JavaScript (Parte 3)

Esdras Lins Bispo Jr.
bispojr@ufg.br

Física para Ciência da Computação
Bacharelado em Ciência da Computação

25 de outubro de 2016

UFG
Regional Jataí

Plano de Aula

UFG
Regional Jataí

Sumário

UFG
Regional Jataí

Funções e Métodos

Atribuindo uma função como propriedade de um objeto...

```
1 nomeDoObjeto.nomeDaPropriedade = nomeDaFuncao;
```

Funções e Métodos

Atribuindo uma função como propriedade de um objeto...

```
1 nomeDoObjeto.nomeDaPropriedade = nomeDaFuncao;
```

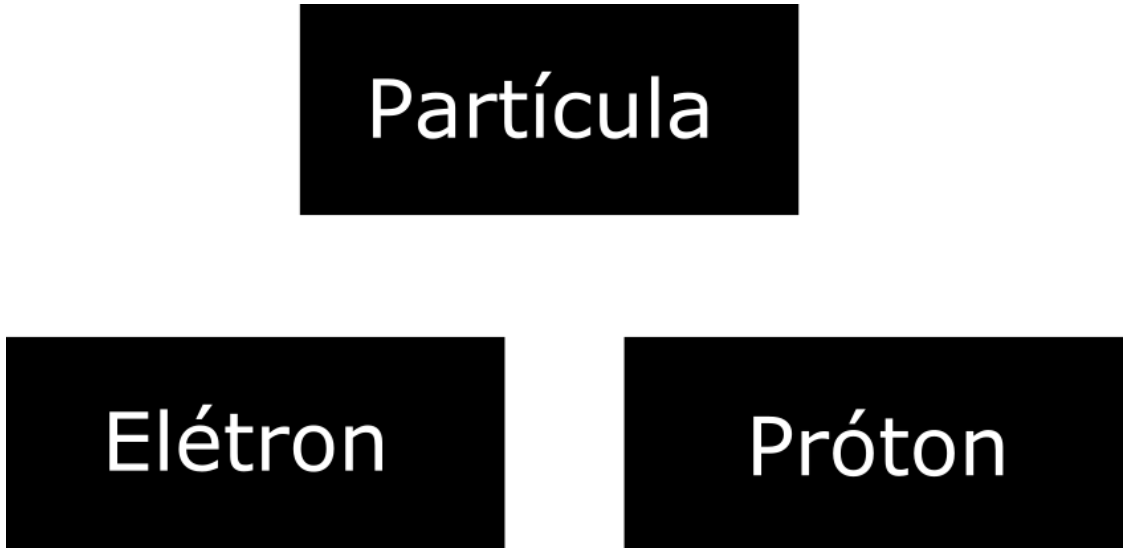
Exemplo

```
1 obj.multiplicar = multiplicar;
```

