JavaScript parte 1

DIO CEZAR GO

diogoc@utfpr.edu.br

www.diogocezar.com

UTFPR 2018/02

O QUE É?

- é uma linguagem de programação ***interpretada*** com características de orientação a objetos;
  - desenvolvida pela Netscape a fim de estender as capacidades de seu *browser*;
  - permite que um conteúdo executável seja incluído em páginas *web*;
  - sintaticamente semelhante a C e C++;

O QUE É?

- uma linguagem de programação utilizada para criar páginas web interativas;
- é executada totalmente no computador do cliente ou seja, navegador;
- é uma linguagem baseada em objetos;
- não é necessário compilar;
- não é necessário um servidor web;
- é só criar um arquivo .js e começar a codificar!

JAVASCRIPT NÃO É JAVA!

- eles têm um nome similar mas são completamente diferentes;
- ***Java*** é um linguagem mais complexa;
    - uma linguagem de alto nível;
- ***JavaScript*** é uma linguagem de scripts! que é interpretada pelo navegador quando a sua página HTML é lida;

O BASICÃO DO JAVASCRIPT

- funções;
- variáveis;
- arrays
- loops;
- condicionais;
- comparação e operadores;
- objetos;
- funções;
- tipos de dados;
- protótipos;
- eventos;

O QUE EU PRECISO PARA RODAR JS?

- tudo o que você precisa é um navegador moderno;
  - Google Chrome;
  - Mozilla Firefox;
  - Safari;
  - ~~Internet Explorer~~;
- um editor de textos de sua preferência: eu utilizo o Visual Studio Code;

