Concurrency Model and Event Loop

Samer Buna

@samerbuna www.jscomplete.com

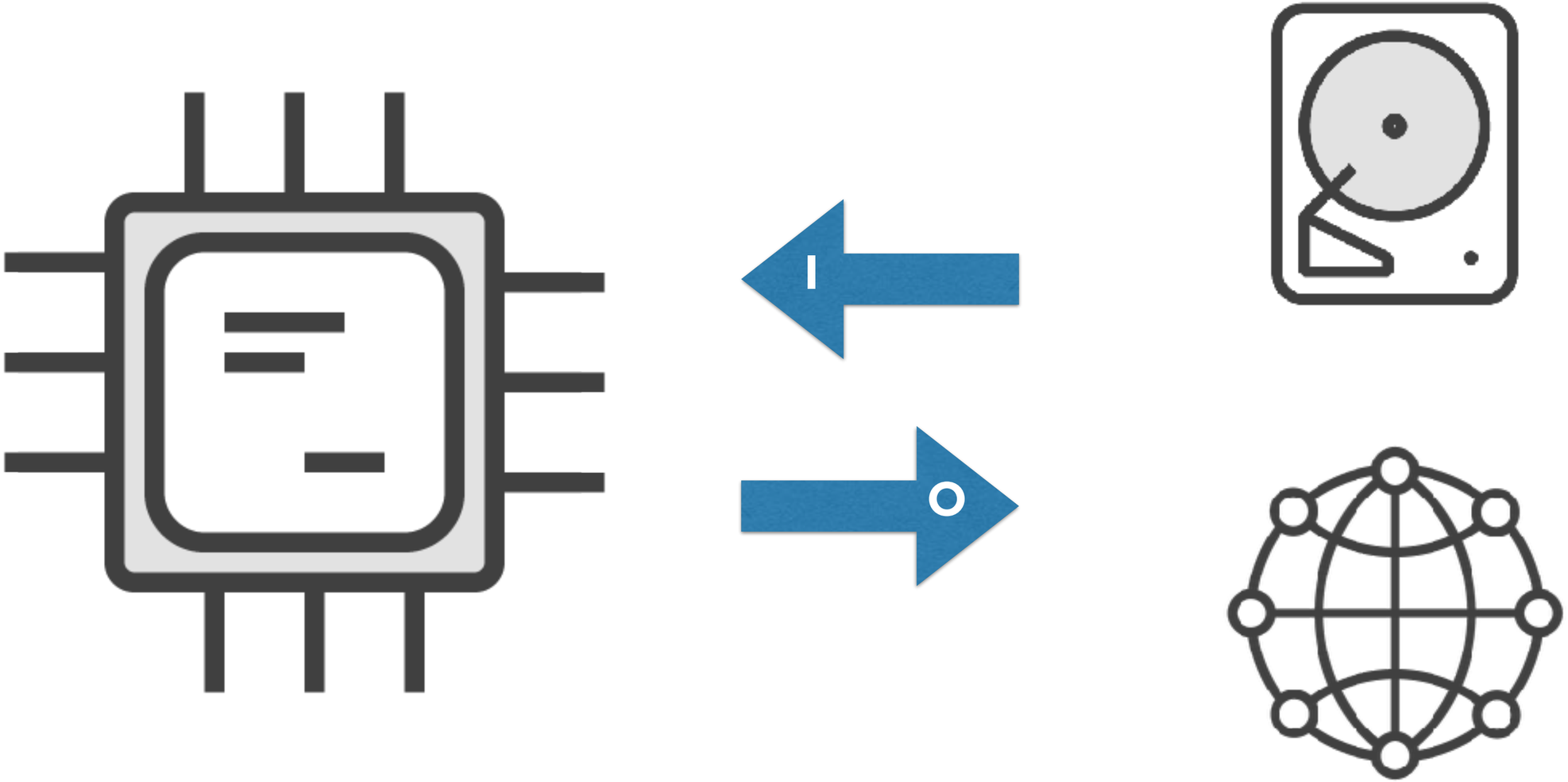
Event Model

Nodejs	Ruby	Python
libuv	Event Machine	Twisted

The Event Loop

Slow I/O operations

What Is I/O Anyway?


