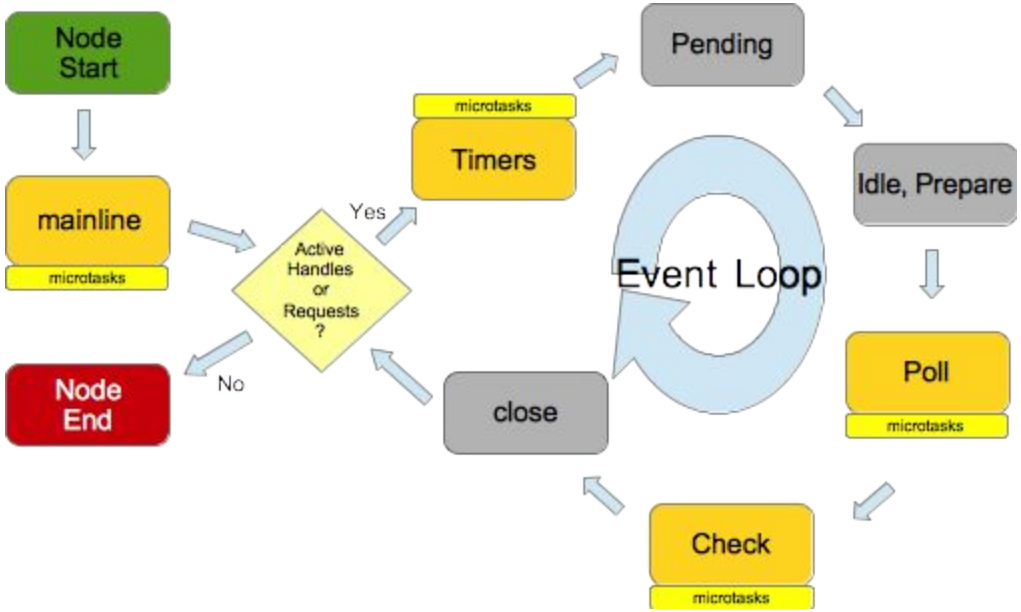
CS 337
Event Queue, Asynchronicity

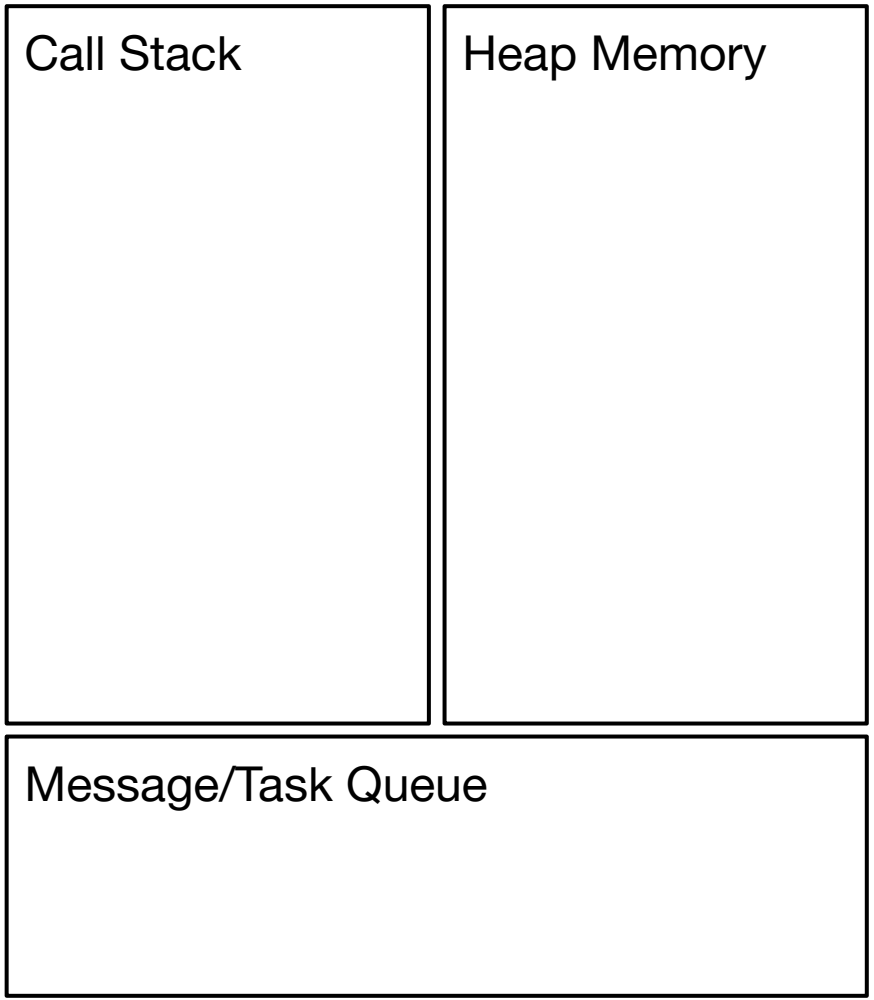
Benjamin Dicken

The Event Loop

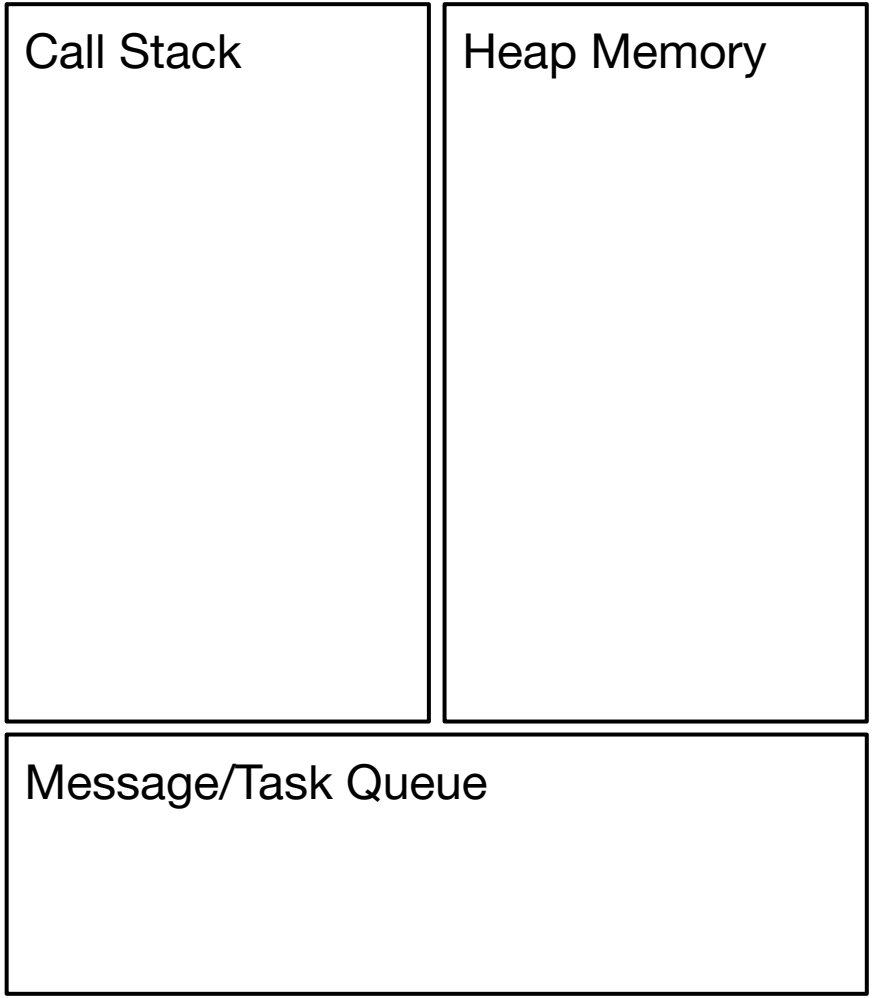
JavaScript has a runtime model based on an event loop, which is responsible for executing the code, collecting and processing events, and executing queued sub-tasks. This model is quite different from models in other languages like C and Java.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/EventLoop

