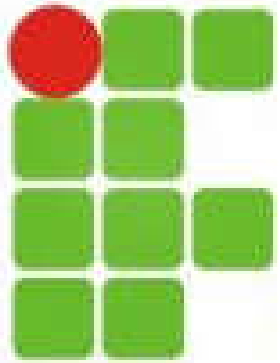
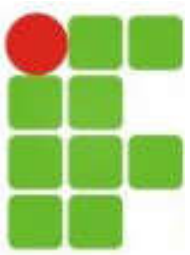
Fundamentos de Web Design 2

Professor Eng. Dr. Will Roger Pereira

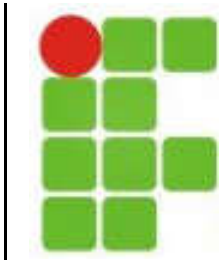


Conteúdo



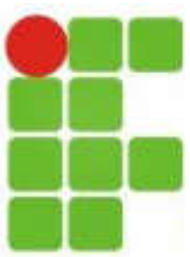
- Variáveis e tipos de dados;
- Identificadores;
- Tipos primitivos;
- Conversões de dados.

Características



- Variáveis dinamicamente tipadas;
- Tipo depende do conteúdo da variável;
- O conteúdo da variável pode mudar, inclusive seu tipo;
- Tomar bastante cuidado, visto que o tipo da variável muda completamente seu comportamento perante funções e operações.

Identificadores de uma variável



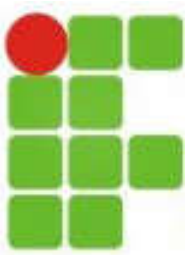
- Deve começar com:
  - Caractere;
  - Cifrão (padrão PHP);
  - Underline.

- Não pode haver espaço!!!

- _teste, valor, $n, x... → OK

- 9will, x hue, (jovem → ERRO!!!

Identificadores de uma variável

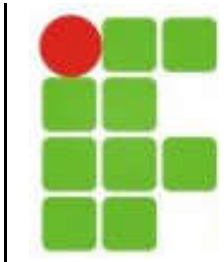


- Não pode ser uma palavra reservada:

• abstract (*)	• final	• protected (*)
• as (2)	• finally	• public (2)
• boolean	• float	• return
• break	• for	• short
• byte	• function	• static (2)
• case	• goto (*)	• super (2)
• catch	• if	• switch
• char	• implements (*)	• synchronized (*)
• class (2)	• import (2)	• this
• continue	• in	• throw
• const (2)	• instanceof	• throws (*)
• debugger (*)	• int	• transient (*)
• default	• interface (2)	• true
• delete	• is (2)	• try
• do	• long	• typeof
• double	• namespace (2)	• use (2)
• else	• native (*)	• var
• enum (*)	• new	• void
• export (2)	• null	• volatile (*)
• extends (2)	• package (2)	• while
• false	• private (2)	• with

(2) próxima versão – (*) reservada para uso futuro

Identificadores de uma variável



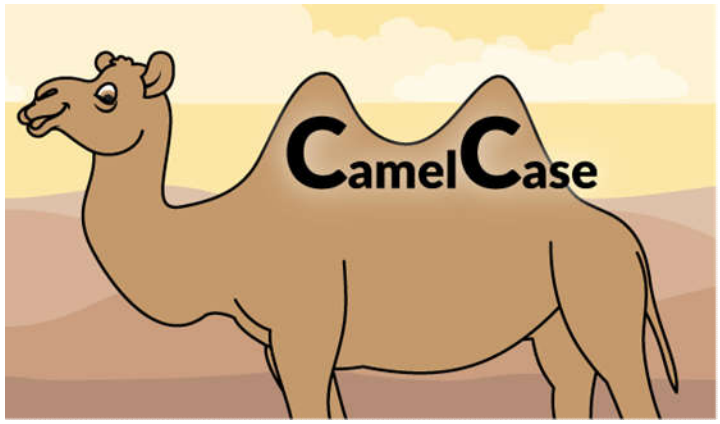
- Também há as palavras reservadas por implementação:

alert, array, blur, date, document, escape, eval, focus, function, history, image, isNaN, length, location, math, name, navigator, number, object, onLoad, open, outerHeight, parent, parseFloat, regExp, status, string.

