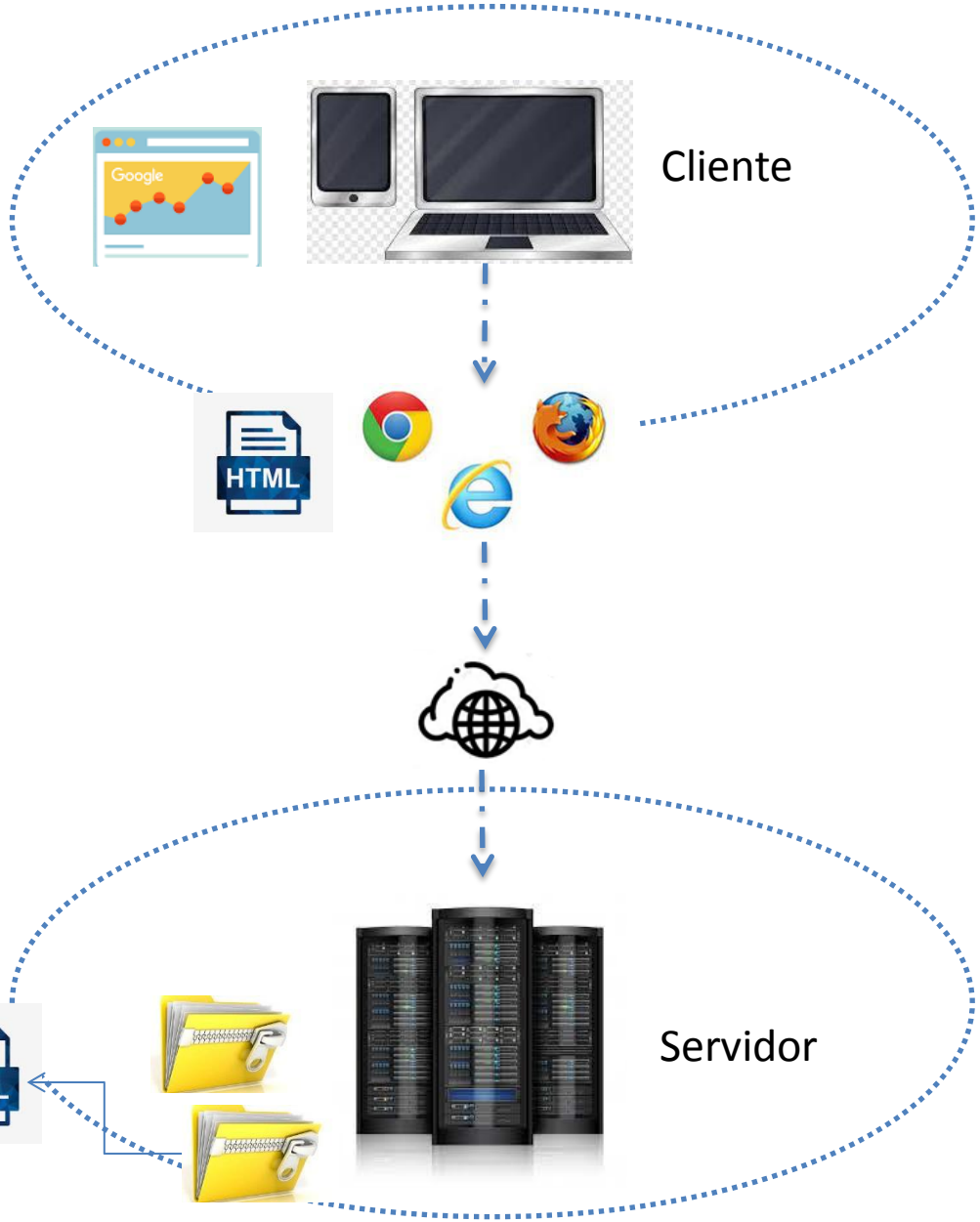
Programação Mobile – Java Script

Profa. Ingrid Thais Ribeiro

Centro Paula Souza
SÃO PAULO
GOVERNO DO ESTADO

Relembrando alguns conceitos...
Cliente x Servidor (Client x Server)

