Programação para Web I

Inteligência na Marcação e Interação Web

Prof. Adriano Sepe

- Competência da Unidade: Inteligência na Marcação e Interação Web
- Resumo: inteligência na marcação do documento HTML, com elementos de semântica, e ainda através de programação em JavaScript.
- Palavras-chave: web; interação; semântica; html5; javascript; js; dom; documento object model;
- Teleaula nº: 2

Conteúdo

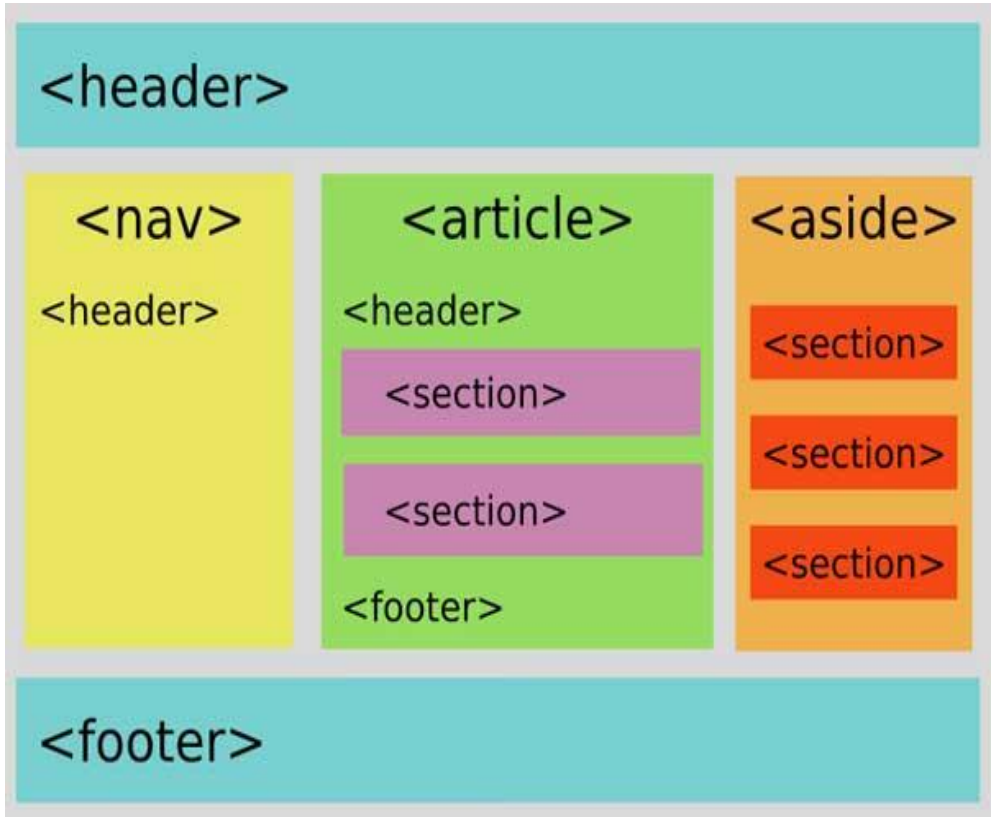
- Elementos de Semântica no HTML
- Programando no Navegador
- Avançando em JS
- SP 1
- Estruturas e Controle de Fluxo em JS
- Modularização em JS
- Documento Object Model (DOM)
- SP 2

Elementos de Semântica no HTML

Elementos Semânticos (HTML5)

- Possibilita a utilização de elementos de acordo com o propósito.
- Um elemento semântico expressa o seu propósito, para o desenvolvedor e também para o agente cliente (navegador).
- Elementos não semânticos: div e span.
- Elementos semânticos: p, ui, form e table.

HTML5 – Elementos Semânticos



Fonte: https://bit.ly/2xNS9ww

Tag <header>

Um elemento <header> representa um container para introdução de um conteúdo, ou ainda um conjunto de links de navegação.

Tag <footer>

Utilizamos o elemento <footer> para definir o conteúdo de rodapé, seja de um documento ou de uma seção (<section>).

É possível ter muitos <footer> em um documento.

Tag <main>

Utilizamos o elemento <main> para especificar o conteúdo principal de um documento. O conteúdo inserido neste elemento, deverá ser único no documento, e não poderá repetir através de outros documentos.

Ele deve ser declarado apenas uma vez em um documento.

Tag <nav>

Através do elemento <nav>, podemos especificar os links de navegação principal de um documento.

Nem todos os links serão disponibilizados nessa seção, mas será tratado como meio padrão de navegação.