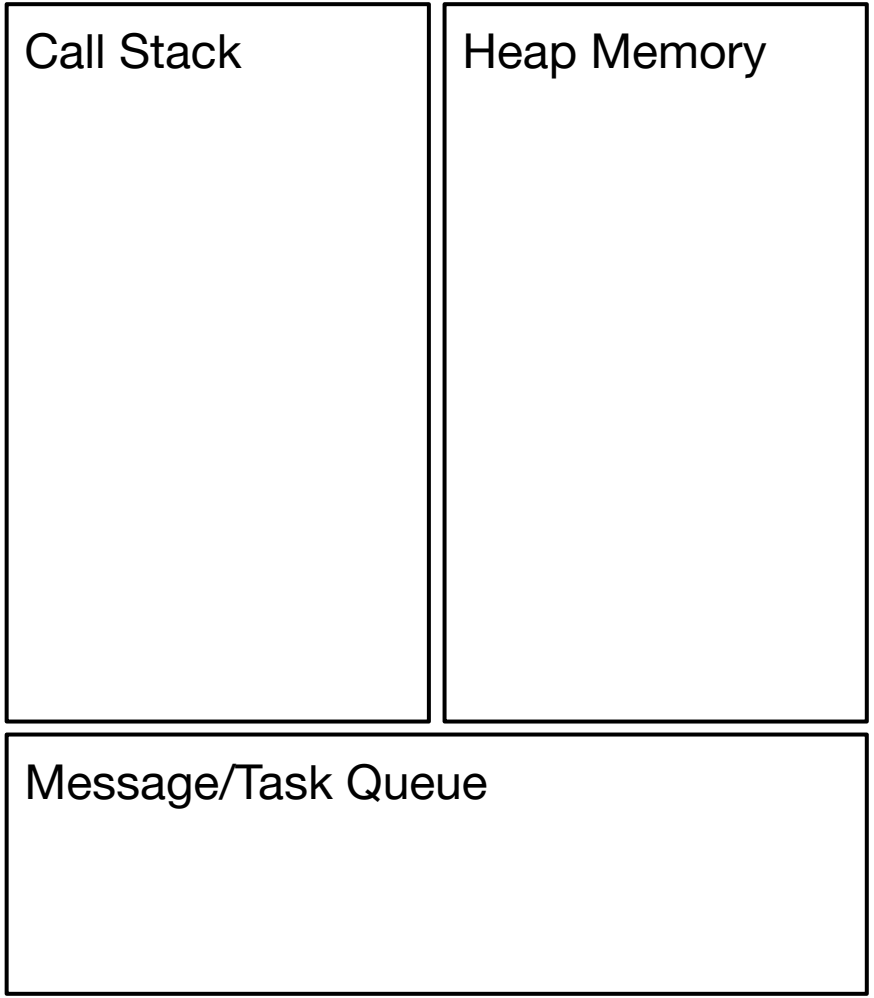

Node Start
mainline
microtasks
Node End
No
Active Handles or Requests ?
Yes
microtasks
Timers
Pending
Idle, Prepare
Poll
microtasks
Check
microtasks
close
Event Loop


Call Stack
Heap Memory
Message/Task Queue

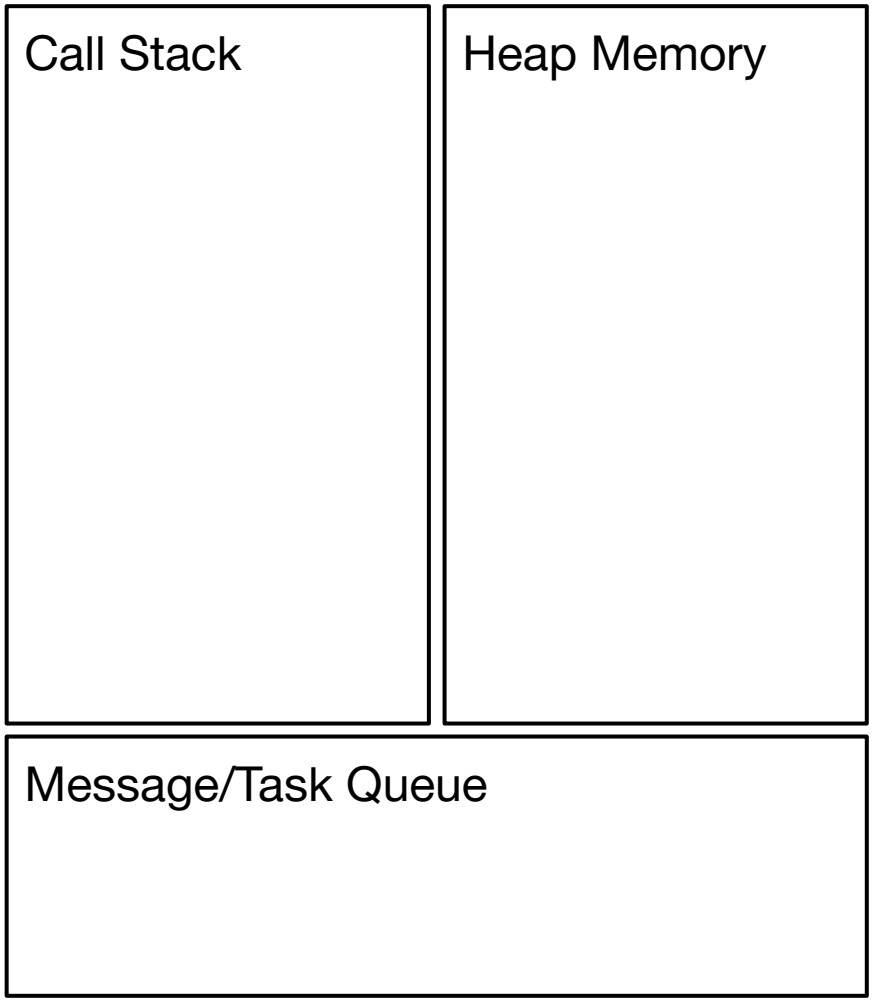
```
function language() {
  console.log(LANGUAGE);
}

function operation() {
  x = {'cats': 5};
  console.log('OPERATION');
  language()
  x['dogs'] = 20;
}

function android() {
  z = [10, 50];
  console.log('ANDROID');
  operation()
  z.push(70);
}

android()
```