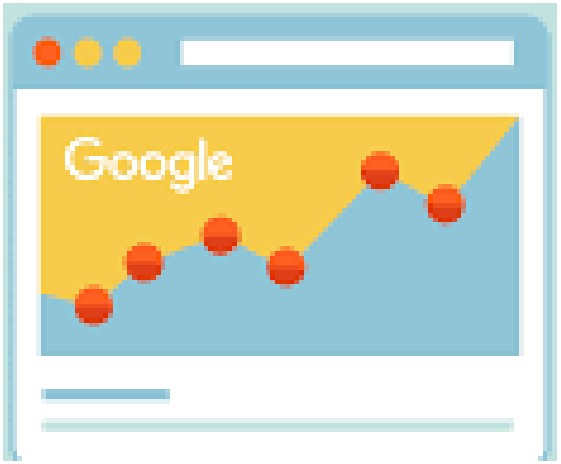


Tecnologia Client-side

Relembrando conceito de cliente x servidor

- Lado cliente, somos nós!
- Por meio dos navegadores, acessamos a internet para solicitar um documento ao servidor;
- O Servidor faz uma cópia desse documento ao navegador que carrega para nós uma página;
- Note que para esse processo acontecer existem tecnologias envolvidas, tanto no lado cliente como no lado servidor;
- O JS inicialmente fora desenvolvido para o lado cliente, mas é uma tecnologia que também pode ser utilizada no lado servidor;

Relembrando alguns conceitos...
Tecnologias envolvidas no lado cliente



Conteúdo
=
HTML

Desing
=
CSS

Interações
=
JavaScript

JavaScript – O que é?

- Também chamada de JS, é a linguagem de criação de scripts para a Web;

- É utilizado por bilhões de páginas para:

  - Adicionar funcionalidades;
  - Verificar formulários;
  - Comunicar com servidores;
  - E muitos mais.

JavaScript – História

- Originalmente criada na Netscape por Brendan Eich em 1994;

- Disputa: Netscape X Microsoft

- Microsoft -> Visual Basic;
- Visual Basic -> VB Script;

- Java da Sun surgia como potencial;
- Java para programadores não profissionais: Javascript!

JAVA ≠ JAVASCRIPT

- Java e JavaScript são “coisas” completamente distintas e desconexas;
- Compartilham apenas um passado de “disputa territorial” contra a Microsoft;

JavaScript – Diferenças chaves em relação ao Java

JAVA	JavaScript
➢ Java é uma linguagem de programação;	➢ JavaScript é uma linguagem de script;
➢ Aplicativos Java são executados pela máquina virtual Java;	➢ Scripts JavaScript são executados pelos browsers;
➢ Java é compilado;	➢ JavaScript é texto puro;

➢ Cada tecnologia requer um plug-in diferente

JavaScript – Características

- JavaScript não permite a criação de applets nem de aplicativos;
- JavaScript reside dentro de documentos HTML e pode prover diferentes níveis de interatividades não suportados pelo HTML sozinho;

JavaScript – Fundação Mozilla

- Atualmente, o maior mantedor da linguagem é a Fundação Mozilla;

- Encontramos ótimos materiais e tutoriais sobre JavaScript na W3School, mas também encontramos referência completa do JavaScript no site do Mozilla:

JavaScript – Funcionalidades

- Com o tempo, muitas funcionalidades foram criadas em forma de Script para os browser e foram “incorporadas” ao JavaScript:

- JavaScript hoje é um conjunto de funcionalidades e, até mesmo, diferentes padrões.

JavaScript – Funcionalidades

- A Linguagem Núcleo:

  - ECMAScript (Versão 7, de Junho de 2016);
  - Padrão mantido por ECMA InternationalAssociação Industrial de padronização de tecnologias da Informação e Comunicação;

- DOM:

  - Document Object Model;
  - Define a Interface da Linguagem com o Browser;
  - Padrão mantido por W3C;

- A linguagem JS equivale a programação orientada a objetos.
- Todos os elementos de uma página da Web são tratados como objetos.
- Estes objetos são agrupados de acordo com seu tipo ou finalidade.
- São criados automaticamente objetos que permitem que o usuário possa criar novos objetos de acordo com sua necessidade.

