BCDV 1022
Introduction to Node

2023 Fall

week 01 - class 01

Topics

- V8 Engine
- Intro to Node.js
- Node Core Architecture
- Installing Node.js

V8 Engine

V8 Google

WHAT ?

2 of 13

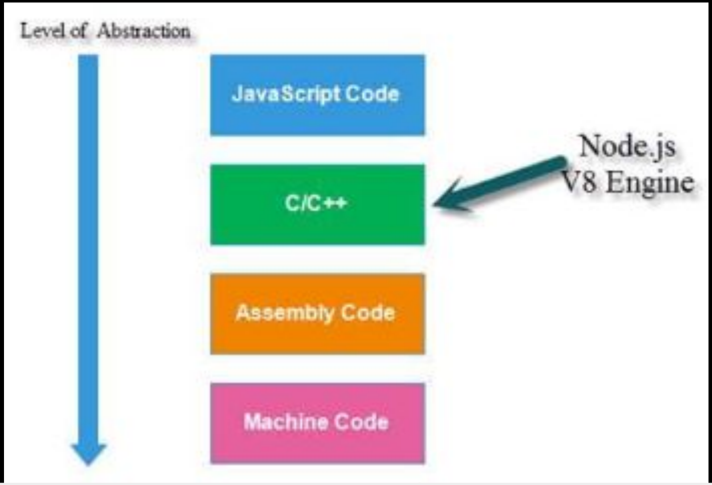
Chrome V8

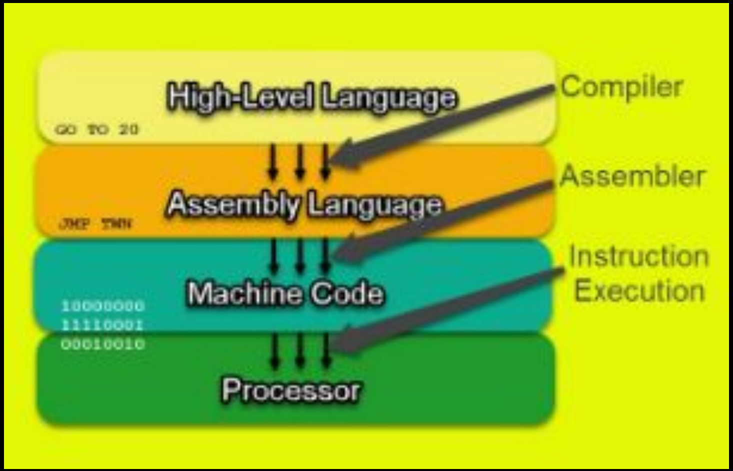
https://v8.dev/

- V8 is Google's open source high-performance JavaScript and WebAssembly engine, written in C++. It is used in Chrome and in Node.js, among others.
- It implements ECMAScript and WebAssembly, and runs on Windows 7 or later, macOS 10.12+, and Linux systems that use x64, IA-32, ARM, or MIPS processors.
- V8 can run standalone, or can be embedded into any C++ application.

