FIAP GRADUAÇÃO

FIAP

Responsive Web Development

PROF. RAFAEL MATSUYAMA
profrafael.matsuyama@fiap.com.br

JavaScript

O QUE É JAVASCRIPT

FIAP

- ***JavaScript*** é uma linguagem de programação interpretada. Foi originalmente implementada como parte dos navegadores web para que scripts pudessem ser executados do lado do cliente e interagissem com o usuário sem a necessidade deste script passar pelo servidor, controlando o navegador, realizando comunicação assíncrona e alterando o conteúdo do documento exibido.
- É atualmente a principal linguagem para programação cliente-side em navegadores web. Começa também a ser bastante utilizada do lado do servidor através de ambientes como o node.js. Foi concebida para ser uma linguagem script com orientação a objetos baseada em protótipos, tipagem fraca e dinâmica e funções de primeira classe. Possui suporte à programação funcional e apresenta recursos como fechamentos e funções de alta ordem comumente indisponíveis em linguagens populares como Java e C++. É a linguagem de programação mais utilizada do mundo.

INTRODUÇÃO

FIAP

JAVASCRIPT É UMA LINGUAGEM DE SCRIPTS MAIS UTILIZADA NA INTERNET E FUNCIONA NA MAIORIA DOS BROWSERS;

O PRINCIPAL OBJETIVO DA JAVASCRIPT É ADICIONAR INTERATIVIDADE NAS PÁGINAS.

NÃO É NECESSÁRIO COMPILADOR PARA EXECUTAR OS SCRIPTS.

JAVASCRIPT NÃO TEM NADA A VER COM O JAVA, EXCETO ALGUMAS SIMILARIDADES DE SINTAXE;

NÃO É NECESSÁRIO TIPAR AS VARIÁVEIS;

JAVASCRIPT É CASE SENSITIVE;

REPRESENTAÇÃO EM CAMADAS

CLIENT-SIDE

COMPORTAMENTO

APRESENTAÇÃO

ESTRUTURA

PRINCIPAIS UTILIZAÇÕES

EVENTOS - "SÃO UTILIZADOS QUANDO ALGO ACONTECE NA PÁGINA".

- CLIQUE DO MOUSE;
- CARREGAMENTO DA PÁGINA;
- ENVIO DE INFORMAÇÕES.

LER E ESCREVER

É POSSIVEL ESCREVER EM CONTEÚDO DE ELEMENTOS HTML E DA MESMA FORMA LER OS SEUS VALORES.

VALIDAR DADOS

É POSSÍVEL VALIDAR INFORMAÇÕES ANTES MESMO DELAS SEREM ENVIADAS AO SERVIDOR.

SINTAXE E UTILIZAÇÃO DO JAVASCRIPT

SINTAXE - ELA É DEFINIDA PELA TAG HTML:

```
<script> ... </script>
```

UTILIZAÇÃO – EXISTEM 3 FORMAS DE UTILIZAÇÃO DO JAVASCRIPT:

- EM UM EVENTO INLINE;
- DIRETAMENTE NO CABEÇALHO OU CORPO DA PÁGINA;
- EM UM ARQUIVO EXTERNO.

EM UM EVENTO DA TAG HTML (INLINE):

<tag evento="instrução JS"></tag>

EX: `<button onclick="alert('Olá Mundo!!!');">Boas Vindas</button>`

INTERNO NO CABEÇALHO OU CORPO DA PÁGINA

NO CABEÇALHO

```
<html>
<head>
<script type="text/javascript">
        alert('Olá Mundo!!!');
</script>
</head>
<body>
    .......
</body>
</html>
```

NO CORPO

```
<html>
<head>
     ........
</head>
<body>
<script type="text/javascript">
        alert('Olá Mundo!!!');
</script>
</body>
</html>
```

EM UM ARQUIVO JAVASCRIPT “EXTERNO”

ARQUIVO HTML

```
<html>
<head>
</head>
<body>
    <script type="text/javascript"  src="js/script.js">
    </script>
</body>
</html>
```

ARQUIVO JAVASCRIPT

```
alert('Olá Mundo!!!');
```

COMENTÁRIOS NO CÓDIGO JAVASCRIPT

FIAP

EM JAVASCRIPT OS COMENTÁRIOS PODEM SER ADICIONADOS NO CÓDIGO DE DUAS MANEIRAS:

LINHA ÚNICA, REPRESENTADA POR DUAS BARRAS SEGUIDAS DO TEXTO

```
// UM EXEMPLO DE COMENTÁRIO NUMA LINHA ÚNICA
```

BLOCO, REPRESENTADO POR UM TEXTO ENTRE UMA BARRA E UM ASTERISCO

```
/*
 ESTE É UM COMENTÁRIO
 JAVASCRIPT
 EM BLOCO
*/
```

SÃO CARACTERISTICAS DAS VARIÁVEIS DO JAVASCRIPT:

EM JAVASCRIPT AS VARIÁVEIS NÃO SÃO TIPADAS, LOGO BASTA DECLARÁ-LAS UTILIZANDO A PALAVRA RESERVADA “VAR”;

O JAVASCRIPT É CASE SENSITIVE, OU SEJA, ELE FAZ DIFERENCIAÇÃO ENTRE LETRAS MAIÚSCULAS E MINÚSCULAS;

A FORMA DE DECLARAÇÃO DAS VARIÁVEIS UTILIZADA NO MERCADO É O CAMEL CASE, ONDE A PRIMEIRA PALAVRA COMEÇA COM LETRA MINÚSCULA E AS DEMAIS COM LETRA MAIÚSCULA SEM ESPAÇO ENTRE ELAS;

VARIÁVEIS E TIPOS DE DADOS

FIAP

O JAVASCRIPT PODE INTERPRETAR AS VARIÁVEIS COMO OS TIPOS:

INT = NÚMEROS INTEIROS .

EX:
```
var idadeAluno = 18;
```

FLOAT = NÚMEROS FLUTUANTES COM CASAS DECIMAIS.

EX:
```
var valorProduto = 5.35;
```

STRING = VARIÁVEIS DE TEXTO, ELAS DEVEM SER REPRESENTADAS ENTRE ASPAS SIMPLES OU DUPLAS.

EX:
```
var nomeAluno = "Pedro Henrique";
```

BOOLEANOS = QUE RECEBE OS VALORES FALSO OU VERDADEIRO.

EX:
```
var casado = true;
```

ARRAY = CONJUNTO DE ELEMENTOS, DEVE SER REPRESENTADO ENTRE CONCHETES E OS VALORES SEPARADOS POR VIRGULA.

