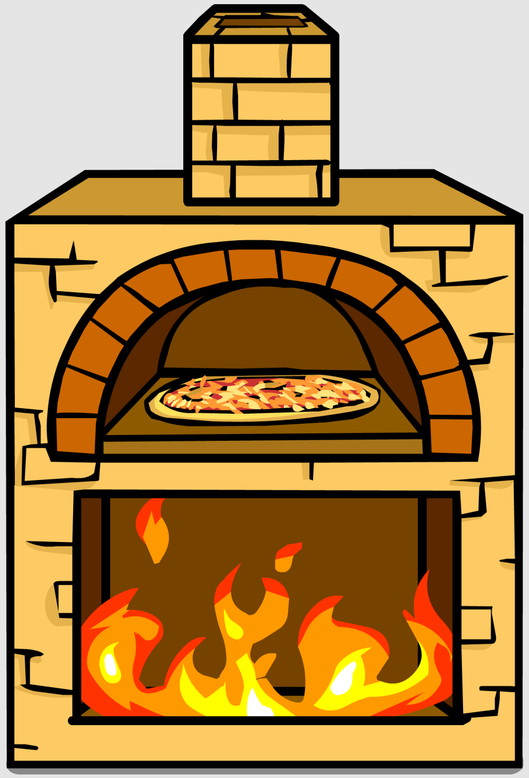
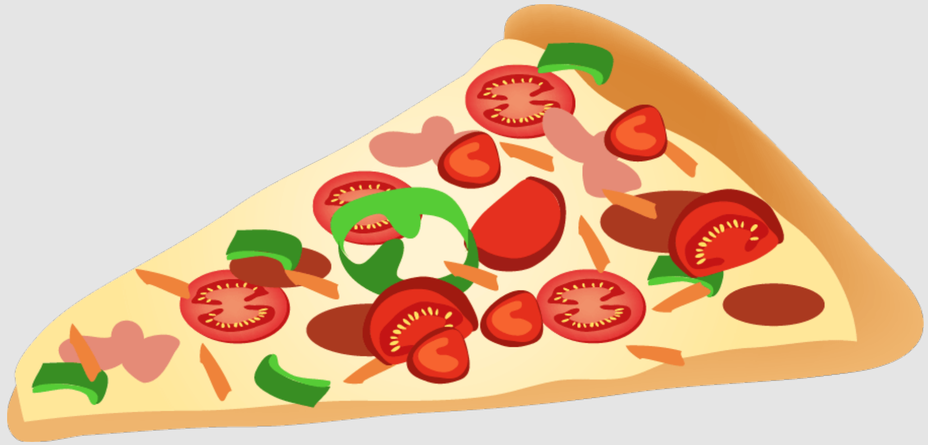
Recap

- GitHub
- Node, npm
- Datatypes and Array methods
- Data structures
- Testing with Jest
- TDD

Async, Promises and File System





Agenda

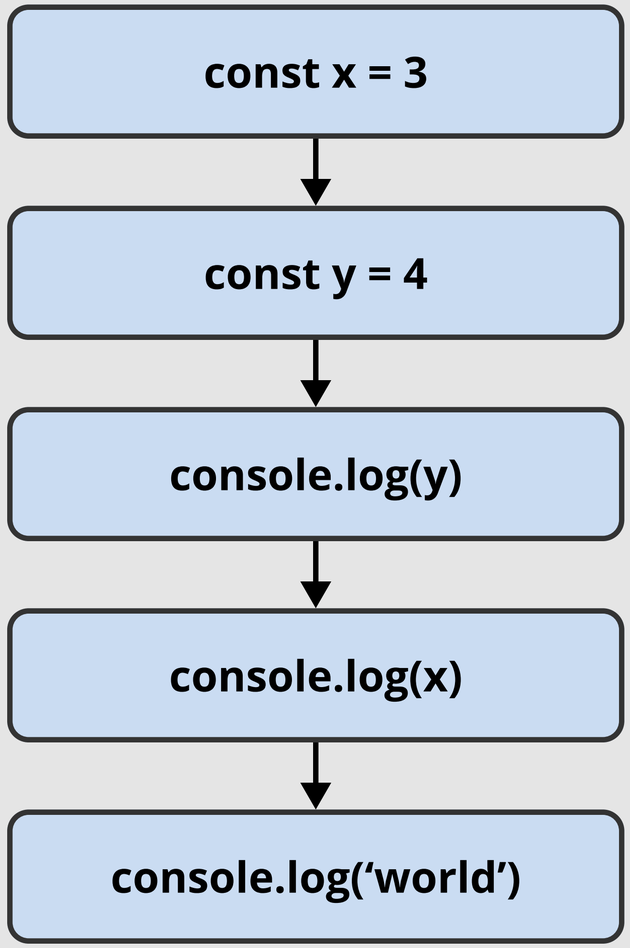
- **Asynchronous Code**
  - Code that starts now and finishes later