I
O

Handling Slow I/O

Synchronous	fork()
Threads	Event Loop

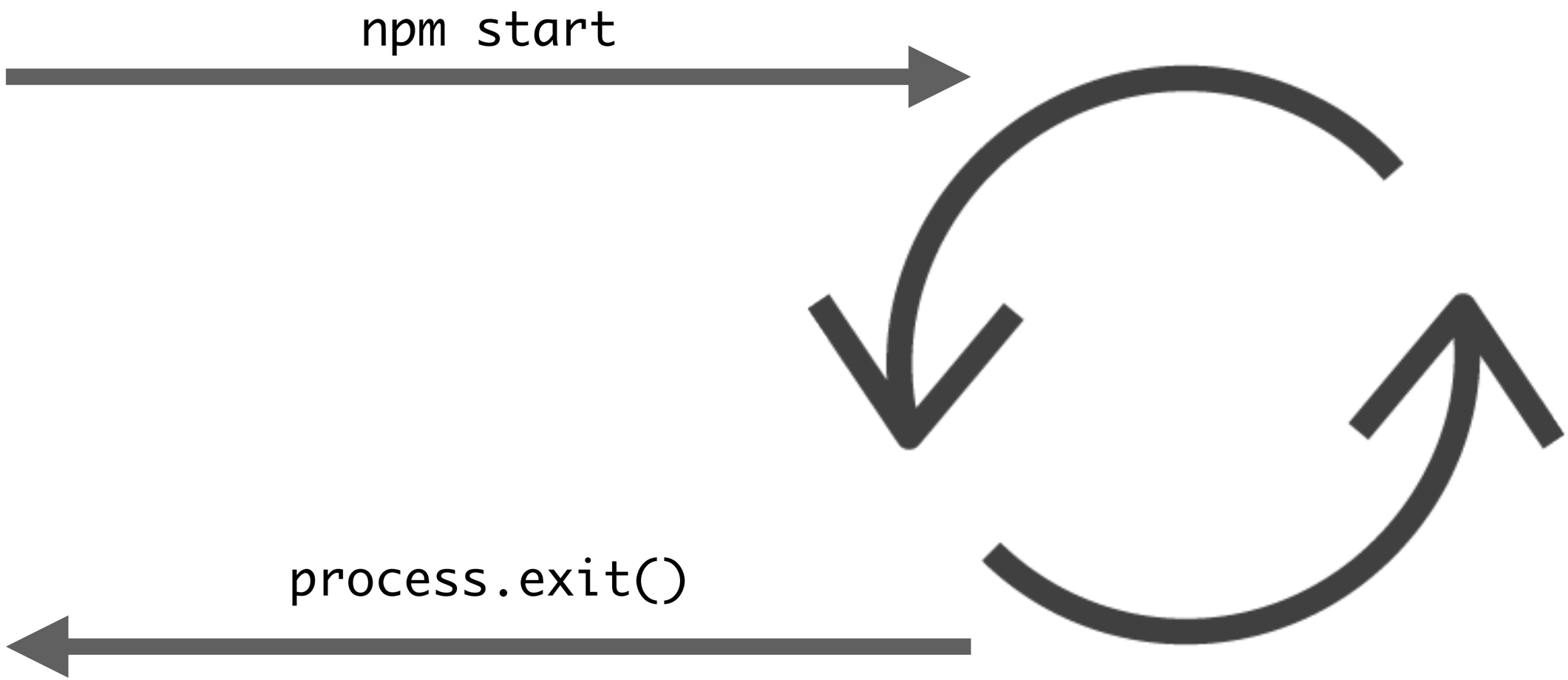
The Event Loop

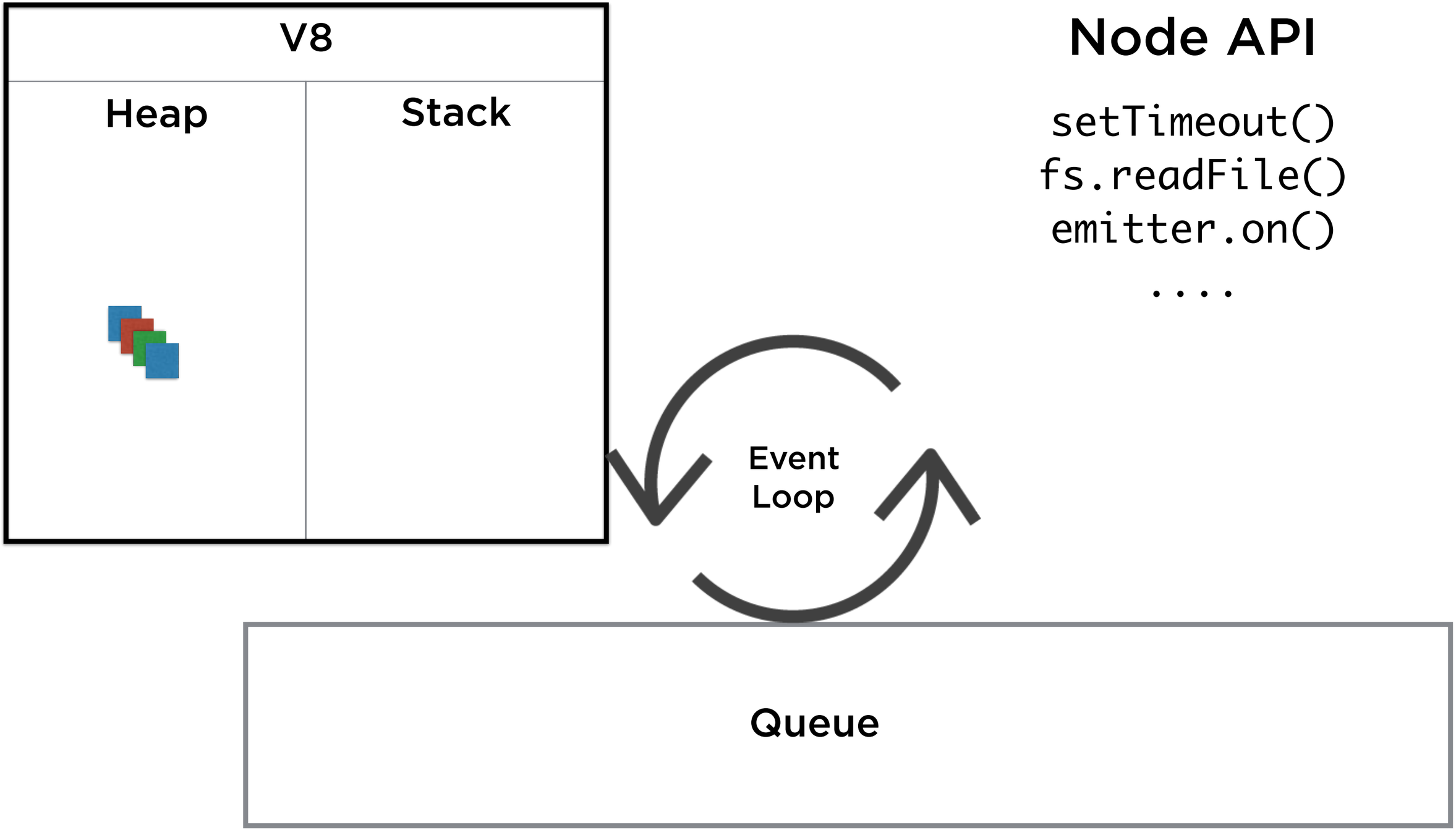
The Event Loop

The entity that handles external events and converts them into callback invocations

The Event Loop

A loop that picks events from the event queue and pushes their callbacks to the call stack


npm start
process.exit()