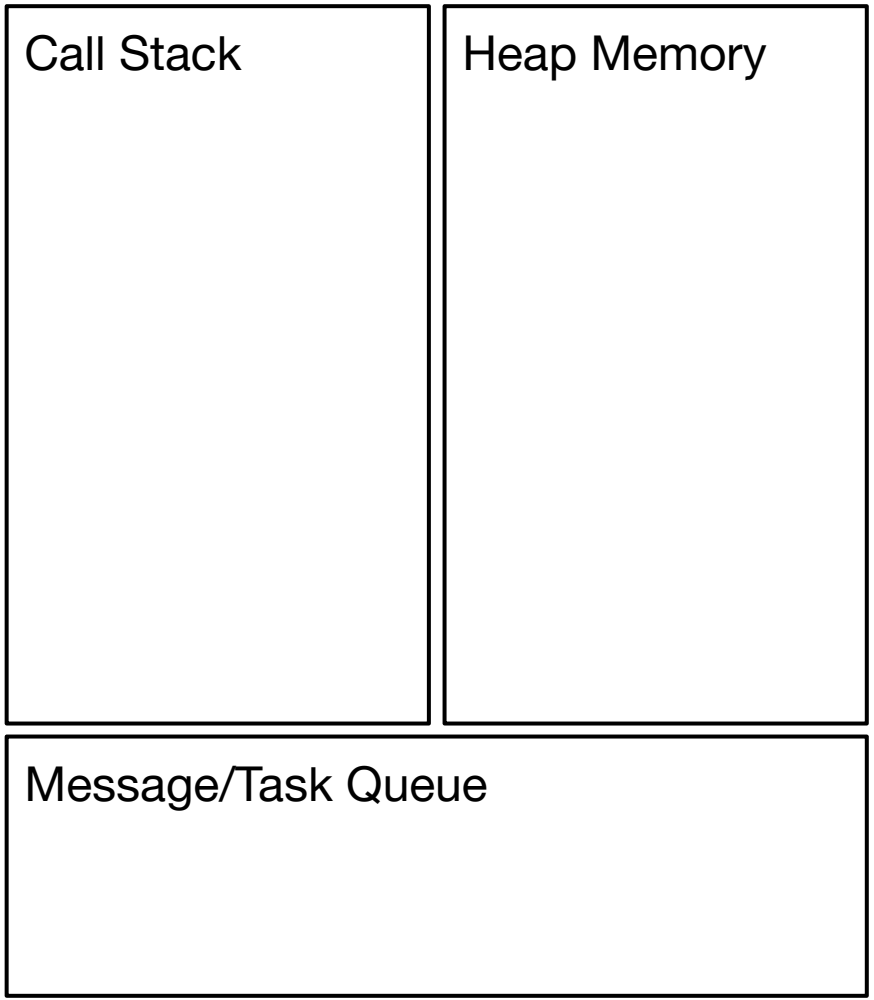
```
function xyz() {
    function doSomething() {
      console.log('elephant');
    }
    console.log('zebra');
    setTimeout(doSomething, 0);
    console.log('antelope');
}

xyz();
```



