BEM VINDOS!

BOOT CAMP

ANGULAR
by Google

OBJETIVOS

1. *Revisitar a história e o contexto do Angular;*
2. *Aprender o essencial sobre o framework;*
3. *Construir o front-end de uma aplicação de chat!*

MOTIVAÇÃO

- *Foco em desenvolver a aplicação, não a infraestrutura;*
- *Servidor torna-se um conjunto de APIs (microservices?);*
- *Usar o poder de processamento do cliente;*
  - *Redução do volume de requests e tráfego de rede;*
  - *Requer menos poder de processamento no servidor!*
- *Código mais enxuto e limpo;*
  - *Manipulação de DOM feita onde deve ser feita;*
  - *Tags HTML customizadas a partir da criação de componentes.*
- *Teste (unitário e end-to-end) bem mais simples e direto.*

AGENDA

- *O que é Angular?*
- *Paradigmas*
- *Angular 5.0*
- *Vantagens*
- *Perguntas*
- *Mão na massa!*

O QUE É ANGULAR?

Uma introdução

SOBRE O FRAMEWORK

- *Plataforma open source para desenvolvimento front-end;*
- *Focado em SPAs (Single Page Applications) por natureza;*
- *Permite desenvolvimento de aplicações complexas (enterprise) em JS;*
- *Criado e mantido pelo Google;*
  - *Comunidade ativa ao redor do mundo e manutenção constante;.*
- *Multiplataforma (e mobile-first);*
- *Recursos interessantes para um framework front-end.*
  - *Templates, injeção de dependências, expressions...*

QUEM USA ANGULAR HOJE?

Google

The Walt Disney Studios

ionic

Microsoft

@WalmartLabs

aws

intel

Fonte: *https://www.reddit.com/r/Angular2/comments/6jljwf/who_uses_angular_24/*

ANGULAR 1, 2 E... 4?

- *AngularJS: revolução em termos de frameworks FE;*
- *Angular 2: reimplementação "mantendo o melhor" e removendo soluções que não funcionavam tão bem;*
- *Novas major versions (manutenção) a cada 6 meses.*
  - *Angular 4 - Mar/2017;*
  - *Angular 5 - Nov/2017;*
  - *Angular 6 - Mar/2018...*
- *Angular 3* ***não existiu*** *devido a desalinhamento inicial de versões.*

@angular/core	v2.3.0
@angular/compiler	v2.3.0
@angular/compiler-cli	v2.3.0
@angular/http	v2.3.0
@angular/router	**v3.3.0**

