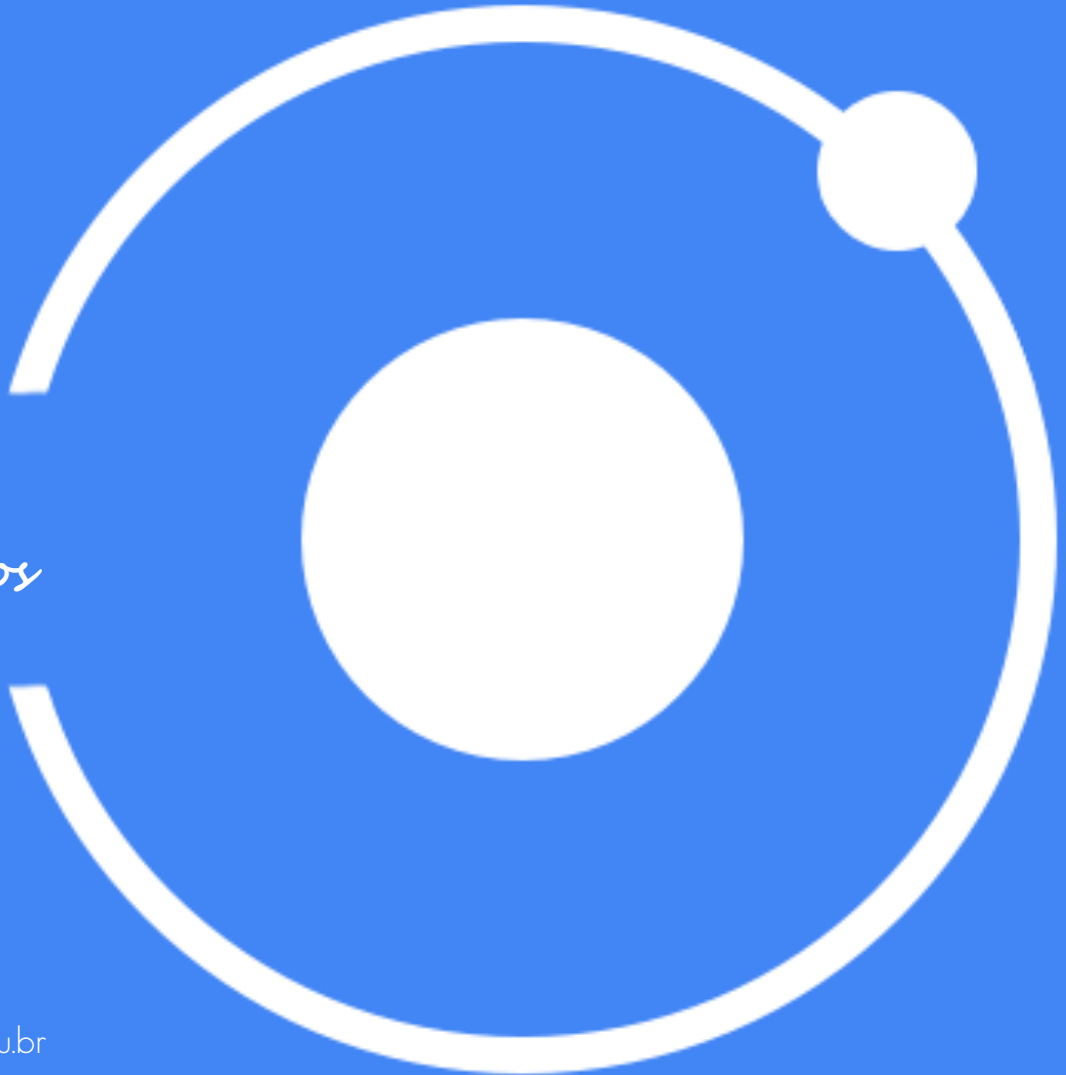
Ionic Framework

Arquitetura base de diretórios



Prof. Fernando Luis - fernando.oliveira@iffarroupilha.edu.br

Introdução

- Para criar um novo aplicativo utilizando ionic podemos utilizar o comando:

```
$ ionic start nomeProjeto template
```