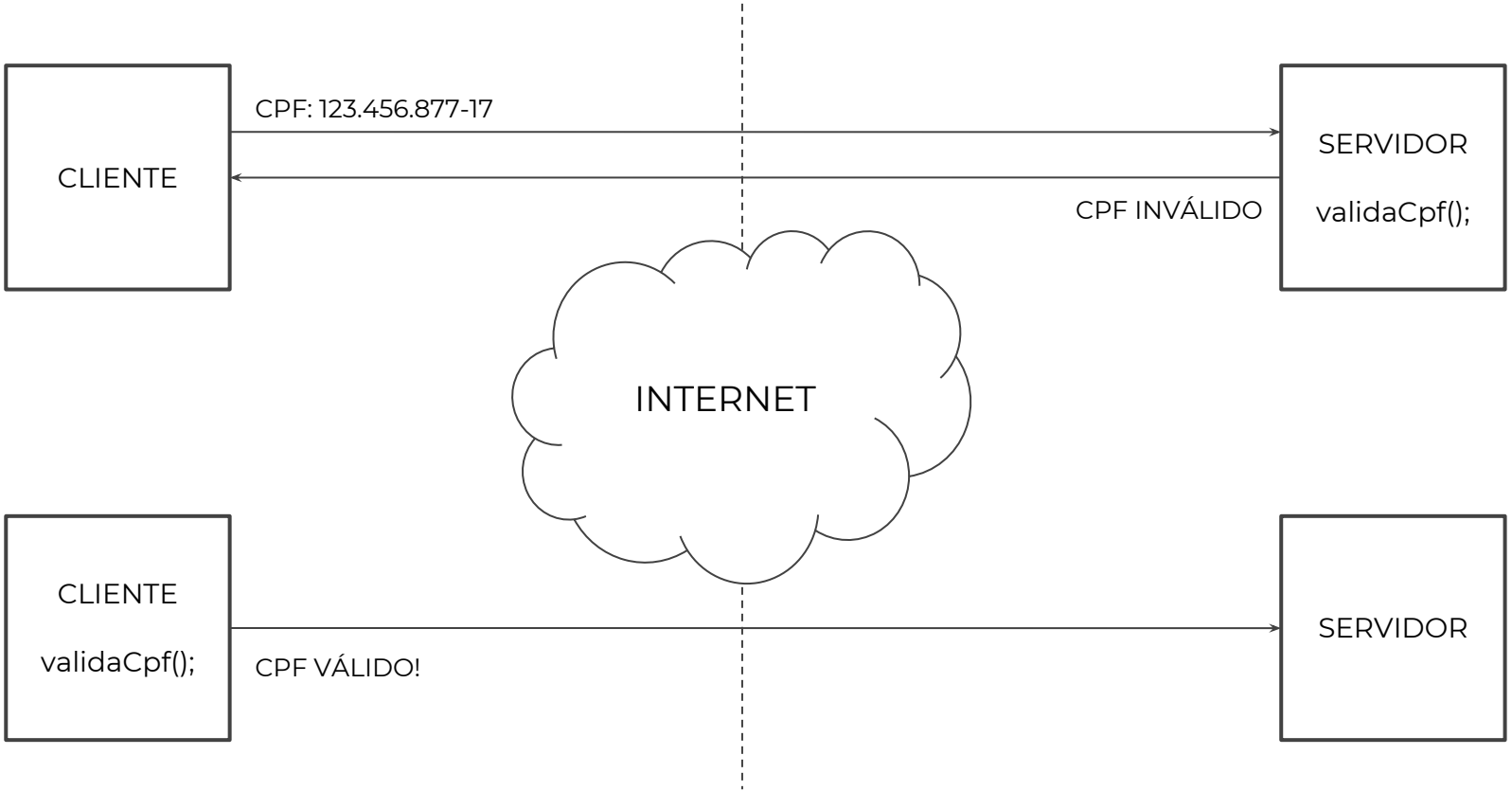
HTML VS JAVASCRIPT

- eles são completamente diferentes, mas trabalham juntos!
- HTML é uma linguagem de marcação;
- JavaScript é uma linguagem de programação;
- o JS utiliza eventos e ações para tornar uma página HTML interativa;
- por isso, você precisa ter conhecimentos de HTML;

E O QUE CONSIGO FAZER COM JAVASCRIPT?

- colocar um conteúdo em uma página HTML em tempo de execução;
- detectar versões e informações do navegador;
- criar cookies;
- validar formulários;
- criar animações, slideshows, scrollers, carrosséis;
- controlar o navegador → abrir novas janelas, obter suas dimensões, etc...;
- criar aplicativos complexos com frameworks: AngularJS, ReactJS, VueJS;

EXEMPLO DE FUNCIONAMENTO



SINTAXE E ESTRUTURA

- é uma linguagem case sensitive;
- ignora espaços em branco;
- ponto e vírgula é opcional;
- suporta comentários no estilo clássico /* */ e //;
- não é ***necessário*** realizar a declaração de variáveis;
- até a especificação **ES5** tudo é função!
  - consegue-se *emular* classes com prototype;
- tipos primitivos: *string*, *number*, *boolean*, *undefined* e *null*;
  - são implícitos;
- *objetos* e *arrays* → a grande sacada!

funções

FUNÇÕES

- a definição da função (também chamada de declaração de função) consiste no uso da palavra chave *function*, seguida por:
  - nome da função;
  - lista de argumentos para a função, entre parênteses e separados por vírgulas;
  - declarações JavaScript que definem a função, entre chaves { }.

EXEMPLO DE FUNÇÃO

pode-se por exemplo definir uma função que retorna o quadrado de um número:

COMO FAZER?

```
function square(numero) {
     return numero * numero;
}

console.log(square(2));
// 4
```

EXEMPLO DE FUNÇÃO

uma função, ainda pode ser atribuída a uma variável:

EXEMPLO DE EXPRESSÃO DE FUNÇÃO

```
var square = function(numero) {return numero * numero};
var x = square(4) //x recebe o valor 16
```

FUNÇÕES

- as funções podem ainda ser passadas como parâmetro de uma outra função;
- muito utilizado na linguagem JavaScript;
- exemplo: callBacks;

EXEMPLO DE FUNÇÃO

função como parâmetro

EXEMPLO DE FUNÇÃO COMO PARÂMETRO

```
function print(f){
     f('testing');
}

print(function(param){
     console.log(param);
})
```

hierarquia de objetos

HIERARQUIA DE OBJETOS

- **window**: o objeto mais acima na hierarquia, contém propriedades que se aplicam a toda a janela.
  - **location**: contém as propriedades da URL atual;
  - **history**: contém as propriedades das URLs visitadas anteriormente
  - **document**: contém as propriedades do documento contido na janela, tais como o seu conteúdo, título, cores, etc;

incluindo um js

INCLUINDO UM ARQUIVO EXTERNO

como se faz para incluir um arquivo js em uma página html?

