Department of
Media Software

Mobile Computing
10th Lecture

useState/useEffect/useRef/useMemo/커스텀 Hooks 만들기
일정관리 앱 보강

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Learning objective

useState/useEffect/useRef/useMemo/커스텀 Hooks 의 이해를 통해 개발할 수 있는 능력을 배양한다.

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

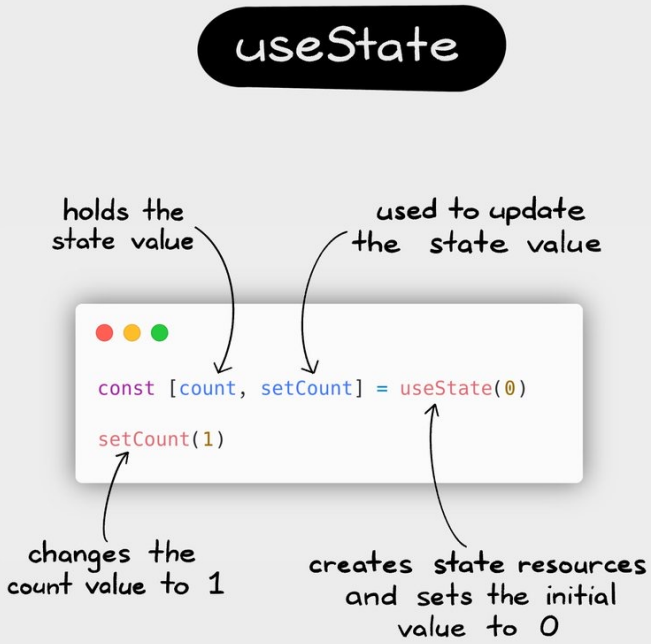
React Native에서 Hooks는 함수형 컴포넌트에서 클래스 컴포넌트의 기능을 사용할 수 있게 도와주는 기능입니다. Hooks를 사용하면 상태 관리, 생명주기 메서드, 컨텍스트 등의 기능을 함수형 컴포넌트에서도 활용할 수 있으며, 코드를 간결하게 유지하면서 구조를 간단하게 만들 수 있습니다. 주요 Hooks로는 useState, useEffect, useContext, useRef, useMemo, useCallback 등이 있습니다.

- React Native는 React를 기반으로 한 프레임워크로, 모바일 애플리케이션을 개발할 때 사용됩니다. React Native에서도 React와 마찬가지로 Hooks를 사용할 수 있습니다.
- Hooks는 React 함수형 컴포넌트에서 클래스 컴포넌트의 기능들을 사용할 수 있게 해주는 기능입니다. Hooks는 React 16.8 버전부터 도입되었습니다.

Prof.
Gabkeun Choi Ph.D.

