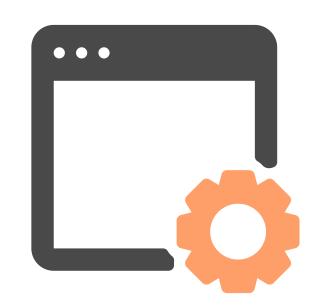
Asynchronous Control Flow

M1W4 – October 18 Cohort



Single Threaded

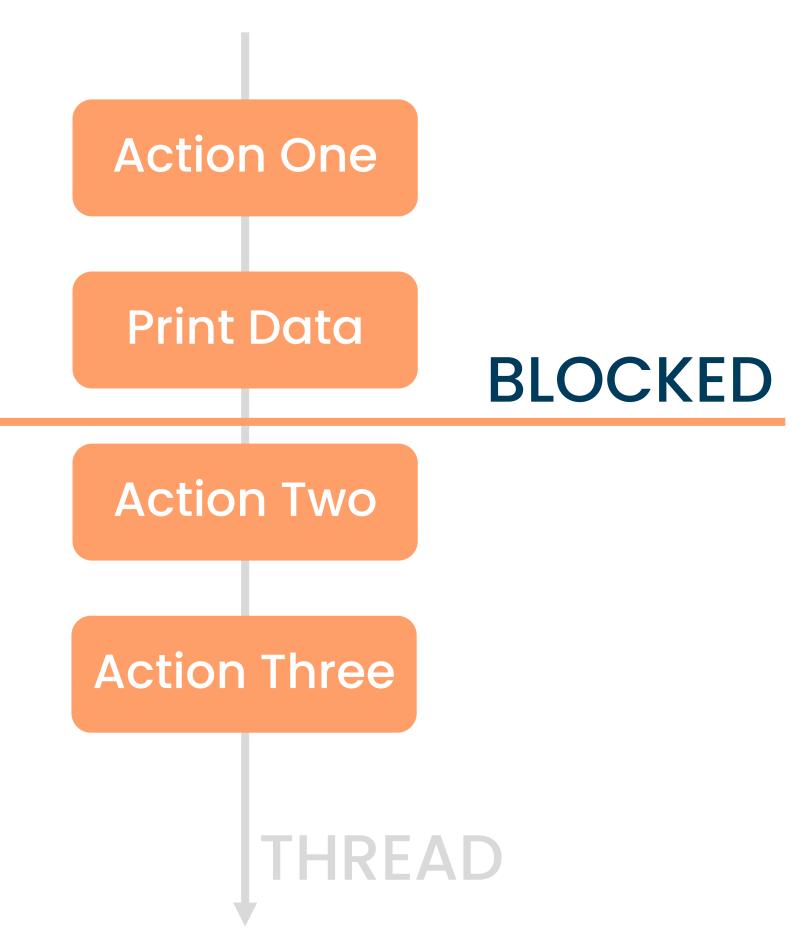
Each code is executed one after the other

```
Action One
Action Two
Action Three
     THREAD
```

```
function actionOne() {
       console.log("(1) Action One!");
     function actionTwo() {
       console.log("(2) Action Two!");
     function actionThree() {
       console.log("(3) Action Three!");
10
11
     actionOne();
     actionTwo();
     actionThree();
16
17
```

