding

- In Angular wirken sich Änderungen in HTML-Input-Elementen unmittelbar auf das Model und vice versa aus.
- Single-Source-of-Truth!

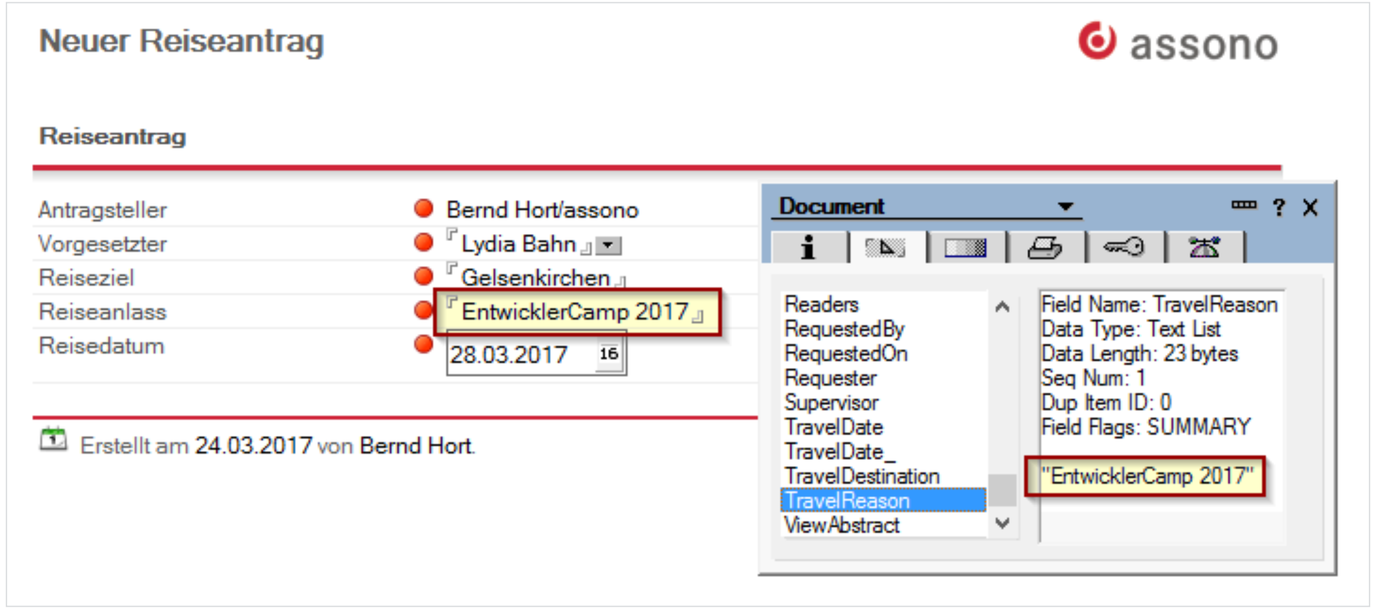


Grafik dem AngularJS Tutorial entnommen: https://docs.angularjs.org/guide/databinding

Neuer Wert unmittelbar im Model sichtbar



Ganz wie in Notes ;-)



Angular Expression

- Angular Expression sind kleine Code Snippets.
- Meistens werden sie in doppelte geschweifte Klammern
  `<span class="badge">{{noOfTries}}</span>`
  gepackt.
- Angular Expressions unterscheiden sich von JavaScript Expressions durch
  - Kontext bezieht sich auf den aktuellen Scope anstelle des Window Objektes
  - Fehlertoleranter
  - Keine Schleifen, If-Abfragen und ähnliches
  - Nicht anfällig für Cross-Site Scripting (xss)
  - Angular Filter können benutzt werden

One-Time Binding

- Die bidirektionale Datenbindung bedeutet einen gewissen Overhead.
- Wird dieser permanente Abgleich nicht benötigt, kann eine so genannte One-Time Binding eingesetzt werden.
- In der Expression wird durch vorgestellte zwei Doppelpunkte signalisiert, dass der Wert nur einmal beim Laden der Seite ausgewertet werden soll.
  ```
  <h1>{{::model.title}}</h1>
  ```
- Vergleichbar in XPages mit
  - Compute Dynamically
    ```
    value="#{javascript:database.getTitle()}"
    ```
  - Compute on Page Load
    ```
    value="${javascript:database.getTitle()}"
    ```