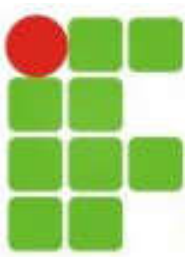
Notação CamelCase

- CamelCase é uma notação utilizada para nomenclatura de funções, variáveis, etc;

- Para variáveis, a primeira letra da primeira palavra é minúscula, enquanto as primeiras letras das demais palavras, caso hajam, são maiúsculas.



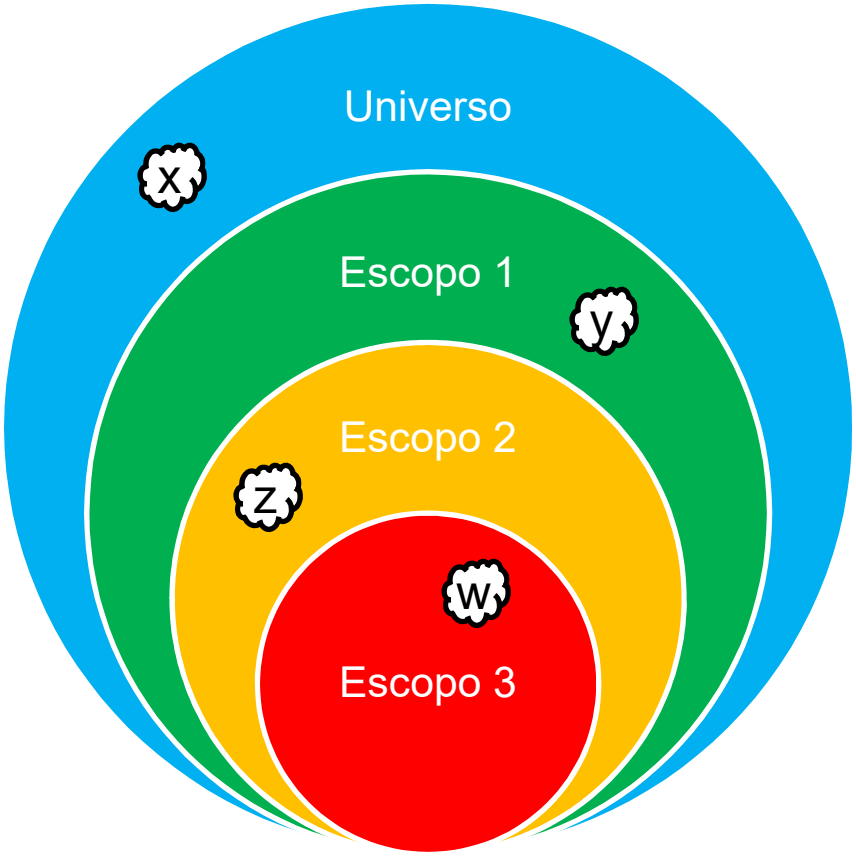
Escopo das variáveis



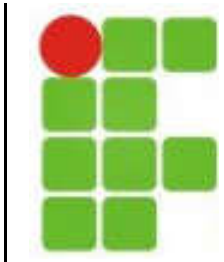
- As variáveis podem ser locais ou globais;
- Depende de onde são declaradas;
- Se declaradas **dentro** do escopo, não estão disponíveis fora dele;
- Se declaradas **fora** do escopo, estão disponíveis dentro dele;

Escopo das variáveis

- Universo:
  - <script> ... </script>
- Escopo:
  - Prisão;
  - Membro externo pode adentrar a prisão;
  - Membro interno não pode sair.
  - Em quais escopos as variáveis são conhecidas??

