AngularJS

Superheroic JavaScript MVW Framework

Agenda

- **Parte 1 – Fernando Rodrigues Baroni**
- **Parte 2 – Introdução ao AngularJS**
- **Parte 3 – Diretivas**
- **Parte 4 – Controllers**
- **Parte 5 – Diretiva Customizada**
- **Parte 6 - Desafio**

Fernando Rodrigues Baroni

Fernando Rodrigues Baroni

- **Nasceu em Itapetininga**
- **Vive em Sorocaba**
- **Formado na UNIP**
- **Desenvolvedor por paixão**
- **Gosta de qualidade em codigos**
- **http://baroni.tech/**

IBM

GFT


GFT
GFT
GFT
Expansão das instalações da GFT Brasil.
Estamos contratando!
careerbrasil@gft.com

Stefanini

BMega

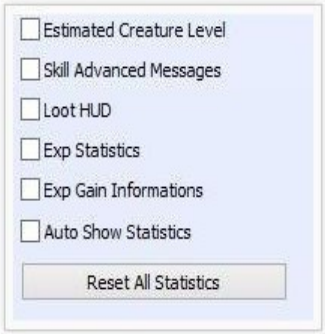
Home | Features | Account | Download | Scripts | Documentation | Forum

Features

Super Powers

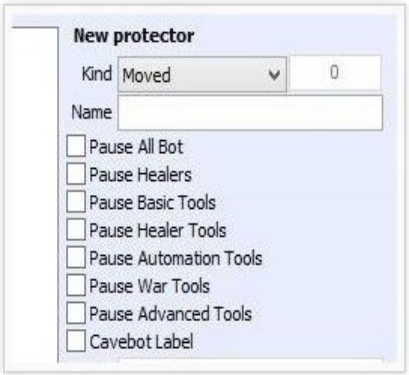
Basic Features

Many different tools to help with various problems



HUD and Informations

Some HUD to make the game a little more fun



Protector

You will need some help to sleep in peace

Introdução ao AngularJS

Introdução, material e ambiente

AngularJS

Superheroic JavaScript MVW Framework

O que é AngularJS?

- **É uma Framework para Javascript**
- **Implementa o padrão MVC**
- **Foco em Single Page Applications (SPA)**
- **É fácil e divertido**
- **Extende o HTML**

