고급소프트웨어실습1 7주차 과제

컴퓨터공학 20172141 김미소

1. 주요 컴포넌트 상호관계 및 구조

이번 실습에서는 TodoTemplate.jsx 파일만 사용하였으므로 나머지 jsx파일에 담긴 컴포넌트는 간단히 설명만 하겠다.

1) DeletedList와 DeletedTodoItem의 상호관계

DeletedList는 deletedTodos의 각 항목 객체를 가지고 DeletedTodoItem 컴포넌트를 생성하여 DELETED LIST를 보여준다. DeletedTodoItem은 삭제된 각 항목 객체를 한 줄씩 보여주는 컴포넌트이다.

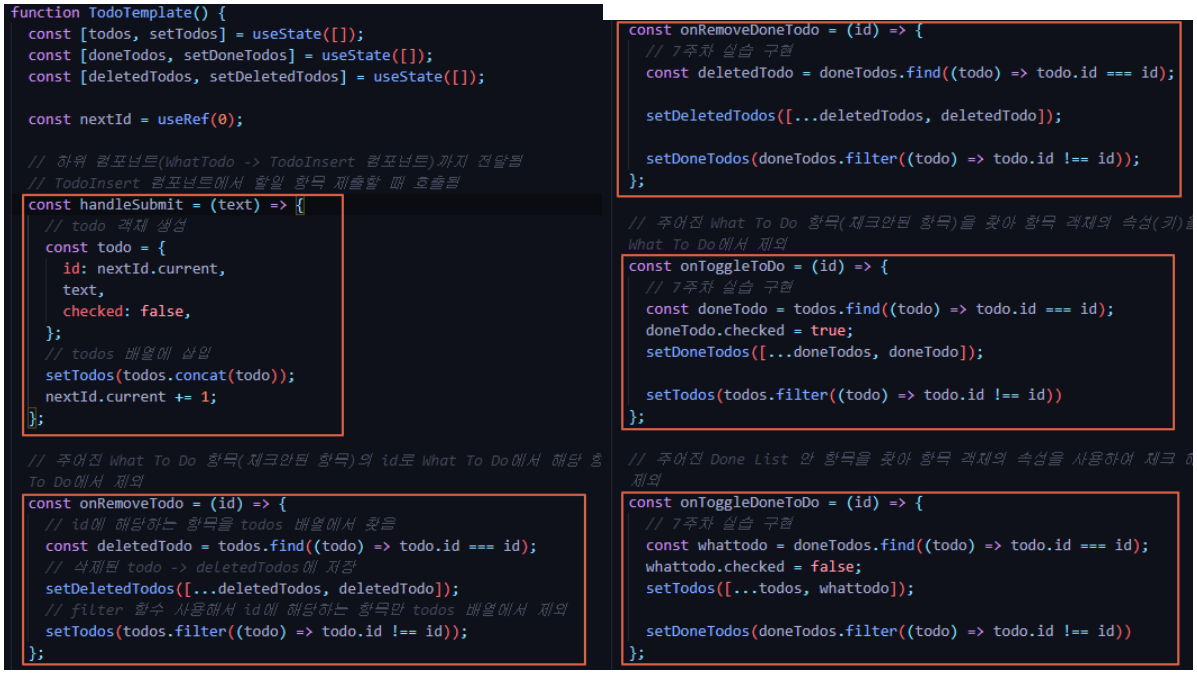
2) DoneList와 TodoItem의 상호관계

DeletedList와 유사하게 doneTodos의 각 항목 객체를 가지고 TodoItem 컴포넌트를 생성하여 DONE LIST 부분에 리스트로 보여주는 컴포넌트이다. TodoItem은 checked 값이 true이면 줄 표시와 함께 항목 객체가 한 줄씩 표시되고 checked 값이 false이면 그냥 표시된다. DONE LIST는 checked 값이 true이므로 줄 표시와 함께 표시된다.

3) TodoInsert, TodoTemplate, WhatTodo, TodoItem의 상호관계

할 일을 삽입하는 입력 폼을 형성하는 컴포넌트이다. 내용 없이 입력 시도를 하는 경우 아무 입력도 하지 않고 내용을 입력한 후에는 입력 폼에 내용을 비워 초기화한다. 이 컴포넌트를 통해 생성된 입력 폼은 TodoTemplate에서 할 일을 입력하고 삽입할 때 사용된다. 그리고 삽입된 할 일은 WhatTodo 컴포넌트를 통해 화면에 리스트로 표시된다. WHAT TO DO는 checked 값이 false이므로 줄 표시 없이 할 일이 표시된다.

