



# Introduction to Node JS

- JavaScript knowledge
  - Variables, functions, loops
  - Arrays
  - Native objects - Number, String, Date etc.
  - Arrow functions
  - JSON
- Familiarity with command line
- Web Concepts - Client/server, Request/response, HTTP etc.
- File handling - read/write/rename/append/delete operations
- Database concepts and SQL knowledge

# What is Node.js?

---



Node.js® is an **open-source, cross-platform JavaScript runtime environment**.

- Open-source: Source code publicly available for sharing and modification.
- Cross-platform: Runs on Windows, MAC or Linux O/S
- Runtime environment: ??

# Going back in time...



- **1993**: The first web browser with a user interface called Mosaic was released.
- **1994**: The lead developers of Mosaic founded a company called Netscape and released a browser called Netscape Navigator.
- Though this was popular, it lacked dynamic capabilities. It could only serve static web pages and there was no user interactivity.
- **1995**: In September 1995, a Netscape programmer named Brendan Eich developed a new scripting language in just 10 days. It was originally named Mocha, but later modified to LiveScript and then, JavaScript. It was used on the client-side of websites, to add dynamic and interactive capabilities to static HTML.
- In 1995, Microsoft came up with Internet Explorer, another browser competing with Netscape Navigator.

- Microsoft wanted to use JavaScript but there was no specification available for them to follow.
- **1996**: Microsoft reverse engineered the Netscape interpreter to create its own scripting language called Jscript (which was similar to JavaScript but implementation was different).
- This made it difficult for the developers to make their websites compatible with both the browsers. The badges like "Best viewed in Netscape" and "Best viewed in Internet Explorer" became common on the websites.
- **Nov 1996**: Netscape submitted JavaScript to Ecma International, industry association dedicated to the standardization of information and communication systems.

- Netscape wanted a standard specification that all the browsers conform to, so as to keep the implementation consistent across browsers.
- In case of JavaScript, the standard is called as ECMA-262.
- Ecma used a term ECMAScript to talk about the official language. So ECMAScript refers to a standard language and JavaScript is what we use in practice and builds on top of ECMAScript.

# ECMA version history



Version	Official Name	Description
ES1	ECMAScript 1 (1997)	First edition
ES2	ECMAScript 2 (1998)	Editorial changes
ES3	ECMAScript 3 (1999)	Added regular expressions Added try/catch Added switch Added do-while
ES4	ECMAScript 4	Never released
ES5	ECMAScript 5 (2009)	Added "strict mode" Added JSON support Added String.trim() Added Array.isArray() Added Array iteration methods Allows trailing commas for object literals

# ECMA version history



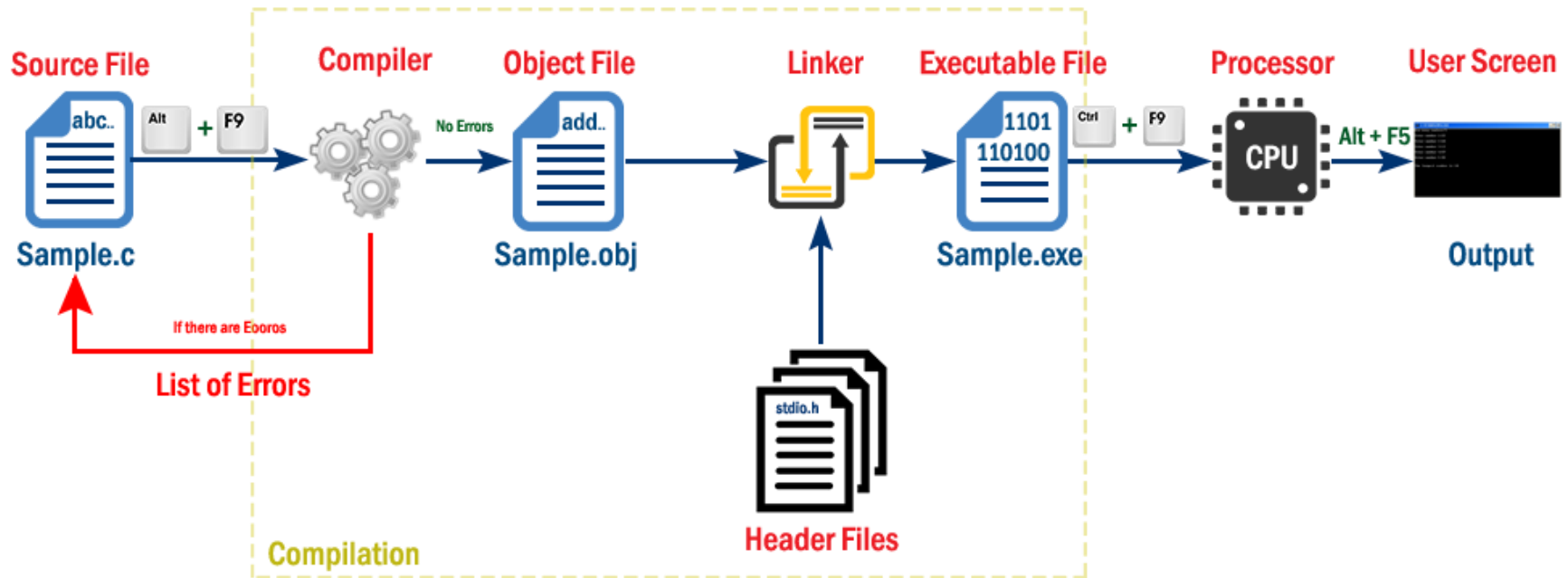
Version	Official Name	Description
ES6	ECMAScript 2015	Added let and const Added default parameter values Added Array.find() Added Array.findIndex()
	ECMAScript 2016	Added exponential operator (**) Added Array.includes()
	ECMAScript 2017	Added string padding Added Object.entries(), Object.values() Added async functions Added shared memory Allows trailing commas for function parameters
	ECMAScript 2018	Added rest / spread properties Added asynchronous iteration Added Promise.finally() Additions to RegExp
	ECMAScript 2019	String.trimStart(), String.trimEnd() Array.flat() Object.fromEntries Optional catch binding
	ECMAScript 2020	The Nullish Coalescing Operator



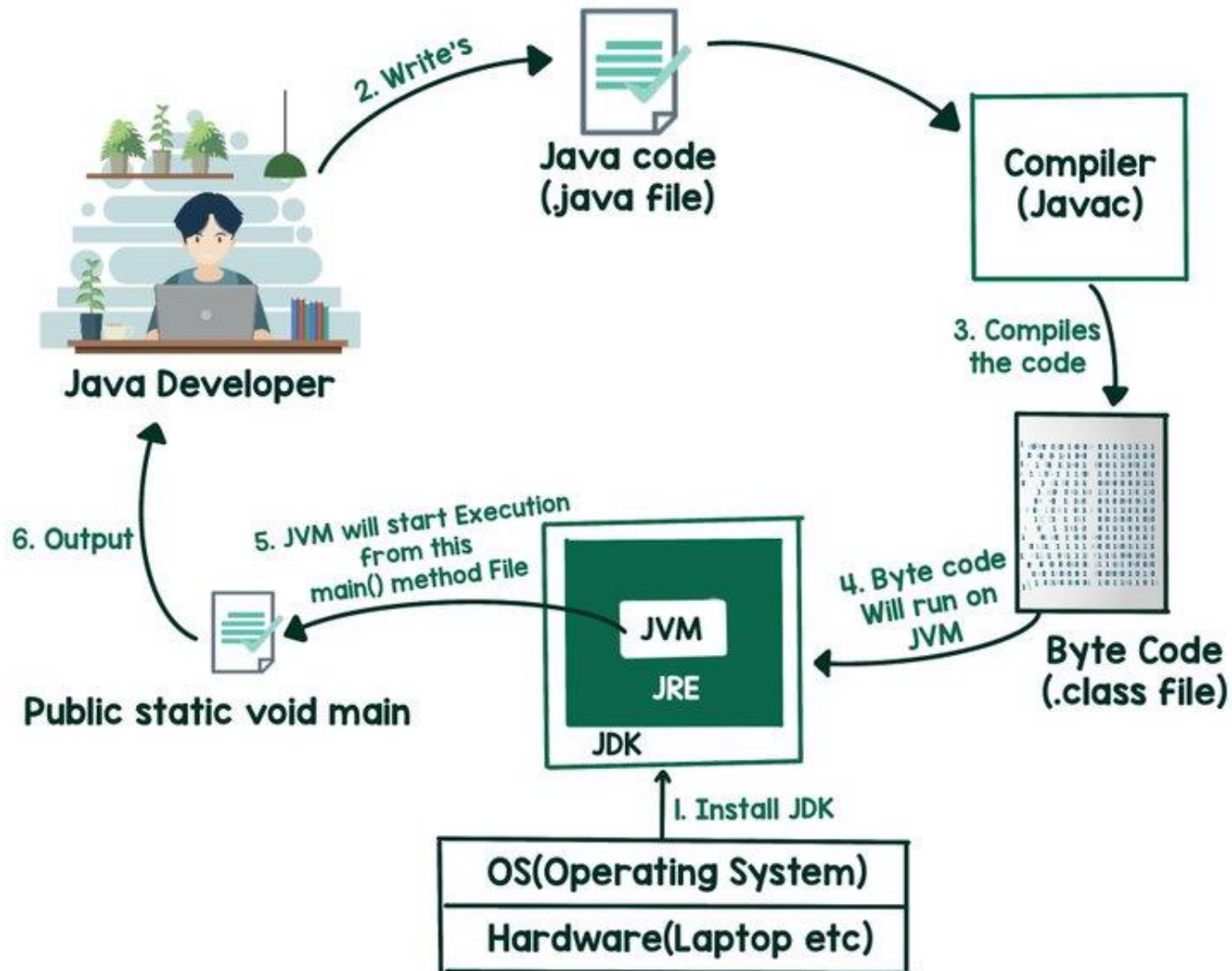
How do you compile C/C++ code?