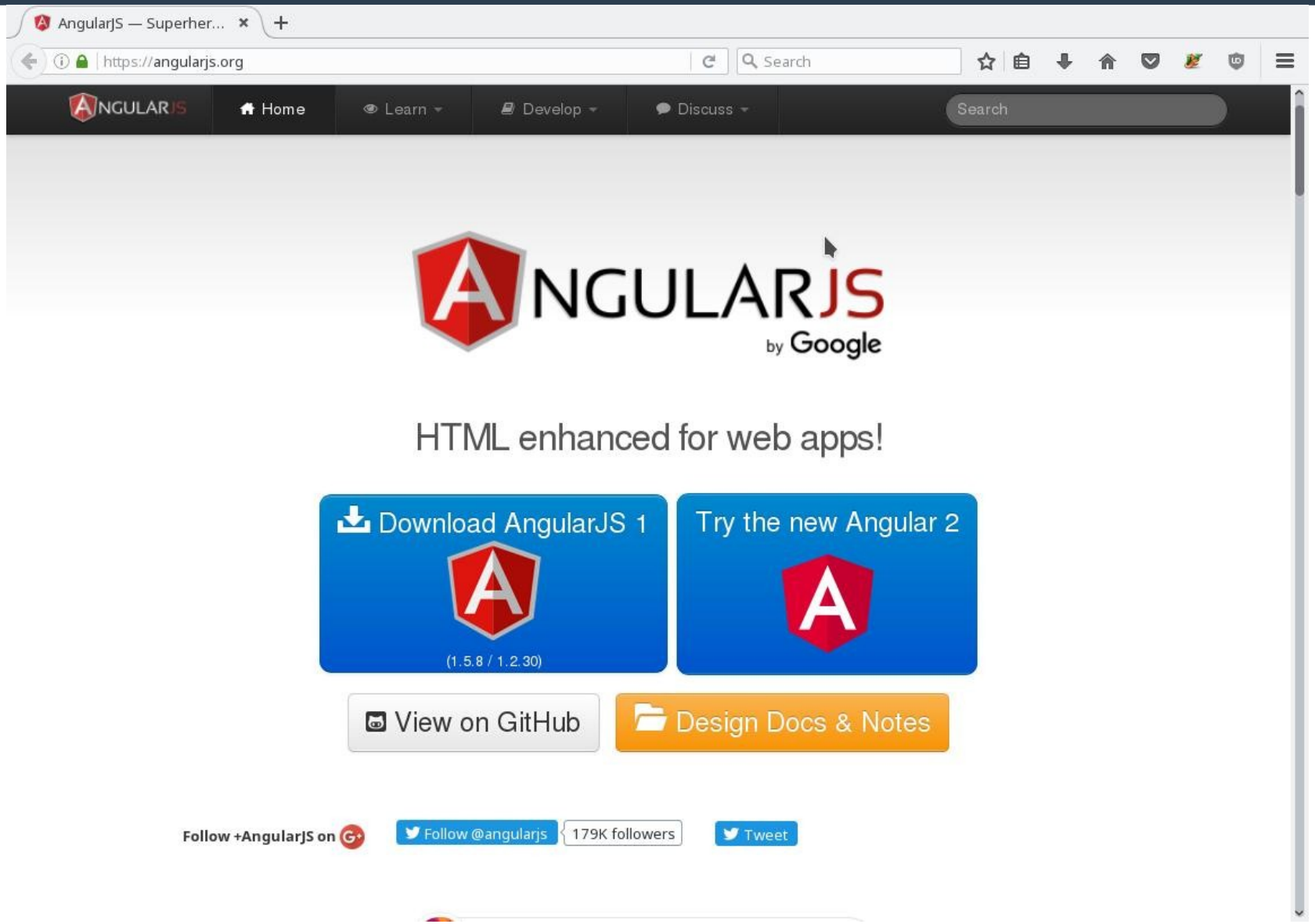
Para que aprender AngularJS?

- **Alta demanda no mercado de trabalho**
- **Simplifica a vida dos programadores para criar websites mais poderosos com menor esforço**
- **Tem uma comunidade imensa, é fácil pedir e prover ajuda**

