Chapter 9. 조건부 렌더링

교재: 처음만난 리액트 (저자: 이인제, 한빛출판사)

Contents

- CHAPTER 9: 조건부 렌더링

SECTION 9.1 조건부 렌더링이란?

◦ 조건부 렌더링(Conditional Rendering)

- 어떠한 조건에 따라서 렌더링이 달라지는 것
- 조건문의 결과 true/false에 따라서 렌더링을 다르게 하는 것

  ex. 조건문의 값이 true이면 버튼을 보여주고 false이면 버튼을 가림

```
function UserGreeting(props) {
  return <h1>다시 오셨군요!</h1>;
}

function GuestGreeting(props) {
  return <h1>로그인을 해주세요.</h1>
}
```

```
function Greeting(props) {
  const isLoggedIn = props.isLoggedIn;
  if (isLoggedIn) {
    return <UserGreeting />;
  }
  return <GuestGreeting />;
}
```

- Greeting 컴포넌트 – props로 들어오는 isLoggedIn의 값에 따라서 화면에 출력되는 결과가 달라짐 (조건부 렌더링)

SECTION 9.1 조건부 렌더링이란?

자바스크립트의 Truthy와 Falsy

- truthy - true는 아니지만 true로 여겨지는 값
  - { } (empty object)
  - [ ] (empty array)
  - 42 (number, not zero)
  - "0", "false" (string, not empty)

- flasy – flase는 아니지만 flase로 여겨지는 값
  - 0, -0 (zero, minus zero)
  - 0n (Bigint zero)
  - ' ', " ", ` ` (empty string)
  - null
  - undefined
  - NaN (not a number)

SECTION 9.2 엘리먼트 변수

◦ 엘리먼트 변수(Element Variables)

- 조건부 렌더링을 할 때, 렌더링해야 될 컴포넌트를 변수처럼 다루고 싶을 때 사용
- 컴포넌트(리액트 엘리먼트)를 변수처럼 저장해서 사용하는 방법

```
let button;

if (isLoggedIn) {
  button = <LogoutButton onClick={handleLogoutClick} />;
} else {
  button = <LoginButton onClick={handleLoginClick} />;
}
```

SECTION 9.2 엘리먼트 변수

```
function LoginButton(props) {
  return (
    <button onClick={props.onClick}>
      로그인
    </button>
  );
}

function LogoutButton(props) {
  return (
    <button onClick={props.onClick}>
      로그아웃
    </button>
  );
}
```

```
function LoginControl(props) {
  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const handleLoginClick = () => {
    setIsLoggedIn(true);
  }

  const handleLogoutClick = () => {
    setIsLoggedIn(false);
  }

  let button;
  if (isLoggedIn) {
    button = <LogoutButton onClick={handleLogoutClick} />;
  } else {
    button = <LoginButton onClick={handleLoginClick} />;
  }

  return  (
    <div>
      <Greeting isLoggedIn={isLoggedIn} />
      {button}
    </div>
  )
}
```

SECTION 9.3 인라인 조건

◦ 인라인 조건(Inline Conditions)

- 조건문을 코드 안에 직접 집어넣는 것
- 인라인 If, 인라인 If-Else

◦ 인라인 If

- 조건에 따라 엘리먼트를 보여주거나 감출 때 사용
- If문과 동일한 효과를 내는 논리곱 연산자(&&) 사용 (AND 연산자)
  - 양쪽에 나오는 조건문이 모두 참인 경우에만 전체 결과가 true
  - 첫 번째 조건문이 true이면 두 번째 조건문 평가
  - 첫 번째 조건문이 false이면 전체 결과가 false이므로 두 번째 조건문은 평가하지 않음(단축 평가)

```
true && expression -> expression
false && expression -> false
```

SECTION 9.3 인라인 조건

인라인 If

- 조건문이 true이면 오른쪽에 나오는 엘리먼트가 결과값, false이면 false가 결과값
- && 연산자를 JSX 코드 안에 중괄호를 사용하여 직접 넣는 방법

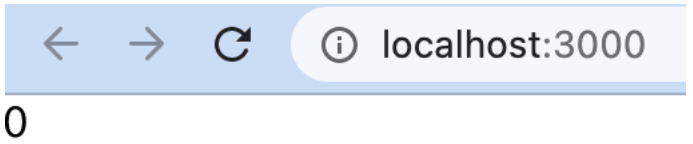
```
function Mailbox(props) {
  const unreadMessages = props.unreadMessages;

  return (
    <div>
      <h1>안녕하세요!</h1>
      {unreadMessages.length > 0 &&
        <h2>
          현재 {unreadMessages.length}개의 읽지 않은 메시지가 있습니다.
        </h2>
      }
    </div>
  );
}
```