Eingabevalidierung im Frontend

- Unterstützung von require Eigenschaft der HTML Input-Elemente
- AngularJS ng-messages für die Anzeige von Eingabevalidierungsnachrichten
- ng-required für Eingabevalidierung bei bestimmten Bedingungen
- Überprüfung im model, ob alle notwendigen Bedingungen erfüllt sind, bevor das model gespeichert wird.
  - Vergleichbar mit einem QuerySave

```
angular.module('app')
    .controller('ProjectDetailController', ['$scope', 'ProjectService', '$routeParams', 'ProjectBuilderService', '$location',
        function($scope, ProjectService, $routeParams, ProjectBuilderService, $location) {

            $scope.message = "";
            $scope.warning = "";
            $scope.error = "";

            $scope.save = function() {
                $scope.submitted = true; //
                $scope.warning = "";
                if ($scope.formProject.$invalid) return;
                var validationResult = $scope.project.isValid();
                if (!validationResult.isValid) {
                    $scope.warning = validationResult.messages;
                    return;
                }

                ProjectBuilderService.save().then(
                    function(createdProject) {
                        $scope.formProject.$setPristine();
                        $scope.submitted = false;
                        $scope.project = createdProject;
                        $scope.warning = "";

                        $scope.message = "Project '" + createdProject.projectId + "' created.";


                        $location.path('/projects');
```

ng-Validierung

model-Validierung

```
angular.module('app')
    .factory('Project', function () {
        function Project(args) {
            this.project = args.project;
            this.projectId = args.projectId;
            this.customer = args.customer;
            this.customerId = args.customerId;

            this.isValid = function(message) {
                var validationResult = {
                    isValid: true,
                    messages: []
                };

                if(this.projectId!==null){
                  if(/^[A-Za-z0-9 ]{1,15}$/m.test(this.projectId)===false){
                    validationResult.isValid = false;
                    validationResult.messages.push('Only alpha numberic values are allowed in project id with max length 15.');
                  }
                }

                return validationResult;

            };

        }
        return Project;
    });
```

Eingabevalidierung auch im Backend!

- Immer auch im Backend die übertragenen Daten auf Gültigkeit prüfen.
- Auf der Server-Seite kann nicht unterschieden werden, ob der REST-Aufruf aus Ihrer Anwendung oder von einer anderen Web-Anwendung stammt.
- **Vertrauen Sie niemanden!**

Single Page Application - SPA

- Bei SPA wird initial nur minimaler HTML Code geladen.
- Jegliche weitere Inhalte werden entweder
  - per JavaScript berechnet
  - per AJAX nachgeladen.
- In Abhängigkeit vom Anwendungsfluss werden Teile ausgetauscht bzw. nachgeladen.
- Herausforderungen sind
  - Lesezeichen setzen
  - Back-Button
  - Navigation zwischen verschiedenen „Seiten“

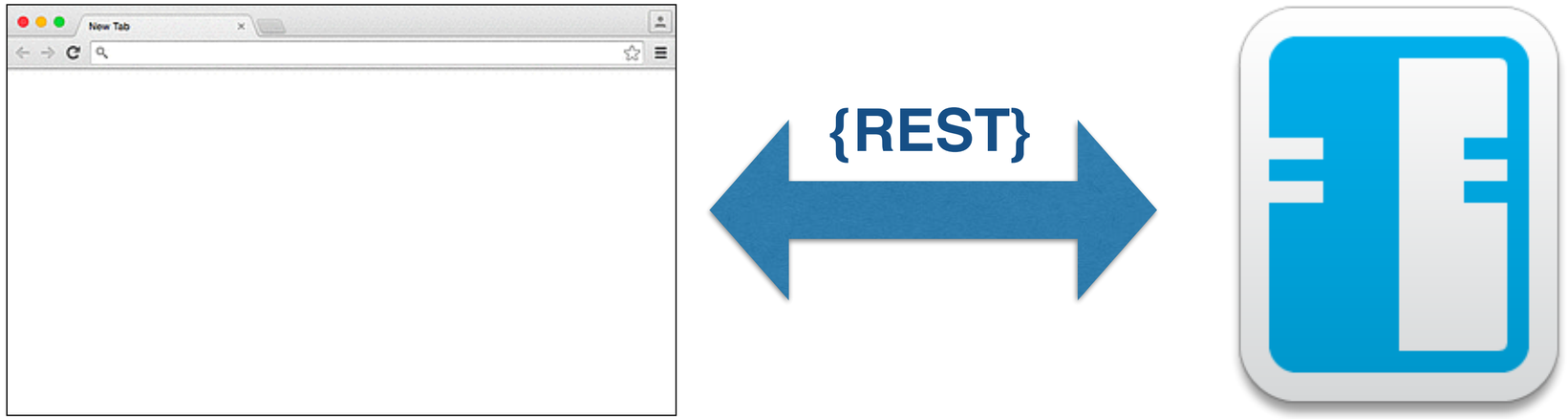
Routen - ngRoute

- In Angular wird diese Aufgabe vom $route Service übernommen.
- Der zugehörige Code muss als separate JS-Datei „angular-route.js“ geladen werden.
- Als Depedency muss „ngRoute“ definiert werden.
- Die Wege zwischen den Teilseiten werden als sogenannte Routen definiert.

```
angular.module('app').
config(function ($routeProvider) {

    $routeProvider.when('/projects', {
      templateUrl: 'partials/projects.html',
      controller: 'ProjectListController' });
    $routeProvider.when('/project/new', {
      templateUrl: 'partials/project.html',
      controller: 'ProjectDetailController'});
    $routeProvider.when('/project/:id', {
      templateUrl: 'partials/project.html',
      controller: 'ProjectDetailController'});


    $routeProvider.otherwise({ redirectTo: '/projects' });

});
```

