NODE.JS

Saramma Varghese

WHAT IS NODE.JS ?

- Node.js is a powerful framework developed on **Chrome’s V8 JavaScript engine** that compiles the JavaScript directly into the native machine code.
- It is a lightweight framework used for creating server-side web applications.
- Node.js makes use of an event-driven, non-blocking I/O model which makes it a right pick for the data-intensive real-time applications.
- node.js makes use of packages and modules.

What Can Node.js Do?

- Node.js can generate dynamic page content
- Node.js can create, open, read, write, delete, and close files on the server
- Node.js can collect form data
- Node.js can add, delete, modify data in your database

NPM (NODE PACKAGE MANAGER)

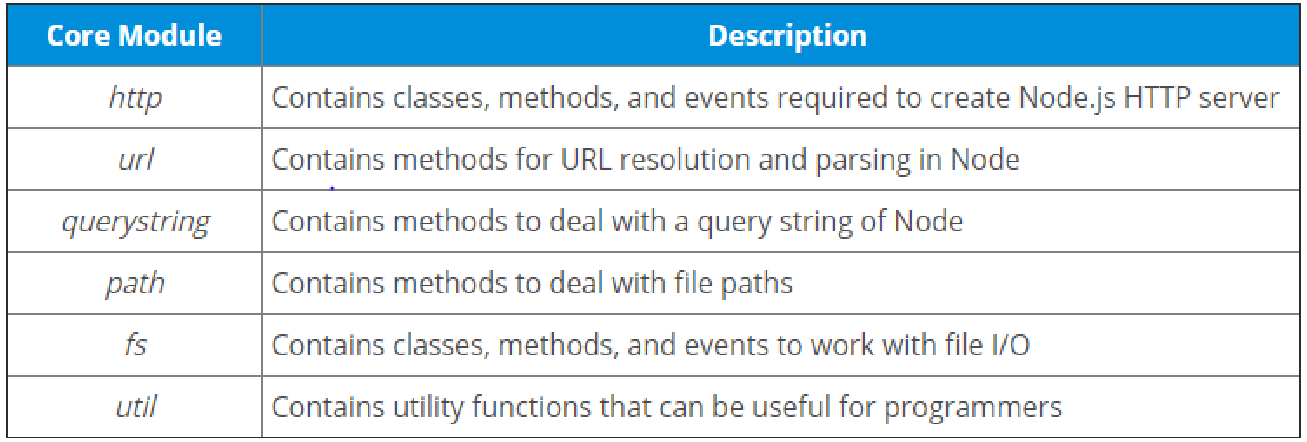
- It is a package manager for Node.js packages/modules.
- NPM basically helps in two ways:
  - Provides and hosts Online repositories for node.js packages/modules which can be easily downloaded and used in our projects
  - Provides the Command line utility in order to install various Node.js packages, manage Node.js versions and dependencies of the packages.

MODULES

- It represents various functionalities that are bundled up into single or multiple JS files.
- Node.js basically provides three types of modules:
  - Core Modules
  - Local Modules
  - Third-Party Modules

CONT…

- **CORE MODULE** :
  - It bundle the absolute minimum functionalities.
  - These modules generally get loaded when the Node process starts its execution.

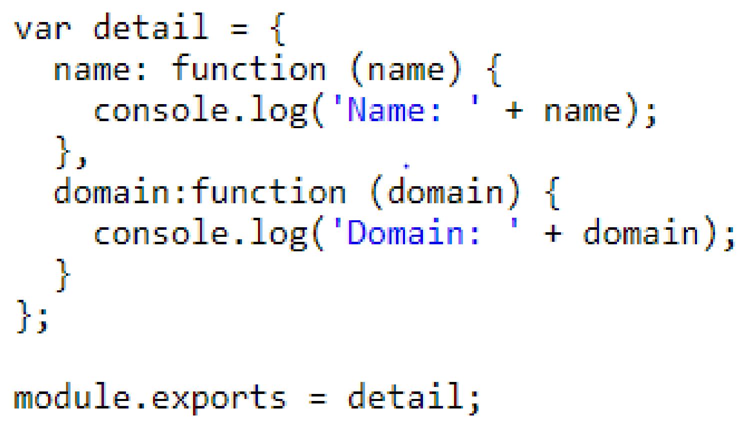


Core Module	Description
http	Contains classes, methods, and events required to create Node.js HTTP server
url	Contains methods for URL resolution and parsing in Node
querystring	Contains methods to deal with a query string of Node
path	Contains methods to deal with file paths
fs	Contains classes, methods, and events to work with file I/O
util	Contains utility functions that can be useful for programmers

  - **Var module = require('module_name');** used to load core module.

