Swipe Down to Refresh List View Using Refresh Control

Swipe Down to Refresh

Here is an example of React Native Swipe Down to Refresh ListView Using Refresh Control. It was first introduced in Android Material Design and became very popular. Almost all Apps are using Swipe down to refresh. In React Native you can use this feature using RefreshControl provided by React Native.

To Import Refresh Control in Code

import { RefreshControl} from 'react-native'

Swipe Down to Refresh ListView Using Refresh Control

<FlatList

data={this.state.dataSource}

ItemSeparatorComponent={this.ListViewItemSeparator}

enableEmptySections={true}

keyExtractor={(item, index) => index.toString()}

renderItem={({item}) => (

<Text

style={styles.rowViewContainer}

onPress={() => alert(item.id)}>

{item.title}

</Text>

)}

refreshControl={

<RefreshControl

//refresh control used for the Pull to Refresh

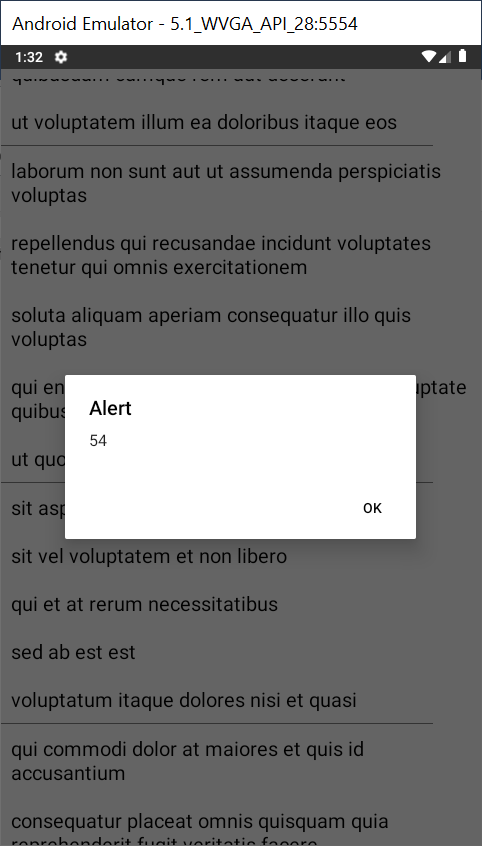
refreshing={this.state.refreshing}

onRefresh={this.onRefresh.bind(this)}

/>

}

/>



//This is an example code for React Native Swipe Down to Refresh ListView Using RefreshControl//

import React, { Component } from 'react';

//import react in our code.

import {

StyleSheet,

ActivityIndicator,

FlatList,

Text,

View,

Alert,

RefreshControl,

} from 'react-native';

//import all the components we are going to use.

export default class Project extends Component {

constructor(props) {

super(props);

//True to show the loader

this.state = { refreshing: true };

//Running the getData Service for the first time

this.GetData();

}

GetData = () => {

//Service to get the data from the server to render

return fetch('https://jsonplaceholder.typicode.com/posts')

.then(response => response.json())

.then(responseJson => {

this.setState({

refreshing: false,

//Setting the data source for the list to render

dataSource: responseJson

});

})

.catch(error => {

console.error(error);

});

};

ListViewItemSeparator = () => {

return (

//returning the listview item saparator view

<View

style={{

height: 0.2,

width: '90%',

backgroundColor: '#808080',

}}

/>

);

};

onRefresh() {

//Clear old data of the list

this.setState({ dataSource: [] });

//Call the Service to get the latest data

this.GetData();

}

render() {

if (this.state.refreshing) {

return (

//loading view while data is loading

<View style={{ flex: 1, paddingTop: 20 }}>

<ActivityIndicator />

</View>

);

}

return (

//Returning the ListView

<View style={styles.MainContainer}>

<FlatList

data={this.state.dataSource}

keyExtractor={(item, index) => index.toString()}

ItemSeparatorComponent={this.ListViewItemSeparator}

enableEmptySections={true}

renderItem={({item}) => (

<Text

style={styles.rowViewContainer}

onPress={() => alert(item.id)}>

{item.title}

</Text>

)}

refreshControl={

<RefreshControl

//refresh control used for the Pull to Refresh

refreshing={this.state.refreshing}

onRefresh={this.onRefresh.bind(this)}

/>

}

/>

</View>

);

}

}

const styles = StyleSheet.create({

MainContainer: {

justifyContent: 'center',

flex: 1,

marginTop: 10,

},

rowViewContainer: {

fontSize: 20,

padding: 10,

},

});