EX:
```
var linguagens = ["Java", "C#", "Python"];
```

O JAVASCRIPT PODE INTERPRETAR AS VARIÁVEIS COMO OS TIPOS:

OBJETOS = É UM CONJUTO DE INFORMAÇÕES SOBRE ALGO EM ESPECÍFICO COMO UM ARRAY, MAS AO INVÉS DE SEREM ACESSADAS POR SUAS POSIÇÕES, PODEMOS DAR NOMES AS POSIÇÕES.

EX:

```
var carro = {
    cor: "azul",
    tipo: "sedan",
    numPortas: 4,
    modelo:"voyage",
    marca: "VW"
};

Console.log(carro.cor);
```

VARIÁVEIS E TIPOS DE DADOS

FIAP

O JAVASCRIPT PODE INTERPRETAR AS VARIÁVEIS COMO OS TIPOS:

OBJETOS = ASSIM COMO OS OBJETOS NO JAVA, ALÉM DE ATRIBUTOS DO OBJETO, PODEMOS TER TAMBÉM AÇÕES, SÃO OS MÉTODOS.

EX:

```
var carro = {
cor: "azul",
tipo: "sedan",
numPortas: 4,
modelo:"voyage",
marca: "VW",
acelerar: function(){alert("estou correndo!")}
};

carro.acelerar();
```

MANIPULANDO TEXTOS E NÚMEROS

OS TEXTOS (STRING) DEVEM SEMPRE ESTAR ENTRE ASPAS DUPLAS OU SIMPLES.

EX:
```
var nomeAluno = "Pedro Henrique";
var nomeAluno = ‘Pedro Henrique’;
```

QUANDO PRECISAMOS USAR ALGUM CARATER RESERVADO EM NOSSO TEXTO USAMOS A BARRA INVERTIDA “\” ANTES DO CARACTERE PARA ELE FAZER PARTE DO TEXTO.

EX:
```
var pedido = ‘Quero um copo d\’água.’;
```

PODEMOS VERIFICAR O TAMANHO DA NOSSA STRING UTILIZANDO O MÉTODO “length”

EX:
```
var txt = 'quero um copo d\'água';
console.log(txt.length);
```

MANIPULANDO TEXTOS E NÚMEROS

OUTRO MÉTODO QUE PODEMOS UTILIZAR É O “indexOf( )”, ELE RETORNA A PRIMEIRA POSIÇÃO NA STRING DE UM TRECHO DO TEXTO.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.indexOf("as"));
```

ELE IRÁ RETORNAR O VALOR “15” QUE É A POSIÇÃO DA PRIMEIRA OCORRÊNCIA.

PARA SABERMOS A ÚLTIMA VEZ QUE O TRECHO APARECE NO TEXTO UTILIZAMOS O “lastIndexOf( )”.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.lastIndexOf("as"));
```

ELE IRÁ RETORNAR O VALOR “22” QUE É A POSIÇÃO DA ÚLTIMA OCORRÊNCIA.

MANIPULANDO TEXTOS E NÚMEROS

NÓS PODEMOS TAMBÉM EXTRAIR PARTE DO TEXTO, PARA ISSO UTILIZAMOS O "slice( )", QUANDO APONTAMOS O INÍCIO E O FIM DO TRECHO QUE QUEREMOS EXTRAIR E ELE NOS RETORNA O TRECHO COMO UMA NOVA STRING.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.slice(0,5));
```

ELE IRÁ RETORNAR A STRING "Estão" QUE É O TRECHO DO TEXTO DE 0 A 5.

OBS. O MÉTODO "substring" FAZ A MESMA FUNÇÃO DO SLICE.

TEMOS TAMBÉM O "substr( )" QUE RECEBE A POSIÇÃO INICIAL E A QUANTIDADE DE CARACTERES QUE QUEREMOS PEGAR.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.substr(6,8));
```

MANIPULANDO TEXTOS E NÚMEROS

É POSSÍVEL TAMBÉM SUBSTITUIRMOS UM TRECHO DO TEXTO O “replace( )”, RECEBE O TRECHO QUE DEVE SER SUBSTITUIDO E O TRECHO QUE ENTRARÁ NO LUGAR DELE.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.replace("provas","avaliações"));
```

ELE IRÁ RETORNAR A FRASE “Estão chegando as avaliações!” .

O MÉTODO “toUpperCase( )” IRÁ CONVERTER TODA A STRING EM LETRAS MAIÚSCULAS.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.toUpperCase());
```

O MÉTODO “toLowerCase( )” IRÁ CONVERTER TODA A STRING EM LETRAS MINÚSCULAS.

EX:
```
var txt = 'Estão chegando as provas!';
console.log(txt.toLowerCase());
```

MANIPULANDO TEXTOS E NÚMEROS

PARA OS NÚMEROS NÓS TAMBÉM PODEMOS DEFINIR ALGUMAS FORMAS DE APRESENTAÇÃO, POR EXEMPLO TEMOS O "toFixed( )", QUE DEFINE O NÚMERO DE CASAS DECIMAIS DO VALOR.

EX:
```
var num = 3.5432;
console.log(num.toFixed(2));
```

ELE IRÁ RETORNAR O VALOR COM 2 CASAS DECIMAIS "3.54" .

SE QUISERMOS DEFINIR UM NÚMERO DE CASAS INDEPENDENTE DO PONTO USAMOS O "toPrecision( )", QUE DEFINE O NÚMERO DE CASAS DO VALOR COMO UM TODO.

EX:
```
var num = 123.5432;
console.log(num.toPrecision(4));
```

ELE IRÁ RETORNAR O VALOR COM 2 CASAS DECIMAIS "123.5" .

MANIPULANDO TEXTOS E NÚMEROS

PODEMOS PRECISAR MUDAR O TIPO DA VARIÁVEL, COMO POR EXEMPLO DE FLOAT PARA INTEGER, NESSE CASO USAMOS O "parseInt( )", O VALOR PERDERÁ AS CASAS DEPOIS DA VIRGULA.

EX:
```
var numFloat = 123.5432;
console.log(parseInt(numFloat));
```

ELE IRÁ RETORNAR O VALOR COMO INTEIRO "123" .

SE A STRING FOR COMPOSTA DE NUMEROS PODEMOS TRANSFORMA-LA TAMBÉM, PARA ISSO PODEMOS USAR O "parseFloat( )", NO CASO DO VALOR TIVER CASAS DEPOIS DO PONTO.