- O ionic *CLI* se encarrega de gerar toda a estrutura do aplicativo, que vai desde a arquitetura de pastas a criação de arquivos de configuração.

primeioApp

VsCode

EXPLORER

OPEN EDITORS

PRIMEIOAPP

- e2e
- node_modules
- src
- .gitignore
- angular.json
- browserslist
- ionic.config.json
- karma.conf.js
- package-lock.json
- package.json
- tsconfig.app.json
- tsconfig.json
- tsconfig.spec.json
- tslint.json

OUTLINE

NPM SCRIPTS

Estrutura de pastas

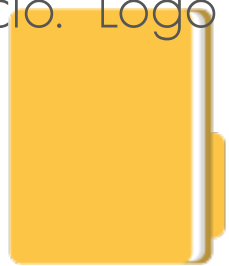
master 0 0

Contextualizando

e2e: Pasta responsável por testes unitários, o co ando para executar os testes é: *ng e2e*. Na prática utilizamos o angular cli para executar os testes, os quais são executados considerando o framework *Selenium*;

Node _ modules: pasta onde ficam armazenados os plugins (depêndecias) externos a aplicação, instalados via npm. Salienta-se que as dependências ficam registradas no arquivo package.json

src: diretório principal da aplicação, neste local ficam todo o código fonte da aplicação, ou seja, as telas (paginas) e lógica de negócio. Logo vamos interagir com essa pasta e suas subpastas



Contextualizando

- *.gitignore*: Arquivo do git que utilizamos para o gerenciamento dos arquivos que serão ignorados no momento do nosso commit.
- *karma.config.js*: O Karma é uma biblioteca utilizada para criação de testes unitários desenvolvida pela própria equipe do Angular.
- *package.json*: Esse arquivo é o responsável por gerenciar as dependências do nosso projeto, quando nós executamos o comando npm install, ele verifica os pacotes que estão dentro desse arquivo e baixa para o nosso diretório node modules.
- *tsconfig.json*: Arquivo de configuração do TypeScript.

.

Contextualizando

- *tslint.json*: O tslint fica verificando se estamos escrevendo o nosso código corretamente, ele verifica a sintaxe do nosso projeto em tempo de execução e em caso de algum erro ou warning ele lança uma exception no console.
- *angular.json*: Fornece as configuração padroões para todo worksspace desde configurações gerais até as ferramentas utilizadas no desenvolvimento fornecidas pela CLI Angular tais como TSLint, Karma, and Protractor entre outros

primeioApp

EXPLORER

OPEN EDITORS

PRIMEIOAPP

- e2e
- node_modules
- src
  - app
  - assets
  - environments
  - theme
  - global.scss
  - index.html
  - main.ts
  - polyfills.ts
  - test.ts
  - zone-flags.ts
- .gitignore
- angular.json
- browserslist

OUTLINE

NPM SCRIPTS

Subdiretórios SRC

master 0 0

Contextualizando - SRC/APP

- *index.html*: Este é o arquivo root (principal), dentro dele é montado a SPA (Single Page Application).

- *main.ts*: Esse é o arquivo principal da aplicação. Ele é definido dentro do nosso arquivo angular.json, ele representa o bootstrap da aplicação.

- *polyfills.ts*: Esse arquivo funciona como um tradutor. Ex.: Nós precisamos utilizar algo novo do ES6, mas os nossos navegadores só conseguem entender o ES5, ele irá interpretar e passar o código correto para os nossos navegadores.

Contextualizando - SRC/APP

- *global.scss*: Como todos os componentes tem o seu próprio arquivo de estilo (.scss), este arquivo é utilizado para criar definicões globais de estilo, bem como variáveis para nossa aplicação.
- *Environments*: Aqui há dois arquivos .ts, um para o ambiente de produção e um outro para o ambiente de desenvolvimento. Neles nos permitem utiliZar configurações específicas de acordo com a fase de desenvolvimento
- *Assets*: Esse diretório éutilizado para definir arquivos extras a aplicação, recursos estáticos como imagens. Esse diretório é configurado de[illegible] nosso arquivo angular.json.