- **Promises**
  - A special kind of object that is *promising* to eventually do or get something
  - A way of working with Async

- **File System (fs)**
  - How we access the files on a computer in Node

Async

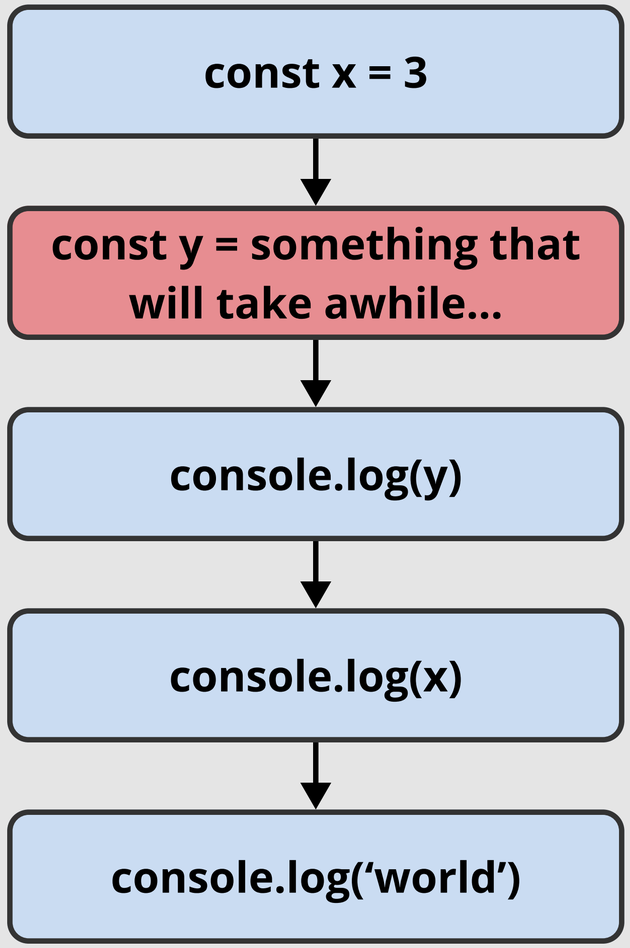
Synchronous Code in JavaScript



- Blocking code
- Most JavaScript runs really fast
- But some actions aren't so speedy
  - Internet requests
  - Database calls
  - File Systems

