

Sintaxe assíncrona

A palavra-chave `async` antes de uma função faz com que a função retorne uma promessa:

Exemplo

```
async function myFunction() {
  return "Hello";
}
```

É o mesmo que:

```
async function myFunction() {
  return Promise.resolve("Hello");
}
```

Aqui está como usar a promessa:

```
myFunction().then(
  function(value) { /* code if successful */ },
  function(error) { /* code if some error */ }
);
```

Exemplo

```
async function myFunction() {
  return "Hello";
}
myFunction().then(
  function(value) {myDisplayer(value);},
  function(error) {myDisplayer(error);}
);
```

Tente você mesmo "

Ou mais simples, já que você espera um valor normal (uma resposta normal, não um erro):

Exemplo

```
async function myFunction() {
  return "Hello";
}
myFunction().then(
  function(value) {myDisplayer(value);}
);
```

Tente você mesmo "

Await Syntax

A palavra-chave `await` antes de uma função faz com que ela espere por uma promessa:

```
let value = await promise;
```

A `await` palavra-chave só pode ser usada dentro de uma `async` função.

Exemplo

Vamos devagar e aprendamos como usá-lo.

Sintaxe Básica

```
async function myDisplay() {
  let myPromise = new Promise(function(myResolve, myReject) {
    myResolve("I love You !!");
  });
  document.getElementById("demo").innerHTML = await myPromise;
}

myDisplay();
```

Tente você mesmo "

Esperando por um tempo limite

```
async function myDisplay() {
  let myPromise = new Promise(function(myResolve, myReject) {
    setTimeout(function() { myResolve("I love You !!"); }, 3000);
  });
  document.getElementById("demo").innerHTML = await myPromise;
}

myDisplay();
```

Tente você mesmo "

Esperando por um Arquivo

```
async function getFile() {
  let myPromise = new Promise(function(myResolve, myReject) {
    let req = new XMLHttpRequest();
    req.open('GET', "mycar.html");
    req.onload = function() {
      if (req.status == 200) {myResolve(req.response);}
      else {myResolve("File not Found");}
    };
    req.send();
  });
  document.getElementById("demo").innerHTML = await myPromise;
}

getFile();
```

Tente você mesmo "

Suporte de navegador

ECMAScript 2017 introduziu as palavras `async` - chave JavaScript e `await` .

A tabela a seguir define a primeira versão do navegador com suporte total para ambos:

				
Chrome 55	Edge 15	Firefox 52	Safari 11	Opera 42
Dec, 2016	Apr, 2017	Mar, 2017	Sep, 2017	Dec, 2016

< Anterior

Próximo >

PROPAGANDA

Ensino 100% online Pergunte sobre bolsas de estudo e programas de formação

FUNIBER

Abriu

Reportar erro

Fórum

Cerca de

Comprar

Principais tutoriais

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples

HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Web Courses

HTML Course
CSS Course
JavaScript Course
Front End Course
SQL Course
Python Course
PHP Course
jQuery Course
Java Course
C++ Course
C# Course
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 lido e aceite nossos termos de uso , cookies e política de privacidade .

Copyright 1999-2021 de Refsnes Data. Todos os direitos reservados.
W3Schools é desenvolvido por W3.CSS .