SECTION 9.3 인라인 조건

◦ 인라인 If

- 주의 사항: 조건문에 Falsy expression을 사용하면 뒤에 나오는 expression은 평가되지 않지만 Falsy expression의 결과값이 리턴되므로 주의

```
function Counter(props) {
  const count = 0;

  return (
    <div>
      {count && <h1>현재 카운트: {count}</h1>}
    </div>
  );
}
```



count > 0 : 조건식을 사용

SECTION 9.3 인라인 조건

◦ 인라인 If-Else

- 조건문에 따라 다른 엘리먼트를 보여줄 때 사용
- 삼항 연산자 ? 사용

```
조건문 ? 참일 경우 : 거짓일 경우
```

```
function UserStatus(props) {
  return (
    <div>
      이 사용자는 현재 <b>{props.isLoggedIn ? '로그인' : '로그인하지 않은'}</b> 상태입니다.
    </div>
  )
}
```

SECTION 9.3 인라인 조건

- **인라인 If-Else**
  - 엘리먼트 변수 예제를 삼항 연산자로 변경

```
function LoginControl(props) {
  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const handleLoginClick = () => {
    setIsLoggedIn(true);
  }

  const handleLogoutClick = () => {
    setIsLoggedIn(false);
  }

  return (
    <div>
      <Greeting isLoggedIn={isLoggedIn} />
      {isLoggedIn
        ? <LogoutButton onClick={handleLogoutClick} />
        : <LoginButton onClick={handleLoginClick} />
      }
    </div>
  )
}
```

조건에 따라 각기 다른 엘리먼트를 렌더링하고 싶을 때 사용

SECTION 9.4 컴포넌트 렌더링 막기

◦ 컴포넌트를 렌더링하지 않는 방법

- 컴포넌트를 렌더링 하고 싶지 않을 때 null을 리턴
- 리액트에서는 null을 리턴하면 렌더링하지 않음

```
function WarningBanner(props) {
  if (!props.warning) {
    return null;
  }

  return  (
    <div>경고!</div>
  )
}
```