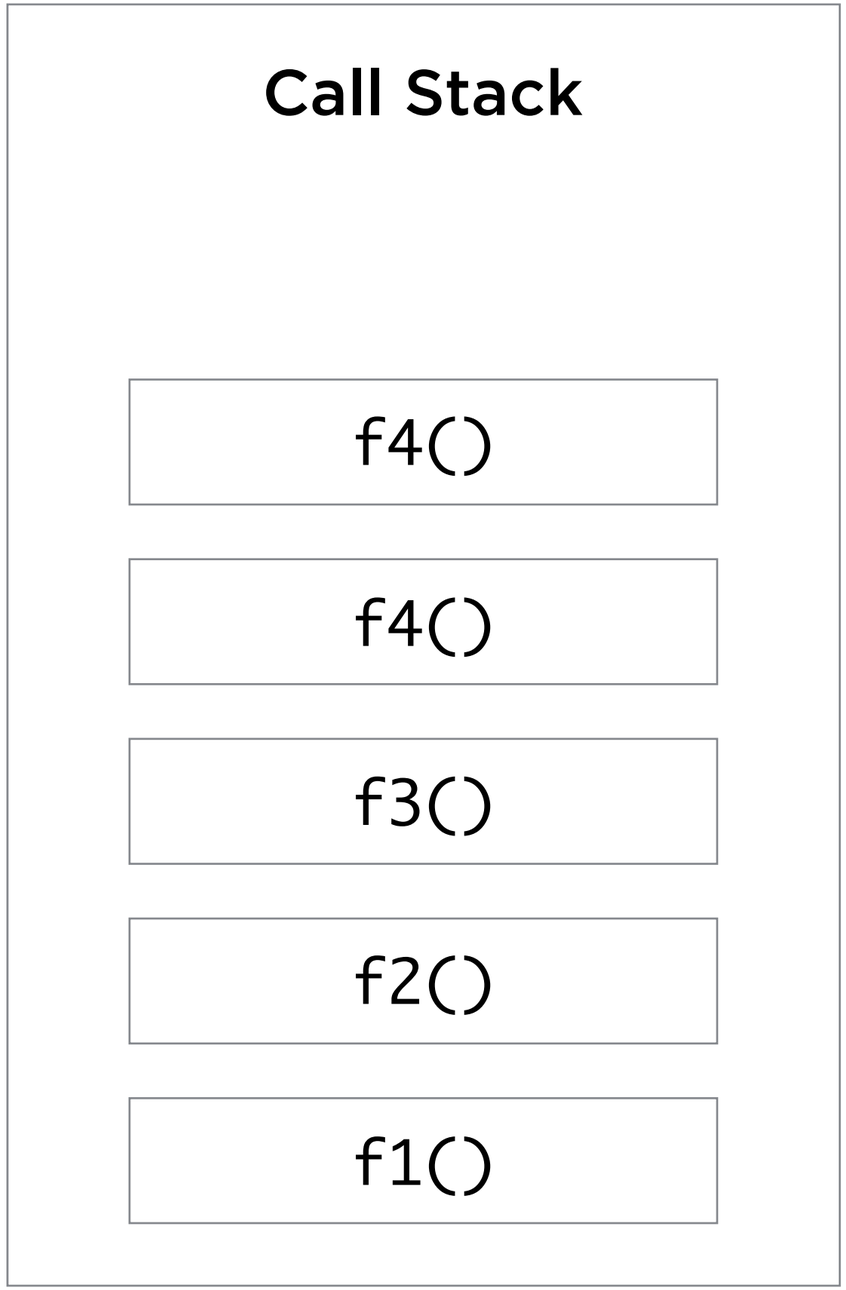


Node API

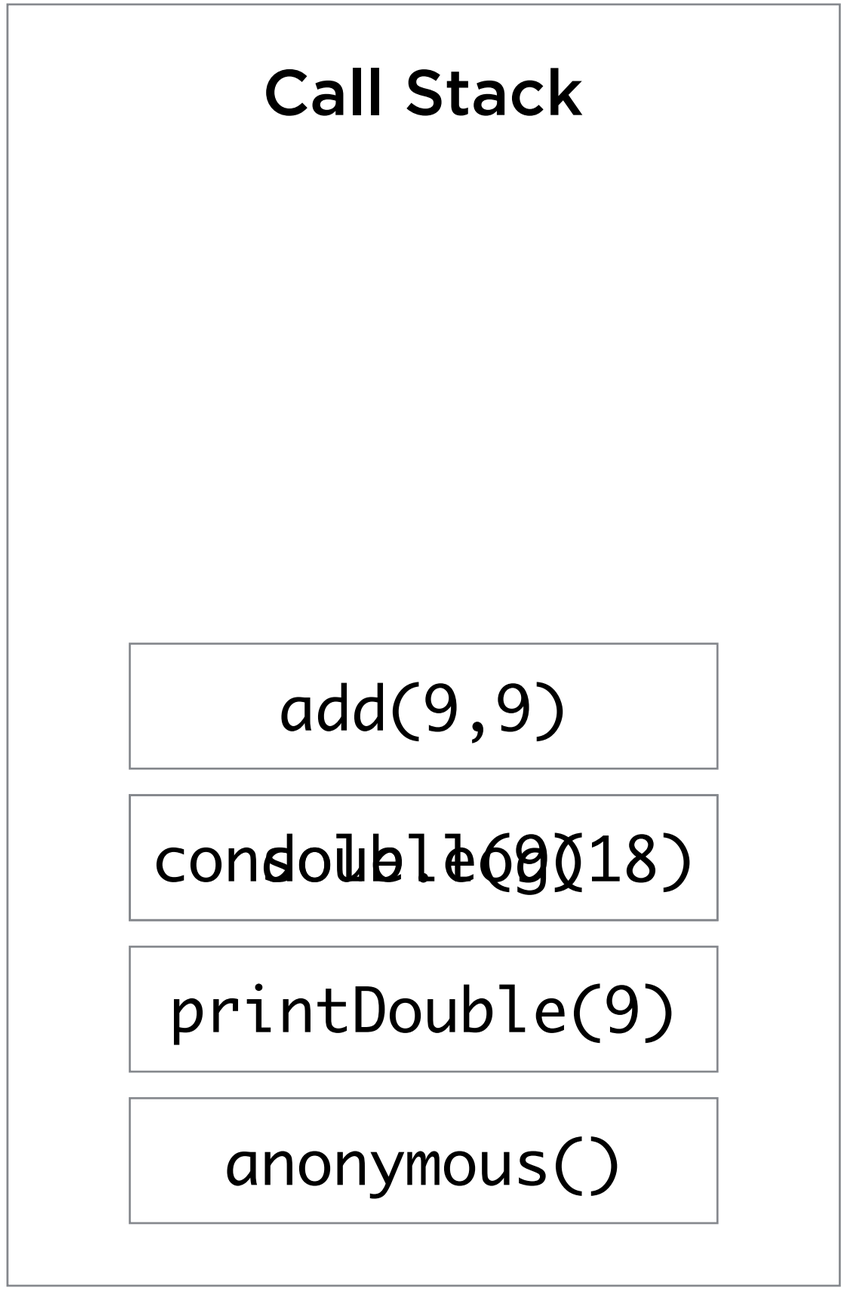
```
setTimeout()
fs.readFile()
emitter.on()
....
```

The Call Stack

```
const f1 = () => { f2(); };

const f2 = () => { f3(); };

const f3 = () => { f4(); };

const f4 = () => { f4(); };
```



```
const add = (a, b) => a + b;

const double = a =>
  add(a, a);

const printDouble = a => {
  const output = double(a);
  console.log(output);
};

printDouble(9);
```



```
const add = (a, b) => x + b;

const double = a =>
  add(a, a);

const printDouble = a => {
  const output = double(a);
  console.log(output);
};

printDouble(9);
```

Elements Console » 1

top Preserve log

```
> const add = (a, b) => x + b;

  const double = a =>
    add(a, a);

  const printDouble = a => {
  const output = double(a);
  console.log(output);
  };

  printDouble(9);
```

```
Uncaught ReferenceError: x is not defined                VM462:1
    at add (<anonymous>:1:20)
    at double (<anonymous>:4:3)
    at printDouble (<anonymous>:7:16)
    at <anonymous>:11:1
  add          @ VM462:1
  double       @ VM462:4
  printDouble  @ VM462:7
  (anonymous)  @ VM462:11
```

>

```
const add = (a, b) =>
  add(a, b);

const double = a =>
  add(a, a);

const printDouble = a => {
  const output = double(a);
  console.log(output);
};

printDouble(9);
```

Elements Console » 1

top Preserve log

```
> const add = (a, b) =>
  add(a, b);

  const double = a =>
    add(a, a);

  const printDouble = a => {
  const output = double(a);
  console.log(output);
  };

  printDouble(9);
```

```
Uncaught RangeError: Maximum call stack size exceeded        VM980:2
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
    at add (<anonymous>:2:1)
```