Materiais necessários para o curso

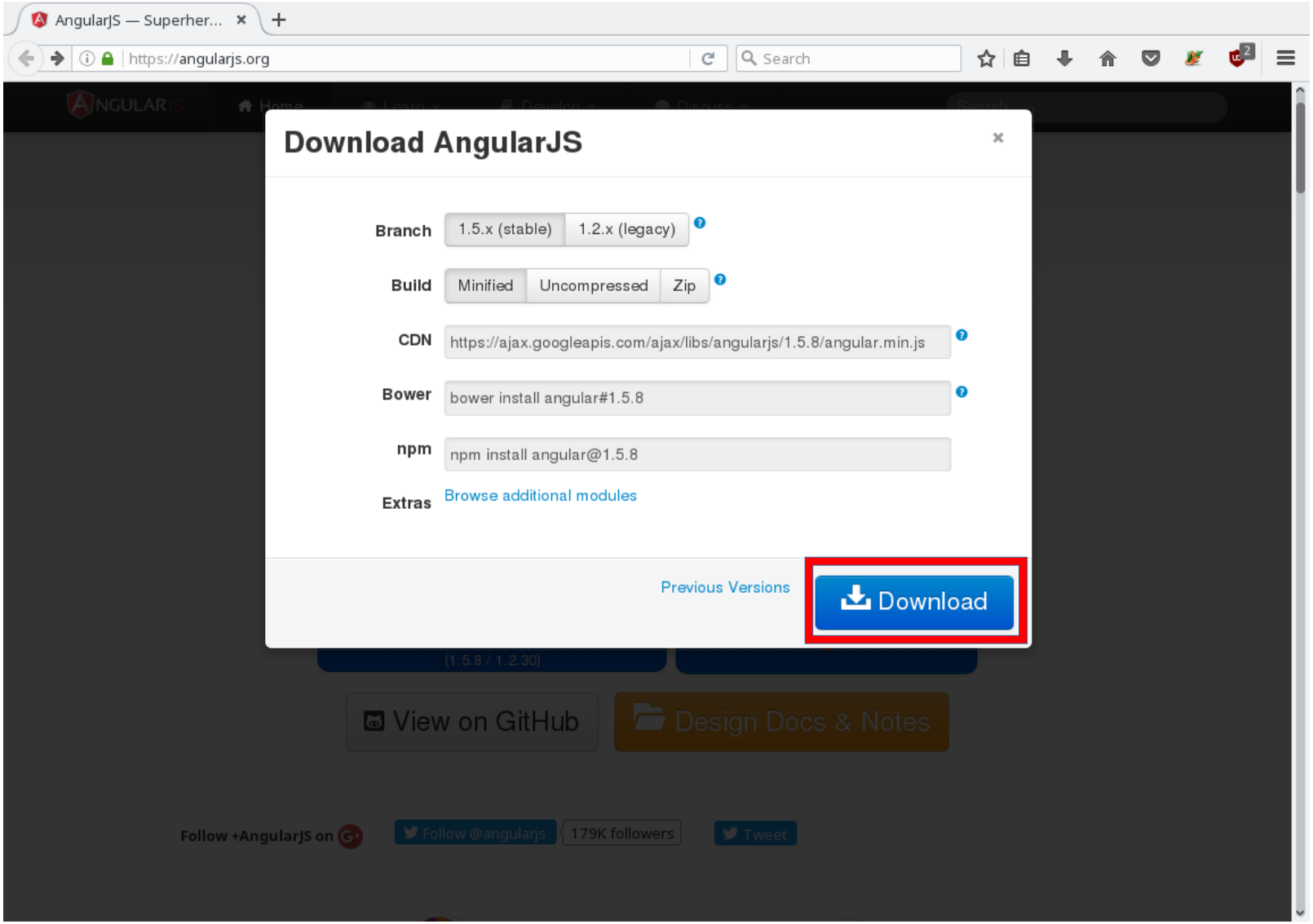
- **Servidor HTTP**
- **Editor de textos**
- **Navegador (preferencialmente Chrome)**
- **AngularJS**
- **Conhecimento básico de logica de programação e Javascript**

