TESTYANTRA

SOFTWARE SOLUTIONS (INDIA) PVT. LTD.

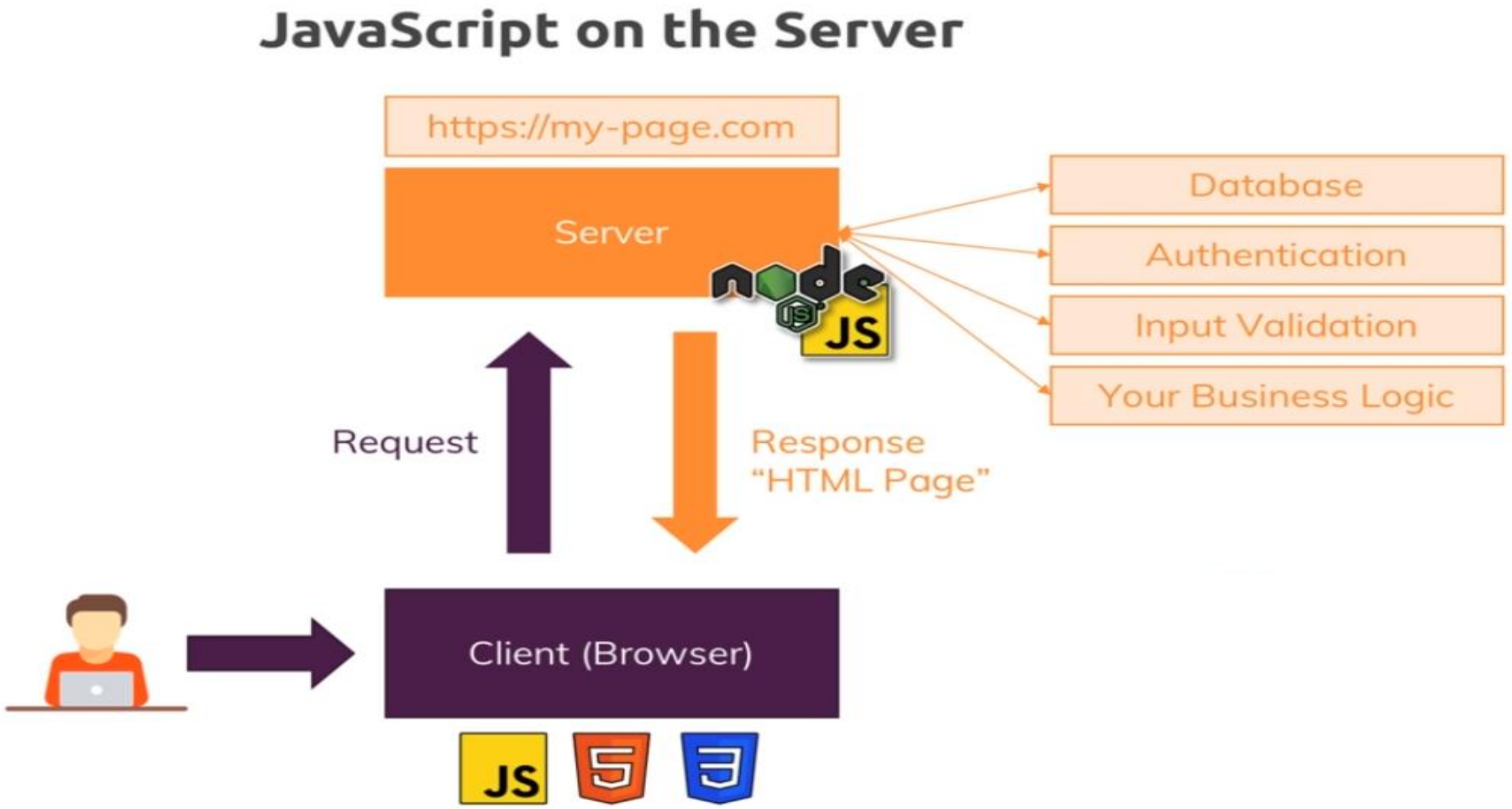
Node JS

EXPERIENTIAL
learning factory

What is Node JS

- Node.js is an open-source **server side Javascript runtime environment** built on Chrome's V8 JavaScript engine
- It provides an **event driven, non-blocking (asynchronous) I/O and cross-platform runtime environment** for building highly scalable server-side applications using JavaScript
- It is written in C++
- Node.js can be used to build different types of applications such as command line application, web application, real-time chat application, REST API server etc. However, it is mainly used to build network programs like web servers, similar to PHP, Java, or ASP.NET.
- Node.js was written and introduced by Ryan Dahl in 2009
- Node.js official web site: https://nodejs.org
- Node.js on GitHub: https://github.com/nodejs/node

JavaScript on Server Side



Installing Node JS

Go to the Browser and type https://nodejs.org

There are to types of Version available for Download.