How do you compile Java code?

# How C/C++ program works



# How Java program works



How do you compile JavaScript code?

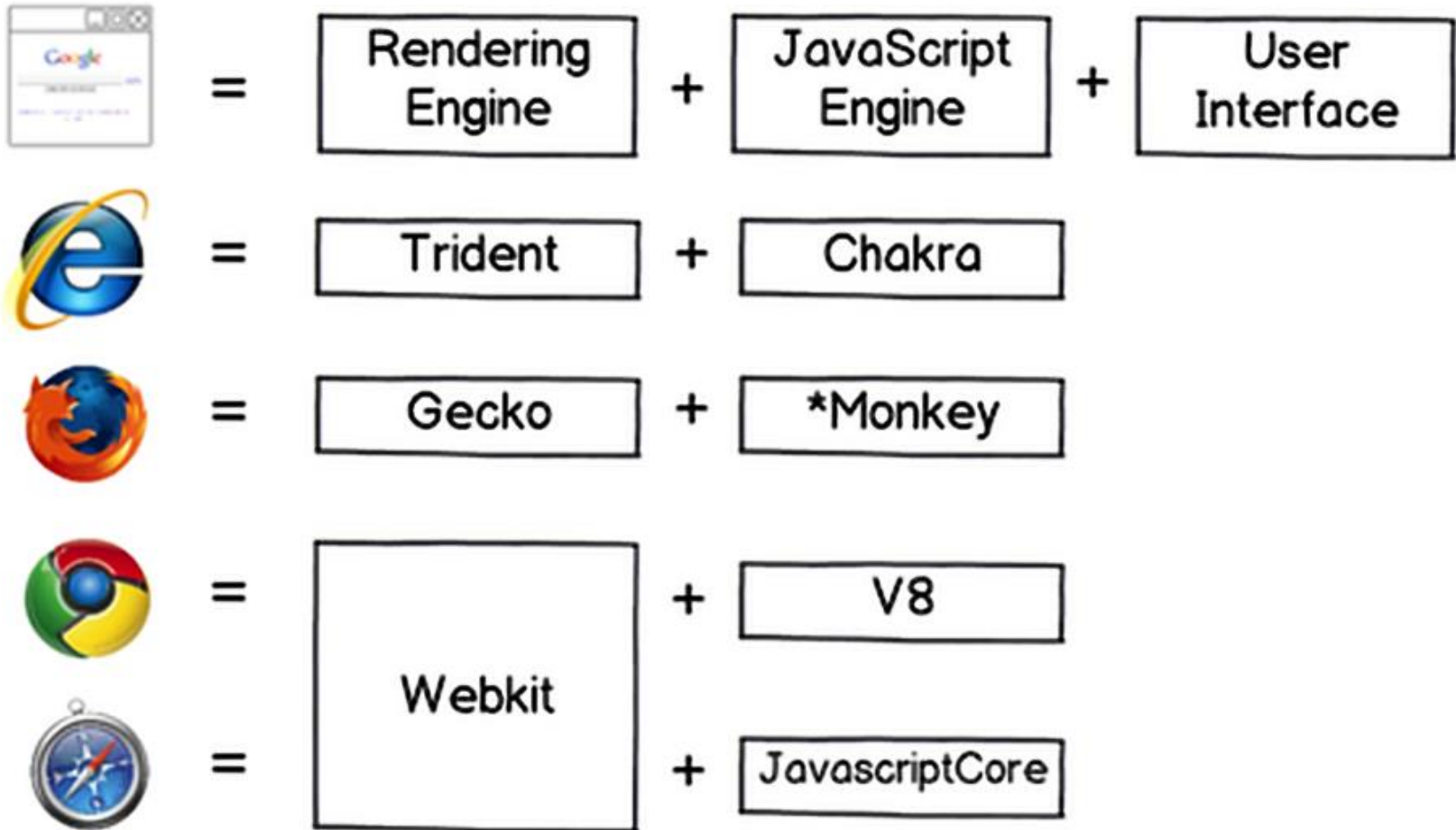
# What is JavaScript Engine?

---



- JavaScript code that we write is not directly understood by the computer/browser.
- The browsers have inbuilt JavaScript engine which helps them understand and interpret JavaScript code.
- A JavaScript engine executes JavaScript code and converts it into computer understandable language.
- JavaScript engines are typically developed by web browser vendors.
  - V8 from Google is the most used JavaScript engine, used by Google Chrome.
  - SpiderMonkey is developed by Mozilla for Firefox.
  - JavaScriptCore is Apple's engine for its Safari browser.
  - Chakra is the engine of the Internet Explorer and original edge browser. But Edge was later rebuilt and now uses V8.

# Browsers and Engines



# What is JavaScript Engine?

---



- In 2008, Google created its own JavaScript engine called V8.
- V8 is written in C++ and can be used as standalone or can be embedded into other C++ programs.
- This allows you to write your own programs that can do everything that V8 can do and much more. Your program can run ECMAScript and additional features that are available with C++ but not available with JavaScript.
- Additional features include - file handling, database connections etc.
- For more information -
  - <https://v8.dev/docs>
  - <https://github.com/v8/v8>

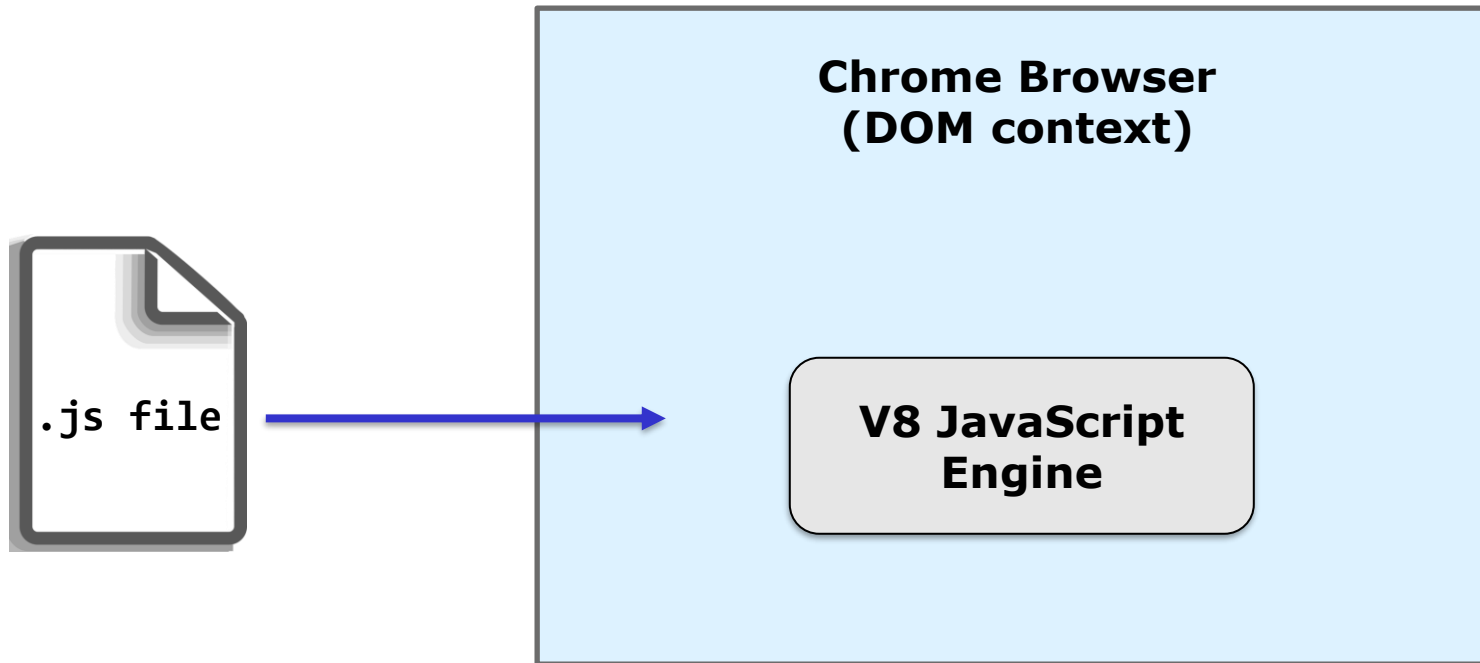
# What is JavaScript runtime?