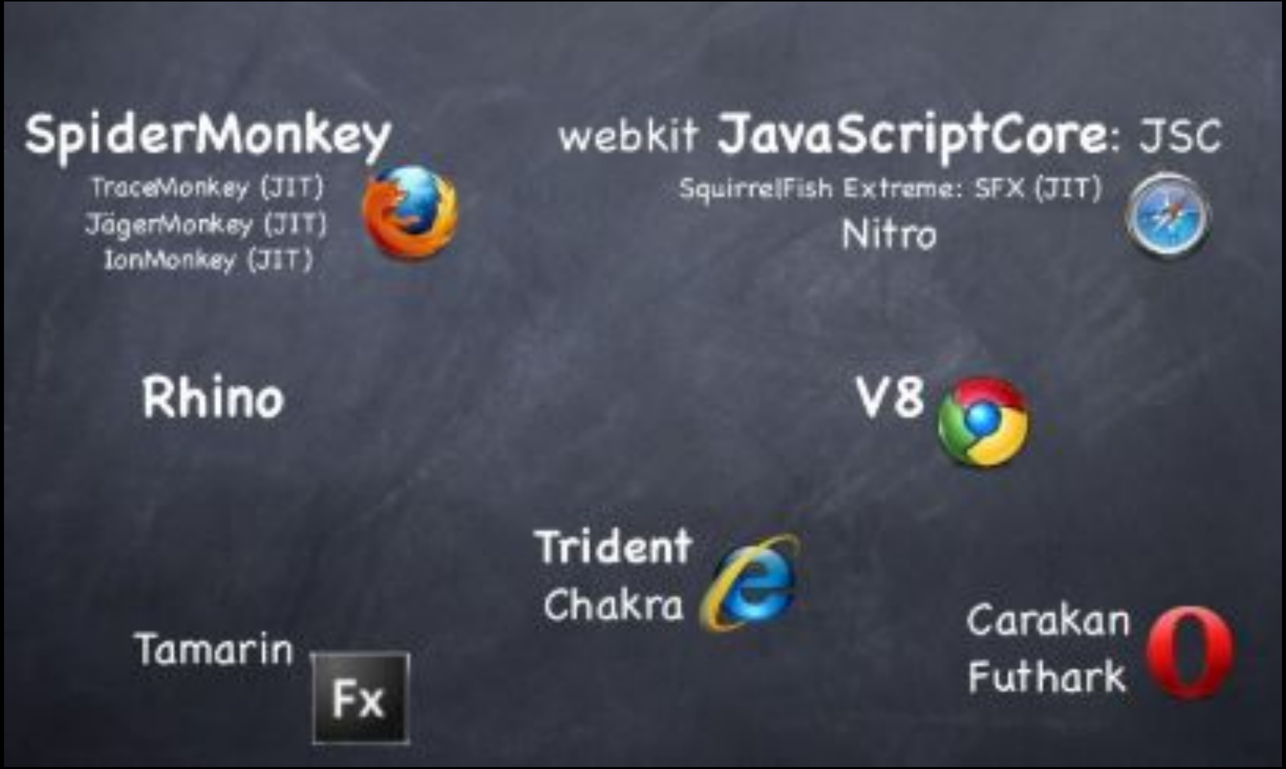
How V8 Translates Javascript into Machine Code

- JavaScript in Node is converted to machine code.
- Machine code is instructions that the process can understand.
- V8 is written in C++





Javascript Engines



Intro to Node.js

Intro to Node.js

- Believe it or not, Node.js is only thirteen years old.
- In comparison, JavaScript is 24 years old and the Web is 30 years old.
- Ten years isn't a very long time in tech, but Node.js seems to have been around forever.
- Node.js event driven approach was very innovative. In the early days you could tell it was going to be very popular

Node.js History

2009

- Node.js is born
- The first form of npm is created

2010

- Express is born
- Socket.io is born

2011

- npm hits 1.0
- Big companies start adopting Node: LinkedIn, Uber Hapi is born

Node.js History cont..

2012

- Adoption continues very rapidly

2015

- The Node.js Foundation is born
- IO.js (major fork ES6) is merged back into Node.js
- Node 4

2016

- The leftpad incident
- Yarn is born *(created by Facebook, another package manager. Better performance?)*
- Node 6

Node.js History cont..

2017

- npm focuses more on security
- Node 8
- HTTP/2
- V8 introduces Node in its testing suite, officially making Node a target for the JS engine, in addition to Chrome
- 3 billion npm downloads every week