>

Handling Slow Operations

```
const slowAdd = (a, b) => {
  for(let i=0; i<999999999; i++) {}
  return a + b;
};

const a = slowAdd(3, 3);

const b = slowAdd(4, 4);

const c = slowAdd(5, 5);

console.log(a);

console.log(b);

console.log(c);
```

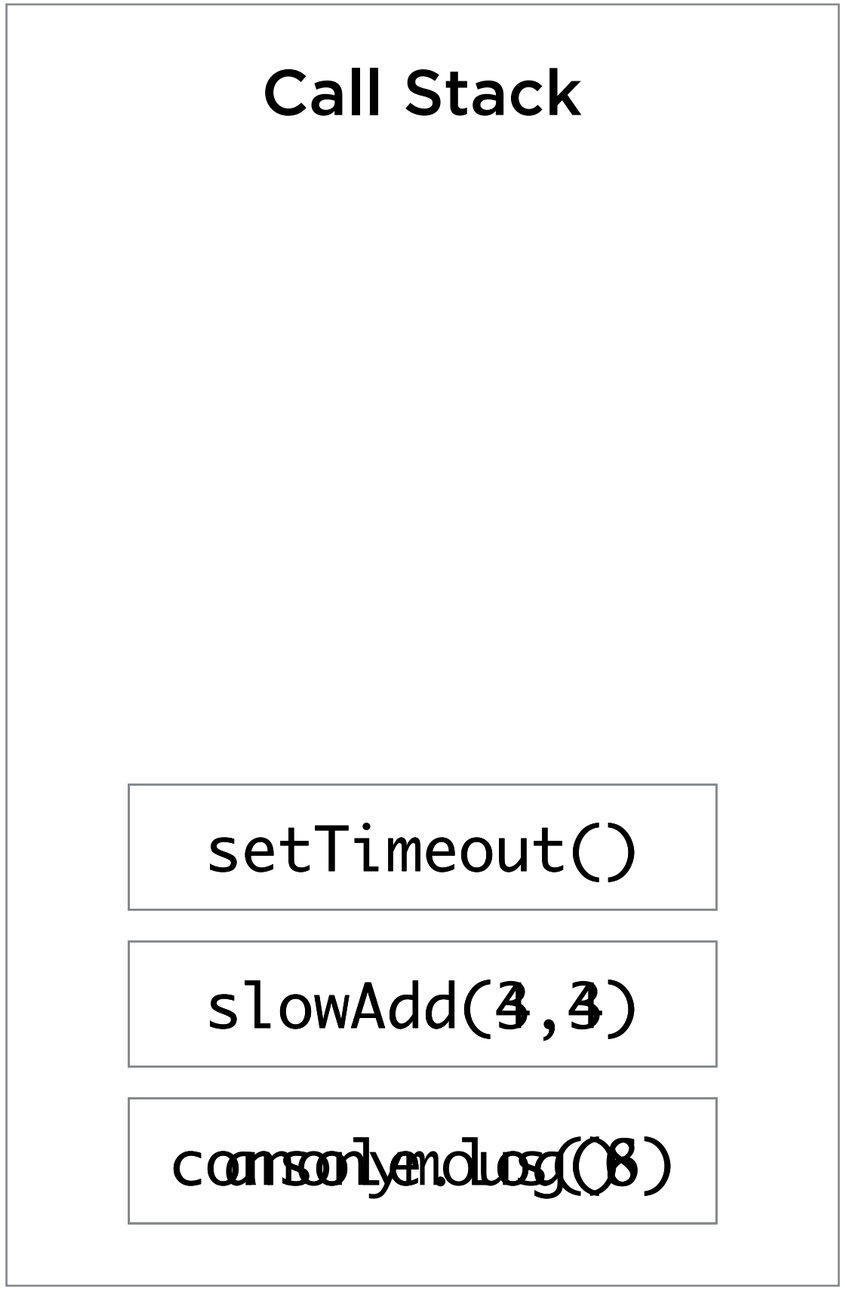
Call Stack

console.log(c)