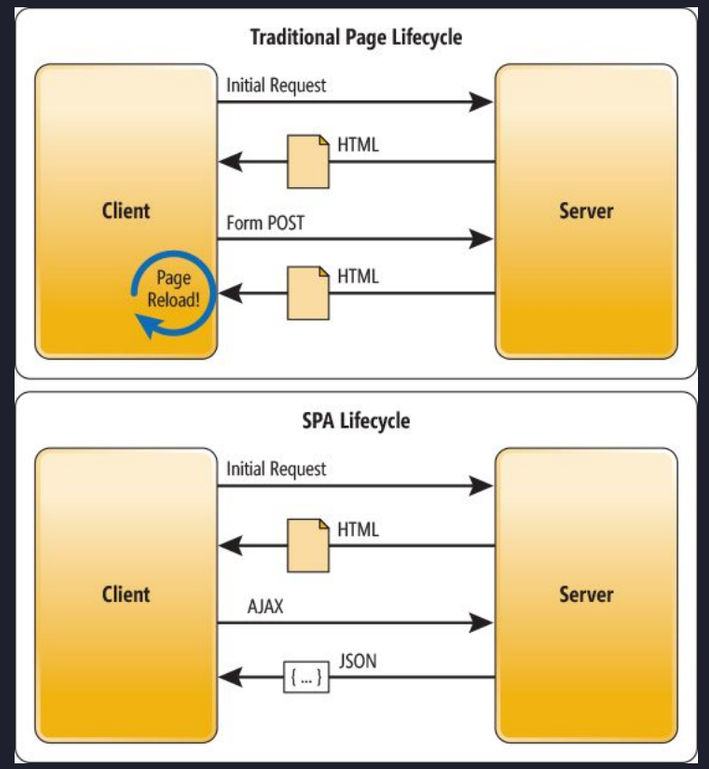
MUDANÇA DE PARADIGMA

- *AngularJS: MVW*
  - *Models, views e controllers para as páginas;*
  - *Navegação baseada em rotas direcionadas a páginas e controllers;*
  - *myApp.controller(...);*
  - *Configuração do projeto e criação de módulos feitos manualmente.*
- *Angular2+: MV**
  - *Orientado a componentes;*
  - *Roteamento robusto e nativo baseado em componentes;*
  - *Utiliza o poder do TypeScript*
    - *Annotation-driven (@Component);*
    - *Orientação a objeto;*
  - *Auxílio do Angular CLI.*

O QUE SABER PARA TRABALHAR COM ANGULAR 2+?

SINGLE PAGE APPLICATIONS (SPA)

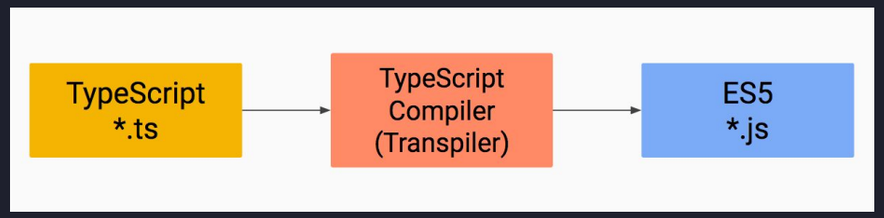
- *Melhor experiência do usuário*
- *Performance*
- *Responsabilidade do client-side*
- *Facilidade de manutenção*
- *Uso de framework*
- *JS pode ser desabilitado no browser*
- *SEO*



ECMASCRIPT 2015 (ES6)

Funcionalidades implementadas em JS Engines (browsers).

- *Arrow functions;*
- *Classes;*
- *Template strings;*
- ***let + const**;*
- *Modules / Modules loaders;*
- *Promises;*
- *Outras funcionalidades.*

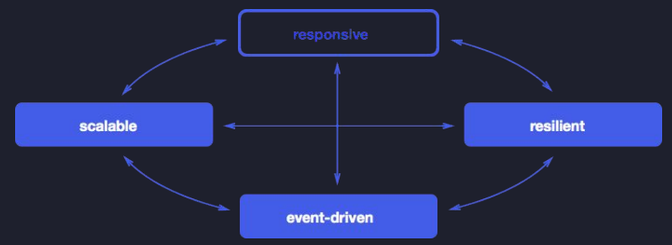


TYPESCRIPT

- *Superset de JavaScript que acrescenta recursos úteis;*
  - *Tipagem estática, classes, interfaces, strict null checks...*
- *Transpilada para JavaScript;*
  - *Suporta diversas versões de EcmaScript;*
  - *Source map disponível para o output;*
- *Robustez em aplicações de grande porte;*
  - *Oferece "erros de compilação" para evitar detecção tardia;*
  - *Suportada na maioria das IDEs;*
  - *Sintaxe similar a Java e C#.*
- *Open source, criada e mantida pela Microsoft.*

PROGRAMAÇÃO REATIVA

- *Espere algo acontecer e reaja a isso.*
  - *API para programação assíncrona baseada em Observables;*
  - *No more Promises!*
- *Inversão do paradigma de "polling", minimizando requests desnecessárias e poupando tráfego de rede;*
  - *Sistemas mais flexíveis, com menor consumo de recursos.*
- *Permite abstrair itens como low-level threading, sincronização e concorrência.*



AHEAD-OF-TIME COMPILATION (AOT)

- *Evolução do Just in Time.*
- *Rápida renderização;*
- *Menos requests;*
  - *Elimina a necessidade de requisições Ajax;*
- *Minimal Angular Framework*
  - *Não requer compilação;*
- *Detecção antecipada de erros em templates;*
- *Maior segurança*
  - *Compila templates e componentes em JavaScript;*
- *Prepara o terreno para Tree Shaking.*