REST-API

- Die Anwendungslogik wird nur im Browser ausgeführt.
- Datenaustausch findet nur via REST API statt.



EntwicklerCamp Vortrag 2016: REST Services in Domino

Server Kommunikation

- Für die Kommunikation mit dem Server gibt es zwei primäre Services in Angular
- $http - AJAX Basis-Operationen
- $resource - Höhere Abstraktionsebene für die Verwendung von REST
  - $resource basiert auf $http

```
angular.module('app')
    .factory("ProjectService", ['RESTPATH','Project','$http','$q', function (RESTPATH, Project, $http, $q) {
        var service = {};
        var projects = [];
        service.getProjects = function () {
            var deferred = $q.defer();

            $http.get(RESTPATH + 'project/').then(
                function(result){
                if (result.data.status==='success'){
                    angular.forEach(result.data.data, function (p, index) {

                        projects.push(
                            new Project(p));
                    });
                    deferred.resolve(projects);
                } else {
                    deferred.reject(result.data);
                }
            }, function(error){
                console.log(error);
                deferred.reject(error);
            });

            //    Now return the promise.
            return deferred.promise;

        };
```

Promise erzeugen

GET starten

Function Rückgabe Daten

Erfüllen des Promise

Fehlerfall

Fehlerfall

Promise zurückgeben

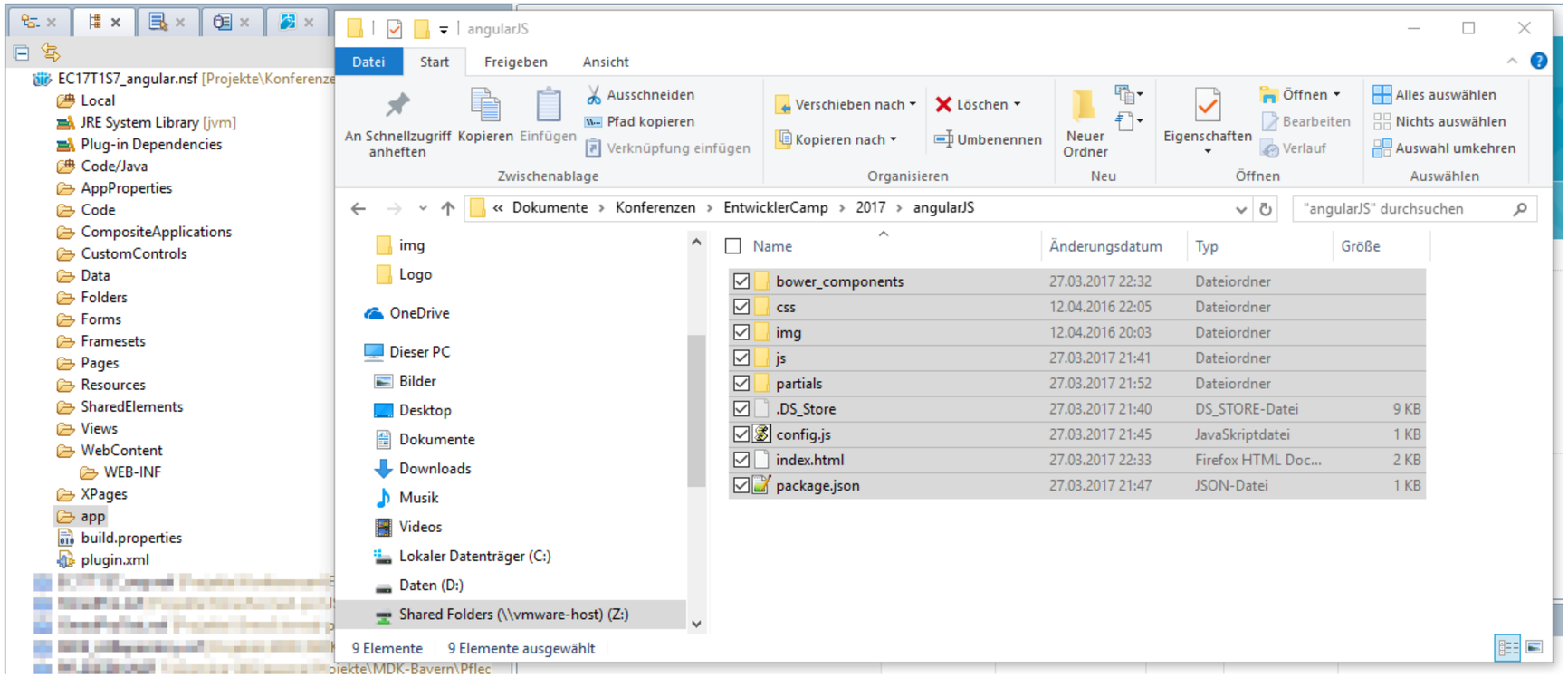
```
$scope.save = function() {
    $scope.submitted = true; // Will force validations
    $scope.warning = "";
    if ($scope.formProject.$invalid) return;
    var validationResult = $scope.project.isValid();
    if (!validationResult.isValid) {
        $scope.warning = validationResult.messages;
        return;
    }

    ProjectBuilderService.save().then(
        function(createdProject) {
            $scope.formProject.$setPristine();
            $scope.submitted = false;
            $scope.project = createdProject;
            $scope.warning = "";

            $scope.message = "Project '" + createdProject.projectId + "' created.";

            $location.path('/projects');

        },
        function(failureResult) {
            $scope.warning = failureResult;
            $scope.message = "";

        }
    );
};
```