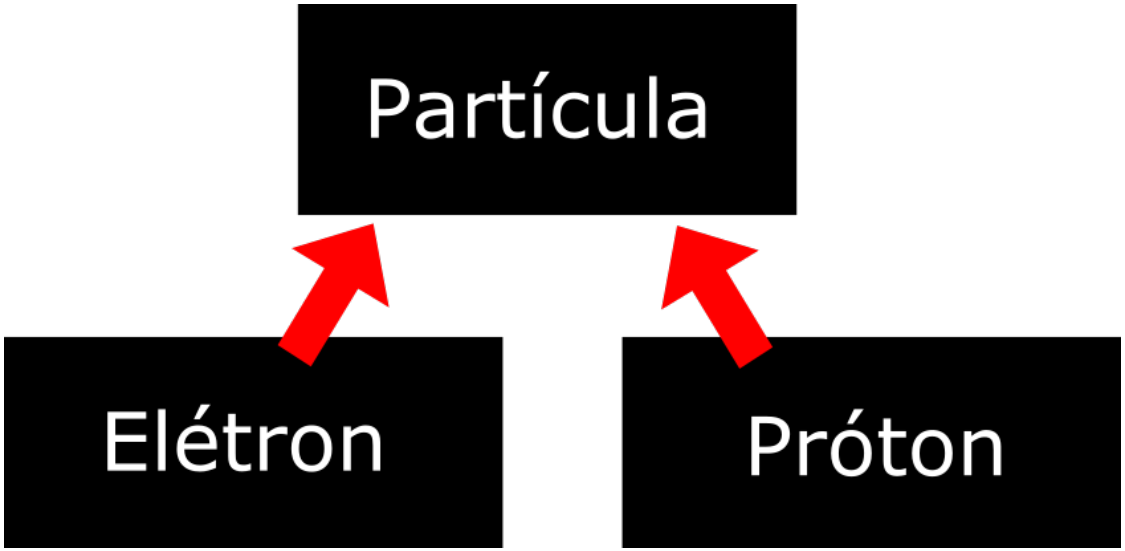
Sumário

UFG
Regional Jataí

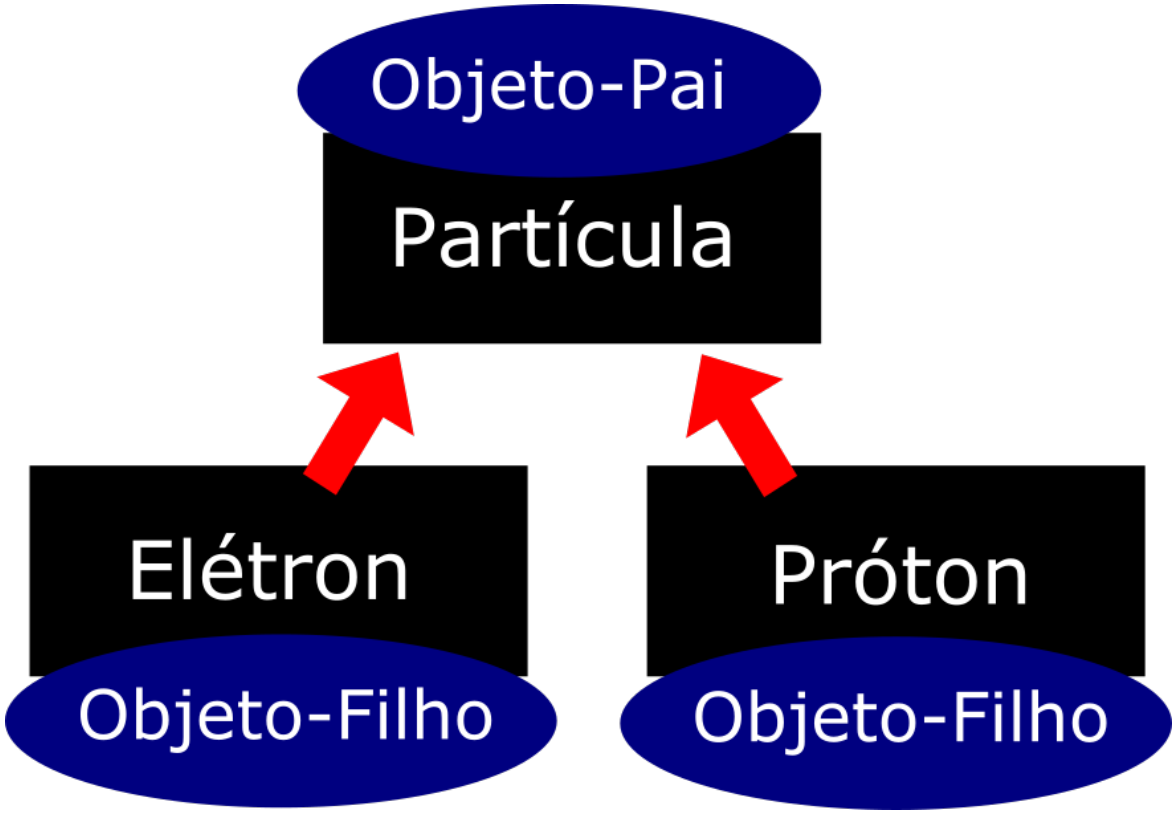
Herança



Herança



Herança


Objeto-Pai
Partícula
Elétron
Objeto-Filho
Próton
Objeto-Filho

Construtor

Objeto a partir de uma função

É possível construir um objeto a partir de uma "função-modelo".

```
1 function Particula(pnome){
2   this.nome = pnome;
3   this.mover = function(){
4     console.log(this.nome + " em movimento!");
5   };
6 }
```

