Exploring Asynchronous Code in JavaScript

GDG Algiers

Synchronous vs Asynchronous

Synchronous

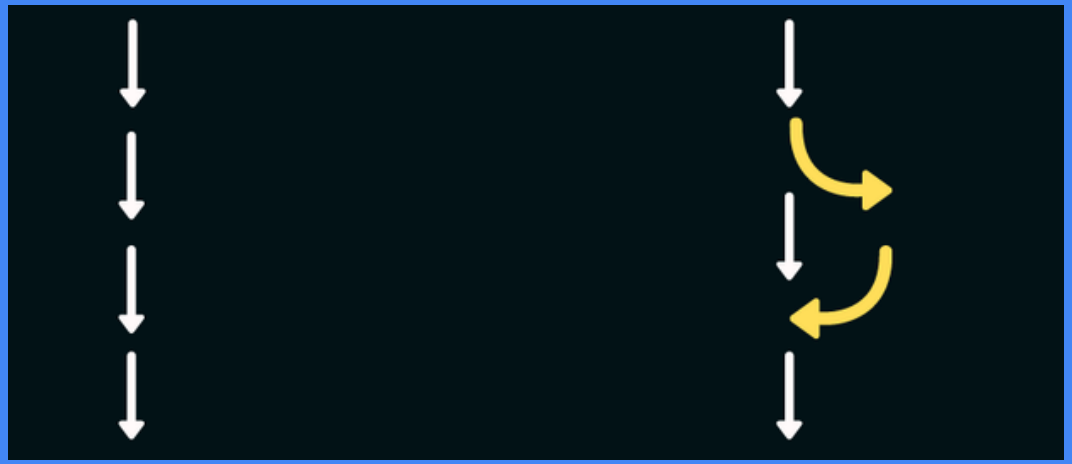
Synchronous code runs sequentially, meaning each line is executed one after the other in a specific order. It waits for each operation to complete before moving on to the next line.

Asynchronous

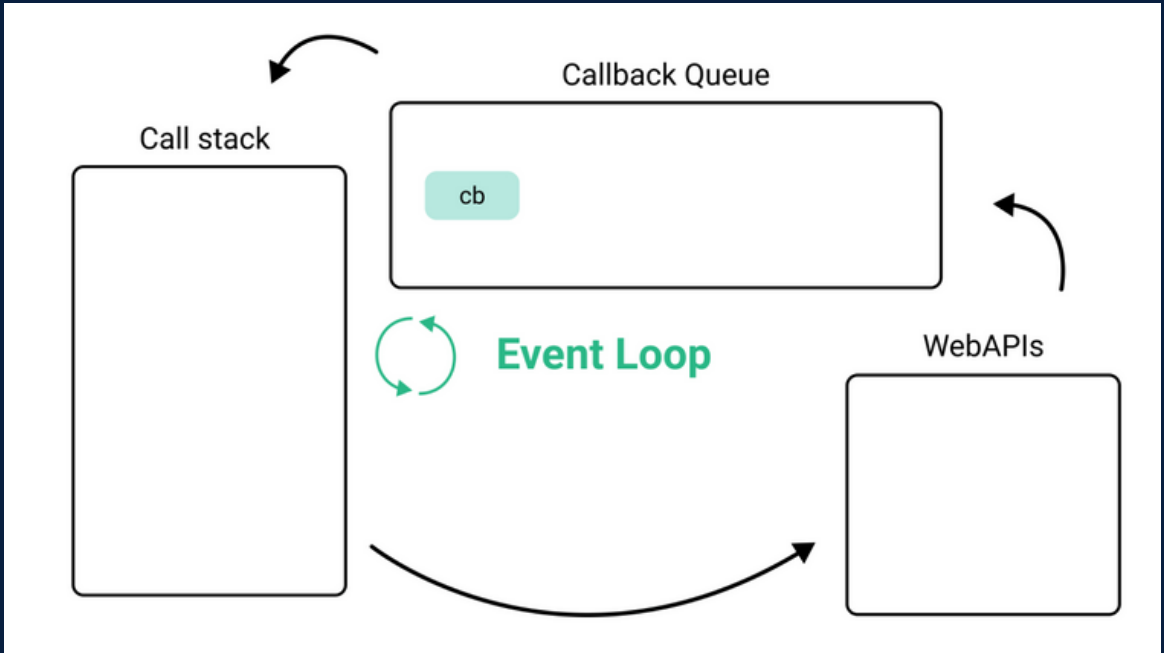
Asynchronous code allows for multiple operations to be performed simultaneously, without waiting for each operation to complete before moving on to the next line. This can make programs more efficient and responsive, particularly for long-running or I/O-intensive tasks.

