

CRIANDO WEBAPPS

OVERVIEW DE DESENVOLVIMENTO COM **ANGULARJS**



- Telefonica Digital

O QUE É O **ANGULARJS**

Um framework para criar aplicações web dinâmicas, de maneira rápida, testável e de fácil manutenção.

PRINCÍPIOS DO **ANGULARJS**

- Desenvolvimento rápido
- Escrever menos código
- Modularidade
- Pensado para ser testado

O QUE O **ANGULARJS** OFERECE

- Controllers
- Templates
- Two-Way data bindings
- Services
- Directives
- Integração com REST
- Injeção de dependências

UM EXEMPLO BEM SIMPLES

```
<!doctype html>
<html ng-app>
  <head>
    <script src="js/angular.min.js"></script>
  </head>
  <body>
    <form>
      <label>Seu nome?</label>
      <input type="text" ng-model="name" placeholder="Seu nome?">
    </form>
    <h3>Meu nome é {{ name }}!</h3>
  </body>
</html>
```

MODEL

O modelo da aplicação é representado pelo objeto **\$scope**, que liga a *View* e o *Controller*.

myController.js

```
function MyController($scope) {  
  $scope.name = 'Vitor Leal';  
}
```

myView.html

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

CONTROLLERS

São funções usadas para "aumentar" a instância do **\$scope**.
Adicionando um valor ou um comportamento a um objeto.

```
function TodoCtrl($scope) {  
  $scope.todos = [  
    { text: 'Aprender AngularJS', done: true },  
    { text: 'Criar um novo App', done: false }  
  ];  
  
  $scope.addTodo = function () {  
    $scope.todos.push({ text: $scope.todoText, done: false });  
    $scope.todoText = '';  
  };  
};
```


VIEWS

As views no angular são simplesmente **HTML5**.
É onde toda a *presentation logic* deve estar.

```
<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>
```

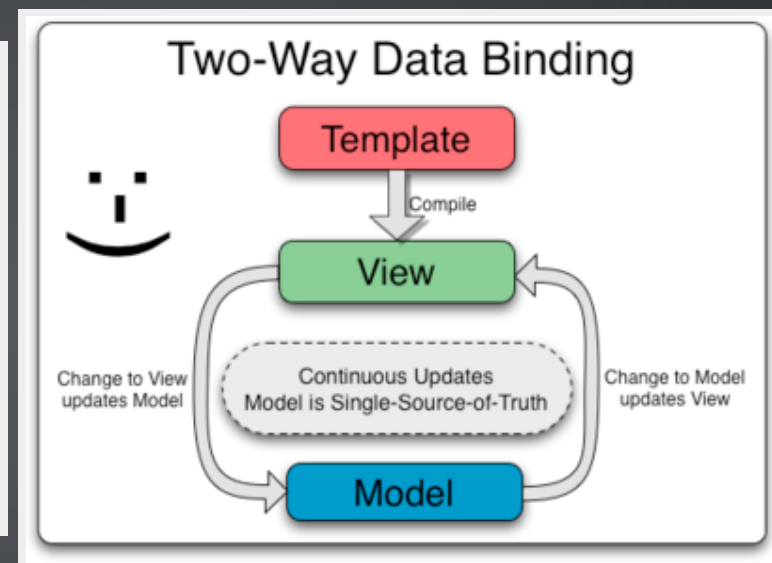
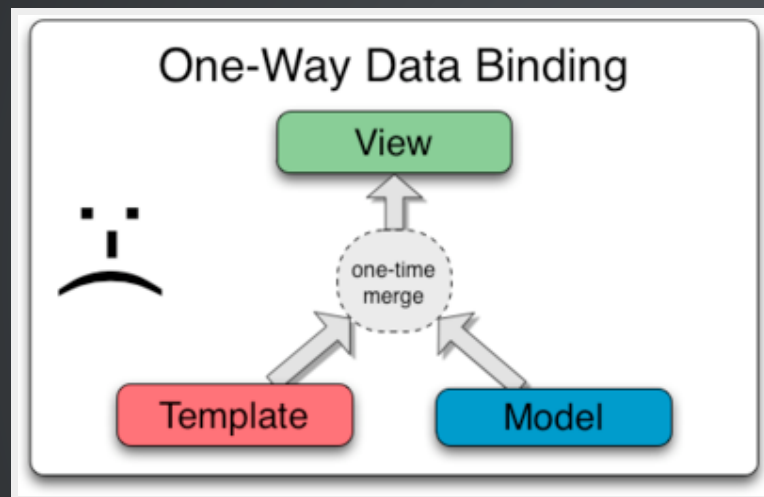
RESULTADO

