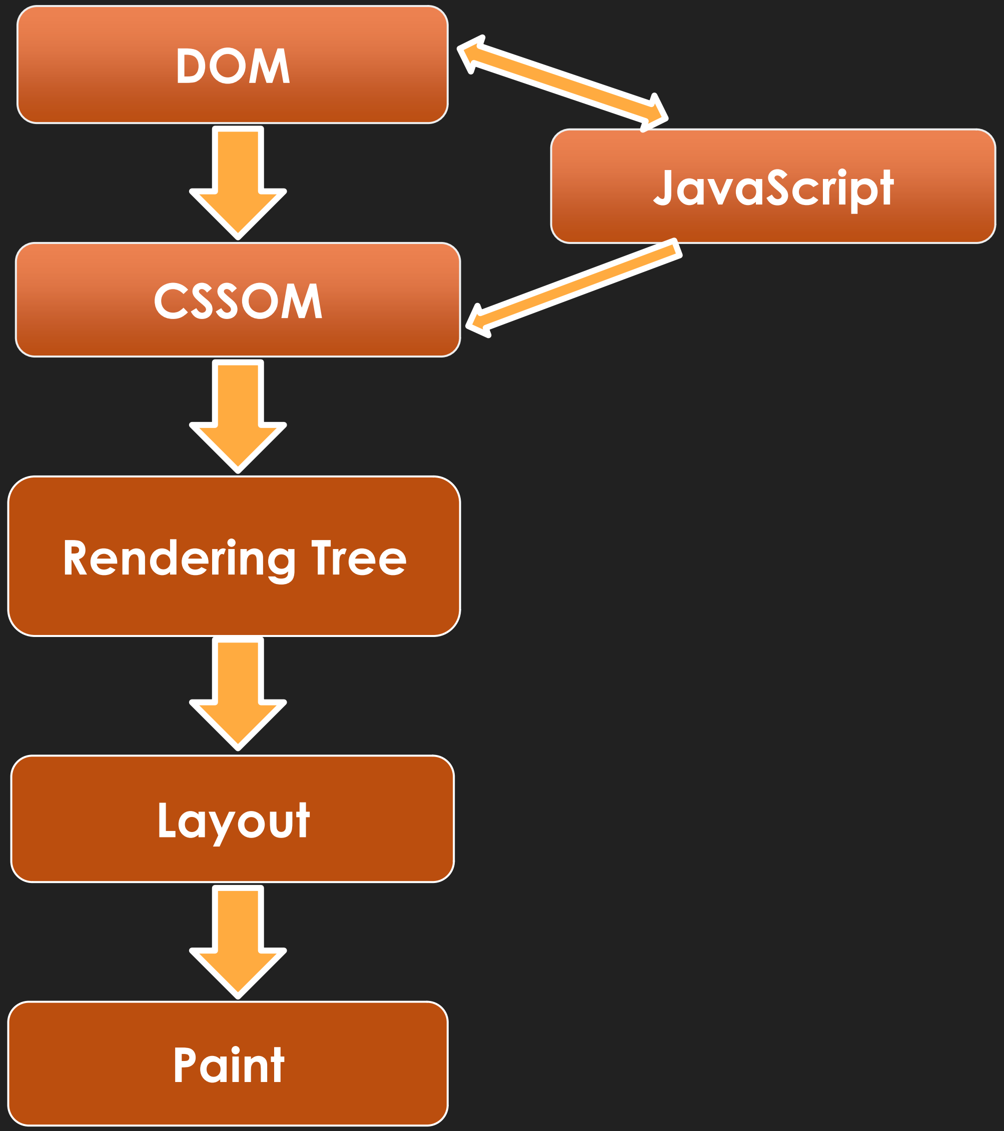
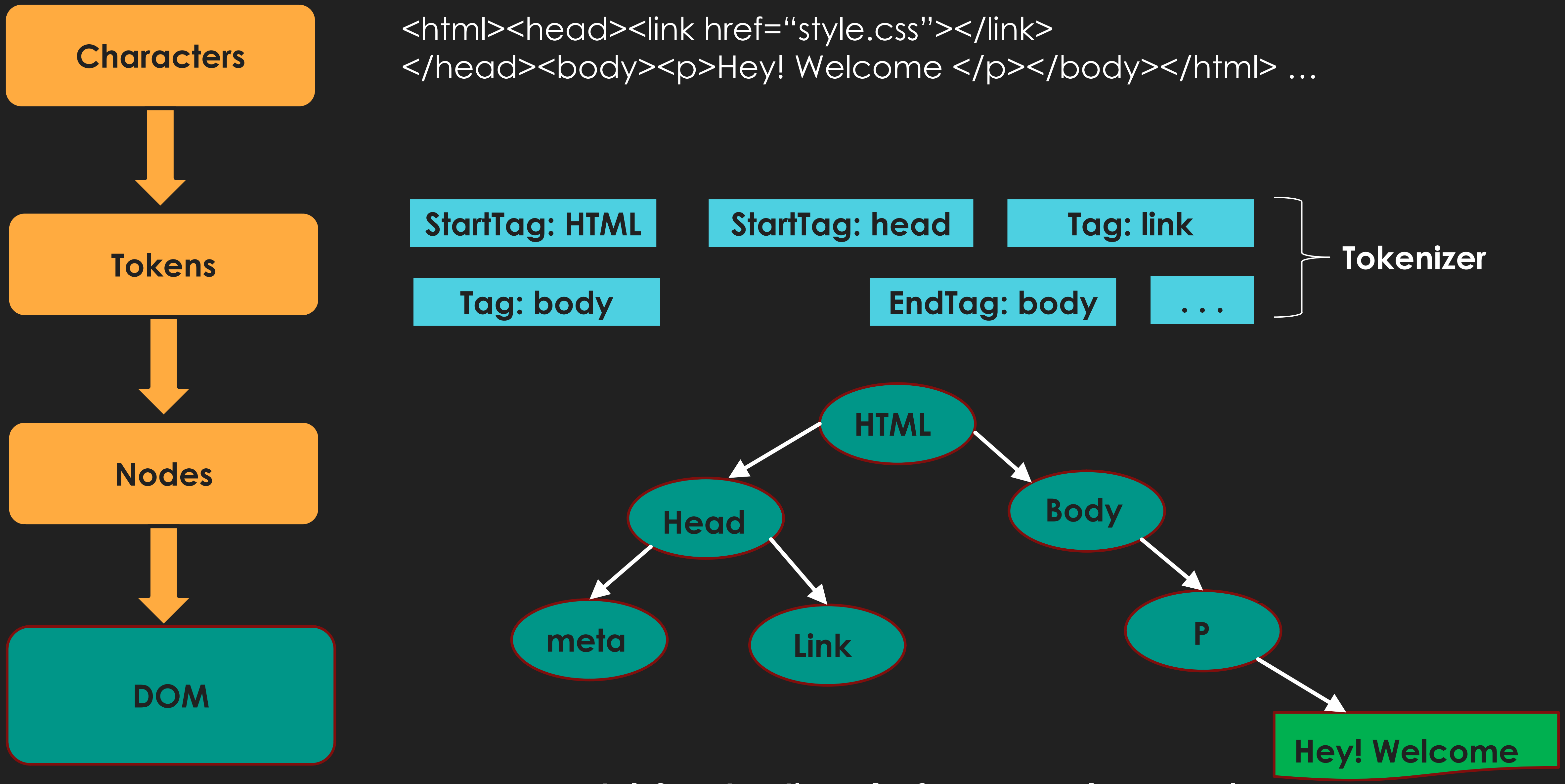
NodeJs & Web