- Ao ser carregada uma página da Web, é criado um determinado número de objetos JS, com propriedades e valores próprios que são ajustados pelo conteúdo da própria página.

- Todos os objetos seguem uma hierarquia que reflete toda a estrutura de uma página HMTL.

- A linguagem JavaScript pode ser utilizada para a criação de scripts tanto do lado cliente como do lado servidor.

- Seguindo a hierarquia de objetos da linguagem JavaScript, são criados os seguintes objetos ao ser carregada uma página:

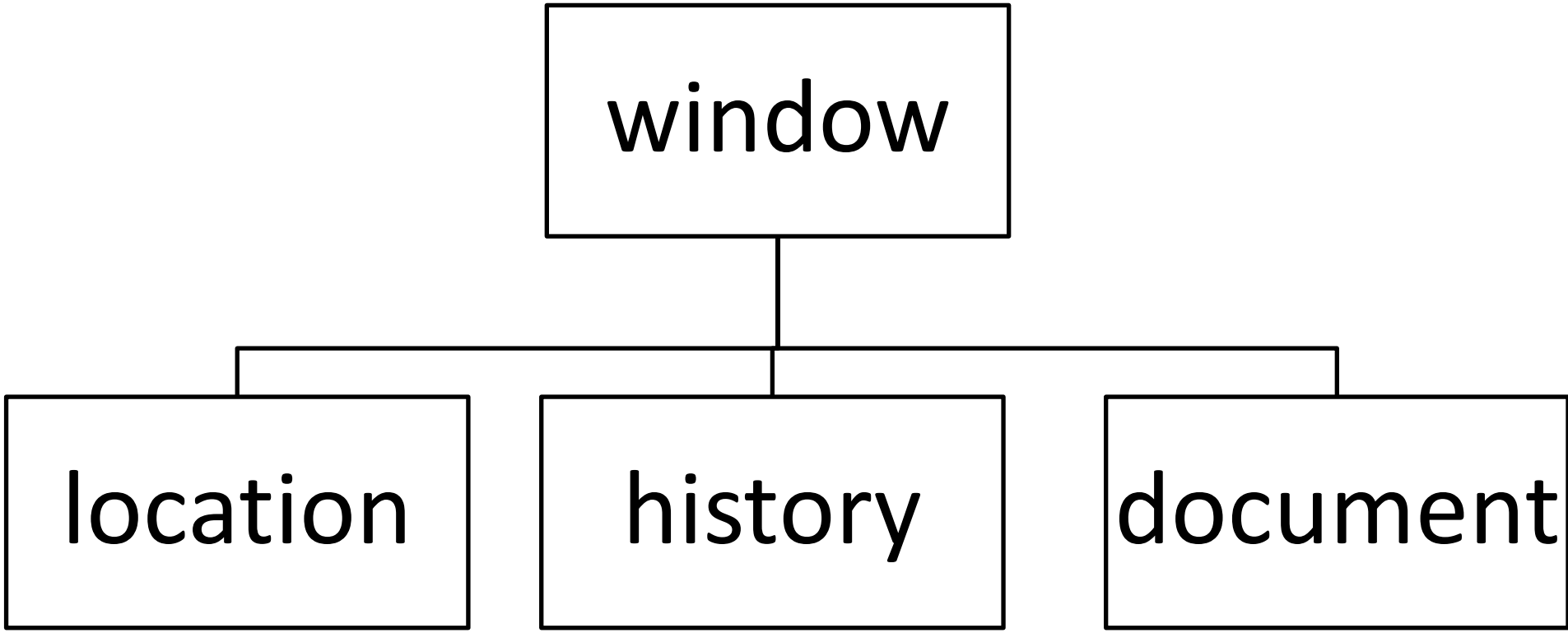
- window: objeto superior na hierarquia, contém propriedades que se aplicam a toda a janela. Há também um objeto desta classe para todas as "sub-janelas" de um documento com frames.