Department of Media Software

Hook

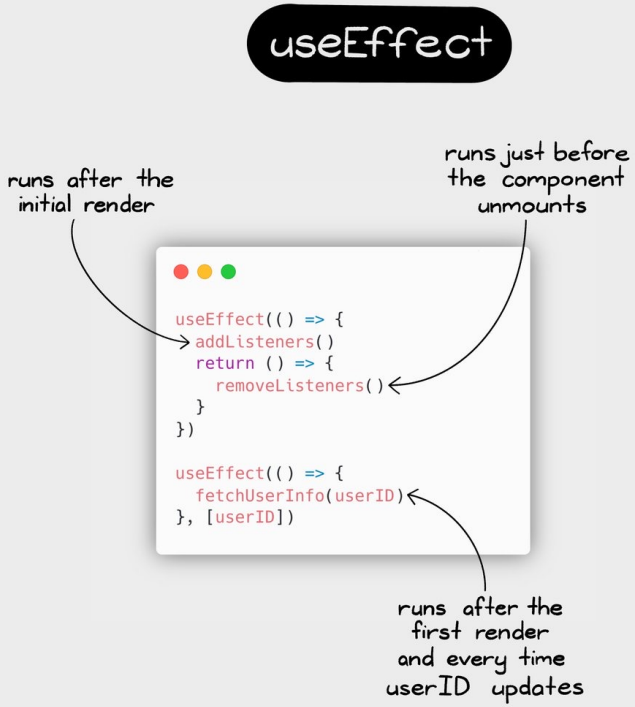


- useState는 가장 기본적인 Hook으로, 함수형 컴포넌트에서 상태를 관리하기 위해 사용됩니다.
- useState는 초기 상태 값을 인자로 받고, 배열을 반환합니다. 이 배열에는 현재 상태 값과 상태를 업데이트하는 함수가 포함되어 있습니다. 배열 구조 분해를 사용해 상태 값과 업데이트 함수를 각각의 변수에 할당할 수 있습니다.
- count는 현재 상태 값이고, setCount는 상태를 업데이트하는 함수입니다. setCount를 호출하면 상태가 업데이트되고 컴포넌트가 다시 렌더링됩니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

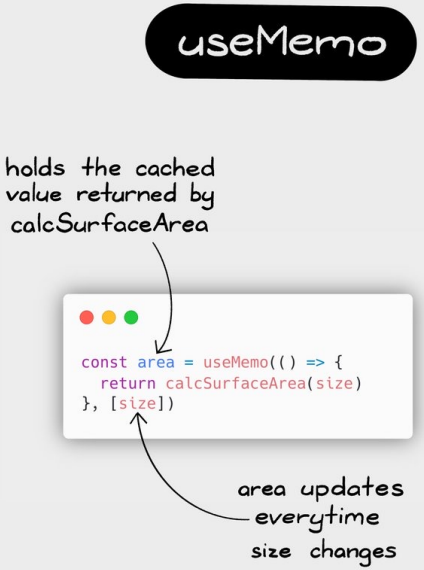


- useEffect는 React Native의 Hook 중 하나로, 함수형 컴포넌트에서 side effect를 관리할 수 있게 해줍니다. Side effect란, 컴포넌트 외부에 영향을 미치는 작업을 말하며, API 호출, 이벤트 구독, DOM 조작 등이 해당됩니다. useEffect는 기본적으로 클래스형 컴포넌트의 componentDidMount, componentDidUpdate, componentWillUnmount와 같은 생명주기 메서드를 대체합니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

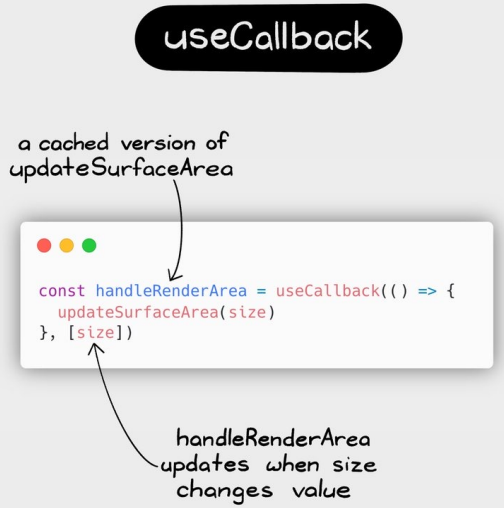


- useMemo는 React Native의 Hook 중 하나로, 함수형 컴포넌트에서 메모이제이션(memoization)을 사용하여 성능을 최적화할 수 있게 해줍니다. 메모이제이션은 결과를 캐시에 저장하여 동일한 입력에 대해 이전에 계산된 결과를 재사용함으로써 비싼 연산을 최소화하는 기법입니다.
- useMemo는 두 개의 인자를 받습니다. 첫 번째 인자는 결과를 생성하는 함수이고, 두 번째 인자는 의존성 배열입니다.
- useMemo는 의존성 배열의 값이 변경될 때마다 결과를 생성하는 함수를 호출하고, 그 결과를 메모이제이션합니다. 의존성 배열의 값이 변경되지 않으면 이전에 메모이제이션된 결과를 재사용합니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook



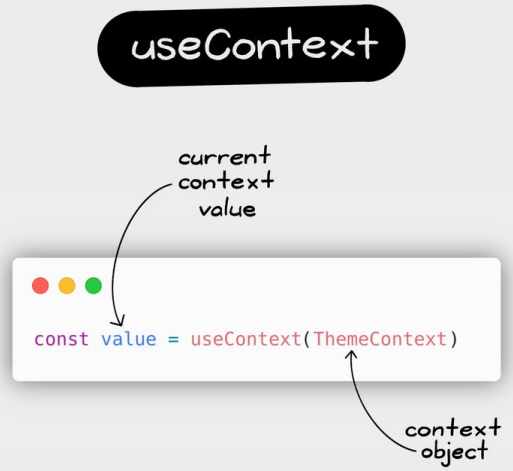
- useCallback은 React Native의 Hook 중 하나로, 함수형 컴포넌트에서 함수를 메모이제이션(memoization)할 수 있게 해줍니다. 이를 통해 불필요한 리렌더링을 방지하고 성능을 최적화할 수 있습니다. 일반적으로 이벤트 핸들러와 같이 자주 호출되는 함수에 대해 사용됩니다.

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

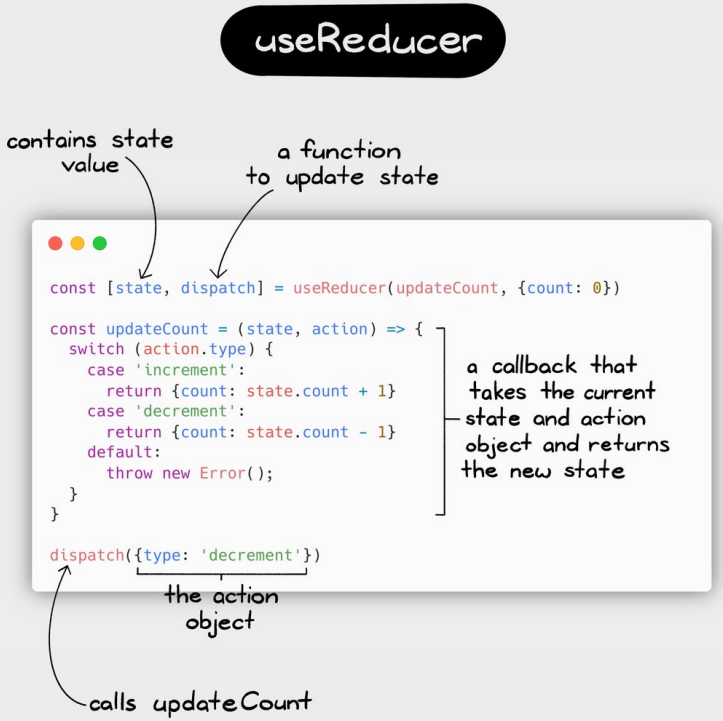


- useContext는 React Native의 Hook 중 하나로, 함수형 컴포넌트에서 React의 Context API를 사용하여 전역 상태를 관리할 수 있게 해줍니다. Context API는 상위 컴포넌트에서 하위 컴포넌트로 데이터를 전달할 때 프롭스(props)를 여러 레벨에 걸쳐 전달하는 대신, 상위 컴포넌트에서 생성한 데이터를 하위 컴포넌트에서 바로 사용할 수 있게 해줍니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook



- useReducer는 React Native의 Hook 중 하나로, 함수형 컴포넌트에서 상태 관리를 더 명확하게 하기 위한 방법을 제공합니다. useReducer는 특히 복잡한 상태 로직이 있는 경우에 유용하며, 여러 상태 값을 함께 관리할 때 사용할 수 있습니다. useReducer는 리듀서(reducer) 함수를 사용하여 상태를 업데이트하고, 이를 통해 더욱 예측 가능한 방식으로 상태 변경을 관리할 수 있습니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

