npm i react-transition-group -g

import { CSSTransition } from ‘react-transition-group’;

**React Transition Group**

**"react-transition-group"**은 리액트 컴포넌트 생명주기와 관련하여 전환 효과를 쉽게 적용하기 위한 라이브러리입니다.

기존 CSS만을 사용하여 transition 혹은 animation 프로퍼티를 사용하여 전환효과를 적용할 수 있었습니다. 하지만 컴포넌트가 돔에서 제거되거나 추가될 때 전환 효과를 적용하기 위해 CSS 만을 사용하는 것은 한계점이 존재하게 됩니다.

**컴포넌트가 돔에서 마운트나 언마운트되는 즉시 추가되거나 제거**되기 때문에 스타일 적용이 불편합니다. 이때 react-transition-group을 사용하면 **마운트될 때 언마운트될 때 생명주기와 관련지어 전환 효과를 쉽게 적용**할 수 있습니다.

자세한 내용은 [공식문서](https://reactcommunity.org/react-transition-group/)에서 확인할 수 있습니다.

react-transition-group 라이브러리를 설치하기 위해서 터미널에 아래 명령어를 작성해줍니다.

npm install react-transtion-group

**Transition Component**

Transition 컴포넌트는 react-transition-group/Transition에서 Transition 컴포넌트를 import합니다.

전환 효과를 필요로하는 컴포넌트를 Transtion 컴포넌트로 감싼 뒤, Transition 컴포넌트 내에는 **함수를 작성**하고 **함수의 반환값으로 렌더링될 리액트 엘리먼트를 작성**해줍니다.

import Transition from 'react-transitino-group/Transtion';

<Transtion in={불리언값} timeout={정수 | 객체} mountOnEnter unmountOnExit>

{state => {

,,,

// state: \*\* -> entering -> entered -> exiting -> exited -> \*\*

return <Component />;

}}

</Transition>

* **"in" 어트리뷰트**  
  : in 어트리뷰트 값에 의해 **state 값이 동적으로 변경**됩니다.  
  in 어트리뷰트 값이 **true가 되면 entering에서 entered로 상태가 변경(entering -> entered)**가 되고, **false가 되면 exiting에서 exited로 변경(exiting -> exited)**
* **"timeout" 어트리뷰트**  
  : timeout 어트리뷰트에는 **전환 효과가 재생되는 시간을 결정**합니다.  
  즉, **entering에서 entered로, exiting에서 exited로 변경될 때의 지속되는 시간을 결정**합니다.
  + **정수값 작성시 밀리초**로 동작하게 됩니다.  
    만약 timeout에 300을 작성시 entering에서 entered로 전환하는데 300ms, exiting에서 exited로 전환하는데 300ms가 걸리게됩니다.
  + **객체 작성시** 작성한 숫자 또한 밀리초도 동작합니다.  
    만약 { enter: 1000, exit: 2000 }이라면 entering -> entered가 1000ms, exiting -> exit는 2000ms로 상태가 변환됩니다.
* **"state(상태값)"**  
  : Transtion 컴포넌트 내 작성한 함수는 **인수로 상태값을 전달**받습니다. 이 상태값은 **in 어트리뷰트 값에 의해 동적으로 변경**됩니다. 상태값의 종류로 **"entering", "entered", "exiting", "exited"** 네 가지가 존재합니다.  
  상태값은 "entering" -> "entered" -> "exiting -> "exited" 흐름으로 변경됩니다.
* **"mountOnEnter" 어트리뷰트**  
  : mountOnEnter 어트리뷰트는 **불리언 값**을 갖습니다.  
  **in 어트리뷰트 값이 true**라면 함수의 반환값에 작성된 **엘리먼트를 돔에 마운트**되도록, 즉 렌더링되도록 설정합니다. in 어트리뷰트 값이 true라는 것은 state 값이 entering이 되었다는 의미이며 **상태가 entering이 되면 엘리먼트가 돔에 마운트**되도록 합니다.
* **"unmountOnExit" 어트리뷰트**  
  : unmountOnExit 어트리뷰트는 **불리언 값**을 갖습니다.  
  **in 어트리뷰트 값이 false**일 때 함수의 반환값에 작성된 **엘리먼트를 돔에서 언마운트**, 즉 렌더링되지 않도록 설정하는 어트리뷰트입니다. in 어트리뷰트 값이 false가 되면 state 값이 exiting에서 exited로 변경됩니다. 이때 **상태가 exited일 때 돔에서 엘리먼트를 언마운트**되도록 합니다.
* **"onEnter" 어트리뷰트**  
  : onEnter 어트리뷰트의 경우 값으로 **함수를 전달받습니다**. 이때 작성한 함수의 경우 **상태값이 entering 직전**에 실행됩니다.
* **"onEntering" 어트리뷰트**  
  : **상태값이 entering으로 변경된 직후**에 실행되는 함수입니다.
* **"onEntered" 어트리뷰트**  
  : **상태값이 entered로 변경된 직후** 실행되는 함수입니다.
* **"onExit" 어트리뷰트**  
  : **상태값이 exiting으로 변경된 직전**에 실행되는 함수입니다.
* **"onExiting" 어트리뷰트**  
  : **상태값이 exiting으로 변경된 직후**에 실행되는 함수입니다.
* **"onExited" 어트리뷰트**  
  : **상태값이 exited로 변경된 직후** 실행되는 함수입니다.