Construtor

Objeto a partir de uma função

```
1 particula1 = new Particula("eletron");
2 console.log(particula1.nome); //exibe "eletron"
3 particula1.mover();
4       //exibe "eletron em movimento"
```

Construtor

Objeto a partir de uma função

```
1 particula1 = new Particula("eletron");
2 console.log(particula1.nome); //exibe "eletron"
3 particula1.mover();
4       //exibe "eletron em movimento"
```

Construtor...

Se uma função é utilizada intencionalmente para criar novos objetos, a chamamos de **construtor**.

UFG
Regional Jataí

Protótipo

Adicionando propriedades...

Após declarado, é possível adicionar propriedades a um construtor.

```
1 Particula.prototype.massa = 1;
2 Particula.prototype.parar =
3     function(){console.log("Eu parei.");};
```

Protótipo

Adicionando propriedades...

Após declarado, é possível adicionar propriedades a um construtor.

```
1 Particula.prototype.massa = 1;
2 Particula.prototype.parar =
3     function(){console.log("Eu parei.");};
```

Para novos objetos...

```
1 particula2 = new Particula("proton");
2 console.log(particula2.massa); //exibe 1
```

Protótipo

Entretanto...

```
1 particula3 = new Particula("neutron");
2 particula3.massa = 2; //propriedade com valor 2
3 console.log(Particula.prototype.massa); //exibe 1
```

Necessário ter em mente...

A propriedade `prototype` modifica apenas o objeto-pai.

Protótipo

Algo interessante...

É possível adicionar propriedades apenas para o objeto-filho.

```
1 particula4 = new Particula("neutrino");
2 particula4.spin = 0; //exibe 0
```

Exemplo...

Objeto Bola

```
1 function Bola (raio, cor) {
2   this.raio = raio;
3   this.cor = cor;
4   this.x = 0;
5   this.y = 0;
6   this.vx = 0;
7   this.vy = 0;
8 }
```

Exemplo...

Objeto Bola

```
1 Bola.prototype.desenhar = function (contexto) {
2   contexto.fillStyle = this.cor;
3   contexto.beginPath();
4   contexto.arc(this.x, this.y,
5                this.raio, 0, 2*Math.PI, true);
6   contexto.closePath();
7   contexto.fill();
8 };
```

Exemplo...

HTML Canvas arc

A propriedade `arc()` cria um arco
(utilizado para criar círculos ou parte de círculos).

