LINGUAGENS DE PROGRAMAÇÃO

Seminário:

JavaScript

(ECMAScript)

Roger Senna Rosa

INTRODUÇÃO

- Linguagem interpretada;
- Multi-paradigma;
  - Orientado a objetos
  - Estruturada
- Criada em 1995;
- Desenvolvida pela Netscape;
- “Criada em apenas 10 dias”;

INTRODUÇÃO

- Pensada para ser concorrente do C++ e Java;
- Padrão ECMAScript
  - ActionScript (Adobe)
  - JScript(Microsoft, IE)
- Originalmente desenvolvido como linguagem client-side;

INTRODUÇÃO

- Engines (motores) usam várias abordagens:
  - V8(Chromium): Compilação em assembly language.
  - SpiderMonkey(Netscape/Mozilla): C++ Just-in-Time Compiling.
  - Rhino(Netscape/Mozilla): Java bytecode.

INTRODUÇÃO

- Implementações server-side:
  - Node.js(V8)
  - Whitebeam(Rhino)
  - APE (SpiderMonkey)

VISÃO GERAL

- Linguagem amplamente usada para aplicações web;
- Usada comumente em conjunto com HTML e CSS;
- Dinamismo para páginas web;

VISÃO GERAL

- Ajax = JavaScript + XML (Páginas interativas);
- JQuery, biblioteca que facilita interação com HTML;

VISÃO GERAL

- Cursos online, como W3Schools, CodeCademy...
- “Desafios”: CodeCombat, CodeFights...
- Interpretadores online: repl.it, e outros.
- Node.js: interpretador JavaScript para linha de comando.

VISÃO GERAL

- Pouca literatura disponível;
- Códigos pouco homogêneos;
- Programadores “amadores”;

GETTING STARTED

- Para começar (do jeito difícil) podemos abrir o bloco de notas, ou outro editor de texto como pluma, gedit, etc.

- Agora, vamos criar nosso “Hello World” em JavaScript.

GETTING STARTED

- Hello World:

HelloWorld.htm

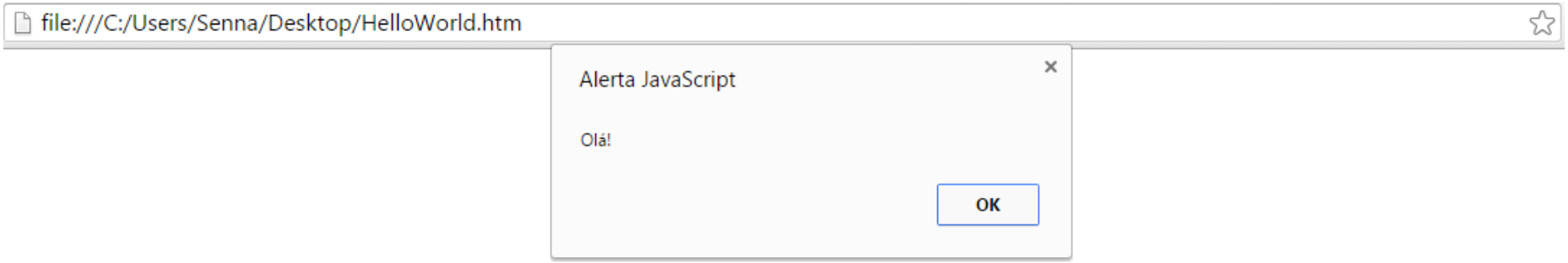
```
1  <script language="javascript">
2  alert("Ola!");
3  </script>
```

GETTING STARTED

- Agora salvamos o arquivo como helloworld.htm, e abrimos o mesmo arquivo com um navegador qualquer:

GETTING STARTED

- Hello World:



GETTING STARTED

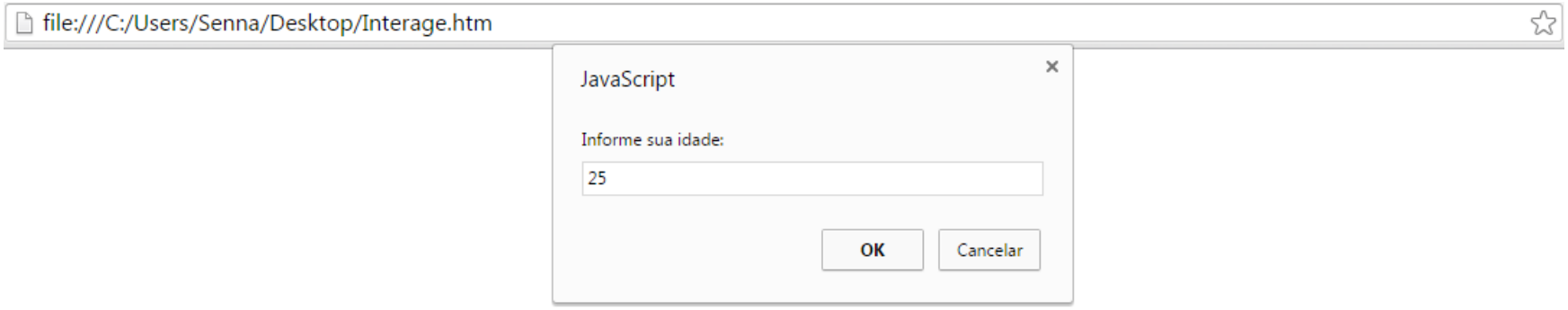
- Pedindo informações diretamente do teclado:

Interage.htm

```
1  <script language="javascript">
2  var a = prompt("Informe sua idade:")
3  alert("Voce tem "+a+" anos!")
4  </script>
5
```

GETTING STARTED

- Pedindo informações diretamente do teclado:

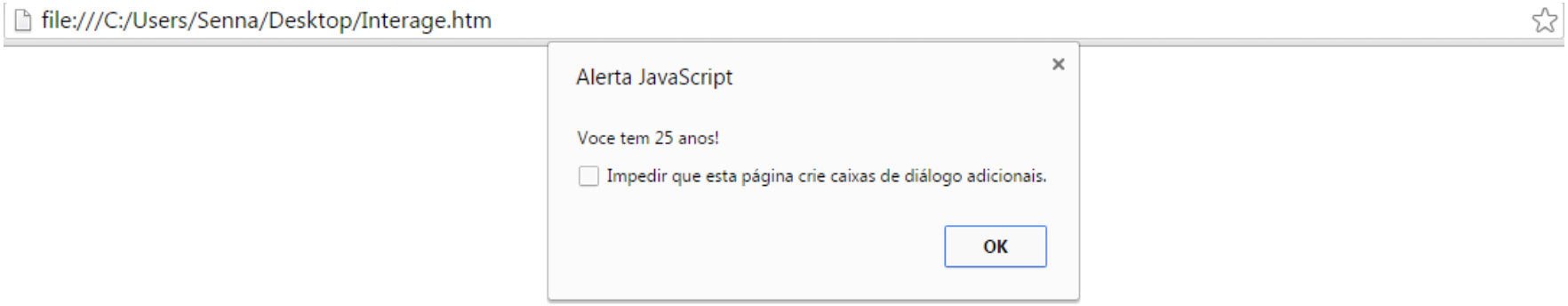


GETTING STARTED

- Alert: Comando padrão para exibição de avisos no navegador.
- Prompt: Comando que pede ao usuário para digitar alguma informação.

GETTING STARTED

- Agora, exibimos a informação fornecida pelo usuário:



GETTING STARTED

- Podemos também salvar o arquivo hello.js com o script:

hello.js

```
1  alert("Hello World")
```

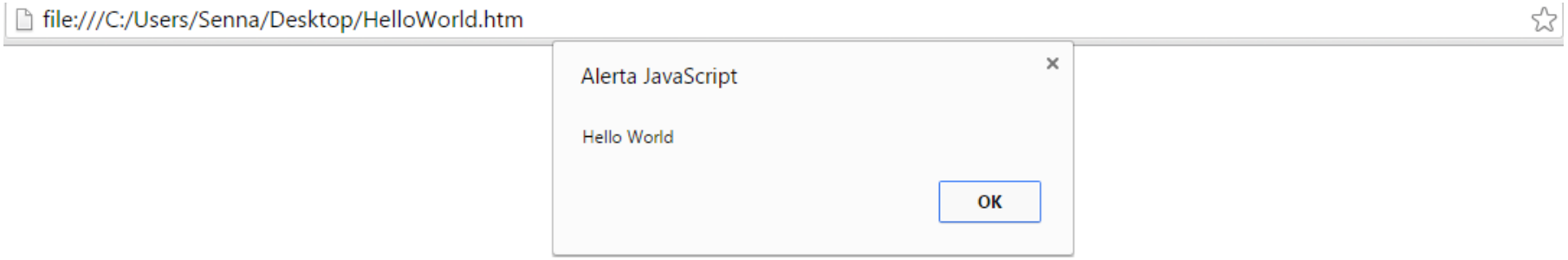
- E incluí-lo em uma página html para ser executado, assim:

HelloWorld.htm

```
1  <script type="text/javascript" src="hello.js">
2
3  </script>
```

GETTING STARTED

- E o resultado será:



Detalhes da Linguagem

- Tipos de dados:
  - string
  - number
  - boolean
  - object
  - Function

- Tipos de objetos:
  - Object
  - Date
  - Array

Detalhes da Linguagem

- Funções são cidadãos de primeira classe!
- Closures;
- Tipos *null* e *undefined;*

Detalhes da Linguagem

- Loop for:

```
for (var i =0; i < 4; i++){
    console.log(i);
}
```

```
> 
0
1
2
3
> 
```

Detalhes da Linguagem

- Loop for:

```
var time = {Ataque:"Joao",Meio: "Marcos",Defesa: "Carlos"}
for (var i in time){
    console.log(i+" "+time[i]);
}
```

```
>
Ataque Joao
Meio Marcos
Defesa Carlos
>
```

Detalhes da Linguagem

```
function main (){

    var time = {Ataque:"Joao",Meio: "Marcos",Defesa: "Carlos"}
   /* for (var i in time){
        console.log(i+" "+time[i]);
    }*/

    console.log(time);

    equipe = function(n1,n2,n3){
        this.Ataque = n1
        this.Meio= n2
        this.Defesa=n3
    }

    var brasil = new equipe("Joao","Marcos", "Carlos")

    console.log(brasil)

}
```

```
>
{ Ataque: 'Joao', Meio: 'Marcos', Defesa: 'Carlos' }
{ Ataque: 'Joao', Meio: 'Marcos', Defesa: 'Carlos' }
>
```

Detalhes da Linguagem

- Função construtor e prototype:

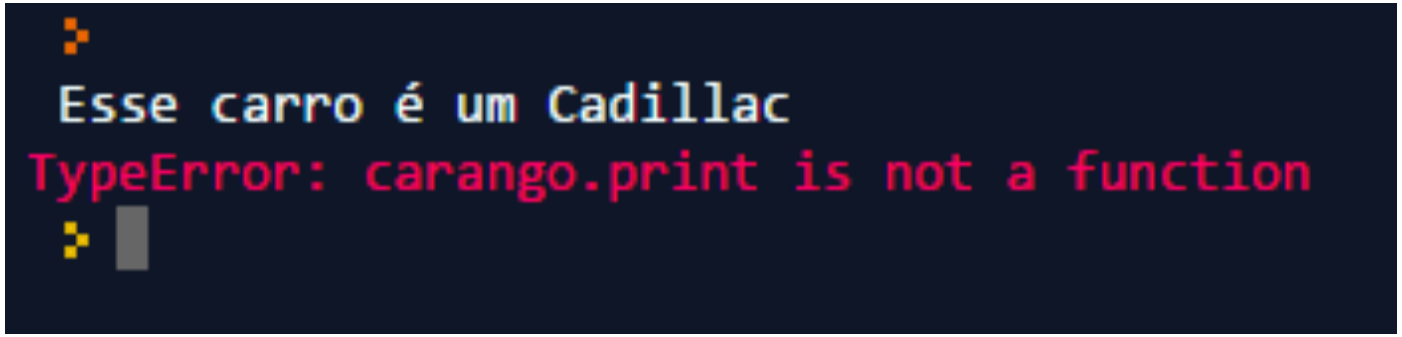
```
function Carro (marca, modelo){
    this.marca = marca;
    this.modelo = modelo;
}

var novo = new Carro("Ford", "Cadillac");
var carango = new Carro ("Fiat", "Palio");

novo.print = function(){
    console.log("Esse carro é um "+this.modelo);
}
novo.print()
carango.print();
```

Detalhes da Linguagem

- Função construtor e prototype:



Detalhes da Linguagem

- Função construtor e prototype:

```
var novo = new Carro("Ford", "Cadillac");
var carango = new Carro ("Fiat", "Palio");


Carro.prototype.print = function(){
    console.log("Esse carro é um "+this.modelo);
}
carango.print();
novo.print();
```

Detalhes da Linguagem

- Função construtor e prototype:

```
Esse carro é um Palio
Esse carro é um Cadillac
```