GO IT

Node.JS #1

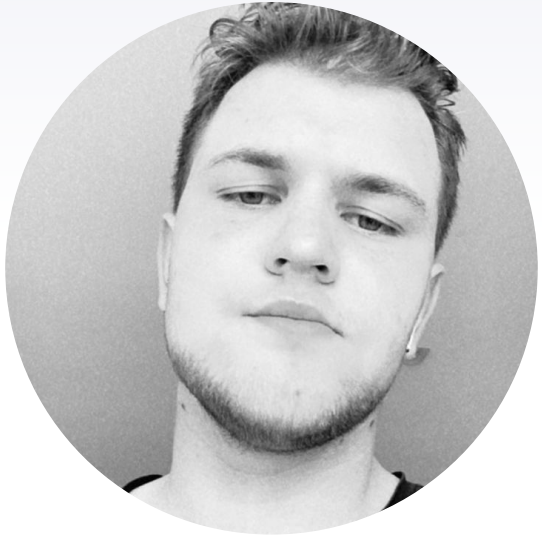
Yezhov Kyrylo

Software Engineer at EPAM systems

ezhov.kirill98@gmail.com

ezhov-kirill

ezhovkirill



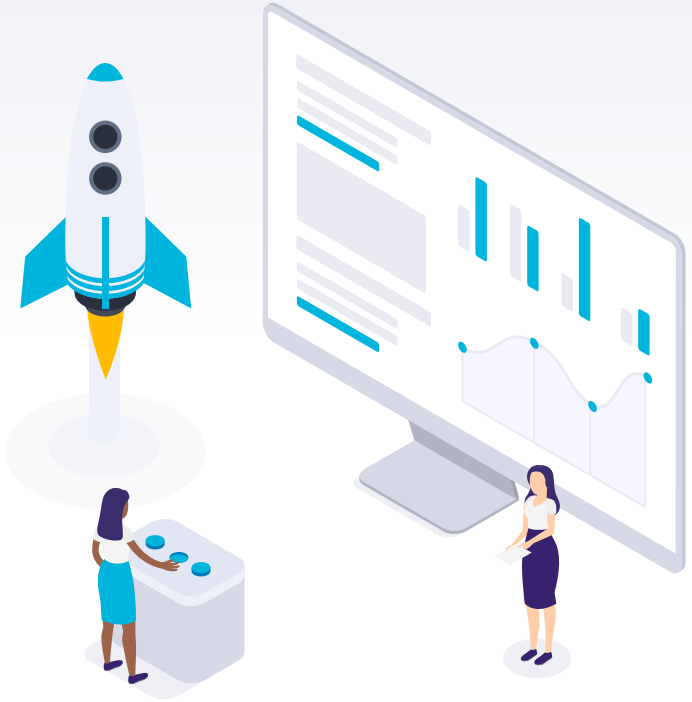
Regulations

- 12 lectures
- Questions anytime
- Initiative and involvement (Write to chat)

Agenda

- Node.js intro
- Modules in Node.js
- NPM
- Globals
- CLI
- File System
- Debug
- Questions

1 Node.js intro



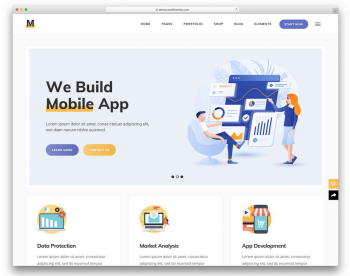
Node.js intro

Node.js is an open-source, cross-platform, JavaScript run-time environment that executes JavaScript code outside of a browser.

Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser.

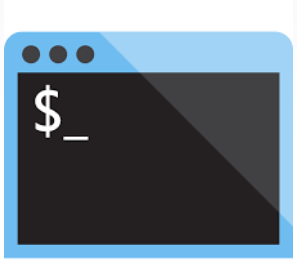
Consequently, **Node.js** represents a "JavaScript everywhere" paradigm, unifying web application development around a single programming language, rather than different languages for server- and client-side scripts.