1. LTS – Long Term Support

Long term Support is defined as the life cycle management policy in which a stable release of computer Software is maintained.

2. Current

Current Version is the Latest Version released Which is Not supported by every Browser.

Node JS Installation Verification.

Open the Command Prompt (Window + R)

Type node –v

How JavaScript Works

- Chrome and Node.JS are the two Environment which helps to run the JavaScript code In Browser Side and Server Side Respectively.
- When JavaScript code runs in the Browser, it sends the code to the V8 Engine Which compiles and return the machine Code.
- V8 Engine is a Google Open Source Which helps to compile JavaScript code.
- V8 is written in C++, So some one can write the c++ code and add V8 to increase the features, In the same way Node and Chrome developers has done.
- They have developed Chrome and Node with C++ Programming and added the V8 Engine features.
- Chrome has Implemented the Window Object along with Original Java Script Code.
- Node has Implemented the Global Object along with Original Java Script Code.

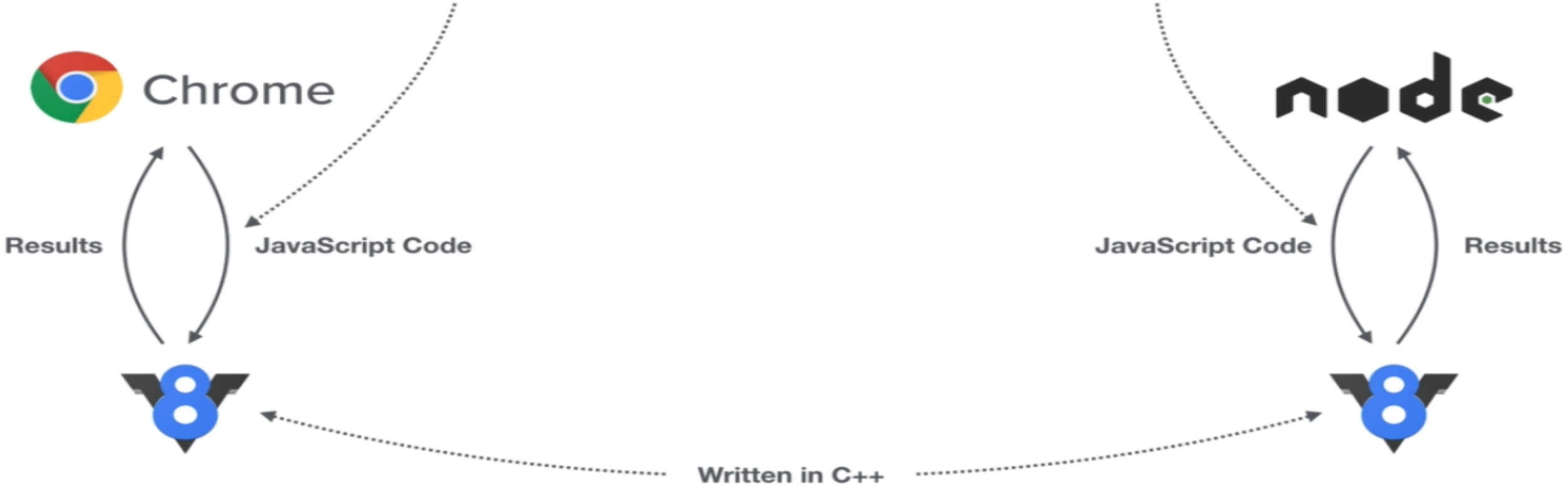
How JavaScript Works

The V8 JavaScript Engine

JavaScript (Chrome)	C++
localStorage.getItem	Some C++ function
document.querySelector	Some C++ function

C++ Bindings

JavaScript (Node.js)	C++
fs.readFile	Some C++ function
os.platform	Some C++ function



Node.JS Roles

Run Server

- Node.js can create a server and listen to incoming requests

Business Logic

- Handle requests
- Validate input
- Connect to database

Responses

- Return Responses(HTML pages, JSON…)

Advantages of Node.js

- Node.js is an open-source framework under MIT license. (MIT license is a free software license originating at the Massachusetts Institute of Technology (MIT)).
- Uses JavaScript to build entire server side application.
- Lightweight framework that includes bare minimum modules. Other modules can be included as per the need of an application.
- Asynchronous by default. So it performs faster than other frameworks.
- Cross-platform framework that runs on Windows, MAC or Linux.

Execution of Node JS Code

Node JS Code can be Executed in Two Ways

1. Using REPL

REPL

R - > Reading Input

E - > Executing Input

P - > Printing

L - > Loping for Next Input

2 . Using Files

Creating JavaScript File and writing the Code and Executing.