Single Threaded

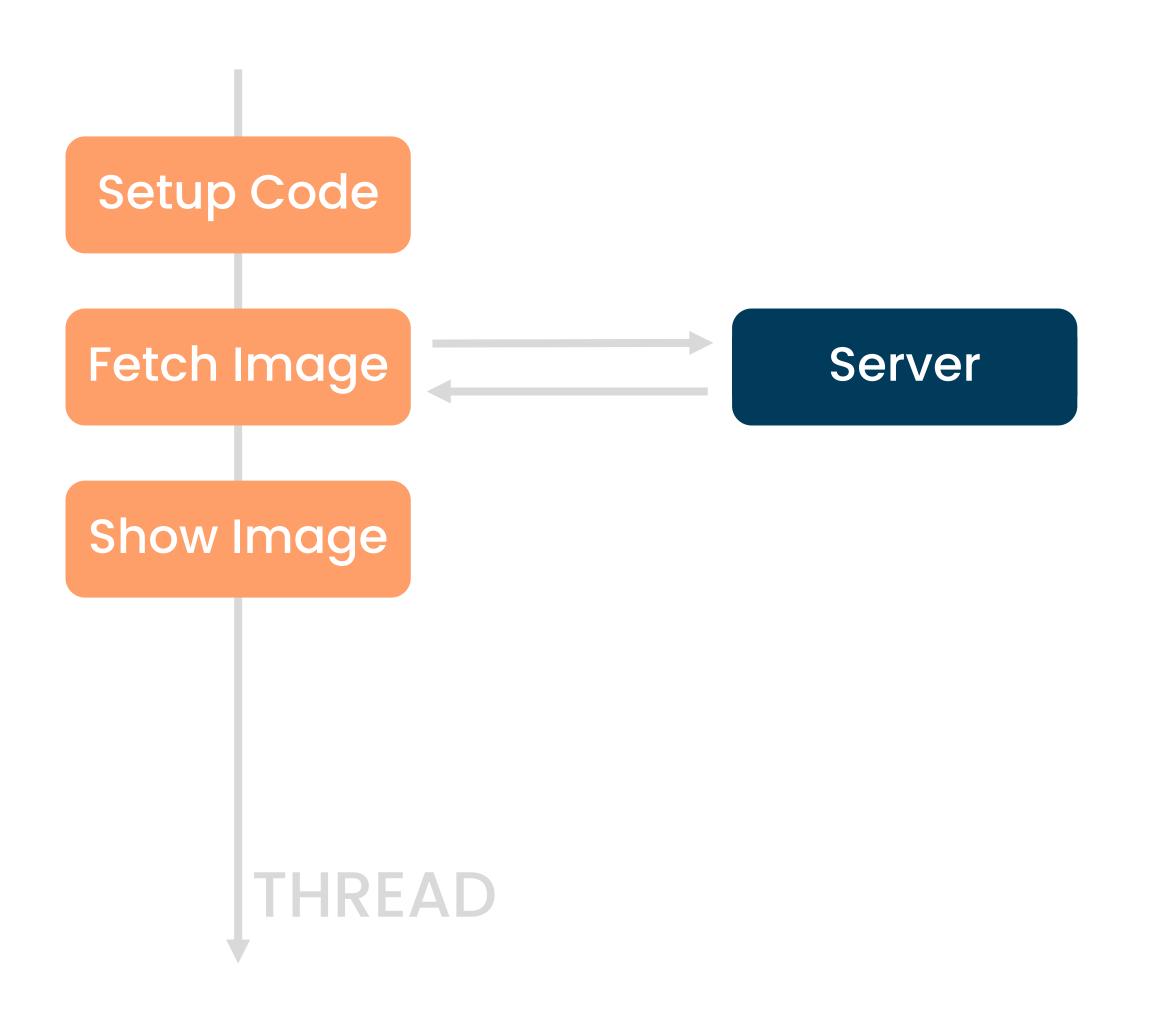
Each code is executed one after the other



```
function actionOne() {
       console.log("(1) Action One!");
     function printData() {
       const stop = 1000;
       for (let i = 0; i < stop; i++) {
         const date = new Date();
         console.log(date);
10
11
12
     function actionTwo() {
14
       console.log("(2) Action Two!");
15
16
     function actionThree() {
       console.log("(3) Action Three!");
18
19 }
20
    actionOne();
    printData();
23 actionTwo();
24 actionThree();
```

Callbacks

How do we return values from Asynchronous Operations?



```
import fetch from "node-fetch";
     const url = "https://dog.ceo/api/breeds/image/random";
     const response = fetch(url)
       .then((response) => {
         return response.json();
       .then((data) => {
         console.log("data", data);
10
      });
11
12
     const showImage = response.data;
     console.log("showImage", showImage);
15
16
```