http://angularjs.org/



Download AngularJS 1

Fazer download

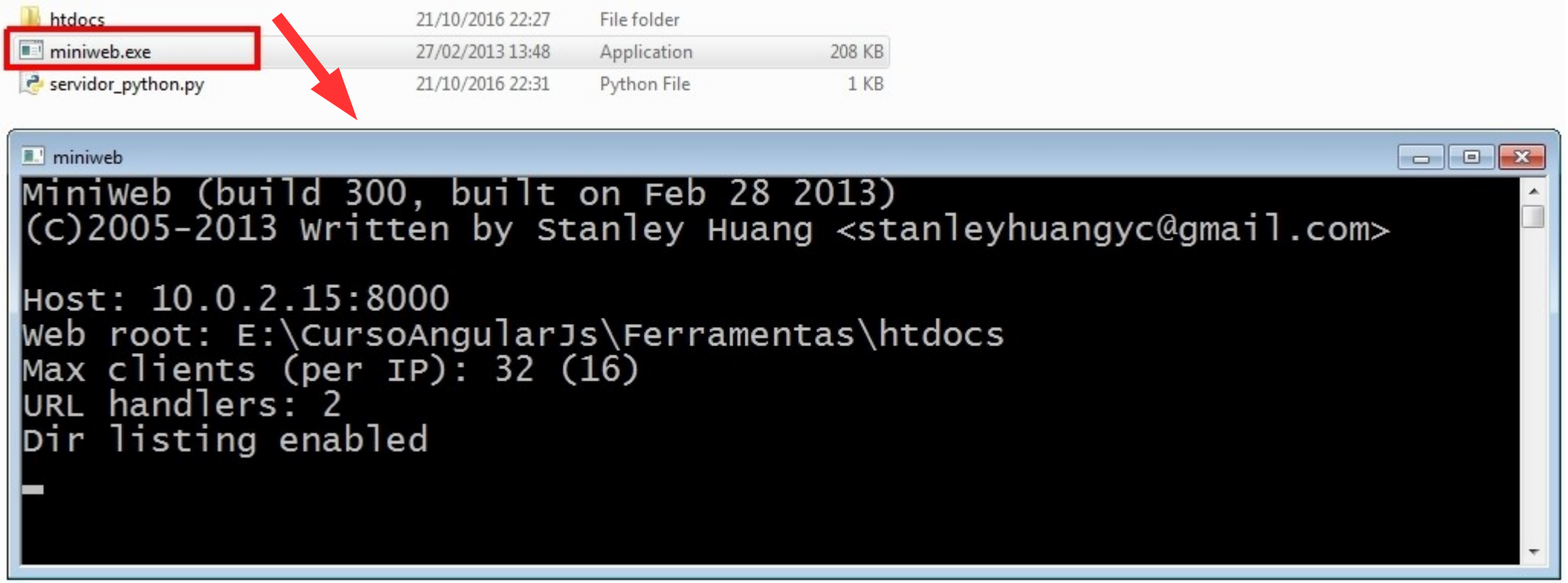


Ou simplesmente

http://baroni.tech/curso-angular

Iniciar o servidor HTTP

Para Windows:



Para Linux/MacOS:

```
┌─fernando at fernando in ~/CursoAngularJs/Ferramentas using
└─○ python2.7 servidor_python.py
serving at  0.0.0.0 : 8000
```

Criando a aplicação AngularJS

Temos nosso template HTML index.html:

```
1  <!DOCTYPE html>
2  <html lang="pt">
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6      </head>
7      <body>
8              Olá, mundo!
9      </body>
10 </html>
```

Como transformar um HTML básico em uma aplicação AngularJS?

Etapa 1

Adicionamos o script AngularJS

```
1  <!DOCTYPE html>
2  <html lang="pt">
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7      </head>
8      <body>
9              Olá, mundo!
10     </body>
11 </html>
```

Etapa 2

Adicionamos a diretiva ngApp

```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app>
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7      </head>
8      <body>
9              Olá, mundo!
10     </body>
11 </html>
```

Essa diretiva marca o inicio de uma aplicação AngularJS

Etapa 3

Testamos o AngularJS

```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app>
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7      </head>
8      <body>
9          Olá, mundo! 5 + 8 = {{5 + 8}}
10     </body>
11 </html>
```

Usamos uma simples expressão matemática para provar que o AngularJS esta funcionando