ANGULAR 5.0

ANGULAR CLI

- *CLI nativa;*
- *Poupa tempo dos devs!*
  - *Já importa os serviços e os defines no app.module.ts;*
  - *Scaffolding de arquivos TS, HTML, CSS e testes unitários;*
  - *Integrada com o Git;*
  - *Comandos para rodar testes de forma simples e rápida.*

- *Automatiza tarefas triviais.*
  - *ng new*
  - *ng generate*
  - *ng serve*
  - *ng test / ng e2e*
- *Gera estrutura maior que a realmente necessária em muitos projetos.*

EXERCÍCIO 1

Iniciando a aplicação

1. *Abra o terminal e digite*

```
ng --version
ng new vntchat
cd vntchat
ng serve
```

2. *No browser, acesse:*

```
http://localhost:4200/
```

7

Structural Directives

Modules

Data Binding

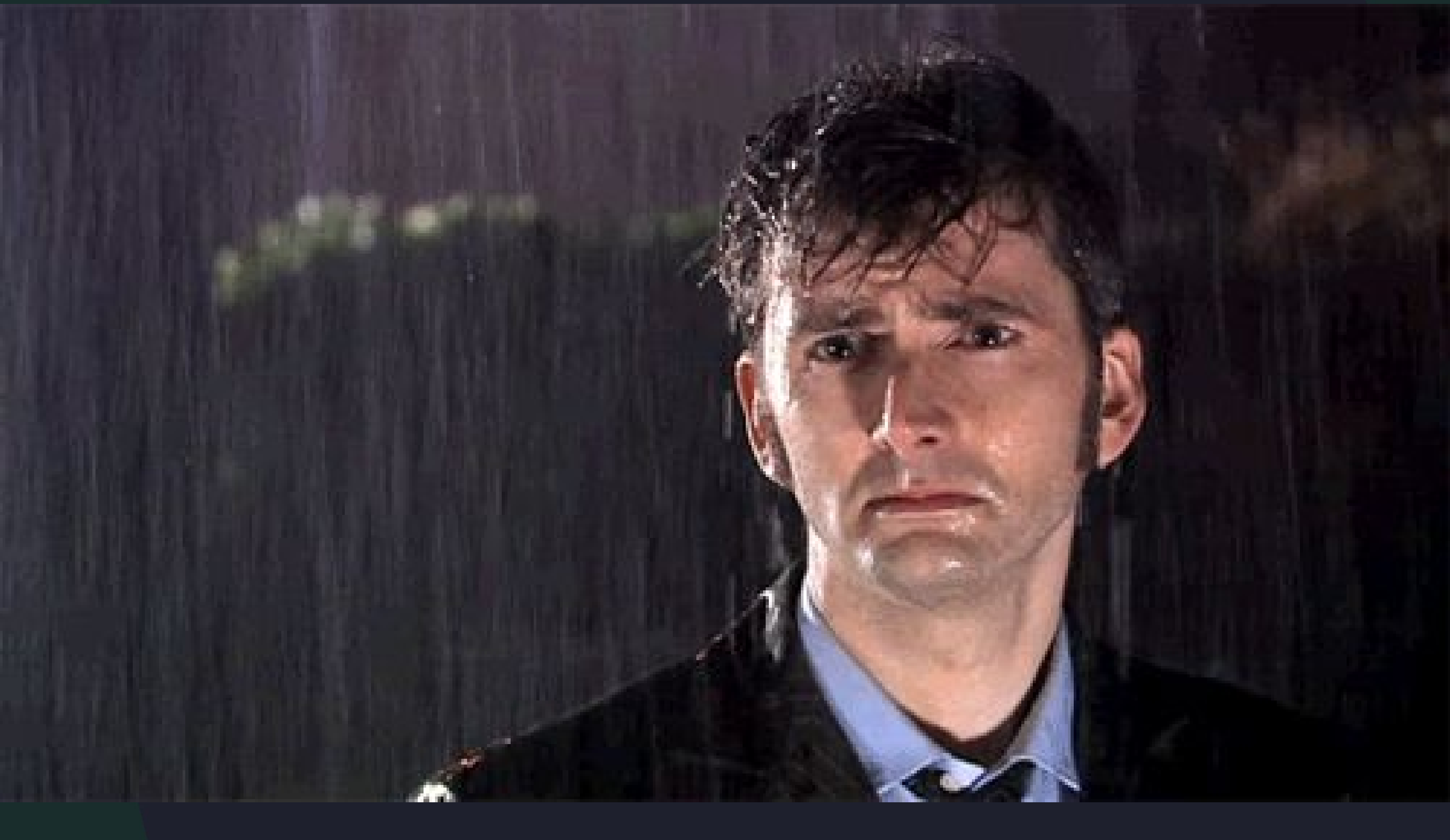
Components

Services

Templates

Dependency Injection

Source: [7 Keys to Angular 2 by John Papa]