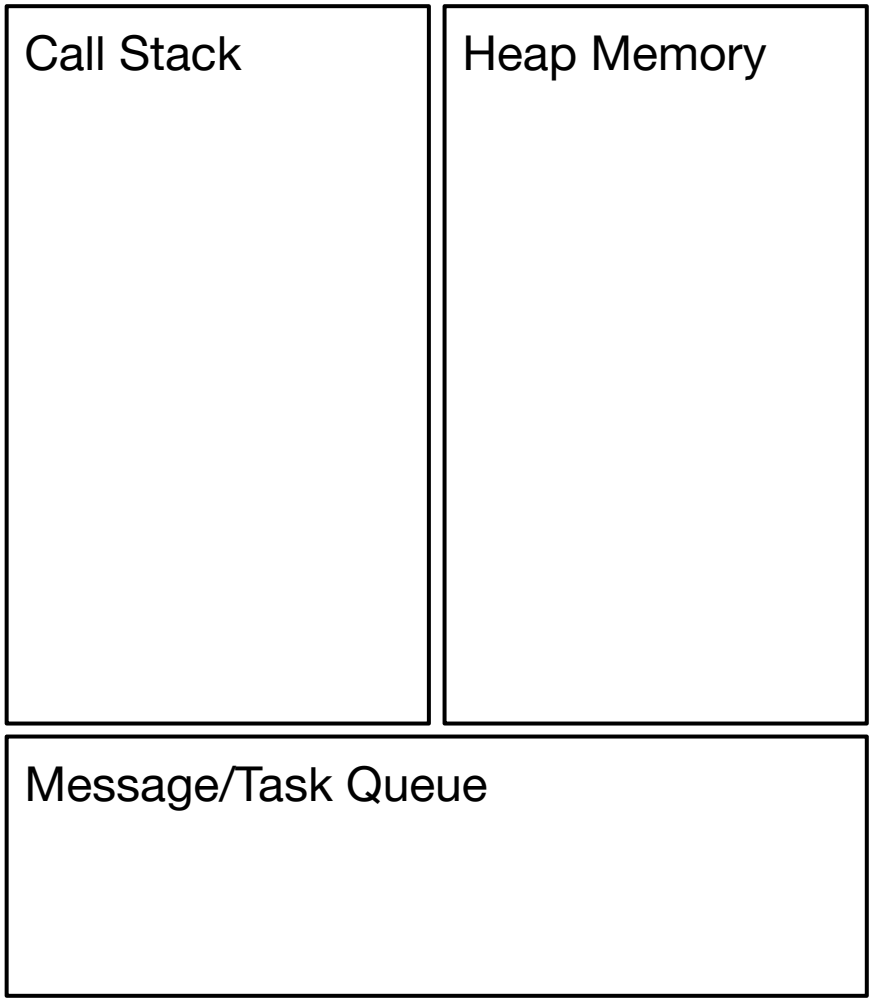
```
function xyz() {
    function doSomething() {
      console.log('elephant');
    }
    function another() {
      console.log('bear');
    }
    console.log('zebra');
    setTimeout(doSomething, 0);
    console.log('antelope');
    setTimeout(another, 0);
    console.log('tarantula');
}

xyz();
```



```
function xyz() {
    function another() {
      console.log('bear');
    }
    function doSomething() {
      console.log('elephant');
      setTimeout(another, 0);
      console.log('tiger');
    }
    console.log('zebra');
    setTimeout(doSomething, 0);
    console.log('antelope');
}

xyz();
```



```
function main() {
const lr = require('line-reader');
console.log('giraffe');
lr.eachLine('./bear.txt', (line, last) => {
  console.log('bear');
  lr.eachLine('./elephant.txt', (line, last) => {
    console.log('elephant');
  });
});
console.log('zebra');
}

main()
```



Activity

```
1  function testing1() {
2    console.log('A');
3    var httpRequest = new XMLHttpRequest();
4    if (!httpRequest) { return false; }
5    console.log('B');
6    httpRequest.onreadystatechange = () => {
7      if (httpRequest.readyState === XMLHttpRequest.DONE) {
8        if (httpRequest.status === 200) {
9          console.log('C');
10       } else { alert('Response failure'); }
11     }
12   }
13   console.log('D');
14   let url = '/test/1/';
15   httpRequest.open('GET', url);
16   httpRequest.send();
17   console.log('E');
18 }
```

What will this print?

Promise Queue

The ECMAscript 2015 (ES6) specification requires tasks to run Promise reaction callbacks to form their own job queue called "PromiseJobs".

https://stackoverflow.com/questions/40880416

```
function xyz() {
    function something() {
      console.log('elephant');
    }
    let url = 'http://localhost/path';
    var p = fetch(url);
    p.then((response) => {
      console.log('hello');
    });
    setTimeout(something, 0);
}

xyz();
```