What can you do with Node.js

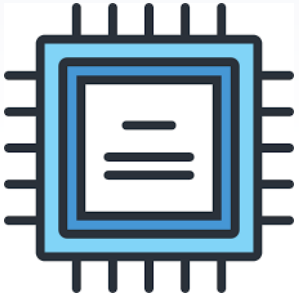


SPA
(Single page application)

RPA
(Real time application)



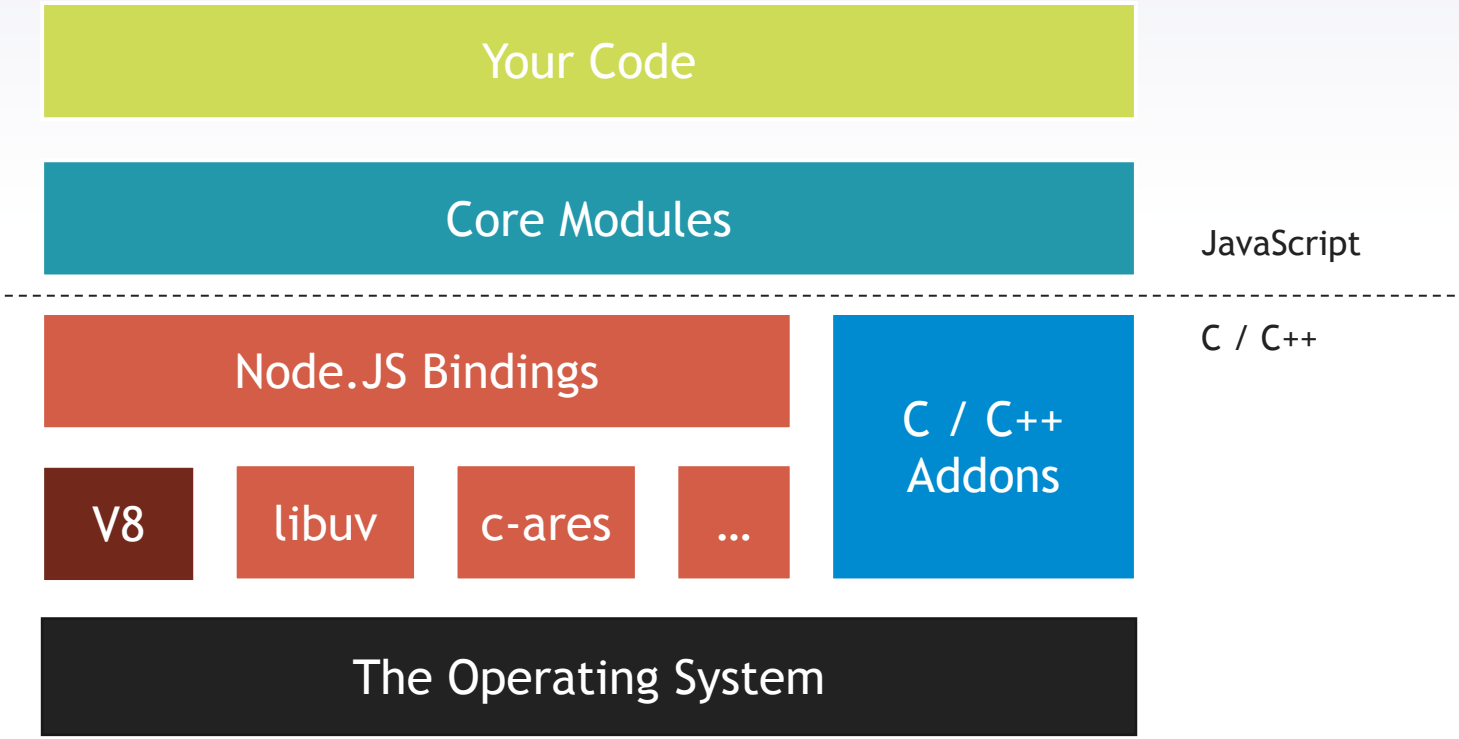
CLI
(Command-line interface)