- ✓ Aprender AngularJS
- Criar um novo App

```
[{"text": "Aprender AngularJS", "done": true}, {"text": "Criar um novo App", "done": false}]
```

TWO-WAY DATA BINDING

Data binding no angular funciona de maneira bi-direcional, qualquer alteração na *View* se reflete no *Model* e vice-versa.



HERANÇA DE \$SCOPE

O **\$scope** de um *controller* é herdado em um *controller* filho.

MyControllers.js

```
function ParentController($scope) {  
  $scope.person = {  
    name      : 'Vitor Leal',  
    helloText: ''  
  };  
};  
  
function ChildController($scope) {  
  $scope.sayHello = function () {  
    $scope.person.helloText = "Hi " + $scope.person.name;  
  }  
};
```

HERANÇA DE \$SCOPE 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>
```

preson.helloText =

SERVICES

São objetos que executam tarefas específicas e que são controlados pela injeção de depêndencias.

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

\$HTTP

O service **\$http** facilita a comunicação com APIs.

Principais métodos: .get, .post, .put, .delete, .jsonp

```
$http.get('http://endereco.da/api')  
  .success(successCallback)  
  .error(errorCallback);  
  
$http.post('http://endereco.da/api', data)  
  .success(successCallback)  
  .error(errorCallback);
```

\$HTTP - EXEMPLO

```
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 - RESULTADO

Get beers

MODULES

No angular voce utiliza o "*module*" para especificar como um aplicativo deve ser inicializado.

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

app.config(function () { ... });      <!-- Configurações -->
app.controller('myController', ...);  <!-- Criando um controller -->
app.service('myService', ...);       <!-- Criando um service -->
```

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

ROUTES

Em apps mais complexos utilize o *\$routeProvider* do angular, para definir qual *controller* e *template* deve ser utilizado para cada *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

"Ensine novos truques ao HTML"

1. Crie atributos customizados para tags HTML
2. Crie tags HTML customizadas
(baseado na especificação *webcomponents* da W3C)

DIRECTIVES ²

Todos os atributos que iniciam com "ng" são *directives* criadas pelo Angular.

- ng-app
- ng-controller
- ng-model
- ng-repeat
- ng-click
- ng-view
- entre muitas outras

DIRECTIVES ³

Exemplo de *directives* sendo utilizadas

```
<!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

Você pode criar sua própria *directive*.

- `:<my-directive></my-directive>`
- `:`
- `:`
- `:<!-- 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 como um elemento -->`

`<div my-directive> </div>` `<!-- Directive como um atributo -->`

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>
```

```
<!-- Nossa tag customizada -->
```

```
<h1 class="title">Text from directive</h1> <!-- Se converte no template -->
```

TESTANDO SEU APP

Um framework criado com o intuito de ser facilmente testado

FERRAMENTAS RECOMENDADAS:

- Karma Test runner
- Jasmine Test framework

TESTANDO CONTROLLERS

```
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();  
  });  
});
```

REFERÊNCIAS

- Angular -
- Karma -
- Jasmine -
- HandsOn -

FIM

- Telefonica Digital