Cheatography

react-native Cheat Sheet

by aezzat (rathetsu) via cheatography.com/194747/cs/40667/

React vs React Native

Usage Scope	Syntax	Animation & Gestures	Routing
React.js is a JS library for building Responsive User Interfaces for Building Web Application. React 0Native is a framework for creating mobile applications with a native feel.	Both use JSX, but React uses HTML tags while React Native uses <view> <text> etc.	React uses CSS animations on a major scale to achieve animations for a web page while the recommended way for react native is to use the Animated API.	React uses react-router for routing and doesn't have any inbuilt routing capabilities but React Native has a built-in Navigator library.

useLocalStorage

```
import { useState, useEffect } from 'react';
import AsyncS torage from '@reac t-n ati ve- asy -
nc- sto rag e/a syn c-s tor age';
function useLoc alS tor age (key, initia lValue) {
    const [store dValue, setSto red Value] =
useSta te(() => {
       try {
            const item = AsyncS tor age.ge tIt -
em( key);
            return item ? JSON.p ars e(item) :
initia lValue;
       } catch (error) {
            con sol e.e rro r(e rror);
            return initia lValue;
       }
   });
    const setValue = async (value) => {
       try {
            const valueT oStore = value instanceof
Function ? value( sto red Value) : value;
            set Sto red Val ue( val ueT oSt ore);
            await AsyncS tor age.se tIt em(key,
JSON.s tri ngi fy( val ueT oSt ore));
       } catch (error) {
            con sol e.e rro r(e rror);
       }
   };
    return [store dValue, setValue];
}
// Usage
// const [username, setUse rname] = useLoc alS -
tor age ('u ser name', 'defau ltN ame');
```

useDeviceOrientation

useDeviceOrientation (cont)

```
>     return () => Dimensions.removeEventListener('change',
onChange);
  }, []);
  return orientation;
}
// Usage:
// const orientation = useDeviceOrientation();
```

User Input

```
import { useState, useEffect } from 'react';
import { Dimensions } from 'react -na tive';
function useDev ice Ori ent ation() {
    const [orien tation, setOri ent ation] =
useState(
        Dim ens ion s.g et( 'wi ndo w').width >
Dimens ion s.g et( 'wi ndo w').height
          ? 'LANDS CAPE'
          : 'PORTRAIT'
   );
    use Eff ect(() => {
        const onChange = ({ window }) => {
            set Ori ent ati on( win dow.width >
window.height ? 'LANDS CAPE' : 'PORTR AIT');
       };
        Dim ens ion s.a ddE ven tLi ste ner ('c -
hange', onChange);
```

```
import React, { useState } from 'react';
import { View, TextInput, Button, Text } from
'react -na tive';
import axios from 'axios';
import AsyncS torage from '@reac t-n ati ve- asy -
nc- sto rag e/a syn c-s tor age';
const Login = ({ navigation }) => {
    const [email, setEmail] = useSta te('');
    const [password, setPas sword] = useSta -
te('');
    const [error, setError] = useSta te('');
    const handle Login = async () => {
       try {
            const response = await axios.p os -
t(' YOU R_A PI_ END POI NT/ login', {
                email,
                pas sword,
           });

            const { token } = respon se.d ata;
           if (token) {
                await AsyncS tor age.se tIt -
em( 'au thT oken', token);
                nav iga tio n.n avi gat e(' Home');
           }
       } catch (err) {
            con sol e.e rro r("Login error: ",
err);
            set Err or( 'In valid login creden -
tia ls');
       }
   };
    return (
        <Vi ew>
            <Te xtInput
                pla ceh old er= " Ema il"
                val ue= {email}
                onC han geT ext ={s etE mail}
                aut oCa pit ali ze= " non e"
            />
```

User Input (cont)

```
>      <TextInput
       placeholder="Password"
       value={password}
       onChangeText={setPassword}
       secureTextEntry
      />
      <Button title="Login" onPress={handleLogin} />
      {error && <Text>Error: {error}</Text>}
    </View>
  );
};
export default Login;
```

Securely Store Auth Token

```
import * as Keychain from 'react-native-keychain';
// Storing the token securely
async function storeT oke n(t oken) {
    try {
          await Keycha in.s et Gen eri cPa ssw ord -
('u ser Token', token);
    } catch (error) {
          con sol e.e rro r(' Error storing token:',
error);
    }
}
// Retrieving the token
async function getToken() {
    try {
          const creden tials = await Keycha in.g -
et Gen eri cPa ssw ord();
        if (crede ntials) {
             return creden tia ls.p as sword;
        }
    } catch (error) {
          con sol e.e rro r(' Error retrieving
token:', error);
    }
     return null;
}
// Clearing the token
async function clearT oken() {
    try {
          await Keycha in.r es etG ene ric Pas swo -
rd();
    } catch (error) {
```

