



1

 This is the second slide of the presentation. It has a blue header bar with the title 'useEffect' in white. In the top right corner, there is a small green Android robot icon surrounded by colorful dots. The main content area is white and contains a text box with the following text: 'useEffect is a crucial hook in React Native for managing side effects in functional components.' Below this text is a bulleted list:

- Fetching data from an API.
- Subscribing and unsubscribing from event listeners.
- Updating the component's state based on props.

 At the bottom of the slide, there is a code snippet: `useEffect(callback, dependencies) {}`. To the left of the code is a small image of an iPhone, and to the right is a small image of a Samsung smartphone. The number '2' is in the bottom right corner.

2

UseEffect

```
import React, { useState, useEffect } from 'react';
import { View, Text, Button, Alert } from 'react-native'

export default App=()=> {
  const [count, setCount] = useState(0);
  const [flag, setFlag] = useState(0);
  useEffect(() => {
    Alert.alert("Render");
  }, [])
  return (
    <View>
      <Text>You clicked {count} times</Text>

      <Button title="Click me" onPress={() => setCount(count + 1)}>
      <Button title="Click me" onPress={() => setFlag(!flag)}>
    </View>
  );
}
```

dependencies

<https://snack.expo.dev/@taitv/useeffect>

3

3

Validation

```
import React, { useState, useEffect } from 'react';
import {
  View,
  TextInput,
  TouchableOpacity,
  Text,
  StyleSheet,
} from 'react-native';

const App = () => {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [errors, setErrors] = useState({});

  useEffect(() => {
    let errors = ''
    // Validate name field
    if (!name) {
      errors += '\nName is required.';
    }
    // Validate email field
    if (!email) {
      errors += '\nEmail is required.';
    } else if (!/\S+@\S+\.\S+/.test(email)) {
      errors += '\nEmail is invalid.';
    }
    // Validate password field
    if (!password) {
      errors += '\nPassword is required.';
    } else if (password.length < 6) {
      errors += '\nPassword must be at least 6 characters.';
    }

    // Set the errors and update form validity
    setErrors(errors);
  }, [name, email, password]);

  return (
    <View style={styles.container}>
      <TextInput
        style={styles.input}
        placeholder="Name"
        value={name}
        onChangeText={setName}
      />
      <TextInput
        style={styles.input}
        placeholder="Email"
        value={email}
        onChangeText={setEmail}
      />
      <TextInput
        style={styles.input}
        placeholder="Password"
        value={password}
        onChangeText={setPassword}
      />
    </View>
  );
}
```

4

4

Validation

```

    placeholder="Password"
    value={password}
    onChangeText={setPassword}
    secureTextEntry
  />
  <TouchableOpacity
    style={styles.button}>
    <Text style={styles.buttonText}>Submit</Text>
  </TouchableOpacity>
  <Text style={styles.error}>{errors.toString()}</Text>
</View>
);
};

// Styles for the components
const styles = StyleSheet.create({
  container: {
    flex: 1,
    padding: 16,
    justifyContent: 'center',
  },
  input: {
    height: 60,
    borderColor: '#ccc',
    borderWidth: 1,
    marginBottom: 12,
    paddingHorizontal: 10,
    borderRadius: 8,
  },

```

```

    borderColor: '#ccc',
    borderWidth: 1,
    marginBottom: 12,
    paddingHorizontal: 10,
    borderRadius: 8,
    fontSize: 16,
  },
  button: {
    backgroundColor: 'green',
    borderRadius: 8,
    paddingVertical: 10,
    alignItems: 'center',
    marginTop: 16,
    marginBottom: 12,
  },
  buttonText: {
    color: 'fff',
    fontWeight: 'bold',
    fontSize: 16,
  },
  error: {
    color: 'red',
    fontSize: 20,
    marginBottom: 12,
  },
});

export default App;

```



5

5

Validation

Name is required.
 Email is required.
 Password is required.



<https://snack.expo.dev/@taitv/validation>



6

6

Validation

```
import React, { useState } from 'react';
import { View, Text, TextInput, Button, StyleSheet } from 'react-native';

const ValidationExample = () => {
  const [email, setEmail] = useState('');
  const [isValid, setIsValid] = useState(true);

  const validateEmail = () => {
    // Simple email validation with a regular expression
    const emailRegex = /\S+@\S+\.\S+/;
    const valid = emailRegex.test(email);
    setIsValid(valid);
  };

  return (
    <View style={styles.container}>
      <Text>Email Address:</Text>
      <TextInput
        style={styles.input}
        value={email}
        onChangeText={(text) => setEmail(text)}
        onBlur={validateEmail}
      />
      {!isValid && <Text style={styles.errorText}>Invalid email address</Text>}
      <Button title="Submit" onPress={() => validateEmail()} />
    </View>
  );
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
  input: {
    width: 200,
    borderColor: 'gray',
    borderWidth: 1,
    padding: 10,
    marginBottom: 10,
  },
  errorText: {
    color: 'red',
  },
});

export default ValidationExample;
```



7

7

Validation

Email Address:

tran.tai

Invalid email address

SUBMIT



<https://snack.expo.dev/@taitv/validation2>



8

8

Exercise

Perform validation with the following form in Lab 3:

1. All fields can't be blank
2. "Description" must be larger than 50 characters.
3. "Price", "Discount Percentage", "rating", "stock": Number.
4. "Image" includes HTTP or HTTPS.



1. `toLowerCase()`
2. `toUpperCase()`
3. `trim()`
4. `length()`
5. `substring()`
6. `charAt(index)`
7. `replaceAll(find, replace)`
8. `Split(separator)`
9. `equals()`
10. `equalsIgnoreCase()`
11. `Contains()`
12. `startsWith()`
13. `endsWith()`
14. `matches()`
15. `indexOf()`
16. `lastIndexOf()`



9

9



Q&A



TRƯỜNG ĐẠI HỌC
QUỐC TẾ
MIỀN ĐÔNG
EASTERN
INTERNATIONAL
UNIVERSITY

Mobile programming
10

10