primeioApp

EXPLORER

OPEN EDITORS

PRIMEIOAPP

- e2e
- node_modules
- src
  - app
    - home
    - app-routing.module.ts
    - app.component.html
    - app.component.scss
    - app.component.spec.ts
    - app.component.ts
    - app.module.ts
  - assets
  - environments
  - theme
  - global.scss
  - index.html

OUTLINE

NPM SCRIPTS

Subdiretórios SRC/APP

master 0 0

Contextualizando - SRC/APP

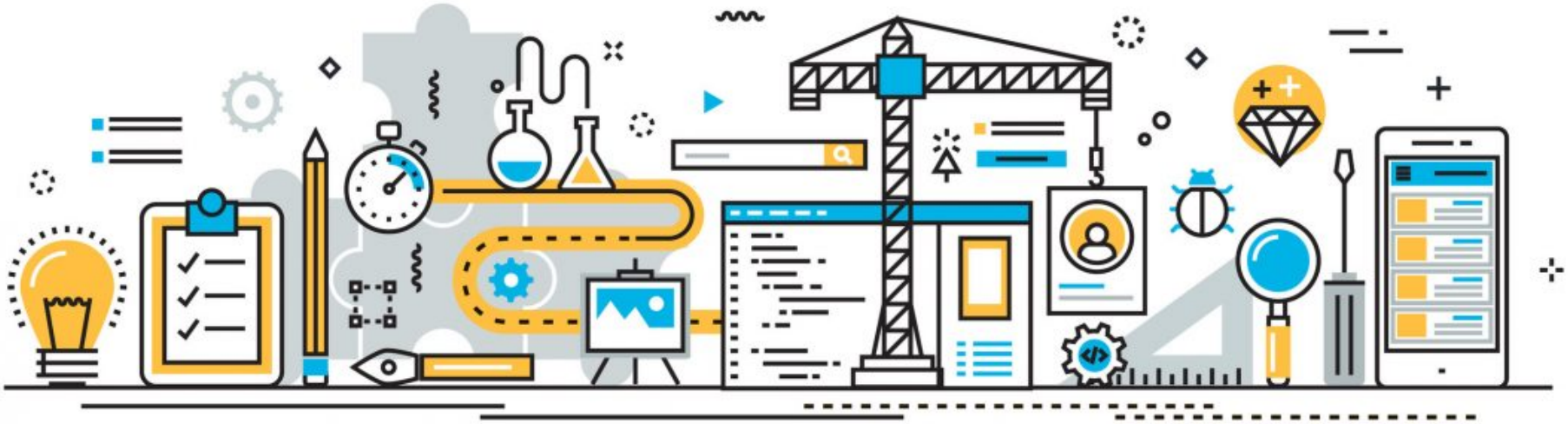
- *app.routing.module.ts*: Arquivo que define as rotas do módulo (endereços que podem ser acessados)
- *app.component.scss*: Arquivo responsável pelo estilo do módulo. Com o Angular nós trabalhamos com os estilos separados para cada componente, assim conseguimos ter um desacoplamento de estilos. Permite o uso de arquivos .css e .sass.
- *app.component.html*: *Arquivo HTML do componente App, neste caso permite o uso de tag html e componentes Ionic.*
- *app.component.spec.ts*: *Arquivo de teste do componente.*

Contextualizando - SRC/APP

- *app.component.ts*: Arquivo equivalente ao controle lógico da aplicação, local onde devem ficar as regras de negócio.

- *app.module.ts*: O Angular é um framework modular, ele precisa de um ou mais módulos para que possamos gerenciar os nossos componentes, esse módulo acaba sendo um default, entretanto, é possível criar módulos separados e só importá-los

Páginas com Ionic



Criando uma página

Para criar uma página basta utilizar o comando abaixo:

```
$ ionic generate page nomePagina
```