---

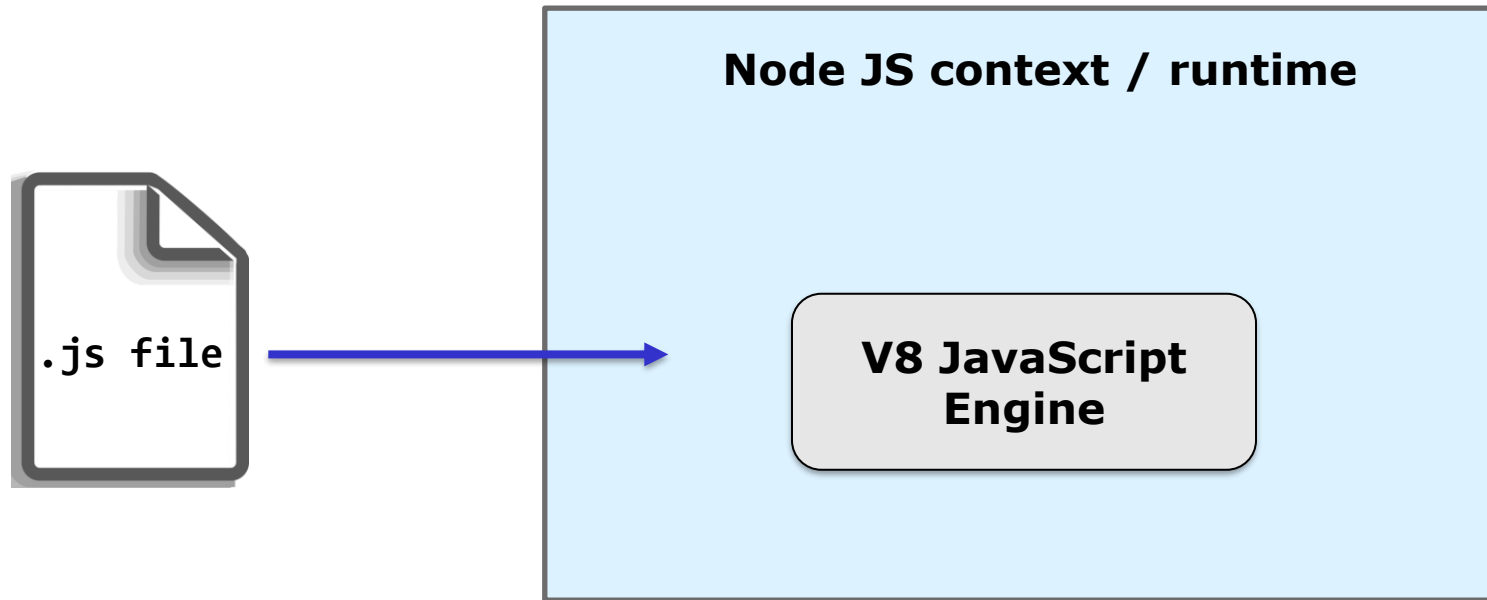


- Every browser has a JavaScript engine. But this engine can not run in isolation, it needs a context.
- For a web browser, this context is nothing but DOM (Document Object Model).
- Chrome's V8 JavaScript engine runs in the context of Chrome browser and can access DOM.
- JavaScript runtime is an environment which provides all the necessary components in order to use and run a JavaScript code.





DOM context provides limited access to JavaScript engine.  
i.e. window and document objects only.



- In Node.js, V8 engine is taken out of the browser and given a Node JS context.
- Node JS runtime provides an access to -
  - System resources
  - Memory, file system, database, network API and more

Node JS is JavaScript runtime built on Chrome's V8 JavaScript engine.

# Definition - What is Node.js?

---



Node.js® is an **open-source, cross-platform** JavaScript **runtime environment**.

- Open-source: Source code publicly available for sharing and modification.
- Cross-platform: Runs on Windows, MAC or Linux O/S
- Runtime environment: Provides all the necessary components in order to use and run a JavaScript code outside a browser.

# What is Node.js?

---



To summarize -

- Node.js is a JavaScript runtime environment.
- Node.js is not a language and not a framework.
- Capable of executing JavaScript code outside a browser.
- In Node.js, you don't have document, window and all other objects that are provided by the browser.
- It makes use of C++ and JavaScript features.
- It can execute not only standard ECMAScript code but also new features that are made available through C++ using v8 engine.

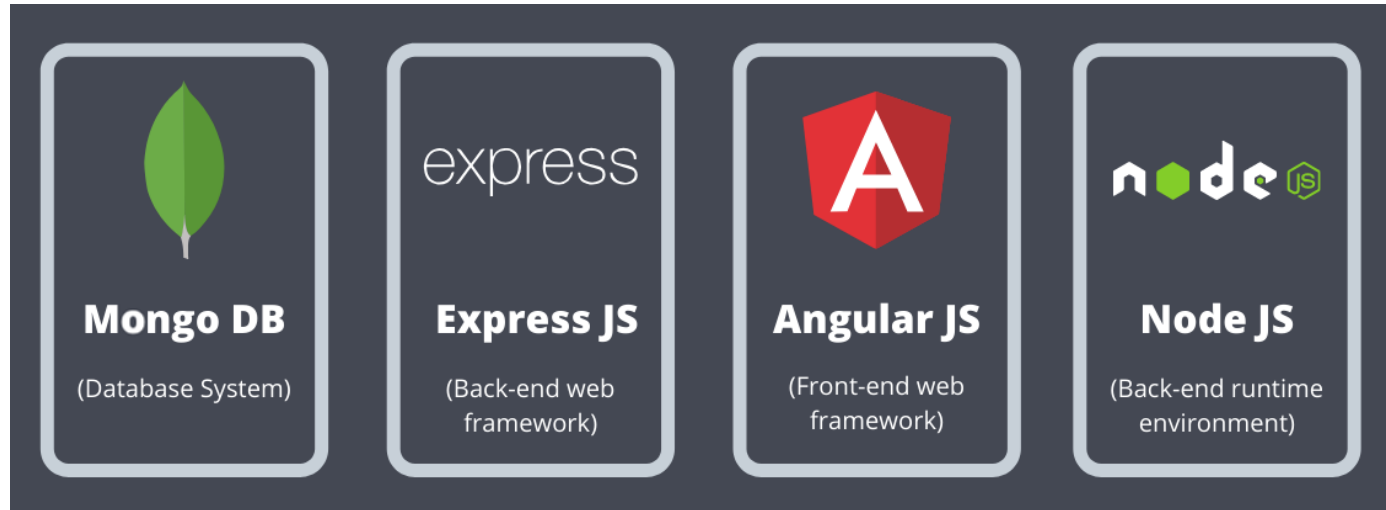
- **Full Stack** technology refers to a set of tools and programming languages that a full stack developer works on both the front-end and the back-end of an application or a website.
- Full Stack developers work on “Full Stack” of an application i.e. Front End technologies, Back End development, Version Controlling Systems, Servers, APIs, and Database etc.
- Full Stack comprises of four critical components:
  - A) Frontend
  - B) Backend
  - C) Testing
  - D) Mobile app

- Frontend: CSS, Single page applications using HTML5, Ajax, JavaScript, Angular, React, TypeScript, etc.
- Backend: Python, PHP, Node JS, Express JS, Django, Middleware, etc.
- Database: Postgres, MySQL, MongoDB, MS SQL Server etc.
- Mobile Apps: iOS and Android
- DevOps: CI/CD pipelines