Não trabalhar com protocol file://

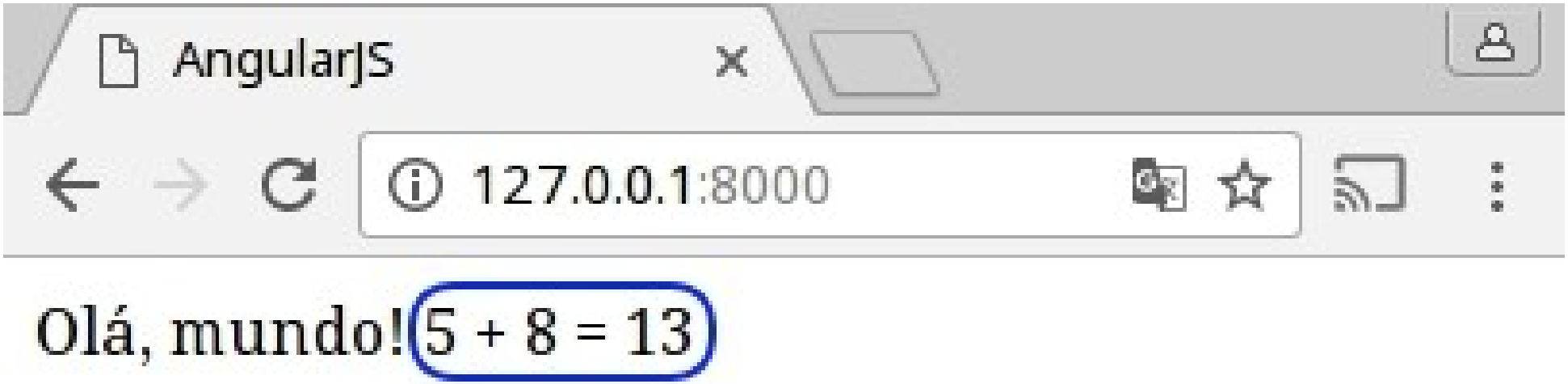
Protocolo file:// **ERRADO**

Protocolo http:// **CORRETO**

AngularJS no protocolo file:// faz aparecer bugs obscuros.

Etapa 4

Rodamos nossa aplicação



Sucesso!

Diretivas

Definição de Diretiva e as mais comuns

O que são Diretivas?

Diretivas são marcas que controlam como o AngularJS injeta comportamentos em um elemento. Comportamentos podem ser estilos, EventListeners e até mesmo transformações para outros elementos.

Tags	<my-dir></my-dir>
Atributos	
Classes CSS	
Comentários HTML	<!-- directive: my-dir exp -->

Normalização nos nomes das Diretivas

Os nomes das diretivas são escritas usando camelCase. Quando usadas em HTML, o nome deve ser convertido! Exemplo: ngApp → ng-app.

O $compilador do AngularJS também suporta outros formatos:

- **Usando traços:** `<span ng-bind="name"></span>` **(mais comum)**
- **Usando dois-pontos:** `<span ng:bind="name"></span>`
- **Usando underscore:** `<span ng_bind="name"></span>`
- **Prefixado com “data-”:** `<span data-ng-bind="name"></span>`
- **Prefixado com “x-”:** `<span x-ng-bind="name"></span>`

Diretivas

Vamos conhecer algumas das diretivas mais comuns, e logo em seguida aplicá-las em alguns simples exercícios.

Expressões e ngInit

ngInit inicializa variáveis;

Expressões permitem executar comandos no HTML

```
1       <div ng-init="
2           mensagem='Olá Mundo!';
3           resposta_do_universo=42;
4           pessoa={nome: 'Fernando', sobreNome: 'Rodrigues Baroni', idade: 10}
5           ">
6           Expressões matemáticas: {{10*(1+2)}}  <br/>
7           Two-way binding: {{mensagem}}         <br/>
8           Single-time binding: {{::resposta_do_universo}}   <br/>
9           Objetos: {{pessoa.sobreNome}}
10      </div>
```

É possível fazer uma expressão ser executada só uma vez, adicionando {{::nome}} duas vezes o simbolo de dois pontos no inicio da expressão.