CONT…

- **LOCAL MODULE** :
  - They are custom modules that are created locally by user/developer in the application.
  - Create local module.js file



```
var detail = {
  name: function (name) {
    console.log('Name: ' + name);
  },
  domain:function (domain) {
    console.log('Domain: ' + domain);
  }
};

module.exports = detail;
```

CONT…

- Include module file in your main application file.

```
var myLogModule = require('./Local_module.js');
myLogModule.name('Edureka');
myLogModule.domain('Education');
```

- execute these files using below command

```
node application.js
```

CONT…

- **EXTERNAL MODULE**:
  - can use the external modules only by downloading them via NPM.

    npm install --g <module_name>
  - Include your module file in your main application file:

    npm install --save <module_name>

JSON FILE

- The **package.json file** in Node.js is the heart of the entire application.
- It is basically the manifest file that contains the metadata of the project.
- Contents in json file categorized into two:
    - **Identifying metadata properties:** This consists of properties like the project name, current module version, license, author of the project, project description etc.
    - **Writing directly to file:** You can directly write the necessary information into the package.json file and include it, in your project.

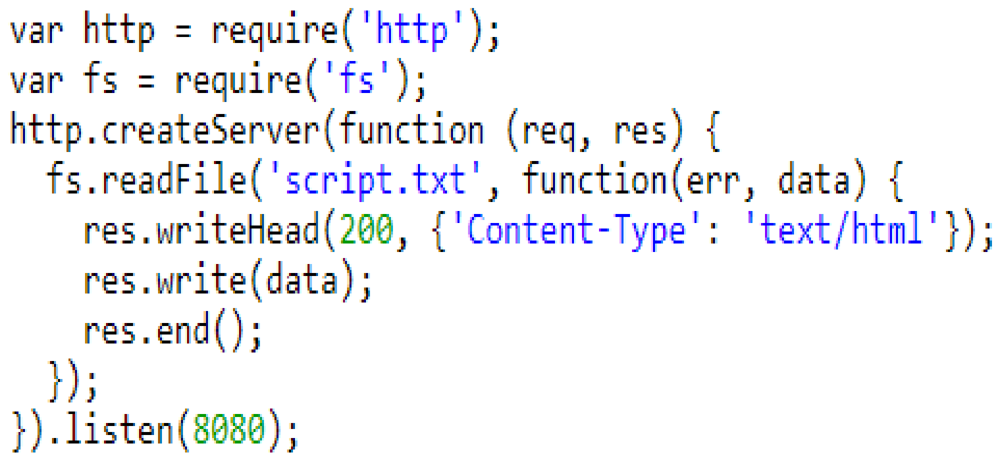
FILE SYSTEM

- To access the physical file system, Node.js makes use of the **fs** module which basically takes care of all asynchronous and synchronous file I/O operations.
- Command for module import:

```
var fs = require(‘fs’);
```

