Log in

Paid Courses

Próximo >

Tela JS

Localização JS

História JS

JS Timing

web

Cookies JS

JS Navigator

Alerta pop-up JS

JS Web APIs

API Web Worker

API Web Geolocation

API Web Fetch

JS AJAX

AJAX Intro

AJAX XMLHttp

Dadida A IAV

Introdução à API da Web

API de histórico da web

API de formulários da web

API de armazenamento da

References ▼

Exercises ▼

API Web Workers

Anterior

Um web worker é um JavaScript em execução em segundo plano, sem afetar o desempenho da página.

O que é um Web Worker?

Ao executar scripts em uma página HTML, a página deixa de responder até que o script seja concluído.

Um web worker é um JavaScript executado em segundo plano, independentemente de outros scripts, sem afetar o desempenho da página. Você pode continuar a fazer o que quiser: clicar, selecionar coisas, etc., enquanto o web worker é executado em segundo plano.

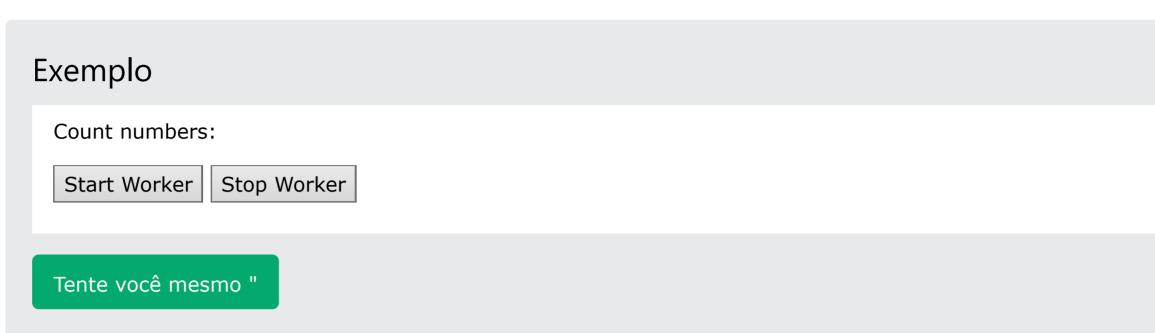
Suporte de navegador

Os números na tabela especificam as primeiras versões de navegador que oferecem suporte total para Web Workers:

	e			O
Chrome 4	IE 10	Firefox 3.5	Safari 4	Opera 11.5
Jan 2010	Sep 2012	Jun 2009	Jun 2009	Jun 2011

Exemplo de Web Workers

O exemplo abaixo cria um web worker simples que conta números em segundo plano:



Verifique o suporte do Web Worker

Antes de criar um web worker, verifique se o navegador do usuário é compatível:

```
if (typeof(Worker) !== "undefined") {
  // Yes! Web worker support!
  // Some code....
} else {
  // Sorry! No Web Worker support..
```

Criar um arquivo de trabalhador da web

Agora, vamos criar nosso web worker em um JavaScript externo.

Aqui, criamos um script que conta. O script é armazenado no arquivo "demo_workers.js":

```
let i = 0;
function timedCount() {
  i ++;
  postMessage(i);
  setTimeout("timedCount()",500);
timedCount();
```

The important part of the code above is the postMessage() method - which is used to post a message back to the HTML page.

Note: Normally web workers are not used for such simple scripts, but for more CPU intensive tasks.

Create a Web Worker Object

Now that we have the web worker file, we need to call it from an HTML page.

The following lines checks if the worker already exists, if not - it creates a new web worker object and runs the code in "demo_workers.js":

```
if (typeof(w) == "undefined") {
  w = new Worker("demo_workers.js");
```

Then we can send and receive messages from the web worker.

Add an "onmessage" event listener to the web worker.

```
w.onmessage = function(event){
   document.getElementById("result").innerHTML = event.data;
 };
When the web worker posts a message, the code within the event listener is executed. The data from the web
```

worker is stored in event.data.

Quando um objeto de trabalho da web é criado, ele continua a escutar mensagens (mesmo depois que o script externo é concluído) até que seja encerrado.

w.terminate();

w = undefined;

Exemplo

Terminate a Web Worker

Para encerrar um web worker e liberar recursos de navegador / computador, use o terminate() método:

```
Reutilizar o Web Worker
```

Se você definir a variável de trabalho como indefinida, depois de encerrada, você pode reutilizar o código:

Código de exemplo de trabalhador da Web completo

Já vimos o código do Worker no arquivo .js. Abaixo está o código para a página HTML:

```
<!DOCTYPE html>
  <html>
  <body>
 Count numbers: <output id="result"></output>
 <button onclick="startWorker()">Start Worker</button>
 <button onclick="stopWorker()">Stop Worker</button>
  <script>
 let w;
 function startWorker() {
   if (typeof(w) == "undefined") {
     w = new Worker("demo_workers.js");
   w.onmessage = function(event) {
     document.getElementById("result").innerHTML = event.data;
   };
 function stopWorker() {
   w.terminate();
   w = undefined;
 </script>
  </body>
  </html>
  Tente você mesmo "
Web Workers e o DOM
```

Como os web workers estão em arquivos externos, eles não têm acesso aos seguintes objetos JavaScript:

• O objeto da janela O objeto do documento

- O objeto pai
- Anterior

Próximo >

Fórum Reportar erro Cerca de Comprar **Top Tutorials Top References Top Examples** Web Courses **HTML Tutorial** HTML Reference HTML Examples **HTML Course CSS Tutorial CSS** Reference CSS Examples CSS Course JavaScript Tutorial JavaScript Reference JavaScript Examples JavaScript Course How To Tutorial How To Examples Front End Course SQL Reference **SQL** Tutorial Python Reference SQL Course SQL Examples Python Tutorial W3.CSS Reference Python Course Python Examples W3.CSS Tutorial **Bootstrap Reference** W3.CSS Examples PHP Course PHP Reference jQuery Course **Bootstrap Tutorial** Bootstrap Examples PHP Tutorial HTML Colors Java Course PHP Examples Java Tutorial Java Reference C++ Course Java Examples C++ Tutorial Angular Reference XML Examples C# Course jQuery Tutorial jQuery Reference jQuery Examples XML Course Get Certified » W3Schools é otimizado para aprendizagem e treinamento. Os exemplos podem ser simplificados para melhorar a leitura e o aprendizado. Tutoriais, referências e exemplos são constantemente revisados para evitar erros, mas não podemos garantir a correção total de todo o conteúdo. Ao usar o W3Schools, você concorda em ter





COLOR PICKER



COMO NÓS





JOGO DE

