JavaScript Async

"async and await make promises easier to write"

async makes a function return a Promise

await makes a function wait for a Promise

Async Syntax

The keyword `async` before a function makes the function return a promise:

Example

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

Is the same as:

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

Here is how to use the Promise:

```
function(error) { /* code if some error */ }
);
```

Example

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

Try it Yourself »

Or simpler, since you expect a normal value (a normal response, not an error):

Example

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

Try it Yourself »

Await Syntax

The `await` keyword can only be used inside an `async` function.

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

Example

Let's go slowly and learn how to use it.

Basic Syntax

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

myDisplay();
```

Try it Yourself »

The two arguments (resolve and reject) are pre-defined by JavaScript.

We will not create them, but call one of them when the executor function is ready.

Very often we will not need a reject function.

Example without reject

```
async function myDisplay() {
  let myPromise = new Promise(function(resolve) {
```

```
myDisplay();
```

Try it Yourself »

Waiting for a Timeout

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

myDisplay();
```

Try it Yourself »

Waiting for a File

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

Browser Support

ECMAScript 2017 introduced the JavaScript keywords `async` and `await` .

The following table defines the first browser version with full support for both:

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

❮ Previous   Next ❯

Track your progress - it's free!   Sign Up   Log in

CHECK IT OUT!

W3
schools

COLOR PICKER

W3
schools

PLUS   SPACES

GET CERTIFIED   FOR TEACHERS

FOR BUSINESS   CONTACT US

Top Tutorials

HTML Tutorial
CSS 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

Get Certified

HTML Certificate
CSS Certificate
JavaScript Certificate
Front End Certificate
SQL Certificate
Python Certificate
PHP Certificate
jQuery Certificate
Java Certificate
C++ Certificate
C# Certificate
XML Certificate

FORUM   ABOUT   ACADEMY