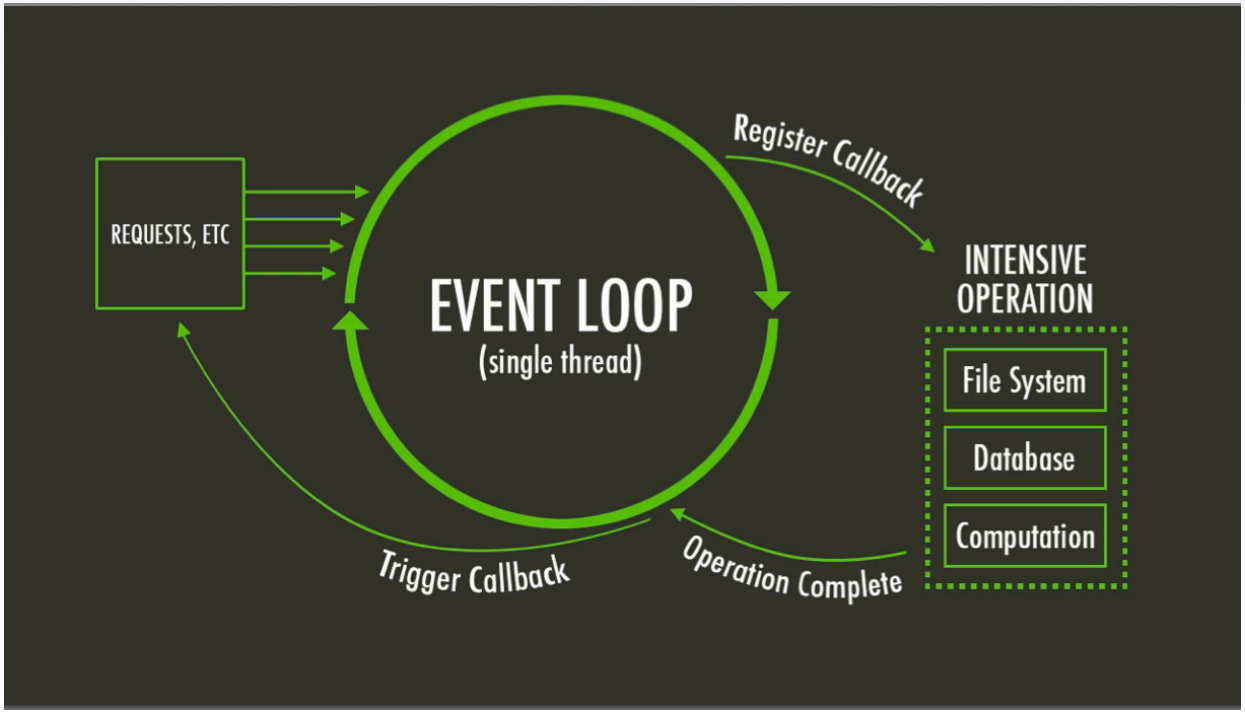
Hardware

Why it worth to learn it?

Architecture



Event loop in Node.js



Node.js Environment

Now you have downloaded and installed Node.js on your computer, **Let's try to launch our first app!**

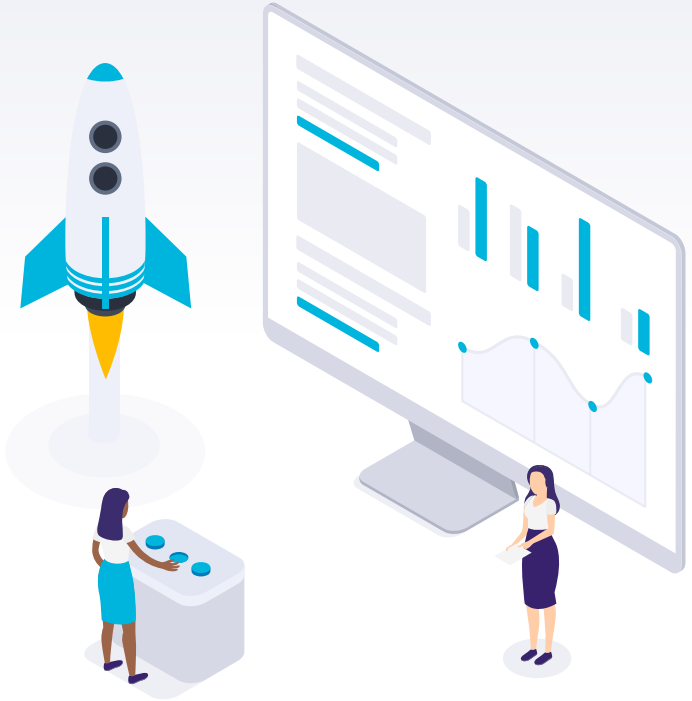
1. **Create file**
   Create a JS file named "test.js", and add the following code:

```
1
2   console.log("Hello World!");
3
```