SECTION 9.4 컴포넌트 렌더링 막기

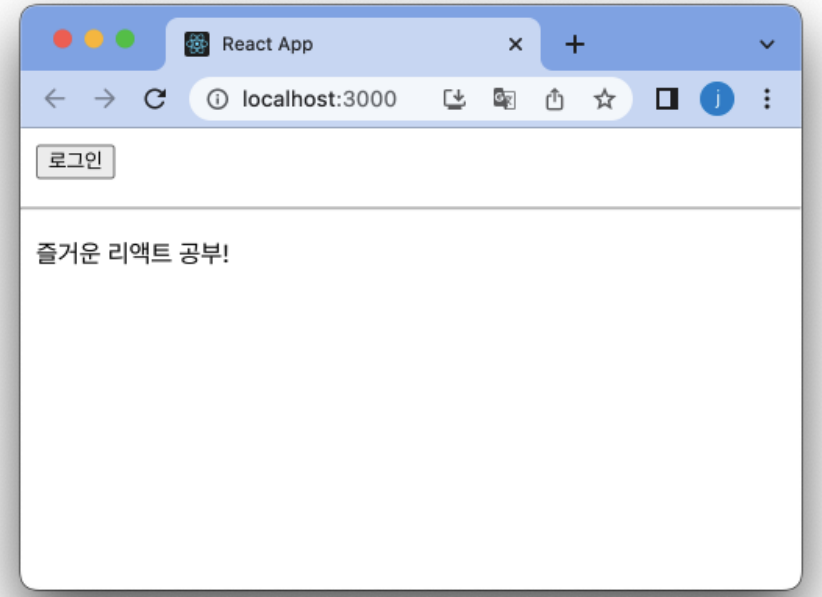
- **컴포넌트를 렌더링하지 않는 방법**

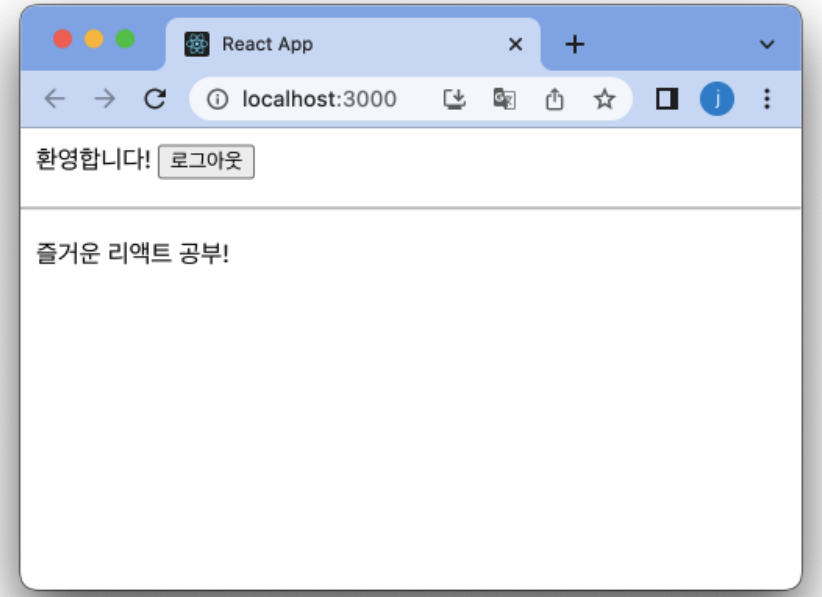
```
function MainPage(props)  {
  const [showWarning, setShowWarning] = useState(false);

  const handleToggleClick = () => {
    setShowWarning(prevShowWarning  => !prevShowWarning);
  }

  return (
    <div>
      <WarningBanner warning={showWarning} />
      <button onClick={handleToggleClick}>
        {showWarning ? '감추기' : '보이기'}
      </button>
    </div>
  )
}
```

❖클래스 컴포넌트의 render() 함수에서 null을 리턴하는 것은 컴포넌트의 생명주기 함수에 영향을 미치지 않음. (생명주기 함수가 정상적으로 호출됨)

PRACTICE 9.5 로그인 여부를 나타내는 툴바 만들기

- 사용자의 로그인 여부를 나타내는 툴바 컴포넌트 만들기
  - 조건부 렌더링을 사용하여 환영 메시지, 로그인/로그아웃 버튼을 보여주는 역할





PRACTICE 9.5 로그인 여부를 나타내는 툴바 만들기

- 툴바 컴포넌트 - Toolbar.jsx

```jsx
function Toolbar(props) {
  const { isLoggedIn, onClickLogin, onClickLogout } = props;

  return  (
    <div style={{ padding: 10 }}>
      {isLoggedIn && <span>환영합니다! </span>}

      {isLoggedIn ?
        <button onClick={onClickLogout}>로그아웃</button>
       :
        <button onClick={onClickLogin}>로그인</button>
      }
    </div>
  );
}

export default Toolbar;
```

PRACTICE 9.5 로그인 여부를 나타내는 툴바 만들기

- 랜딩페이지 컴포넌트 – LandingPage.jsx

```jsx
import React, { useState } from "react";
import Toolbar from "./Toolbar";

function LandingPage(props) {
  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const onClickLogin = () => {
    setIsLoggedIn(true);
  }

  const onClickLogout = () => {
    setIsLoggedIn(false);
  }
```

```jsx
  return (
    <div>
      <Toolbar
        isLoggedIn={isLoggedIn}
        onClickLogin={onClickLogin}
        onClickLogout={onClickLogout}
      />
      <hr />
      <div style={{ padding: 10 }}>
        즐거운 리액트 공부!
      </div>
    </div>
  );
}

export default LandingPage;
```

[요약]

- 조건부 렌더링
  - 조건에 따라 렌더링의 결과가 달라지도록 하는 것
- 엘리먼트 변수
  - 리액트 엘리먼트를 변수처럼 저장해서 사용하는 방법
- 인라인 조건
  - 인라인 If
    - 논리 연산자 &&를 사용
    - 앞에 나오는 조건문이 true일 경우에만 뒤에 나오는 엘리먼트를 렌더링
  - 인라인 If-Else
    - 삼항 연산자 ?를 사용
    - 앞에 나오는 조건문이 true면 첫 번째 항목, false면 두 번째 항목을 리턴
    - 조건에 따라 각기 다른 엘리먼트를 렌더링 하고 싶을 때 사용

[요약]

- 컴포넌트 렌더링 막기
  - 리액트에서는 null을 리턴하면 렌더링되지 않음
  - 특정 컴포넌트를 렌더링하고 싶지 않을 경우 null을 리턴하면 됨