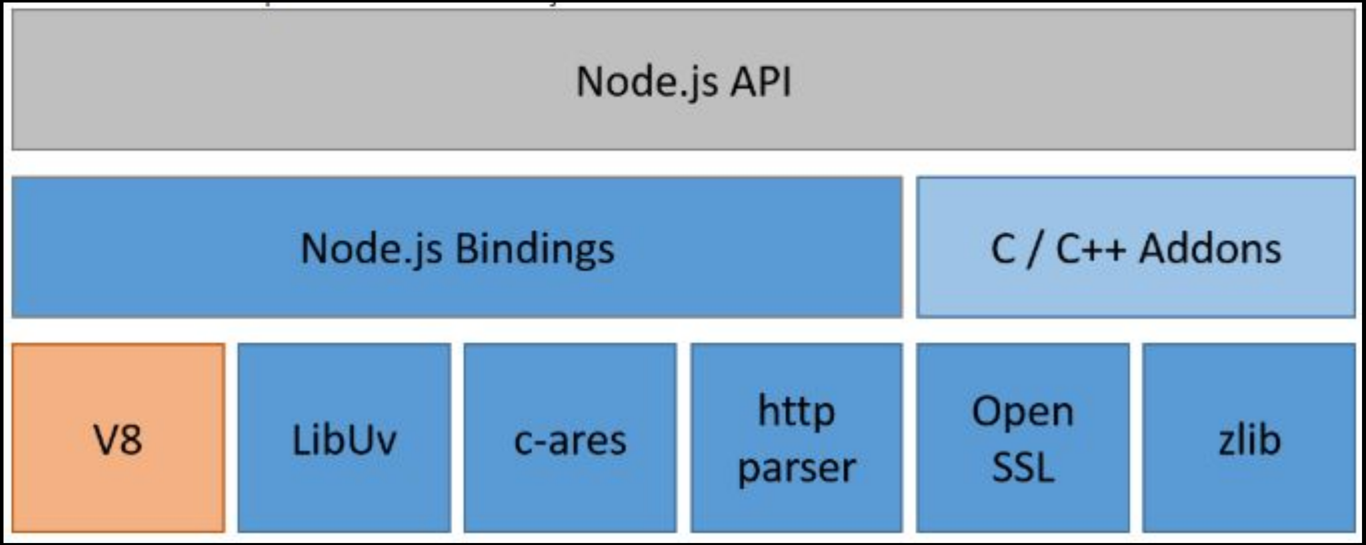
2018

- Node 10
- ES modules .mjs experimental support

BCDV 1021 - Front End Web Development
Mike Denton/Andrew Starling

Node Core

Node.js Architecture



Node.js Architecture cont..

libuv
(Unicorn Velociraptor Library)

- libuv - libuv is originally developed to provide asynchronous I/O that includes asynchronous
    - TCP & UDP sockets,
    - (famous) event loop,
    - asynchronous DNS resolution,
    - file system read/write, and etc.
- libuv is written in C.

Node.js Architecture cont..

- Other Low-Level Components - such as c-ares, http parser, OpenSSL, zlib, and etc, mostly written in C/C++.
- Application (API)- here is your code, modules, and Node.js' built in modules, written in JavaScript (or compiled to JS through TypeScript, CoffeeScript, etc.)
- Binding - basically wrappers or bridges between codes, so different languages can communicate.

Installing Node.js

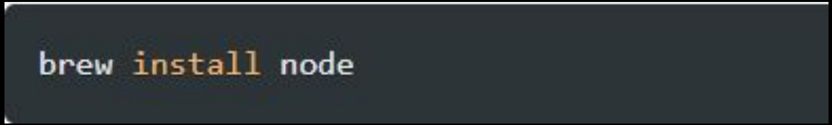
Installing Node.js

- Official packages for all the major platforms are available at https://nodejs.org/en/download/
- One very convenient way to install Node.js is through a package manager. In this case, every operating system has its own.
- Other package managers for Linux and Windows are listed in https://nodejs.org/en/download/package-manager/

Installing Node.js cont..

https://brew.sh/

- On macOs, Homebrew is the de-facto standard, and - once installed - allows to install Node.js very easily, by running this command in the CLI:



```
brew install node
```

Running Node from command line

- The usual way to run a Node.js program is to run the node globally available command (once you install Node.js) and pass the name of the file you want to execute.

- If your main Node.js application file is app.js, you can call it by typing:

```
node app.js
```

- While running the command, make sure you are in the same directory which contains the app.js file.

- Remember to exit a running Node program, use the Ctrl + C command

Differences between Node and Browser

While learning to code, you might also be confused at where does JavaScript end, and where Node.js begins, and vice versa.