COMO FAZER?

```
<!DOCTYPE html>
<html lang="pt-br">
     <head>
          <meta charset="utf-8">
          <title></title>
     </head>
<body>
     <script language="JavaScript" src="./js/myLib.js"></script>
</body>
</html>
```

INCLUINDO UM ARQUIVO EXTERNO

como se faz para incluir um arquivo js em uma página html?

COMO FAZER?

```
<!DOCTYPE html>
<html lang="pt-br">
     <head>
          <meta charset="utf-8">
          <title></title>
     </head>
<body>
     <script>
          console.log('teste');
     </script>
</body>
</html>
```

variáveis e tipos de dados

VARIÁVEIS E TIPOS DE DADOS

- temos alguns "tipos de dados" interessantes em JS:
  - Number
  - String
  - Array
  - Object

VARIÁVEIS E TIPOS DE DADOS

exemplos de variáveis

ALGUNS EXEMPLOS DE VARIÁVEIS

```
var number1 = 35; // Number
var number2 = 40;
     console.log(number1 + number2);
var number3 = '35'; // String
var number4 = '40';
     console.log('My number: ' + number3 + number4);
```

VARIÁVEIS E TIPOS DE DADOS

- convenções para os nomes de variáveis:
  - as variáveis podem contem:
    - letras;
    - números;
    - underlines;
    - sinal de dollar ($);
  - uma variável sempre deve começar com uma letra;
  - ou iniciar com underline (_) ou iniciar com dollar ($);
  - também são case sensitive, ou seja diferencia-se maiúsculas de minúsculas;

CONVENÇÕES DE NOMENCLATURA

- camel case → `var myFavoriteNumber;`
- partial case → `var MyFavoriteNumber;`
- underscore → `var my_favorite_number;`

ARRAYS

- são os nossos vetores;
- guardam múltiplos valores em uma mesma variável;
- iniciam do índice 0;
- vão até o índice n-1;
- um array é um objeto, e por isso possui métodos e propriedades;

VARIÁVEIS E TIPOS DE DADOS

exemplos de arrays

ALGUNS EXEMPLOS DE ARRAYS

```
var colors = ['red', 'blue', 'green'];
console.log(colors);
console.log(colors[1]);

var colors = new Array('red, 'yellow', 'orange');
colors[3] = 'green';
// push é um método do objeto array
colors.push('purple');

console.log(colors);
```

VARIÁVEIS E TIPOS DE DADOS

exemplos de arrays

ALGUNS EXEMPLOS DE ARRAYS

```
// podemos ter múltiplos tipos de dados em um mesmo array
var numbers = [5,77,6,'Seven'];
console.log(numbers[0] + numbers[3]);
// esta é uma propriedade
console.log(numbers.length);
// estes são métodos
console.log(numbers.sort());
console.log(numbers.reverse());
```

laços

LAÇOS

- você já sabe como isso funciona, mas...
- temos algumas particularidades úteis em JavaScript;

LAÇOS

exemplos de laços

ALGUNS EXEMPLOS DE LAÇOS

```
for(var i = 0; i<10; i++){
     console.log(i);
}
```

LAÇOS

exemplos de laços

ALGUNS EXEMPLOS DE LAÇOS

```
var i = 0;
while(i < 10){
     console.log(i);
     i++;
}
```

LAÇOS

exemplos de laços

ALGUNS EXEMPLOS DE LAÇOS

```
var numbers = [33, 54, 76, 34, 2, 6];
numbers.forEach(function(number){
     console.log(number)
});
```

LAÇOS

exemplos de laços

ALGUNS EXEMPLOS DE LAÇOS

```
var numbers = [33, 54, 76, 34, 2, 6];
for(var i = 0; i < numbers.length; i++){
     console.log(numbers[i]);
}
```

condições

CONDIÇÕES

- você já sabe como isso funciona, mas...
- podemos comparar de algumas formas diferentes por aqui...
- `if(1 = 1) // isto é uma atribuição`
- `if(1 == 1) // isso é uma comparação comum;`
- `if(1 === '1') // isso é uma comparação de tipos;`

CONDIÇÕES

exemplos de condições

ALGUNS EXEMPLOS DE CONDIÇÕES

```
if(1 == 1) console.log('Oks!');
if(1 == 2) console.log('Oks!');
if(1 == '1') console.log('Oks!');
if(1 === '1') console.log('Oks!');
```