- 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

Linguagem JavaScript - Organograma

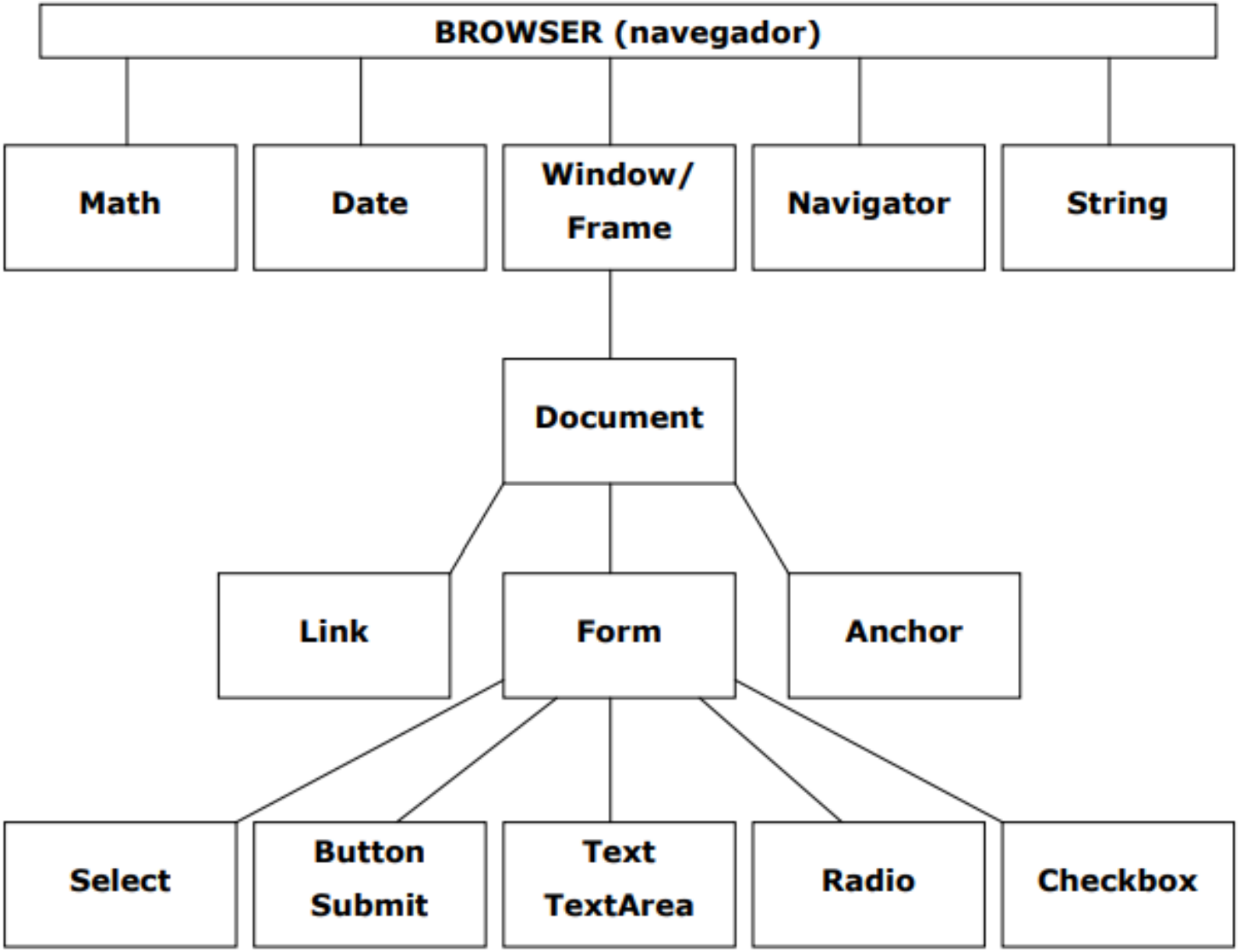


Linguagem JavaScript – Manipulação do Objeto

- A linguagem JavaScript manipula vários tipos de objetos através do uso de suas propriedades e métodos.