Filters

Filtros permitem transformar expressões

```
1       {{"heLl0" | uppercase}} <BR/>
2       {{"wOrLD" | lowercase}} <BR/>
3       {{123456789 | number}} <BR/>
4       {{[1,8,2,4,7,5] | limitTo: 3 }} <BR/>
5       {{['John', 'Maria', 'Jose'] | filter:'J' }}
```

Podem formatar números, datas, filtras coleções e até mesmo I18N

ngModel

Vincula um componente de entrada, para uma variável no escopo

```
1       <input type="text" placeholder="Nome" ng-model="nome"/>
2
3       <input type="datetime-local" placeholder="Data de nascimento"
4           ng-model="nascimento"/>
5
6       <label><input type="checkbox" ng-model="temFilhos"/>
7           Tem filhos?</label>
```

ngBind

Vincula um elemento qualquer a uma variável do escopo

```
1   Seu nome é <span ng-bind="nome"></span>.
2
3   Seu nome é {{nome}}.
```

É parecido a uma {{expressão}} e *quase* sempre dá o mesmo resultado.

ngHide, ngShow e ngIf

Mostra ou esconde elementos baseados em uma expressão booleana

```
1       <p ng-hide="expr"></p>
2
3       <span ng-show="expr"></span>
4
5       <div ng-if="expr"></div>
```

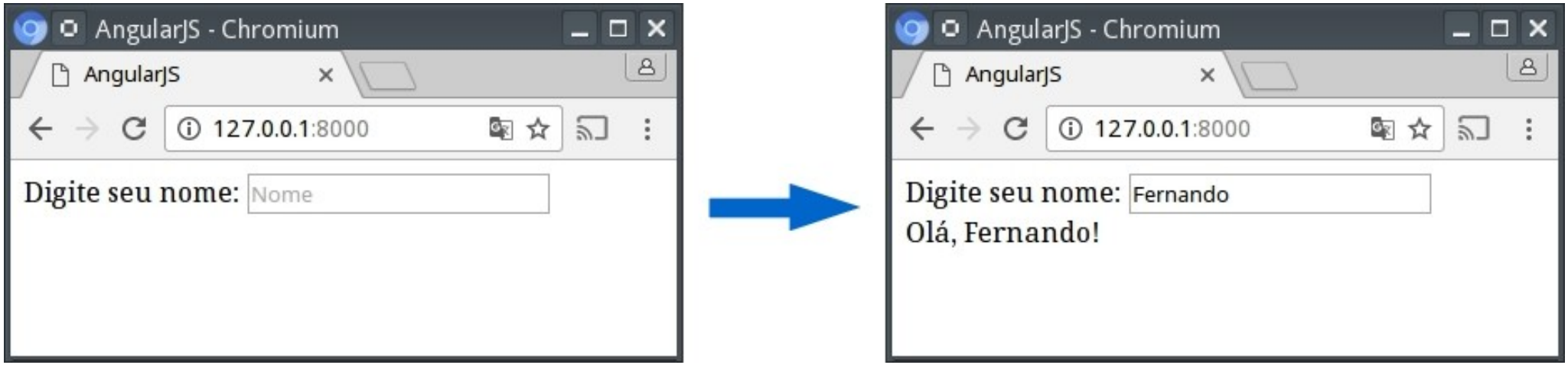
Atenção: ngIf não é igual ao ngShow e ngHide!

ngIf adiciona ou remove o elemento do DOM.

ngHide e ngShow mostra ou esconde o elemento usando CSS.

Desafio: Hello, Fernando!

Usando as diretivas aprendidas, fazer uma aplicação Hello World usando models e condicionais para mostrar a mensagem “Olá Mundo” apenas quando o nome for atribuído