- useState: 이 Hook은 컴포넌트의 상태(state)를 관리합니다. 초기값을 설정하고, 상태를 변경하기 위한 함수를 반환합니다. 클래스 컴포넌트의 this.state와 this.setState에 해당하는 기능을 수행합니다.

```
import React, {useState} from 'react';
import {Text, TouchableOpacity} from 'react-native';

const Counter = () => {
  const [count, setCount] = useState(0);

  return (
    <>
      <Text>{count}</Text>
      <TouchableOpacity onPress={() => setCount(count + 1)}>
        <Text>증가</Text>
      </TouchableOpacity>
    </>
  );
};
```

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

- useEffect: 이 Hook은 componentDidMount, componentDidUpdate, componentWillUnmount와 같은 생명주기 메서드를 대체합니다. useEffect 내부에서 수행되는 함수는 종속성 배열(dependency array)에 명시된 값이 변경될 때마다 실행됩니다.

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

const Timer = () => {
  const [time, setTime] = useState(0);

  useEffect(() => {
    const timer = setInterval(() => {
      setTime((prevTime) => prevTime + 1);
    }, 1000);

    // cleanup 함수입니다. 컴포넌트가 unmount될 때 실행됩니다.
    return () => clearInterval(timer);
  }, []);

  return <Text>경과 시간: {time}초</Text>;
};
```

Department of
Media Software

Hook

- useContext: 이 Hook은 주어진 context 객체의 현재 값을 반환합니다. 이를 통해 전역 상태를 관리할 수 있습니다.
- useRef: 이 Hook은 변경 가능한 ref 객체를 생성합니다. useRef는 일반적으로 DOM 요소에 접근할 때 사용되며, React Native에서도 비슷한 목적으로 사용할 수 있습니다.
- useMemo, useCallback: 이 두 Hook은 최적화 목적으로 사용됩니다. useMemo는 메모이제이션된 값의 결과를 반환하고, useCallback은 메모이제이션된 함수를 반환합니다. 이들은 연산 비용이 큰 함수나 값들을 캐싱하여 성능을 개선할 수 있습니다.

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

Hook