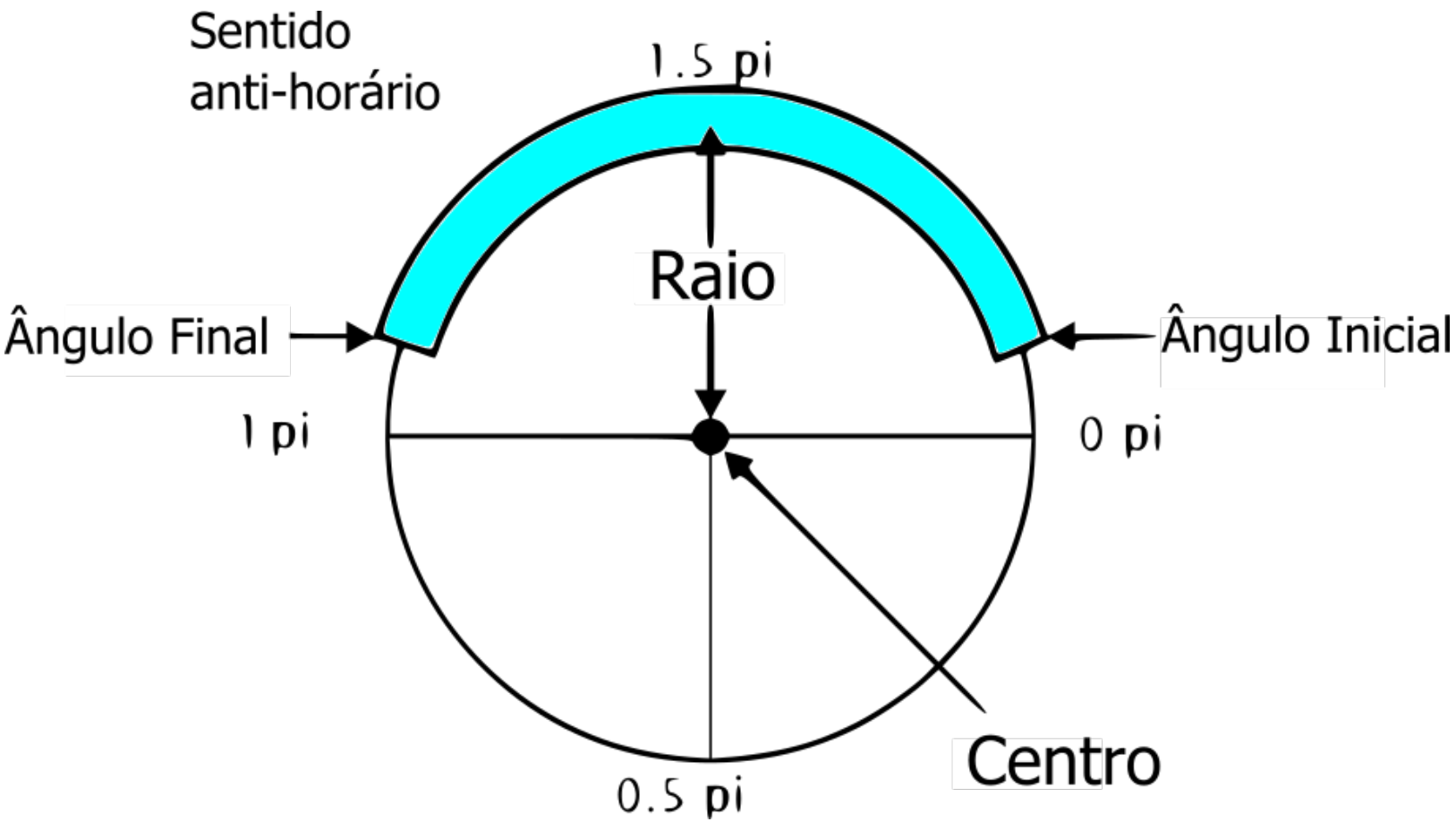
Sintaxe

```
arc(x, y, raio, angIni, angFin, antiHor);
```

Argumentos

- `x` e `y`: coordenadas do arco;
- `raio`: raio do arco;
- `angIni` e `angFin`: ângulos inicial e final do arco (em radianos);
- `antiHor`: valor booleano para o sentido anti-horário.

Exemplo...


Sentido
anti-horário
1.5 pi
Raio
Ângulo Final
Ângulo Inicial
1 pi
0 pi
0.5 pi
Centro

Exemplo...

Objeto Bola

```
1 var canvas = document.getElementById('canvas');
2 var contexto = canvas.getContext('2d');
3 var bola = new Bola(50,'#0000ff');
4 bola.x = 100;
5 bola.y = 100;
6 bola.desenhar(contexto);
```

Sumário

Conceitos Básicos em JavaScript

Variáveis

Uma **variável** funciona como uma caixa que armazena valores de dados.

Conceitos Básicos em JavaScript

Variáveis

Uma **variável** funciona como uma caixa que armazena valores de dados.