We should have a good grasp of the main JavaScript concepts before diving into Node.js:

- Lexical Structure
- Expressions
- Types
- Variables
- Functions
- this
- Arrow Functions
- Loops
- Scopes
- Arrays
- Template Literals
- Semicolons
- Strict Mode
- ECMAScript 6, 2016, 2017

Main JavaScript concepts

The following concepts are also key to understand asynchronous programming, which is one fundamental part of Node.js:

- Asynchronous programming and callbacks
- Timers
- Promises
- Async and Await
- Closures
- The Event Loop

Video - Node Pros & Cons

Differences between Node and Browser

Both the browser and Node.js use JavaScript as their programming language.

Building apps that run in the browser is a completely different thing than building a Node.js application.

Despite the fact that it's always JavaScript, there are some key differences that make the experience radically different.

From the perspective of a frontend developer who extensively uses JavaScript, Node.js apps bring with them a huge advantage: the comfort of programming everything - the frontend and the backend - in a single language.

Differences between Node and Browser cont..

What changes is the ecosystem.

In the browser, most of the time what you are doing is interacting with the DOM, or other Web Platform APIs like Cookies. Those do not exist in Node.js, of course. You don't have the document, window and all the other objects that are provided by the browser.

And in the browser, we don't have all the nice APIs that Node.js provides through its modules, like the filesystem access functionality.

Differences between Node and Browser cont..

Another big difference is that in Node.js you control the environment. Unless you are building an open source application that anyone can deploy anywhere, you know which version of Node.js you will run the application on.

Compared to the browser environment, where you don't get the luxury to choose what browser your visitors will use, this is very convenient.

This means that you can write all the modern ES6-7-8-9 JavaScript that your Node.js version supports.

Since JavaScript moves so fast, but browsers can be a bit slow and users a bit slow to upgrade, sometimes on the web, you are stuck with using older JavaScript / ECMAScript releases.

Differences between Node and Browser cont..

You can use Babel to transform your code to be ES5-compatible before shipping it to the browser, but in Node.js, you won't need that.

Another difference is that Node.js uses the CommonJS module system, while in the browser we are starting to see the ES Modules standard being implemented.

In practice, this means that for the time being you use require() in Node.js and import in the browser.

Command Line

Run Node.js scripts from the command line

The usual way to run a Node.js program is to run the node globally available command (once you install Node.js) and pass the name of the file you want to execute.

If your main Node.js application file is app.js, you can call it by typing:

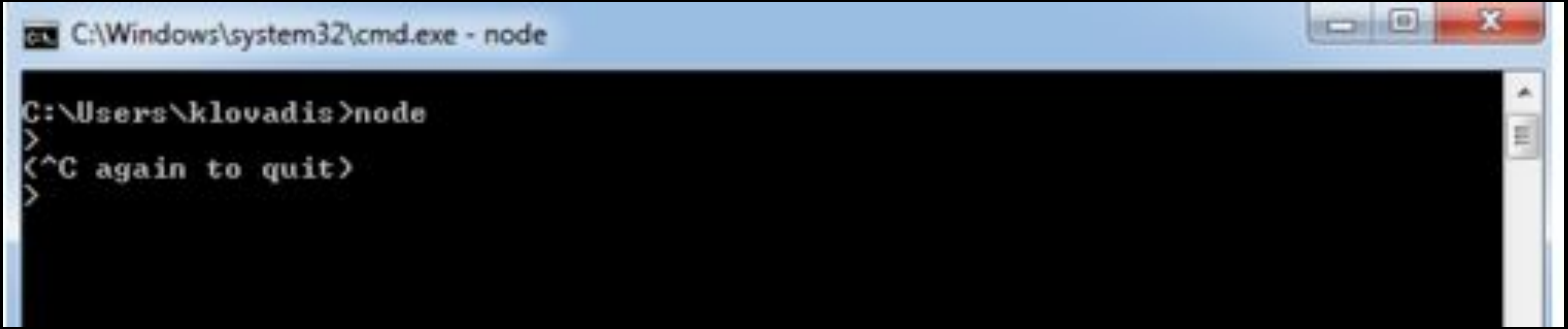
```
node app.js
```

While running the command, make sure you are in the same directory which contains the app.js file.

How to exit a running Node.js script from command line

There are various ways to terminate a Node.js application.

When running a program in the console you can close it with ctrl-C



C:\Windows\system32\cmd.exe - node

```
C:\Users\klovadis>node
>
(^C again to quit)
>
```

Video - What is Node.js