



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\(\)](#)

# Run Node.js scripts from the command line

The usual way to run a Node.js program is to run the globally available `node` command (once you install Node.js) and pass the name of the file you want to execute.

If your main Node.js application file is `app.js`, you can call it by typing:

```
node app.js
```

Above, you are explicitly telling the shell to run your script with `node`. You can also embed this information into your JavaScript file with a "shebang" line. The "shebang" is the first line in the file, and tells the OS which interpreter to use for running the script. Below is the first line of JavaScript:

```
#!/usr/bin/node
```

Above, we are explicitly giving the absolute path of interpreter. Not all operating systems have `node` in the bin folder, but all should have `env`. You can tell the OS to run `env` with `node` as parameter:

```
#!/usr/bin/env node
```

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

```
// your javascript code
```

To use a shebang, your file should have executable permission. You can give `app.js` the executable permission by running:

```
chmod u+x app.js
```

While running the command, make sure you are in the same directory which contains the `app.js` file.

## Pass string as argument to `node` instead of file path

To execute a string as argument you can use `-e` , `--eval "script"` . Evaluate the following argument as JavaScript. The modules which are predefined in the REPL can also be used in script.

On Windows, using `cmd.exe` a single quote will not work correctly because it only recognizes double `"` for quoting. In Powershell or Git bash, both `'` and `"` are usable.

```
node -e "console.log(123)"
```

## Restart the application automatically

As of nodejs V16, there is a built-in option to automatically restart the application when a file changes. This is useful for development purposes.

To use this feature, you need to pass the `--watch` flag to nodejs.

```
node --watch app.js
```

So when you change the file, the application will restart automatically. Read the `--watch` [flag documentation](#).

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](#)