

# WEBAPPS WITH ANGULARJS

AN OVERVIEW OF THE FRAMEWORK



Henk Bakker



# WHAT IS ANGULARJS

It's a structural framework for dynamic web apps that's easy to maintain, in a fast and testable way.

# THE PRINCIPLES

- Rapid development
- Modularity
- Built to be testable
- Write less code

# WHAT IT OFFERS US

- Controllers
- Templates
- Two-Way data bindings
- Services
- Directives
- Dependency Injection

# THE SIMPLEST EXAMPLE

```
<!DOCTYPE html>
<html ng-app>
  <head>
    <script src="js/angular.min.js"></script>
  </head>
  <body>
    <form>
      <label>Your name?</label>
      <input type="text" ng-model="name" placeholder="Your name?">
    </form>
    <h3>My name is {{ name }}!</h3>
  </body>
</html>
```

---

What's your name?

What's your name?

MY NAME IS !

# \$SCOPE

**\$scope** is an object that refers to the application model.  
It connects the **View** and the **Controller**

myController.js

```
function MyController($scope) {  
  $scope.name = 'Henk Bakker';  
}
```

myView.html

```
<form ng-controller="MyController">  
  <input type="text" ng-model="name">  
</form>
```

# CONTROLLERS

The controller is a function that augment the **\$scope** object.  
It's used to add a value or to add a behavior to the **\$scope**.

```
function TodoCtrl($scope) {  
  $scope.todos = [  
    { text: 'Learn AngularJS', done: true },  
    { text: 'Create an App', done: false }  
  ];  
  
  $scope.addTodo = function () {  
    $scope.todos.push({ text: $scope.todoText, done: false });  
    $scope.todoText = '';  
  };  
};
```



# TEMPLATES

The templates are simply **HTML5**.  
All the presentation logic of the app must be in there.

```
<div ng-controller="TodoCtrl">
  <ul>
    <li ng-repeat="todo in todos">
      <input type="checkbox" ng-model="todo.done">
      <span>{{ todo.text }}</span>
    </li>
  </ul>
  <form ng-submit="addTodo()">
    <input type="text" ng-model="todoText">
    <button type="submit">Add</button>
  </form>
</div>
```

