# Your Next Week

| Tuesday May 5 | Wednesday May 6 | Thursday May 7 | Friday May 8 |
|---|---|---|---|
| *6:30 PM*<br>**— DUE Class 15 Lab**<br>**— DUE Class 16 Reading**<br>— Class 16A | *6:30 PM*<br>— Class 16B<br><br>*MIDNIGHT*<br>**— DUE Class 16 Learning Journal** | *6:30 PM*<br>— Co-working | |
| **Saturday May 9** | **Sunday May 10** | **Monday May 11** | **Tuesday May 12** |
| *6:30 PM*<br>**— DUE Class 16 Code Challenge**<br>**— DUE Class 16 Lab**<br>**— DUE Class 17 Reading**<br>— Class 17<br>— Interview Prep 03<br><br>*MIDNIGHT*<br>**— DUE Class 17 Learning Journal** | *MIDNIGHT*<br>— Class 16-17 Feedback | | *6:30 PM*<br>**— DUE Class 17 Code Challenge**<br>**— DUE Class 17 Lab**<br>**— DUE Class 18 Reading**<br>— Class 18A |

# What We've Covered

*Module 01*

Javascript Fundamentals and Data Models

*C01 — Node Ecosystem, TDD, CI/CD*
*C02 — Classes, Inheritance, Functional Programming*
*C03 — Data Modeling & NoSQL Databases*
*C04 — Advanced Mongo/Mongoose*
*C05 — DSA: Linked Lists*

*Module 02*

API Servers

*C06 — HTTP and REST*
*C07 — Express*
*C08 — Express Routing & Connected API*
*C09 — API Server*
*C11 — DSA: Stacks and Queues*

*Module 03*

Auth/Auth

*C10 — Authentication*
*C12 — OAuth*
*C13 — Bearer Authorization*
*C14 — Access Control (ACL)*
*C15 — DSA: Trees*

*Module 04*

Realtime

**C16 — Event Driven Applications**
C17 — TCP Server
C18 — Socket.io
C19 — Message Queues
C20 — Midterms Prep

Midterms

*Module 05*

React Basics

C21 — Component Based UI
C22 — React Testing and Deployment
C23 — Props and State
C24 — Routing and Component Composition
C25 — DSA: Sorting and HashTables

*Module 06*

Advanced React

C26 — Hooks API
C27 — Custom Hooks
C28 — Context API
C29 — Application State with Redux
C30 — DSA: Graphs

*Module 07*

Redux State Management

C31 — Combined Reducers
C32 — Asynchronous Actions
C33 — Additional Topics
C34 — React Native
C35 — DSA: Review