Agenda

- How HTML, CSS, Javascript renders on browser
- How PHP Works
- Node.js
  - Introduction
  - Single Threaded
  - Non-blocking I/O
  - Event Loop
  - JavaScript is Asynchronous
  - NPM (Node Package Manager)
  - How Node.Js works
- Cluster
- Real time chat application with Demo

Critical Rendering Path





Characters

Tokens

Nodes

DOM

```
<html><head><link href="style.css"></link>
</head><body><p>Hey! Welcome </p></body></html> …
```

StartTag: HTML

StartTag: head

Tag: link

Tag: body

EndTag: body

. . .

Tokenizer

HTML

Head

Body

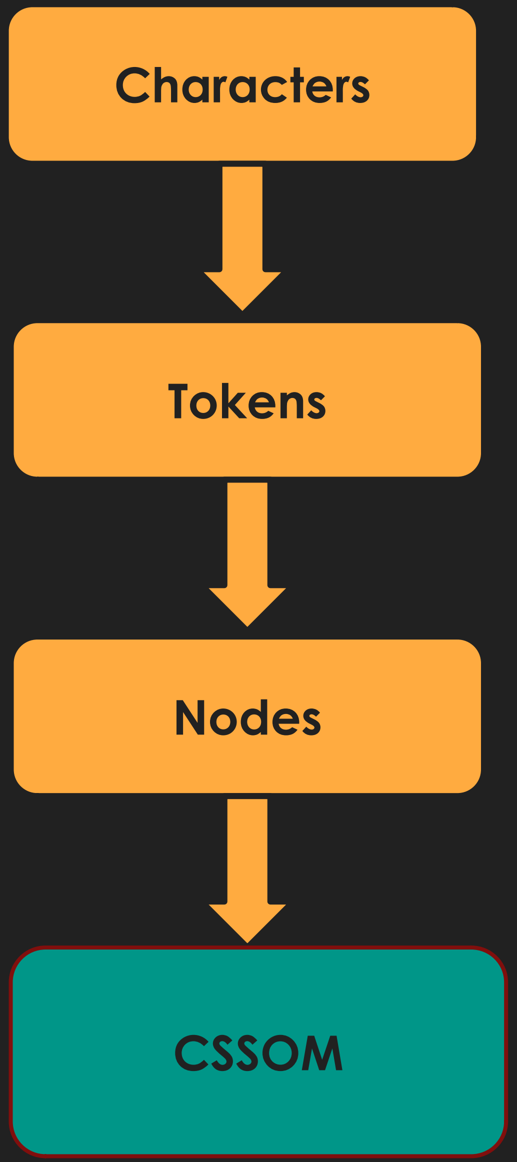
meta

Link

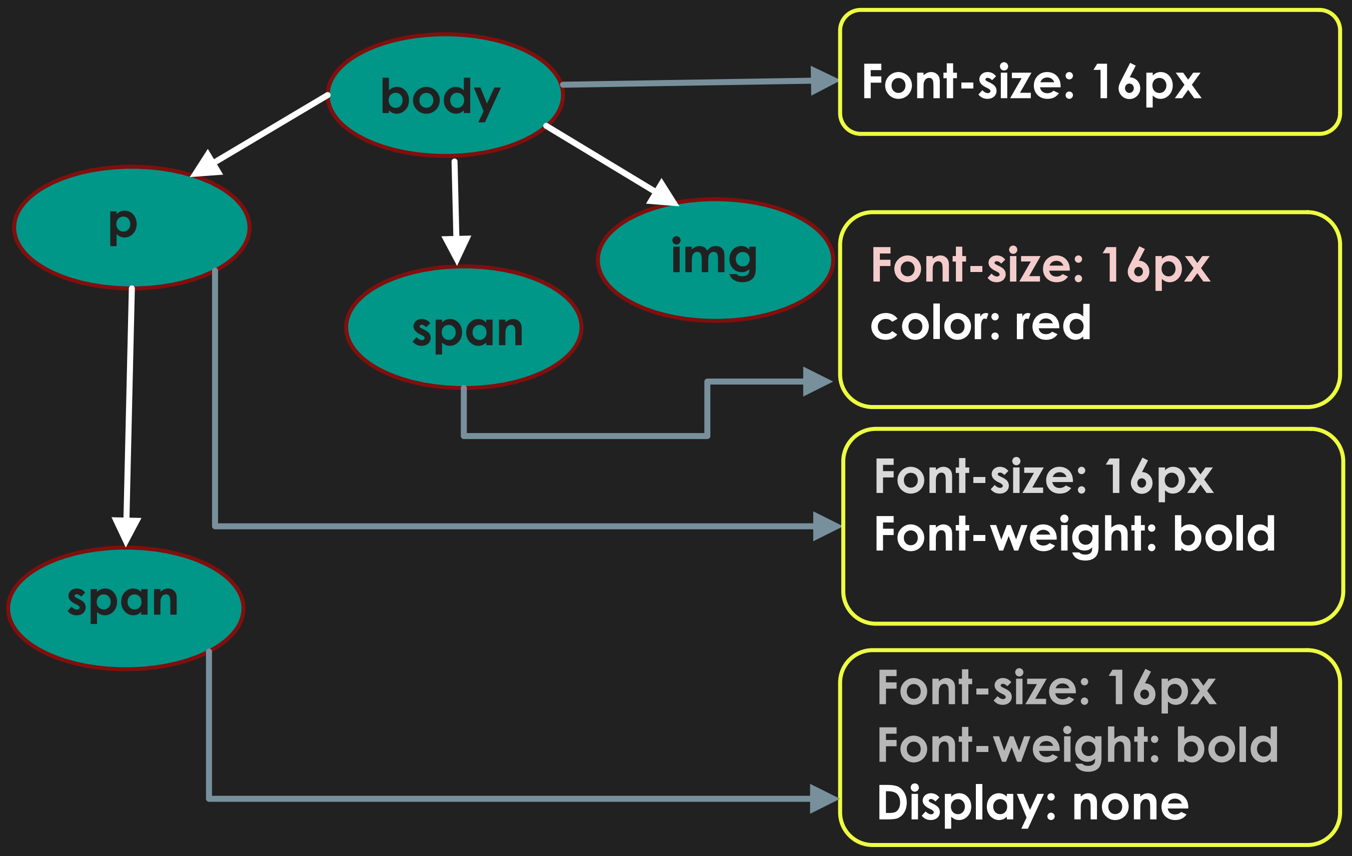
P

Hey! Welcome

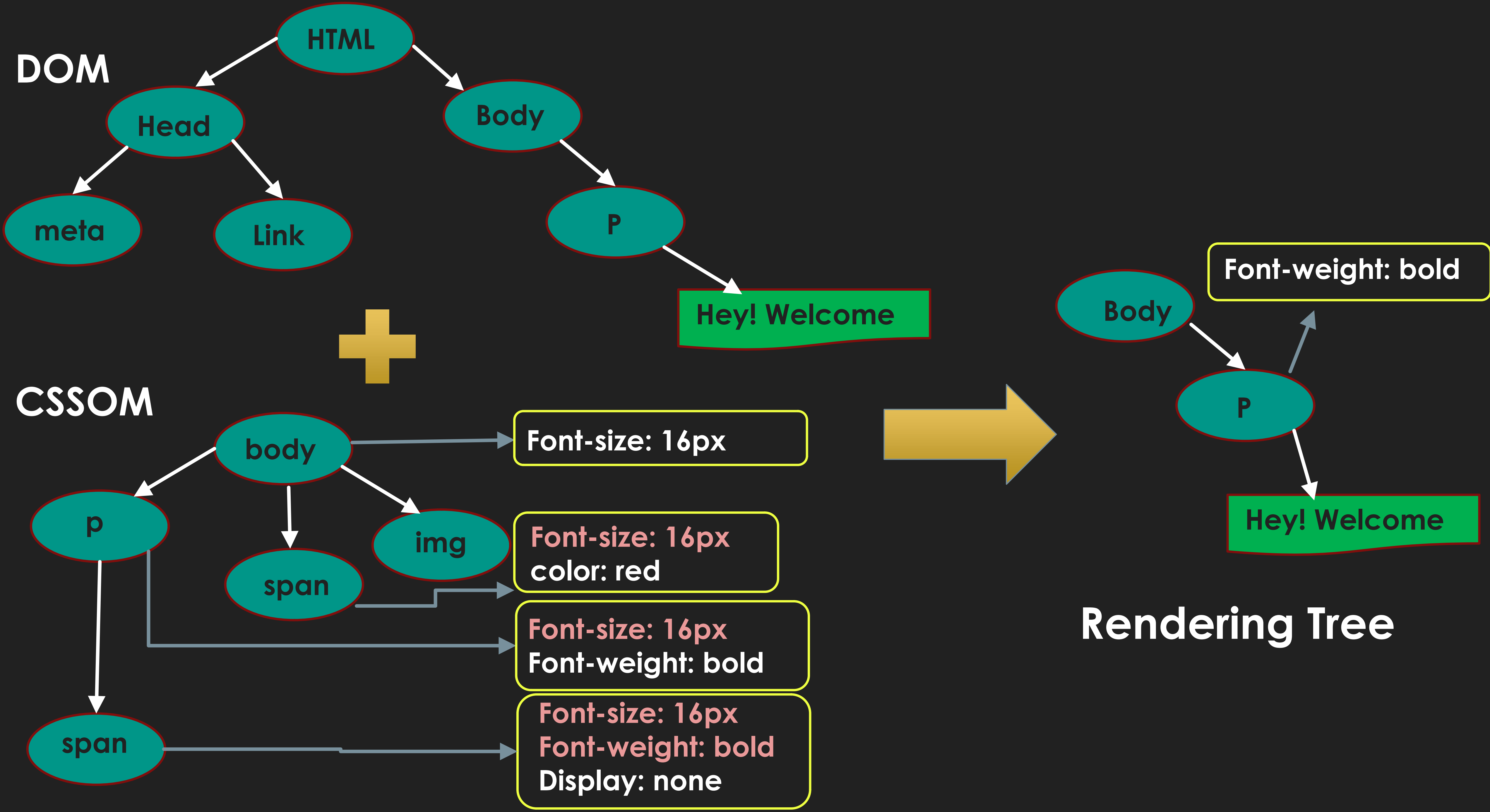
Incremental Construction of DOM. Example - google.com



```
body{ font-size: 16px} p {font-weight: bold} span {color: red}
p span{display: none} img{float: right} …
```



Incremental Construction of CSSOM – NOT POSSIBLE (Thus CSS is render blocking)


DOM
HTML
Head
Body
meta
Link
P
Hey! Welcome
CSSOM
body
Font-size: 16px
p
span
img
Font-size: 16px
color: red
Font-size: 16px
Font-weight: bold
span
Font-size: 16px
Font-weight: bold
Display: none
Font-weight: bold
Body
P
Hey! Welcome
Rendering Tree

How PHP Works

```
<?php
class Controller_SuperAdmin_AccountManager extends Controller
{
        function __construct(Request $request, Response $response)
        {
                $this->common_function = new CommonFunction;
        }
        function action_index()
        {
                echo  ‘Hello ISS !’;
        }
?>
```

Result :-   Hello ISS !

