**COMMON REACT AND NATIVE COMPONENTS**

<https://youtu.be/9Nt1tXV3y0c>

When writing HTML, we're used to using <div> and <span> tags to define sections or to contain other elements on the page. In React Native, a similar principle applies, but this time we're using React Native's <View> component to build the application UI. Just like HTML's <div>, <View>components can accommodate several props (e.g. style), and can even be nested inside other <View> components!

<Text> works just how you'd expect, as well. Its main objective is to, by no surprise, render text in the application. Just like <View>, styling and nesting capabilities apply to <Text> components, as well.

Let's see how they work!

<https://youtu.be/_Qv4NGKNuug>

**In-Class Project overview**

<https://youtu.be/HZSi_XB3drA>

**Icons**

<https://youtu.be/tYb0-l81x4U>

Right out of the box, **Create React Native App** offers support for thousands of vector icons to use in your applications. Feel free to bookmark and check out Expo's [vector icon directory](https://expo.github.io/vector-icons) for a complete list. Whichever icon set you choose, just be sure that it fits the overall look and feel of your application (e.g., using platform-specific icons).

<https://youtu.be/sH4D8b7MotQ>

**Review**

**<View>**

When writing HTML, we're used to using <div> and <span> tags to define sections or to contain other elements on the page. In React Native, a similar principle applies, but this time we're using React Native's <View> component to build the application UI. Just like HTML's <div>, <View>components can accommodate several props (e.g. style), and can even be nested inside other <View> components!

**<Text>**

<Text> works just how you'd expect, as well. Its main objective is to, by no surprise, render text in the application. Just like <View>, styling and nesting capabilities apply to <Text> components, as well.