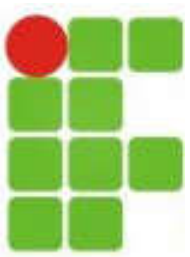


Problemas com variáveis globais



- São compartilhadas dentre todo seu código e aplicação;
- Se forem alteradas, seu valor será alterado para todos os que utilizarem tal variável;
- Evite o uso de variáveis globais;
- Caso vá utilizar mesmo assim, utilizar um identificador de fácil visualização: G_x, G_nome...

Criando variáveis



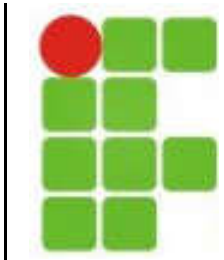
- Palavra-chave **var**;

- Variável não inicializada:
  - Sintaxe: **var** identificador;
  - Valor não-definido → **undefined**.

- Variável inicializada:
  - Deve ser utilizado o operador de atribuição **"="**;
  - Sintaxe: **var** identificador = valor inicial.

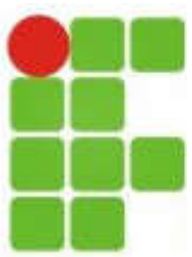
```
var jovem;

var nome = "Will";

var x = 10;

var falso = false;
```

Tipos primitivos



- **String** → Valor textual, texto, palavras;
- **Number**→ Valor numérico, números inteiros, reais;
- **Boolean** → Valor booleano, verdadeiro ou falso;
- Lembre-se: Variáveis são identificados pelo tipo do dado contido na mesma;
- É possível saber o tipo de uma variável através da função typeof → **typeof(variável).**

String

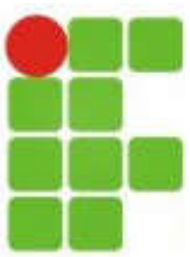


- Valores entre aspas (“...”) ou apóstrofos (‘...’);

- Exemplos:

```
var str1 = "";                                  //string vazia
var str2 = '';                                  //string vazia
var str3 = "Fundamentos de Web Design";         //OK
var str4 = "Fundamentos de 'Web' Design";       //OK
var str5 = 'Fundamentos de "Web" Design';       //OK
var str6 = "Fundamentos de "Web" Design";       //ERRO
var str7 = "Fundamentos de \nWeb Design";       //OK, salta linha
var str8 = "Fundamentos \"de Web\" Design"; //OK
```

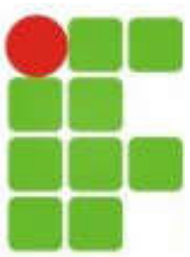
Number



- Valores numéricos, com ou sem parte real, negativos e positivos, ou infinito;
- O separador entre parte real e fracionária é o ponto (.).
- Exemplos:

```
var n1 = 0;
var n2 = 7;
var n3 = 199;
var n4 = -255;
var n5 = 3.1415;
var n6 = -0.25;
var n7 = Infinity; //Infinito
```

Boolean

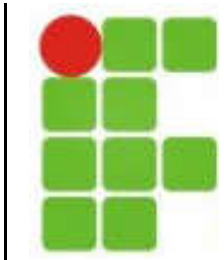


- Valores: true (verdadeiro) ou false (falso);
- Exemplos:

```
var b1 = true;
var b2 = false;
```

- Muito utilizados na lógica de programação e na comparação com outros valores.

Valores Nulos ou Indefinidos

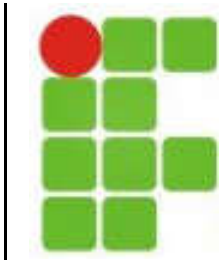


- null:
  - Variável criada e atribuída valor nulo à ela;
  - var nulo = null;

- undefined:
  - Variável criada mas não inicializada;
  - var indefinido;

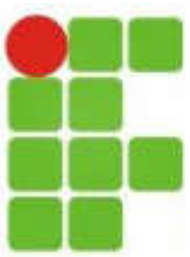
- NaN:
  - Not a Number;
  - Uma variável do tipo string que não pôde ser convertida para number.