Node.js

- Ryan Dahl - Creator of Node.js
- Built on top of Chrome’s v8 JavaScript Engine for building fast, scalable network application
- Uses an Event Driven, non - blocking I/O model that makes it lightweight and efficient
- Single threaded

Non-blocking I/O

- All request temporarily saved on heap
- To avoid blocking, Node makes use of the event driven nature of JS by attaching callbacks to I/O requests
- Can support nearly 1 million concurrent connection
- Script waiting on I/O waste no space because they get popped off the stack when their non-I/O related code finishes executing

Single Threaded

- Most other web platform are multi-threaded, but JS has single thread

  One thread === One Call Stack === One thing at a time
- When threads are bad :-
  - Hard to program
  - shared the state and locks
  - Deadlocks
  - Giant locks decrease concurrency
  - Context switching cases
- When threads are good :-
  - Support Multi core CPUs
  - CPU - heavy work
  - Little or no shared state
  - Thread count == CPU core count

Event Loop

- Instead of threads Node uses event loop with a stack

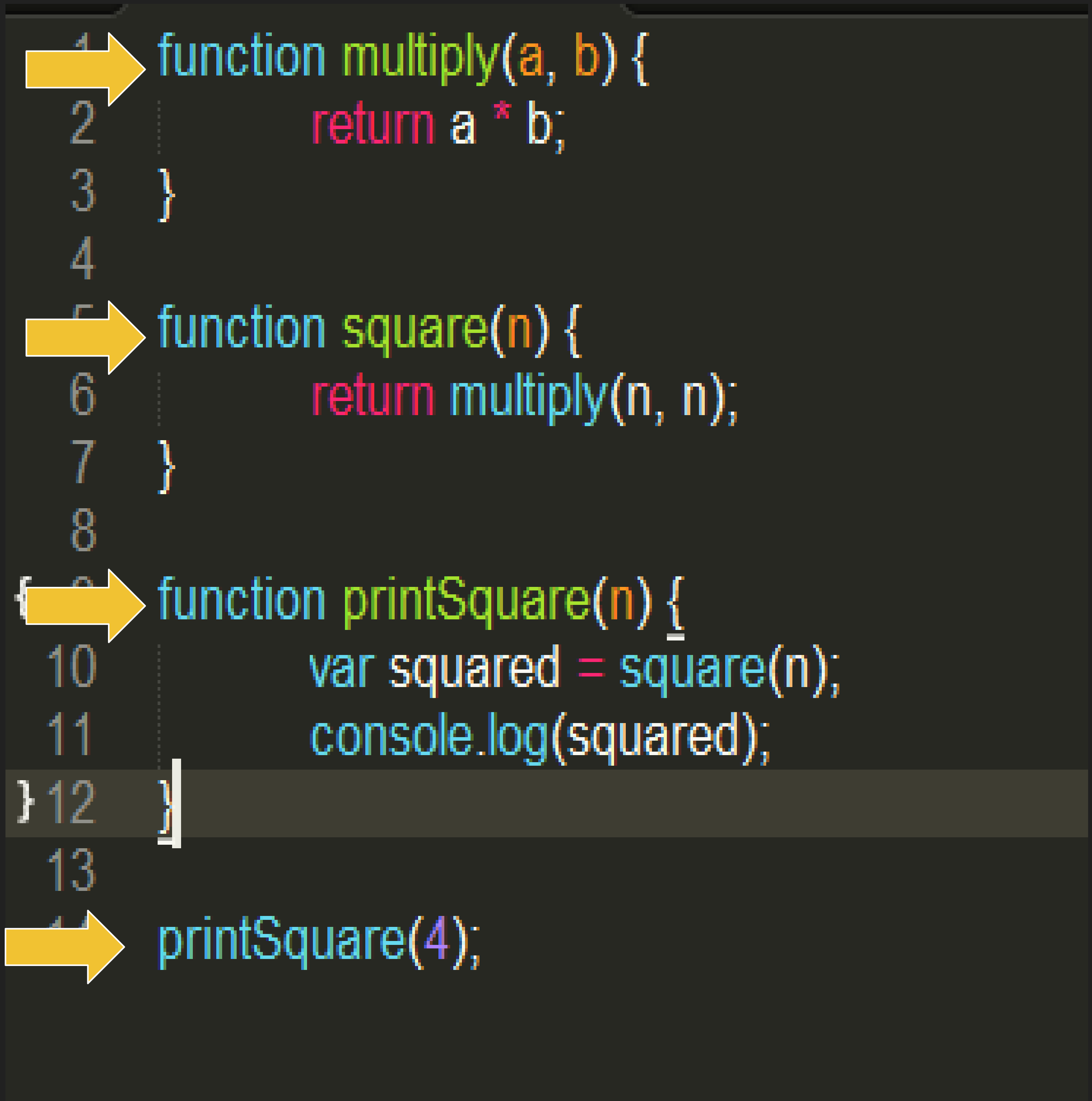
- You already use callback and event loop

```
Ext.getStore(‘location_store’).on(‘load’, function(store, eOpts) {
                        // Some Code
});
```

or

```
$.ajax(. . . , function(...) {
        console.log(‘Ajax Request!’);
});
```