*Module 08*

UI Frameworks

C36 — Gatsby and Next
C37 — JavaScript Frameworks
C38 — Finals Prep

Finals

# Lab 15 Review

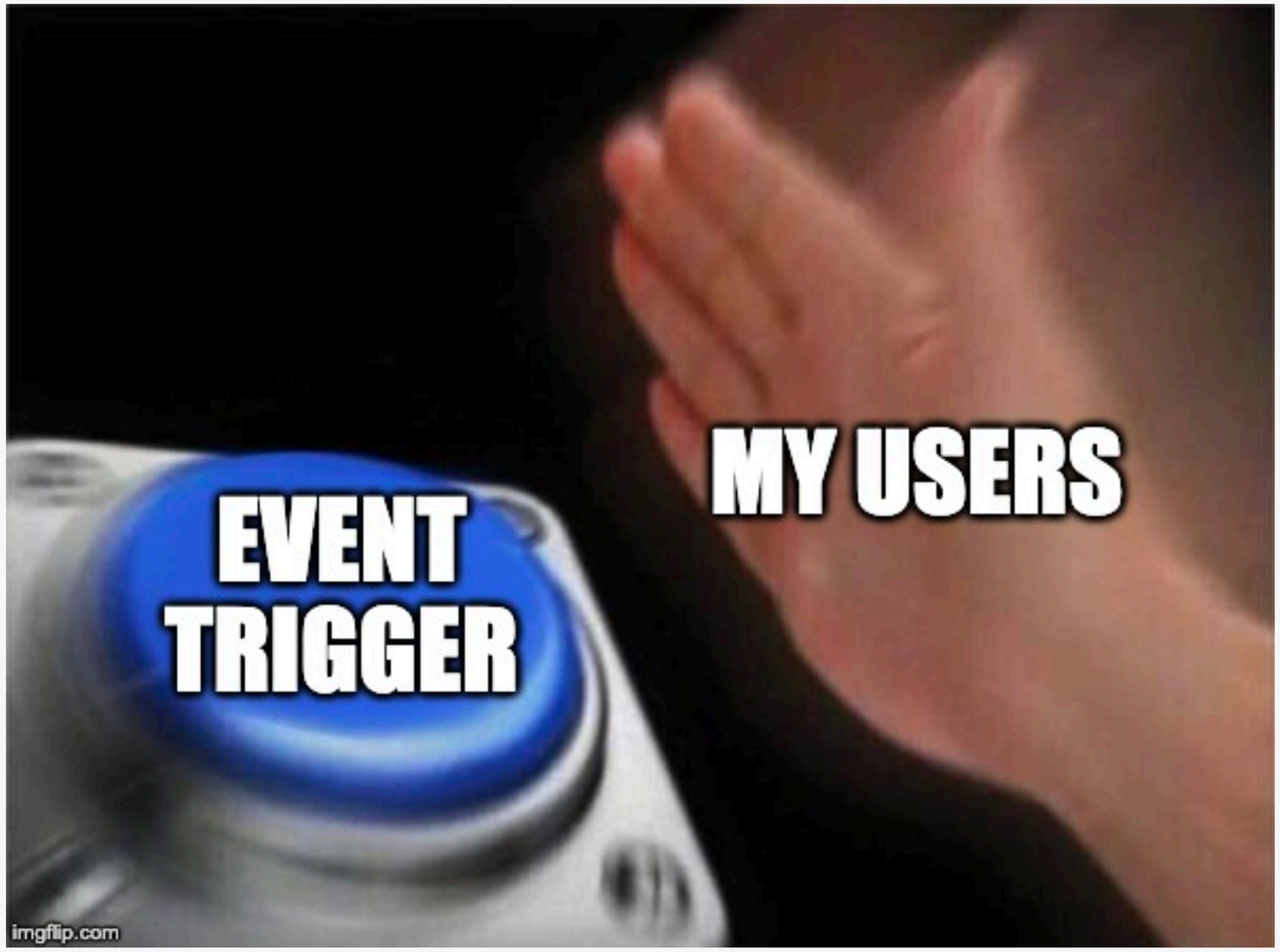
# Class 16

# Event Driven Applications

seattle-javascript-401n16

# What is an Event?

- A signal that something happened
    - Someone clicked a button, pressed a key, etc.
- Events are system-wide or application-wide
- They are raised/triggered/**emitted**
- Anywhere in the application (or system) you can **listen** for events and respond to them



# Listeners and Handlers

- Events by themselves don’t do anything
    - Like a person shouting in a forest. If no one is there to hear it…did they make a sound?
- If we care about an event, we create a **listener**
- The listener runs a **handler** function when it “hears” an event
- There can be multiple listeners for an event!