Async

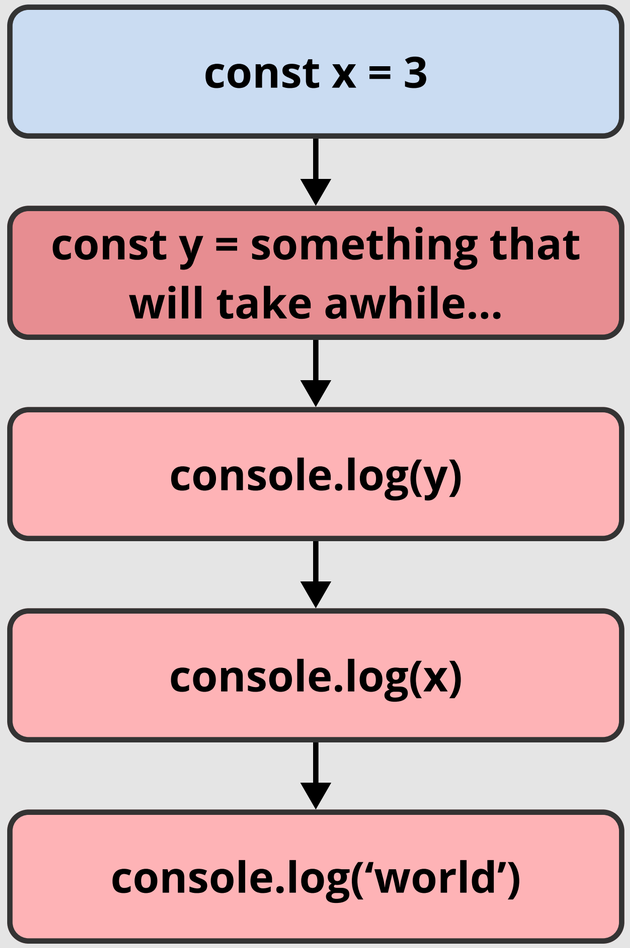
Synchronous Code in JavaScript



- Blocking code
- Most JavaScript runs really fast
- But some actions aren't so speedy
  - Internet requests
  - Database calls
  - File Systems

Async

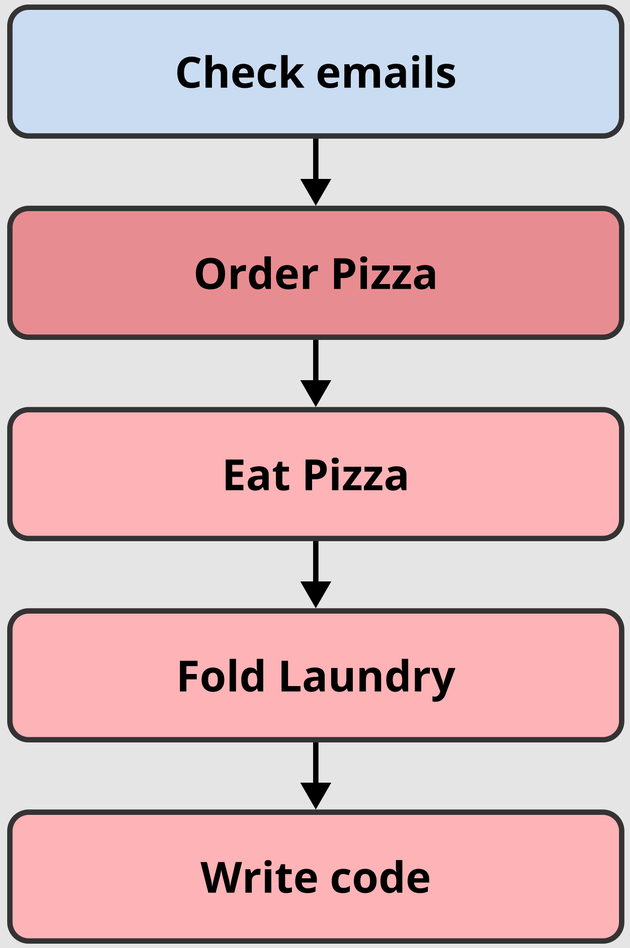
Synchronous Code in JavaScript



- Blocking code
- Most JavaScript runs really fast
- But some actions aren't so speedy
  - Internet requests
  - Database calls
  - File Systems