- Estes objetos são representados por uma hierarquia, fazendo com que alguns objetos se tornem propriedades de outros.

Linguagem JavaScript – Hierarquia dos Objetos



Linguagem JavaScript – Propriedades de Objetos

- Cada objeto existente na manipulação do JavaScript possuem propriedades (características).

- Exemplo, sabemos que um documento HTML possuem título e corpo, estas características do documento podemos chamar de propriedades que existem neste documento.

- Estas propriedades existem de dois tipos, algumas são os objetos propriamente ditos enquanto outras não.

- Um exemplo disto, é o objeto form (formulário) que é uma propriedade do objeto document (documento).

Linguagem JavaScript – Propriedades de Objetos

- Já a propriedade de título da página (title), é pertencente ao objeto document não havendo nenhuma propriedade sobre ela.

- Então, podemos dizer que a propriedade form do objeto document é um objeto-filho e o objeto document é o objeto-pai.

- Em geral, as propriedades podem conter valores (string, números, entre outros tipos). A utilização de propriedades se dá acompanhada de seu objeto sendo separados por um ponto apenas. Veja abaixo a sintaxe de utilização de propriedades: nomeObjeto.propriedade

Linguagem JavaScript – Métodos de Objetos

- Além das propriedades, os objetos podem conter métodos que são funções pré-definidas pela linguagem JavaScript que irão executar determinada operação.

- Por exemplo dentro de um documento o usuário poderá utilizar o método de escrever neste documento para exibir um texto qualquer.

- Os métodos estarão sempre associados à algum objeto presente no documento e cada método faz parte de um objeto específico.

Linguagem JavaScript – Métodos de Objetos

- Não tente usar métodos em objetos que não o utilizam, isto faz com que a linguagem JavaScript cause erro na execução do script.

- Na maioria das vezes os métodos são usados para alterar o valor de uma propriedade ou executar uma tarefa específica.

- Veja a sintaxe de utilização dos métodos: nomeObjeto.método(argumento) Na sintaxe apresentada, nomeObjeto faz referência ao objeto a ser utilizado e o qual sofrerá uma ação do método, já método é o nome de identificação do método usado e entre parênteses (argumento) é a expressão ou valor opcional que será usada para alterar sobre o objeto.

JavaScript – Tag

- Para inserir códigos JavaScript, iremos fazê-lo em uma Tag HTML apropriada:

<script> </script>

- O JavaScript é orientado a objetos;

- Primeira Classe: document

JavaScript – JavaScript Externo

- Da mesma forma como nos arquivos CSS, podemos deixar funções e comandos JavaScript em arquivos externos:
- Estes arquivos devem ter a extensão .JS
- Para importar: <script src="meuScript.js"></script>

JS – Dados primitivos

- JavaScript é uma linguagem de tipagem *dinâmica*.
- Tipagem é o ato de você atribuir a um elemento abstrato um formato de dados.
- O tipo será automaticamente determinado quando o programa for processado.
- O JS permite você atribuir diversos tipos de valor a uma variável sem que isso seja fixado, uma variável pode em um momento ser do *tipo a* e logo depois posso atribuir um *tipo b* a ela.

JS – Tipagem Dinamica - Exemplo

```
var foo = 42; // foo é um Number agora

foo = "bar"; // foo é um String agora

foo = true; // foo é um Boolean agora
```

JS - Tipo de Dados Primitivos

number
- infinity
- NaN

string

boolean

null

undefined

object
- Array

function

Dados Primitivos - Number

- Tipo de dado numérico;
- Em outras linguagens de programação diferentes tipos numéricos podem existir, por exemplo: Integers (Inteiros), Floats (Pontos Flutuantes), Doubles (Dobros), ou Bignums.
- Mas para JS, todo número é um number