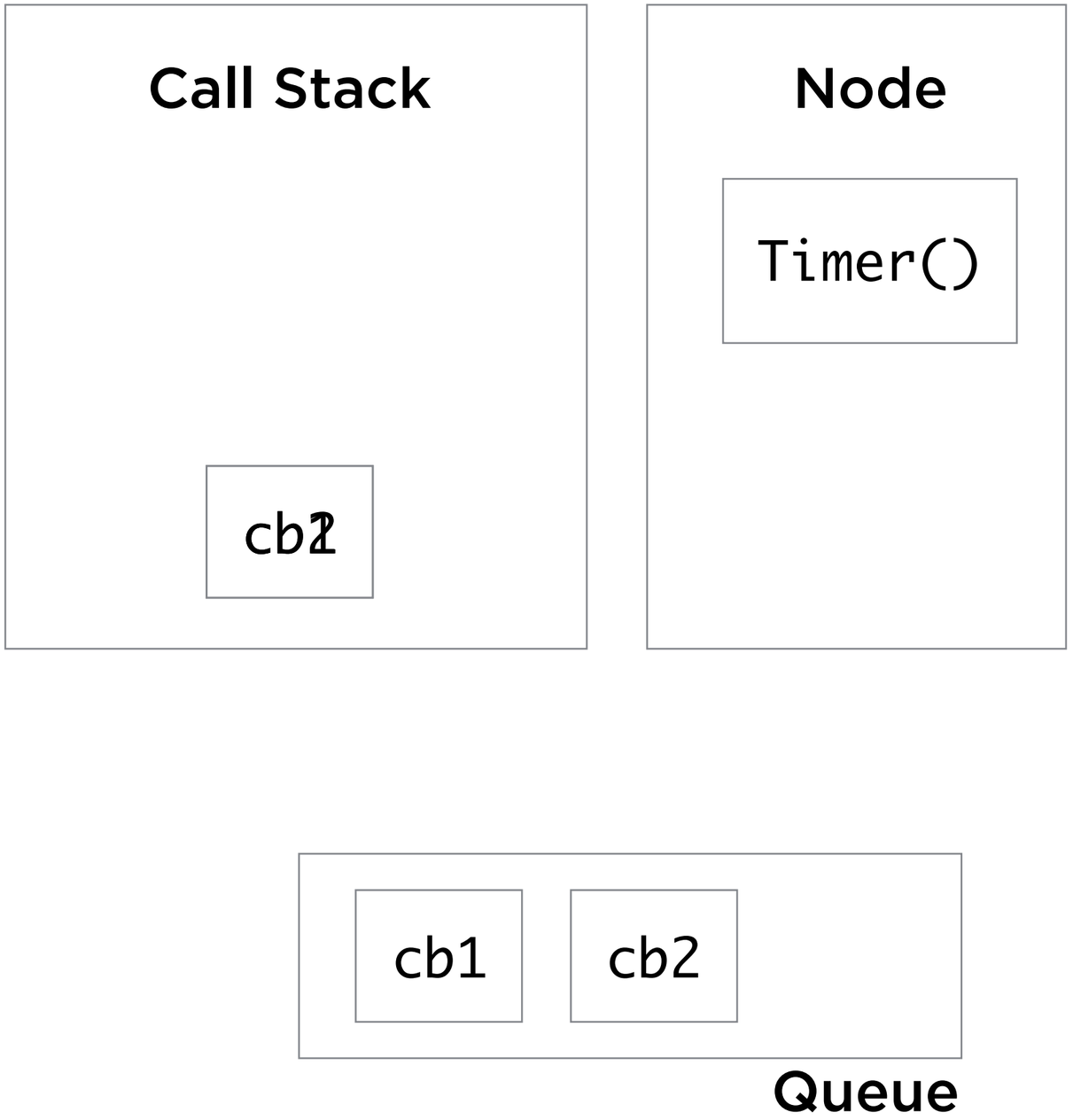
anonymous()

How Callbacks Actually Work

```
const slowAdd = (a, b) => {
  setTimeout(() => {
    console.log(a+b);
  }, 5000);
};

slowAdd(3, 3);

slowAdd(4, 4);
```

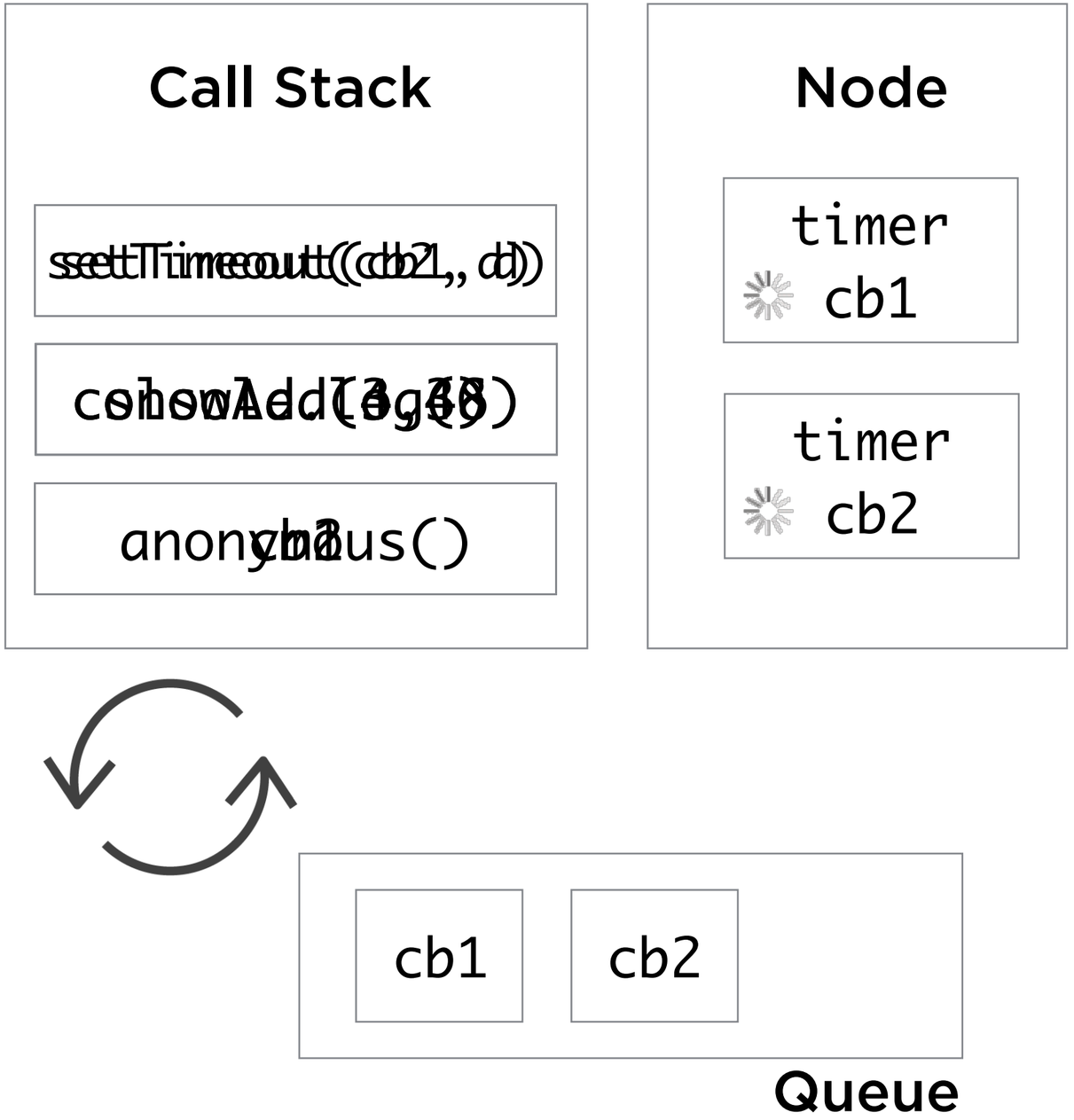


```
const slowAdd = (a, b) => {
  setTimeout(cb, 5000);
};

slowAdd(3, 3);

slowAdd(4, 4);
```