CONDIÇÕES

exemplos de condições

ALGUNS EXEMPLOS DE CONDIÇÕES

```
var var1 = 3;
var var2 = 4;
if(var1 == var2) console.log('Oks!');
if(var1 != var2){
    console.log('Oks!');
} else {
    console.log('Not ok!');
}
if(var1 == var2 && var1 == 3) console.log('Oks!');
if(var1 == var2 || var1 == 3) console.log('Oks!');
```

SWITCH

- em JS o switch pode ser bem interessante;
- diferente da linguagem c, por exemplo, que só aceita números;
- podemos testar quaisquer valores;

CONDIÇÕES

exemplos de switch

ALGUNS EXEMPLOS DE CONDIÇÕES

```
var fruit = 'apple';
switch(fruit){
     case 'banana' : console.log('bananas!'); break;
     case 'apple' : console.log('apples!); break;
     case 'orange' : console.log('oranges!'); break;
     default : console.log('not selected');
}
```

objetos

OBJETO LITERAL

- **o que é?** → é um tipo básico de objetos em *JavaScript*;
- formato popularizado através do *JSON*;
- objeto é criado utilizando um par de chaves {};
- suas propriedades e métodos são públicos;
- todo objeto literal é único (como se fosse static);
- seu uso é recomendado em situações onde não podem existir mais de uma instância do objeto;
- **DICA** → utilize-o para representar uma página html e organizar sua estrutura JavaScript;

OBJETO LITERAL

exemplos de objeto literal

ALGUNS EXEMPLOS DE OBJETO LITERAL

```
var person = {
     firstName: 'Brad',
     lastName: 'Traversy',
     age: 34
}

console.log(person.firstName);
```

OBJETO LITERAL

exemplos de objeto literal

ALGUNS EXEMPLOS DE OBJETO LITERAL

```
var person = {
     firstName: 'Brad',
     lastName: 'Traversy',
     age: 34
}

console.log(person.age);
```

OBJETO LITERAL

exemplos de objeto literal

ALGUNS EXEMPLOS DE OBJETO LITERAL

```
var person = {
     firstName: 'Brad',
     lastName: 'Traversy',
     age: 34,
     children: ['Brianna', 'Nicholas']
}

console.log(person.children[0]);
```

OBJETO LITERAL

exemplos de objeto literal

ALGUNS EXEMPLOS DE OBJETO LITERAL

```
var person = {
      firstName: 'Brad',
      lastName: 'Traversy',
      age: 34,
      children: ['Brianna', 'Nicholas'],
      address: {
            street: '555 Something st',
            city: 'Boston',
            state: 'MA'
      }
}
console.log(person.address);
console.log(person.address.city);
```

OBJETO LITERAL

ALGUNS EXEMPLO DE FUNÇÕES EM OBJETOS LITERAIS

```
var person = {
      firstName: 'Brad',
      lastName: 'Traversy',
      age: 34,
      children: ['Brianna', 'Nicholas'],
      address: {
            street: '555 Something st',
            city: 'Boston',
            state: 'MA'
      }
      fullName: function(){
            // this faz referência ao objeto atual
            return this.firstName + " " + this.lastName;
      }
}
console.log(person.fullName());
```

OBJETO CONSTRUTOR

- **o que é?** → nada mais é que uma função JavaScript;
- pode ser executada como uma função ou...
- ser utilizada para instanciar um objeto utilizando a palavra reservada *new*;
- se executada como uma função normal → this equivalerá ao window;

OBJETO CONSTRUTOR

EXEMPLO DE OBJETO CONSTRUTOR

```
var apple = new Object();

apple.color = 'red';
apple.shape = 'round';

apple.describe = function(){
      return 'An apple is the color ' + this.color + ' and is the shape ' +
this.shape;
}

console.log(apple.describe());
```

OBJETO CONSTRUTOR

- mas a grande sacada é que pode-se utilizar isso para construir objetos mais genéricos...
- se tivéssemos, laranjas, bananas...?
- utilizamos então um ***constructor pattern***;

CONSTRUCTOR PATTERN

EXEMPLO DE CONSTRUCTOR PATTERN

```
function Fruit(name, color, shape){
     this.name = name;
     this.color = color;
     this.shape = shape;
}

var apple = new Fruit('apple', 'red', 'round');

console.log(apple);
```

