Cheatsheets / **Learn React Native**

Core Components

Core Components

Core components are ready-to-use components available from React Native, which include `<View>`, `<Text>`, `<Image>`, `<ScrollView>`, `<Button>`, and `<TextInput>`.

```
<ScrollView>
  <Text>Some text</Text>
  <View>
    <Text>Some more text</Text>
    <Image
      source={{
        uri:
'https://picsum.photos/64/64',
      }}
    />
  </View>
  <TextInput
    defaultValue="You can type here"
  />
  <Button
    onPress={() => {
      alert('You tapped the button!');
    }}
    title="Press Me"
  />
</ScrollView>
```

Importing Core Components

You can import core components into your Expo project from the `react-native` package.

```
import { View, Text } from 'react-
native';
```

Core Components Optional Functionality

Core components may have more optional functionality, which you can configure through the props.

```
function App() {
  return (
    {/* Text components can be with
`numberofLines` prop */}
    <Text numberOfLines={1}>
      quam elementum pulvinar etiam non
quam lacus suspendisse faucibus interdum
posuere lorem ipsum dolor sit
    </Text>
  );
}
```

`<View>` Component

The `<View>` component is a generic "visible" container without any semantic meaning or noticeable performance impact, best translated as `<div>` from web.

```
function App() {
  return (
    {/* Base layout structure */}
    <View style={{ flex: 1 }}>
      {/* Simple background color */}
      <View style={{ padding: 8, color:
'red' }}>
        <Text>Text with background
color</Text>
      </View>
      {/* Space layout structure */}
      <View style={{ margin: 16 }} />
    </View>
  );
}
```

<ScrollView> Component

The `<ScrollView>` component is a generic "visible" container with scrolling, but it's less performant than `<View>`, making it less suitable for simple styling and short lines of text.

```
function App() {
  return (
    <ScrollView>
      <Text style={{ margin: 16 }}>Scroll here to see more!</Text>
      <View style={{ marginTop: 1024 }} />
      <Text style={{ margin: 16 }}>Made you look!</Text>
    </ScrollView>
  );
}
```

<Text> Component

The `<Text>` component is the only way to display text in React Native. These components can be nested to inherit and modify styling.

```
<Text style={{ height: 40, borderWidth: 1 }}>
  Here's some text!
</Text>
```

<Image> Component

The `<Image>` component is an optimized way to render images from various sources, including remote HTTP access, local assets imported with `require`, and base64 encoded strings.

```
<Image source={require('./local/asset.jpg')} />

<Image source={{ uri: 'https://docs.expo.io/static/images/header/sdk.svg' }} />

<Image source={{ uri: 'data:image/png;base64,<base64-string>=' }} />
```

`<TextInput>` Component

The `<TextInput>` component can capture alphanumeric input from the user. Its behavior can be modified with the `onChangeText` prop, which accepts a function.

```
const [input, setInput] = useState('');

// example use of input
console.log(input);

return (
  <TextInput
    placeholder="What is your name?"
    onChangeText={setInput}
  />
);
```

Business-Logic Focused Components

Creating business-logic focused components is done by “composing” components into one and connecting functionality by passing props and event handlers.

```
function ReadMoreParagraph(props) {
  const [isOpen, setOpen] =
useState(false);

  return (
    <View style={{ flex: 1,
flexDirection: 'column' }}>
      <Text style={{ fontSize: 16 }}
numberOfLines={!isOpen ? 2 : undefined}>
        {props.children}
      </Text>
      {!isOpen
        ? <Button title='Read More'
onPress={() => setOpen(true)} />
        : <Button title='Read Less'
onPress={() => setOpen(false)} />
    </View>
  )
}
```

Custom Components

Similar to React, it's common to create custom components to wrap configuration and usage of core components.

```
const App = () => (
  <View style={{ flex: 1, justifyContent:
'center' }}>
    <Box color="red" />
    <Box color="green" />
    <Box color="blue" />
  </View>
);

export const Box = (props) => (
  <View style={{ width: 100, height: 100,
backgroundColor: props.color }} />
);
```

Print Share