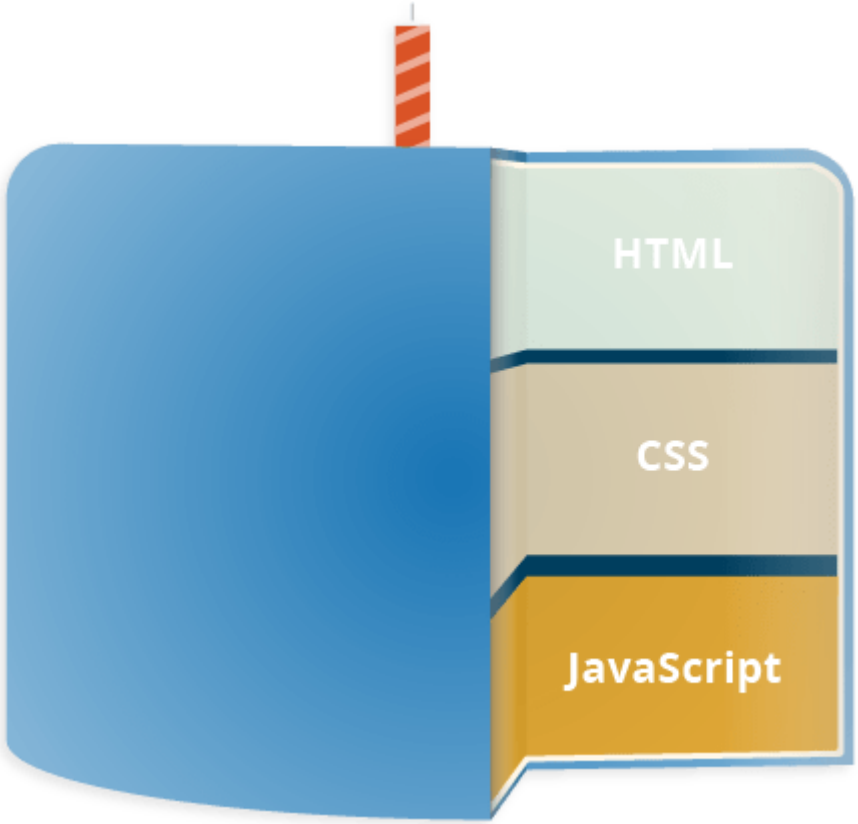
Curso de JavaScript

escola avançada

Sumário

JavaScript é uma linguagem de programação que permite a você implementar itens complexos em páginas web — toda vez que uma página da web faz mais do que simplesmente mostrar a você informação estática — mostrando conteúdo que se atualiza em um intervalo de tempo, mapas interativos ou gráficos 2D/3D animados, etc. — você pode apostar que o JavaScript provavelmente está envolvido. É a terceira camada do bolo das tecnologias padrões da web, duas das quais (HTML e CSS) nós falamos com muito mais detalhes em outras partes da Área de Aprendizado.



HTML é a linguagem de marcação que nós usamos para estruturar e dar significado para o nosso conteúdo web. Por exemplo, definindo parágrafos, cabeçalhos, tabelas de conteúdo, ou inserindo imagens e vídeos na página.

CSS é uma linguagem de regras de estilo que nós usamos para aplicar estilo ao nosso conteúdo HTML. Por exemplo, definindo cores de fundo e fontes, e posicionando nosso conteúdo em múltiplas colunas.

JavaScript é uma linguagem de programação que permite a você criar conteúdo que se atualiza dinamicamente, controlar múltimídias, imagens animadas, e tudo o mais que há de intessante. Ok, não tudo, mas é maravilhoso o que você pode efetuar com algumas linhas de código JavaScript.

As três camadas ficam muito bem uma em cima da outra. Vamos exemplificar com um simples bloco de texto. Nós podemos marcá-lo usando HTML para dar estrutura e propósito:

```
<p>Jogador 1: Chris</p>
```

Então o que ele pode realmente fazer?