- Examples

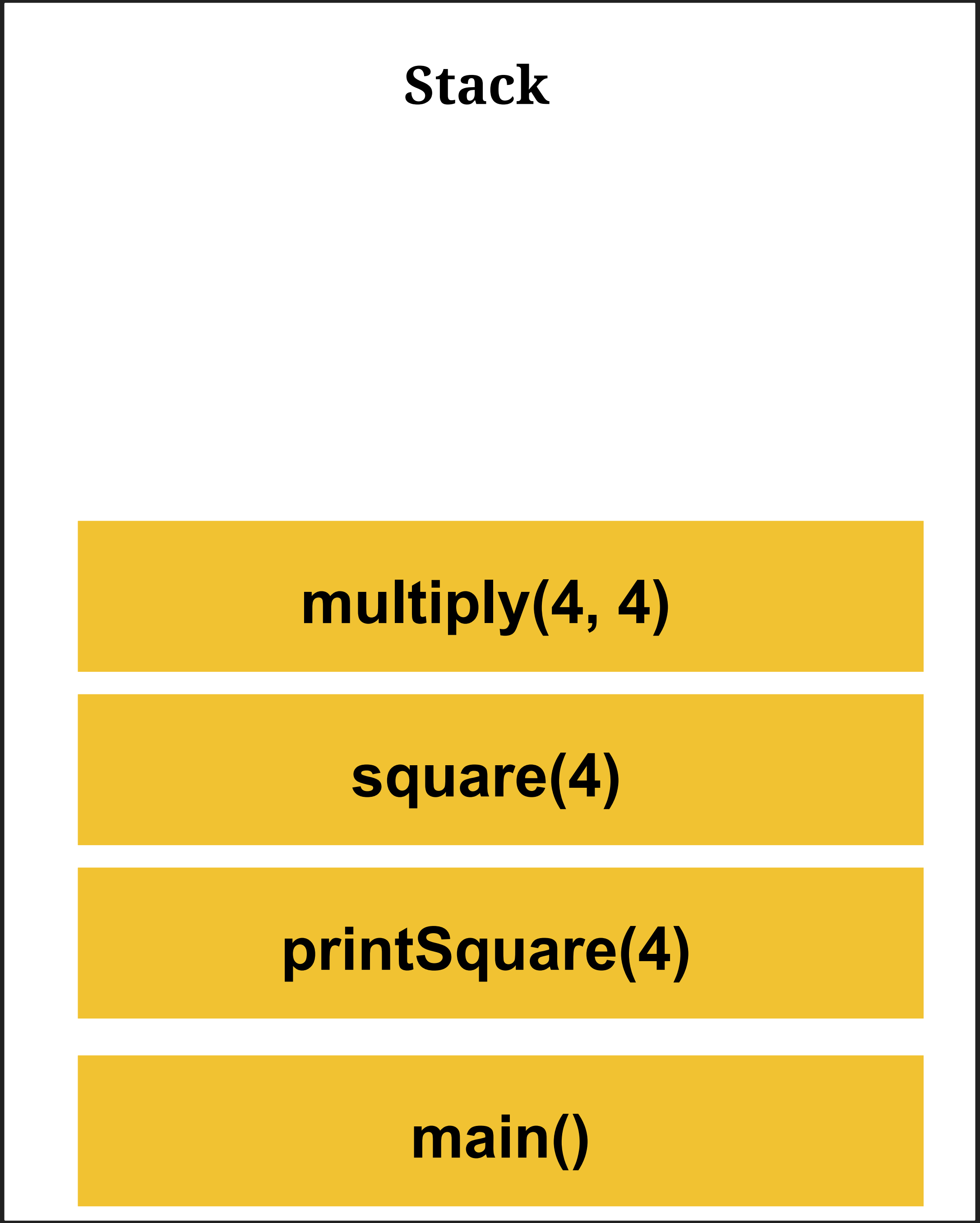


```
function multiply(a, b) {
        return a * b;
}

function square(n) {
        return multiply(n, n);
}

function printSquare(n) {
        var squared = square(n);
        console.log(squared);
}

printSquare(4);
```



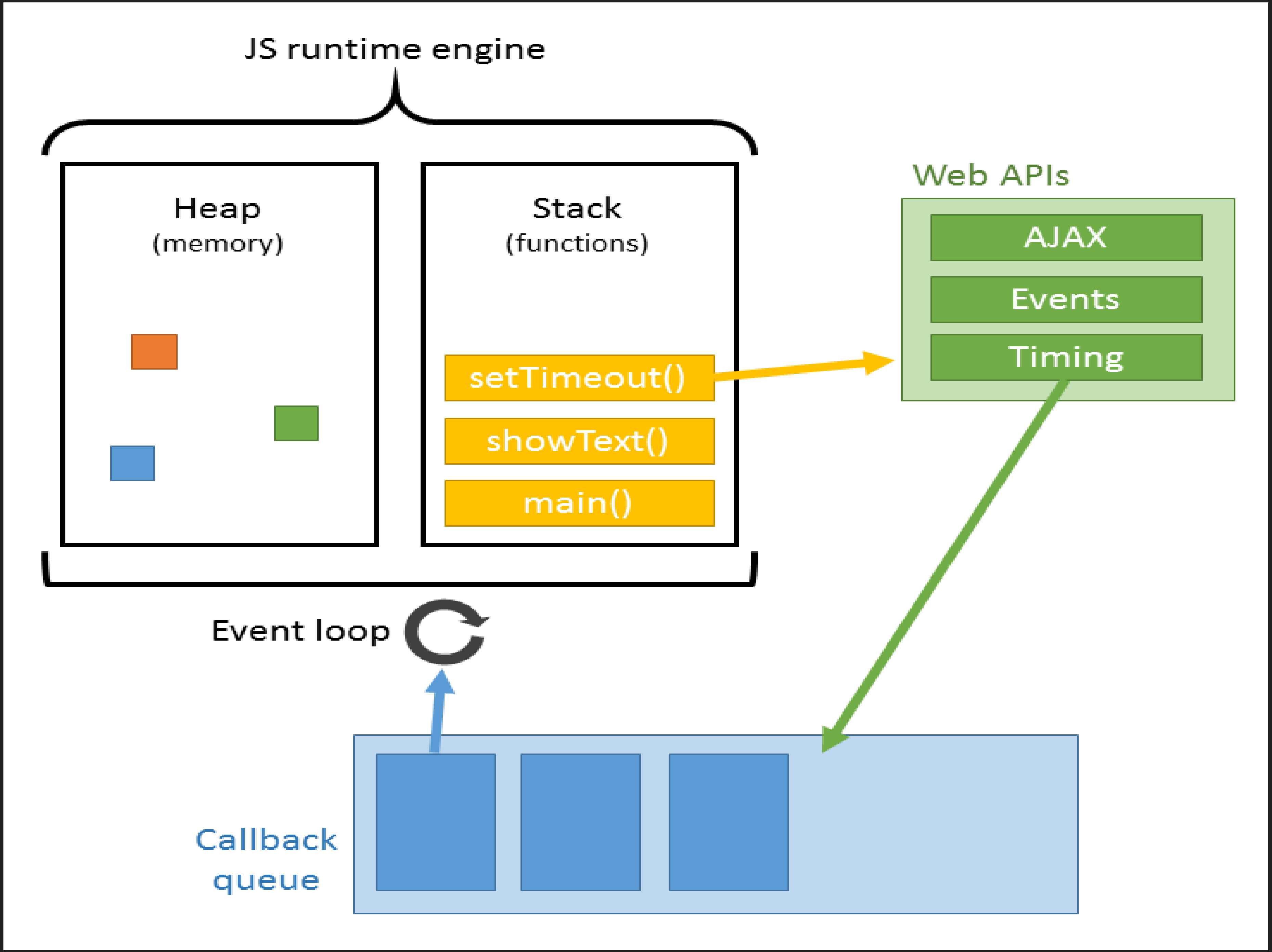
Stack

multiply(4, 4)

square(4)

printSquare(4)

main()