# RESULT

- ☒ ~~Learn AngularJS~~
- ☐ Create an App

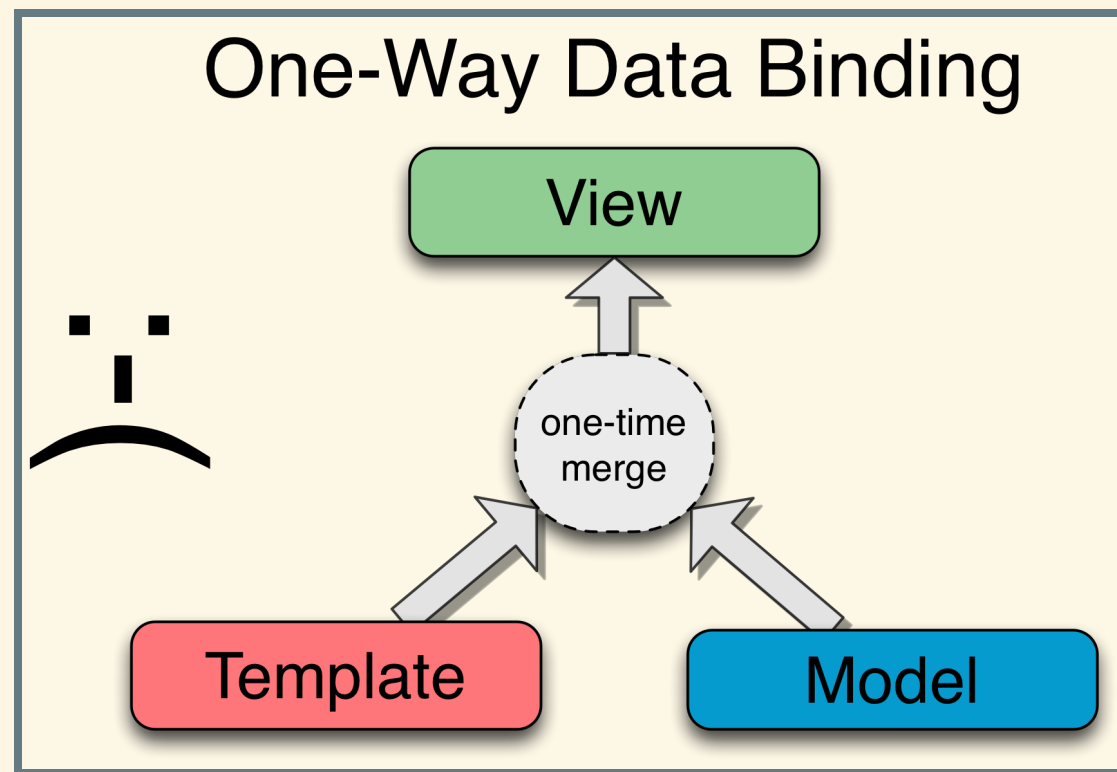
add new task

Add

```
[{"text": "Learn AngularJS", "done": true}, {"text": "Create an App", "done": false}]
```

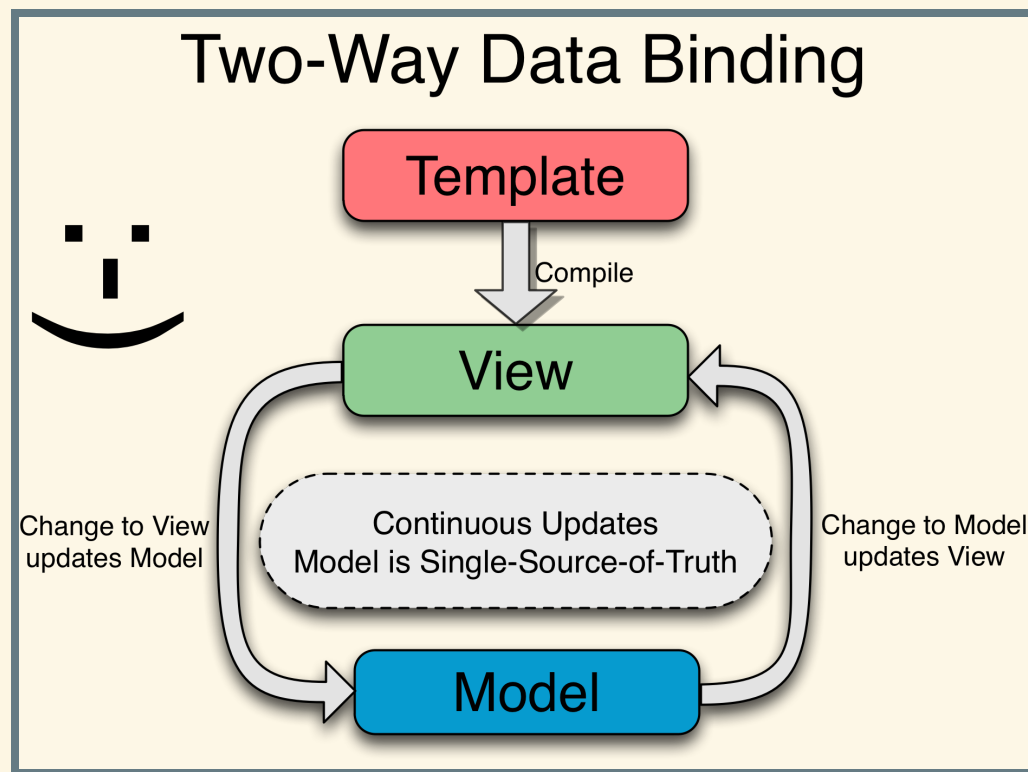
# ONE-WAY DATA BINDING

Most templating systems bind data in only one direction: they merge template and model components together into a view



# TWO-WAY DATA BINDING

Two-way data binding is the automatic synchronization of data between the model and view.



# \$SCOPE INHERITANCE

When a new `$scope` is created, it's added as a child of its parent `$scope`.