Note :

REPL way of executing can be used for practicing purpose , note for storing, Code we have written Is just for temporary.

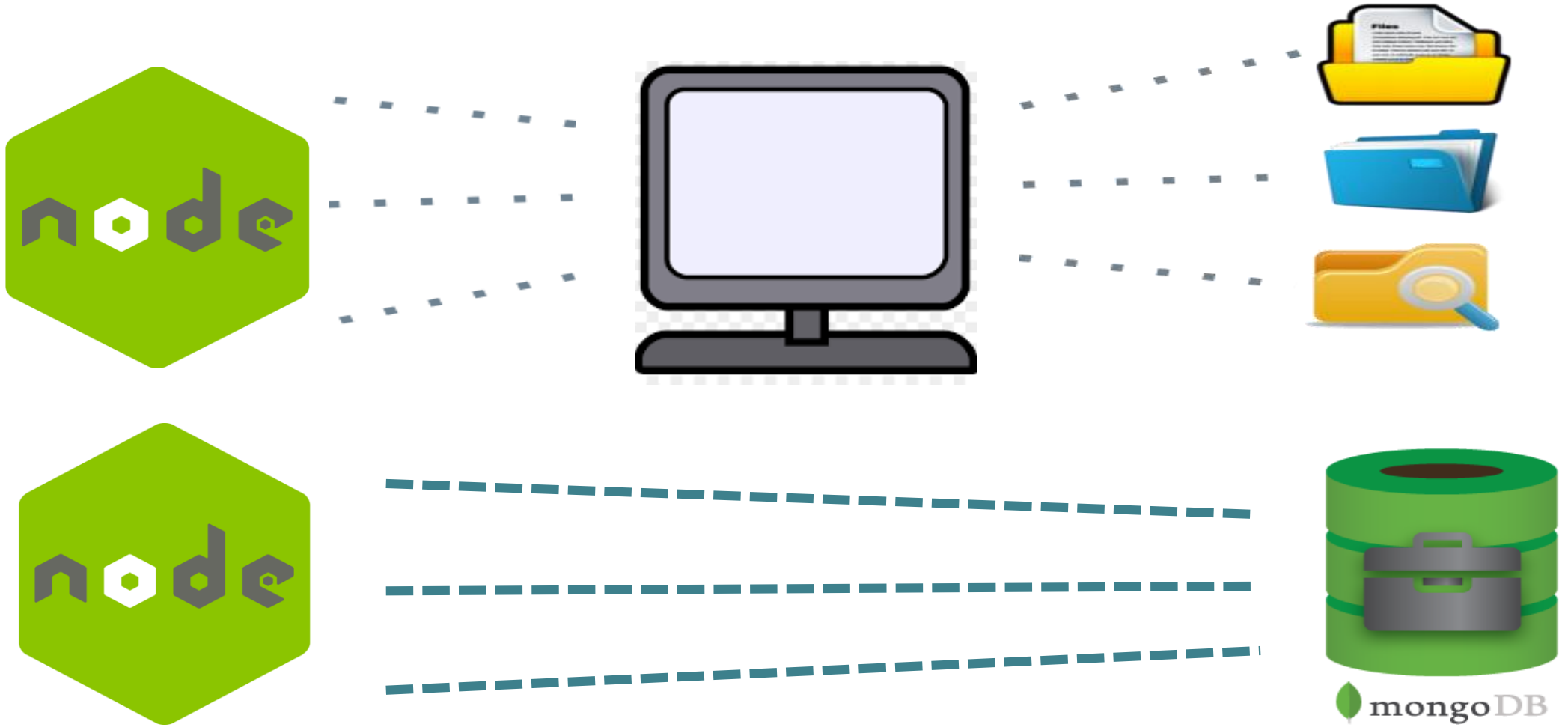
Node.js Console - REPL

- Node.js comes with virtual environment called REPL (Node shell).
- REPL stands for Read-Eval-Print-Loop.
- It is a quick and easy way to test simple Node.js/JavaScript code.
- To launch the REPL, open command prompt (in Windows) or terminal (in Mac or UNIX/Linux) and type ”node”. It will change the prompt to > in Windows and MAC.