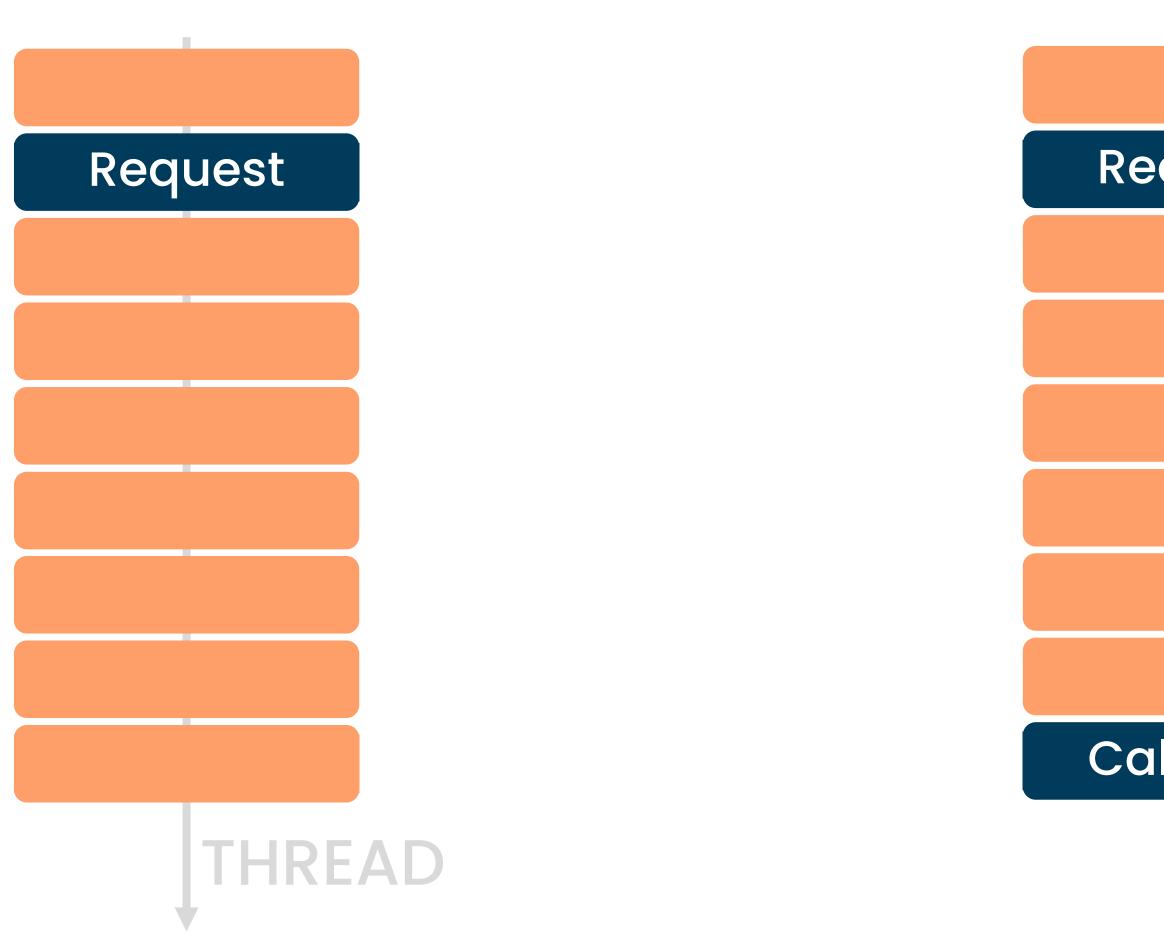
Callbacks

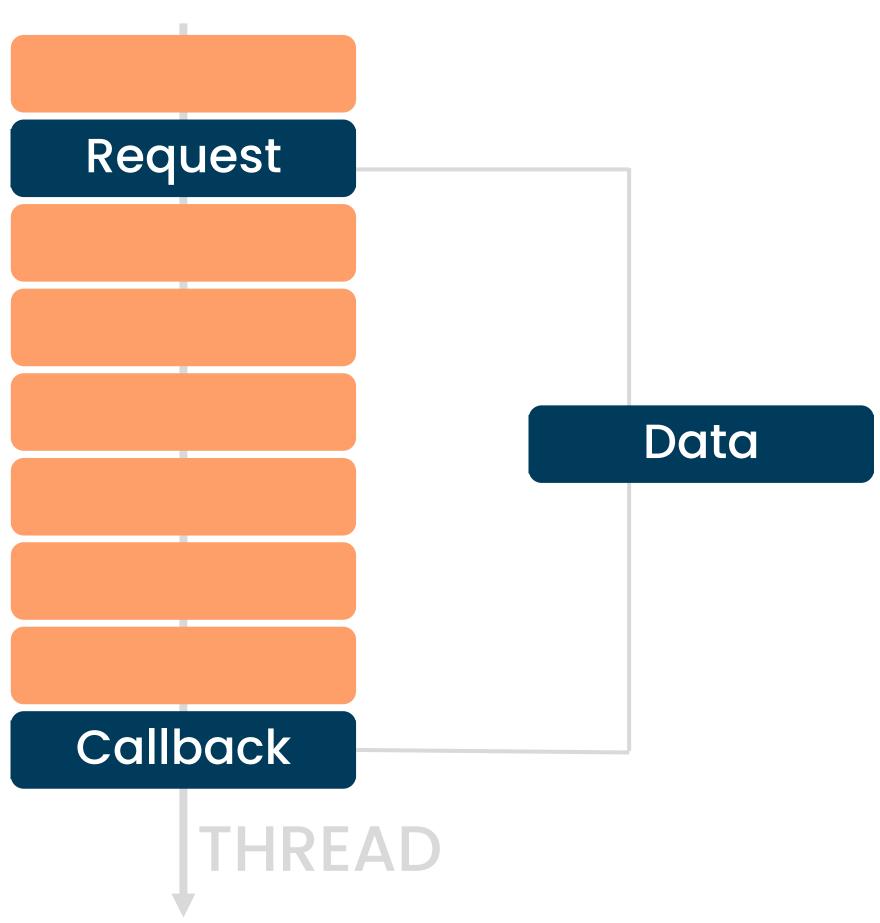
How do we return values from Asynchronous Operations?

```
- const map = (list, callback) => {
            const output = [];
            for (const element of list) {
higher
order
              output.push(callback(element));
function
            return output;
          const result = map([1, 2, 3], (num) \Rightarrow num * num);
          console.log(result);
                                              anonymous function
```

Blocking vs Non-Blocking

Synchronous vs Asynchronous Operations





What's happening under the hood?

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
call stack
```

```
const addition = (num1, num2) => {
                                                                                  web/c++
                                                    call stack
return num1 + num2;
};
                                                                  task queue
```

```
web/c++
const addition = (num1, num2) => {
                                                     call stack
return num1 + num2;
};
const sayHi = (name) => {
return `Hi ${name}`;
};
                                                                   task queue
```

```
web/c++
const addition = (num1, num2) => {
                                                      call stack
return num1 + num2;
};
const sayHi = (name) => {
return `Hi ${name}`;
};
addition(3, 8);
                                             addition(3, 8);
                                                                    task queue
```

```
web/c++
const addition = (num1, num2) => {
                                                      call stack
return num1 + num2;
};
const sayHi = (name) => {
return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
                                                                            callback1
                                             setTimeout(callback1);
                                                                     task queue
```

```
web/c++
const addition = (num1, num2) => {
                                                      call stack
return num1 + num2;
};
const sayHi = (name) => {
return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
                                                                            callback1
                                                                     task queue
```

What's happening under the hood?

```
const addition = (num1, num2) => {
                                                       call stack
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
                                              sayHi("Maui");
                                                                      task queue
                                               callback1
```

web/c++

What's happening under the hood?

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
                                               callback1
```

web/c++call stack

What's happening under the hood?

callback1

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
         call stack
                              callback2
getData(callback2);
```

What's happening under the hood?

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
         call stack
                              callback2
callback1
```

What's happening under the hood?

callback2

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
call stack
```

What's happening under the hood?

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
        call stack
callback2
```

What's happening under the hood?

```
const addition = (num1, num2) => {
 return num1 + num2;
};
const sayHi = (name) => {
 return `Hi ${name}`;
};
addition(3, 8);
setTimeout(() => {
 console.log("I'm Asynchronous");
}, 1000);
sayHi("Maui");
getData((data) => {
 console.log(data);
});
```

```
web/c++
call stack
```