EX:
```
var numString = "123.5432";
console.log(parseFloat(numString));
```

ELE IRÁ RETORNAR O VALOR COMO TIPO FLOAT "123.5432" .

MANIPULANDO TEXTOS E NÚMEROS

CASO QUEIRA CONVERTER UMA VARIÁVEL PARA STRING DEVEMOS USAR O MÉTODO "toString( )".

EX.
```
var numString = 123.5432;
console.log(numString.toString());
```

OBS. O RESULTADO SERÁ IGUAL AO ORIGINAL, SÓ QUE EM FORMATO STRING.

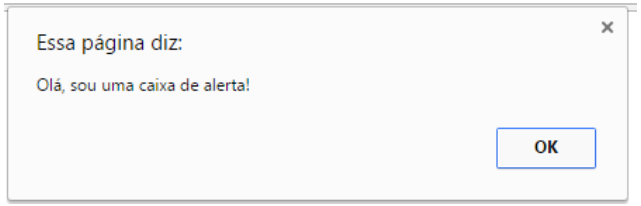
PARA CONFERIR A CONVERSÃO DOS VALORES, PODEMOS UTILIZAR O VERIFICADOR "typeof", QUE RETORNARÁ O TIPO DA VARIÁVEL.

EX.
```
var numFloat = 123.5432;
var numString = numFloat.toString();
var verifica = typeof numString;
console.log(verifica);
```

CAIXAS DE MENSAGEM

Em JS as caixas de mensagens são usadas para apresentar informações para o usuário. São elas:

CAIXA DE ALERTA:

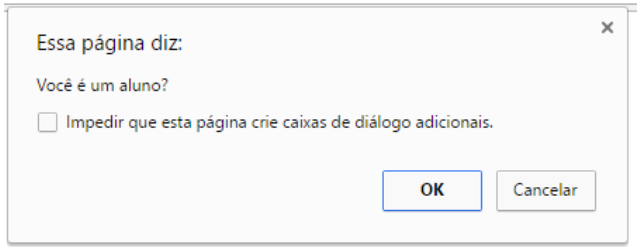


EX:

```
alert("Olá, sou uma caixa de alerta!");
```

CAIXAS DE MENSAGEM

CAIXA DE CONFIRMAÇÃO:



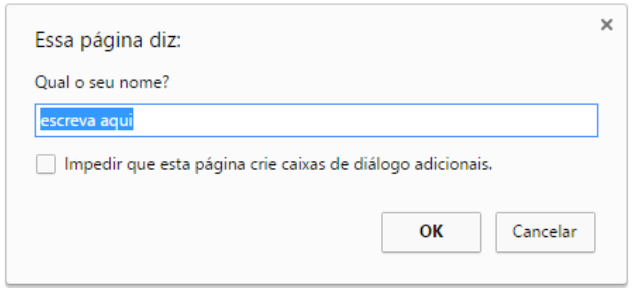
ELA RETORNA UM VALOR BOLEANO, TRUE PARA OK E FALSE PARA CANCELAR

EX:

```
var teste = confirm("Você é um aluno?");
console.log("Resultado da caixa confirm: "+teste);
```

CAIXAS DE MENSAGEM

CAIXA DE TEXTO:



ELA RETORNA A STRING DIGITADA PARA OK E NULL PARA CANCELAR.

EX:

```
var texto = prompt("Qual o seu nome?","escreva aqui");
console.log("O nome dele é "+texto);
```

OBS. SE QUISER, SEPARANDO POR UMA VIRGULA VOCÊ PODE INSERIR UMA MENSAGEM QUE FICA SELECIONADA DENTRO DO CAMPO.

FIAP

FUNÇÃO É UM CONJUNTO DE INSTRUÇÕES QUE EXECUTA UMA TAREFA OU CALCULA UM VALOR QUANDO CHAMADA.

```
function nomeFuncao(arg1, arg2){
        return  arg1 + arg2;
}
alert(nomeFunção(5,4));
```

****

```
function avisar(){
   alert("esta é uma função!!!");
}
avisar();
```

MANIPULAR ELEMENTOS C/ JS

FIAP

- Podemos acessar e manipular as informações e atributos dos nossos elementos HTML pelo **ID**, pelo **name** e pelo **nome da TAG**. Sendo que a opção mais usada é pelo ID (por ser único). Os demais usamos quando queremos percorrer ou coletar várias informações.

- **Usando o ID:** Para acessarmos o ID usamos a sintaxe:

```
document.getElementById("nome").innerHTML = "Luís";
```

document se refere a página html;
getElementById( ) recupera um elemento pelo seu ID;
innerHTML utilizamos para alterar/inserir o conteúdo do elemento.

MANIPULAR ELEMENTOS C/ JS

- EXEMPLO:

exemploPorId.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="ISO-8859-1">
5  <title>Exemplos JavaScript</title>
6  </head>
7  <body>
8      <h1 id="idTitulo">Exemplo por ID</h1>
9
10     <label for="idNovo">Novo Nome:</label>
11     <input type="text" id="idNovo"><br>
12     <button onclick="mudar()">Mudar Nome</button>
13
14 <script type="text/javascript">
15     var novo = document.getElementById("idNovo");
16     function mudar(){
17         document.getElementById("idTitulo").innerHTML = novo.value;
18     }
19 </script>
20 </body>
21 </html>
22
```

MANIPULAR ELEMENTOS C/ JS

- **Usando o Name:** Acessamos utilizando o Name quando precisamos retornar uma coleção de elementos com um mesmo name :

```
var x = document.getElementsByName("nome")[0].value;
```

document se refere a página html;
getElementsByName("name")[0] recupera um elemento com o name especificado pela sua posição, começando por "0";
value retorna o valor "conteúdo" do elemento.

MANIPULAR ELEMENTOS C/ JS

- EXEMPLO:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Exemplos JavaScript</title>
</head>
<body>
<h1 id="idTitulo">Exemplo por NAME</h1>

Primeiro:<input type="text" name="corredor"><br>
Segundo:<input type="text" name="corredor"><br>
Terceiro:<input type="text" name="corredor"><br>
<br>
Novo Classificado: <input type="text" id="novo"><br>
Posição:<input type="number"  id="posicao" min="1" max="3">

<button onclick="inserir()">Inserir</button>

<script>
function inserir(){
    var numero = document.getElementById("posicao").value - 1;
    var novo = document.getElementById("novo").value;
    document.getElementsByName("corredor")[numero].value = novo;
}
</script>
</body>
</html>
```

MANIPULAR ELEMENTOS C/ JS

- **Usando o TagName:** Assim como com o name, podemos acessar elementos pelo nome da Tag se precisarmos retornar uma coleção de elementos dessa tag :

```
var x = document.getElementsByTagName("li")[0].value;
```

document se refere a página html;
getElementsByTagName("tag")[0] recupera um elemento pelo nome da tag especificado pela sua posição, começando por "0";
value retorna o valor "conteúdo" do elemento.