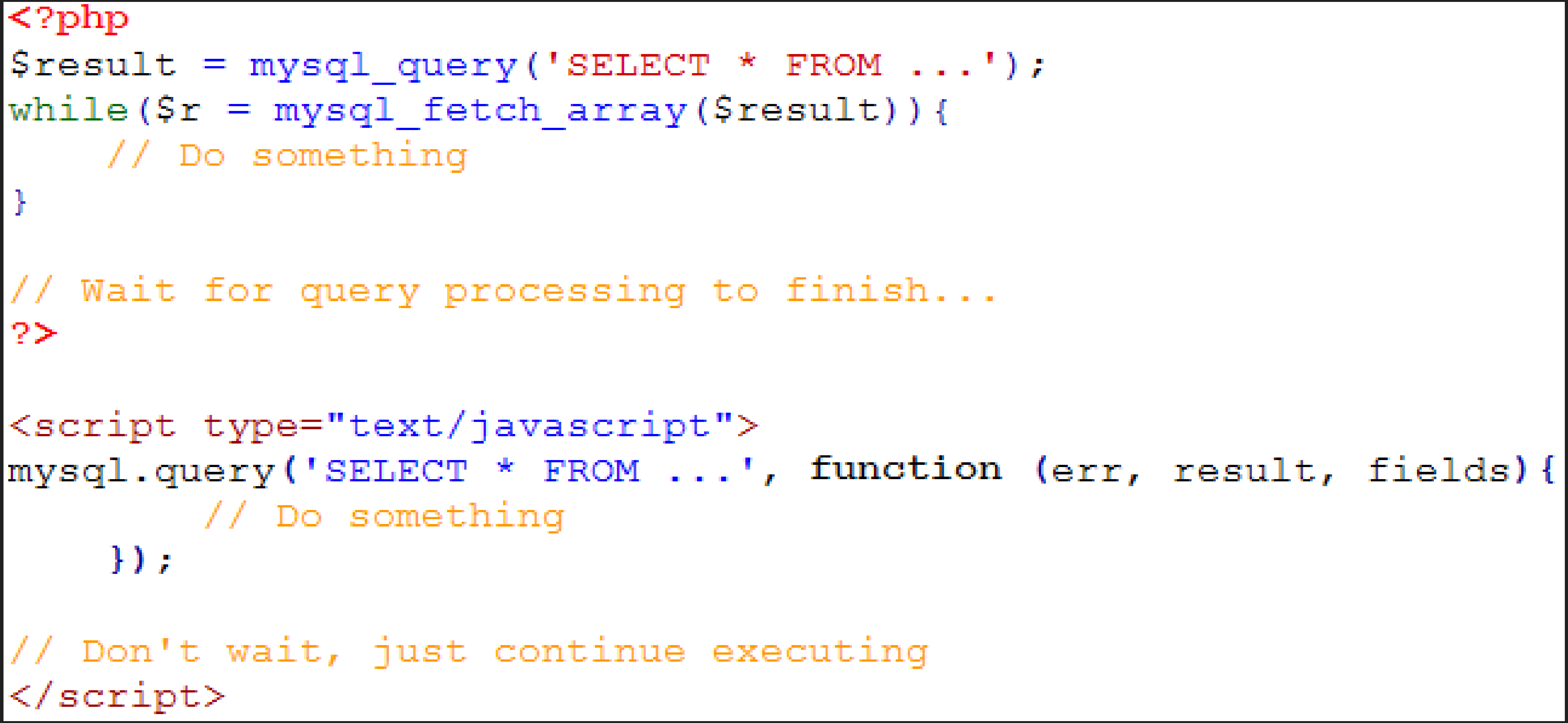


Here is real Concurrency

Event Loop Example

- Demo I
- Demo II

I/O Example



```
<?php
$result = mysql_query('SELECT * FROM ...');
while($r = mysql_fetch_array($result)){
    // Do something
}

// Wait for query processing to finish...
?>

<script type="text/javascript">
mysql.query('SELECT * FROM ...', function (err, result, fields){
        // Do something
    });

// Don't wait, just continue executing
</script>
```

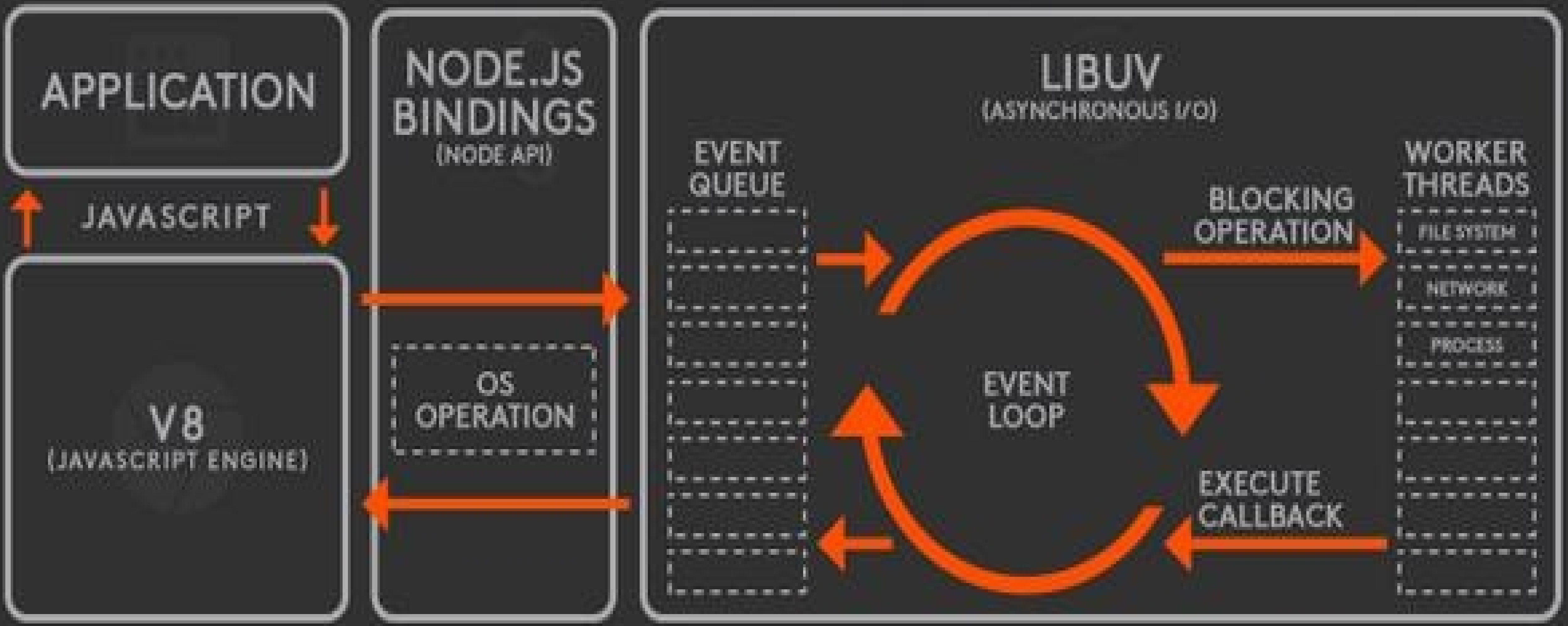
NPM (Node Package Manager)