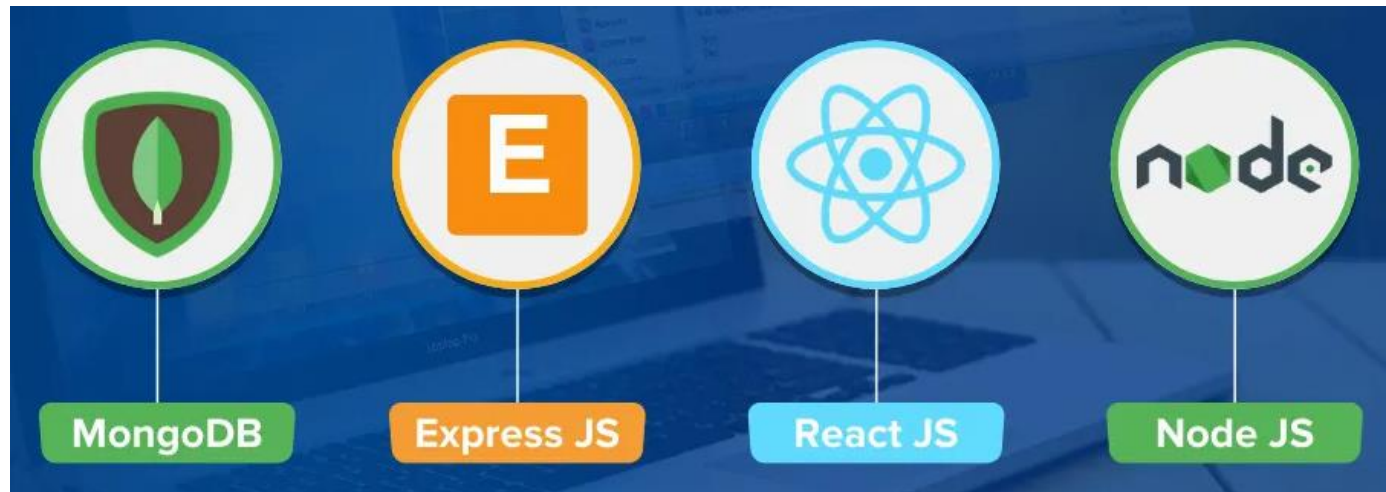
# MEAN Stack vs MERN Stack



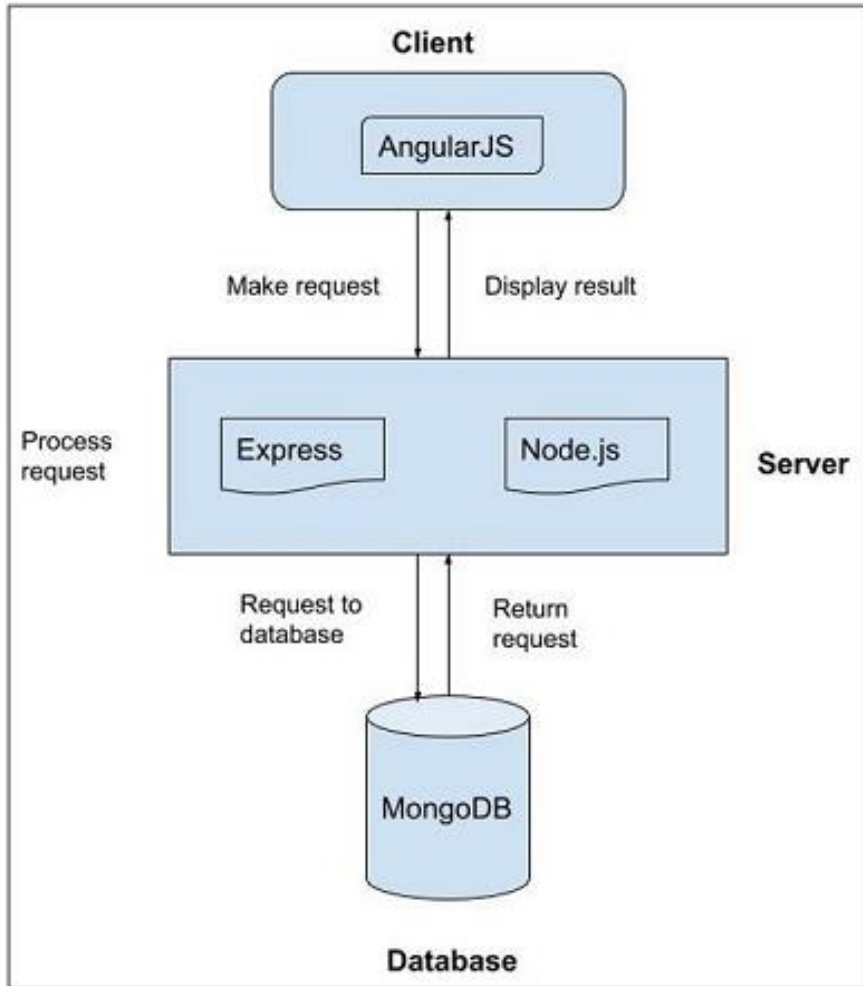
MEAN stack



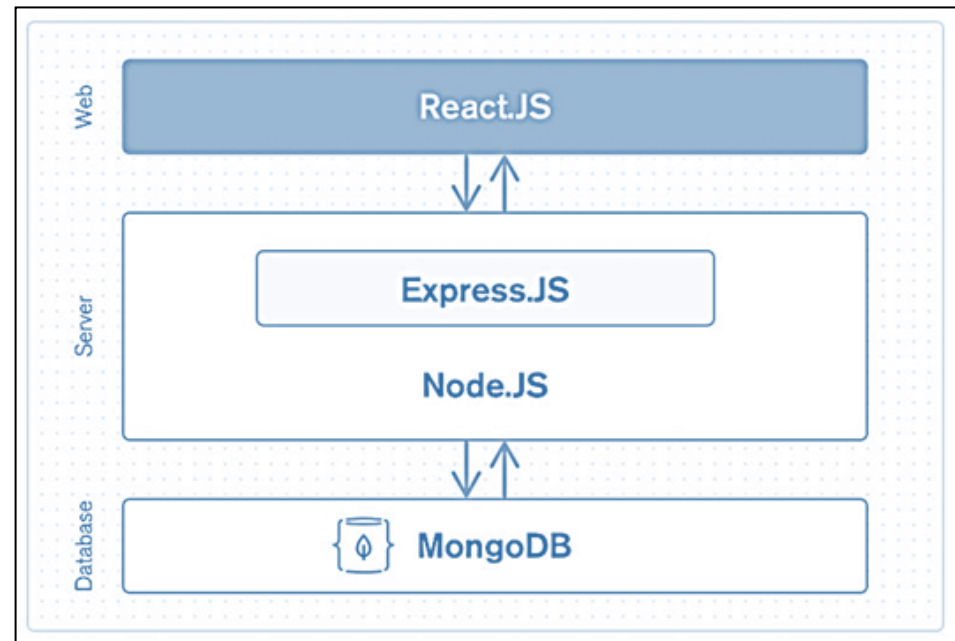
MERN stack



# MEAN Stack vs MERN Stack



MEAN stack



MERN stack



# Who uses Node.js?

---



Walmart 

ebay

YAHOO!

 PayPal

NETFLIX



 Trello

  
UBER

GROUPON

Linked 

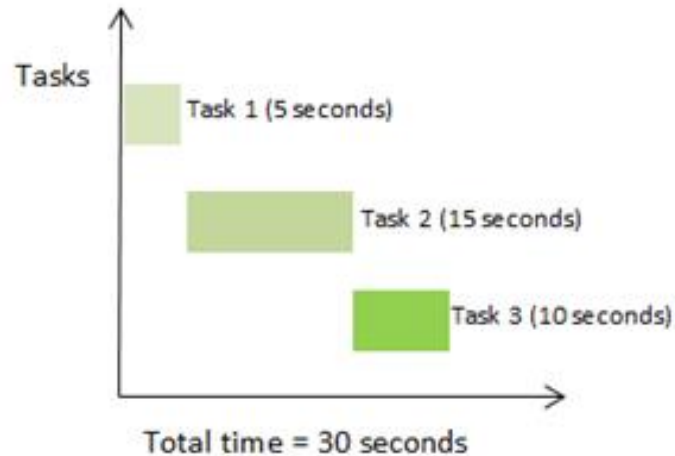
# What is synchronous and asynchronous?

---

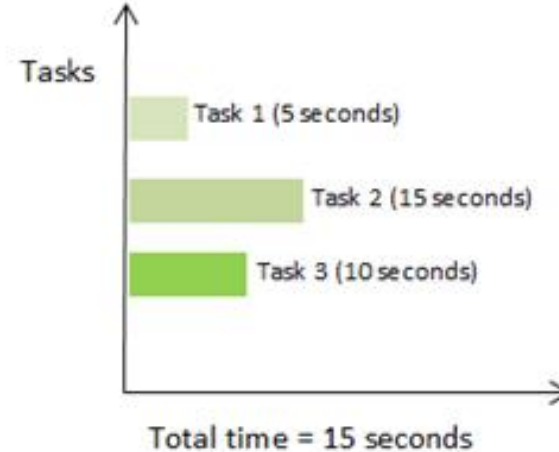


- Synchronous code is also called as "blocking". It halts the program execution until all the resources are available.
- Synchronous execution means the code is executed in sequence line by line (one line at a time).
- For example, when a function is called, the program execution waits until that function returns before going to the next line of code.
- Asynchronous code is also called as "non-blocking". The program continues its execution and doesn't wait for external resources (I/O) to be available.
- Asynchronous execution means the code doesn't necessarily run in the sequence. The program doesn't wait for the code block to finish its execution and can move on to the next piece of code.