4) TodoTemplate 컴포넌트 구조



```
function TodoTemplate() {
  const [todos, setTodos] = useState([]);
  const [doneTodos, setDoneTodos] = useState([]);
  const [deletedTodos, setDeletedTodos] = useState([]);

  const nextId = useRef(0);

  // 하위 컴포넌트(WhatTodo -> TodoInsert 컴포넌트)까지 전달됨
  // TodoInsert 컴포넌트에서 할일 항목 제출할 때 호출됨
  const handleSubmit = (text) => {
    // todo 객체 생성
    const todo = {
      id: nextId.current,
      text,
      checked: false,
    };
    // todos 배열에 삽입
    setTodos(todos.concat(todo));
    nextId.current += 1;
  };

  // 주어진 What To Do 항목(체크안된 항목)의 id로 What To Do에서 해당 항
  To Do에서 제외
  const onRemoveTodo = (id) => {
    // id에 해당하는 항목을 todos 배열에서 찾음
    const deletedTodo = todos.find((todo) => todo.id === id);
    // 삭제된 todo -> deletedTodos에 저장
    setDeletedTodos([...deletedTodos, deletedTodo]);
    // filter 함수 사용해서 id에 해당하는 항목만 todos 배열에서 제외
    setTodos(todos.filter((todo) => todo.id !== id));
  };
```

```
  const onRemoveDoneTodo = (id) => {
    // 7주차 실습 구현
    const deletedTodo = doneTodos.find((todo) => todo.id === id);

    setDeletedTodos([...deletedTodos, deletedTodo]);

    setDoneTodos(doneTodos.filter((todo) => todo.id !== id));
  };

  // 주어진 What To Do 항목(체크안된 항목)을 찾아 항목 객체의 속성(키)
  What To Do에서 제외
  const onToggleToDo = (id) => {
    // 7주차 실습 구현
    const doneTodo = todos.find((todo) => todo.id === id);
    doneTodo.checked = true;
    setDoneTodos([...doneTodos, doneTodo]);

    setTodos(todos.filter((todo) => todo.id !== id))
  };

  // 주어진 Done List 안 항목을 찾아 항목 객체의 속성을 사용하여 체크 ㅎ
  제외
  const onToggleDoneToDo = (id) => {
    // 7주차 실습 구현
    const whattodo = doneTodos.find((todo) => todo.id === id);
    whattodo.checked = false;
    setTodos([...todos, whattodo]);

    setDoneTodos(doneTodos.filter((todo) => todo.id !== id))
  };
```

```
  return (
    <div>
      <Container>
        <Title>
          <h1>TODO LIST</h1>
        </Title>
        <Contents>
          <WhatTodo handleSubmit={handleSubmit} todos={todos} onRemove={onRemoveTodo} onToggle={onToggleToDo}/>
          {/* 구분선 위한 div*/}
          <div style={{'borderLeft': '2px solid', height: '400px', 'marginLeft': '10px'}}/>
          <DoneList doneTodos={doneTodos} onRemove={onRemoveDoneTodo} onToggle={onToggleDoneToDo} />
        </Contents>
      </Container>
      <footer>
        <DeletedList deletedTodos={deletedTodos} />
      </footer>
    </div>
  );
}
```

TodoTemplate 컴포넌트 안에는 할 일의 모음인 todos와, 끝낸 할 일의 모음인 doneTodos, 삭제된 할 일의 모음인 deletedTodos가 있고 함수로는

1. 할 일을 입력하여 todos에 할 일을 삽입하는 handleSubmit 함수

2. WHAT TO DO에서 DELETED LIST로 할 일을 옮기는 onRemoveTodo 함수

3. DONE LIST에서 DELETED LIST로 item을 옮기는 onRemoveDoneTodo 함수

4. WHAT TO DO에서 DONE LIST로 item을 옮기는 onToggleToDo 함수

5. DONE LIST에서 WHAT TO DO로 item을 옮기는 onToggleDoneTodo 함수가 있다.

그리고 그 아래 return 부분에는 화면에 TODO LIST를 viewing하는 부분이 있다. WhatTodo 컴포넌트에 관여하는 요소들은 handleSubmit, todos, onRemoveTodo, onToggleTodo가 있고 DoneList 컴

포넌트에 관여하는 요소들은 doneTodos, onRemoveDoneTodo, onToggleDoneToDo가 있다. DeletedList 컴포넌트는 삭제된 할 일만 표시하면 되므로 deletedTodos만 관여한다.