O núcleo da linguagem JavaScript consiste em alguns benefícios comuns da programação que permite a você fazer coisas como:

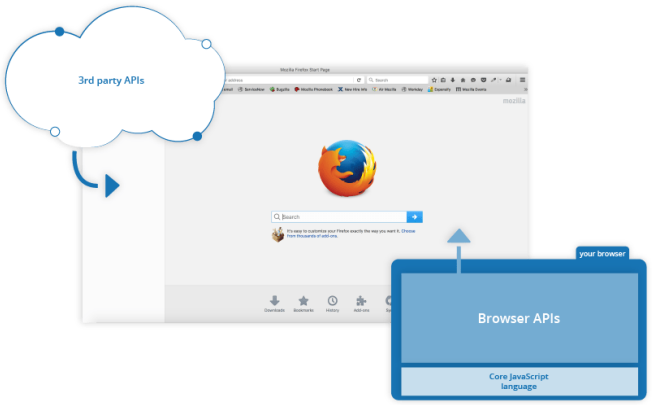
Armazenar conteúdo útil em variáveis. No exemplo acima, a propósito, nós pedimos que um novo nome seja inserido e armazenamos o nome em uma variável chamada nome.

Operações com pedaços de texto (conhecidos como "strings" em programação). No exemplo acima, nós pegamos a string "Jogador 1: " e concatenamos (juntamos) com a variável nome para criar o texto completo "Jogador 1: Chris".

Executar o código em resposta a determinados eventos que ocorrem em uma página da Web. Nós usamos o click (en-US) no nosso exemplo acima para que quando clicassem no botão, rodasse o código que atualiza o texto.

E muito mais!

O que é ainda mais empolgante é a funcionalidade construída no topo do núcleo da linguagem JavaScript. As APIs (Application Programming Interfaces - Interface de Programação de Aplicativos) proveem a você superpoderes extras para usar no seu código JavaScript.



APIs são conjuntos prontos de blocos de construção de código que permitem que um desenvolvedor implemente programas que seriam difíceis ou impossíveis de implementar. Eles fazem o mesmo para a programação que os kits de móveis prontos para a construção de casas - é muito mais fácil pegar os painéis prontos e parafusá-los para formar uma estante de livros do que para elaborar o design, sair e encontrar a madeira, cortar todos os painéis no tamanho e formato certos, encontrar os parafusos de tamanho correto e depois montá-los para formar uma estante de livros.

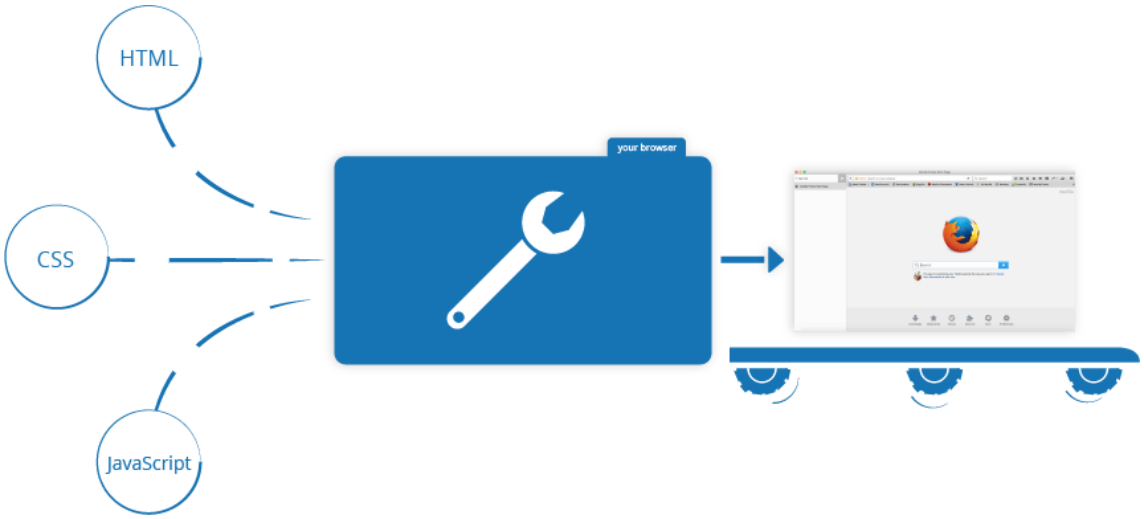
Elas geralmente se dividem em duas categorias.