MyControllers.js

```
function ParentController($scope) {
  $scope.person = {
    name      : 'Henk Bakker',
    helloText: ''
  };
};

function ChildController($scope) {
  $scope.sayHello = function () {
    $scope.person.helloText = "Hi " + $scope.person.name;
  }
};
```

# \$SCOPE INHERITANCE 2

MyView.html

```
<div ng-controller="ParentController">
  <form ng-controller="ChildController">
    <input type="text" ng-model="person.name" placeholder="Name">
    <button ng-click="sayHello()">Say hello</button>
  </form>
  <p>{{ person.helloText }}</p>
</div>
```

person.helloText =

# SERVICES

Injectable objects that carry out specific tasks.  
Provide a way to separate concerns and re-use code.

- \$http (ajax)
- \$locale
- \$timeout
- \$filter

# \$HTTP

The **\$http** service makes easier to integration with external APIs.

Main methods: .get, .post, .put, .delete, .jsonp

```
$http.get('http://address-of.the/api')  
  .success(successCallback)  
  .error(errorCallback);  
  
$http.post('http://address-of.the/api', data)  
  .success(successCallback)  
  .error(errorCallback);
```



# \$HTTP - EXAMPLE

```
function BeerController($scope, $http) {
  $scope.makeRequest = function () {
    $http.jsonp('http://api.openbeerdatabase.com/v1/beers.json')
      .success(function (data, status) {
        $scope.beers = data.beers;
      })
      .error(function () {
        $scope.beers = [{ name: "No beer for you :(" }];
      });
  };
};
```

```
<div ng-controller="BeerController">
  <ul>
    <li ng-repeat="beer in beers | limitTo:15">
      {{ beer.name }} - {{ beer.brewery.name }}
    </li>
  </ul>
  <button ng-click="makeRequest()">Get beers</button>
</div>
```

# \$HTTP - RESULT

Get beers

# MODULES

Angular **modules** declaratively specify how an application should be bootstrapped.

```
var app = angular.module('myApp', ['third-part-module']);

app.config(function () { ... });      <!-- Configuration -->
app.controller('myController', ...);  <!-- Create a controller -->
app.service('myService', ...);        <!-- Create a service -->
```

```
<html ng-app="myApp">
  ...
</html>
```

# ROUTING

In a more complex app you can use the *\$routeProvider* service, to define which *controller* and *template* will be loaded in each *path*.

```
var app = angular.module('myApp');

app.config(function ($routeProvider) {
  $routeProvider
    .when('/', {
      controller : 'mainController', templateUrl : '/views/index.html'
    })
    .when('/newPost/', {
      controller : 'newPostController', templateUrl : '/views/newPost.html'
    })
    .when('/posts/:id', {
      controller : 'postsController', templateUrl : '/views/posts.html'
    })
    .otherwise({ redirectTo : '/' });
});
```

# DIRECTIVES

*"Teach new tricks to the HTML"*

1. Create custom attributes
2. Create custom HTML tags  
(based on W3C *webcomponents* specification)

# DIRECTIVES <sup>2</sup>

All the attributes that begin with "ng" are AngularJS directives.

- ng-app
- ng-controller
- ng-model
- ng-repeat
- ng-click
- ng-view
- ...

# DIRECTIVES 3

## How to use *directives*

```
<!doctype html>
<html ng-app="myApp">
  <head>
    <script src="../js/angular.min.js"></script>
  </head>
  <body>
    <ul ng-controller="myListController">
      <li ng-repeat="item in items">
        {{ item.name }}
      </li>
    </ul>
  </body>
</html>
```

# DIRECTIVES 4

You can also create you own *directive*.

- **element:** `<my-directive></my-directive>`
- **atribute:** `<span my-directive="value"></span>`
- **class:** `<span class="my-directive: value;"></span>`
- **comment:** `<!-- directive: my-directive value -->`



# DIRECTIVES 5

```
var app = angular.module('myApp');

app.directive('myDirective', function () {
  return {
    restrict: 'EA',
    link: function ($scope, element) {
      element.text('Text from directive');
    }
  };
});
```

```
<my-directive> </my-directive>  <!-- Directive as element -->

<div my-directive> </div>       <!-- Directive as attribute -->
```