Synchronous

Asynchronous



event loop



Callbacks

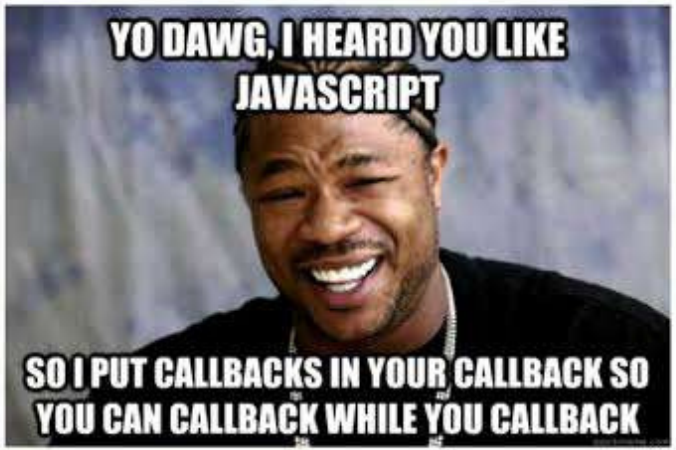
Callbacks are functions passed to another functions as arguments, which are then invoked inside the outer function


JS

Callback Hell

Callback hells

```
getData(function(a) {
 getMoreData(function(b) {
  getMoreData(function(c) {
   getMoreData(function(d) {
    getMoreData(function(e) {
     // do something
    });
   });
  });
 });
})
```