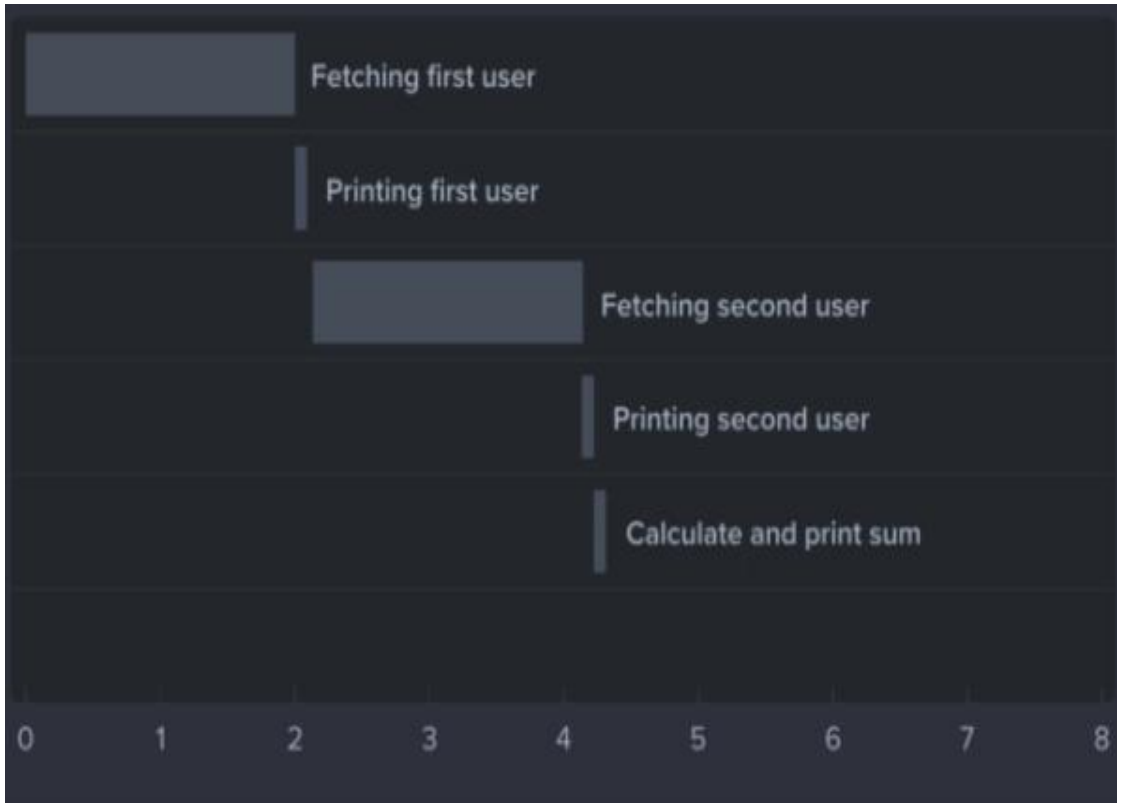
```
>
> function add(a,b) {
... return a+b;
... }
undefined
> add(10,20)
30
>
```

I/O Operation

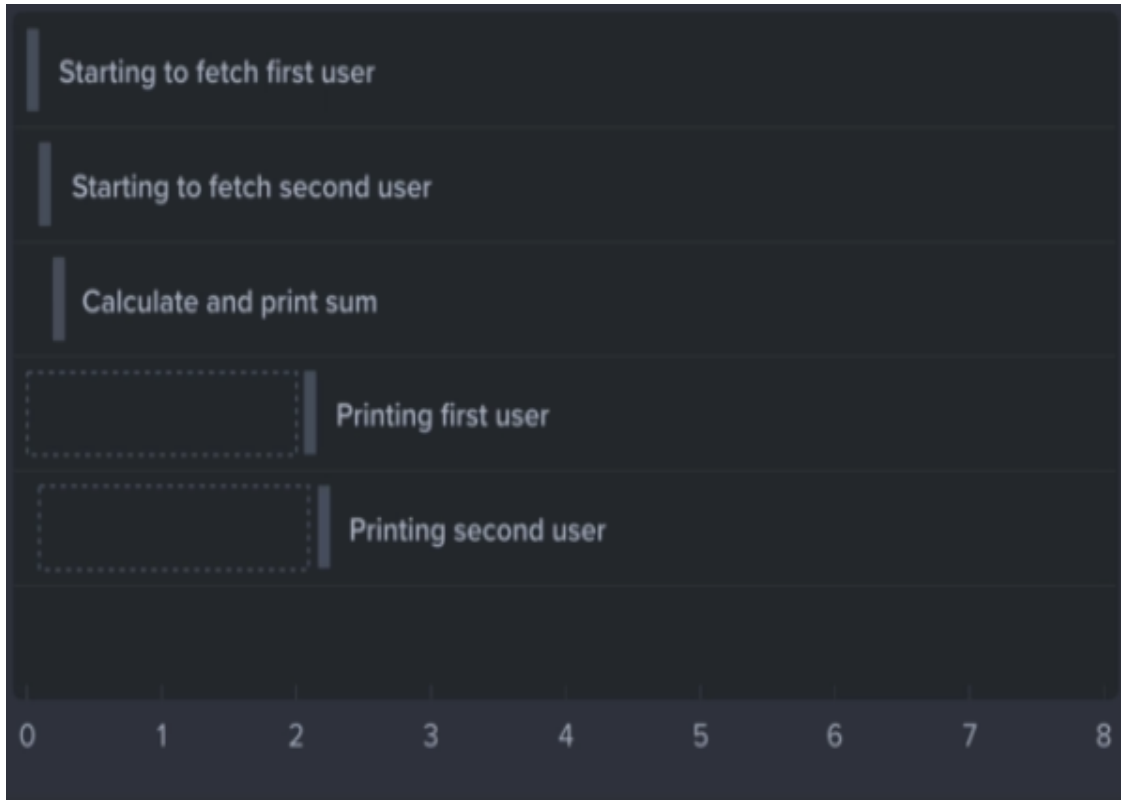
Input / Output Operation (I/O Operation)