# DIRECTIVES 6

```
var app = angular.module('myApp');

app.directive('myDirective', function () {
  return {
    restrict: 'E',
    replace: true,
    template: '<h1 class="title"></h1>'
    link: function ($scope, element) {
      element.text('Text from directive');
    }
  };
});
```

```
<my-directive> </my-directive>
```

```
<!-- Custom tag -->
```

```
<h1 class="title">Text from directive</h1> <!-- Replaced by the template -->
```

# TESTING YOUR APP

A framework to be easily tested

## TOOLS:

- **Karma** Test runner
- **Jasmine** Test framework

# TESTING A CONTROLLER

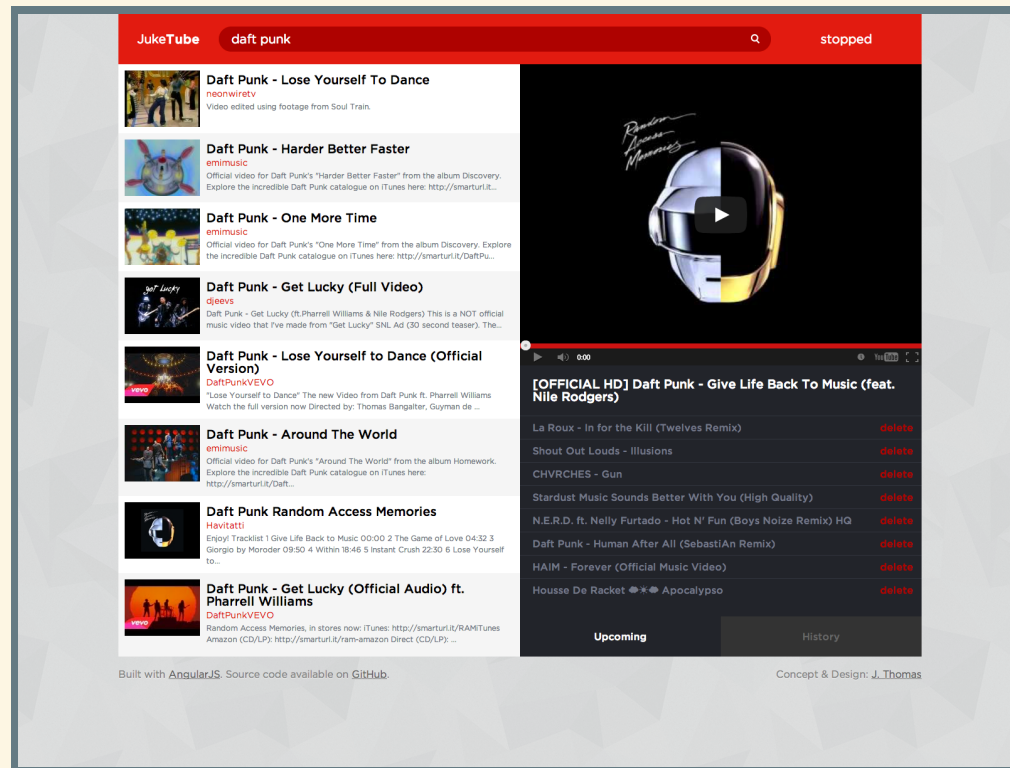
```
describe('Testing Controller', function () {  
  var ctrl, scope;  
  
  beforeEach(angular.mock.module('myApp'));  
  
  beforeEach(inject(function ($controller, $rootScope) {  
    scope = $rootScope.$new();  
    ctrl = $controller('myController', { $scope: scope });  
  }));  
  
  it('should exist a controller called myController', function() {  
    expect(scope).not.toBeUndefined();  
  });  
});
```

# END-TO-END TESTS

```
describe('Grocery list', function () {  
  beforeEach(function () {  
    browser().navigateTo('/');  
  });  
  
  it('filters the grocery list based on the search query', function() {  
    expect(repeater('.groceries li').count()).toBe(5);  
  
    input('query').enter('b');  
    expect(repeater('.groceries li').count()).toBe(3);  
  
    input('query').enter('blueberry');  
    expect(repeater('.groceries li').count()).toBe(1);  
  });  
});
```

