6. 리액트 상태 관리 (실습)

Prof. Seunghyun Park (sp@hansung.ac.kr)

Division of Computer Engineering

한성대학교 HANSUNG UNIVERSITY

학습 목표: 6. 리액트 상태 관리

https://create-react-app.dev/docs/getting-started

- **실습1-1: 프로젝트 생성**

  - create-react-app를 활용하여 빈 프로젝트 생성

- **실습1-2: 프로젝트 생성 후 빌드**

  - 프로젝트 생성: 예제 코드 활용 (famous-location-02)

  - 컴포넌트 구조 확인

- **실습2: 별점 프로젝트**

  - 프로젝트 생성

  - 별 1개 렌더링 (Star 컴포넌트)

  - 별 5개 렌더링 (StarRating 컴포넌트)

  - 상태변수 selectedStars 활용: useState() 훅

  - click 이벤트 처리, 상태변경 함수 setSelectedStars()

- **실습3: 별점 프로젝트 (계속)**

  - 데이터를 파일로 입력 받아 화면에 렌더링 (Color 컴포넌트)

  - 데이터를 배열로 확장, 화면에 렌더링 (ColorList 컴포넌트)

  - 아이템 삭제 (click 이벤트, colors 상태, onRemoveColor)

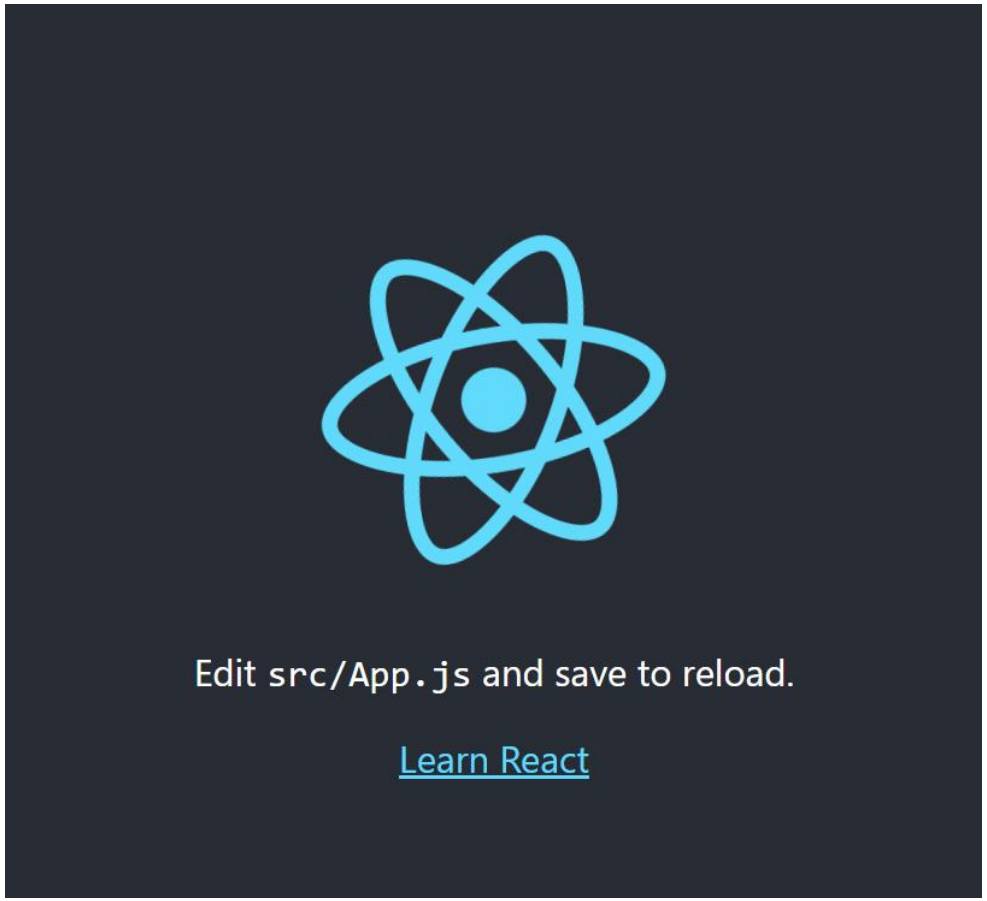
  - 별점변경 (click 이벤트, colors 상태, onRateColor)

실습1-1: 빈 프로젝트 생성 – create-react-app 활용

새로운 프로젝트 생성: `famous-location-01`

```
> create-react-app famous-location-01
> cd famous-location-01
> npm start
```