- Reusable components
- NPM installs application dependencies locally, not globally
- NPM handles multiple versions of the same module at the same time
- It's really easy to publish your own module to the npm registry

Where is node.js ?

- Single Threaded
- Non-blocking I/O
- Event Loop
- JavaScript is Asynchronous
- V8
- NPM

THE NODE.JS SYSTEM



APPLICATION

JAVASCRIPT

V8
(JAVASCRIPT ENGINE)

NODE.JS
BINDINGS
(NODE API)

OS
OPERATION

LIBUV
(ASYNCHRONOUS I/O)

EVENT
QUEUE

EVENT
LOOP

BLOCKING
OPERATION

WORKER
THREADS

FILE SYSTEM

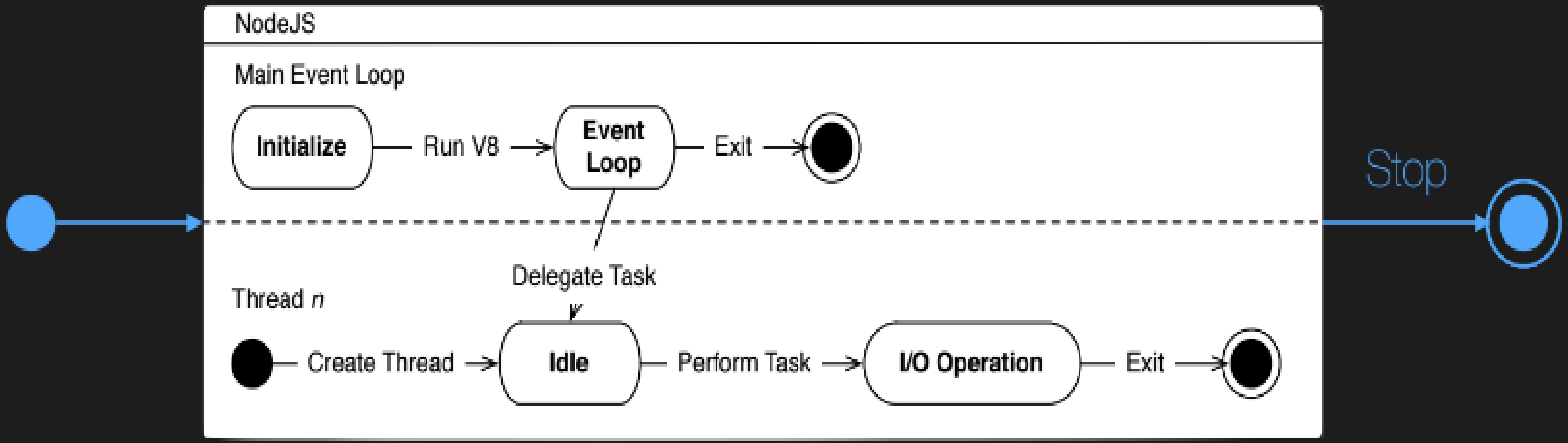
NETWORK

PROCESS

