RN – Lists, CheckBoxes and Multiple Screens

# Multple Screens Using React-navigation

## Entry program – App.js

* Use Expo’s default multi screen installation code as a good example initialization
* App.js
  + Imports the AppNavigator class at the top

import AppNavigator from './navigation/AppNavigator';

## Create the navigation containers – AppNavigator.js

* AppNavigator
  + Imports the createAppContain and createSwitchNavigator

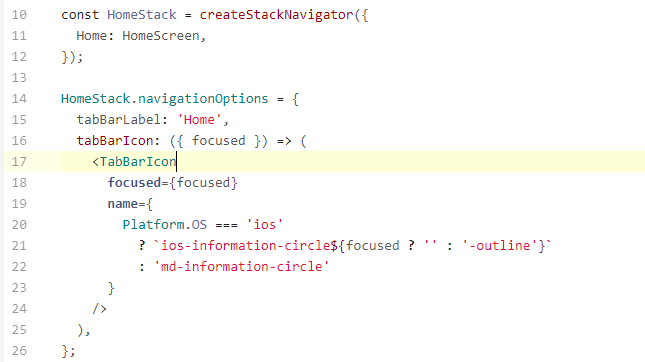


## Create the Navigation stacks– Maintabnavigator.js

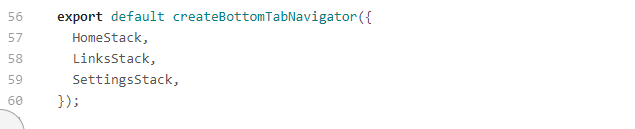
* MainTabNavigator
  + Import stackNavigator and bottomTabNavigator component
  + Import screens to be used



* Create a stack for each screen that needs navigation. Start with the HomeScreen. You can set the label and the icon for it.



* Put each navigation stack on the screen. You can put them at the bottom with the BottomTabNavigator



# Lists – display map data

There are several List components that come with react-native.

## FlatList

Flatlist takes in a prop

1. “data”which is a keyed data array javascript Map (like a Python dictionary) that has key:value pairs. The value can include multiple fields.
2. renderItem is a function that describes how to display each item. “\_renderItem” will be passed a prop called “item which is each item in the map.

### Flatlist component example

Given a map: data={[{title: 'Title Text', key: 'item1'}]}

<FlatList data={this.state.data} renderItem={this.\_renderItem} />

### \_renderItem example

renderItem={({item}) => (

<TouchableHighlight

<View style={{backgroundColor: 'white'}}>

<Text>{item.title}</Text>

</View>

</TouchableHighlight>

)}

# Maps – Arrays items with a key

## Create an array of items

### Array of items example

const initialList = [

{

author: "Dr.Seuss",

checked: true,

},

{

author: "Isaac Asimov",

checked: false,

},

];

## Add keys to the items to create a map

1. Create the map in the constructor using the arraylist declared (initial list in the example above)

class SimpleList extends Component {

constructor(props) {

super(props);

this.state = { data: this.\_addKeysToList(initialList) };

}

1. Write the addKeys function

\_addKeysToList = initList => {

// assigns a unique key based on the field listed

// in this case it is the author field

return initList.map(initList => {

return Object.assign(initList, { key: initList.author });

});

};

## Display the item with a checkbox

1. Install library to add to react native components. This is like twitter bootstrap.

### install react-native-elements

npm install --save react-**native**-elements

1. Now you can use all of the components from this library (go to <https://react-native-training.github.io/react-native-elements/> to check it out. We are going to use the checkbox. See an example it the renderItem function below.

### checkbox example

\_renderItem = ({item}) => {

/\*return <Text style={styles.row}>{products.item.key}</Text>;\*/

return(

<CheckBox

title={item.author}

onPress={() => {this.checkFunction(item);}}

checked={item.checked}

/>

);

};

1. Change the value in the map based when the user presses the check box. I have created a function called “checkFunction” and passed it the item as a prop. The checkFunction walks through each item of the map checking to see if it is the item that was passed in. If it was, it changes the value of checked to be the opposite of what it was and then displays it. If the box was checked it becomes unchecked, and if it was not checked, it becomes checked.
   1. This uses the Javascript map function which returns a new map. That is why we have to create a temp map and set the map in the state to the new one.