# What is synchronous and asynchronous?



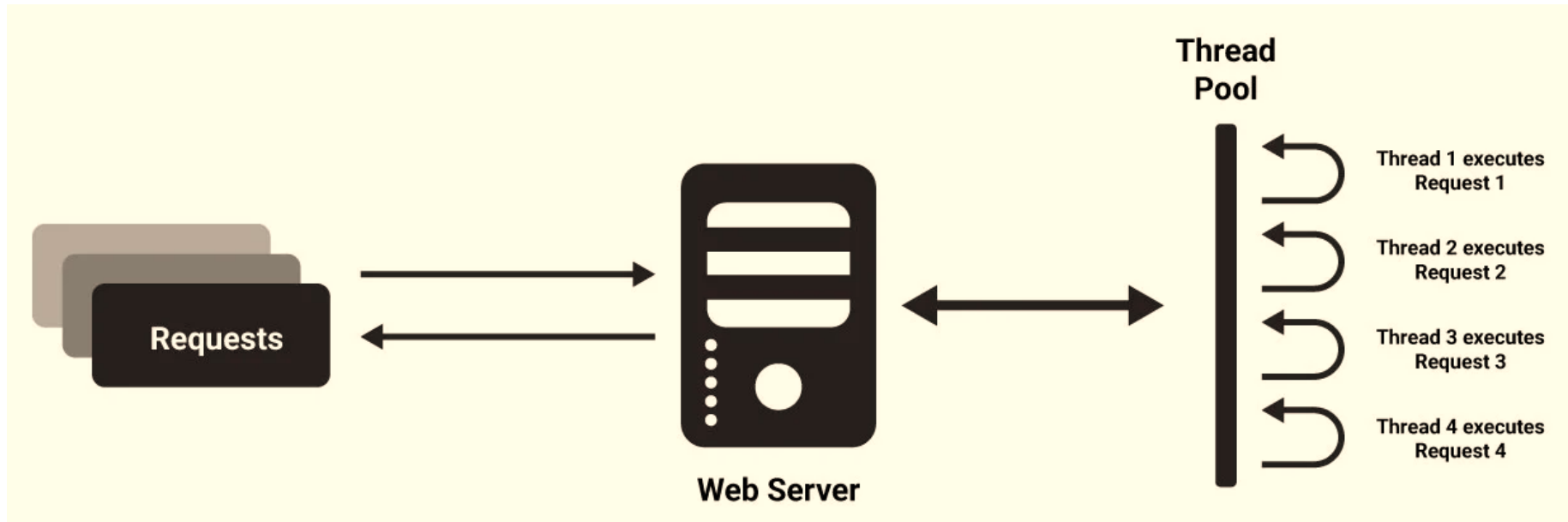
**Synchronous**



**Asynchronous**

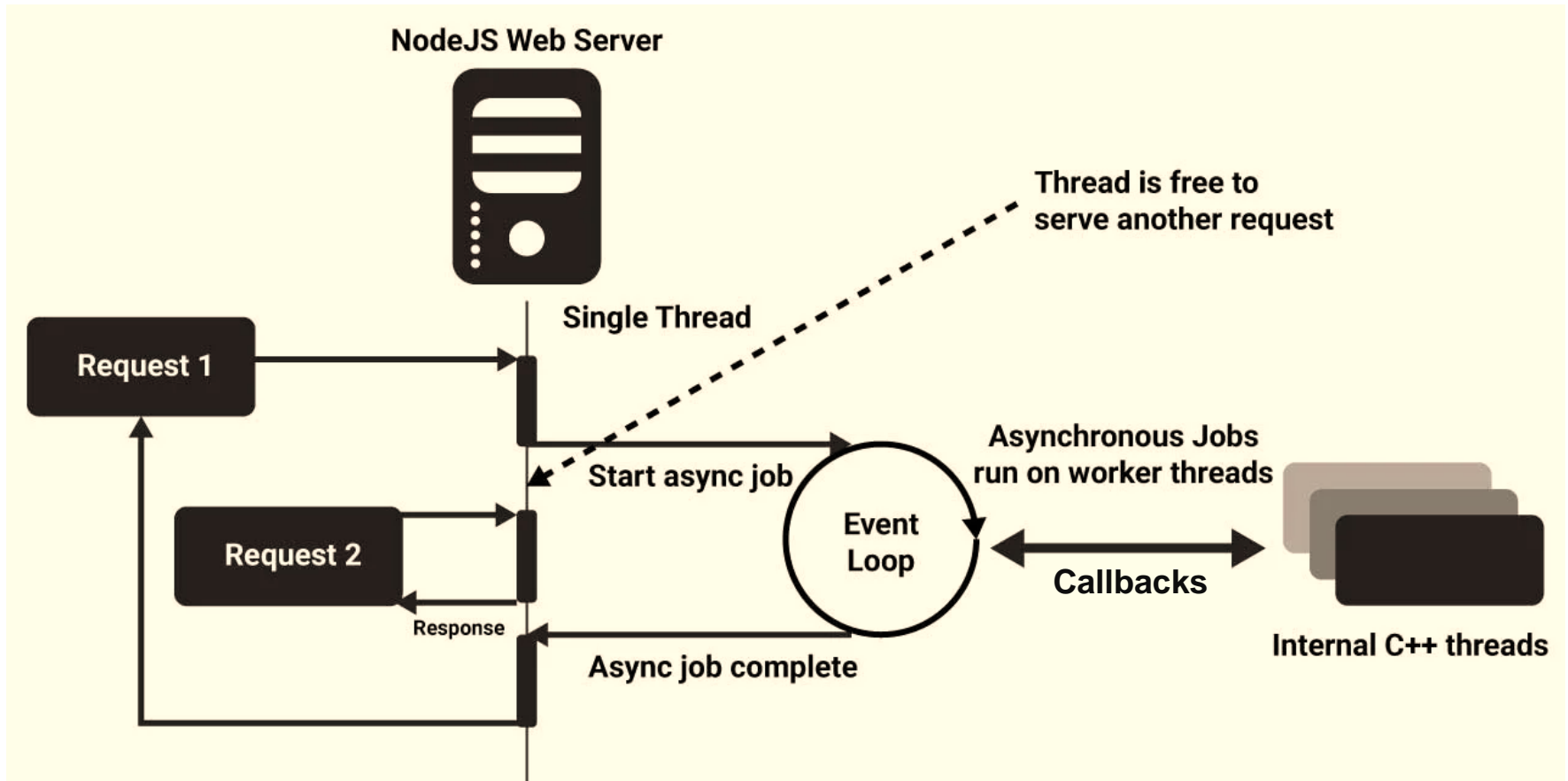
- Synchronous code wastes around 90% of CPU cycles waiting for the network or disk to get the data, whereas Asynchronous code is much more performant.
- Asynchronous code is a more efficient to achieve concurrency without dealing with multiple execution threads.

# Traditional Web Server Model

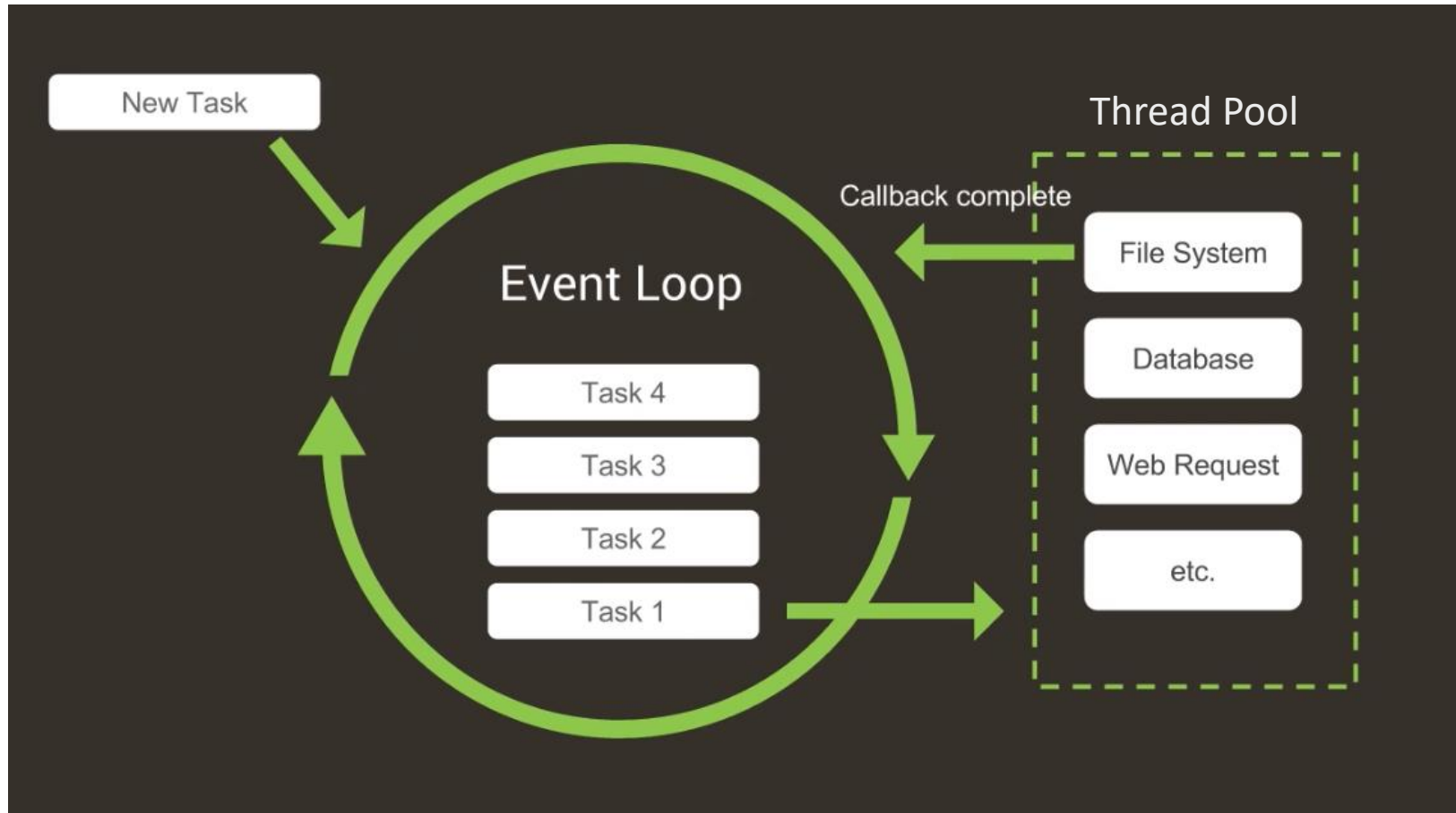


- Pool of threads to process requests
- Each new request is assigned to a different thread in the pool
- If the new request comes in and the thread is not available, the request will have to wait until a previous request finishes (i.e., response is returned and thread returns to the pool)
- This model is synchronous and blocking.