CONSTRUCTOR PATTERN

EXEMPLO DE CONSTRUCTOR PATTERN

```
function Fruit(name, color, shape){
     this.name = name;
     this.color = color;
     this.shape = shape;
}

var apple = new Fruit('apple', 'red', 'round');
var melon = new Fruit('melon', 'green', 'round');

console.log(melon.shape);
```

CONSTRUCTOR PATTERN

EXEMPLO DE CONSTRUCTOR PATTERN

```
function Fruit(name, color, shape){
     this.name = name;
     this.color = color;
     this.shape = shape;
     this.describe = function(){
          return 'A ' + this.name + ' is the color ' + this.color
+ 'and id the shape ' + this.shape;
     }
}

var melon = new Fruit('melon', 'green', 'round');

console.log(melon.describe());
```

arrays de objetos

ARRAYS DE OBJETOS

- pode ser interessante trabalhar com arrays de objetos;
- como vimos, um array pode ter qualquer tipo;
- até mesmo outros objetos, e outros arrays;
- dentro dos objetos, também podemos declarar arrays como propriedades;

ARRAYS DE OBJETOS

EXEMPLO DE ARRAYS DE OBJETOS

```
var users = [
      {
            name: 'John Doe',
            age : 30
      },
      {
            name: 'Mark Smith',
            age : 44
      },
      {
            name: 'Shelly Williams',
            age : 20
      }
];

console.log(users[0].name);
console.log(users);
```

interações com HTML

CAPTURANDO UM ELEMENTO

pode-se fazê-lo de algumas formas:

COMO FAZER?

```
<script>
     var nome = document.getElementById("nome");
     var nome = document.getElementByName("nome");
</script>
```

- a partir desse momento a variável **nome** assume as propriedades do elemento com id nome;
- se for um *input* terá o atributo **value**;
- se for um *div* terá o atributo **style**;

CAPTURANDO UM ELEMENTO

ou de uma forma mais moderna (parecido com JQuery):

COMO FAZER?

```
<script>
     var nome = document.querySelectorAll('.name');
     var nome = document.querySelector('#name');
</script>
```

ALTERANDO UM ESTILO

pode-se alterar o estilo de um elemento por seu **id**:

COMO FAZER?

```
<script>
     var texto = document.getElementById('mudaCor');
     texto.style.color = 'red';
</script>
```

ALTERANDO O CONTEÚDO DE UMA DIV

pode-se alterar o estilo de um elemento por seu **id**:

COMO FAZER?

```
<script>
     var myDiv = document.getElementById(‘myDiv’);
     myDiv.innerHTML = 'O texto foi trocado';
</script>
```

eventos

EVENTOS

- os eventos são a forma de adicionar funcionalidades aos elementos do HTML;
- existem diversas formas de se fazer isso;
- mas afinal o que são?
    - "quando" se clica;
    - "quando" se passa o mouse;
    - "quando" se faz o scroll;

EVENTOS

- estão atreladas quase sempre a ações por partes dos usuários:
  - *onclick*;
  - *onmouseenter*;
  - *onmouseleave*;
  - *onkeyup*;
- mais sobre eventos:
  - http://bit.ly/2q1xCjY
  - http://bit.ly/2GLq5Ap

EVENTOS

EXEMPLO DE ATRIBUIÇÃO DE EVENTOS

```
<button onclick="doClick()">Click Me!</button>
<script>
     function doClick(){
          alert('You Clicked!');
     }
</script>
```

EVENTOS

EXEMPLO DE ATRIBUIÇÃO DE EVENTOS

```
// note que, dentro das " " podemos usar qualquer comando JavaScript
<button onclick="this.innerHTML = 'You Clicked!';">Click Me!</button>
```

EVENTOS

EXEMPLO DE ATRIBUIÇÃO DE EVENTOS

```
<button onclick="changeText(this)">Click Me!</button>
<script>
      function changeText(obj){
            obj.innerHTML = 'You Cliked!';
      }
</script>
```

EVENTOS

EXEMPLO DE ATRIBUIÇÃO DE EVENTOS

```
<h1 id="heading">Hey There!</h1>
<button onclick="changeText()">Click Me</button>
<script>
      function changeText(){
            var heading = document.getElementById('heading');
            heading.innerHTML = 'You Clicked!';
      }
</script>
```

