Single Value Components

tl;dr

- Components in React Native are not HTML but it might help to relate them to some HTML elements
- Text can't be bare. It must be in a <Text>
- <TextInput> is for editing text. The onChangeText event returns the text in the box
- <Images> are usually bundled with the executable.
- But when they are downloaded live, we must reserve space for them with a width and height or they won't show at all.

- There are about 250 HTML tags and exactly 0 can be used in React Native.
- There are about 40 RN components and exactly 0 can be used in a browser.
- So even though React Native looks like HTML and is structured like the HTML-based React, the components are completely different.
- It's important that we learn them.

Categories of RN components

Category	Some components
Layout	Modal, View, SafeAreaView, ScrollView, RefreshControl, KeyboardAvoidingView, StatusBar, WebView
Single-value	Text, TextInput, Slider, Switch, Image
List	Picker, FlatList, SectionList
Touchable	Button, TouchableHighlight, TouchableNativeFeedback, TouchableOpacity, TouchableWithoutFeedback
Others	ActivityIndicator, Platform-specific components

<Text>

<Text>

- Kind of like a <p>
- But only text can appear in it.
- It is permissible to nest another <Text> inside of it for styling purposes.

Text must be wrapped

From the React Native documentation:

> In React Native, we are more strict about it: you must wrap all the text nodes inside of a <Text> component. You cannot have a text node directly under a <View>.

RN will throw if there is any text not wrapped in a <Text> element

Some Text Props

Name	Notes
numberOfLines	Truncate text after this many lines instead of continuing to wrap and grow.
ellipsizeMode	When truncating, • head - put the ellipsis at the beginning and cut off the start • tail - put the ellipsis at the end and cut off the end • middle - put the ellipsis in the middle • clip - Don't show any ellipses, just clip the end off
selectable	If false, user can't select text, like for copying for example
selectionColor	What the text looks like while being selected
adjustFontSizeToFit	A bool. If true, shrink the fontSize until all the text fits in the container. (iOS only)

Some Text Events

Name	Notes
onPress	A tap or click
onLongPress	Usually takes the place of a right-click
onLayout	Fires when the scene is being laid out

<TextInput>

A component for keyboard entry

Kind of like all these:

```
<input type='text' />
<textarea />
<input type="number" />
```

But combined into one control

Some TextInput Props

Name	Notes
autoCorrect	Yep, it'll turn on the sometime embarrassing feature
dataDetectorTypes	Can make the text clickable while the user is typing. If it detects one of 'phoneNumber', 'link', 'address', 'calendarEvent', or 'all' (iOS)
keyboardType	number-pad, email-address, phone-pad
multiline	bool. If false, it's a single line. Some other properties are ignored if it is single-line. Like borderLeft and borderRight for instance.
numberOfLines	Used with multiline=true (Android)
placeholder	Just like in the web. Ghost text
secureTextEntry	bool. True makes it like a password box
value	What is in the textbox

Some TextInput Events

Name	Notes
onBlur/onFocus	Just like their web counterparts
onChange	Sends the event object (See example later)
onChangeText	Sends the text (See example later)
onKeyPress	note: no onKeyUp, onKeyDown, etc.

The event object when needed:

ModifyPerson.js

```
export class ModifyPerson extends Component {
 ...
 render() {
  <TextInput value={this.state.first}
    onChange={this.handleChange} />
 }
 handleChange(e) {
this.setState(
   {first: e.nativeEvent.value});
 }
}
```

onChangeText sends the text itself

ModifyPerson.js

```
export class ModifyPerson extends Component {
 ...
 render() {
  <TextInput value={this.state.first}
    onChangeText={this.handleChange} />
 }
 handleChange(text) {
  this.setState({first: text});
 }
}
```

<Image>

HTML <img> != RN <Image>

HTML <img> tag	<Image> component
• For showing an image • Easily scaled • A separate download • Unknown size • Only one virtual machine -- the browser	• For showing an image • Easily scaled • Part of your bundle* • Known size • Many VMs - iPhone X, 7, 6s, etc. Nexus 6, 5, etc. GS4, 5.. *usually

Source

- In the web we ask the server for the image so it must live in a public, static, web server folder.
- In RN, we don't ask for things from the web except for Ajax data requests
- So we place our image files alongside our source

```
.
├── button.js
└── img
    ├── check@2x.png
    └── check@3x.png
```

For local images

- These are compiled into the install package
- For local images:

```
<Image src={require('./assets/foo.jpg')} />
```

For remote images

- A simple request

```
<Image source={uri: 'http://foo.com/img.jpg'} />
```

- A more complex request

```
<Image source={uri: 'https://foo.com/img.jpg',
 method='POST',
 headers: { 'accepts-type': 'png, jpg, gif' },
 body: { key1: val1, key2: val2 ...}
} />
```

- Flow of the page layout happens (kind of) like this...
- Component is compiled
- Scene is laid out on the page
- The layout engine makes room for the image, which at this point is 0 x 0.
- The image is downloaded to the device and scaled to fit in its allocated space (0x0).
- Result: You see no image.

Remote images must be sized

To solve the problems, give your image a height and/or width in it style

Briefly, how to size your Image

```
<Image source={{uri: "http://imgur.com/i.jpg"}}
  style={{
    height:100, width: 100,
    resizeMode: "cover",   // The default
 }} />
```

Much more on styling later in the course

Sizing images: resizeMode

- resizeMode: stretch | cover | contain | repeat | center
  - stretch: Lose aspect ratio. Grow both x and y to fill. Nothing is cut off and no blank space top/bottom or left/right
  - cover: Maintain aspect ration. Grow until x or y covers all blank space. top/bottom or left/right will be cut off. No blank space.
  - contain: Maintain aspect ratio. Grow until x or y fits. Nothing is cut off but there is blank space top/bottom or left/right.
  - repeat: duh
  - center: same as contain.

tl;dr

- Components in React Native are not HTML but it might help to relate them to some HTML elements
- Text can't be bare. It must be in a <Text>
- <TextInput> is for editing text. The onChangeText event returns the text in the box
- <Images> are usually bundled with the executable.
- But when they are downloaded live, we must reserve space for them with a width and height or they won't show at all.