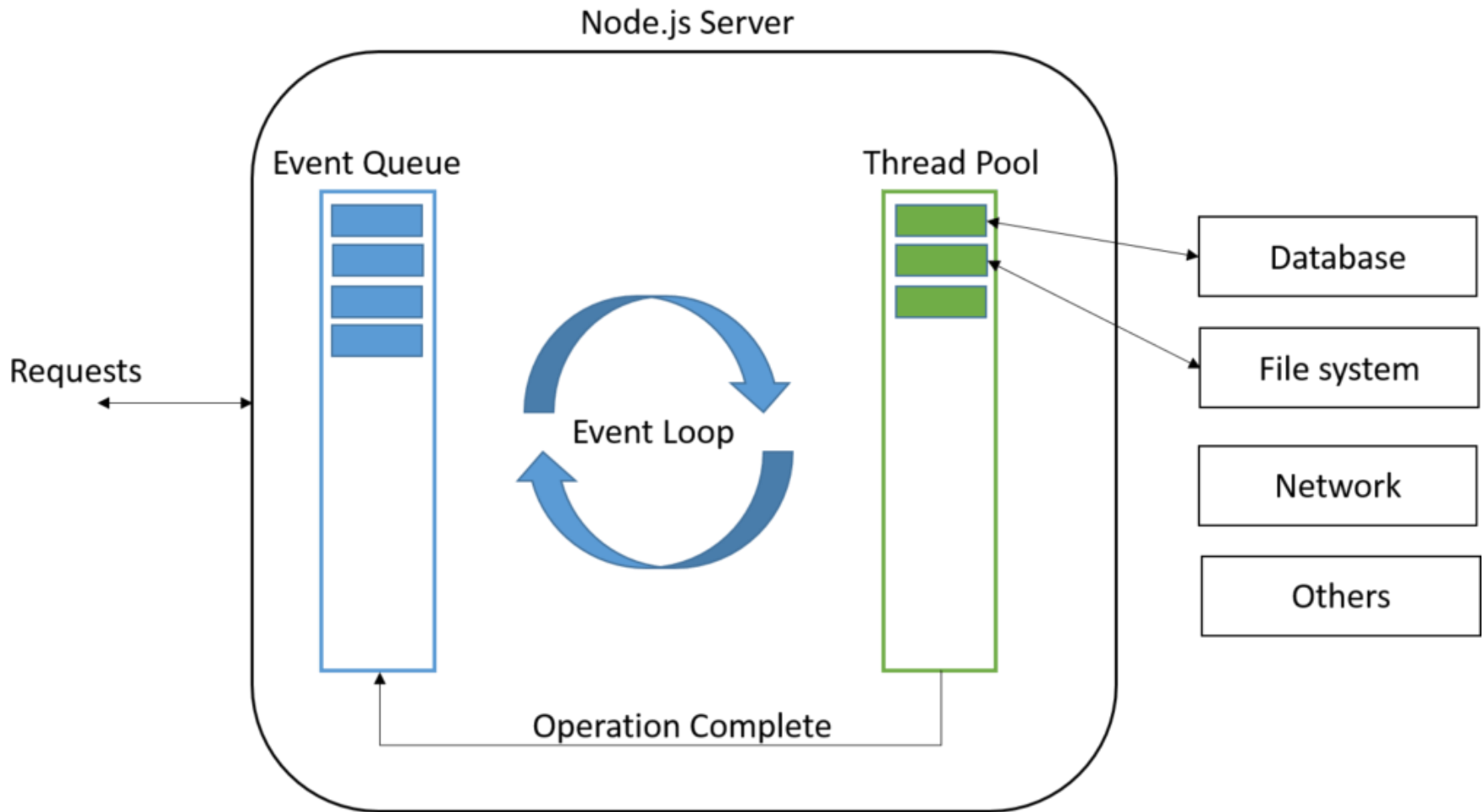
# Node.js Process Model



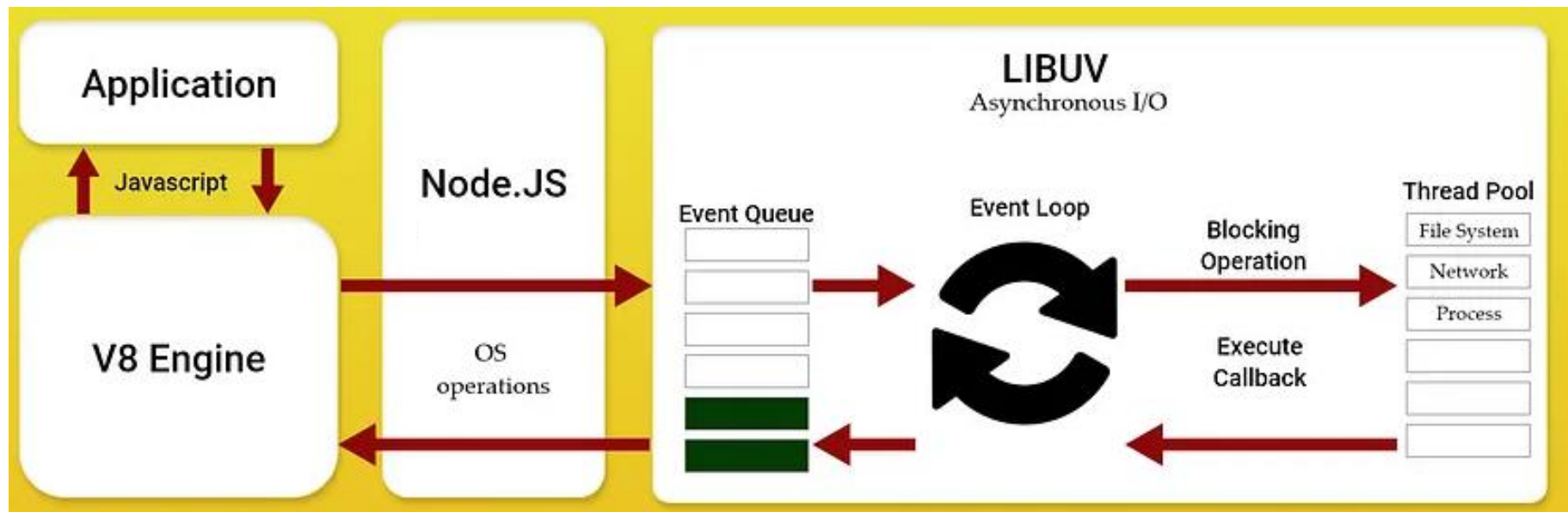
# Node.js Process Model



# Node.js Process Model

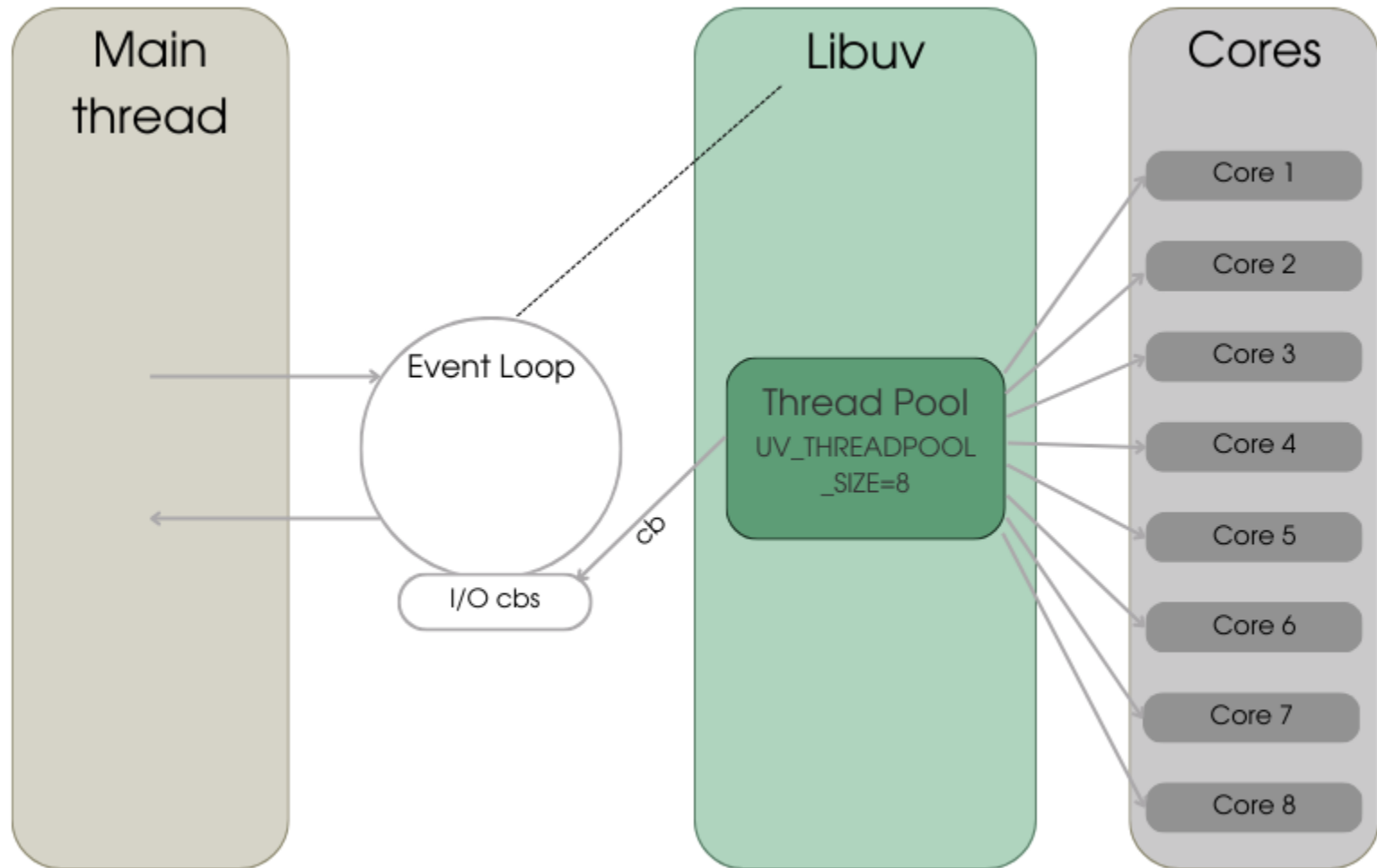


# Node.js Process Model





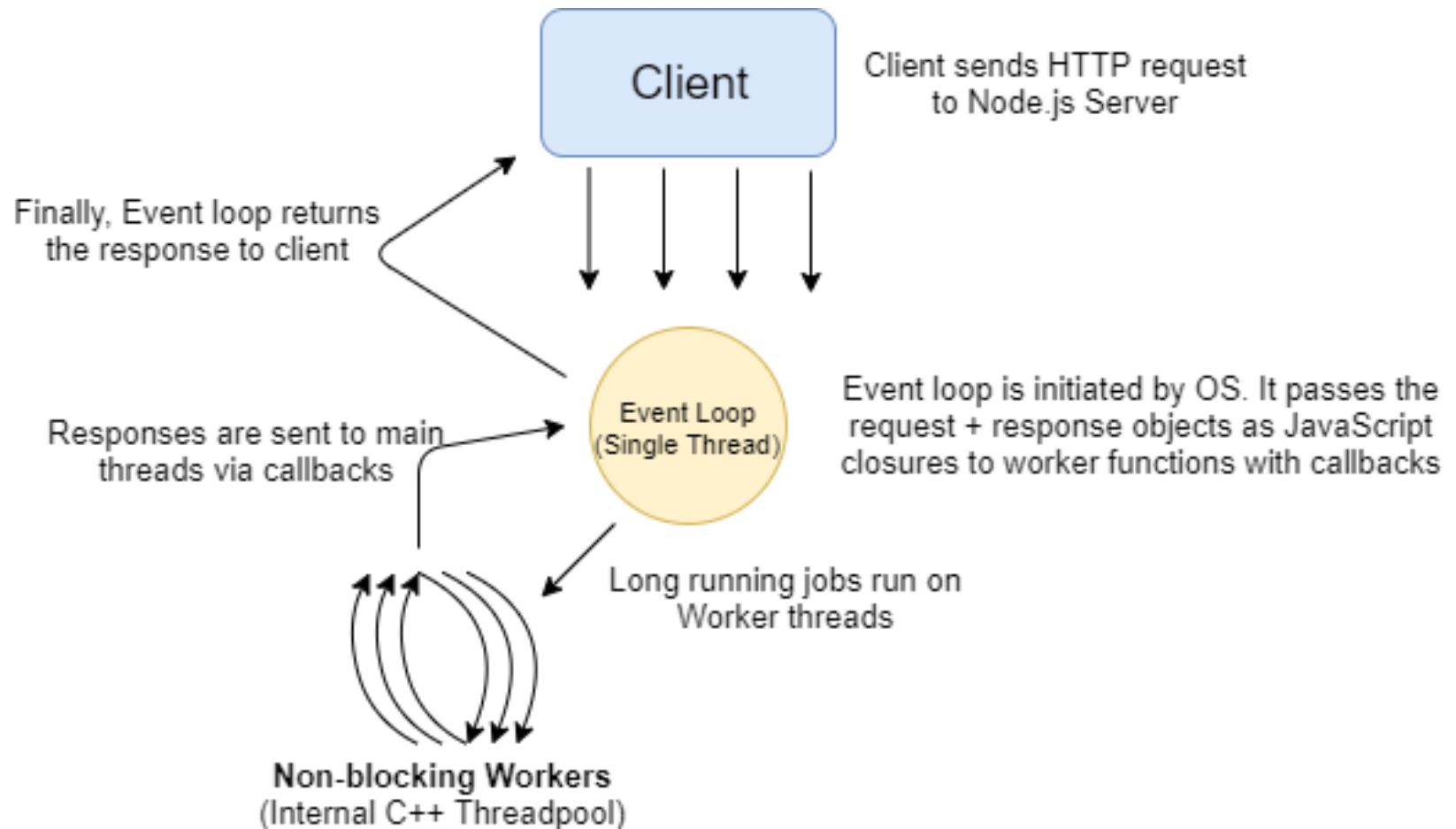
# Node.js Process Model



- Node.js runs in a single process with requests being processed on a single main thread (application thread)
- When a request comes in, it will be placed in an event queue.
- Node.js uses an event loop which continuously runs, receiving requests from the event queue.
- If the request is non-blocking, (i.e. it does not involve any long-running processes or data requests), the response will be immediately prepared and then sent back to the client.
- If the request is blocking, (i.e. requiring I/O operations), the request will be sent to a worker thread pool.

- Node.js uses internal C++ thread pool to provide asynchronous I/O.
- The request will have an associated callback function that will fire when the request is finished and the worker thread can send the response to the event loop. This response is then sent back to the client.
- The single main thread (application thread) receives a request, it hands it off so that it can process other requests in the meantime. In this way, Node.js is inherently asynchronous.

# Node.js Process Model



# Features of Node.js

---



- Node JS is built on Chrome's V8 JavaScript engine
- Node.js is open-source under MIT license  
*(MIT license is a free software license originating at the Massachusetts Institute of Technology in Cambridge)*
- Uses JavaScript to build server side applications
- Cross-platform that runs on Windows, MAC or Linux O/S