Exemplo

```
1 var x;
2 x = 2;
3 var y = 3;
4 z = 2*x + y;  //z - variavel global
5 x = x + 1;
```

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

UFG
Regional Jataí

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

Descrição

- `Number`: número de ponto flutuante com dupla precisão de 64 bits.

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

Descrição

- `Number`: número de ponto flutuante com dupla precisão de 64 bits.
- `String`: uma sequência de caracteres de 16 bits.

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

Descrição

- `Number`: número de ponto flutuante com dupla precisão de 64 bits.
- `String`: uma sequência de caracteres de 16 bits.
- `Boolean`: tem dois valores possíveis: `true` e `false`, ou 1 e 0.

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

Descrição

- `Number`: número de ponto flutuante com dupla precisão de 64 bits.
- `String`: uma sequência de caracteres de 16 bits.
- `Boolean`: tem dois valores possíveis: `true` e `false`, ou 1 e 0.
- `Undefined`: é retornado para uma propriedade de objeto não-existente ou uma variável sem um valor.

Conceitos Básicos em JavaScript

Tipos de Dados

- Variáveis em JavaScript têm tipos de dados dinâmicos;
- i.e., podem armazenar tipos de dados diferentes em momentos diferentes.

Descrição

- `Number`: número de ponto flutuante com dupla precisão de 64 bits.
- `String`: uma sequência de caracteres de 16 bits.
- `Boolean`: tem dois valores possíveis: `true` e `false`, ou 1 e 0.
- `Undefined`: é retornado para uma propriedade de objeto não-existente ou uma variável sem um valor.
- `Null`: tem apenas um valor → `null`

Conceitos Básicos em JavaScript

Descrição

- `Object`: armazena uma coleção de propriedades e métodos.

Conceitos Básicos em JavaScript

Descrição

- `Object`: armazena uma coleção de propriedades e métodos.
- `Array`: um objeto consistindo de uma lista de dados de algum tipo.

Conceitos Básicos em JavaScript

Descrição

- `Object`: armazena uma coleção de propriedades e métodos.
- `Array`: um objeto consistindo de uma lista de dados de algum tipo.
- `Function`: um objeto passível de ser chamado, que executa um bloco de código.

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;
- O maior valor que `Number` pode armazenar é $1,8 \times 10^{308}$;

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;
- O maior valor que `Number` pode armazenar é $1,8 \times 10^{308}$;
- O número de átomos do universo visível é estimado em 10^{80};

UFG
Regional Jataí

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;
- O maior valor que `Number` pode armazenar é $1,8 \times 10^{308}$;
- O número de átomos do universo visível é estimado em 10^{80};
- Permite armazenar números bem pequenos como 5×10^{-324}.

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;
- O maior valor que `Number` pode armazenar é $1,8 \times 10^{308}$;
- O número de átomos do universo visível é estimado em 10^{80};
- Permite armazenar números bem pequenos como 5×10^{-324}.

Valores especiais em `Number`

- `NaN` (Not a Number): e.g. raiz de -1, ou 0/0;

Conceitos Básicos em JavaScript

Números

- Existe um único tipo de dado numérico em JavaScript: `Number`;
- Não há diferença entre números inteiros e pontos-flutuantes, por exemplo;
- O maior valor que `Number` pode armazenar é $1,8 \times 10^{308}$;
- O número de átomos do universo visível é estimado em 10^{80};
- Permite armazenar números bem pequenos como 5×10^{-324}.

Valores especiais em `Number`

- `NaN` (Not a Number): e.g. raiz de -1, ou 0/0;
- `Infinity` e `-Infinity`: divisão de um valor não-zero por zero (e.g. 5/0 ou (-3)/0).

Conceitos Básicos em JavaScript

Strings

Uma **string** é uma sequência de caracteres. Normalmente, vem delimitada por aspas (simples ou dupla).

Conceitos Básicos em JavaScript

Strings

Uma **string** é uma sequência de caracteres. Normalmente, vem delimitada por aspas (simples ou dupla).

Exemplo

```
1 var str = "Olah jovens";
2 console.log(str);
```

Conceitos Básicos em JavaScript

Booleanos

Um **booleano** pode ter apenas dois valores: `true` ou `false`.