Desafio: Hello, Fernando!

```
<!DOCTYPE html>
<html lang="pt" ng-app>
    <head>
        <meta charset="utf-8"/>
        <title>AngularJS</title>
        <script src="angular.min.js"></script>
    </head>
    <body>
        Digite seu nome:
            <input type="text" placeholder="Nome" ng-model="nome"/>

        <div ng-show="nome">
            Olá, {{nome}}!
        </div>
    </body>
</html>
```

ngClass

Gerencia classes CSS de um elemento, ativando ou desativando a classe condicionalmente.

```
<p ng-class="
    {
     'css_sucesso': com_sucesso
     'css_falha':   !com_sucesso
    }
     ">
        ng-Class
</p>
```

ngRepeat com lista

Replica um template para cada elemento de uma coleção

```
1       <div ng-init="
2           listaPessoas = ['Fernando', 'João', 'Maria', 'Ricardo', 'Marcelo']
3           ">
4           <p ng-repeat="nome in listaPessoas">
5               A pessoa {{$index + 1}} se chama {{nome}}
6           </p>
7       </div>
```

ngRepeat com dicionário ou objeto

Replica um template para cada elemento de uma coleção

```
1     <div ng-init="
2         objetoFernando = {
3             nome: 'Fernando',
4             sobreNome: 'Rodrigues Baroni',
5             idade: 23}
6         ">
7         <p ng-repeat="(chave, valor) in objetoFernando">
8             A chave '{{chave}}' tem valor '{{valor}}'
9         </p>
10    </div>
```

Controllers

Como criar um controller, $scope e Dependency Injection.

Adicionar um novo arquivo para trabalho

Primeiro, vamos adicionar um novo script em nossa aplicação, para organizar melhor nosso código.

A nossa Controller, vai ficar nesse novo arquivo, desta forma, mantemos a View e a lógica da nossa aplicação separadas.

Etapa 1

Adicionamos um script em nossa aplicação

```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app>
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7          <script src="app.js"></script>
8      </head>
9      <body>
10             Olá, mundo!
11     </body>
12 </html>
```

Etapa 2

Atribuímos um nome para a aplicação AngularJS

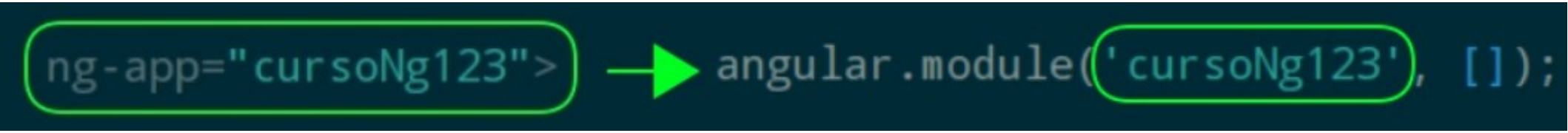
```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app="cursoNg123">
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7          <script src="app.js"></script>
8      </head>
9      <body>
10             Olá, mundo!
11     </body>
12 </html>
```

Etapa 3

Iniciamos o nosso modulo AngularJS

```
1 (function() {
2
3     angular.module('cursoNg123', []);
4
5 })();
```

Atenção: o nome do modulo no arquivo app.js deve bater com o nome da diretiva ng-app na view (html). O nome não pode começar com “ng”



Criando Controller

Agora que temos um arquivo Javascript para trabalhar, também temos o nosso modulo AngularJS, podemos finalmente criar o nosso Controller.

```
1   angular.module(...).controller(
2           'NomeDoController',
3           ['dependencia', 'dependencia2',
4               function(dependencia, dependencia2) {
5                   // Codigo do Controller
6               }
7           ]);
```

Injeção de Dependências

O AngularJS tem um sistema de Injeção de Dependências:

```
1    angular.module(...).controller(
2            'NomeDoController',
3            ['dependencia', 'dependencia2',
4                function(dependencia, dependencia2) {
5                    // Codigo do Controller
6                }
7            ]);
```

Talvez você se pergunte, por que nos repetimos as dependências, uma vez na lista de dependências e outra nos argumentos da função do Controller?

Falando em dependências: $scope

A dependência mais comum é o $scope.

O $scope, é um objeto de contexto, que permite que a view (o HTML) leia dados da Controller. Quando queremos expor alguma variável do Controller, ou até mesmo uma função, nos exportamos esses para o $scope, assim a view pode usar esses objetos exportados.