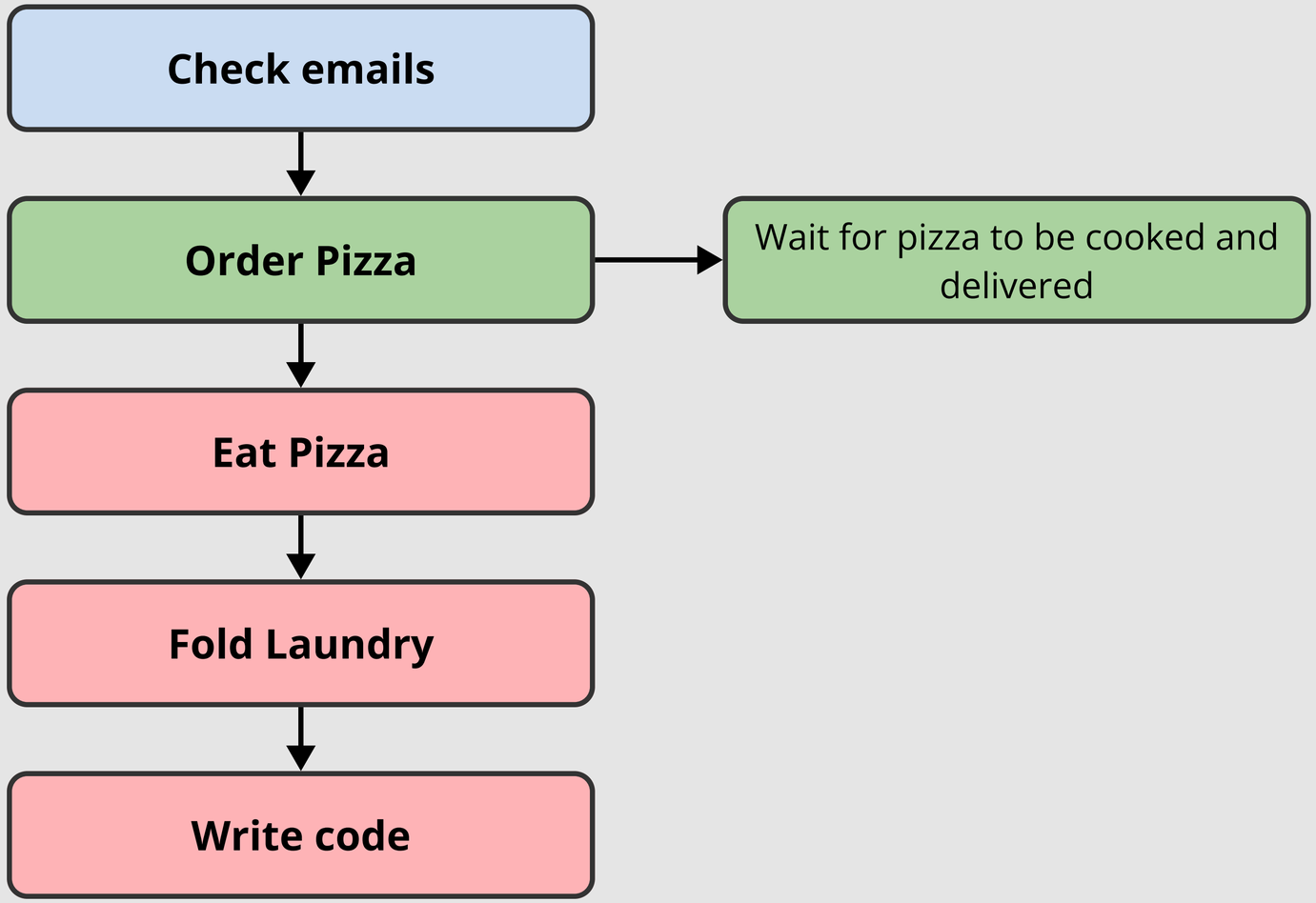
Async

Synchronous Code in JavaScript



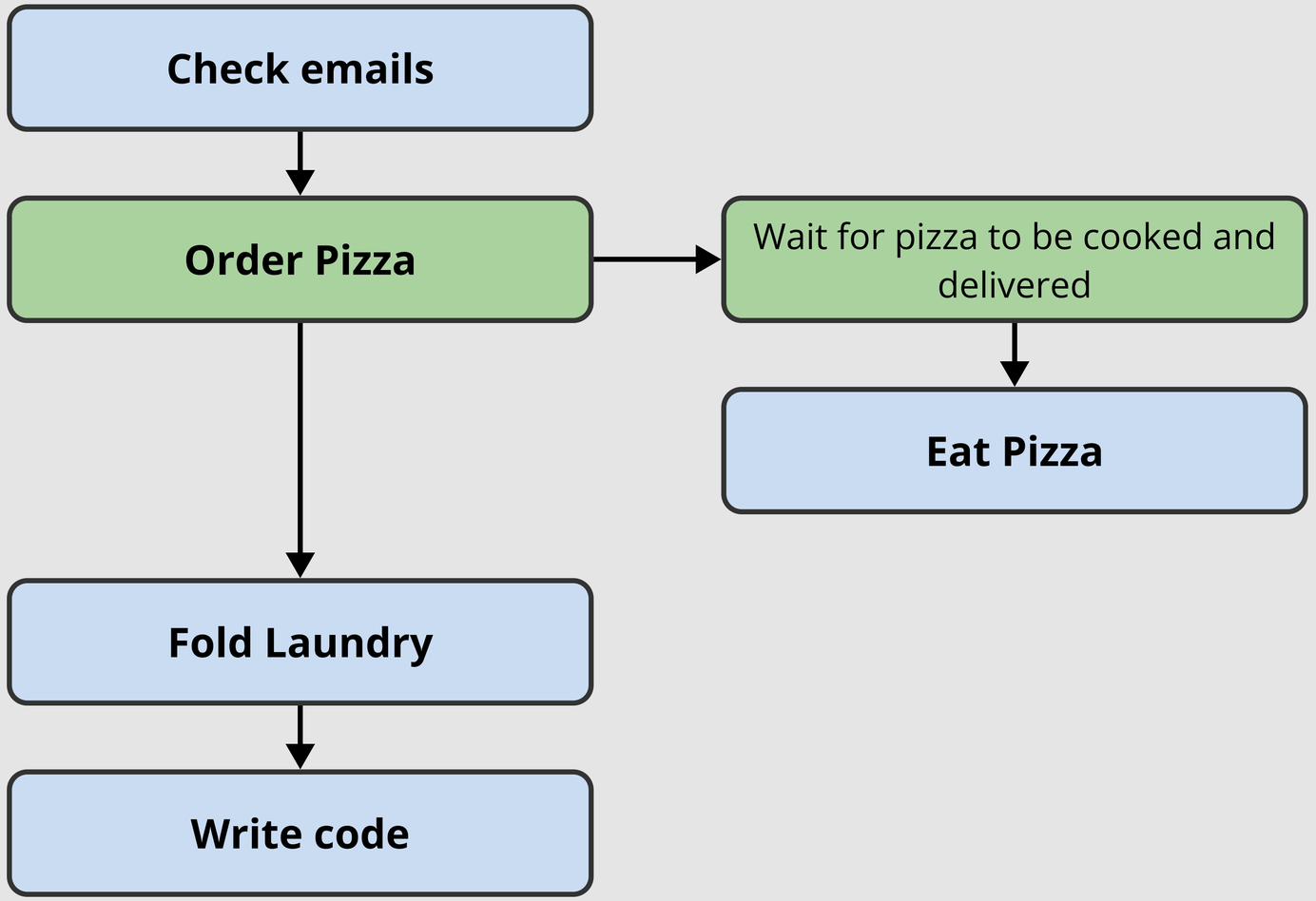
Async

Asynchronous Code in JavaScript



Async

Asynchronous Code in JavaScript



Async

Demo

Async

Ways to write Async code

- Callback functions

- Promises
  - dot-then
  - async/await

Async

Ways to write Async code

- Callback functions

- **Promises**
  - dot-then
  - async/await

Promises

What even is a promise?

