In Node.js, the path module provides utilities for working with file and directory paths. It is especially useful for handling file paths in a cross-platform way, as it takes care of differences between Windows (\) and POSIX (/) path separators.

To use the path module, you first import it with const path = require('path');. Here are some common methods and examples:

javascript

Copy code

const path = require('path');

### **1. Getting the Directory Name of a Path**

* **path.dirname()**: Returns the directory name of a given path.

javascript

Copy code

const filePath = '/home/user/documents/file.txt';

console.log(path.dirname(filePath)); // Output: /home/user/documents

### **2. Getting the Base Name (File Name) of a Path**

* **path.basename()**: Returns the last portion of a path, which is typically the file name. You can also specify an extension to remove.

javascript

Copy code

const filePath = '/home/user/documents/file.txt';

console.log(path.basename(filePath)); // Output: file.txt

console.log(path.basename(filePath, '.txt')); // Output: file

### **3. Getting the Extension of a File**

* **path.extname()**: Returns the extension of the path (including the dot).

javascript

Copy code

const filePath = '/home/user/documents/file.txt';

console.log(path.extname(filePath)); // Output: .txt

### **4. Joining Paths**

* **path.join()**: Joins multiple path segments together, normalizing the resulting path.

javascript

Copy code

const dir = '/home/user';

const file = 'documents/file.txt';

console.log(path.join(dir, file)); // Output: /home/user/documents/file.txt

### **5. Resolving an Absolute Path**

* **path.resolve()**: Resolves a sequence of paths or path segments into an absolute path.

javascript

Copy code

console.log(path.resolve('user', 'documents', 'file.txt'));

// Output: /<current\_working\_directory>/user/documents/file.txt

### **6. Normalizing a Path**

* **path.normalize()**: Normalizes a path by resolving .. and . segments.

javascript

Copy code

const weirdPath = '/home/user/../documents/./file.txt';

console.log(path.normalize(weirdPath)); // Output: /home/documents/file.txt

### **7. Checking if a Path is Absolute**

* **path.isAbsolute()**: Returns true if the path is absolute; otherwise, false.

javascript

Copy code

console.log(path.isAbsolute('/home/user')); // Output: true

console.log(path.isAbsolute('documents/file.txt')); // Output: false

### **8. Parsing a Path into an Object**

* **path.parse()**: Returns an object with properties for the root, dir, base, name, and ext.

javascript

Copy code

const filePath = '/home/user/documents/file.txt';

const parsedPath = path.parse(filePath);

console.log(parsedPath);

/\* Output:

{

root: '/',

dir: '/home/user/documents',

base: 'file.txt',

ext: '.txt',

name: 'file'

}

\*/

### **9. Formatting a Path Object into a String**

* **path.format()**: Accepts an object with properties like dir, base, name, and ext and constructs a path string.

javascript

Copy code

const pathObject = {

dir: '/home/user/documents',

base: 'file.txt'

};

console.log(path.format(pathObject)); // Output: /home/user/documents/file.txt

### **10. Cross-Platform Path Separator**

* **path.sep**: Provides the platform-specific path segment separator.

javascript

Copy code

console.log(path.sep); // Output: / on POSIX, \ on Windows

### **Example: Combining Multiple Path Operations**

javascript

Copy code

const filePath = '/home/user/../documents/file.txt';

// Normalize and parse the path

const normalizedPath = path.normalize(filePath);

const parsed = path.parse(normalizedPath);

console.log('Directory:', parsed.dir); // Output: /home/documents

console.log('Base:', parsed.base); // Output: file.txt

console.log('File Extension:', parsed.ext); // Output: .txt

console.log('File Name:', parsed.name); // Output: file

// Join paths

const newFilePath = path.join(parsed.dir, 'newfile.md');

console.log('New File Path:', newFilePath); // Output: /home/documents/newfile.md

### **Summary**

| **Method** | **Description** |
| --- | --- |
| path.dirname() | Gets the directory name of a path |
| path.basename() | Gets the base name (file name) of a path |
| path.extname() | Gets the extension of a file |
| path.join() | Joins multiple path segments |
| path.resolve() | Resolves a sequence of paths to an absolute path |
| path.normalize() | Normalizes a path, resolving .. and . segments |
| path.isAbsolute() | Checks if a path is absolute |
| path.parse() | Parses a path into an object with components |
| path.format() | Formats a path object into a string |
| path.sep | Provides the platform-specific path separator |

The path module makes it easy to work with file paths in a reliable and platform-agnostic way in Node.js.