Securely Store Auth Token (cont)

```
>    console.error('Error clearing token:', error);
  }
}
```

Context API

```
import { createContext } from 'react';
const MyContext = create Con text();
export const MyProvider = ({ children }) => {
    const [value, setValue] = useSta te( 'in iti -
alV alue');
    return (
        <My Con tex t.P rovider value={{ value,
setValue }}>
            {ch ildren}
        </M yCo nte xt.P ro vid er>
    );
}
// Usage
import React, { useContext } from 'react';
const MyComp onent = () => {
    const { value, setValue } = useCon tex t(M -
yCo ntext);
    return (
        <di v>
            {value}
            <button onClic k={() => setVal -
ue( 'ne wVa lue ')} >Set Value< /bu tto n>
        </d iv>
    );
};
// Wrapping components in the provider
const App = () => (
    <My Pro vid er>
        <My Com ponent />
    </M yPr ovi der>
);
```

FlatList

```
const DATA = [
    { id: '1', title: 'Item 1' },
    { id: '2', title: 'Item 2' },
    // ... more items
];
const Item = ({ title }) => (
```

FlatList (cont)

```
>   <View style={styles.item}>
    <Text style={styles.title}>{title}</Text>
  </View>
);
// Pull to refresh
const [refreshing, setRefreshing] = React.useState(false);
const onRefresh = React.useCallback(() => {
  setRefreshing(true);
  // Fetch new data and set refreshing to false
  // For example:
  // fetchData().then(() => setRefreshing(false));
}, []);
const MyFlatListComponent = () => (
  <FlatList
    horizontal={false}
    refreshing={refreshing}
    onRefresh={onRefresh}
    data={DATA}
    renderItem={({{ item }) => <Item title={item.title} />}
    keyExtractor={item => item.id}
  />
);
```

Stack Navigation

```
import { NavigationContainer } from '@react-
navigation/native';
import { create Sta ckN avi gator } from '@reac t-
n avi gat ion /st ack';
const Stack = create Sta ckN avi gat or();
const App = () => (
    <Na vig ati onC ont ain er>
        <St ack.Na vigator initia lRo ute Nam -
e="H ome ">
            <St ack.Screen name="H ome " compon -
ent ={H ome Screen} />
            <St ack.Screen name="D eta ils "
compon ent ={D eta ils Screen} />
        </S tac k.N avi gat or>
    </N avi gat ion Con tai ner>
);
```

Bottom Tab Navigation

```
import { NavigationContainer } from '@react-
navigation/native';
import { create Bot tom Tab Nav igator } from
'@reac t-n avi gat ion /bo tto m-t abs';
const Tab = create Bot tom Tab Nav iga tor();
const App = () => (
    <Na vig ati onC ont ain er>
        <Ta b.N avi gat or>
            <Ta b.S creen name="H ome " compon -
ent ={H ome Screen} />
            <Ta b.S creen name="S ett ing s"
compon ent ={S ett ing sSc reen} />
        </T ab.N av iga tor>
    </N avi gat ion Con tai ner>
);
```

Drawer Navigation

```
import { NavigationContainer } from '@react-
navigation/native';
import { create Dra wer Nav igator } from '@reac -
t-n avi gat ion /dr awer';
const Drawer = create Dra wer Nav iga tor();
const App = () => (
    <Na vig ati onC ont ain er>
        <Dr awe r.N avi gator initia lRo ute Nam -
e="H ome ">
            <Dr awe r.S creen name="H ome "
compon ent ={H ome Screen} />
            <Dr awe r.S creen name="S ett ing s"
compon ent ={S ett ing sSc reen} />
        </D raw er.N av iga tor>
    </N avi gat ion Con tai ner>
);
```

GradientButton

```
import React from 'react';
import { View, Toucha ble Opa city, Text } from
'react -na tive';
import Linear Gra dient from 'react -na tiv e-l -
ine ar- gra dient';
const Gradie ntB utton = ({
hollow = false,
border Start,
borderEnd,
start,
end,
onPress,
contai ner Style,
button Style,
textStyle,
text,
```