- 다음의 예제는 React Native에서 useState, useEffect, useContext, useRef, useMemo, useCallback을 모두 사용하는 간단한 앱입니다. 이 앱은 글로벌 상태로 테마를 관리하고, 타이머 기능과 리스트를 가지고 있습니다. 리스트 항목을 추가할 때마다 useCallback을 사용하여 최적화를 수행하며, useMemo를 사용하여 리스트 렌더링을 최적화합니다.

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

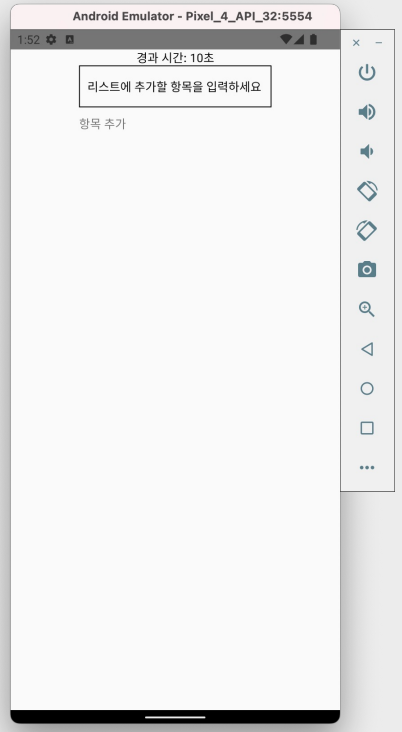
Hook

- 배포코드 참조

배포 예제에서는 다음과 같은 작업이 이루어집니다:

1) 글로벌 테마 상태를 관리하기 위해 `ThemeContext`를 생성합니다.
2) `App` 컴포넌트에서 테마를 변경하는 기능과 `Timer` 및 `List` 컴포넌트를 렌더링합니다.
3) `Timer` 컴포넌트에서 `useEffect`를 사용하여 타이머를 구현하고, `useContext`로 테마에 따른 텍스트 색상을 설정합니다.
4) `List` 컴포넌트에서 리스트를 관리하고, 새 항목을 추가하는 기능을 구현합니다. 여기서 `useRef`로 `TextInput`에 접근하여 입력을 지우는 작업을 수행하고 있습니다. 또한 `useCallback`을 사용하여 `renderItem`과 `addItem` 함수를 최적화하고, `useMemo`를 사용하여 `keyExtractor`를 최적화합니다.

이 예제를 통해 여러 Hooks를 조합하여 React Native 애플리케이션의 기능을 구현하는 방법을 확인할 수 있습니다.



ex10_1

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

일정관리

I. 메인 화면에는 일정 목록이 표시됩니다.
II. 새 일정을 추가하는 "+" 버튼을 추가합니다.
III. 각 일정에는 제목, 날짜, 시간 및 설명이 표시됩니다.

이 코드는 일정 관리 앱에서 제목, 내용, 날짜를 모두 작성할 수 있습니다. 일정을 추가하려면 "+" 버튼을 눌러 입력 창이 나타나고, 일정 제목, 내용, 날짜를 입력한 후 "Add" 버튼을 눌러 일정을 추가할 수 있습니다. 일정을 삭제하려면 각 일정 항목 옆에 있는 "X" 버튼을 누르면 됩니다.

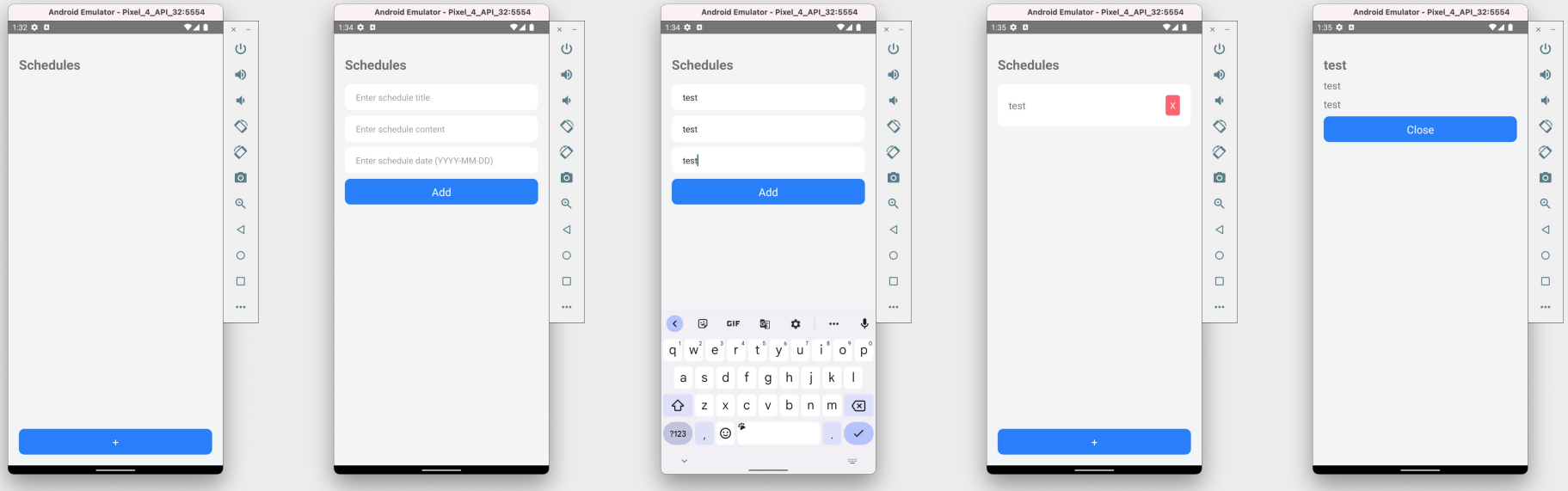
Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

일정관리



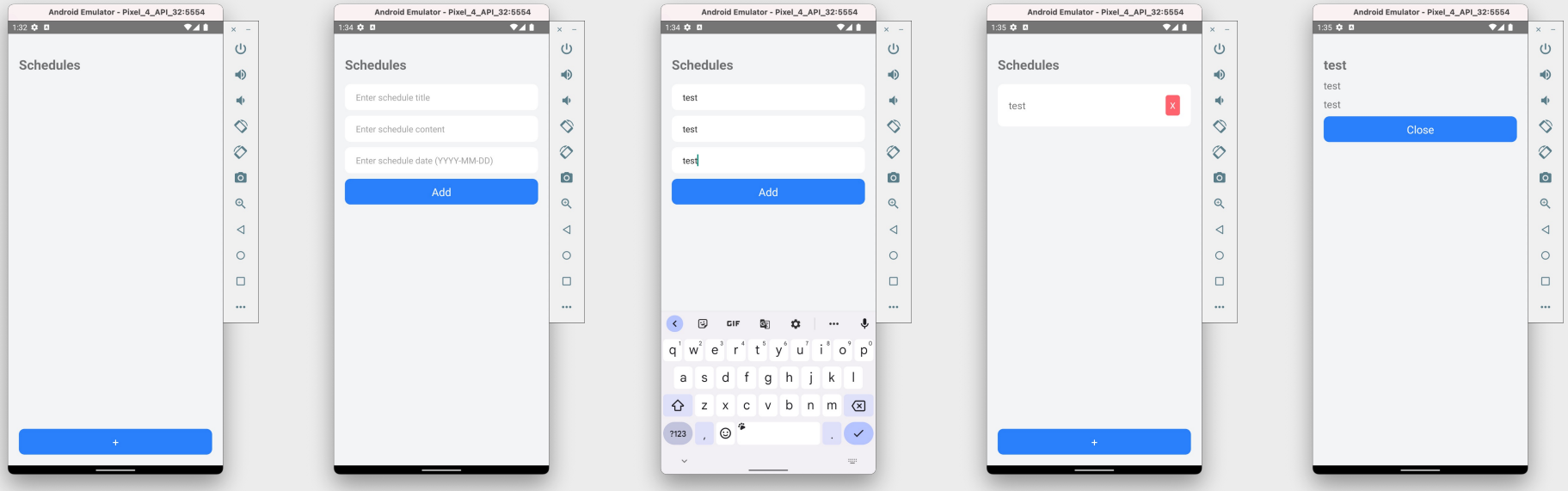
예제코드 : ex10_2

Department of
Media Software

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

일정관리



예제코드 : ex10_2

Department of
Media Software

실습

I. eventItem에 삭제 버튼을 추가
II. 이벤트를 삭제하는 함수 deleteEvent를 작성
III. delete_event.php 스크립트를 작성하여 데이터베이스에서 이벤트를 삭제

2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

실습

I. eventItem에 삭제 버튼을 추가