### Change the checkbox - onpress example

checkFunction (item) {

let tempData = this.state.data.map(i => {

if (item.author === i.author) {

i.checked = !i.checked;

}

return i;

});

this.setState({ data: tempData });

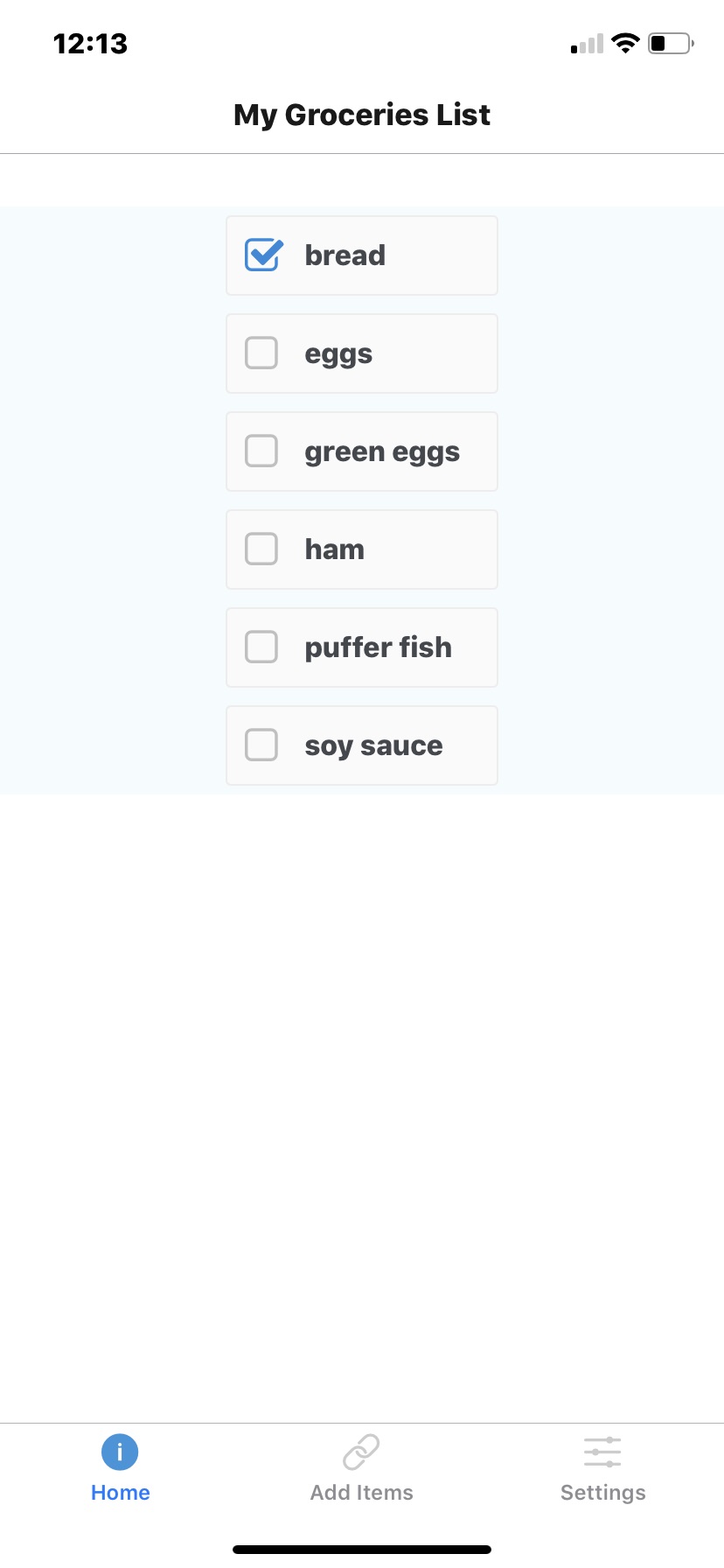
}

# Put it all together – a List of checkboxes with navigation to other screens



1. Run “npm init shopList” to create a multi-screen expo file structure. Make sure you select the navigation screen option. (arrow down to the second option)
2. Run npm install --save react-**native**-elements
   1. (so you can use the checkbox component
3. Download and copy into the
   1. Mainfolder:
      1. App.js
   2. Navigation folder:
      1. AppNavigator.js
      2. MainTabNavigator.js
   3. Screens folder:
      1. HomeScreen.js
      2. Note where the SimpleList javascript is called in the render