Ou

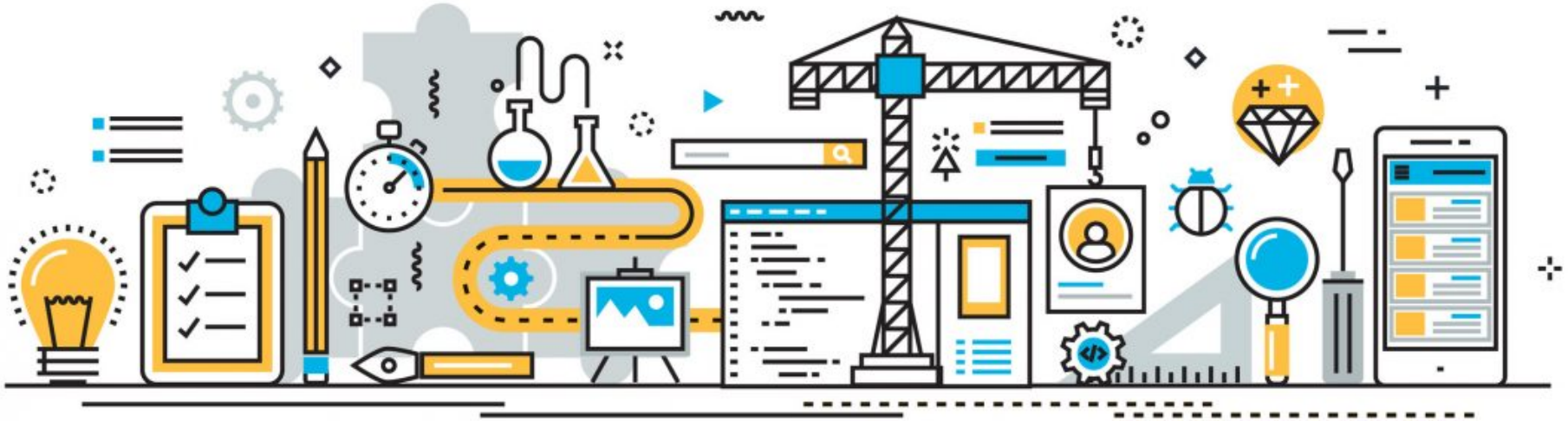
```
$ ionic g page nomePagina
```

Obs: utilize uma estrutura com pasta para organizar melhor sua aplicação, como por exemplo: ionic g page pages/nomePagina

Páginas

- Para cada página criada teremos: um pasta com nome da página, a qual comportará arquivo HTML, um arquivo SCSS e uma arquivo de lógica (typescript), bem como os demais módulos do Angular e testes.
- Uma entrada no arquivo de rotas da aplicação (app-routing.module.ts)

.



HTML

- O arquivo HTML define o layout da página. Ela pode ser constituída de tags do HTML 5 ou componentes de interface do Ionic

```
1   <ion-header>
2     <ion-navbar>
3       <ion-title>
4         App Ionic
5       </ion-title>
6     </ion-navbar>
7   </ion-header>
8
9   <ion-content padding>
10    <h3>Aplicativo híbrido</h3>
11    <p>
12      Ionic é uma framework para a construção
13    </p>
14  </ion-content>
```

SCSS

- SASS / SCSS é uma plataforma pra desenvolver CSS de forma mais produtiva, trata-se de uma linguagem de programação baseada em CSS.

```scss
page-home {
  $cor-base: #ff9900;
  h3{
    color:$cor-base;
  }
  p {
    border: dotted 1px $cor-base;
  }
}
```

App Ionic

Aplicativo híbrido

Ionic é uma framework para a construção de aplicativos híbridos utilizando HTML, CSS e JavaScript. Ele vem com um conjunto de componentes de UI e funções que você pode utilizar para criar aplicações mobile totalmente funcionais e atrativas.

Navegação

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Ionic Blank
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>

<ion-button href="/teste"  > vai</ion-button>
ou
<ion-button (click)="navegar()" > vai</ion-button>
</ion-content>
```

TypeScript

- Possibilita que você escreva código JavaScript semelhante ao Orientação a Objetos. Ele habilita a criação de classes, construtores, tipar variávies, utilizar herança entre outros.
- Com typescript definimos nosso módulo do Angular, e através deste implementamos nossas regras de negócio..

TypeScript

```typescript
import { Component } from '@angular/core';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  constructor() {}

}
```

TypeScript

```typescript
import { Component } from '@angular/core';
import { Router } from '@angular/router';

@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

  constructor(private router: Router) {}

  navegar(){
    this.router.navigate(['/teste']);
    // ou
    this.router.navigateByUrl('/teste');
  }
}
```