Dados Primitivos – Number – Problema Clássico

```
1 const meunumero = 0.1 + 0.2 //0.30000000000004
2 meunumero.toFixed();  //0
3 meunumero.toFixed(1); //0.3
4 meunumero.toFixed(2); //0.30
5
6 meunumero.toPrecision(); //0.30000000000004;
7 meunumero.toFixed(1); //0.3
8 meunumero.toFixed(2); //0.30
```

- Utilizamos os artifícios como o método toFixed() e o toPrecision() e isso serve para outros casos também, como arredondar um dado importante;

Dados Primitivos - Infinity

- Valor numérico que representa o infinito;
- ***+Infinity*** que é uma maneira abstrata de representar o valor infinito positivo;
- ***-Infinity*** que segue a mesma lógica que sua contraparte porém com o valor infinito negativo

Dados Primitivos - NaN

- Não é um número. (Not a Number);
- O **NaN** é utilizado para representar o resultado de uma tentativa de operação numérica, mas que não é um número;
- Ele é retornado quando uma operação matemática falha ou quando uma função tenta transformar uma string em inteiro.

Dados Primitivos - String

- É uma sequencia de caracteres;
- Pode ser representada com aspas simples (') ou com aspas duplas("");
- Diferente de linguagens como C, strings em JS são imutáveis. Isto significa que: uma vez criada a string, não é possível modificá-la.
- Entretanto, ainda é possível criar outra string baseada em um operador na string original. Por exemplo:

  - Uma substring da original a partir de letras individuais ou usando String.substr().
  - Uma concatenação de duas strings usando o operador (+) ou String.concat().

Dados Primitivos - Boolean

- Uma vez entendido o conceito de sistema binário, modo operante dos computadores, podemos associar esse mesmo conceito lógico para os dados do tipo boolean;

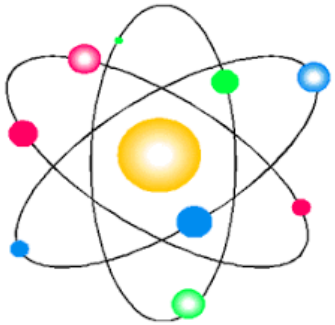
- Podemos fazer uso da função Boolean() passando como parâmetro uma condição ou até mesmo uma expressão. A função irá validar o parâmetro e retornará um valor “booleano” como resultado;

- Se é falso ou verdadeiro

Dados Primitivos - Null

- Um valor nulo representa uma referencia que aponta, geralmente de maneira intencional, para um objeto ou endereço de memória inválido ou inexistente;

- O significado do valor nulo varia entre as implementações das linguagens.



- Fazendo uma analogia com átomos...

Imagine a estrutura de um átomo, que é composta por um núcleo envolto por elétrons, o Null apontaria essa estrutura a qual sabemos que existe mas está em outro nível espacial, ou seja, mundos diferentes.

Dados Primitivos - Undefined

- algo que não teve seu valor definido, algo sem um valor;
- O undefined existe mas ainda não teve seu valor definido.
- Ele é designado automaticamente quando por exemplo, uma variável vazia é criada .
- Assim como o ***null*** também podemos utilizá-lo para esvaziar um objeto. Uma função também retornará ***undefined*** caso ela não retorne nenhum valor .

Dados Primitivos – Null ≠ Undefined

- eles são iguais em valor, mas diferentes em tipos;
- O tipo do ***undefined*** é o próprio ***undefined***
- E ***null*** é mostrado que é um objeto

Tratando dados do tipo número

```
var n1 = window.prompt("Digite um número: ")//String;
var n2 = window.prompt("Digite outro número: ")//String;
var s = n1 + n2 //converter em number;
window.alert(" A soma dos valores é: " + s);
// + adição
// + concatenação
```

- Por exemplo, se digitado n1 = 1 e n2 = 4;
- Dessa forma o valor retornado na janela é 14 e não 5!

