Node.js File System Module

Node.js as a File Server

The Node.js file system module allows you to work with the file system on your computer.

To include the File System module, use the `require()` method:

```
var fs = require('fs');
```

Common use for the File System module:

- Read files
- Create files
- Update files
- Delete files
- Rename files

Read Files

The `fs.readFile()` method is used to read files on your computer.

Assume we have the following HTML file (located in the same folder as Node.js):

demofile1.html

```
<p>My paragraph.</p>
</body>
</html>
```

Create a Node.js file that reads the HTML file, and return the content:

Example

Get your own Node.js Server

```
var http = require('http');
var fs = require('fs');
http.createServer(function (req, res) {
  fs.readFile('demofile1.html', function(err, data) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write(data);
    return res.end();
  });
}).listen(8080);
```

Run example »

Save the code above in a file called "demo_readfile.js", and initiate the file:

Initiate demo_readfile.js:

```
C:\Users\Your Name>node demo_readfile.js
```

If you have followed the same steps on your computer, you will see the same result as the example: http://localhost:8080

Create Files

The File System module has methods for creating new files:

The `fs.appendFile()` method appends specified content to a file. If the file does not exist, the file will be created:

Example

Create a new file using the appendFile() method:

```
var fs = require('fs');

fs.appendFile('mynewfile1.txt', 'Hello content!', function (err) {
  if (err) throw err;
  console.log('Saved!');
});
```

Run example »

The `fs.open()` method takes a "flag" as the second argument, if the flag is "w" for "writing", the specified file is opened for writing. If the file does not exist, an empty file is created:

Example

Create a new, empty file using the open() method:

```
var fs = require('fs');

fs.open('mynewfile2.txt', 'w', function (err, file) {
  if (err) throw err;
  console.log('Saved!');
});
```

Run example »

The `fs.writeFile()` method replaces the specified file and content if it exists. If the file does not exist, a new file, containing the specified content, will be created:

```
var fs = require('fs');

fs.writeFile('mynewfile3.txt', 'Hello content!', function (err) {
  if (err) throw err;
  console.log('Saved!');
});
```

Run example »

Update Files

The File System module has methods for updating files:

- `fs.appendFile()`
- `fs.writeFile()`

The `fs.appendFile()` method appends the specified content at the end of the specified file:

Example

Append "This is my text." to the end of the file "mynewfile1.txt":

```
var fs = require('fs');

fs.appendFile('mynewfile1.txt', ' This is my text.', function (err) {
  if (err) throw err;
  console.log('Updated!');
});
```

Run example »

The `fs.writeFile()` method replaces the specified file and content:

```
var fs = require('fs');

fs.writeFile('mynewfile3.txt', 'This is my text', function (err) {
  if (err) throw err;
  console.log('Replaced!');
});
```

Run example »

Delete Files

To delete a file with the File System module, use the `fs.unlink()` method.

The `fs.unlink()` method deletes the specified file:

Example

Delete "mynewfile2.txt":

```
var fs = require('fs');

fs.unlink('mynewfile2.txt', function (err) {
  if (err) throw err;
  console.log('File deleted!');
});
```

Run example »

Rename Files

To rename a file with the File System module, use the `fs.rename()` method.

Example

Rename "mynewfile1.txt" to "myrenamedfile.txt":
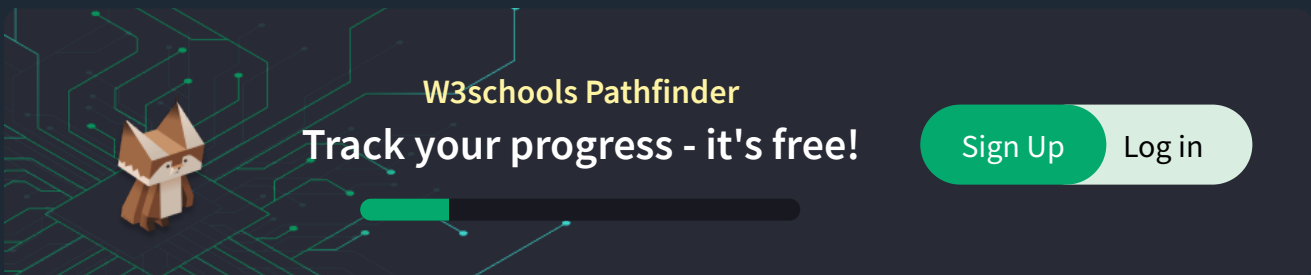
```
var fs = require('fs');

fs.rename('mynewfile1.txt', 'myrenamedfile.txt', function (err) {
  if (err) throw err;
  console.log('File Renamed!');
});
```

Run example »

Upload Files

You can also use Node.js to upload files to your computer.

Read how in our Node.js Upload Files chapter.

‹ Previous   **Next ›**

W3schools Pathfinder
Track your progress - it's free!   Sign Up   Log in