Blocking and Non-Blocking Input and Output Operation.



Blocking I/O Operation



Non-Blocking I/O Operation

Blocking v/s Non-Blocking I/O

- Blocking and Non-Blocking Input/Output Operation

Node JS provides a feature of Non-blocking I/O

Blocking I/O

When User Performs the Input and Output Operation, Which takes the some amount of time at Which User cannot be able to interact with the Browser.

Non-Blocking I/O

When User Performs the I/O operation then Node will freezes the Browser and allow the User to interact, The I/O operation is running in the background.

Non Blocking I/O operation can be achieved with the help of callback Function which will got to the Node API and waits in the callback Que and comes to the stack when stack is Empty.

Globals

Node.js global objects are global in nature and they are available in all modules. We do not need to include these objects in our application, rather we can use them directly. These objects are modules, functions, strings and object itself as below.

- __filename
- __dirname
- setTimeout()
- setInterval()
- clearTimeout()
- clearInterval()
- Module
- Console
- Process

Module System

Node Module System is the features provided by the node to add the functionality to our script file to do the interesting things.

1. Core Module
2. Third Party Module
3. Local Module

Module System

1. Core Modules :

Core Modules are the Modules Provided by the Node JS itself but Module is not available in script File until we import the Module.

Examples : fs Module , os Module, path Module, url Module, http Module, util Module, Buffer Module.

2. Third Party Modules :

Third Party Modules are the Modules provided by the node package manager.

How to Use the Third party Module :

1. npm install module-name
2. const preferedName = require(“module-name”);

3. Local Module :

Local Module is defined as the Module created by us and used in other script Files to add the extra features.

```
module.exports = { name , calAge} ;

const imports = require('./exporting.js');
console.log(imports.name);
console.log(imports.calAge());

// Assignment :
1. Create the File Name as notes.js
2. Create a function to return the some message.
3. export the function
4. import in the importing.js and display the output.
```

Basic Core Modules

- http
  Hyper Text Transfer Protocol
  A protocol for transferring data which is understood by browser and server.

- https
  Hyper Text Transfer Protocol Secure
  HTTP + data encryption (during transmission).

- fs
  fs => File System
  fs module can access physical file system.
  The fs module is responsible for all the asynchronous or synchronous file I/O operations.

- path
  The path module provides utilities for working with file and directory paths.

- os
  The os module provides a number of operating system-related utility methods.

Working on File System

Reading and Writing the File in Synchronous Way

- **Writing the File**

  Syntax :

  fs.writeFileSync(‘ File Name ‘ , “Data or File Content");

  Note : fs.writeFileSync() donot returns anything.

- **Reading the Fie**

  Syntax :

  fs.readFileSync(‘File Name’);

  Note : fs.readFileSync() returns the Buffer Data.

Working on File System

Reading and Writing the File in Synchronous Way

- **Writing the File**

  Syntax :

  ```
  fs.writeFile( ' File Name ' , "Data or File Content" , (err => {  console.log(err) } );
  ```

  Note : fs.writeFile() donot returns anything.