DATA BINDING

- *Uma das "wow" features introduzidas no AngularJS;*
- *Comunicação transparente entre business e view.*
- *Mudanças são "automagicamente" transferidas!*
  - *Business --> View: One-Way Databinding*
  - *Business <-> View:* ***Two-Way Databinding***
- *Surpreendentemente fácil em Angular!*

```
[Property] + (Event) binding
```

```
<input [(ngModel)]="username">
<p>Hello {{username}}!</p>
```

COMPONENTES

- *Bloco mais básico da UI Angular;*
- *Principal forma de criar elementos e atributos na página;*
  - *Pode ser usado como uma tag HTML.*
  - *Ciclo de vida: cada vez que é usado, o Angular cria um novo componente, que é destruído assim que deixa de ser visualizado.*
- *Inicialmente criado por aplicação com Angular CLI.*
- *Associado a templates e, opcionalmente, estilos CSS.*

EXEMPLO

```
@Component({
  selector: 'vnt-component',
  templateUrl: './vnt.component.html',
  styleUrl: ['./vnt.component.scss']
})
export class VntComponent implements OnInit {

    constructor(private service: MessageService) { }

    ngOnInit() {
        this.messages = this.service.getMessages();
    }

}
```

Tag HTML

Template

Estilos CSS

Classe TypeScript

TEMPLATES

- *Visualização (View) do componente;*
- *Escrito em HTML;*
- *Pode acessar atributos e funções do componente, usando notação especial (**Angular expressions**);*
- *Pode usar diretivas estruturais do Angular como **if**, **for** etc.*
- *Pode referenciar e incluir outros componentes.*

EXEMPLO

```
<h2>Hero List</h2>
<p><i>Pick a hero from the list</i></p>

<ul>
    <li *ngFor="let hero of heroes"
        (click)="selectHero(hero)">
            {{hero.name}}
    </li>
</ul>

<hero-detail *ngIf="selectedHero"
             [hero]="selectedHero">
</hero-detail>
```

Binding de evento

Angular Expression

Outro componente

Binding de propriedade

EXERCÍCIO 2

Criando o componente principal da aplicação

1. *No terminal, digite*

   ```
   ng g c chat
   ```

2. *Veja os arquivos do ChatComponent e crie o HTML.*

MÓDULOS

- *Agrupamento lógico (funcional) de componentes e serviços;*
- *Permite que os componentes sejam visíveis para a aplicação;*
- *Angular CLI já gera um módulo básico (que geralmente é suficiente para aplicações menores);*
- *Em aplicações maiores, é uma boa prática isolar cada "parte" do sistema em seu próprio módulo.*

EXEMPLO

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

@NgModule({
    declarations: [ AppComponent ],
    imports: [
        BrowserModule,
        ReactiveFormsModule
    ],
    providers: [ Logger ],
    bootstrap: [ AppComponent ]
})

export class AppModule { }
```

Bibliotecas do Angular

Componentes

Dependências

Serviços

EXERCÍCIO 3

Criando o componente de cabeçalho

1. *No terminal, digite*

```
cd chat
ng g c chat-header
```

2. *Veja os arquivos do ChatHeaderComponent e crie o HTML.*

INJEÇÃO DE DEPENDÊNCIA

- *Problema: relacionamento entre componentes*
  - *Aplicação gerenciava ciclo de vida de serviços, componentes etc.*
  - *Binding de componentes estaticamente no código → pouco reuso;*
- *Solução: injeção de dependência*
  - *Devs não precisam fazer bindings em tempo de compilação;*
  - *Aplicação só especifica pontos de injeção - framework lida com eles;*
  - *Injector repassa as dependências aos objetos que precisam delas;*
- *Outcome: baixo acoplamento e alta flexibilidade.*
- *TypeScript facilita ainda mais: private class members no construtor!*

SERVIÇOS

- *Define funções que podem ser usadas em toda a app;*
- *Geralmente associados a contextos maiores do que um único componente (ou view);*
  - *Singletons;*
- *Prática: usar @Injectable() - DI;*
- *Principais usos:*
  - *Comunicação entre componentes;*
  - *Realização de tarefas comuns (evitar duplicação de código);*
  - *Permitem maior nível de abstração.*

EXEMPLO

Boa prática - DI

Injeção de dependência

