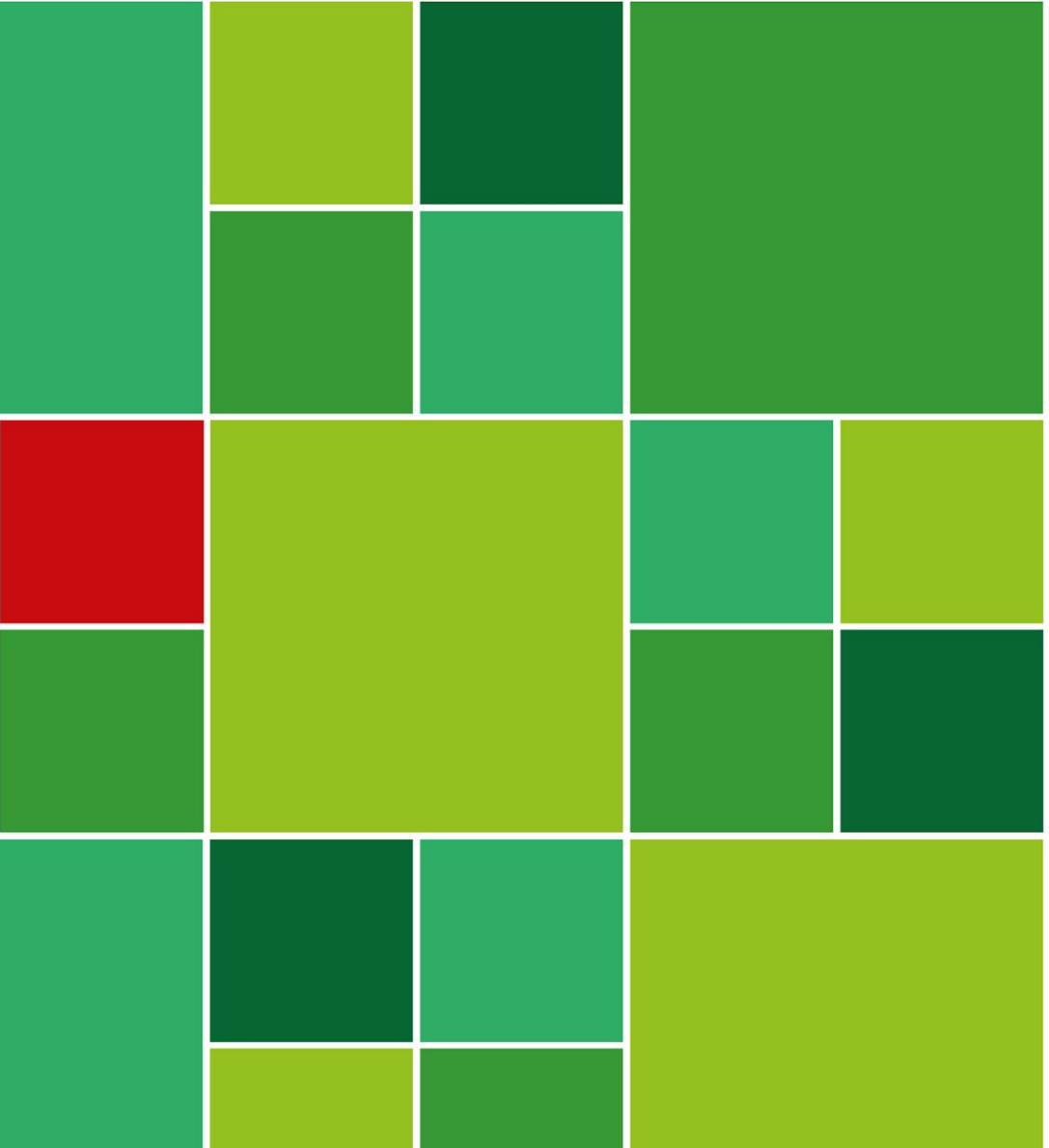


Programação WEB III

Avanços do TypeScript

Prof. Dr. Danilo Barbosa

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PERNAMBUCO

Conteúdo

- Repetição em TypeScript
- condicional em TypeScript
- Função em TypeScript
- Referências

- **Tomada de Decisão**
  - if
  - else
  - else if
  - switch

```
if (true) {
    if (10) {
        console.log("apareci");//apareci
        if (0) {
            console.log("não apareci");
        }
        else if ("2") {
            console.log("apareci");//apareci
        }
    }
}
```

```
var x = 3;
switch (x)
{
    case 0:
        console.log("passei por
        aqui");  break;
    case 1:
        console.log("passei por
        aqui");  break;
    default:
        console.log("não passei por
        nada");
}
```

INSTITUTO FEDERAL
Espírito Santo

- **Loops**
  - For
  - For of
  - For in
  - while
  - do ... while
- **Escapes**
  - Break
  - Continue
  - Return

- For of Vs For in

  For in serve para inspecionar propriedades do objeto, sempre fazendo um loop em cima dos nomes das propriedades, enquanto for of é para repetir dados, funcionando apenas em objetos iteráveis.

