# Estudo

## TypeScript

## TypeScript - Características e Configurações

Características:

* Superconjunto de JS
* Compila para JavaScript simples
* JS fortemente tipado
* POO
* Pré-processador (superset)
* A tipagem é um ponto forte pois evita bugs ainda em desenvolvimento
* É transpilado para ECMAscript pois os navegadores não interpretam TS
* Necessita NODEJS e NPM

Configurações:

* Compilador configurável -> arquivo tsconfig.json
* noEmitOnError em tsconfig.json previne que erros passem para o arquivo JS (deve ser ativado no arquivo, ou seja ser TRUE)
* Para utilizar um pacote que instalamos, precisamos fazer o import dele, como no caso do prompt-sync: import promptSync = require(‘prompt-sync’)
* Para utilizar uma função de Prompt-Sync, precisamos criar um const prompt = promptSync

**Comandos:**

* INSTALAÇÃO: npm install -g typescript (-g significa global)
* DICA: utiliza TSC para seus comandos
* TRANSPILAR (TS para JS) -> tsc nomeArquivo.ts
* EXECUTAR JS -> node nomeArquivo.js
* CRIAR tsconfig.json -> tsc –init
* TRANSPILAR TODOS -> tsc -w
* PROMPT-SYNC -> npm i prompt-sync

## TypeScript - Código

Devemos definir o tipo quando criamos uma varíavel (fortemente tipado)

var, let e const:

* let -> acessada apenas dentro do escopo
* var -> acessada de qualquer parte do código
* const -> readonly, imutável

Tipos:

* string

nome : string

* number

preco : number

* boolean

status : boolean

* array

numeros : number[] ou

numeros : Array

* readOnlyArray

pessoas = ReadOnlyArray

* tuplas

pessoa : [nome : string, idade : number, email : string]

* enum

export enum DiaSemana

{Segunda = 1, Terca = 2}

* union -> aceita ambos tipos especificados

exemplo : (string | number)

* any

exemplo : any

* never

usado para tratar situações que não deviam ocorrer

Funções

function nome(x : string) : string

function nome(x : number) : number

## RESUMO COMANDOS

Angular

* npm install -g angular@cli
* ng -v
* ng new nomeProjeto
* ng serve
* ng g component nomeComponente
* ng g class nomeClass
* npm i bootstrap, jquery, popper

TS

* npm install -g typescript
* tsc -v
* tsc nomeArquivo.ts
* tsc -w
* node nomeArquivo.js
* tsc –init
* npm i prompt-sync

## Angular

## Angular: - Característícas e Comandos

* Plataforma e framework para construção de interface de aplicações
* Precisa de Node e TS instalado

**Comandos:**

* INSTALAÇÃO -> npm install -g angular@cli
* VERIFICAR VERSÃO -> ng -v
* CRIAR PROJETO -> ng new NomeProjeto
* SERVIANDO -> ng serve

## Angular - Composição

Possui componentes, diretivas, roteamento, serviços, template, metadata, databinds e injeção de dependências

* Componentes:

Marcadores ou extensões de elementos que compõem o DOM, funcionalidade específica a um elemento

-> Template (HTML)

-> Metadata (Processamento e leitura das classes do Angular)

-> Dados da tela (DataBinding, dados e HTML)

-> Comportamento da view (junção template, controller e escopo do Angular)

* Serviço

-> Métodos para acesso

-> Regras de negócio

-> Manipulação de dados no banco

-> API

* Roteamento

-> Roteamento das páginas

-> SPA (single page application)

-> Caminhos relativos

-> Restrições de acesso

* Diretivas

-> Modificar comportamento do DOM

-> Exemplo: aumentar tamanho do input ao clicar

## Angular - Estrutura

app-root -> seletor existente no arquivo app.component.ts (cada componente tem seu seletor)

app.module.ts -> importação de todos módulos e componentes

Para incluir um componente em outro, precisamos fazer o import dele no arquivo TS de onde estamos inserindo o componente…

Por exemplo, ao inserir o componente navbar dentro de header, precisamos fazer a importação do módulo de NAVBAR em header.component.ts

import{NavbarComponent} from ‘caminho’

ADIÇÃO DE BOOTSTRAP

* npm i bootstrap

Para referenciar o bootstrap após instalado, precisamos acessar o arquivo angular.json, que é onde as configurações principais do projeto estão definidas

Dentro de build-> options, dentro de script e style, adicionamos a referencia aos arquivos adicionados a node\_modules

## Angular - Códigos

Para utilizar uma variável criada no TS de um componente:

{{this.variavel}}

DIRETIVAS

Utilizando DIRETIVA IF

* CommonModule deve ser adicionado aos imports do componente onde a diretiva será usada
* \*ngIf=“this.boolean”
* Se for false, o elemento com a diretiva será ocultado, se for true ele reaparece

UTILIZANDO DIRETIVA FOR

* Processa cada item de um objeto iterável (array de objetos por exemplo) gerando uma marcação para cada um
* Diretiva estrutural
* Útil para conteúdos repetidos, como cards
* \*ngFor=“let item of items; index as i”
* {{item.nome}} -> acessa o atributo nome do item atual
* Podemos criar um objeto em uma classe e então criamos o array de objetos dessa classe para iterar no ngFor

ROUTES

Usadas para redirecionamento entre componentes

* app-routes.ts
* Devemos adicionar nossas rotas ao arquivo app-routes.ts, em Routes
* Cada rota tem path (caminho) e componente (oq vai ser carregado)
* Carrega o componente referente a rota na tela
* RouterOutlet renderiza o componente da rota
* Permite passagem de parâmetros (só adicionar no atributo path)

NodeJs:

* Runtime JavaScript
* Ambiente de servidor de código aberto
* Executar JavaScript no servidor
* Conteúdo dinâmico
* Programação assincrona

Instalação:

site do node

node -v (verifica versão do node)

NPM (Node Package Manager)

* Repo online para publicação de projetos de código aberto
* Instalação de pacotes, gerenciamento de versão
* Gerenciamento de dependências