MANIPULAR ELEMENTOS C/ JS

- EXEMPLO:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Exemplos JavaScript</title>
</head>
<body>
<h1 id="idTitulo">Exemplo por TAGNAME</h1>

<ul>
    <li>Janeiro</li>
    <li>Fevereiro</li>
    <li>Março</li>
    <li>Abril</li>
    <li>Maio</li>
</ul>

Escolha um Mês: <input type="number" id="numero" min="1" max="5"><br>
<button onclick="mostrar()">Mostrar</button>
<h2 id="resultado"></h2>
<script>
function mostrar(){
    var num = document.getElementById("numero").value - 1;
    var mes = document.getElementsByTagName("LI")[num].innerHTML;
    document.getElementById("resultado").innerHTML = mes;
}
</script>
</body>
</html>
```

ATIVIDADE

UTILIZANDO O PROJETO “FORMATARFONTE” QUE SERÁ DISPONIBILIZADO, CRIE O PROJETO ABAIXO COM OS CÓDIGOS QUE VIRÃO A SEGUIR.

ATIVIDADE

FIAP

VAMOS COMEÇAR MONTANDO O ARQUIVO HTML.

```
 9 <div id="idPrincipal">
10 <img src="img/html-logo.png" alt="Logo Site">
11 <h1>Formate seu texto</h1>
12 <p id="idParagrafo">Teste - Formatação de texto</p>
13 <div id="idCaixa">
14 <h3>Use os recursos para formatar o texto acima</h3>
15 <label for="idTexto">Seu texto</label>
16 <input type="text" id="idTexto" maxlength="30"><br>
17 <label for="idFonte">Fonte</label>
18 <select id="idFonte">
19     <option>Arial</option>
20     <option>Broadway</option>
21     <option>Brush Script Std</option>
22     <option>Calibri</option>
23     <option>Cooper</option>
24     <option>Forte</option>
25     <option>Impact</option>
26     <option>Monotype Corsiva</option>
27 </select><br>
28 <label for="idTamanho">Tamanho</label>
29 <input type="number" id="idTamanho" min="8" max="40" step="4" value="16"><br>
30 <label for="idCor">Cor do Texto</label>
31 <input type="color" id="idCor"><br>
32 <label for="idCorBg">Cor do Fundo</label>
33 <input type="color" id="idCorBg" value="#ffffff"><br>
34 <button id="idTestar">Testar</button>
35 </div>
36 </div>
```

ATIVIDADE

AGORA O CÓDIGO JAVASCRIPT NO NOSSO ARQUIVO JS.

```
5  document.getElementById("idTestar").onclick = function(){
6      var texto = document.getElementById("idTexto");
7      var paragrafo = document.getElementById("idParagrafo");
8      var fonte = document.getElementById("idFonte");
9      var tamanho = document.getElementById("idTamanho");
10     var cor = document.getElementById("idCor");
11     var corBg = document.getElementById("idCorBg");
12
13     if(texto.value != ""){
14         paragrafo.innerHTML = texto.value;
15     }
16     paragrafo.style.fontFamily = fonte.value;
17     paragrafo.style.fontSize = tamanho.value + "px";
18     paragrafo.style.color = cor.value;
19     paragrafo.style.backgroundColor = corBg.value;
20 }
21
```

AGORA É SÓ EXECUTAR NO NAVEGADOR!

EXERCÍCIOS

1.Faça um programa que leia dois números, calcule e imprima a soma desses dois números.

2. Faça um programa que receba dois números, calcule e imprima a divisão do primeiro número pelo segundo.

3. Faça um programa que leia um número e informe a metade e o dobro desse número.

4. Escrever um programa que permita receber o nome e a idade de uma pessoa e em seguida, informar o nome digitado e a idade da pessoa daqui a 30 anos.

EXERCÍCIOS

5. Faça um programa que leia três notas de um aluno, calcule e exiba a média aritmética entre essas notas.

6. Faça um programa que receba dois números inteiros, calcule e retorne em parágrafos:

- soma dos dois números;
- subtração do primeiro pelo segundo;
- subtração do segundo pelo primeiro;
- Multiplicação dos dois números;
- divisão do primeiro pelo segundo;

EVENTOS

FIAP

EVENTOS SÃO AÇÕES QUE ACONTECEM A ELEMENTOS HTML. QUANDO USAMOS JAVASCRIPT EM NOSSAS PÁGINAS PODEMOS DETERMINAR O QUE ACONTECERÁ NESSES EVENTOS.

UTILIZAMOS ESTES EVENTOS COMO ATRIBUTOS QUE CONTÉM CÓDIGO JAVASCRIPT PARA DISPARAR AS AÇÕES.

<tag evento="instrução JS"></tag>

EX: `<button onclick="alert('Olá Mundo!!!');">Boas Vindas</button>`

EVENTOS

FIAP

ATRAVÉS DOS EVENTOS PODEMOS DISPARAR AÇÕES NO PRÓPRIO ELEMENTO OU EM QUALQUER OUTRO DA PÁGINA.

INTERAGINDO COM OUTRO ELEMENTO

EX:

```
<button onclick="document.getElementById('texto').innerHTML='Boa tarde!'">
Saudação</button>

<p id="texto"></p>
```

INTERAGINDO COM O PRÓPRIO ELEMENTO

EX:

```
<button onclick="this.innerHTML = 'Já clicou!'">Clique aqui</button>
```

OBS. PARA INTERAGIRMOS COM O PRÓPRIO ELEMENTO PODEMOS UTILIZAR O TERMO “THIS”.

FIAP

OS PRINCIPAIS EVENTOS SÃO:

- ONCHANGE = DISPARA O EVENTO QUANDO O ELEMENTO É ALTERADO.

INSTRUÇÃO DENTRO DO EVENTO

```
Escala:<input type="range"
onchange="document.getElementById('valor').innerHTML = this.value">
<span id="valor"></span>
```

INSTRUÇÃO DENTRO DA FUNÇÃO