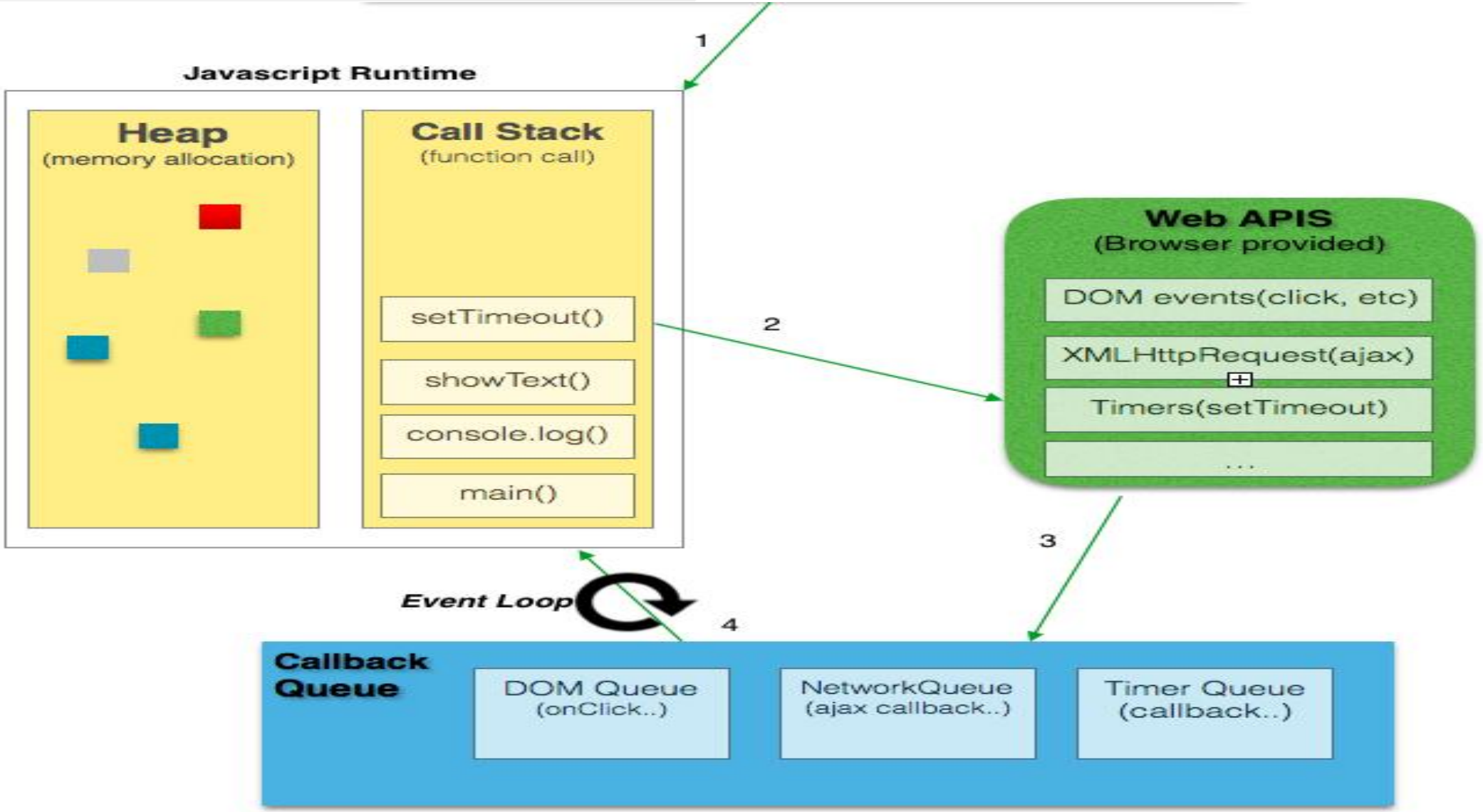
- **Reading the Fie**

  Syntax :

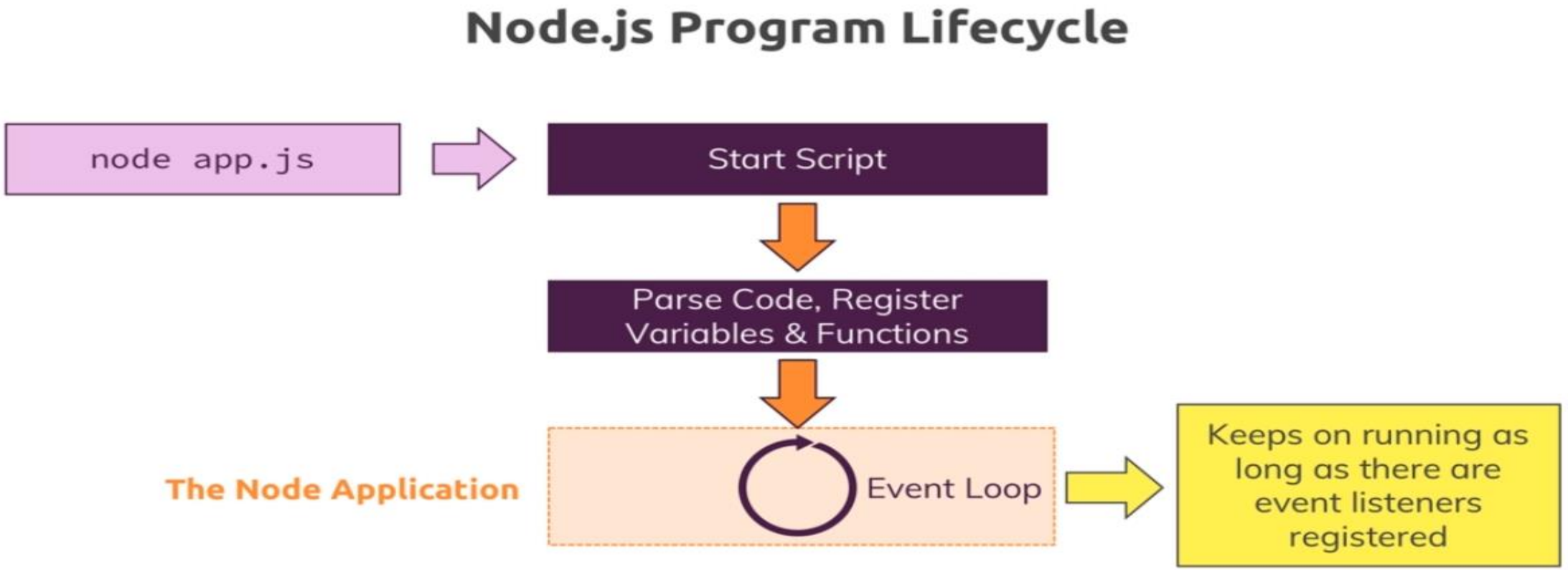
  ```
  fs.readFile('File Name' , (err , data) =>  {
          if(err) {    } else {    }
          };
  ```