- Event driven in nature, uses callbacks. Callback functions require fewer resources on the server side and also take up less memory. Due to this feature, Node JS applications are lightweight.
- Asynchronous (non-blocking) and single threaded in nature, leads to faster response time

# When to use Node.js and when not to?

---



- Non blocking
- Event driven
- Data intensive
- I/O intensive



- Data calculations
- Processor intensive
- Blocking operations



## Home Work

- What is JavaScript engine? Explain with examples
- What is JavaScript runtime? Compare DOM context and Node.js context/runtime
- What is Node.js? Write a short note on features of Node.js



## Home Work

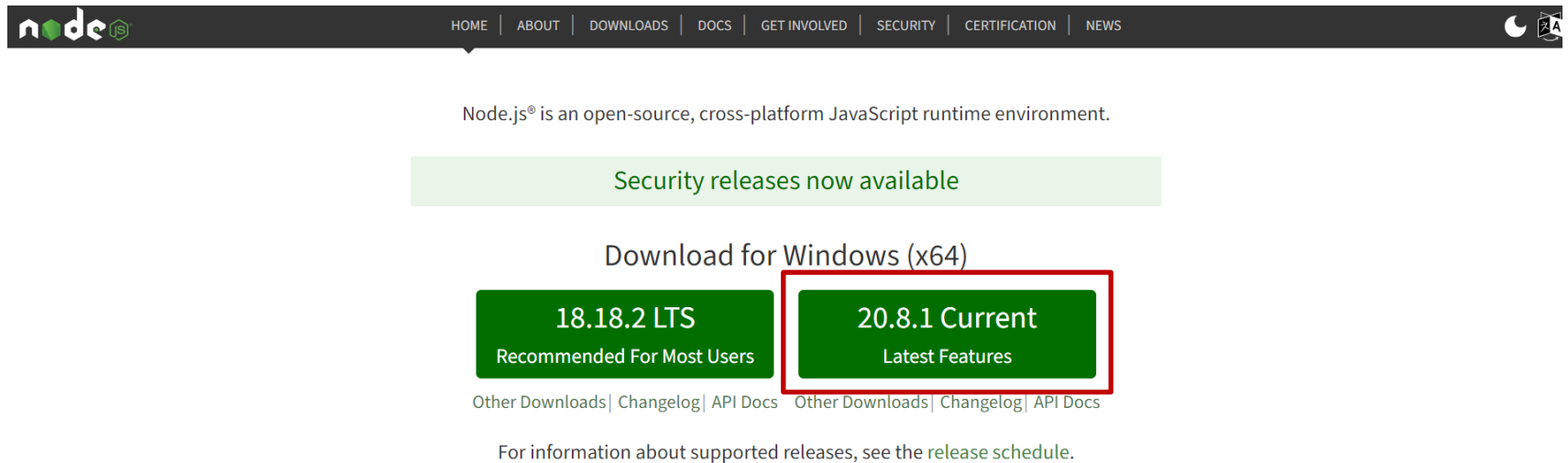
- Explain blocking and non-blocking operations/approach  
OR Explain synchronous and asynchronous operations/approach
- Explain Traditional Web Server model (with diagram)
- Explain Node.js Process model (with diagram)
- What is REPL? Explain the uses of REPL



# Install Node.js on Windows



- Go to <https://nodejs.org>
- Download the installer for Windows

A screenshot of the Node.js website homepage. The header is dark grey with the Node.js logo on the left and navigation links (HOME, ABOUT, DOWNLOADS, DOCS, GET INVOLVED, SECURITY, CERTIFICATION, NEWS) in the center. On the right of the header is a moon icon and a flag icon. Below the header, a light green banner reads "Security releases now available". The main content area has the heading "Download for Windows (x64)". Below this heading are two green buttons: "18.18.2 LTS Recommended For Most Users" and "20.8.1 Current Latest Features". The "20.8.1 Current" button is highlighted with a red border. Below the buttons are two links: "Other Downloads | Changelog | API Docs". At the bottom, a line of text reads "For information about supported releases, see the release schedule.".

