React Native 2

CS571: Building User Interfaces

Cole Nelson

What will we learn today?

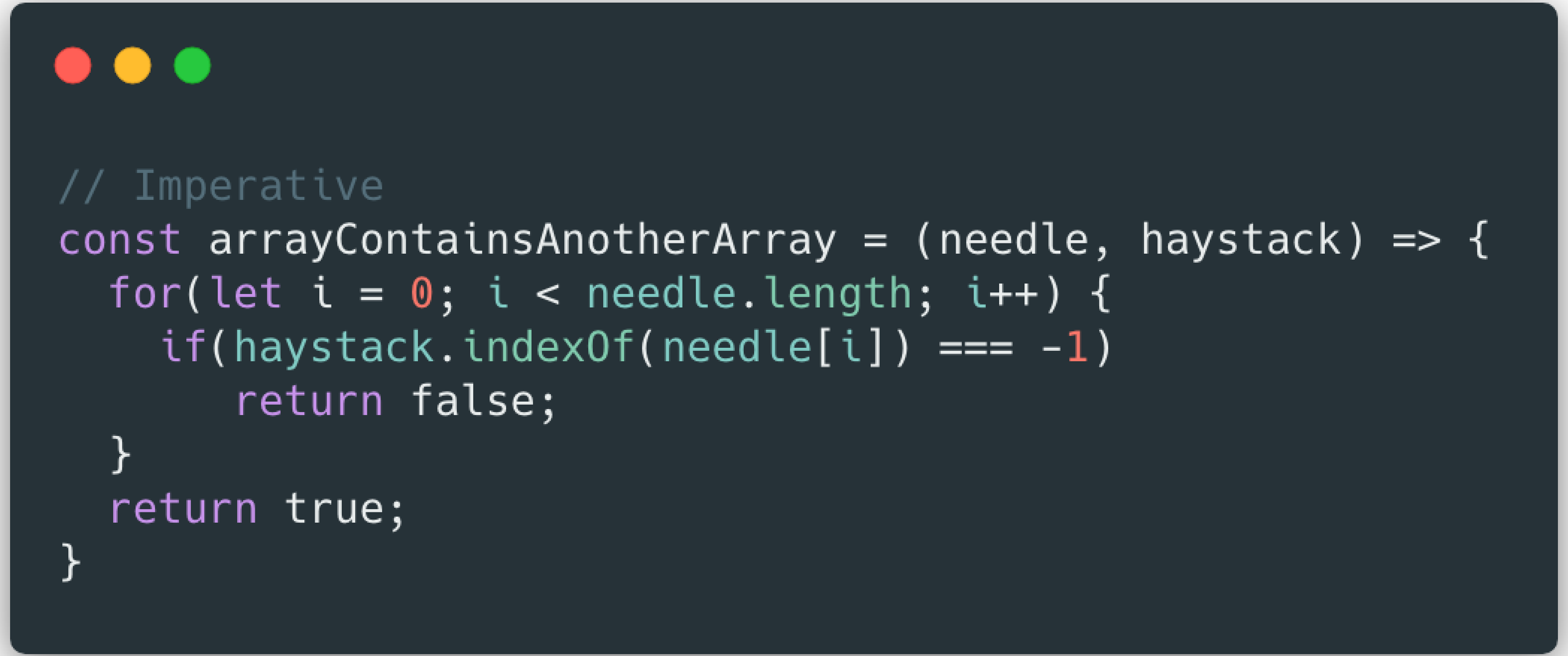
- Imperative vs Declarative Programming
- Review of Mobile App Development
- Navigation in React Native
- A Note on Expo

An Intervention

Our code is getting !pretty messy...