Exemplos

```
for (var x = 0; x < 2; x++) {
    console.log("iteração n° ",
    x);
}
```

```
var vetor = ["Ai", "meu", "deus", "o",
"que", "eu", "to", "fazendo",
"da",  "minha", "vida"];
for (var value of vetor)
    {
    console.log(value);
}
var obj = { nome: "cão brabo", atack: 50, defesa:
20,  habilidade: "morder" }
for (var propriedade in obj) {
    console.log(obj[propriedade])
    ;
}
```

INSTITUTO FEDERAL
Espírito Santo

```
while ("s" && 0) {
    console.log("que
    bizarro");
}

do {
    console.log("normal por
    aqui");
} while (true);
```

- **Funções**
  - Funções nomeadas
  - Funções anônimas
  - Funções seta gorda ou lambda
  - Funções do gerador

- **Parâmetros**
  - Momento: avaliação normal.
  - Direção: tipos primitivos por cópia, enquanto objetos são por cópia de referência.
  - Mecanismo: posicional.

- Função nomeada

```
function MinhaFunção(x: number, y: number): any
    {   return (x ** y);
}
```

- Função anônima

```
var anonimo = function () {
    console.log("Mr.Nobody")
    ;
}
```

- Função seta gorda

  Usada em funções anônimas, o uso da seta => implica em não utilizar mais a palavra chave function.

```
let foo = () => console.log("função lambda ou seta
gorda");
let bar = (a: number, b: number) => a + b;
foo();//função lambda ou seta gorda
console.log(bar(2, 5));//7
```

- Funções geradoras

São funções iteráveis que acumulam valores antigos a partir da palavra chave yield. Os valores são obtidos pela chamada do next().

```
function* pares(init: number)     {
    var a = init;
    while (true) {
        a == 0 ? yield a += 2     : yield a *= 2;
    }
}
```

```
var generator = pares(0);
var x = 0;
while (x < 3) {
    console.log(generator.next());
    x++;
}
```

Saídas

```
{ value: 2, done: false }
{ value: 4, done: false }
{ value: 8, done: false }
```

- Características

Funções podem sofrer elevação assim como aconteceu com declarações de variáveis.

```
m1();//subiiiiiiiiiindo!


function m1() {


        console.log("subiiiiiiiiiindo!");
}
```

Também podemos trabalhar com parâmetros opcionais, default, e funções que esperam argumentos variados.

```
function option(first = "Default", second: string,
third?:  string) {
    console.log(first, second, third);
}
option(undefined, "au au");//Default au au
undefined  option("mudei", "miau",
"mordi");//mudei miau mordi
```

```
function option(first: string, second?: string, third =
    "me") {  console.log(first, second, third);
}
option("help");//help undefined me
option("i need", "somebody", "help");//i need somebody help
```

```
function option(...args) {
    for (var value of args)
    {
        console.log(value);
    }
}
option("Eu", "quero", "cafe!!")//Eu
\nquero\ncafe!!
```

- **Módulos**
  - Arquivo
  - Compilação
  - Importação

- Arquivo utils.ts

Cria-se o arquivo. Todos os elementos que tem que ser visíveis fora do módulo tem que ter a palavra chave export.

```
export function calcArea(a: number, b: number)
    {   return a * b;

}
export function calcVolume(a: number, b: number, c:
    number) {   return a * b * c;

}
```

INSTITUTO FEDERAL
Espírito Santo

- Compilação

Para poder usar o arquivo utils.ts como um módulo é necessário compilar o código de forma a gerar um arquivo módulo.
> “tsc --module amd

utils\utils.ts” isso vai gerar um

arquivo utils.js.

- Importação

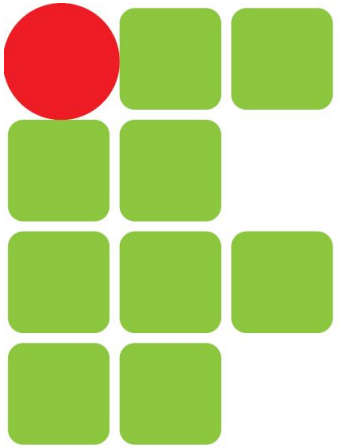
```
import * as util from "./utils/utils"
console.log(util.calcArea(1, 2));//2
console.log(util.calcVolume(1,2,3));//6

import {calcArea} from "./utils/utils"
console.log(calcArea(1, 2));//2
console.log(util.calcVolume(1,2,3));//erro
```

Referências

TypeScript is JavaScript with syntax for types. https://www.typescriptlang.org

Guia Pratico de Typescript https://github.com/KAYOKG/BibliotecaDev/blob/main/LivrosDev/Guia%20pr%C3%A1tico%20de%20TypeScript%20-%20Melhore%20suas%20aplica%C3%A7%C3%B5es%20JavaScript%20-%20Autor%20(Casa%20do%20C%C3%B3digo).pdf

TypeScript TypeScript is a typed superset of JavaScript that compiles to plain JavaScript. https://devfreebooks.github.io/typescript/



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PERNAMBUCO