```
<TouchableOpacity
  style={styles.deleteButton}
  onPress={() => deleteEvent(item.id)}>
  <Text style={styles.deleteButtonText}>삭제</Text>
</TouchableOpacity>
```


Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

실습

II. 이벤트를 삭제하는 함수 deleteEvent를 작성(App 컴포넌트에 deleteEvent 함수를 추가)

```
const deleteEvent = async (id) => {
  try {
    await axios.post('http://34.64.185.101/delete_event.php', { id });
    fetchEvents();
  } catch (error) {
    console.error(error);
  }
};
```


Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

실습

III. delete_event.php 스크립트를 작성하여 데이터베이스에서 이벤트를 삭제

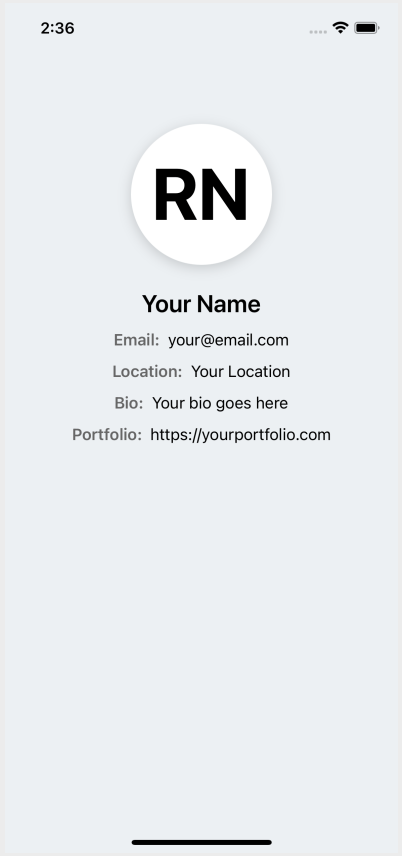
```
<?php
// Database connection
require_once 'db.php';