Promises

Web it up

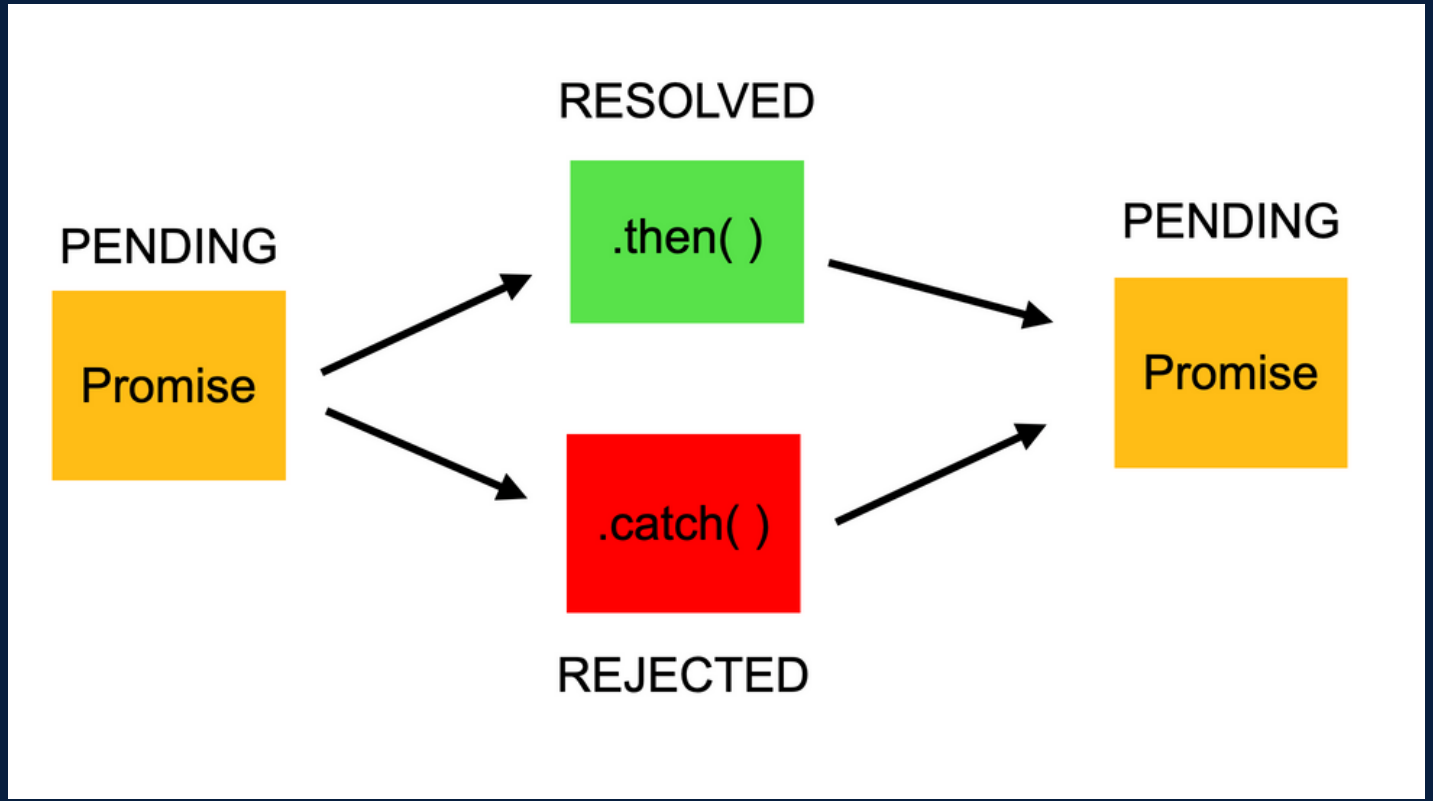
Promise states

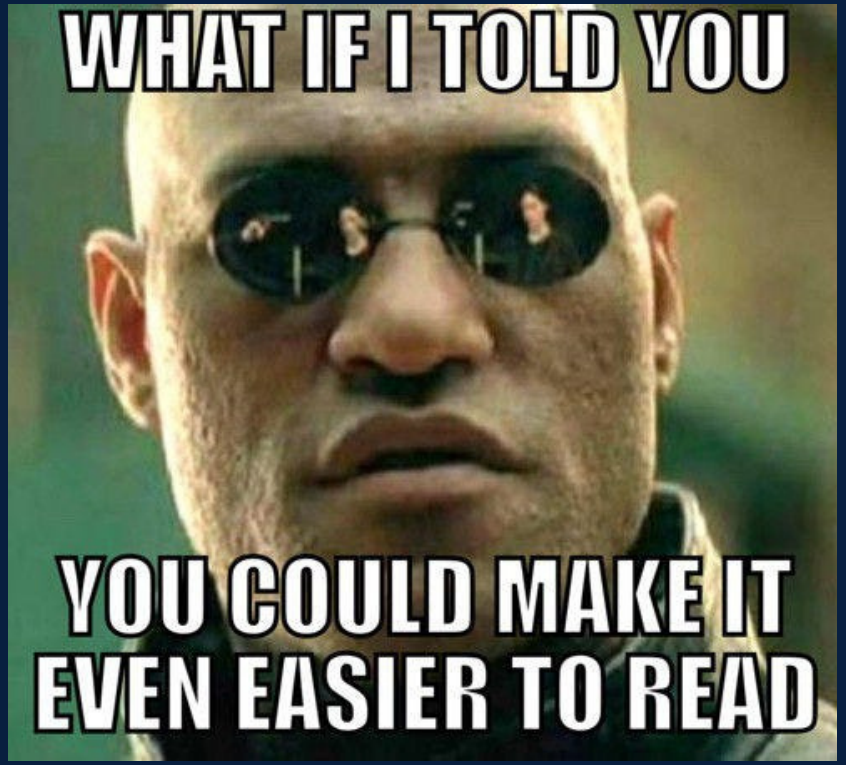
pending

resolved

rejected

Promises


RESOLVED
PENDING
.then( )
PENDING
Promise
Promise
.catch( )
REJECTED


WHAT IF I TOLD YOU
YOU COULD MAKE IT
EVEN EASIER TO READ

Async/await

Using Async/await to Write Clean and Readable Asynchronous Code

GDG Algiers

uses

Web it up

Here are some common use cases for asynchronous code:

Network requests

user interfaces

File I/O

Helpful resources

Thank you for your time

GDG Algiers