2)

The fs (File System) module in Node.js is used to interact with the file system on your machine. It allows you to perform operations such as reading, writing, deleting, and updating files and directories.

**Common Operations with fs Module**

1. **Reading Files**
2. **Writing Files**
3. **Appending to Files**
4. **Deleting Files**
5. **Renaming Files**
6. **Creating Directories**
7. **Deleting Directories**

**Importing the fs Module**

Before using any file system operations, you need to import the fs module:

javascript

Copy code

const fs = require('fs');

Let's go through each of these operations with code examples.

**1. Reading Files**

You can read files both synchronously and asynchronously using the fs.readFileSync() and fs.readFile() methods.

**Asynchronous File Reading (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.readFile('example.txt', 'utf8', (err, data) => {

if (err) {

console.error('Error reading file:', err);

return;

}

console.log('File contents:', data);

});

* This method reads the file asynchronously. It doesn't block the event loop.
* The second argument, 'utf8', specifies the encoding. Without this, you'd get a Buffer object instead of a string.

**Synchronous File Reading (Blocking):**

javascript

Copy code

const fs = require('fs');

try {

const data = fs.readFileSync('example.txt', 'utf8');

console.log('File contents:', data);

} catch (err) {

console.error('Error reading file:', err);

}

* This method reads the file synchronously and blocks the execution until the file is completely read.

**2. Writing to Files**

You can write data to a file using fs.writeFile() and fs.writeFileSync().

**Asynchronous File Writing (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.writeFile('output.txt', 'Hello, Node.js!', (err) => {

if (err) {

console.error('Error writing to file:', err);

return;

}

console.log('File has been written successfully');

});

* If the file doesn't exist, it will be created. If the file exists, it will be overwritten.

**Synchronous File Writing (Blocking):**

javascript

Copy code

const fs = require('fs');

try {

fs.writeFileSync('output.txt', 'Hello, Synchronous Node.js!');

console.log('File has been written successfully');

} catch (err) {

console.error('Error writing to file:', err);

}

**3. Appending to Files**

To append content to a file (rather than overwriting it), use fs.appendFile() or fs.appendFileSync().

**Asynchronous Appending (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.appendFile('output.txt', '\nAppended text.', (err) => {

if (err) {

console.error('Error appending to file:', err);

return;

}

console.log('Content has been appended successfully');

});

**4. Deleting Files**

To delete a file, you can use fs.unlink() or fs.unlinkSync().

**Asynchronous File Deletion (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.unlink('output.txt', (err) => {

if (err) {

console.error('Error deleting file:', err);

return;

}

console.log('File has been deleted');

});

**5. Renaming Files**

You can rename a file using fs.rename() or fs.renameSync().

**Asynchronous File Renaming (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.rename('oldname.txt', 'newname.txt', (err) => {

if (err) {

console.error('Error renaming file:', err);

return;

}

console.log('File has been renamed');

});

**6. Creating Directories**

To create a new directory, use fs.mkdir() or fs.mkdirSync().

**Asynchronous Directory Creation (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.mkdir('newdir', (err) => {

if (err) {

console.error('Error creating directory:', err);

return;

}

console.log('Directory has been created');

});

**7. Deleting Directories**

To delete a directory, use fs.rmdir() or fs.rmdirSync(). Note that the directory must be empty for it to be removed.

**Asynchronous Directory Deletion (Non-blocking):**

javascript

Copy code

const fs = require('fs');

fs.rmdir('newdir', (err) => {

if (err) {

console.error('Error deleting directory:', err);

return;

}

console.log('Directory has been deleted');

});