Before we begin...

- Open up these slides:
  - https://bit.ly/2KoBAMY

ES2015, AJAX & APIs

Learning Objectives

- **Review** JavaScript versioning, scoping and hoisting
- **Identify** the difference between block and function scoping
- **Understand** destructuring and arrow functions
- **Understand** AJAX and the benefits it provides
- **Learn** to use the Fetch API
- **Request** data from APIs and **use** it to create HTML

Agenda

- Prettier
- ES2015
- APIs
- AJAX
- Fetch
- *P5.js*

A quick review

- APIs
- AJAX
- Fetch

Let's install Prettier

- `npm install -g prettier`
- Open up this GitHub Repository
- Install "**Prettier VSCode**"
- Open up your preferences - **CMND + ,**
- Add "editor.formatOnSave": true
  - Make sure that if you need a comma, you add one!

ES2015

For tonight...

- Let and Const
- Destructuring
- Arrow Functions

let and *const*

But wait...

What's wrong with *var*?

- It can be reassigned
- It allows shadowing (can be redeclared)
- It is function scoped (not necessarily a problem)

let and *const*?

- A new way of declaring variables
- They change the style of *scoping*...
  - From *function scoped* to *block scoped*
- They protect us from *shadowing*
- Enter: *TEMPORAL DEAD ZONE*

let

```
let currentScore = 1000;
```

- Block-scoped
- Can be re-assigned
- Named with lowerCamelCase
- Temporal Dead Zone!

const

```
const FAV_NUMBER = 42;
```

- Block-scoped
- It has an immutable binding
  - No reassignment, no redeclaration
- Named with UPPER_SNAKE_CASE
- Temporal Dead Zone!

What to use?

- **const** - By default
- **let** - If you need to redefine
- **var** - Almost never

Destructuring

Destructuring

- A way of easily extracting and saving pieces of nested data
- Syntactic sugar
- What can it be used on?
  - **Arrays** and **Objects**
- Where can it be used?
  - Parameters
  - Variable Assignment (with var, let and const)

Array Destructuring

```
const details = [ 'Groucho', 'Marx', 'Duck Soup' ];
const [ first, last, bestMovie ] = details;

function printUser( [username, email] ) {
    console.log(username, email);
}

printUser( ["kookslams", "kookslams@gmail.com"] );
```

Object Destructuring

```
const explorer = {
    first: "Jacques",
    last: "Cousteau",
};

const { first, last } = explorer;
// Or...
const { first: firstName, last: lastName } = explorer;

function calculateArea({ width, height }) {
    console.log(width * height);
}

calculateArea({ width: 20, height: 40 });
```

Arrow Functions

Arrow Functions

- A more concise approach to creating functions
- They can have **implicit return**
- Trade-offs:
  - Doesn't have it's own **this**
  - Doesn't have it's own **arguments** (though that isn't a big deal)

Arrow Functions

```
const sayHi = () => {
    console.log("Hello!");
};

const printValue = val => {
    console.log(val);
};

const addNumbers = (x, y) => {
    console.log(x + y);
};
```

Arrow Functions

```
const addNumbers = (x, y) => x + y;

const nums = [1, 2, 3, 4, 5];

nums.map(num => num * 5);
// => [5, 10, 15, 20, 25]

nums.reduce((sum, num) => sum + num, 0);
// => 15
```

When not to use them

- When you need *this* to be reassigned
  - Almost never use them with event listeners!
- When you need *arguments*
- When you want your function to have a name
  - They are harder to debug
- Plus more...

Next...

Other features we will see

- Default Parameters
- Classes
- Spread and Rest Operators
- Enhanced Object Literals
- Plus, more...
  - Check these out - they'll make your life easier!

Resources

- CSS Tricks: Let's Learn ES2015
- Babel: Learn ES2015
- Exploring ES6: Axel Rauschmayer
- Luke Hoban: ES6 Features
- CapitalOne: My Favourite Features

Fetch

OpenWeatherMap API

1. Go to the OpenWeatherMap APi website
2. Sign up for an API key here
3. Fill in your details
4. Log in
5. Go to the API Key Tab on your settings page
6. Copy the API Key
7. It'll take ten minutes for the API Key to work

OpenWeatherMap API

```
var baseURL = "http://api.openweathermap.org/data/2.5/weather";
var parameters = "?q=Sydney&units=metric&appid=API_KEY";

fetch(baseURL + parameters)
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
```

Yandex Translate API

- Open up the documentation here

Some other things...

- Using Geolocation
  - getCurrentPosition
- Speech to Text, Text to Speech
  - SpeechRecognition
  - SpeechSynthesis

Resources

- MDN: Using Fetch
- CSS Tricks: Using Fetch
- Scotch.io: Fetch
- David Walsh: Fetch
- Google Developers: Fetch
- Google Developers: Working with the Fetch API
- MDN: Fetch API

Homework

- Create a News Reader, using these APIs
  - Mashable, Reddit, Digg, NYT, and The Guardian
- Turn the Speech thing from tonight into an assistant
- Read up on ES2015
  - Translate some of your previous code into it!
- Finish all exercises from class
- Upload your homework to GitHub
- Prepare for next lesson

Homework (Extra)

- Go through The Modern JavaScript Tutorial
- Read Eloquent JavaScript
- Read Speaking JavaScript
- Go through some tasks in Exercism

What's next?

- Modules
- More ES2015
- Webpack
- Transpilation
- ...

Questions?

Thanks!