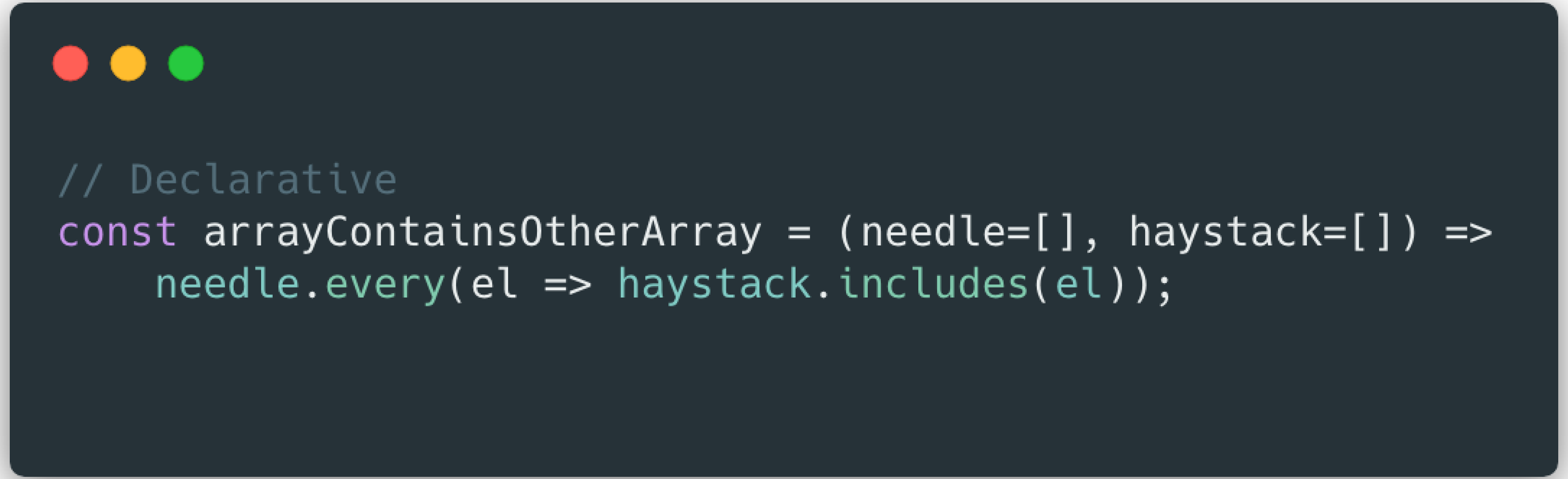
What can we do?

Big Ball of Mud



```
// Imperative
const arrayContainsAnotherArray = (needle, haystack) => {
  for(let i = 0; i < needle.length; i++) {
    if(haystack.indexOf(needle[i]) === -1)
       return false;
  }
  return true;
}
```

Image Source



```
// Declarative
const arrayContainsOtherArray = (needle=[], haystack=[]) =>
    needle.every(el => haystack.includes(el));
```

Image Source

Nifty JS Array Functions!

`slice`, `concat`, `filter`, `some`, `every`, and `reduce`

`slice` and `concat`

- `slice` returns a shallow copy with an optional beginning (inclusive) and ending (exclusive) index.
- `concat` joins two arrays together.

`filter`, `some`, and `every`

Performs a callback function over each element.

- `filter`: returns items where the callback function returns `true`
- `some`: returns `true` if atleast one of the callback functions returns `true`
- `every`: returns `true` if every one of the callback functions returns `true`

`reduce`

Constructs an object, array, or value. An initial value is provided and updated on each iteration via a callback with (prev, curr) parameters.

```
const array1 = [1, 2, 3, 4];
// 0 + 1 + 2 + 3 + 4
const initialValue = 0;
const sumWithInitial = array1.reduce(
  (prev, curr) => prev + curr, initialValue
);
console.log(sumWithInitial); // 10
```

Source: MDN Docs

`reduce` Challenges

Construct an array of strings of the amount in dollars rounded to the nearest cent.

```
const amounts = [1.928182, 29.10192, 3, 8.4, 0.12]
```

Construct an object where the key is the name of the word and the value is its number of letters.

```
const words = ["react", "native", "is", "awesome"]
```

Mobile Development

Native Development and its Alternatives

What is "Native" Development?

Building specifically for the device (e.g. Android or iOS) that you want to support.

iOS: Objective-C or Swift w/ Cocoapods
Android: Java or Kotlin w/ Maven or Gradle

Alternatives to Native Development

No mobile app! Do we really need an app? Could a responsive webpage be just as effective?

WebView! Can we take our existing code and just slap it into a WebView? e.g. Apache Cordova

Cross-Platform! Can we use a library or framework that will make our code work natively on Android *and* iOS? e.g. React Native

HW6: Badger Bakery 🦡🍩

Did it feel like we were making a mobile app?