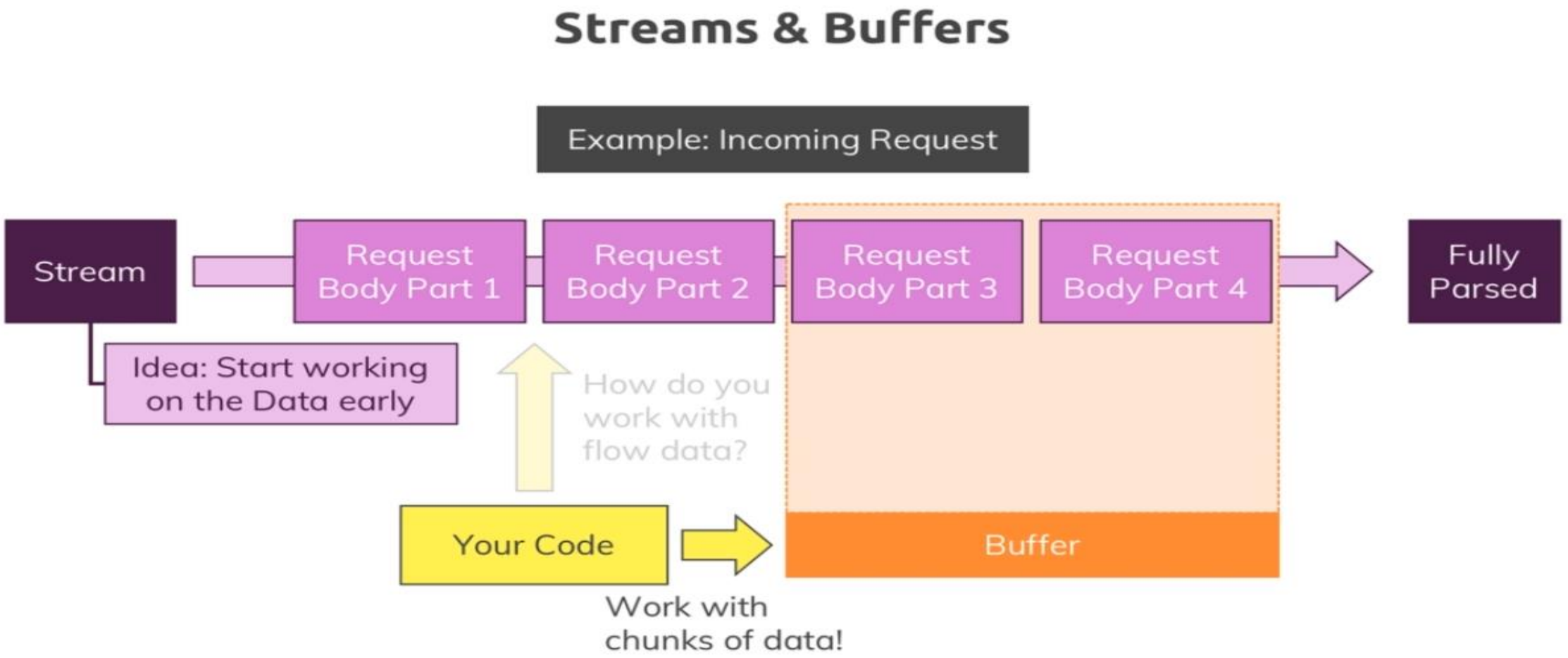
  Note : fs.readFileSync() returns the Buffer Data.