```
Escala:<input type="range" id="idBarra" onchange="mudar();">
<span id="valor"></span>
<script>
function mudar(){
var barra = document.getElementById("idBarra").value;
document.getElementById("valor").innerHTML = barra;}
</script>
```

EVENTOS

FIAP

OS PRINCIPAIS EVENTOS SÃO:

- ONCLICK = DISPARA O EVENTO QUANDO O USUÁRIO CLICA NO ELEMENTO.

INSTRUÇÃO DENTRO DO EVENTO

```
<h1 id="idTexto"></h1>
<button
onclick="document.getElementById('idTexto').innerHTML = 'Olá!!!'">
Clique aqui</button>
```

INSTRUÇÃO DENTRO DA FUNÇÃO

```
<h1 id="idTexto"></h1>
<button onclick="cumprimentar();">Clique aqui</button>

<script>
function cumprimentar(){
document.getElementById("idTexto").innerHTML = "Olá";}
</script>
```

EVENTOS

OS PRINCIPAIS EVENTOS SÃO:

- ONMOUSEOVER = QUANDO O USUÁRIO MOVE O MOUSE SOBRE O ELEMENTO.

INSTRUÇÃO DENTRO DO EVENTO

```
<h1 onmouseover="this.style.color = 'red';">FIAP</h1>
```

OBS. *DIFERENTE DO HOVER NO CSS ELE SÓ ATIVA A MUDANÇA, PARA VOLTAR DEVEMOS USAR O EVENTO QUE VEM A SEGUIR*

INSTRUÇÃO DENTRO DA FUNÇÃO

```
<h1 onmouseover="mudar(this);" onmouseout="voltar(this);">FIAP</h1>
<script>
function mudar(x){ x.style.color = "red" }
</script>
```

EVENTOS

OS PRINCIPAIS EVENTOS SÃO:

- ONMOUSEOUT = QUANDO O USUÁRIO RETIRA PONTEIRO DO MOUSE QUE ESTAVA SOBRE O ELEMENTO.

INSTRUÇÃO DENTRO DO EVENTO

```
<h1 onmouseout="this.style.color = 'black';">FIAP</h1>
```

OBS. *No caso anterior ele seria usado em conjunto com o evento "onmouseover"*

```
<h1 onmouseover="this.style.color = 'red';"
onmouseout="this.style.color = 'black';">FIAP</h1>
```

INSTRUÇÃO DENTRO DA FUNÇÃO

```
<h1 onmouseover="mudar(this);" onmouseout="voltar(this);">FIAP</h1>
<script>
function mudar(x){ x.style.color = "red" }
function voltar(x){x.style.color = "black" }
</script>
```

EVENTOS

OS PRINCIPAIS EVENTOS SÃO:

- ONFOCUS = QUANDO O USUÁRIO SELECIONA UM CAMPO PARA INSERIR DADOS OU SELECIONAR UMA OPÇÃO.

INSTRUÇÃO DENTRO DO EVENTO

```
<input onfocus="this.style.backgroundColor = 'yellow';">
```

OBS. *DIFERENTE DO FOCUS NO CSS ELE SÓ ATIVA A MUDANÇA, PARA VOLTAR DEVEMOS USAR O EVENTO QUE VEM A SEGUIR*

INSTRUÇÃO DENTRO DA FUNÇÃO

```
<input onfocus="destacar(this);">
<script>
function destacar(x){
x.style.backgroundColor = 'yellow';
}
</script>
```

EVENTOS

OS PRINCIPAIS EVENTOS SÃO:

- ONBLUR = QUANDO O USUÁRIO SAI UM CAMPO APÓS TER SIDO VISITADO.

INSTRUÇÃO DENTRO DO EVENTO

```
<input onblur="this.style.backgroundColor = 'white';">
```

INSTRUÇÃO DENTRO DA FUNÇÃO

```
<input onblur="semDestaque(this);">
<script>
function semDestaque(x){
x.style.backgroundColor = 'white';
}
</script>
```

OBS. COMO NOS EXEMPLOS ANTERIORES ELES PODEM SER USADOS EM CONJUNTO.

OS PRINCIPAIS EVENTOS SÃO:

- ONSUBMIT = É UTILIZADO EM FORMULÁRIOS PARA AÇÕES QUANDO O FORMULÁRIO É SUBMETIDO.

```
<form action="" id="cadastro" method="get" onsubmit="agradecer();">
Nome:<input type="text" id="idNome" name="nome">
Idade:<input type="number" id="idIdade" name="idade">
<input type="submit" value="Gravar">
</form>

<script>
function agradecer(){
alert("Muito obrigado pela sua participação!");
}
</script>
```

CONDIÇÕES

FIAP

ASSIM COMO JAVA UTILIZAMOS OS BLOCOS DE CONDIÇÕES "IF, ELSE E ELSE IF" PARA IMPORMOS CONDIÇÕES PARA DEFIRNIR AS AÇÕES.

```
11 Nome:<input type="text" id="idNome" name="nome">
12 Idade:<input type="number" id="idIdade" name="idade">
13 <button onclick="responder()">Enviar</button>
14 <p id="resposta"></p>
15 <script>
16 function responder(){
17     var nome = document.getElementById("idNome").value;
18     var idade = document.getElementById("idIdade").value;
19     var resposta = document.getElementById("resposta");
20     if(idade < 18){
21         resposta.innerHTML = "Olá "+nome+" você não é obrigado a votar.";
22     }else if(idade >= 18 && idade < 65){
23         resposta.innerHTML = "Olá "+nome+" você é obrigado a votar.";
24     }else{
25         resposta.innerHTML = "Olá "+nome+" você já não é obrigado a votar.";
26     }
27 }
28 </script>
```

FIAP

PODEMOS TAMBÉM UTILIZAR A ESTRUTURA DE REPETIÇÃO "FOR" PARA CRIAR REPETIÇÕES E PERCORRER ARRAYS.