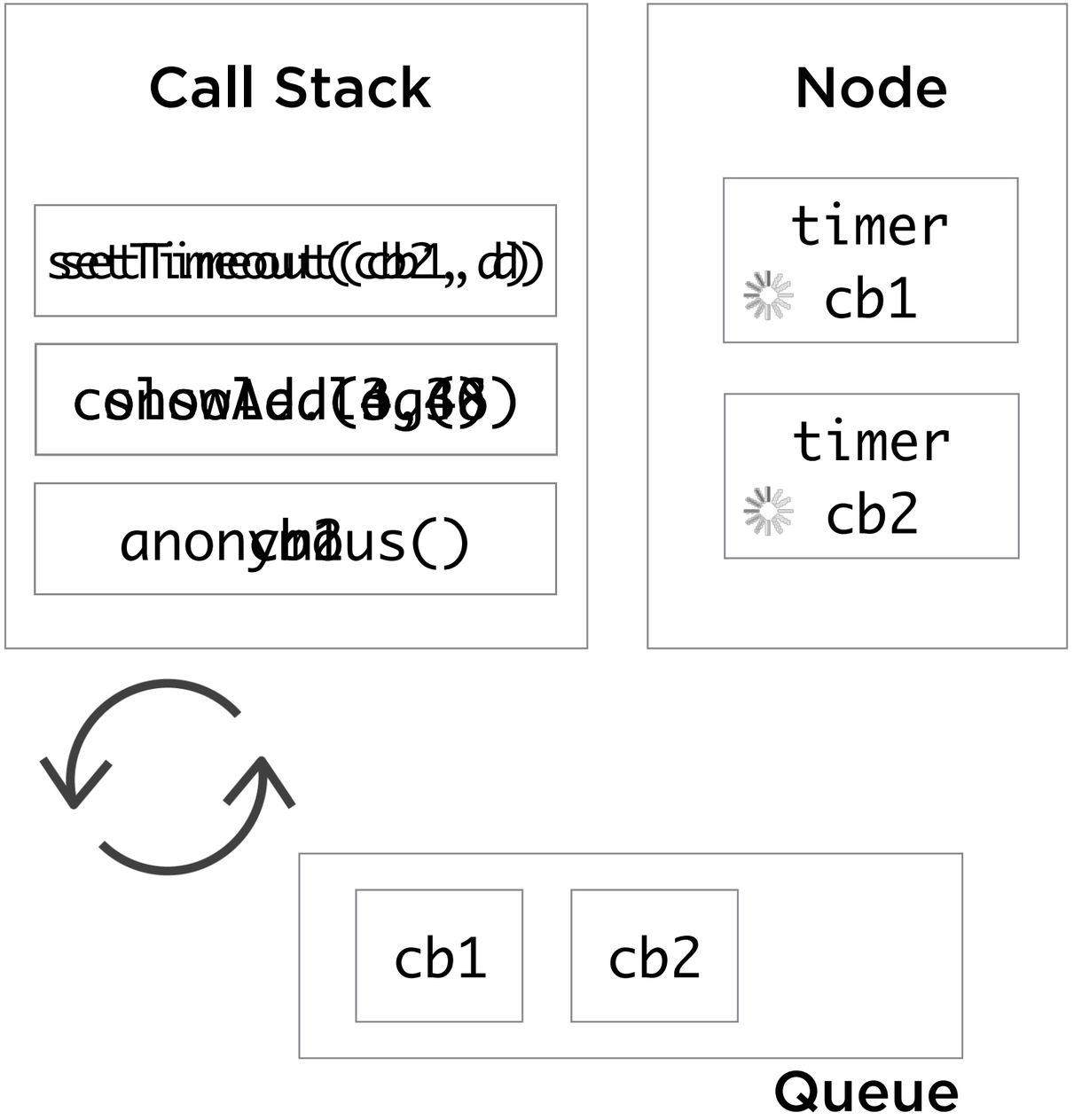
```
const slowAdd = (a, b) => {
  setTimeout(() => {
    console.log(a+b);
  }, 5000);
};

slowAdd(3, 3);

slowAdd(4, 4);
```