2. **Command Line Interface**
   Start your command line interface, write "node test.js» and hit enter, result should look like this:

```
sh-3.2$ node test.js
Hello World!
sh-3.2$
```

1 Modules in Node.js



Modules

What is a Module in **Node.js**? Consider modules to be the same as JavaScript libraries. A set of functions you want to include in your application.

Node.js has **3 types** of modules

3rd party Modules

NPM Registry contains a huge list of 3rd party modules for all your needs

Your Own(Local) Modules

You can create your own modules, and easily include them in your applications.

Built-in(Core) Modules

Node.js has a set of built-in modules which you can use without any further installation.

Check Built-in modules reference for a complete list of modules.

Modules

Your Own Modules

You can create your own modules, and easily include them in your applications.

Example

Create a module that returns the current date and time:

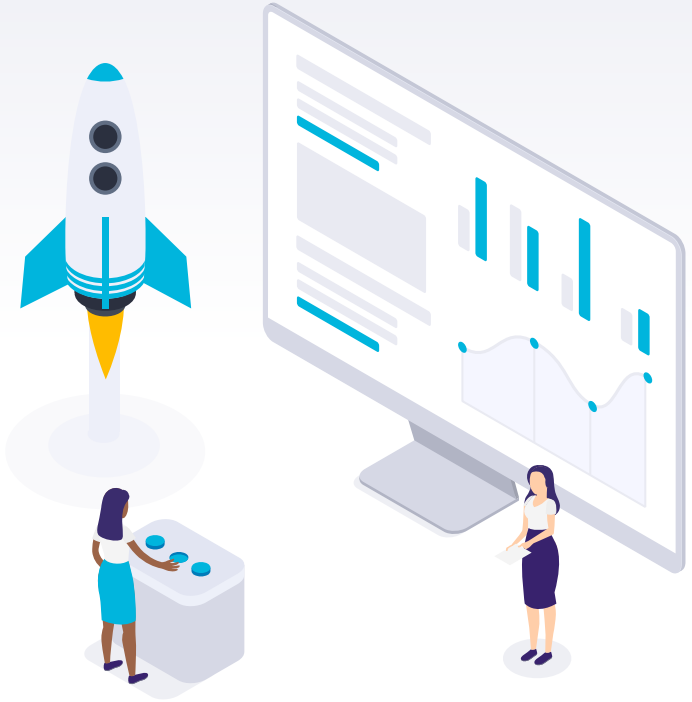
```
3
4   exports.myDateTime = function () {
5     return Date();
6   };
7
```

Use the exports keyword to make properties and methods available outside the module file.

Now you can include and use the module in any of your Node.js files:

```
8
9   const dateUtils = require('./dateUtils');
10  const timestamp = dateUtils.myDateTime();
11  console.log(timestamp);
12
```

1 NPM



What is NPM?

NPM stands for **N**ode **P**ackage **M**anager. It allows for seamless node.js package management. You can install, share and manage node.js packages.

NPM consists of three components:

- **Website**
- **Registry**
- **CLI**

Website

npm official website is https://www.npmjs.com/. Using this website you can find packages, view documentation, share and publish packages.

Registry

npm registry is a large database consisting of more than half a million packages. Developers download packages from the npm registry and publish their packages to the registry.

