`this` and => functions, asynchrony, React Intro

Lecture 5

Let's fix a `this` Problem

- Change your terminal directory to 426demos/05-this-continued
- Open up this workspace in VSCode (the dot is "current directory"): `code .`
- Start a parcel server for this project: `parcel src/index.html --open`
- There are two classes: Point and Path. A Path has a name and a Point[].
- You and your partner's goal: replace the string "PATH_NAME" with the name of the Path (its public name property) in the method printPoints()

A Brief History of the Arrow

We’ll work through it in 06-arrow-functions.

The Trouble with `this` in Higher-order Functions

- In a higher-order function like forEach, map, addEventListener, or any asynchronous function with a callback or Promise is…
- By default, `this` will be bound to whatever the higher-order function wants
  - Often unhelpful, ambiguous, must read documentation
- Usually you want `this` to be bound to its meaning *outside* of the definition
  - Because that's how binding works for *every other variable name lookup*!
  - Arrow functions fix this by *not* allowing `this` to be bound inside the function

The tldr* of Arrow Functions

- An arrow function, whose definition is like:

```
(params) => { /* body */ }
```

- Has essentially the same semantics as:

```
(function(params) { /* body */ }).bind(this)
```

- Two key differences between arrow functions and bind, though:

1. In bind you are defining a function and then the bind method generates *another* function. It's more expensive in time and space.

2. The truth of an arrow function is *not that* **this** is tightly bound inside the arrow function, it's that it's not bound at all, no matter how you call it! Thus it always resolves to what **this** means in the scope it is defined in.

Steps to add React to a Project

- In your HTML, add an empty (typically div) element with an id attribute
- Rename your .ts file to .tsx and update your HTML's script src
- Install React libraries and TypeScript bindings:
  - npm install --save-dev react @types/react react-dim @types/react-dom
- In your tsx file, import the following two React library files:
  - import * as React from "react";
  - import ReactDom from "react-dom";

React Photo Gallery

- Let's work through the example in 426demos to build a React Gallery and Thumbnail components.