**CSSTransition Component**

CSSTransition 컴포넌트는 "react-transition-group/CSSTransition" 패키지에서 import하여 사용합니다.

CSSTransition 컴포넌트는 태그 사이, 즉 Content 영역에 함수를 작성하지 않고 **Content 영역에 렌더링될 리액트 엘리먼트를 작성**해줍니다.

CSSTransition 컴포넌트에도 **위에서 살펴본 Transition 컴포넌트의 어트리뷰트를 모두 사용**할 수 있습니다. 즉, in, timeout, mounOnEnter, unmoundOnExit, onEnter, onEntering, onEntered, onExit, onExiting, onExited 등 모두 사용 가능합니다.

CSSTransition 컴포넌트는 Transition 컴포넌트에는 없는 **"classNames" 어트리뷰트를 사용**할 수 있습니다.

**classNames 어트리뷰트에 작성된 값을 className 어트리뷰트 값의 prefix로 사용**되며 CSSTransition 컴포넌트의 **상태에 따라서 className 값이 설정**됩니다. **설정된 값은 CSSTransition 컴포넌트가 래핑하고 있는 리액트 엘리먼트에 적용**이 됩니다.

예를 들어, classNames 어트리뷰트 값이 "fade-slide"인 경우 CSSTransition 컴포넌트의 상태에 따라 className 값이 같이 결정됩니다.

* 상태가 entering 직후  
  -> "fade-slie**-enter**"로 결정
* 상태가 entering에서 entered로 변경되는 동안  
  -> "fade-slie**-enter** fade-slide**-enter-active**"로 결정
* 상태가 entered된 직후  
  -> "fade**-enter-done**"으로 결정
* 상태가 exiting된 직후  
  -> "fade-slide**-exit**"로 결정
* 상태가 exiting에서 exited로 변경되는 동안  
  -> "fade-slide**-exit** fade-slide**-exit-active**"로 결정
* 상태가 exited된 직후  
  -> "fade**-exit-done**"으로 결정

즉, timeout에 설정한 시간동안 **"-enter -enter-active"**, **"-exit -exit-active"**로 className 값이 유지되는 것을 이용하여 CSS transition, animation 프로퍼티를 적용하여 전환효과를 사용할 수 있습니다.

classNames 어트리뷰트 값에는 문자열이 아닌 **객체를 전달할 수도 있습니다**. 이때 전달되는 객체의 구조는 아래와 같습니다.

<CSSTransition

,,,

classNames = {{

enter: 'enter일 때 적용될 클래스 이름',

enterActive: 'enter-acitve일 때 적용될 클래스 이름',

enterDone: 'enter-done일 때 적용도리 클래스 이름',

exit: 'exit일 때 적용될 클래스이름',

exitActive: 'exit-acitve일 때 적용될 클래스 이름',

exitDone: 'exit-done일 때 적용될 클래스 이름'

}}>

,,,

</CSSTransition>

이때 작성한 프로퍼티 값은 prefix로 사용되는 것이 아니라 **작성된 값 그대로 적용되어 래핑된 엘리먼트의 클래스로 추가**됩니다.

**TransitionGroup Component**

"react-transition-group/TransitionGroup" 패키지에서 TransitionGroup 컴포넌트를 import할 수 있습니다.

TransitionGroup 컴포넌트는 **동적인 리스트**에 사용되는 컴포넌트로 사용됩니다. 즉, **ol이나 ul, Routes 컴포넌트가 갖고 있는 각 리스트들에 전환 효과를 적용하기 위해서 상위 컴포넌트로서 사용**됩니다.

TransitionGroup는 **기본적으로 div 태그로 돔에 렌더링**되며, 이를 변경하고자 하는 경우 **"component" 어트리뷰트 값으로 "ul"이나 "ol"을 작성**한다면 TransitionGroup 컴포넌트는 ul이나 ol 태그로 돔에 렌더링되기 때문에 Content 영역에는 li 요소, 즉 리스트를 작성할 수 있습니다.

<TransitionGroup component="ul">

,,,,

</TransitionGroup>

TransitionGroup 컴포넌트는 리스트에 전환 효과를 적용하기 위해서 사용 되므로 **반드시 Transition이나 CSSTransition 컴포넌트와 함께 사용**해야 합니다.

TransitionGroup 컴포넌트를 사용함으로써 자식으로 작성된 Transition이나 CSSTransition 컴포넌트에 일일이 **in 어트리뷰트를 작성하지 않아도 됩니다.**

TransitionGroup 컴포넌트의 **Content 영역에 작성된 리스트를 자동으로 확인**하여 **제거된 Transition이나 CSSTransition 컴포넌트 in 어트리뷰트 값을 자동으로 설정**해줍니다.

import TransitionGroup from 'react-transition-group/TransitionGroup';

import CSSTransition from 'react-transition-group/CSSTransition';

const ListItems = (props) => {

return props.items.map((item, index) => {

return (

// in 어트리뷰트 작성하지 않음

<CSSTransition classNames="fade" key={index} timeout={300}>

<li className="Listitem" onClick={() => props.removeHandler(index)}>

{item}

</li>

</CSSTransition>

);

};

}

const List = () => {

return (

<TransitionGroup component="ul" className="List">

{listItems}

</TransitionGroup>

);

};