Conversões de Tipo



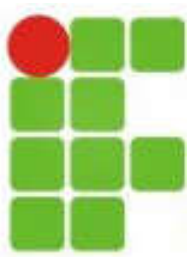
- Em Javascript, caso os dados sejam compatíveis, é possível fazer a conversão de um tipo para outro:
  - number → string;
  - boolean → string;
  - string → number;
  - string → boolean;
  - number → boolean.

Conversões para strings



- Use a função **String(valor)**:
  - String(50) → “50”;
  - String(-67) → “-67”;
  - String(0) → “0”;
  - String(true) → “true”;
  - String(false) → “false”;
  - var nulo = null; String(nulo) → “null”;
  - var indefinido; String(indefinido) → “undefined”.

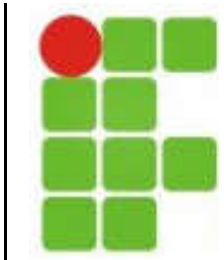
Conversões para number



- Use a função **Number(valor)**:
  - Number(“50”) → 50;
  - Number(“-67”) → -67;
  - Number(“0”) → 0;
  - Number(true) → 1;
  - Number(false) → 0;
  - Number(“fwd2”) → NaN;
  - var nulo = null; Number(nulo) → 0;
  - var indefinido; String(indefinido) → NaN.

Para verificar se um valor não é propriamente um número, utilize a função **isNaN(valor)**. Caso não seja um número, retornará **true.**

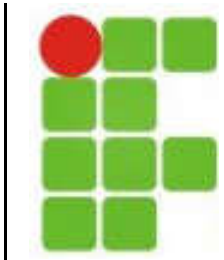
Conversões para boolean



- Use a função **Boolean(valor)**:

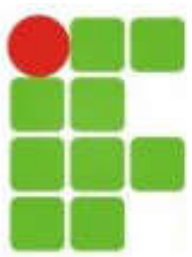
Entrada (input)	Resultado
Undefined	false
Null	false
Boolean	Valor dele mesmo
Number	false se o número for 0 ou NaN, caso contrário, true
String	false se a string for vazia, caso contrário, true

Constantes



- Valor apenas de leitura;
- Após inicialização, é impossível alterar seu valor;
- Pode ter escopo local e global;
- Seu identificador, por convenção, possui somente letras maiúsculas;
- Exemplo:
  - const PI = 3.1415;

Arrays



- São estruturas utilizadas para armazenar múltiplos valores em uma variável;

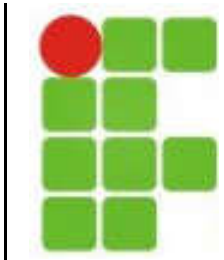
Criação do array:

[valor1, valor2, ..., valorN]

Exemplo:

var carros = [“Fluence”, “Sentra”, “Civic”];

Obtendo o tamanho de um array



- O tamanho de um array é um fator muito importante para operações realizadas com o mesmo.

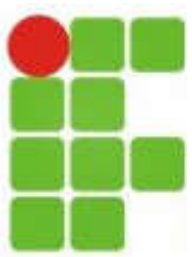
Sintaxe:

refarray.length

Exemplo:

carros.length →3

Acessando elementos de um array



- Posição: 1º, 2º, 3º,...
- Índices de um array: 0, 1, 2,...

Sintaxe:

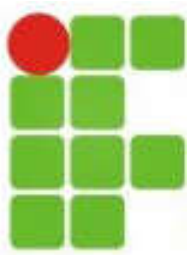
refarray[índice]

Exemplo:

carros[0]; → “Fluence”

- Caso acesse um elemento inexistente → **undefined**.

Substituindo elementos de um array



- Utilize seu índice como referência e o operador de atribuição;

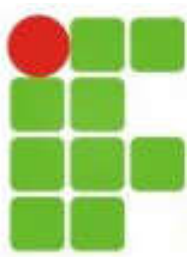
Sintaxe:

refarray[índice] = NovoValor

Exemplo:

carros[0] = “Focus”;

Adicionando elementos ao final de um array



- Adiciona um elemento ao final do array

Sintaxe:

Refarray.push(NovoValor)

Exemplo:

carros.push(“Corolla”);