Tag <section>

Utilizamos o elemento <section> para definir uma seção em nosso documento, o uso está livre para representar capítulos, subcapítulos, cabeçalhos ou rodapé.

Tag <article>

Utilizamos esse elemento para representar uma composição independente em um documento, página, aplicação ou site.

Tag <aside>

Este elemento tem por objetivo criar um grupo de conteúdo que está sempre relacionada a conteúdos subjacentes, como por exemplo uma lista de post mais populares, categorias de um blog ou então comentário recentes.

Programando no Navegador

Definindo JavaScript

- Devemos compreender JavaScript como a linguagem de programação da Web.
- Grande adoção através da Web.
- Linguagem de programação onipresente.
- Inteligência dos sites.
- Linguagem baseada em especificação
- Recursos básicos não são definidos

Para que JavaScript?

Quando desejamos criar algum nível de interatividade entre o documento e nosso usuário, isso fica sobe a responsabilidade da linguagem de programação JavaScript (MANZANO, 2008):

- Manipulação de HTML
- Manipulação de CSS
- Validação de Dados
- Armazenamento de Dados
- Reação a Eventos

Variáveis

Utilizando variáveis do tipo String

```
var categoria = "Categoria 1";
var endereco = "R. Um 123, São Paulo";
```

Fonte: O autor

Tipos de Dados

Abordaremos três tipos de dados, string, numérico e booleano.

```
var categoria = "Categoria 1";

var possuiFilhos = true;

var idade = 30;
var salario = 2034.56;
```

Fonte: O autor

Operadores aritméticos

Operador	Exemplo	Descrição
+	a = b + c	Soma das variáveis b e c e armazena o resultado na variável a
-	a = b - c	Subtração das variáveis b e c e armazena o resultado na variável a
*	a = b * c	Multiplica das variáveis b e c e armazena o resultado na variável a
/	a = b / c	Divisão das variáveis b por c e armazena o resultado na variável a
%	a = b % c	Resto da divisão de b por c e armazena o resultado na variável a

Fonte: O autor

Operadores Relacionais

Operador	Exemplo
==	a == b
===	a === b
!=	a != b
!==	a !== b
>	a > b
<	a < b
>=	a >= b
<=	a <= b

Fonte: O autor

Operadores Lógicos

Operador	Operação
&&	Lógico e (AND)
||	Lógico ou (OR)
!	Lógico não (NOT)

Fonte: O autor

Avançando em JS

Outros Assuntos

Operador de atribuição e Comentários

```
// Comentário de linha, tudo nesta linha será desconsiderado
var idade = 20;
/* Comentários de blocos são práticos
   Quando desejamos criar comentários
   Que ocuparam várias linhas */
var salario = 2034.56;
```

Fonte: O autor

Caixa de Diálogos

Caixa de diálogos são janelas no qual podemos utilizar com o propósito de interagir com o usuário, seja através de um alerta ou solicitando uma informação.

- Alerta
- Confirmação
- Solicitação
- * console.log

Inserindo JavaScript no HTML

Quando desejamos criar algum nível de interatividade em nossos documentos vamos construir isso através de scripts JavaScript, mas onde vamos escrever nossos códigos:

- Inline
- Incorporado
- Externo

Inserindo JavaScript no HTML - Inline

```
<body>
    <p onclick="alert('Você clicou no parágrafo')">Texto para exemplificar</p>
    <button onclick="if(confirm('Iniciar o processamento?')) alert('Processo
iniciado')">Iniciar</button>
</body>
```

Fonte: O autor

Inserindo JavaScript no HTML - Incorporando

```
<head>
    <title>JavaScript em Ação</title>
    <script type="text/javascript">
        // nosso script será escrito dentro deste elemento
    </script>
</head>
```

Fonte: O autor

Inserindo JavaScript no HTML - Externo

```
<head>
    <title>JavaScript em Ação</title>
     <script type="text/javascript" src="js/script1.js">
    </script>
</head>
```

Fonte: O autor

Desenvolvendo Algoritmos em JS

Situação Problema

Precisamos calcular a distância em metros, entre duas coordenadas (lat, lng). Essa distância será utilizada para determinar o combustível necessário para o deslocamento de um avião de pequeno porte. Neste momento, consideraremos apenas a distância em linha reta, com até 100 metros de imprecisão.

Resolvendo SP

VAMOS RESOLVER!

Vamos Interagir!