Node.js® is an open-source, cross-platform JavaScript runtime environment.

Security releases now available

Download for Windows (x64)

**18.18.2 LTS**  
Recommended For Most Users

**20.8.1 Current**  
Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)   [Other Downloads](#) | [Changelog](#) | [API Docs](#)

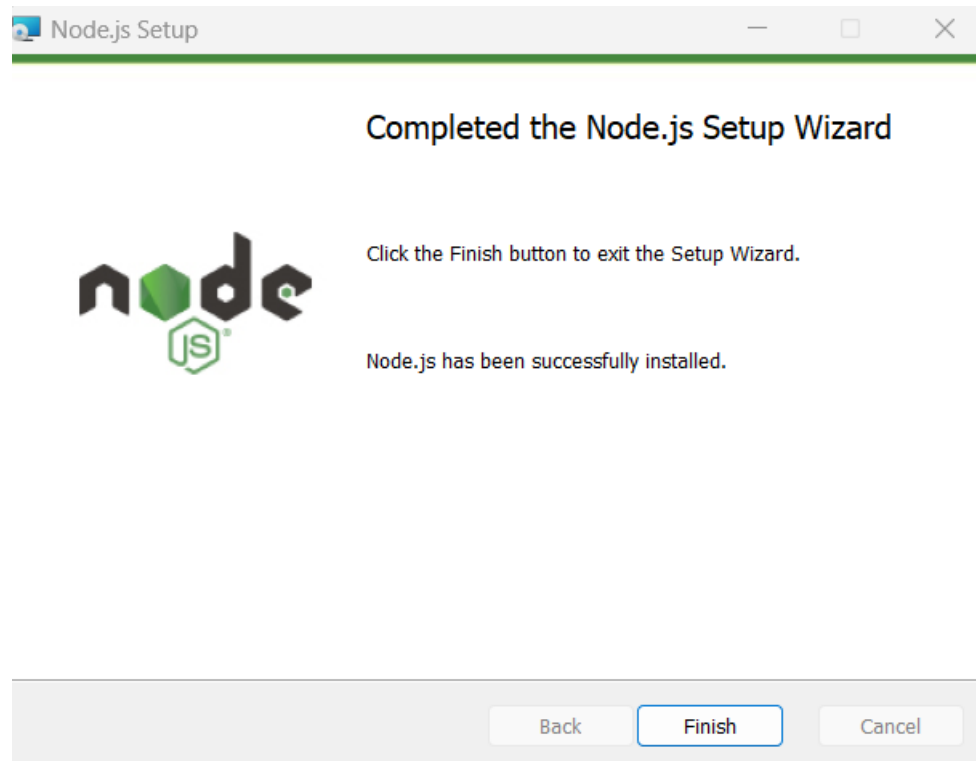
For information about supported releases, see the [release schedule](#).

LTS = Long Term Support Schedule

# Install Node.js on Windows



Execute the downloaded Windows installer and complete the setup



# Install Node.js on Windows



## Verify the installation:

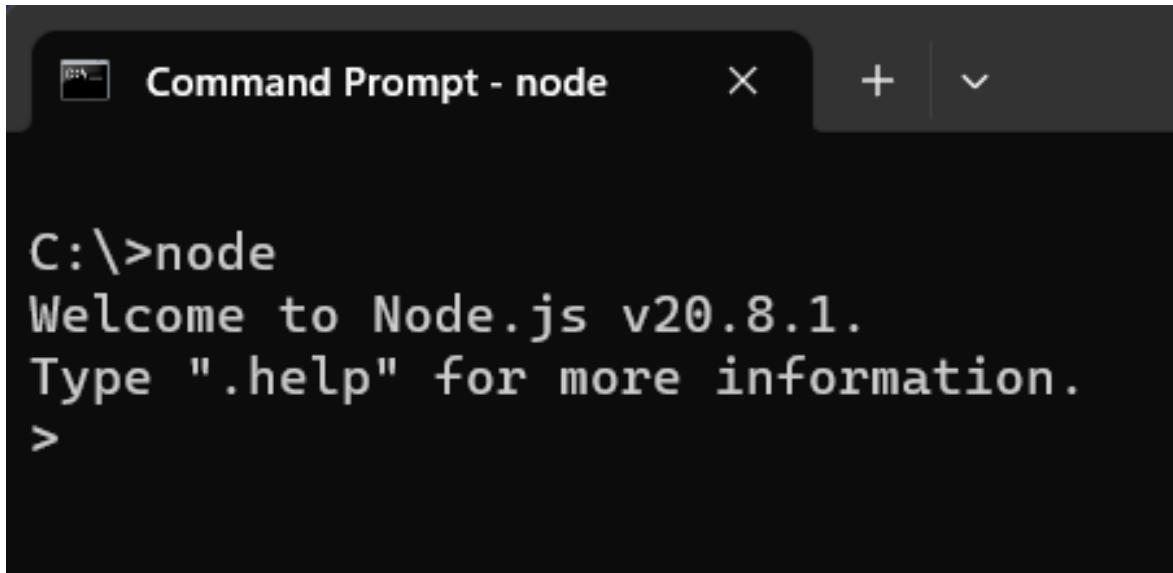
- Open command prompt and type **node -v**
- If Node.js is installed successfully then it will display the version of the Node.js installed on your machine.

A screenshot of a Windows Command Prompt window. The title bar says "Command Prompt" with a close button on the right. The command prompt shows the text "C:\Users>node -v" on one line and "v20.8.1" on the line below it, indicating a successful installation of Node.js version 20.8.1.

- If Node.js is not installed successfully (or the path is not set) then it will show an error - "**command not found**"

- Node.js comes with a virtual environment called REPL (aka Node interactive shell).
- REPL stands for Read-Eval-Print-Loop.
- REPL performs following tasks:
  - Read - Reads user's input, parses the input into JavaScript data-structure, and stores in memory.
  - Eval - Takes and evaluates the data structure.
  - Print - Prints the result.
  - Loop - Loops the above command until the user presses ctrl + c twice.

- It is a quick and easy way to test simple Node.js (JavaScript) code.
- To launch the REPL (or Node shell), open command prompt (in Windows) or terminal (in Mac or UNIX/Linux) and type node as shown below. It will change the prompt to **>** in Windows



```
C:\>node
Welcome to Node.js v20.8.1.
Type ".help" for more information.
>
```

- You can test JavaScript expressions
- Define variables and perform operations
- Define a function and execute it
- You can execute an external JavaScript file by executing the `node` "filename" command

```
Command Prompt - node
C:\>node
Welcome to Node.js v20.8.1.
Type ".help" for more information.
> "Hello" + " World"
'Hello World'
> 10 + 20
30
> var a = 10, b = 20;
undefined
> a + b
30
>
```

```
Command Prompt - node
C:\>node
Welcome to Node.js v20.8.1.
Type ".help" for more information.
> function multiply(a, b)
... {
... return a * b;
... }
undefined
> multiply(10, 20)
200
>
```

## Few important REPL commands

REPL Command	Description
<code>.help</code>	Display help on all the commands
<code>tab</code> Keys	Display the list of all commands
<code>Up/Down</code> Keys	See previous commands applied in REPL
<code>.save filename</code>	Save current Node REPL session to a file
<code>.load filename</code>	Load the specified file in the current Node REPL session
<code>ctrl + c</code>	Terminate the current command
<code>ctrl + c</code> (twice)	Exit from the REPL
<code>ctrl + d</code>	Exit from the REPL
<code>.break</code>	Exit from multi-line expression
<code>.clear</code>	Exit from multi-line expression



## Class Work

Play around with Node REPL, try out different things -

- Test JavaScript expressions
- Define variables and perform operations
- Define and execute a function
- Execute an external JavaScript file
- Backtick syntax





## Class Work

- Write a calculator program which performs add, subtract, multiply, divide operations in separate functions and displays the result.
- Write a program that prints today's date in this format as **Today's date is: 16-Dec-2023**
- Write a program that displays whether the year is a leap year or not.

Hint: The year is a multiple of 4 and not a multiple of 100.  
The year is a multiple of 400.



## Class Work

- Write a program that initializes a string array to store 'subjects'. Check if the particular value exists in the array or not.
- Write a program to create countdown timer for 1 minute (60 seconds).
- Write a program to display today's day as well as current time in below format.

Output: Today is: Saturday  
Current time is: 11:00:00 AM