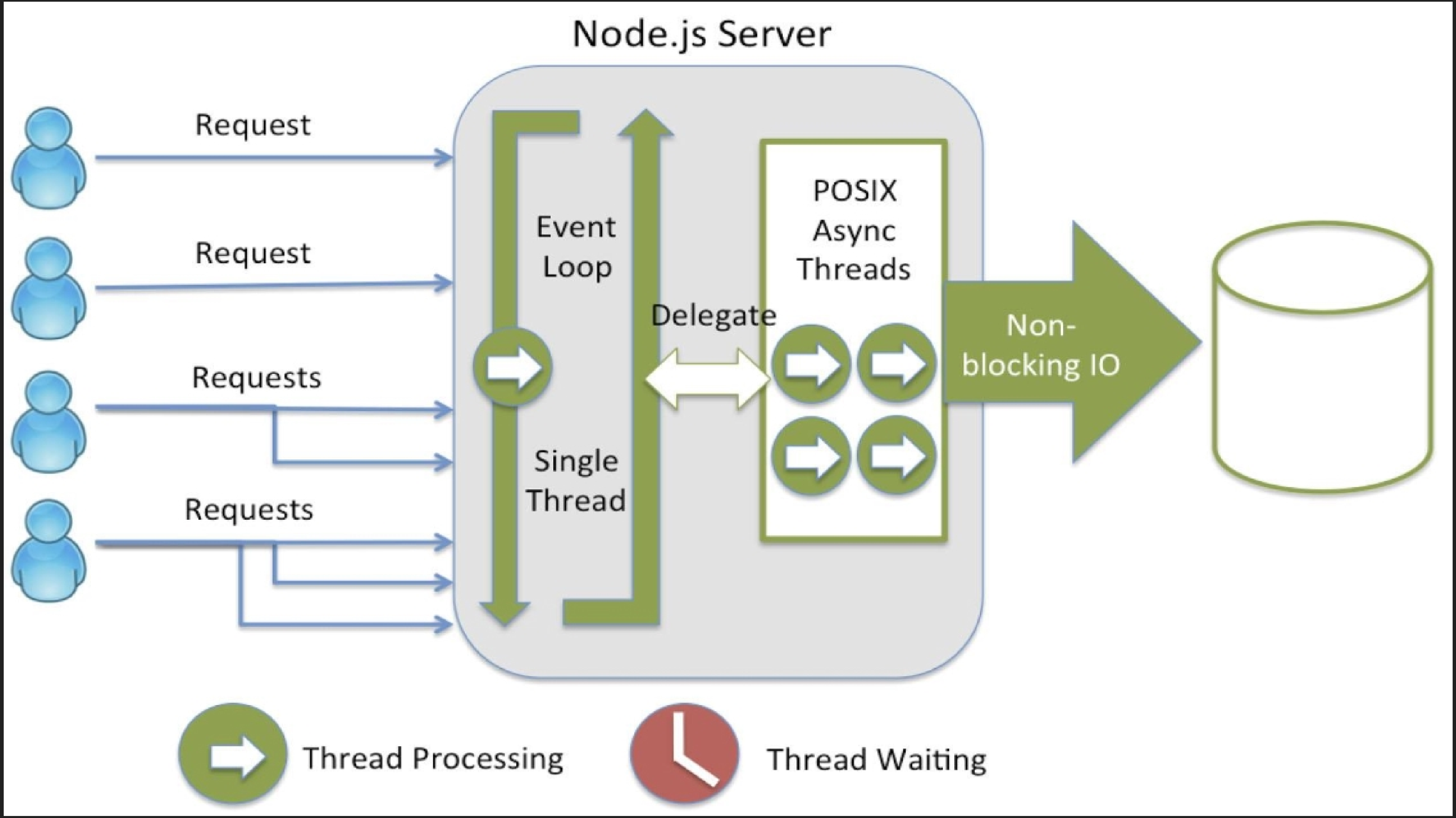
EXECUTE
CALLBACK

Node.js Working

A generic model of Node.js



How Node.js Works



Which is better for my application ?

HTML

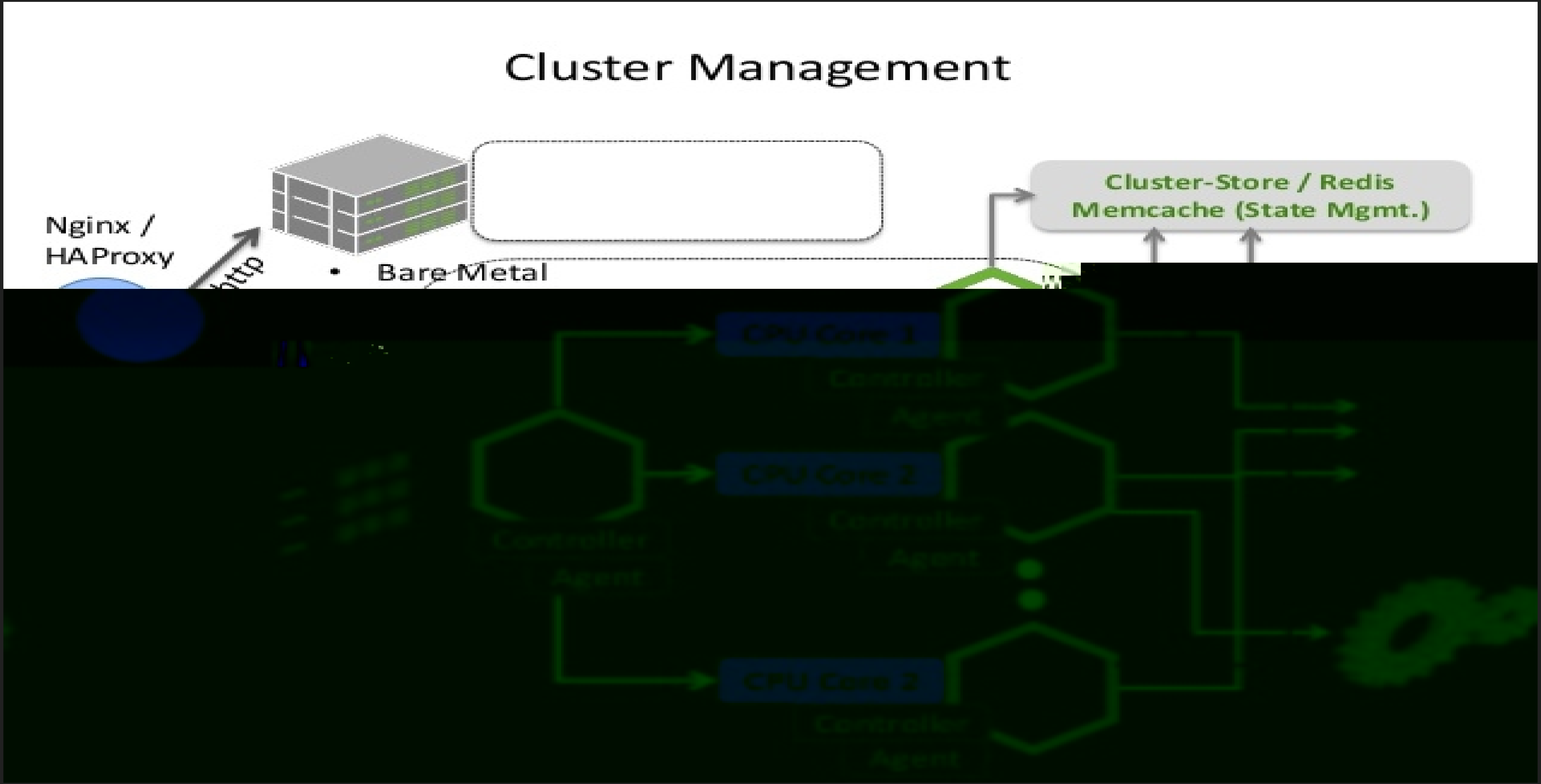
CSS

Javascript

php

node.js

Cluster


Cluster Management
Nginx / HAProxy
http
• Bare Metal
Cluster-Store / Redis Memcache (State Mgmt.)
CPU Core 1
Controller Agent
CPU Core 2
Controller Agent
Controller Agent
CPU Core 2
Controller Agent

Huge Success

Real time Chat Application

Enough theory!

Let’s create cool application using Node.JS

https://github.com/sanket876/chat-application.git

Thank You! ☺