Convertendo em Number - 3 opções

Number.parseInt = número inteiro;

Number.parseFloat = número decimal;

Number = tanto número como inteiro como decimal

Convertendo em Number - Number.parseInt

```
var n1 = Number.parseInt(window.prompt("Digite um número: "));
var n2 = Number.parseInt(window.prompt("Digite outro número: "));
var s = n1 + n2;
window.alert(" A soma dos valores é: " + s);
```

- Por exemplo, se digitado n1 = 1 e n2 = 4;
- Dessa forma o valor retornado na janela é 5!
- Mas se digitado para n1 = 1 e n2 = 4.5;

- Dessa forma o valor retornado na janela também é 5, pois estamos forcando que o retorno seja um número inteiro.

Convertendo em Number - Number.parseFloat

```
var n1 = Number.parseFloat(window.prompt("Digite um número: "));
var n2 = Number.parseFloat(window.prompt("Digite outro número: "));
var s = n1 + n2;
window.alert(" A soma dos valores é: " + s);
```

- Por exemplo, se digitado n1 = 1 e n2 = 4.5;
- Dessa forma o valor retornado na janela é 5.5, pois estamos forcando que o retorno seja um número flutuante.

Convertendo em Number - Number

```
var n1 = Number(window.prompt("Digite um número: "));
var n2 = Number(window.prompt("Digite outro número: "));
var s = n1 + n2;
window.alert(" A soma dos valores é: " + s);
```

- Agora quem vai decidir é o próprio JS se o valor a ser apresentado será inteiro ou decimal.

Convertendo em String

```
var n1 = Number(window.prompt("Digite um número: "));
var n2 = Number(window.prompt("Digite outro número: "));
var s = n1 + n2;
window.alert(" A soma dos valores é: " + s.toString() ); ou
window.alert(" A soma dos valores é: " + String(s))
```

- O resultado não será mais um número e sim um conjunto de caracteres numéricos;

Template String

- *Template* Strings são *string*s que permitem expressões embutidas.
- Você pode *utilizar string* multi-linhas e interpolação de *string* com elas.
- Tornar o seu código um pouco mais legível.
- Estrutura: utiliza – se a crase e *placeholders* ${variável}

Template String - Exemplo

```
var aluno = window.prompt('Digite seu nome: ');
var nota1 = Number(window.prompt("Digite a nota da sua primeira ativi
dade: "));
var nota2 = Number(window.prompt("Digite a nota da sua segunda ativi
dade: "));
var media = (nota1 + nota2) / 2;
window.alert(" A média de " + aluno + " é " + media);
```

- Método de concatenação

```
window.alert(`A média de ${aluno} é ${media}`);
```

- Utilizando o template string

Tratando dados do tipo string

var.lenght = atributo que informa qtos caracteres a string tem;

var.toUpperCase() = deixa tudo em letras maiúsculas;

var.toLowerCase()= deixa tudo em letras minúsculas;

Tratando dados do tipo string – exemplo de var.lenght

```
var nome = window.prompt('Qual o seu nome?');
document.write(`${nome} seu nome tem ${nome.length} letras.`);
```

- document.write irá aparecer a informação na tela e não na janela.

Tratando dados do tipo string – exemplo de var.toUpperCase()

```
var nome = window.prompt('Qual o seu nome?');
document.write(`${nome} seu nome em maiúsculo é ${nome.toUpperCase()}.`);
```

- document.write irá aparecer a informação na tela e não na janela.

Tratando dados do tipo string – exemplo de var.toLowerCase()

```
var nome = window.prompt('Qual o seu nome?');
document.write(`${nome} seu nome em minusculo é: ${nome.toLowerCase()}.`);
```

- document.write irá aparecer a informação na tela e não na janela.

Fixando conteúdo...

Crie um projeto, **pelo método de concatenação** no qual será inserido os dados e exibidos no documento com as seguintes programações:
OBS: pastas separadas, contendo html, css e js

Nome do aluno completo: primeiro nome em minúsculo e sobrenome em maiúsculo

Matéria A
Nota 1 : float
Nota 2: float
Média A

Matéria B
Nota 1: int
Nota 2: int
Média B

Matéria C
Nota 1 number (entrada decimal)
Nota 2 number (entrada inteiro)
Média C

Matéria D
Nota 1: int
Nota 2: float
Média D (saída em int)