```
<h1 >Lista de frutas</h1>
<ul id="idLista"></ul>
<button onclick="listar();">Clique aqui!</button>
<script>
function listar(){
    var frutas = ["banana", "laranja","maça"];
    var lista = document.getElementById("idLista");
    var itens =  "";
    for(i = 0; i< frutas.length; ++i){
        itens += "<li>"+frutas[i]+"</li>";
    }
    lista.innerHTML = itens;
}
</script>
```

VALIDAÇÃO

FIAP

UMA DAS MAIORES FUNCIONALIDADES DO JAVASCRIPT SÃO AS VALIDAÇÕES SOBRE TIPO E FORMATO DE DADOS, QUE PODEMOS FAZER DIRETAMENTE DO NAVEGADOR , FICANDO ASSIM O PROCESSO MUITO MAIS RÁPIDO.

PODEMOS FAZER VALIDAÇÕES SIMPLESMENTE ATRAVÉS DE CONDIÇÕES OU UTILIZAR EXPRESSÕES REGULARES.

COMO O JAVASCRIPT RODA DIRETAMENTE NO NAVEGADOR, NÃO DEVEMOS VALIDAR DADOS DE USUÁRIO OU DO SISTEMA, MAS APENAS SE OS DADOS FORAM INSERIDOS E SE FORAM INSERIDOS DE FORMA CORRETA.

VALIDAÇÃO

VAMOS SUPOR QUE O USUÁRIO SÓ POSSA SOLICITAR EMPRÉSTIMOS DE NO MÁXIMO R$ 9.000,00, COMO FICARIA A CONDIÇÃO?

```
if (valor < 0 || valor > 9000 || isNaN(valor)){
alert("Este valor está muito alto!")

}else{
alert("Vamos mandar seu pedido para análise!")
}
```

PERCEBA QUE O VALOR NÃO PODE SER NEGATIVO, NÃO PODE SER MAIOR QUE 9000 E TEM QUE SER UM NÚMERO.

isNaN( ) = É UMA FUNÇÃO QUE VERIFICA SE SEU CONTEÚDO NÃO É UM NÚMERO.

VALIDAÇÃO – EXEMPLOS COM REGEX

FIAP

VALIDANDO PREENCHIMENTO DE NOME:

TAG HTML COM EVENTO ONBLUR.

```
<label id="Nome" for="nome">Nome</label>
<input type="text" name="nome" id="idNome" onblur="validarNome(this);">
```

SCRIPT COM REGEXP PARA VALIDAR NOME.

```
function validarNome(tag){
        var nome = tag.value;
        var validacao = /^[A-Za-záàâãéèêíïóôõöüúçñÁÀÂÃÉÈÍÏÓÔÕÖÜÚÇÑ' ]+$/;
        if (!validacao.test(nome)){
                alert("Por favor digite somente o seu nome!");
        }else{
                alert("Obrigado por preencher nosso cadastro!!");
        }       }
```

test( ) = UTILIZADO PARA VERIFICAR SE UM VALOR É VALIDO QUANDO USAMOS REGEX.

EX: `regex.test(variável);` RETORNA TRUE OU FALSE!

VALIDAÇÃO

VALIDANDO PREENCHIMENTO DE NÚMERO (RM):

TAG HTML COM EVENTO ONBLUR.

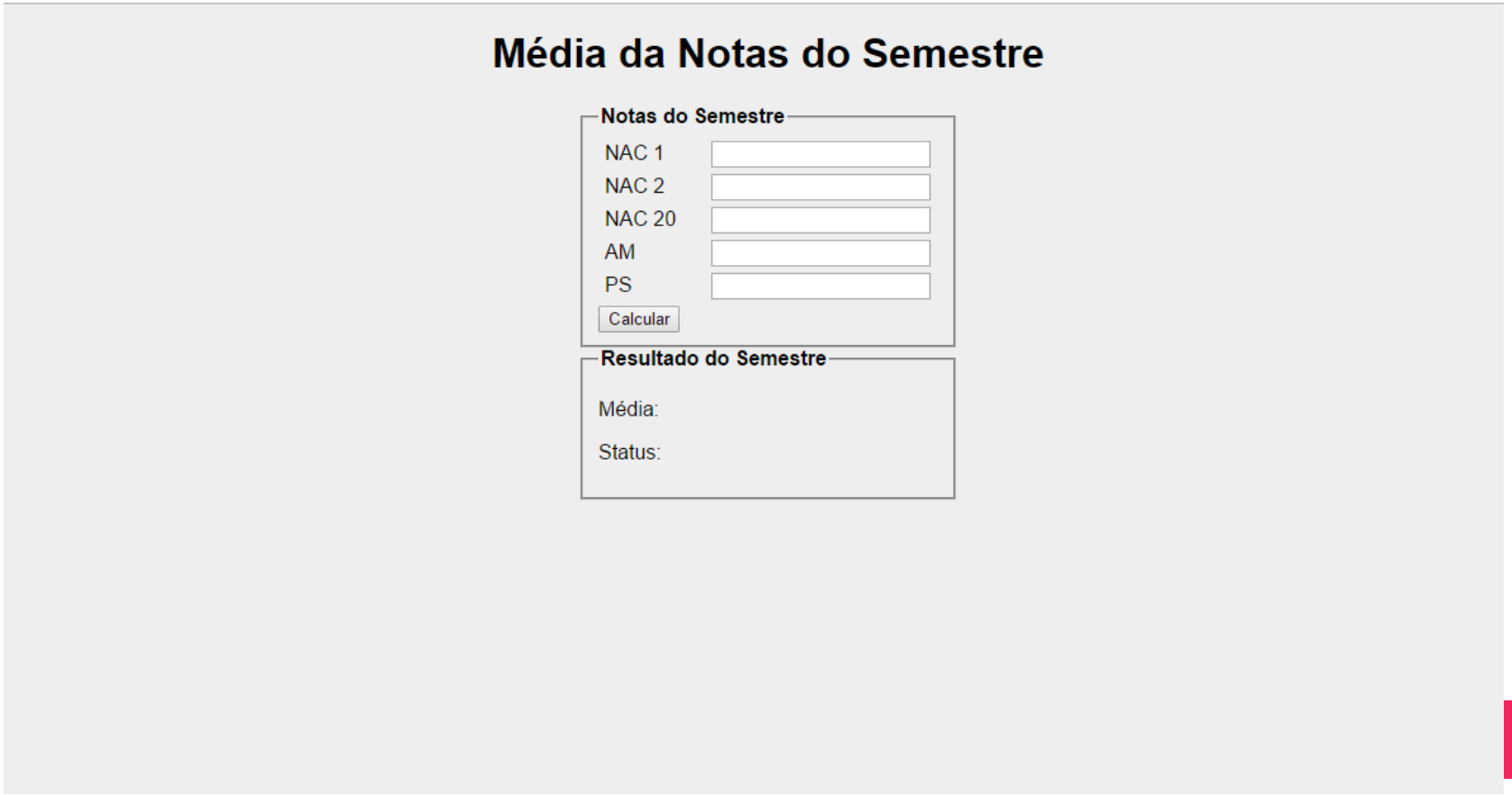
```
<label id="Rm" for="rm">RM</label>
<input type="text" name="rm" id="idRm" onblur="validarRm(this,'erroRm')">
```

SCRIPT COM REGEXP PARA VALIDAR O NÚMERO.