# You might be familiar with…

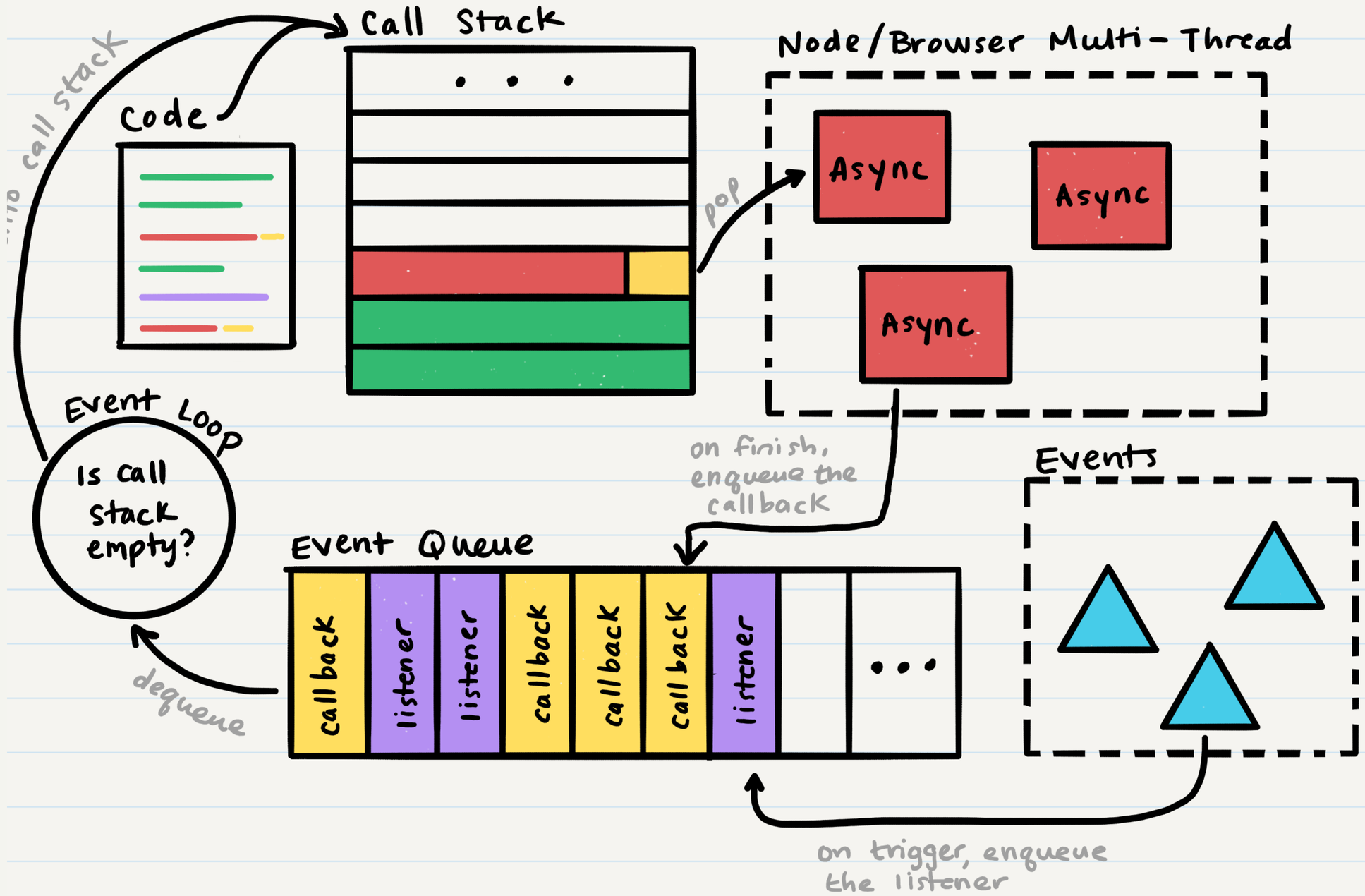
```
<form onsubmit="handleSubmit(event)"></form>

<button onclick="handleClick(event)"></button>


window.addEventListener("resize", function() {} );

$("#my-btn").click(function() {});
```

These are listeners and handlers!


Call Stack
Code
Node/Browser Multi-Thread
Async
Async
Async
pop
Event Loop
Is call stack empty?
on finish, enqueue the callback
Events
Event Queue
callback
listener
listener
callback
callback
callback
listener
dequeue
on trigger, enqueue the listener

# Lab 16 Overview