CONT…

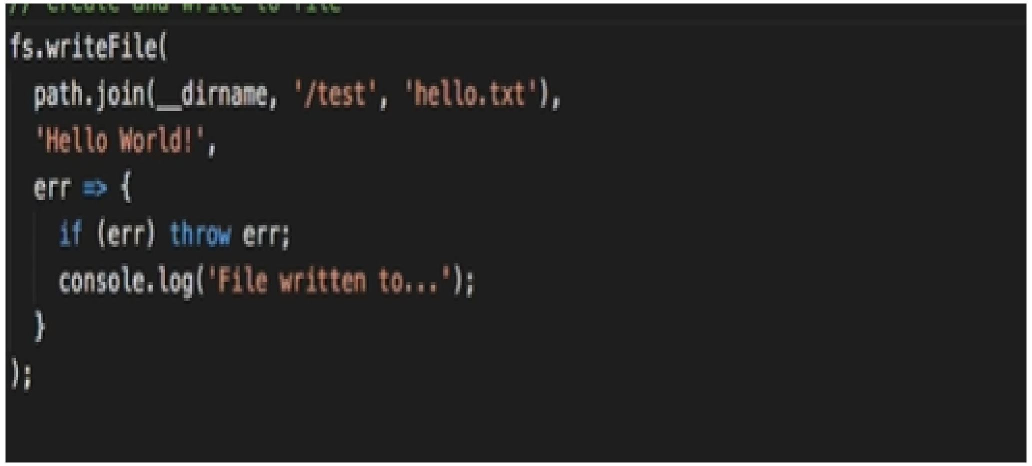
- For Reading File:



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

CONT…

- For writing file :



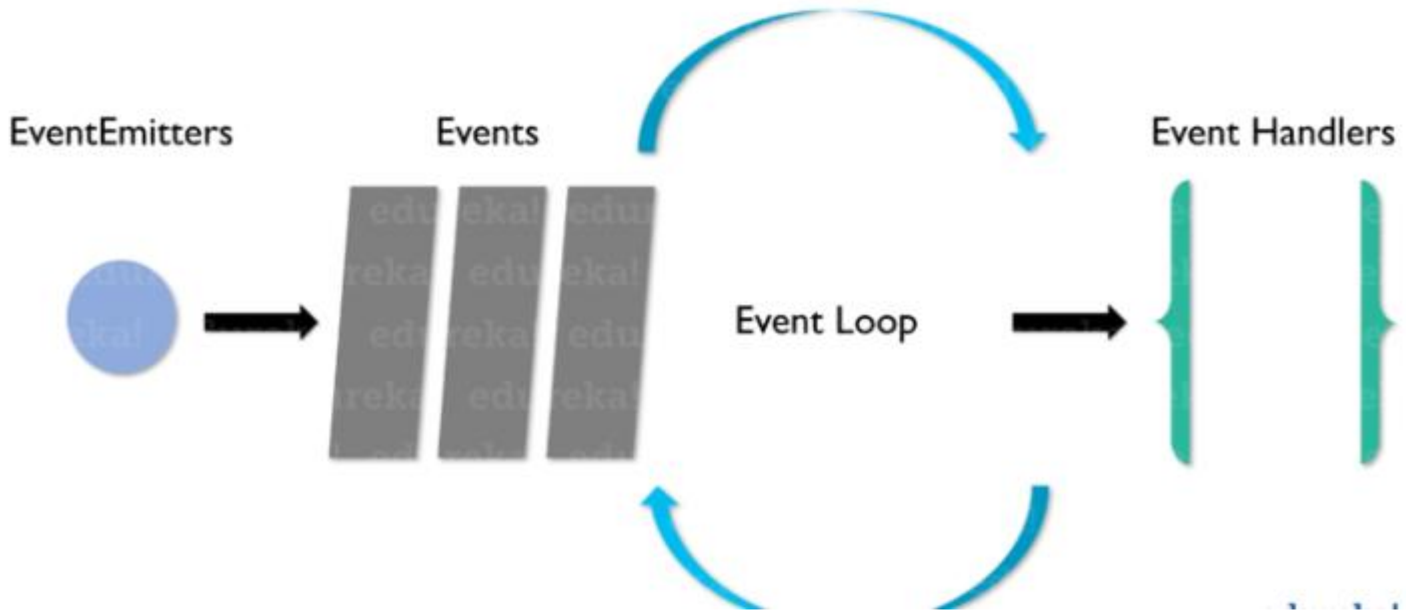
```
fs.writeFile(
  path.join(__dirname, '/test', 'hello.txt'),
  'Hello World!',
  err => {
    if (err) throw err;
    console.log('File written to...');
  }
);
```

EVENT

- Node.js supports concurrency as it is event-driven, and thus makes use of concepts like events and callbacks.
- The async function calls help Node.js in maintaining concurrency throughout the application.
- There is a main loop which waits and listens for events, and once any event is completed, it immediately initiates a callback function.

