

## Documentação de Apoio

- Slides disponíveis na plataforma moodle da Unidade Curricular
- Angular – <https://angular.io/>
- TypeScript – <https://www.typescriptlang.org/>
- TypeScript – <https://www.typescriptlang.org/docs/handbook/basic-types.html>
- TypeScript – <https://www.tutorialspoint.com/typescript/>

# 1 TypeScript

## 1.1 Instalação

Instale o compilador de TypeScript no seu ambiente de desenvolvimento. Use as indicações do slide 48 da aula teórica.

## 1.2 Demonstração

De seguida deve testar o código apresentado no slide 23 usando um ficheiro com a extensão “.ts”. Compile o código com o comando tsc (slide 27) e verifique o resultado. Acrescente código para testar um objeto da classe Car e executar o método disp(); Corra o resultado no runtime de node.js

## 1.3 Classes

Adicione os tipos dos parâmetros nas variáveis do construtor e crie variáveis para guardar os parâmetros do construtor num ficheiro “Person.ts”

```
export class Person {  
  constructor(name, age) {  
    this.name = name;  
    this.age = age;  
  }  
}
```

Num novo ficheiro, “main.ts” importe a classe criada e teste objetos desta classe.

Teste o resultado no runtime de node.js após efetuar a compilação dos ficheiros com o comando:

- tsc Person.ts main.ts
- node main.js

Crie classes para utilizadores específicos (ex: Admin, TechSupport, Users). Crie o método sayMyRole onde cada objecto de cada classe indica o seu role na aplicação. Use herança na criação de classes.

## 1.4 Interfaces

Verifique se o exemplo fornecido está correto. Compile e execute no runtime node.js. Foram encontrados alguns problemas? Qual o objetivo do uso de interfaces?

Crie um novo método que teste se o estudante é maior de idade. Teste o método com o objeto já criado no exemplo.

```
class Student {  
  fullName: string;  
  constructor(public firstName: string, public middleInitial: string,  
    public lastName: string, public age: number) {  
    this.fullName = firstName + " " + middleInitial + " " + lastName;  
  }  
}
```

```
}

interface Person {
  firstName: string;
  lastName: string;
}

function greeter(person: Person) {
  return "Hello, " + person.firstName + " " + person.lastName;
}

let user = new Student("Jane", "M.", "User");

console.log(greeter(user));
```

## 1.5 Importação de módulos

Crie os seguintes ficheiros main.ts e person.ts . Compile o programa usando o comando:

- tsc main.ts

Verifique que foram compiladas ambas os ficheiros pois existe uma relação entre eles.

```
import {Person} from './Person';

var p = new Person("me", 21);
```

**Ficheiro 1 - Main.ts**

```
export class Person {

  private name : string;
  private age: number;

  constructor(name, age) {
    this.name = name;
    this.age = age;
  }


  public isOverAge(): boolean{
    return (this.age > 18);
  }
}
```

**Ficheiro 2 - Person.ts**

Use a mesma estratégia para as classes criadas no exercício 1.3 e 1.4, usando o ficheiro main.ts para demonstrar o seu uso.

## 1.6 Documentação

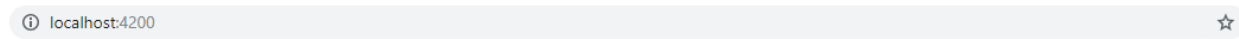
Aceda ao site <https://www.typescriptlang.org/docs/home.html> . Nesta página de internet poderá verificar a documentação da linguagem TypeScript. Em alternativa, poderá consultar o tutorial em <https://www.typescriptlang.org/docs/home.html>.

|  |   |
|--|---|
| <br>ESCOLA<br>SUPERIOR<br>DE TECNOLOGIA<br>E GESTÃO | Programação em Ambiente Web<br>Docentes – FAS, EFE, JCarneiro, OAO<br>Ficha Prática 8 |
|--|---|

## 2 “Hello World” Angular

Tenha em consideração o slide 53 – 55 da aula teórica e desenvolva uma pequena aplicação em Angular para fornecer o website padrão.

Altere os ficheiros “app.component.ts” e “app.component.html”, para que a página da aplicação web em angular fique com o aspeto apresentado na Figura 1.



**Hello World!**

Figura 1 – Hello World em Angular

### 3 (Opcional) – Tutorial Angular

Neste exercício vamos fazer uso do tutorial padrão fornecido pela framework Angular. Desta forma aceda à página de internet: <https://angular.io/tutorial> e execute os passos 1 a 6 do tutorial proposto.

No fim deste tutorial deve aperceber-se do uso de componentes, routing, serviços e layouts master/detail em Angular. Estes conceitos irão ser úteis em aplicações Angular mais avançadas.