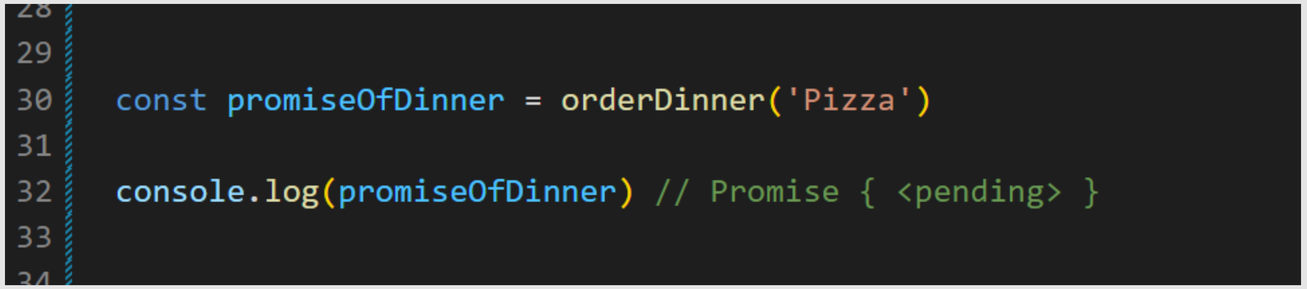
- A special kind of object that is *promising* to eventually do or get something

- A promise is a placeholder for a value that is not known at the point in time when the promise is created.

- A promise can be in one of three states:
  - Pending
  - Fulfilled
  - Rejected