Estruturas e Controle de Fluxo em JS

Vetores

Um array (vetor) em JavaScript é do tipo object;

Pode guardar tipos diferentes de valores;

Informar o número de posições é opcional;

Para criar um vetor em JavaScript pode ser utilizadas duas sintaxes:

```
var vetor = new Array([numero opcional]);
var vetor  = [];
```

Vetores

Inicialização:

```
var vetor = new Array(“Joao”, ”Maria”, 100, 200, true);
var vetor  = [“Joao”, ”Maria”, 100, 200, true];
```

Manipulando:

```
alert(vetor[0]); -> João
alert(vetor[2]); -> 100
```

Propriedade **length**

Estruturas de Programação - IF

if

```
var idade = 18;
if(idade >= 18) {
    alert("Você já tem maioridade penal!");
}
```

Fonte: O autor

Estruturas de Programação – IF-ELSE

If-else

```
var idade = 18;
if(idade >= 18) {
    alert("Você já tem maioridade penal!");
} else {
    alert("Você ainda não  idade tem responsabilidade
criminal!");
}
```

Fonte: O autor

Estruturas de Programação – FOR

for

```
for(contador = 0; contador < 5; contador++) {
    console.log("O valor do contador é: " + contador);
}
```

Fonte: O autor

Estruturas de Programação – WHILE

while

```
var contador = 10;
while (contador > 0) {
    alert("O valor do contador é " + contador);
    contador = contador - 1;
}
```

Fonte: O autor

Estruturas de Programação – DO-WHILE

do-while

```
do {
    var info = prompt("Pra sair digite o texto sair!");
} while (info != "sair");
```

Fonte: O autor

Modularização em JS

Funções

- São blocos de código que serão executados apenas quando chamados ou por um evento;
- São definidas dentro de um bloco <script> ou em um arquivo externo;
- Podem receber parâmetros;
- Podem ter um retorno;

Declarações de Função

Podemos ainda desenvolver nossos scripts utilizando o modelo de desenvolvimento modular através da organização de nossos códigos em função.

```
function nomeFunção([arg1, arg2, ..., argn]) {
    // construção de nosso script
}
```

Fonte: O autor

JS: Funções

Sintaxe:

```
function nome(var1,var2) {
    // esse código será executado apenas
    // quando a função for chamada.
}
```

Os parâmetros var1 e var2 são variáveis ou valores passados para função;

Funções sem parâmetro devem incluir os parênteses;

JS: Funções com Retorno

A palavra chave return é utilizada para especificar um valor que a função irá retornar;

Toda função que irá retornar um valor deve utilizar a palavra chave return;