Asynchronous Way

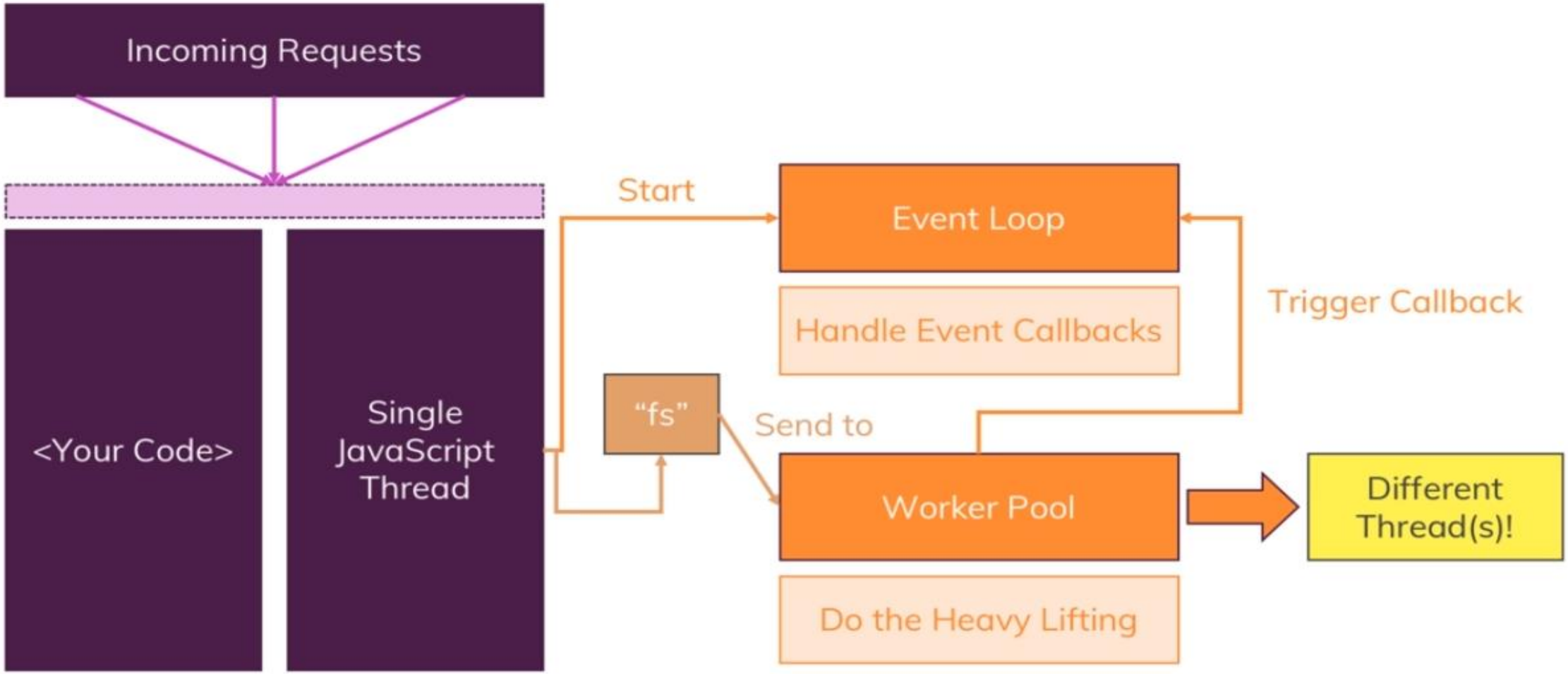


Node JS Life Cycle Event Loop





JavaScript Working with request



Node Package Manager (NPM) provides two main functionalities:

- Online repositories for node.js packages/modules.
- Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.
- NPM comes bundled with Node.js.

package.json

package.json is present in the root directory of any Node application/module and is used to define the properties of a package.

Attributes of Package.json:

- **name** – name of the package
- **version** – version of the package
- **description** – description of the package
- **homepage** – homepage of the package
- **author** – author of the package
- **contributors** – name of the contributors to the package
- **dependencies** – list of dependencies. NPM automatically installs all the dependencies mentioned here in the node_modules folder of the package.
- **repository** – repository type and URL of the package
- **main** – entry point of the package
- **keywords** – keywords

TESTYANTRA

SOFTWARE SOLUTIONS (INDIA) PVT. LTD.

Thank You !!!

No.01, 3rd Cross Basappa Layout, Gavipuram Extension,
Kempegowda Nagar, Bengaluru, Karnataka 560019

praveen.d@testyantra.com

www.testyantra.com

EXPERIENTIAL
learning factory