```
@Injectable()
export class EmployeeService {

    private employees: Employee[];

    constructor(
        private logger: Logger,
        private backend: BackendSe   ce) { }

    public getAllEmployees(): Employee[] {
        this.backend.getAll().then( (employee: Employee) => {
            logger.log(`Retrieved employee ${employee.name}`);
            this.employees.push(employee);
        });
        return this.employees;
}
```

EXERCÍCIO 4

Criando o serviço de back-end

1. *No terminal, digite*

```
ng g s chat
```

2. *Crie os métodos do serviço*

```
getMessages()
sendMessage(message)
```

PIPES (FILTROS)

- *Utilizados nos templates;*
- *Transformam a forma como um dado é exibido na view;*
- *Angular possui uma série de built-in pipes, que podem ser customizados;*
- *Podem ou não receber parâmetros.*
- *Interessante: https://dzone.com/articles/5-usage-ideas-for-angular-pipes*

EXEMPLO

Date Pipe

Parâmetro

```
<p>The employee's birthday is {{ birthday | date:"MM/dd/yy" }}</p>

<p>The total overdue amount is {{ amount | currency }}</p>
```

Currency Pipe

```
<p>Text variations: {{ text | uppercase }}, {{ text | lowercase }}</p>
```

EXERCÍCIO 5

Criando o componente de mensagens

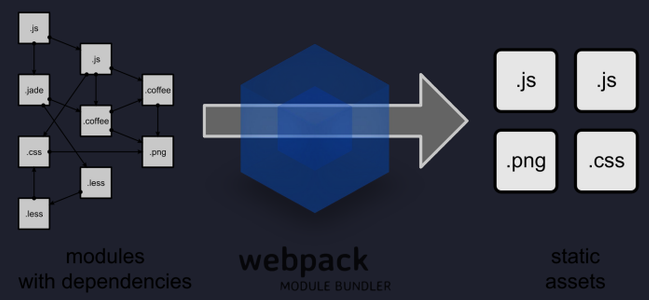
1. *No terminal, digite*