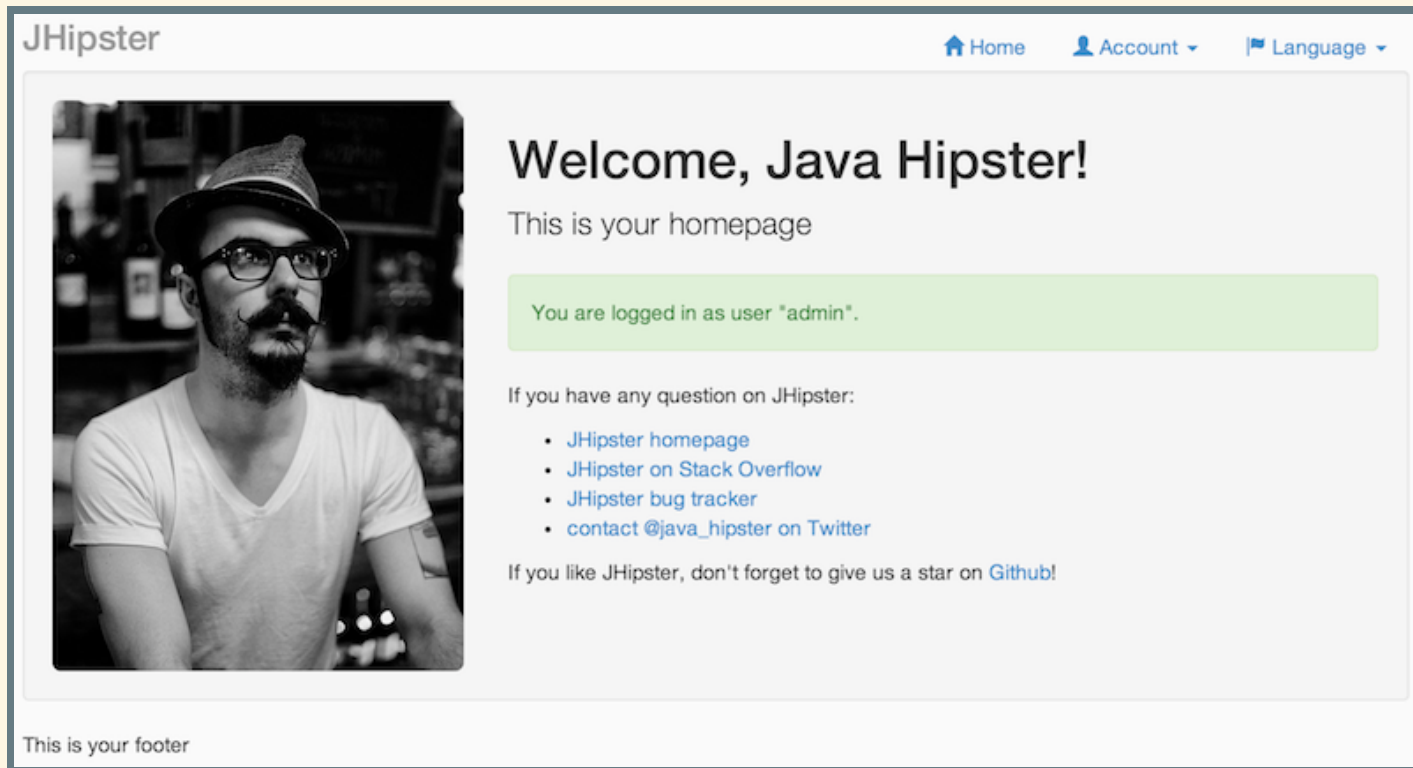
# DEMO TIME

# JUKETUBE



Demo | Source

# JHIPSTER



## Homepage



# IONIC



Homepage

# LEARN

- Thinkster - <http://thinkster.io/>
- Year of Moo - <http://www.yearofmoo.com/>
- HTML5 Rocks - <http://html5rocks.com/>
- TodoMVC - <http://todomvc.com/>

# REFERENCES

- Angular - <http://angularjs.org/>
- Karma - <http://karma-runner.github.io/>
- Jasmine - <http://pivotal.github.io/jasmine/>

**THE END**

**HENK BAKKER**