CLI(Command Line Interface)

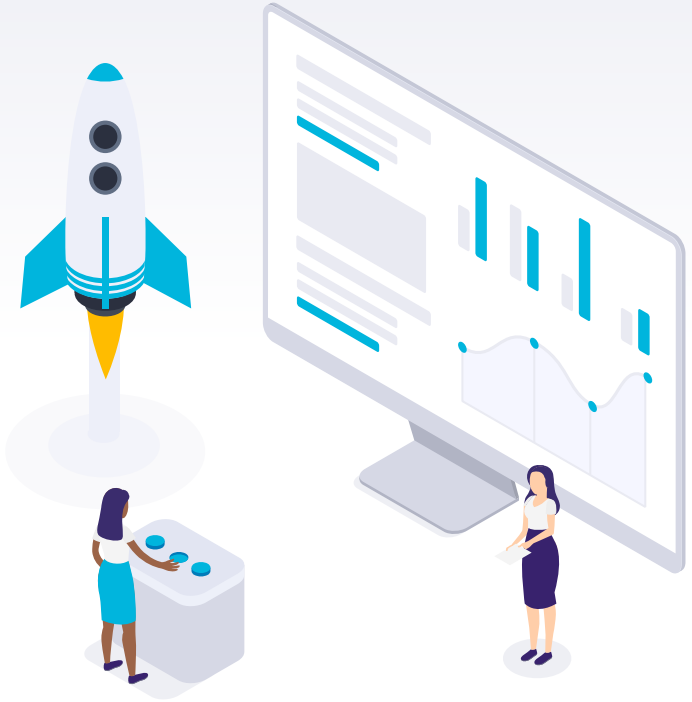
The is the command line that helps in interacting with the npm for installing, updating and uninstalling packages and managing dependencies.

package.json

```
{
  "name": "project",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "devDependencies": {
    "gulp": "^3.9.0"
  }
}
```

The **package.json** is the project manifest file.
Using **package.json** you can manage dependencies and write scripts.
It has all the metadata about the project.

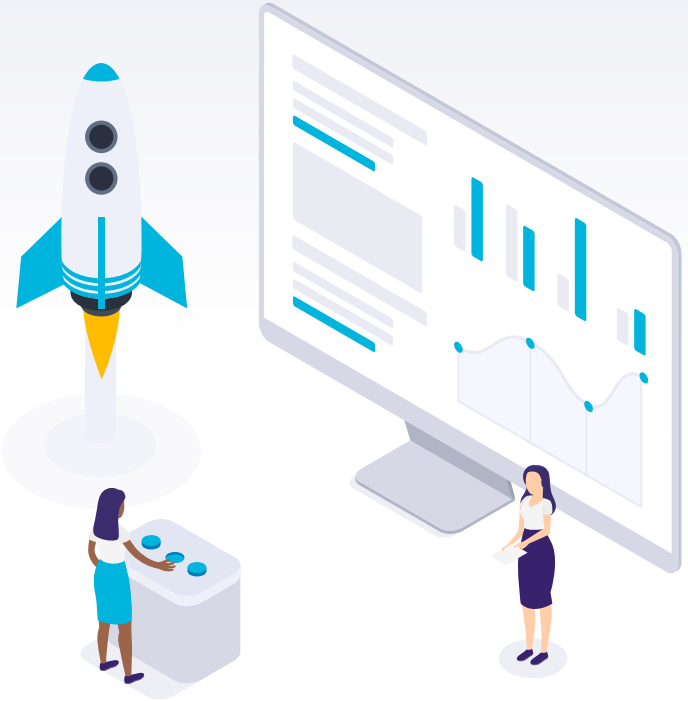
1 Globals



Globals

1. global
2. process.env
3. process.argv
4. process.exit
5. __dirname
6. __filename