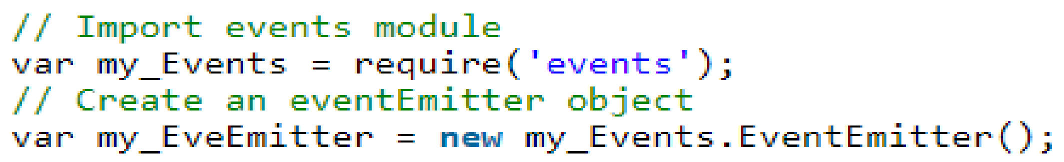
CONT…

- Events driven in Node.js ,



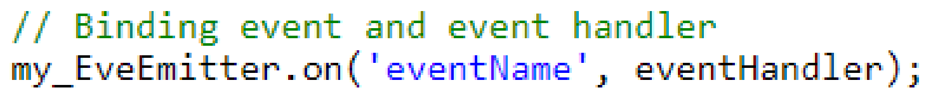
CONT…

- Binding Event to an Event Listener



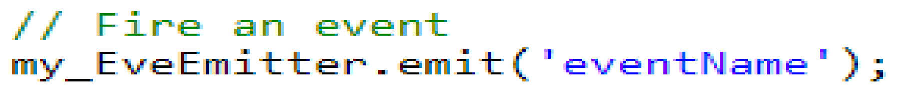
```
// Import events module
var my_Events = require('events');
// Create an eventEmitter object
var my_EveEmitter = new my_Events.EventEmitter();
```

- Binding Event Handler to an Event



```
// Binding event and event handler
my_EveEmitter.on('eventName', eventHandler);
```

- Firing an Event



```
// Fire an event
my_EveEmitter.emit('eventName');
```

HTTP MODULE

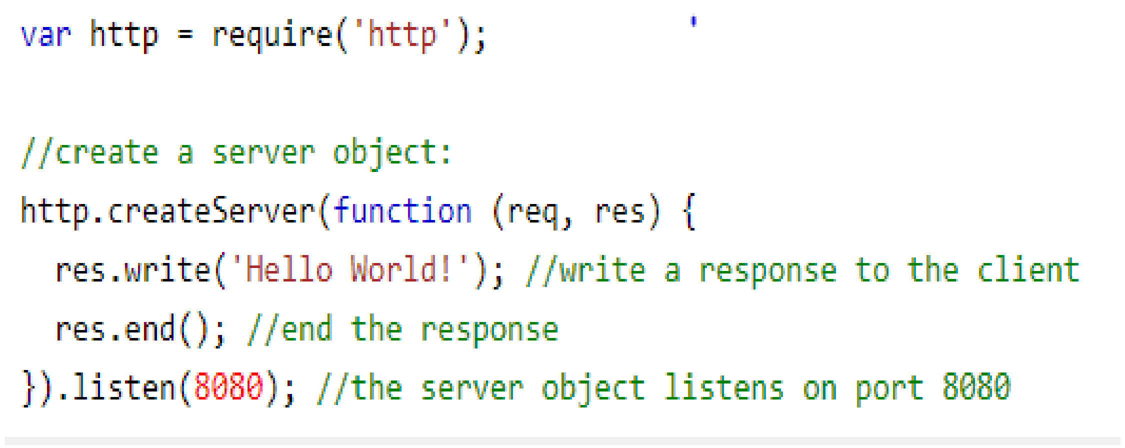
- Allows Node.js to transfer data over the Hyper Text Transfer Protocol (HTTP).
- To include HTTP module, use **require()** method :

  var http = require(‘http’);

- The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.
- **createServer()** is used for creating HTTP server

CONT…

- Function passed into the http.createServer() method , will be executed when someone tries to access the computer on port 8080.



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

//create a server object:
http.createServer(function (req, res) {
  res.write('Hello World!'); //write a response to the client
  res.end(); //end the response
}).listen(8080); //the server object listens on port 8080
```

FEATURES OF NODE.JS

- Open source
- Simple and fast
- Asynchronous
- High Scalability
- No Buffering
- Cross-Platform

THANK YOU….