GradientButton (cont)

```
> }) => {
const computedBorderStart = borderStart || '#4780B5AA';
const computedBorderEnd = borderEnd || '#4C88C1AA';
const computedStart = start || '#4780B5';
const computedEnd = end || '#4C88C1';
return (
<TouchableOpacity onPress={onPress} style={containerStyle}>
<LinearGradient
colors={[computedBorderStart, computedBorderEnd]}
start={{ x: 0, y: 1 }}
end={{ x: 1, y: 0 }}
style={{ borderRadius: 100 }}>
<View style={{ padding: 4 }}>
<LinearGradient
colors={hollow ? ['white', 'white'] : [computedStart, computedEnd]}
start={{ x: 0, y: 1 }}
end={{ x: 1, y: 0 }}
style={{ borderRadius: 100 }}>
<View style={buttonStyle}>
<Text style={textStyle}>
{text}
</Text>
</View>
</LinearGradient>
</View>
</LinearGradient>
</TouchableOpacity>
);
}
export default GradientButton;
```

Redux

```
// npm install redux react-redux redux-thunk
// action s/t ypes.js
export const FETCH_ DAT A_BEGIN = 'FETCH _DA TA_ -
BEGIN';
export const FETCH_ DAT A_S UCCESS = 'FETCH _DA -
TA_ SUC CESS';
export const FETCH_ DAT A_F AILURE = 'FETCH _DA -
TA_ FAI LURE';
// action s/d ata Act ions.js
export const fetchD ata Begin = () => ({
    type: FETCH_ DAT A_B EGIN,
```

Redux (cont)

```
> });
export const fetchDataSuccess = (data) => ({
  type: FETCH_DATA_SUCCESS,
  payload: { data },
});
export const fetchDataFailure = (error) => ({
  type: FETCH_DATA_FAILURE,
  payload: { error },
});
// reducers/dataReducer.js
const initialState = {
  data: [],
  loading: false,
  error: null,
};
export default function dataReducer(state = initialState, action) {
  switch (action.type) {
    case FETCH_DATA_BEGIN:
      return {
        ...state,
        loading: true,
        error: null,
      };
    case FETCH_DATA_SUCCESS:
      return {
        ...state,
        loading: false,
        data: action.payload.data,
      };
    case FETCH_DATA_FAILURE:
      return {
        ...state,
        loading: false,
        error: action.payload.error,
        data: [],
      };
    default:
      return state;
  }
```

Redux (cont)

```
> }
// reducers/index.js
import { combineReducers } from 'redux';
import dataReducer from './dataReducer';
export default combineReducers({
  data: dataReducer,
});
// configureStore.js
import { createStore, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import rootReducer from './reducers';
const initialState = {};
const middleware = [thunk];
const store = createStore(
  rootReducer,
  initialState,
  applyMiddleware(...middleware)
);
export default store;
// App.js
import React from 'react';
import { Provider } from 'react-redux';
import store from './configureStore';
import Home from './Home'; // your main component
const App = () => (
  <Provider store={store}>
    <Home />
  </Provider>
);
export default App;
// Usage in components
import { connect } from 'react-redux';
import { fetchDataBegin } from './actions/dataActions';
const MyComponent = ({ data, fetchData }) => {
  // use data or call fetchData as needed
};
const mapStateToProps = state => ({
  data: state.data.data,
});
```

Redux (cont)

```
> const mapDispatchToProps = dispatch => ({
  fetchData: () => dispatch(fetchDataBegin()),
});
export default connect(mapStateToProps, mapDispatchToProps)(M-
yComponent);
```

React Native app feel smooth on animations

```
// The primary reason and an important one why
well-built native apps feel so smooth
// are by avoiding expensive operations during
intera ctions and animat ions.
// React Native has a limitation that there is
only a single JS execution thread,
// but you can use  Intera cti onM anager to make
sure long-r unning work is scheduled
// to start after any intera cti ons /an ima tions
have completed.
import React, { useState, useEffect } from
'react';
import { Text, Intera cti onM anager } from
'react -na tive';
const Expens ive Com ponent = () => {
    const [isDat aLo aded, setDat aLo aded] =
useSta te( false);
    use Eff ect(() => {
        Int era cti onM ana ger.ru nAf ter Int -
era cti ons(() => {
            loa dDa ta();
        });
   }, []);
    const loadData = () => {
       // Simulate a data loading process
        set Tim eout(() => {
            set Dat aLo ade d(t rue);
        }, 2000);
   };
   if (!isDa taL oaded) {
        return <Te xt> Loa din g...</ Tex t>;
   }
    return <Te xt>Data Loaded !</ Tex t>;
};
export default Expens ive Com ponent;
```