Conceitos Básicos em JavaScript

Booleanos

Um **booleano** pode ter apenas dois valores: `true` ou `false`.

Exemplo

```
1 var bln = false;  // valor booleano false
2 bln = "true";   // string "true"... cuidado!
```

Conceitos Básicos em JavaScript

Undefined

- O tipo de dados `Undefined` tem um único valor: `undefined`.

Conceitos Básicos em JavaScript

Undefined

- O tipo de dados `Undefined` tem um único valor: `undefined`.
- Casos que uma variável assume `undefined`:

Conceitos Básicos em JavaScript

Undefined

- O tipo de dados `Undefined` tem um único valor: `undefined`.
- Casos que uma variável assume `undefined`:
  - Propriedade inexistente;

Conceitos Básicos em JavaScript

`Undefined`

- O tipo de dados `Undefined` tem um único valor: `undefined`.
- Casos que uma variável assume `undefined`:
  - Propriedade inexistente;
  - Variável declarada, mas não inicializada;

Conceitos Básicos em JavaScript

`Undefined`

- O tipo de dados `Undefined` tem um único valor: `undefined`.
- Casos que uma variável assume `undefined`:
  - Propriedade inexistente;
  - Variável declarada, mas não inicializada;
  - Função sem `return`;

Conceitos Básicos em JavaScript

`Undefined`

- O tipo de dados `Undefined` tem um único valor: `undefined`.
- Casos que uma variável assume `undefined`:
  - Propriedade inexistente;
  - Variável declarada, mas não inicializada;
  - Função sem `return`;
  - Argumento não fornecido de uma função.

UFG
Regional Jataí

Conceitos Básicos em JavaScript

Null

- O tipo de dados `Null` tem um único valor: `null`.

Conceitos Básicos em JavaScript

`Null`

- O tipo de dados `Null` tem um único valor: `null`.
- Diferença crucial entre `null` e `undefined`: `null` é associado a variável de forma propositall.

Conceitos Básicos em JavaScript

Null

- O tipo de dados `Null` tem um único valor: `null`.
- Diferença crucial entre `null` e `undefined`: `null` é associado a variável de forma propositadal.

Exemplo

```
1 var noVal = null;
2 console.log(typeof noVal);  // retorna "Object"
```

Conceitos Básicos em JavaScript

Vetores

Um vetor é uma coleção de objetos.

Conceitos Básicos em JavaScript

Vetores

Um vetor é uma coleção de objetos.

Exemplo 1

```
1 var arr = new Array();
2 arr = [2, 4, 6];
3 console.log(arr[1]); // imprime 4
4 arr[3] = 8; // 8 - quarto elemento do vetor
```

Conceitos Básicos em JavaScript

Exemplo 2

```
1 var xArr = new Array();
2 var yArr = new Array();
3 xArr = [1,2];
4 yArr = [3,4];
5 var zArr = new Array(xArr,yArr);
6 console.log( zArr[0][1] ); // retorna 2
7 console.log( zArr[1][0] ); // retorna 3
```

Conceitos Básicos em JavaScript

Operadores (Exemplo 1)

```
1 var x = 5;
2 var y = 3;
3 x%y; // retorna 2
4 var z;
5 z = x++; // atribui o valor de x a z, e depois
      incrementa x
6 console.log(z); // retorna 5
7 z = ++x; // incrementa o valor de x, e depois
      atribui-o a z
8 console.log(z); //retorna 7
```

Conceitos Básicos em JavaScript

Operadores (Exemplo 2)

```
1 var a = 1;
2 a = a + 1;
3 console.log(a); // retorna 2
4 a += 1; // forma reduzida de a = a + 1
5 console.log(a); // retorna 3
6 a = 4*a;
7 console.log(a); // retorna 12
8 a *= 4; // forma reduzida de a = a*4
9 console.log(a); // retorna 48
```