## Create Simplelist.js-create a groceries list with active checkboxes



1. Create your imports

**import React, { Component } from "react";**

**import { StyleSheet, Text, View, FlatList } from "react-native";**

**import { CheckBox } from 'react-native-elements';**

1. Create your “const” initial list (arrays of items example)
2. Declare your SimpleList class and your constructor (make sure you bind your checkfunction)
   1. See add keys to create a map for example
3. Write your checkFunction (see change the checkbox example)
4. Write your renderItem function (see checkbox example)
5. Write your \_addKeysToList function
6. Write your main render function
   1. Include a view
   2. Include the call to FlatList (see flatlist component example)
7. Create your stylesheet

**const styles = StyleSheet.create({**

**container: {**

**flex: 1,**

**justifyContent: "center",**

**alignItems: "center",**

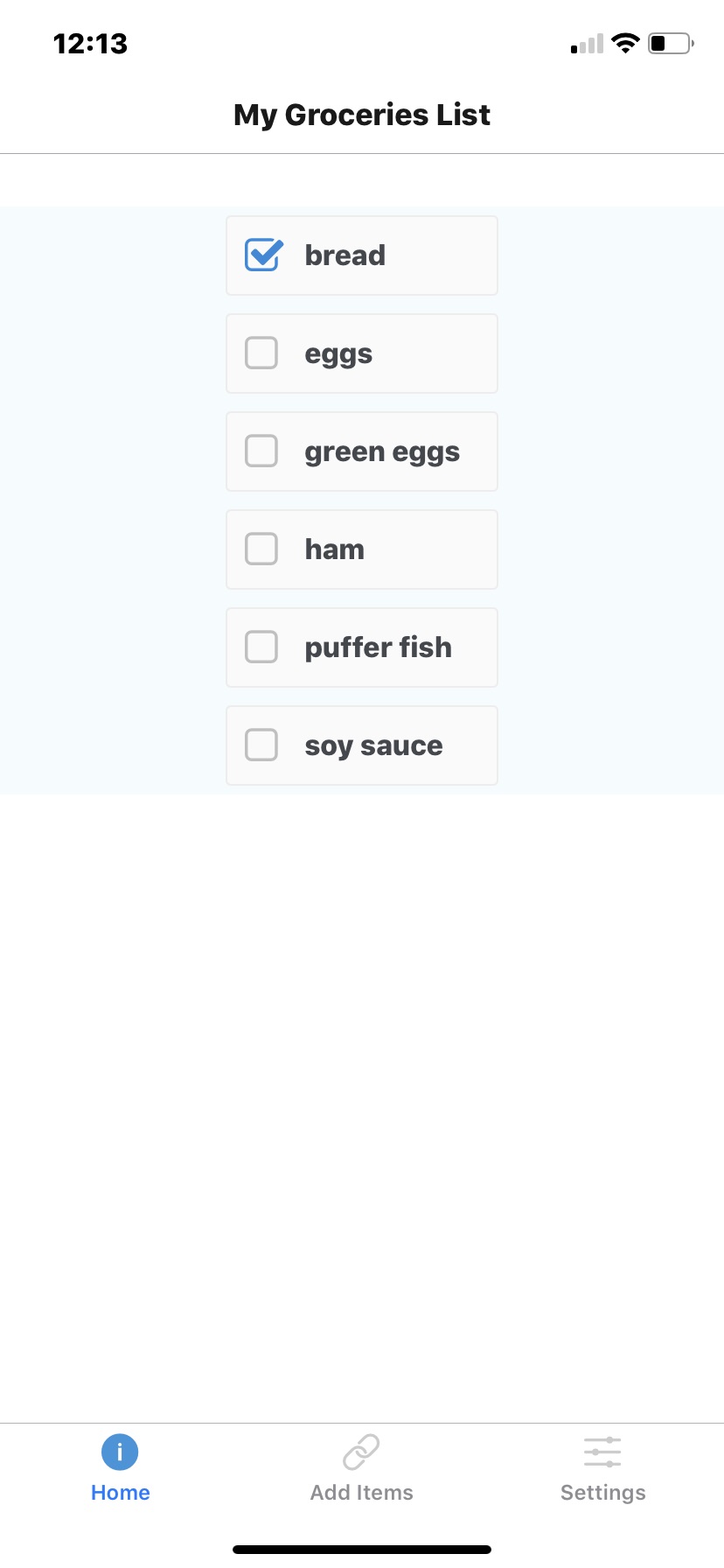
**backgroundColor: "#F5FCFF"**

**},**

**row: { fontSize: 24, padding: 42, borderWidth: 1, borderColor: "#DDDDDD" }**

**});**

1. Write the concluding export

**export default SimpleList; **