O que JavaScript está fazendo na sua página web?

Aqui nós vamos realmente começar a ver algum código, e enquanto fazemos isso vamos explorar o que realmente acontece quando você roda algum código JavaScript na sua página.

Vamos recaptular brevemente a história do que acontece quando você carrega uma página web em um navegador (falamos sobre isso no nosso artigo Como o CSS funciona). Quando você carrega uma página web no seu navegador, você está executando seu código (o HTML, CSS e

JavaScript) dentro de um ambiente de execução (a guia do navegador). Isso é como uma fábrica que pega a matéria prima (o código) e transforma em um produto (a página web).



Um uso muito comum do JavaScript é modificar dinamicamente HTML e CSS para atualizar uma interface do usuário, por meio da API do Document Object Model (conforme mencionado acima). Observe que o código em seus documentos web geralmente é carregado e executado na ordem em que aparece na página. Se o JavaScript carregar e tentar executar antes do carregamento do HTML e CSS afetado, poderão ocorrer erros. Você aprenderá maneiras de contornar isso mais adiante neste artigo, na seção Estratégias de carregamento de scripts .

Segurança do navegador

Cada guia do navegador tem seu próprio espaço para executar código (esses espaços são chamados de "ambientes de execução", em termos técnicos) — isso significa que na maioria dos casos o código em cada guia está sendo executado separadamente, e o código em uma guia não pode afetar diretamente o código de outra guia — ou de outro website. Isso é uma boa medida de segurança — se esse não fosse o caso, então hackers poderiam começar a escrever código para roubar informações de outros websites, e fazer outras coisas más.

Ordem de execução do JavaScript

Quando o navegador encontra um bloco de código JavaScript, ele geralmente executa na ordem, de cima para baixo. Isso significa que você precisa ter cuidado com a ordem na qual você coloca as coisas. Por exemplo, vamos voltar ao bloco JavaScript que nós vimos no primeiro exemplo:

```
const para = document.querySelector('p');

para.addEventListener('click', atualizarNome);

function atualizarNome() {
  ;et nome = prompt('Informe um novo nome:');
  para.textContent = 'Jogador 1: ' + nome;
}
```

Aqui nós estamos selecionando um parágrafo (linha 1) e anexando a ele um event listener (linha 3). Então, quando o parágrafo recebe um clique, o bloco de código atualizarNome() (linhas 5 a 8) é executado. O bloco de código atualizarNome()(esses tipos de bloco de código reutilizáveis são chamados "funções") pede ao usuário que informe um novo nome, e então insere esse nome no parágrafo, atualizando-o.

Se você inverte a ordem das duas primeiras linhas de código, ele não fucionaria — em vez disso, você receberia um erro no console do navegador — TypeError: para is undefined. Isso significa que o objeto para não existe ainda, então nós não podemos adicionar um event listener a ele.

Como você adiciona JavaScript na sua página?

O JavaScript é inserido na sua página de uma maneira similar ao CSS. Enquanto o CSS usa o elemento <link> para aplicar folhas de estilo externas e o elemento <style> para aplicar folhas de estilo internas, o JavaScript só precisa de um amigo no mundo do HTML — o elemento <script>. Vamos aprender como funciona.

JavaScript interno

Antes de tudo, faça uma cópia local do nosso arquivo de exemplo aplicando-javascript.html. Salve-o em alguma pasta, de uma forma sensata.