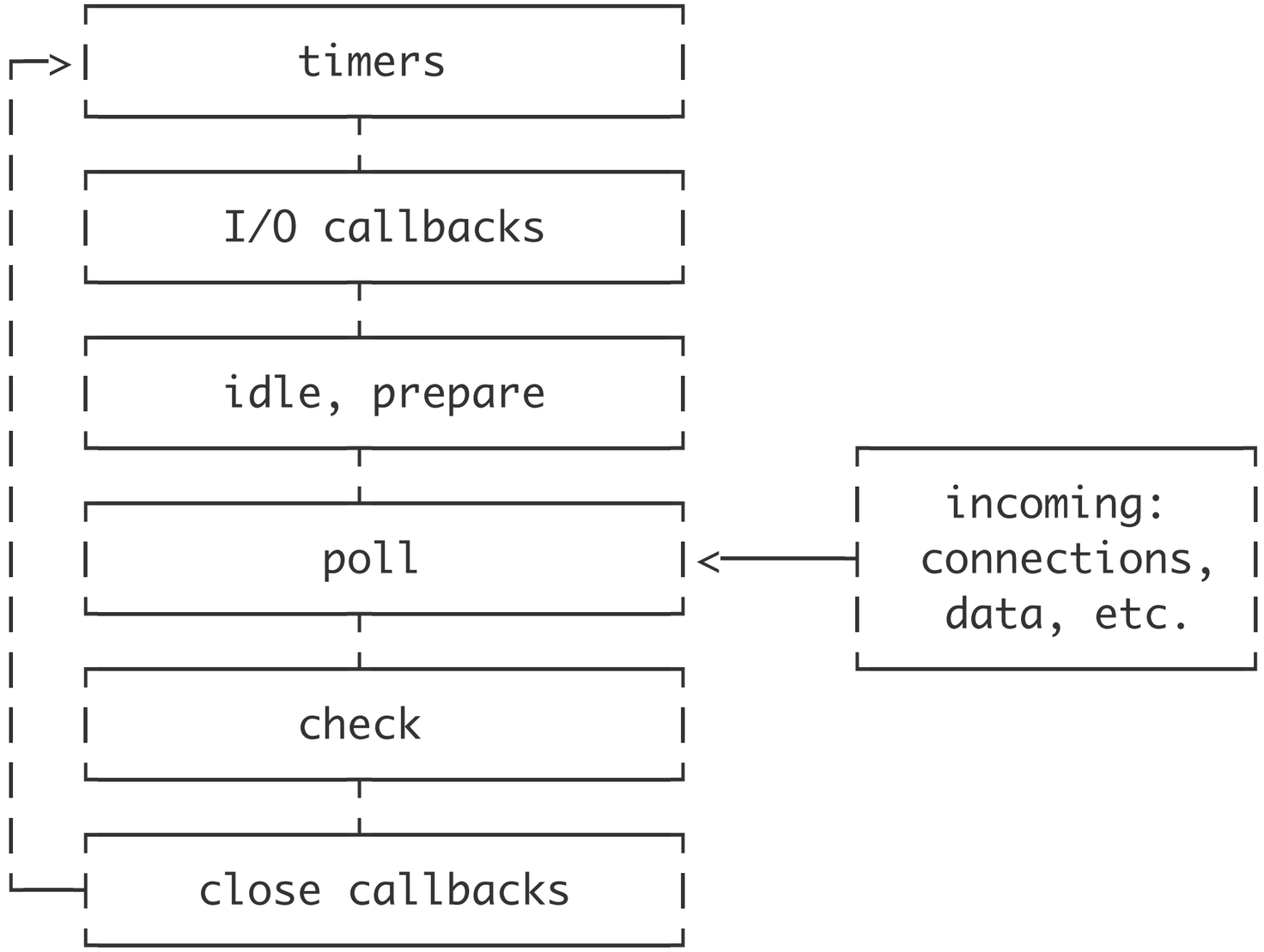
```
When the call stack gets empty:

While the queue is not empty:
  event = dequeue an event
  if there is a callback:
    call the event's callback
```



setImmediate & process.nextTick

```
const slowAdd = (a, b) => {
  setTimeout(() => {
    console.log(a+b);
  }, 0);
};

slowAdd(3, 3);

slowAdd(4, 4);
```





https://github.com/nodejs/node/blob/master/doc/topics/the-event-loop-timers-and-nexttick.md

Who Named These?

setTimeout	setImmediate	process.nextTick

Summary

The event loop

V8's call stack

Slow I/O operations

setTimeout

setImmediate

process.nextTick