UFG
Regional Jataí

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de a;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de `a`;
- `Math.pow(a,b)`: `a` elevado à potência de `b`;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de `a`;
- `Math.pow(a,b)`: `a` elevado à potência de `b`;
- `Math.sqrt(a)`: raiz quadrada de `a`;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de a;
- `Math.pow(a,b)`: a elevado à potência de b;
- `Math.sqrt(a)`: raiz quadrada de a;
- `Math.ceil(a)`: o menor inteiro que é maior que a;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de a;
- `Math.pow(a,b)`: a elevado à potência de b;
- `Math.sqrt(a)`: raiz quadrada de a;
- `Math.ceil(a)`: o menor inteiro que é maior que a;
- `Math.floor(a)`: o maior inteiro que é menor que a;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de a;
- `Math.pow(a,b)`: a elevado à potência de b;
- `Math.sqrt(a)`: raiz quadrada de a;
- `Math.ceil(a)`: o menor inteiro que é maior que a;
- `Math.floor(a)`: o maior inteiro que é menor que a;
- `Math.round(a)`: o inteiro mais próximo de a;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de `a`;
- `Math.pow(a,b)`: `a` elevado à potência de `b`;
- `Math.sqrt(a)`: raiz quadrada de `a`;
- `Math.ceil(a)`: o menor inteiro que é maior que `a`;
- `Math.floor(a)`: o maior inteiro que é menor que `a`;
- `Math.round(a)`: o inteiro mais próximo de `a`;
- `Math.max(a,b,c,...)`: o maior entre `a, b, c, ...`;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de a;
- `Math.pow(a,b)`: a elevado à potência de b;
- `Math.sqrt(a)`: raiz quadrada de a;
- `Math.ceil(a)`: o menor inteiro que é maior que a;
- `Math.floor(a)`: o maior inteiro que é menor que a;
- `Math.round(a)`: o inteiro mais próximo de a;
- `Math.max(a,b,c,...)`: o maior entre a, b, c, ...;
- `Math.min(a,b,c,...)`: o menor entre a, b, c, ...;

Conceitos Básicos em JavaScript

Alguns métodos matemáticos...

- `Math.abs(a)`: valor absoluto de `a`;
- `Math.pow(a,b)`: `a` elevado à potência de `b`;
- `Math.sqrt(a)`: raiz quadrada de `a`;
- `Math.ceil(a)`: o menor inteiro que é maior que `a`;
- `Math.floor(a)`: o maior inteiro que é menor que `a`;
- `Math.round(a)`: o inteiro mais próximo de `a`;
- `Math.max(a,b,c,...)`: o maior entre `a`, `b`, `c`, `...`;
- `Math.min(a,b,c,...)`: o menor entre `a`, `b`, `c`, `...`;
- `Math.random()`: um número pseudo-aleatório `n`, em que $0 \leq n < 1$.

Conceitos Básicos em JavaScript

Exemplo com Math.random()

```
1 vx = Math.random()*5; // 0 <= n < 5
2 vy = (Math.random()-0.5)*4; // -2 <= n < 2
```

Conceitos Básicos em JavaScript

Outros exemplos...

```
1 if (a == 0){
2   // faca algo se a=0
3 } else if (a < 0 ) {
4   // faca algo caso a seja negativo
5 } else if (a > 0) {
6   // faca algo caso a seja positivo
7 } else {
8   // faca algo caso o valor de a seja um NaN
9 }
```

UFG
Regional Jataí

Conceitos Básicos em JavaScript

Outros exemplos...

```
1 var sum = 0;
2 for (var i = 1; i <= 100; i++) {
3   sum += i;
4 }
5 console.log(sum);
```

Conceitos Básicos em JavaScript

Outros exemplos...

```
1 var sum = 0;
2 var i = 1;
3 while (i <= 100) {
4   sum += i;
5   i++;
6 }
7 console.log(sum);
```

JavaScript (Parte 3)

Esdras Lins Bispo Jr.
bispojr@ufg.br

Física para Ciência da Computação
Bacharelado em Ciência da Computação

25 de outubro de 2016

UFG
Regional Jataí