Abra o arquivo no seu navegador web e no seu editor de texto. Você verá que o HTML cria uma simples página web contendo um botão clicável.

Agora, vá até o seu editor de texto e adicione o código a seguir antes da tag de fechamento </body>:

```
<script>

  // O JavaScript fica aqui

</script>
```

Agora nós vamos adicionar um pouco de JavaScript dentro do nosso elemento <script> para que a página faça algo mais interessante — adicione o seguinte código abaixo da linha "// O JavaScript fica aqui":

```
function criarParagrafo() {
  let para = document.createElement('p');
  para.textContent = 'Você clicou no botão!';
  document.body.appendChild(para);
}

const botoes = document.querySelectorAll('button');

for(var i = 0; i < botoes.length ; i++) {
  botoes[i].addEventListener('click', criarParagrafo);
}
```

Salve seu arquivo e recarregue a página — agora você deveria ver que quando você clique no botão, um novo parágrafo é gerado e colocado logo abaixo.

Setinterval

O método setInterval() oferecido das interfaces Window e Worker, repetem chamadas de funções or executam trechos de código, com um tempo de espera fixo entre cada chamada. Isso retorna um ID único para o intervalo, podendo remove-lo mais tarde apenas o chamando clearInterval() (en-US). Este metodo é definido pelo mixin WindowOrWorkerGlobalScope.

Sintaxe.

```
var intervalID = scope.setInterval(func, delay[, param1, param2, ...]);
var intervalID = scope.setInterval(code, delay);
```

Exemplo 1: Sintaxe básica

O seguinte exemplo mostra a sintaxe básica do setInterval()

```
var intervalID = window.setInterval(myCallback, 500);

function myCallback() {
  // Your code here
}
```

Exemplo 2: Alternando duas cores

O seguinte exemplo chama a função flashtext() uma vez por segundo até o botão de parar ser pressionado.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8" />
  <title>setInterval/clearInterval example</title>

  <script>
    var nIntervId;

    function changeColor() {
      nIntervId = setInterval(flashText, 1000);
    }

    function flashText() {
      var oElem = document.getElementById('my_box');
      oElem.style.color = oElem.style.color == 'red' ? 'blue' : 'red';
      // oElem.style.color == 'red' ? 'blue' : 'red' is a ternary operator.
    }

    function stopTextColor() {
      clearInterval(nIntervId);
    }
  </script>
</head>

<body onload="changeColor();">
  <div id="my_box">
    <p>Hello World</p>
  </div>

  <button onclick="stopTextColor();">Stop</button>
</body>
</html>
```

Get Element By Id

Os novatos devem notar que a caixa de 'Id' no nome deste método deve estar correta para o código da função - 'getElementByID não funciona, por mais natural que possa parecer.

Se não existe um elemento com o id fornecido, esta função retorna null. Note que o parâmetro ID diferência maiúsculas e minúsculas. Assim document.getElementById("Main") retornará null ao invés do elemento , devido a "M" e "m" serem diferentes para o objetivo deste método.

Elementos que não estão no documento não são procurados por getElementById. Quando criar um elemento e atribuir um ID ao mesmo, você deve inserir o elemento na árvore do documento com insertBefore ou método similar antes que você possa acessá-lo com getElementById:

```
var elemento = document.createElement("div");
elemento.id = 'testqq';
var el = document.getElementById('testqq'); // el será null!
```

Documentos não-HTML. A implementação do DOM deve ter informações que diz quais atributos são do tipo ID. Atributos com o nome "id" não são do tipo ID a menos que assim sejam definidos nos documentos DTD. O atributo id é definido para ser um tipo ID em casos comuns de XHTML, XUL, e outros. Implementações que não reconhecem se os atributos são do tipo ID, ou não são esperados retornam null.

Atividade

Crie uma calculadora de acordo com o que foi ensinado durante a aula

Tire uma foto ou filme e marque nosso instagram @escolaavancada