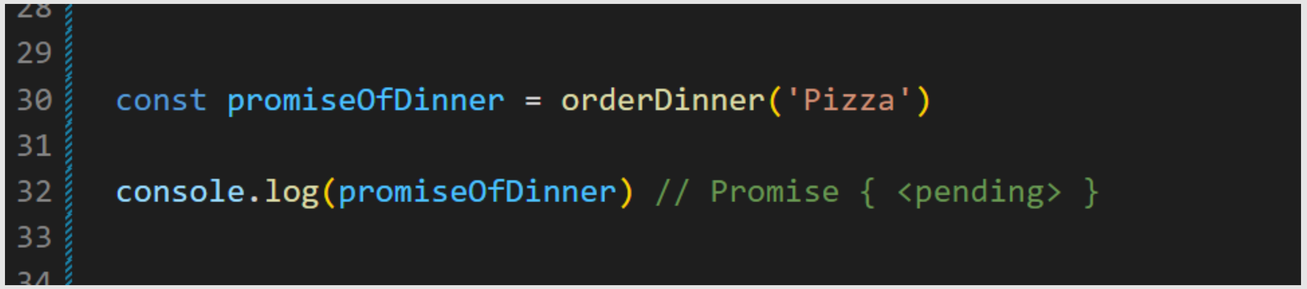
Promises

Example Code

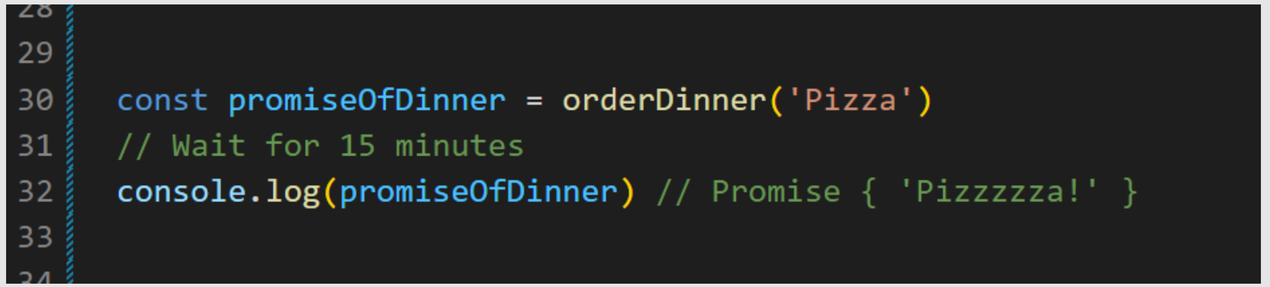


Promises

Example Code



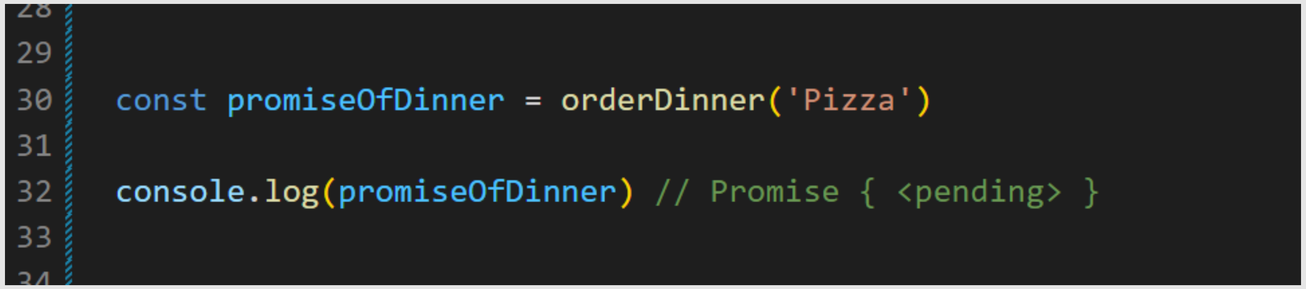
```
29
30  const promiseOfDinner = orderDinner('Pizza')
31
32  console.log(promiseOfDinner) // Promise { <pending> }
33
```



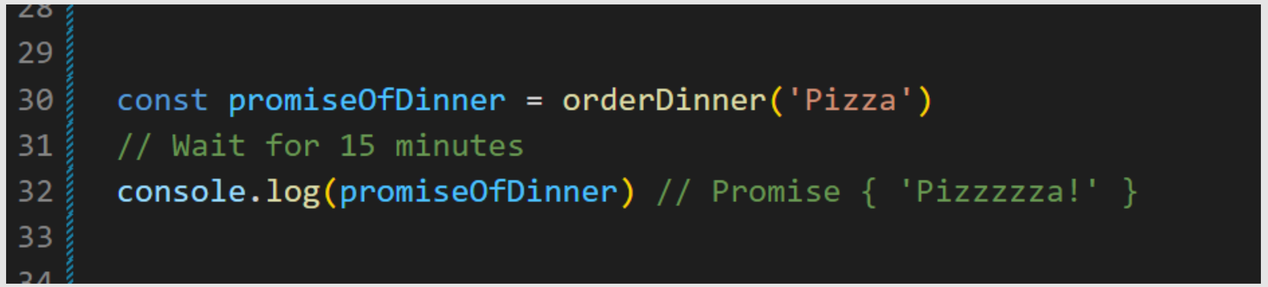
```
29
30  const promiseOfDinner = orderDinner('Pizza')
31  // Wait for 15 minutes
32  console.log(promiseOfDinner) // Promise { 'Pizzzzza!' }
33
```

Promises

Example Code



```
29
30  const promiseOfDinner = orderDinner('Pizza')
31
32  console.log(promiseOfDinner) // Promise { <pending> }
33
```



```
29
30  const promiseOfDinner = orderDinner('Pizza')
31  // Wait for 15 minutes
32  console.log(promiseOfDinner) // Promise { 'Pizzzzza!' }
33
```

```
29
30  const promiseOfDinner = orderDinner('Pizza')
31  // Wait for 15 minutes
32  console.log(promiseOfDinner) // Promise { <rejected> 'No pizza!' }
33
```

Promises

Demo

File System

Memory vs Storage

Memory

- Fast (to access and modify)
- Small (GB)
- Expensive ($ per GB)
- Temporary (gone without Power)

Storage

- Slower (to access and modify)
- Large (GB)
- Cheap ($ per GB)
- Persistence (stays without power)

File System

What is the FileSystem

- fs is a library that comes built-in to node
- How to read and write files, files to your computer using Node
- Some of the functions are async
- NOT a good storage solution for web applications
  - Doesn’t scale well
  - No defined structure (no data validation)
  - Slow
  - We will cover databases later

Takeaways

Pizza

- Async allows us to multi task (start now and finish later)
- Promises will at some point in the future do or get something
- Code that we want to happen after a Promise resolves should be inside a `try` and after the `await`
- Code that we want to happen after a Promise is rejected needs to be inside the `catch`
- Results from a promise are kept inside the promise (not returned)