```
function validarRm(tag,idErro){
        var rm = parseInt(tag.value);
        var validacao = /^\d{5}$/;
        if (!validacao.test(rm)){
                alert("Por favor digite seu RM com 5 digitos!");
        }else{
                alert("RM digitado com sucesso!");
        }
}
```

ATIVIDADE 1

UTILIZANDO O PROJETO "CALCULANOTA" QUE SERÁ DISPONIBILIZADO, CRIE O PROJETO ABAIXO COM OS CÓDIGOS QUE VIRÃO A SEGUIR.



ATIVIDADE 1

VAMOS COMEÇAR MONTANDO O ARQUIVO HTML.

```
 9 <h1>Média da Notas do Semestre</h1>
10     <div id="form">
11 <!-- ============================== CAMPO NAC 1 ============================== -->
12         <fieldset>
13     <legend>Notas do Semestre</legend>
14         <div>
15             <label for="nac1">NAC 1</label>
16             <input type="text" id="nac1" onblur="validarNota(this,'erroNac1')">
17             <span class="erro" id="erroNac1"></span>
18         </div>
19 <!-- ============================== CAMPO NAC 2 ============================== -->
20         <div>
21             <label for="nac2">NAC 2</label>
22             <input type="text" id="nac2" onblur="validarNota(this,'erroNac2')">
23             <span class="erro" id="erroNac2"></span>
24         </div>
25 <!-- ============================== CAMPO NAC 20 ============================== -->
26         <div>
27             <label for="nac20">NAC 20</label>
28             <input type="text" id="nac20" onblur="validarNota(this,'erroNac20')">
29             <span class="erro" id="erroNac20"></span>
30         </div>
```

ATIVIDADE 1

ARQUIVO HTML PARTE 2.

```
<!-- ============================== CAMPO AM ================================== -->
        <div>
            <label for="am">AM</label>
            <input type="text" id="am" onblur="validarNota(this,'erroAm')">
            <span class="erro" id="erroAm"></span>
        </div>
<!-- ============================== CAMPO PS ================================== -->
        <div>
            <label for="ps">PS</label>
            <input type="text" id="ps" onblur="validarNota(this,'erroPs')">
            <span class="erro" id="erroPs"></span>
        </div>
        <input type="button" value="Calcular" onclick="calcularMedia()">
        <span class="erro" id="erroCalc"></span>
        </fieldset>
<!-- ============================== CAMPO RESULTADOS ========================== -->
        <fieldset id="idFielRes">
    <legend>Resultado do Semestre</legend>
        <p>Média: <span id="media"></span></p>
        <p>Status: <span id="resultado"></span></p>
        <img id="imagem" alt="" src="">
        </fieldset>
    </div>
```

ATIVIDADE 1

AGORA O CÓDIGO JAVASCRIPT NO NOSSO ARQUIVO JS.

```
function validarNota(tag,erro){
    var nota = parseFloat(tag.value);
    var spanErro = document.getElementById(erro);
    if (nota < 0 || nota > 10 || isNaN(nota)){
        tag.style.border = "2px solid red";
        spanErro.innerHTML = "Nota inválida";

    }else{
        tag.style.border = "1px solid silver";
        spanErro.innerHTML = "";
        document.getElementById("erroCalc").innerHTML = "";
    }
}

function calcularMedia(){
    //Recupera os valores dos campos
    var notaNac1 = parseFloat(document.getElementById("nac1").value);
    var notaNac2 = parseFloat(document.getElementById("nac2").value);
    var notaNac20 = parseFloat(document.getElementById("nac20").value);
    var notaAm = parseFloat(document.getElementById("am").value);
    var notaPs = parseFloat(document.getElementById("ps").value);

    var validado = true;

    var testErro = document.getElementsByTagName("span");
```

ATIVIDADE 1

ARQUIVO JS PARTE 2.

```
33      //Verifica se tem algum campo inválido
34      for(i = 0; i < 4; i++){
35
36          if(testErro[i].innerHTML != ""){
37              validado = false
38          }
39      }
40
41      //Valida se algum campos está inválido
42      if (validado == false) {
43          document.getElementById("erroCalc").innerHTML = "Existem campos Inválidos!";
44          return; //Sai da função
45      }else{
46          document.getElementById("erroCalc").innerHTML = "";
47      }
48
49      //Verifica qual a maior NAC10
50      var nacMaior;
51      if (notaNac1 >= notaNac2){
52          nacMaior = notaNac1;
53      }else{
54          nacMaior = notaNac2;
55      }
56
57      //Calcula a média das NACs
58      var mediaNAC = (nacMaior + notaNac20*2)/3;
```

ATIVIDADE 1

ARQUIVO JS PARTE 3.

```
60      //Calcula a nota final
61      var mediaFinal = mediaNAC*0.2 + notaAm*0.3 + notaPs*0.5;
62
63      //Exibe o texto na tela
64      document.getElementById("media").innerHTML = mediaFinal.toFixed(1);
65      if (mediaFinal >=6){
66          document.getElementById("resultado").innerHTML = "Aprovado! =)";
67          document.getElementById("idFielRes").style.backgroundColor = "#3f3";
68          document.getElementById("imagem").src = "img/aprovado.png";
69          document.getElementById("imagem").style.height = "60px";
70          document.getElementById("imagem").style.width = "60px";
71      }else if(mediaFinal >=4){
72          document.getElementById("resultado").innerHTML = "Exame! =/";
73          document.getElementById("imagem").src = "img/exame.png";
74          document.getElementById("imagem").style.height = "60px";
75          document.getElementById("imagem").style.width = "60px";
76          document.getElementById("idFielRes").style.backgroundColor = "#ff3";
77      }else{
78          document.getElementById("resultado").innerHTML = "Retido! =(";
79          document.getElementById("imagem").src = "img/reprovado.png";
80          document.getElementById("imagem").style.height = "60px";
81          document.getElementById("imagem").style.width = "60px";
82          document.getElementById("idFielRes").style.backgroundColor = "#f33";
83      }
84  }
```

ATIVIDADE 2

UTILIZANDO O PROJETO "VALIDACAOALUNO" QUE SERÁ DISPONIBILIZADO, CRIE O PROJETO ABAIXO COM OS CÓDIGOS QUE VIRÃO A SEGUIR.

ATIVIDADE 2

VAMOS COMEÇAR MONTANDO O ARQUIVO HTML.