Exemplo:

```
function produto(a,b) {
    return a*b;
}
```

Ciclo de vida de variáveis

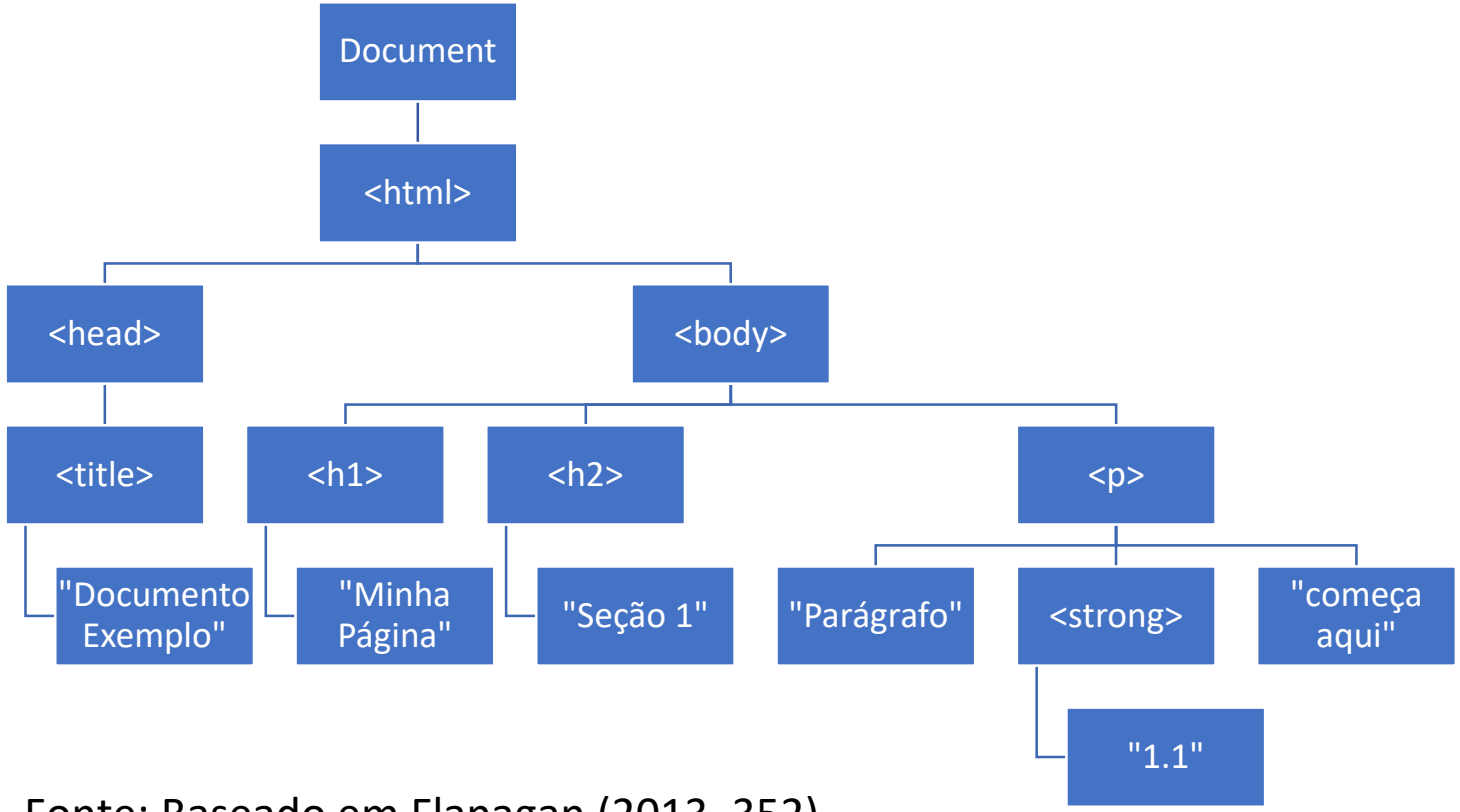
- Variáveis declaradas dentro de funções só podem sem acessadas localmente;
- Variáveis com mesmo nome podem ser declaradas em funções diferentes;
- Após a função ser finalizada, as variáveis são destruídas;
- Se uma variável é declarada fora de uma função ela pode ser acessada por todas as funções;

Documento Object Model (DOM)

DOM (Documento Object Model)

Document Object Model, ou DOM, é a API fundamental para representar e manipular o conteúdo de documentos HTML e XML. A API não é especialmente complicada, mas existem vários detalhes de arquitetura que precisam ser entendidos (FLANAGAN, 2016).

DOM (Documento Object Model)



Fonte: Baseado em Flanagan (2013, 352)

JS: DOM

- Cada página carregada no navegador cria um objeto chamado “document”;
- O objeto document permite o acesso a todos os elementos HTML da página;
- Contém propriedades e métodos para capturar elementos HTML;
- Esses elementos podem ser manipulados se necessário;

JS: DOM

- Cada elemento é considerado um nó;
- Para acessar os nós de um documento, utilizamos a propriedade “childNodes” do objeto document;
- A tag HTML é o nó raiz de um documento;
- O nó HTML encontra-se na primeira posição da coleção de filhos (childNodes);

JS: DOM

- Um nó pode conter filhos;
- Todos os elementos dentro de um nó são considerados seus filhos;
- Para acessar o número de filhos de um nó basta utilizar a propriedade length da coleção childNodes;
- Também é possível descobrir o nome de um nó utilizando a propriedade nodeName;

JS: DOM

- Textos são considerados nós;
- O valor do nodeName de um elemento do tipo texto é #text;
- Para obter o texto de um nó do tipo texto, acessamos a prop. data;
- Alguns navegadores podem considerar uma quebra de linha como um elemento do tipo texto;
- Um nó #text contém uma propriedade length e retorna o número de caracteres do nó.

JS: getElementById

- Retorna uma referência de um elemento através de seu id;
- Caso não seja encontrado será retornado null;

Exemplo:

```
var obj = document.getElementById("txtNome");
```

JS: Obtendo Elementos

- document. getElementsByName
- document. getElementsByTagName
- document.getElementsByClassName

Protótipo de um CHAT HTML

Situação Problema

Precisamos desenvolver um protótipo de um CHAT HTML, para demonstrar como seria o processo de interação do usuário, e manipulação do DOM através da linguagem JavaScript.

Resolvendo SP

VAMOS RESOLVER!

Vamos Interagir!

Recapitulando

Ao final dessa aula vimos:

- Elementos de Semântica no HTML
- Programando no Navegador
- Avançando em JS
- Estruturas e Controle de Fluxo em JS
- Modularização em JS
- Documento Object Model (DOM)