HW7: Badger News 🦡📰

w/ React Navigation

Navigation in React Native

A more mobile-centric library.

React Navigation Alternatives

React Native is a framework* but still lacks support for things like navigation.

- React Navigation new!
- React Router on Week 4!
- `return isHome ? <HomeScreen> : <SettingsScreen>`
- Other outdated libraries...

React Navigation Installation

Just a few dependencies...

```
npm install @react-navigation/native react-native-screens react-native-paper
react-native-safe-area-context react-native-gesture-handler
react-native-reanimated  @react-navigation/native-stack
@react-navigation/drawer @react-navigation/bottom-tabs
```

Beware of your auto-imports!

React Navigation

We will use...

- Tab Navigation: `@react-navigation/bottom-tabs`
- Drawer Navigation: `@react-navigation/drawer`
- Stack Navigation: `@react-navigation/native-stack`

...others exist!

Navigation Basics

- Must be nested inside of a `NavigationContainer`
- Create navigators via a function `createNAVIGATOR()` e.g. `createBottomTabNavigator()`
- Navigators consist of a *navigator* and a set of *screens*

```
<NavigationContainer>
 <SomeNav.Navigator>
   <SomeNav.Screen name="Bookstore" component={BookstoreScreen}/>
   <SomeNav.Screen name="Book" component={BookScreen}/>
 </SomeNav.Navigator>
</NavigationContainer>
```

Navigation Basics

- `useNavigation` is a custom React hook that can be used to help us navigate
  - Supports `navigate`, `reset`, `goBack` among others
- Information can be passed from screen to screen via *route params* (see Native Stack Navigator example)
- Navigators can be styled
- Navigators can be nested

Tab Navigation

```
const SocialTabs = createBottomTabNavigator();

<NavigationContainer>
  <SocialTabs.Navigator>
    <SocialTabs.Screen name="NewsFeed" component={NewsFeedScreen}/>
    <SocialTabs.Screen name="Notifications" component={NotificationScreen}/>
    <SocialTabs.Screen name="AboutMe" component={AboutMeScreen} />
  </SocialTabs.Navigator>
</NavigationContainer>
```

Expo Snack Solution

Drawer Navigation

```
const SocialDrawer = createBottomTabNavigator();

<NavigationContainer>
  <SocialDrawer.Navigator>
    <SocialDrawer.Screen name="NewsFeed" component={NewsFeedScreen}/>
    <SocialDrawer.Screen name="Notifications" component={NotificationScreen}/
    <SocialDrawer.Screen name="AboutMe" component={AboutMeScreen} />
  </SocialDrawer.Navigator>
</NavigationContainer>
```

Expo Snack Solution

Stack Navigation

```
const BookStack = createNativeStackNavigator();

<NavigationContainer>
  <BookStack.Navigator>
    <BookStack.Screen name="Bookstore" component={BookstoreScreen}/>
    <BookStack.Screen name="Book" component={BookScreen}/>
  </BookStack.Navigator>
</NavigationContainer>
```

Expo Snack Solution

Stack Navigation

Can push a screen onto the history stack via `navigation.push(screenName, params)`

- `screenName` is the name of the screen to navigate to, e.g. `Book`
- `params` is an optional object of parameters to pass to the receiving screen.
- `params` is recieved as `props.route.params`

In-Class Example

Pass and receive params while navigating.

Nested Navigation

- Navigators can be nested.
  - Stack in Tabs (e.g. HW7)
  - Stack in Drawer
  - Stack in Tabs in Drawer (e.g. Example Below)
  - Stack in Stack in Tabs
  - Stack in Stack in Stack in Stack in Stack
- Make use of the `headerShown` option!

Expo Snack Solution

A Note on Expo

Expo is a library for quickly getting started with React Native projects. No need to...

- cocoa pods install ❌
- maven/gradle building ❌
- react native linking ❌

You may need to use specific expo libraries, such as @expo/vector-icons

What did we learn today?

- Imperative vs Declarative Programming
- Review of Mobile App Development
- Navigation in React Native
- A Note on Expo

On to Design Patterns! 🚀