



ABOUT

DOWNLOAD

GUIDES

BLOG

DOCS

CERTIFICATION



## Getting Started

[Introduction to Node.js](#)[How to install Node.js](#)[How much JavaScript do you need to know to use Node.js?](#)[Differences between Node.js and the Browser](#)[The V8 JavaScript Engine](#)[An introduction to the NPM package manager](#)[ECMAScript 2015 \(ES6\) and beyond](#)[Node.js, the difference between development and production](#)[Node.js with TypeScript](#)[Node.js with WebAssembly](#)

## Asynchronous Work

[Asynchronous flow control](#)[Overview of Blocking vs Non-Blocking](#)[JavaScript Asynchronous Programming and Callbacks](#)[Discover JavaScript Timers](#)[Understanding process.nextTick\(\)](#)

# Reading files with Node.js

The simplest way to read a file in Node.js is to use the `fs.readFile()` method, passing it the file path, encoding and a callback function that will be called with the file data (and the error):

```
const fs = require('node:fs');

fs.readFile('/Users/joe/test.txt', 'utf8', (err, data) => {
  if (err) {
    console.error(err);
    return;
  }
  console.log(data);
});
```

Alternatively, you can use the synchronous version

`fs.readFileSync()`:

```
const fs = require('node:fs');

try {
  const data = fs.readFileSync('/Users/joe/test.txt', 'utf8');
  console.log(data);
} catch (err) {
  console.error(err);
}
```

You can also use the promise-based

`fsPromises.readFile()` method offered by the `fs/promises` module:

Understanding  
setImmediate()

The Node.js Event  
emitter

## Manipulating Files

Node.js file stats

Node.js File Paths

Working with file  
descriptors in Node.js

Reading files with  
Node.js

Writing files with  
Node.js

Working with folders  
in Node.js

## Command Line

Run Node.js scripts  
from the command  
line

How to read  
environment variables  
from Node.js

How to use the  
Node.js REPL

Output to the  
command line using  
Node.js

Accept input from the  
command line in  
Node.js

```
const fs = require('node:fs/promises');

async function example() {
  try {
    const data = await fs.readFile('/Users/joe');
    console.log(data);
  } catch (err) {
    console.log(err);
  }
}

example();
```

All three of `fs.readFile()`, `fs.readFileSync()` and `fsPromises.readFile()` read the full content of the file in memory before returning the data.

This means that big files are going to have a major impact on your memory consumption and speed of execution of the program.

In this case, a better option is to read the file content using streams.

Copyright OpenJS Foundation and Node.js contributors. All rights reserved. The OpenJS Foundation has registered trademarks and uses trademarks. For a list of trademarks of the OpenJS Foundation, please see our [Trademark Policy](#) and [Trademark List](#). Trademarks and logos not indicated on the list of OpenJS Foundation trademarks are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by them.

[The OpenJS Foundation](#) | [Trademark Policy](#) | [Privacy Policy](#) | [Code of Conduct](#) | [Security Reporting](#)