1 CLI

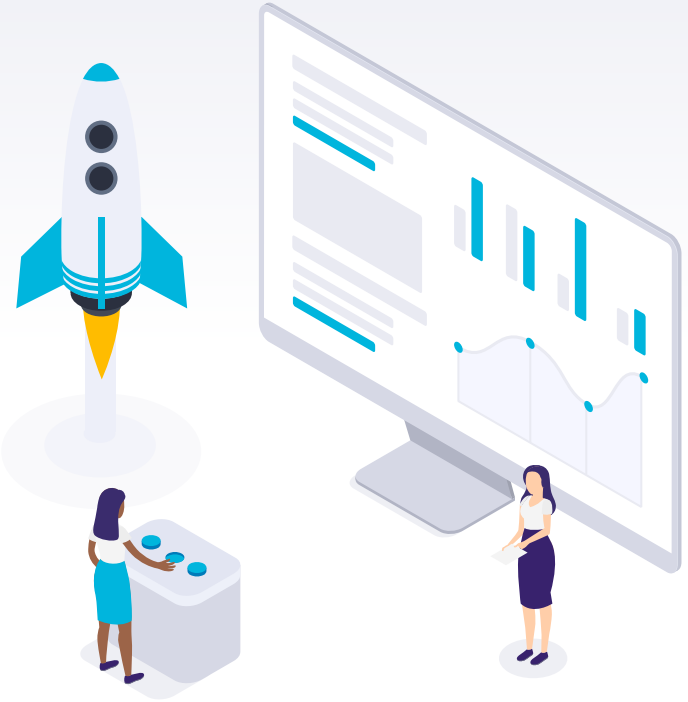


CLI

1. Install calc.js npm package and declare it in your project;

2. Write a small CLI app that returns result of different Math operations when you pass arguments through argv

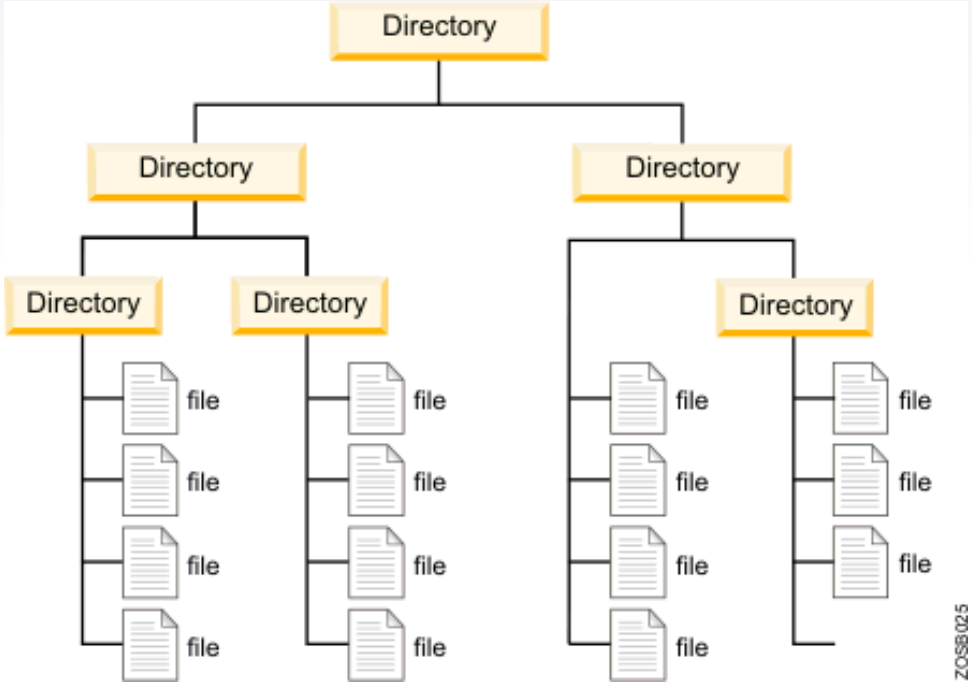
1 File system



Path module

```
1  var path = require("path");
2
3  // Normalization
4  console.log('normalization : ' + path.normalize('/test/test1//2slashes/1slash/tab/..'));
5
6  // Join
7  console.log('joint path : ' + path.join('/test', 'test1', '2slashes/1slash', 'tab', '..'));
8
9  // Resolve
10 console.log('resolve : ' + path.resolve('main.js'));
11
12 // extName
13 console.log('ext name : ' + path.extname('main.js'));
```

