

Debugging in Node JS

Debugging: The task to identify and remove errors from software application and is more than just printing out values in your code.

> Debugging Tools

Debugging tools provide important functionalities that console.log isn't able to provide

Debugging in Node.js application can be done using various tools.

- a. Core - Node.js debugger
- b. Node Inspector
- c. Builtin debugger in IDEs

a. Core Node.js debugger.

- Node.js provides built-in non-graphic debugging tool that can be used on all platforms.
- It provides different commands for debugging Node.js application.

Example

```
var fs = require('fs');
fs.readFile('test.txt', 'utf8', function(err, data) {
  debugger;
  if (err) throw err;
  console.log(data);
});
```

- If we want to check the "data" parameter write debugger inside callback function.
- To debug application command used is:
node debug app.js

Important debugging commands.

(i) next - stop at the next statement.

(ii) cont - continue execute and stop at the debugger statement if any

(iii) step - step in function.

(iv) out - step out of function

(v) watch - Add the expression or variable into watch.

(vi) watcher - See the value of all expressions and variables added into watch

(vii) Pause - pause running code.

2. Node Inspector.

* Node inspector is GUI based debugger.

Example:

```
var fs = require('fs');  
fs.readFile('test.txt', 'utf8', function(err, data)  
{  
  debugger;  
  if (err) throw err;  
  console.log(data);  
});
```

3. The Debugger keyword

- The debugger keyword stops the execution of javascript, and calls (if available) the debugging function.

- This has the same function as setting a breakpoint in the debugger.

- If no debugging is available, the debugger statement has no effect.

- With the debugger turned on, this code will stop executing before it executes the third line.

About MVC

The model-view-controller pattern of designing software (also known as MVC) was one of the first architectural design-patterns based on the responsibilities of its components software constructs. Trygve Reenskaug introduced the MVC while visiting Xerox PARC (the same research facility that pioneered everything from GUIs to laser printers in the 1970s).

MVC was revolutionary then and is still a key concept when discussing how to create a new software. When using MVC pattern, you can lay out the solution to any problem you expect to solve with the software you intend to build. The 3 components included by name in the pattern are:

The model, which describes the behaviour of the application regarding data, logic and rules.

The view which includes any information that the application may output and the controller, which accepts and translates input.

Note that the model, which is the core components on which you will wind up focussing, is independent of any user interface you are planning to create eventually.