```
cd chat
ng g c chat-item
```

2. *Implemente o componente e utilize o* ***DatePipe*** *para exibir a data formatada!*

OUTRAS COISAS LEGAIS

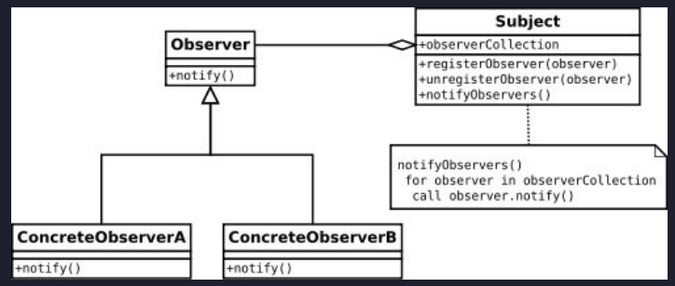
- *Formulários*
  - *Orientados a Templates;*
  - *Orientados a modelos (Reactive Forms);*
- *Rotas (e Guardas);*
- *Aplicações múltiplas vs múltiplos módulos;*
- *Modularização com Webpack.*

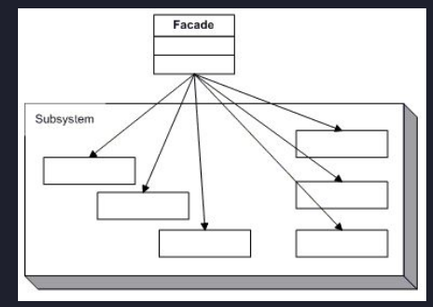


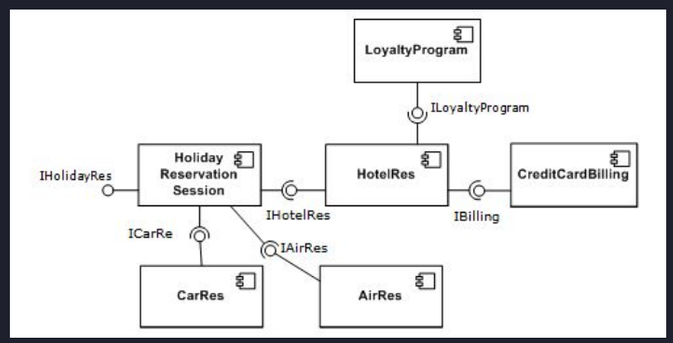
VANTAGENS DE UTILIZAR ANGULAR

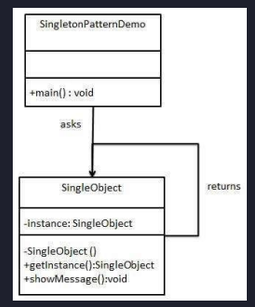
ANGULAR ENFORÇA O USO DE DESIGN PATTERNS

- *Singleton;*
- *Factory;*
- *Decorator;*
- *Façade;*
- *Observer;*
- *Module (vs MVC);*
- *Proxy;*
- *… and many others.*



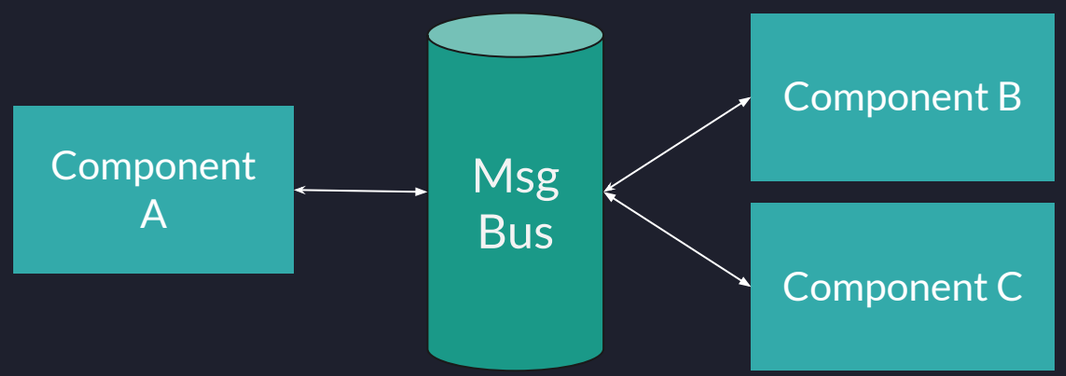






ANGULAR É FEITO PARA APLICAÇÕES GRANDES

- *Reuso de componentes e criação de libraries;*
- *Enterprise Integration Patterns;*
- *Testes unitários e end-to-end (possibilitando TDD);*
- *Boas práticas de desenvolvimento de software.*



ANGULAR TAMBÉM FOCA EM PERFORMANCE

- *Tree-Shaking*
- *Lazy Loading*
- *Minificação e Compressão*
  - ***Angular 5:*** *build optimizer, whitespace removal*
- *AOT*
  - ***Angular 5:*** *AOT all the time*

ONDE APRENDER?

PARA AS MENTES INQUIETAS

- *Tour of Heroes, tutorial (bem básico) no site do Angular;*
- *YouTube: conteúdo gratuito de qualidade;*
  - *Ótimo material da Loiane Groner;*
- *Udemy: ótimos cursos bons e baratos*
  - *Understanding TypeScript*
  - *Angular 5 - The Complete Guide*
  - *Reactive JS: Are you ready for the next big paradigm shift?*
- *Learn Angular 2, feito pela equipe do Ionic;*
- *Angular 4, ebook gratuito do Assim Hussain.*

PERGUNTAS?

MUITO OBRIGADO!

Adriano Gomes
adriano.gomes@venturus.org.br

Bruno Toffolo
bruno.toffolo@venturus.org.br

Renato Ribeiro
renato.ribeiro@venturus.org.br

venturus
inovação & tecnologia