```
<h2>CADASTRO DO ALUNO</h2>

    <form method="get" action="" name="form">
    <fieldset>
    <legend>Dados do Aluno</legend>
    <div>
            <label id="Nome" for="nome">Nome</label>
            <input type="text" name="nome" id="idNome" onblur="validarNome(this,'erroNome');">
            <span class="erro" id="erroNome"></span>
    </div>
    <div>
            <label id="Rm" for="rm">RM</label>
            <input type="text" name="rm" id="idRm" onblur="validarRm(this,'erroRm')">
            <span class="erro" id="erroRm"></span>
    </div>
    <div>
            <label for="CPF">CPF</label>
            <input type="text" name="cpf" id="idCpf" onblur="validarCpf(this,'erroCpf')">
            <span class="erro" id="erroCpf"></span>
    </div>
        <input type="button" value="Cadastrar">
        </fieldset>
    </form>
```

ATIVIDADE 2

AGORA O CÓDIGO JAVASCRIPT NO NOSSO ARQUIVO JS.

```
function validarNome(tag,idErro){
    var nome = tag.value;
    var validacao = /^[A-Za-záàâãéèêíïóôõöüúçñÁÀÂÃÉÈÍÏÓÔÕÖÜÚÇÑ' ]+$/;
    var spanErro = document.getElementById(idErro);
    if (!validacao.test(nome)){
        tag.style.border = "2px solid red";
        spanErro.innerHTML = "Nome Inválido!";
    }else{
        tag.style.border = "1px solid silver";
        spanErro.innerHTML = "";
        }
}

function validarRm(tag,idErro){
    var rm = parseInt(tag.value);
    var validacao = /^\d{5}$/;
    var spanErro = document.getElementById(idErro);
    if (!validacao.test(rm)){
        tag.style.border = "2px solid red";
        spanErro.innerHTML = "RM Inválido!";
    }else{
        tag.style.border = "1px solid silver";
        spanErro.innerHTML = "";
        }
}
```

ATIVIDADE 2

ARQUIVO JS PARTE 2.

```
31 function validarCpf(tag,idErro){
32     var cpf = tag.value;
33     var validacao =/^([\d]{2,3})\.?([\d]{3})\.?([\d]{3})\-?([\d]{2})$/;
34     var spanErro = document.getElementById(idErro);
35     if (!validacao.test(cpf)){
36         tag.style.border = "2px solid red";
37         spanErro.innerHTML = "CPF Inválido!";
38     }else{
39         tag.style.border = "1px solid silver";
40         spanErro.innerHTML = "";
41         var cpfFormat = tag.value.replace(validacao, "$1.$2.$3-$4");
42         idCpf.value = cpfFormat;
43 
44         }
45 }
```

EXERCÍCIO

- Para nosso projeto a ideia é criar uma página para os usuários avaliarem seu indice de massa corporal (IMC).
- Os Requisitos São:
- **Informações do usuário:** Nome, Telefone, email, idade e sexo;
- **Dados do Cálculo:** peso e altura;
- **Complemento:** Botão Calculo e DIV com os dados dos campos e o resultado;

- Além do Resultado do Calculo, ele deverá indicar em qual a faixa das abaixo ele se encontra.
- **Abaixo do peso ideal:** menor que 18.5
- **Peso ideal:** entre 18.5 e 24,9
- **Sobrepeso:** entre 25.0 e 29.9
- **Obesidade I:** entre 30.0 e 34.9
- **Obesidade II:** entre 35.0 e 39.9
- **Obesidade III:** maior que 40.0
-
- **Desejável apresentar imagens correspondentes aos diferentes resultados**

ALGUNS EXEMPLOS

\d{5}-\d{3} = O PADRÃO PARA UMA MÁSCARA DE CEP;

[A-Z]{3}-\d{4} = O PADRÃO DE PLACAS DE AUTOMÓVEIS NO BRASIL;

[012]\d:[0-5]\d = PARA FORMATO DE HORAS. EX: “05:45”.

EXPRESSÕES REGULARES

Especificadores

Especificam o conjunto de caracteres a casar em uma posição.

metacaractere	conhecido como	significado
`.`	curinga	qualquer caractere, exceto a quebra de linha `\n` (ver *flag_dotall*)
`[...]`	conjunto	qualquer caractere incluido no conjunto
`[^...]`	conjunto negado	qualquer caractere não incluido no conjunto
`\d`	dígito	o mesmo que `[0-9]`
`\D`	não-digíto	o mesmo que `[^0-9]`
`\s`	branco	espaço, quebra de linha, tabs etc.; o mesmo que `[ \t\n\r\f\v]`
`\S`	não-branco	o mesmo que `[^ \t\n\r\f\v]`
`\w`	alfanumérico	o mesmo que `[a-zA-Z0-9_]` (mas pode incluir caracteres Unicode; ver *flag_unicode*)
`\W`	não-alfanumérico	o complemento de `\w`
`\`	escape	anula o significado especial do metacaractere seguinte; por exemplo, `\.` representa apenas um ponto, e não o curinga

Quantificadores

Definem o número permitido repetições da expressão regular precedente.

metacaractere	significado
`{n}`	exatamente *n* ocorrências
`{n,m}`	no mínimo *n* ocorrências e no máximo *m*
`{n,}`	no mínimo *n* ocorrências
`{,n}`	no máximo *n* ocorrências
`?`	0 ou 1 ocorrência; o mesmo que `{,1}`
`+`	1 ou mais ocorrência; o mesmo que `{1,}`
`*`	0 ou mais ocorrência

EXPRESSÕES REGULARES

FIAP

Âncoras

Estabelecem posições de referência para o casamento do restante da regex. Note que estes metacaracteres não casam com caracteres no texto, mas sim com posições antes, depois ou entre os caracteres.

metacaractere	significado
`^`	início do texto, ou de uma linha com o flag `re.MULTILINE`
`\A`	início do texto
`$`	fim do texto, ou de uma linha com o flag `re.MULTILINE`; não captura o `\n` no fim do texto ou da linha
`\Z`	fim do texto
`\b`	posição de borda, logo antes do início de uma palavra, ou logo depois do seu término; o mesmo que a posição entre `\W` e `\w` ou vice-versa
`\B`	posição de não-borda

Agrupamento

Definem ou grupos ou alternativas.

metacaractere	significado
`(...)`	define um *grupo*, para efeito de aplicação de quantificador, alternativa ou de posterir extração ou re-uso

FIAP