EVENTOS

- mas calma, isso de colocar a chamada de eventos direto no HTML vai MUITO de encontro com a separação de código!
- JavaScript, tem que ficar junto com JavaScript;
- HTML junto com HTML!
- então qual a solução?
    - ***addEventListner***

EVENTOS

EXEMPLO DE ATRIBUIÇÃO DE EVENTOS

```
<h1 id="heading">Hey There!</h1>
<button id="btn-click">Click Me</button>
<script>
     function changeText(){
          var heading = document.querySelector('#heading');
          heading.innerHTML = 'You Clicked!';
     }
     var button = document.querySelector('#btn-click');
     button.addEventListner('click', changeText);
</script>
```

forms

FORMS

- ***JavaScript*** é uma ótima ferramenta para validação avançada de formulários;
- ***porque?*** → com JS é possível validar, limitar, formatar e melhorar a interação dos usuários com os formulários de sua página;
- pare e pense, o que se usa em um sistema web para inserção de dados? ***forms***!

FORMS

EXEMPLO DE JS COM FORMULÁRIOS

```
<form>
     <label>First Name</label>
     <input type="text" id="firstName"/>
     <label>Last Name</label>
     <input type="text" id="lastName"/>
     <button id="send">Send</button>
</form>
```

FORMS

EXEMPLO DE JS COM FORMULÁRIOS

```
<script>
     function validateForm(){
          var firstName = document.querySelectorAll('#firstName').value;
          var lastName  = document.querySelectorAll('#lastName').value;
          if(firstName != '' && lastName != ''){
               alert('Form Ok!');
          }
          else{
               alert('Form Not Ok.');
          }
     }
     var button = document.querySelector('#send');
     button.addEventListner('click', validateForm);
</script>
```

coisas adicionais

APRENDAM A DEBUGAR!

http://bit.ly/2pZU0LA

UM GUIA COMPLETO PARA APRENDER MÓDULOS

http://bit.ly/2EfbJ5N

JAVASCRIPT 30

https://javascript30.com/

recomendações de estudo

RECOMENDAÇÕES DE ESTUDO

- obviamente nossa matéria é resumida!
- existem MUITAS outras coisas sobre JavaScript que veremos rapidamente:
  - as novidades do ES6;
    - arrow functions;
    - let e const;
    - map;
    - reduce;
    - filter;
    - sort;
  - promises;
  - fetch;
  - async e await; (não veremos)

exemplos

ESTUDO DE EXEMPLOS

https://diogocezar.github.io/sandybox/#/js-example-1	Exemplos de Função
https://diogocezar.github.io/sandybox/#/js-example-2	Exemplos de Função em Variável
https://diogocezar.github.io/sandybox/#/js-example-3	Exemplos de Função como CallBack
https://diogocezar.github.io/sandybox/#/js-example-4	Exemplos de Variáveis e Arrow Functions
https://diogocezar.github.io/sandybox/#/js-example-5	Exemplos de Manipulação de Arrays
https://diogocezar.github.io/sandybox/#/js-example-6	Exemplos de Comparações entre Variáveis
https://diogocezar.github.io/sandybox/#/js-example-7	Exemplo de Objeto Literal
https://diogocezar.github.io/sandybox/#/js-example-8	Exemplo de Objeto Construtor
https://diogocezar.github.io/sandybox/#/js-example-9	Exemplo de Array de Objetos
https://diogocezar.github.io/sandybox/#/js-example-10	Exemplos de Interações com o HTML
https://diogocezar.github.io/sandybox/#/js-example-11	Exemplo Validação Formulário
https://diogocezar.github.io/sandybox/#/js-example-12	Exemplo Orientação a Objeto
https://diogocezar.github.io/sandybox/#/js-example-13	Exemplo Promise

materiais complementares

MATERIAIS COMPLEMENTARES

- http://bit.ly/2GtIqlY
- http://bit.ly/2H4wjZR
- https://mzl.la/2Gt9Kg9
- http://bit.ly/2JcFZC6
- http://bit.ly/2H3wVPj
- https://mzl.la/2q2sFHJ
- http://bit.ly/2McDYKw


The End