FS module

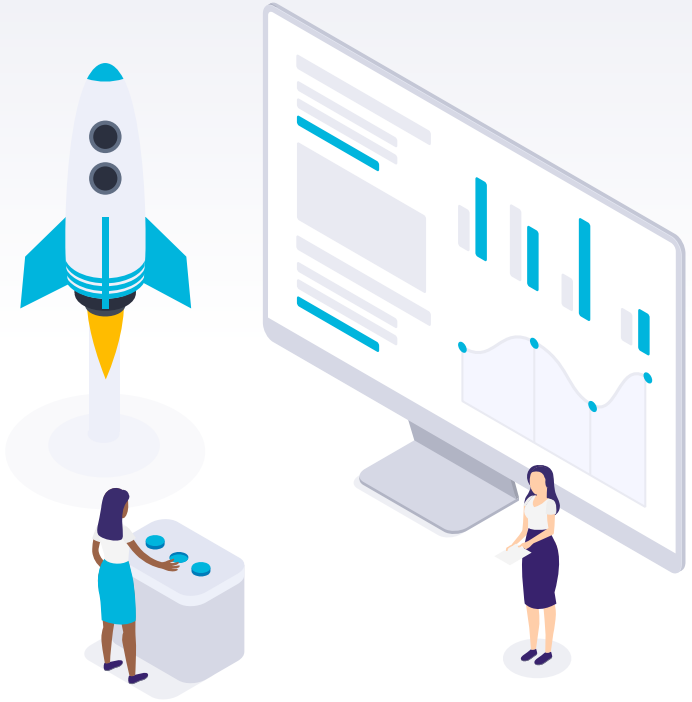


A file system is a process that manages how and where data on a storage disk, typically a hard disk drive (HDD), is stored, accessed and managed. It is a logical disk component that manages a disk's internal operations as it relates to a computer and is abstract to a human user.

FS module

- fs.readFileSync(<path>, <encoding>)
- fs.writeFileSync(<path>, <content>, <encoding>)
- fs.renameSync(<oldPath>, <newPath>)
- fs.readdirSync(<path>)
- fs.unlinkSync(<path>)

1 DEBUG



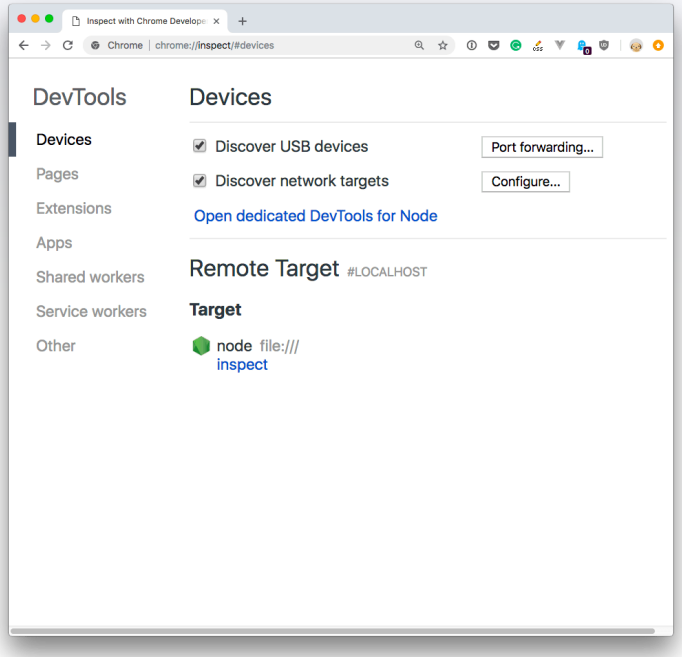
Debug

How can we do the same with Node.js code, and debug Node modules with access to the filesystem and other Node.js capabilities? It's very simple, actually.

Open your terminal and run:

```
node --inspect
```

Then in Chrome type this URL: chrome://inspect.



Click the Open dedicated DevTools for Node link next to the Node target, and you'll have access to Node.js in the browser DevTools

Questions

Thanks!

ezhov.kirill98@gmail.com

ezhov-kirill

ezhovkirill