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
  $id = json_decode(file_get_contents("php://input"))->id;

  try {
    $query = "DELETE FROM events WHERE id = :id";
    $stmt = $pdo->prepare($query);
    $stmt->execute(['id' => $id]);

    echo json_encode(['message' => 'Event deleted successfully']);
  } catch (PDOException $e) {
    echo json_encode(['error' => $e->getMessage()]);
  }
}
?>
```

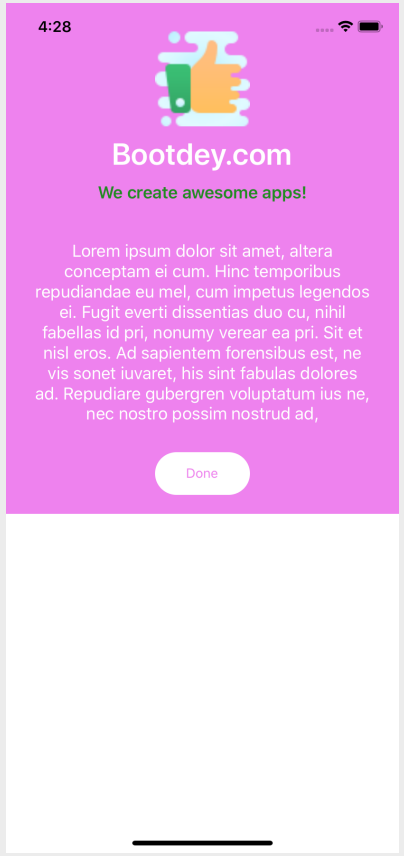
```
header('Content-Type: application/json');
header('Access-Control-Allow-Origin: *');
header('Access-Control-Allow-Methods: POST');
header('Access-Control-Allow-Headers: Access-Contr
ol-Allow-Headers,Content-Type,Access-Control-Allow
-Methods, Authorization, X-Requested-With');
```

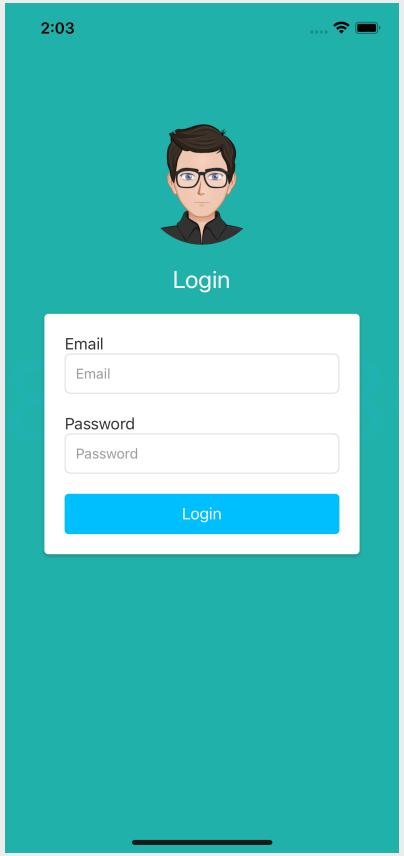

Prof.
Gabkeun Choi Ph.D.

Department of
Media Software

실습 2







2023 1st semester
Mobile Computing

Prof.
Gabkeun Choi Ph.D.