2. 실습 프로젝트 주요 함수 기능 및 코드

1) onRemoveTodo

```
const onRemoveTodo = (id) => {
  // id에 해당하는 항목을 todos 배열에서 찾음
  const deletedTodo = todos.find((todo) => todo.id === id);
  // 삭제된 todo -> deletedTodos에 저장
  setDeletedTodos([...deletedTodos, deletedTodo]);
  // filter 함수 사용해서 id에 해당하는 항목만 todos 배열에서 제외
  setTodos(todos.filter((todo) => todo.id !== id));
};
```

WHAT TO DO에서 DELETED LIST로 할 일을 옮기는 함수이다.

WHAT TO DO에서 할 일을 삭제하므로 삭제할 할 일 deletedTodo를 todos에서 id를 통해 찾는다. 그리고 setDeletedTodos를 통해 deletedTodos에 deletedTodo를 저장하고 WHAT TO DO에서 이 항목을 빼야 하므로 setTodos를 이용하여 todos에서 filter를 통해 삭제할 항목을 제외시킨다.

2) onRemoveDoneTodo

```
const onRemoveDoneTodo = (id) => {
  // 7주차 실습 구현
  const deletedTodo = doneTodos.find((todo) => todo.id === id);

  setDeletedTodos([...deletedTodos, deletedTodo]);

  setDoneTodos(doneTodos.filter((todo) => todo.id !== id));
};
```

DONE LIST에서 DELETED LIST로 item을 옮기는 함수이다.

onRemoveTodo와 비슷하게 완료된 할 일 doneTodos에서 id를 통해 deletedTodo를 찾는다. 그리고 setDeletedTodos를 통해 deletedTodos에 deletedTodo를 저장하고 DONE LIST에서 이 항목을 빼야 하므로 setDoneTodos를 이용하여 doneTodos에서 filter를 통해 삭제할 항목을 제외시킨다.

3) onToggleToDo

```
const onToggleToDo = (id) => {
  // 7주차 실습 구현
  const doneTodo = todos.find((todo) => todo.id === id);
  doneTodo.checked = true;
  setDoneTodos([...doneTodos, doneTodo]);

  setTodos(todos.filter((todo) => todo.id !== id))
};
```

WHAT TO DO에서 DONE LIST로 item을 옮기는 함수이다.

앞의 함수들과 방식이 똑같지만 여기서는 checked를 true로 활성화하여야 한다.

DONE LIST로 항목을 옮기기 위해 todos에서 id를 통해 doneTodo를 찾는다. 그리고 doneTodo의 checked를 true로 활성화하여 DONE LIST에 줄 표시와 함께 출력될 수 있도록 한다. 그리고 setDoneTodos를 통해 doneTodos에 doneTodo를 저장하고 WHAT TO DO에서 이 항목을 빼야 하므로 setTodos를 이용하여 todos에서 filter를 통해 이 항목을 제외시킨다.

4) onToggleDoneTodo

```
const onToggleDoneToDo = (id) => {
  // 7주차 실습 구현
  const whattodo = doneTodos.find((todo) => todo.id === id);
  whattodo.checked = false;
  setTodos([...todos, whattodo]);

  setDoneTodos(doneTodos.filter((todo) => todo.id !== id))
};
```

DONE LIST에서 WHAT TO DO로 item을 옮기는 함수이다.

onToggleToDo 함수와 비슷하게 WHAT TO DO로 항목을 옮기기 위해 doneTodos에서 id를 통해 whattodo를 찾는다. 그리고 whattodo의 checked를 false로 활성화하여 WHAT TO DO에 줄 표시 없이 출력될 수 있도록 한다. 그리고 setTodos를 통해 todos에 whattodo를 저장하고 DONE LIST에서 이 항목을 빼야 하므로 setDoneTodos를 이용하여 doneTodos에서 filter를 통해 이 항목을 제외시킨다.

3. 실습 프로젝트 주요 함수 간 상호관계

onRemoveTodo 함수와 onRemoveDoneTodo 함수는 둘 다 DELETED LIST에 영향을 준다. 그리고 onToggleToDo 함수와 onToggleDoneTodo 함수는 서로 WHAT TO DO와 DONE LIST에 체크 여부와 함께 영향을 준다. DELETED LIST에서 WHAT TO DO로 항목을 보내거나 DONE LIST로 항목을 보내는 기능이 없어 onRemoveTodo 함수와 onRemoveDoneTodo 함수에서는 각 리스트에서 제외하

거나 추가하는 작업을 잘 한다면 주의할 점이 크지 않다. 그러나 onToggleToDo 함수와 onToggleDoneTodo 함수는 서로의 리스트 상태에 영향을 주는 함수들이므로 상태가 꼬이지 않게 주의하여 작성하여야 한다.