Erfüllen des Promise im Controller

Fehlerfall im Controller

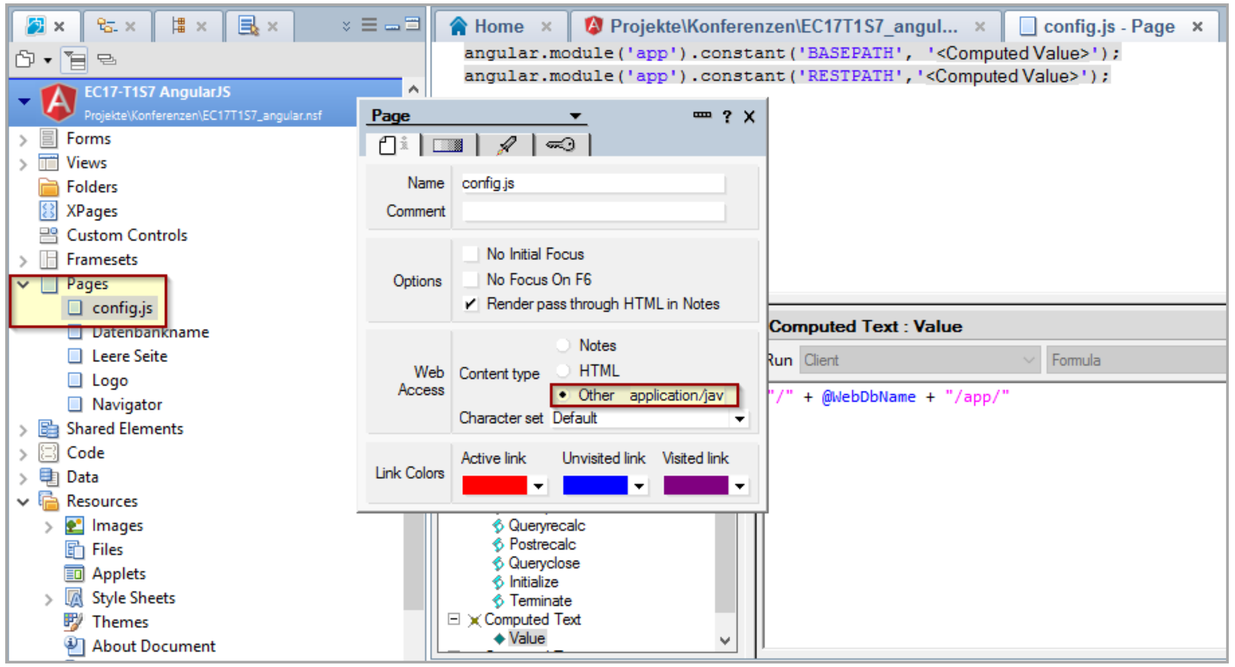
AngularJS in Notes Datenbank

- Über den Package-Explorer Ordnerstruktur in die Notes Anwendung ziehen



AngularJS in Notes Datenbank - Page config.js

- Pfade zur aktuellen Notes-Datenbank und ähnliche Informationen als Page mit Content-Type „application/javascript“ und „Computed Text“



Fragen?

jetzt stellen – oder später:

bhort@assono.de

http://www.assono.de/blog

040/73 44 28-315

assono
IT-Consulting & Solutions

Folien und Beispiel-Datenbank unter
www.assono.de/blog/EntwicklerCamp-2017-AngularJS