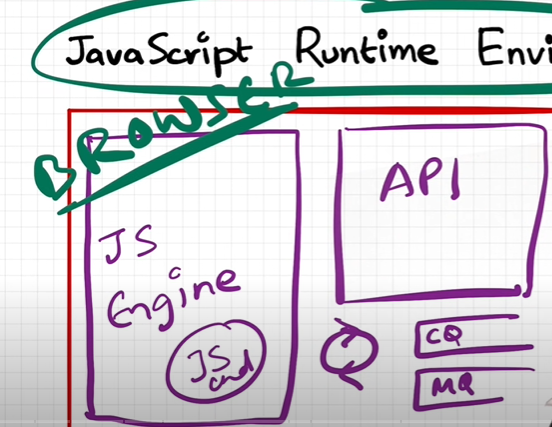
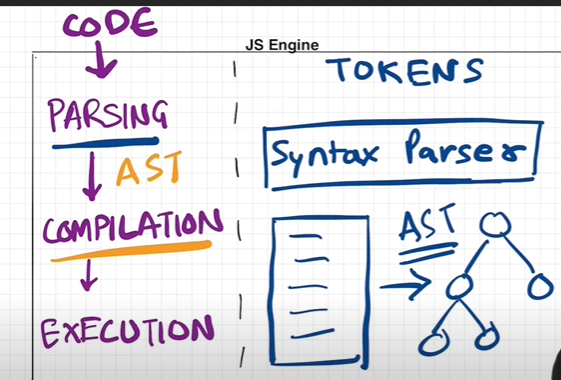
**JS Engine Exposed**

* To run any javascript code we need a Javascript Runtime Environment.
* Every browser has a Javascript Runtime Environment.
* NodeJs also has a Js Runtime Environment to execute Js code outside browser.
* JS runtime environment consist of –
  + JS Engine(Call Stack)
  + Web API
  + Event Loop
  + Callback queue
  + Micro Task queue



* JS Engine is the heart of JS Runtime Environment.
* Every browser has its own JS Engine.
  + Microsoft Edge – Chakra
  + Firefox – Spider Monkey
  + Chrome – v8 Engine
* You can create your own JS Engine using Ecmascript standards.
* First JS Engine was developed – Spider Monkey.

**JS Engine Architecture**

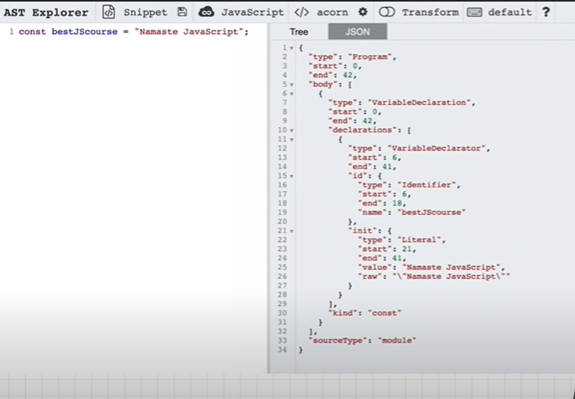


* Every Js Engine consist of following phase – Parsing, Compilation, Execution.
* Each line of Js code it first converts them into separate tokens.

Eg – var name = “xyz”

Var is separate token, name is separate token, assigned value is separate token.

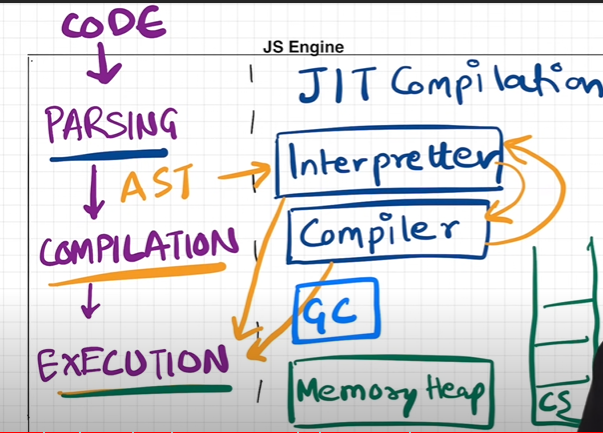
* These tokens are than parse for compilation using Syntax Parser – AST
* AST parser looks like nothing but below json structure.



Before explaining compilation and Execution lets first understand what is Interpretter and Compiler.

**Interpretter** – In Interpretter each line is executed sequentially and checked against the syntax. It doesn’t know whether next line will be executed or not.

**Compilation** – Compilation is the process where entire program is first compiled against the syntax and it provides more optimized version of its code.



* In Js Engine, Compilation and Execution goes hand in hand using **JIT(Just In Time) Compilation**.
* The code which is parsed using AST, it is then interpreted and compiled parallelly before execution. We can say Interpretter reads the Parsed code and it is then checked and compiled against syntax and provides its optimized version, which is then converted into machine readable code during execution.
* Variables and Methods are stored in Memory Heap.
* It also consist of Garbage collection which collects unused code and free memory space.
* Garbage collection does this using Mark and Sweep Algorithm.