Olá Mundo, Controller!

Vamos praticar o uso de Controllers com o clássico: 'Hello World':

```
1 (function() {
2
3     var app = angular.module('cursoNg123', []);
4
5     app.controller('MeuControlador', ['$scope', function($scope) {
6         $scope.mensagem = "Olá, Controller!";
7     }]);
8 })();
```

Injetando a Controller na view

```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app="cursoNg123">
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7          <script src="app.js"></script>
8      </head>
9      <body>
10         <div ng-controller="MeuControlador">
11             {{mensagem}}
12         </div>
13     </body>
14 </html>
```

É necessário injetar a Controller para usarmos a mesma na View, fazemos isso com a seguinte diretiva: ngController

$scope na view

Quando injetamos o Controller, temos acesso ao $scope que o Controller nos expôs!

```
1  <!DOCTYPE html>
2  <html lang="pt" ng-app="cursoNg123">
3      <head>
4          <meta charset="utf-8"/>
5          <title>AngularJS</title>
6          <script src="angular.min.js"></script>
7          <script src="app.js"></script>
8      </head>
9      <body>
10         <div ng-controller="MeuControlador">
11             {{mensagem}}
12         </div>
13     </body>
14 </html>
```

Exportando funções

Podemos também exportar funções para o $scope

```
8         $scope.somar = function(a, b) {
9             return a + b;
10        };
```

E chamar as funções exportadas, usando expressões Angular

```
10        <div ng-controller="MeuControlador">
11            {{mensagem}} <br/>
12            A soma de 123 + 456 = {{ somar(123, 456) }}
13        </div>
```

Diretiva Customizada

Como criar uma diretiva

Como criar uma diretiva basica

Criar diretivas é muito semelhante a criar Controllers:

```
1   angular.module(...).directive('minhaDiretiva', function() {
2       return {
3           restrict: 'A',
4           template: "" +
5               "<p>" +
6               "    <span>{{pessoa.nome + ' ' + pessoa.sobrenome}}</span>" +
7               "</p>"
8       };
9   });
```

Diretivas são criadas, fazendo uma função que retorna um objeto.

Atributo: Template e TemplateURL

- **template → Usado para definir como a diretiva vai ser renderizada no HTML. É uma String, que contem o template final da Diretiva, podendo usar expressões e outras diretivas AngularJS para formar um template.**
- **templateUrl → Usado para definir o template, mas em vez de passar uma String com o template a ser renderizado, podemos passar uma URL para um arquivo HTML de modelo. É bom usar, quando a diretiva fica grande, pois facilita separar a view (o HTML) da logica da diretiva (a própria função da Diretiva).**

Atributo: Restrict

Usado para definir como nossa diretiva vai ser usada no HTML, sendo uma <TAG> ou um Atributo. Pode ser uma combinação das seguintes letras:

E	Tags	<my-dir></my-dir>
A	Atributos	
C	Classes CSS	
M	Comentários	<!-- directive: my-dir exp -->

Exemplos: E → como tag EA → como tag ou atributos

Desafio

Agenda de Telefone

Faça uma pequena aplicação para exibir uma agenda telefônica.

Requisito 1: Controller

Crie um Controller que inicializa uma lista de objetos, cada objeto vai ter os seguintes atributos: nome, sobreNome, telefone e cidade.

Exemplo:

```
{
    nome: 'Fernando',
    sobrenome: 'Rodrigues Baroni',
    telefone: 55159962180 11,
    cidade: 'Sorocaba'
}
```

Requisito 2: Diretiva

Na view, use alguma diretiva para desenhar todos os contatos registrados no escopo, utilizando a diretiva criada

```
1   app.directive('meuContato', function() {
2       return {
3           restrict: 'A',
4           template: "" +
5               "<p>" +
6               "   Mostrar dados do contato" +
7               "</p>"
8       };
9   });
```

Obrigado!