※ 실행 결과는 `ex-01-1`과 비교

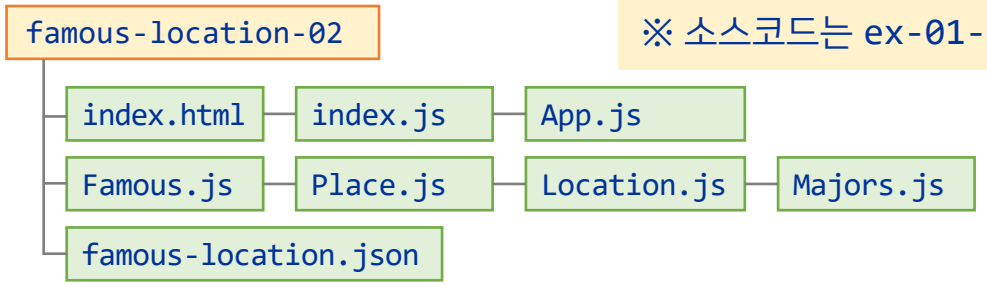


실습1-2: 빈 프로젝트 생성하여 famous-location-02 앱 빌드

1. 새로운 프로젝트 생성: famous-location-02

```
> mkdir famous-location-02
> copy files from ex-01-2/. to famous-location-02/.
```

※ 소스코드는 ex-01-2 활용



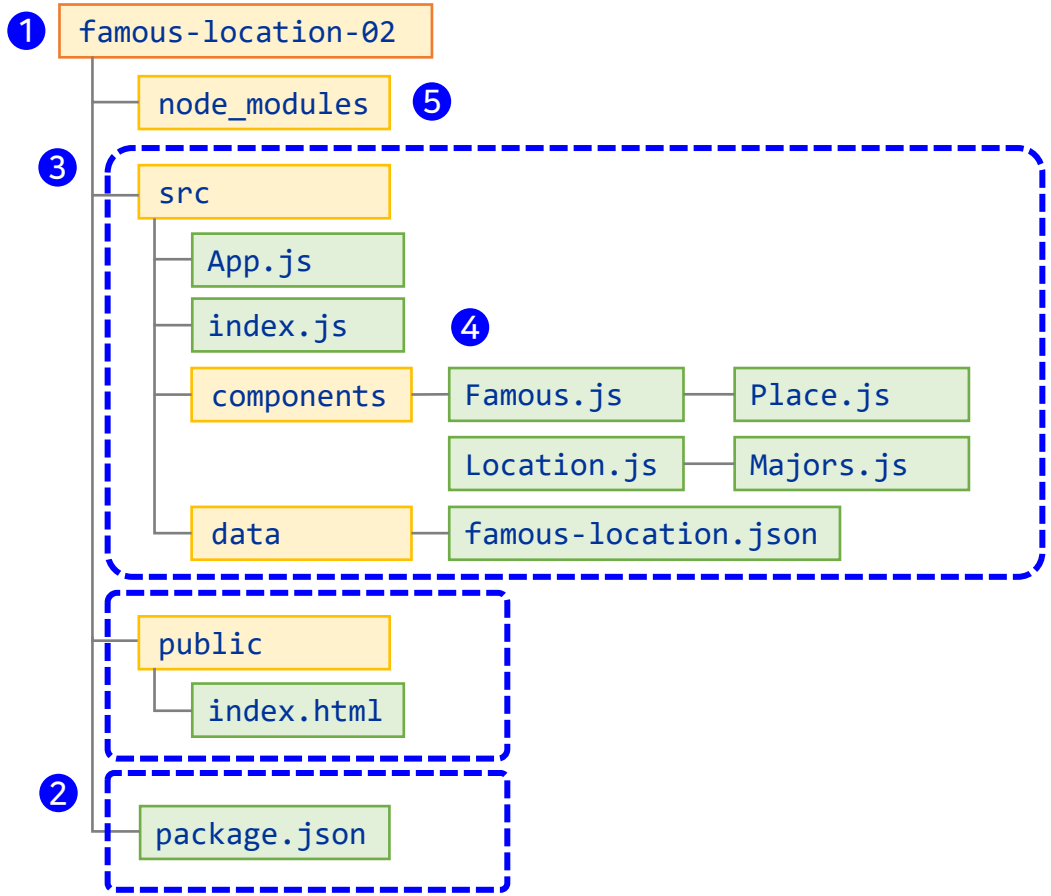
2. package.json 생성 및 수정

```
> npm init
> vi package.json
```

```
...
  "scripts": {
     "start": "react-scripts start"
  },
...
```

3. 프로젝트 폴더 구조 생성 및 모듈 배치

- ./src, ./src/data, ./src/components, ./public



4. 모든 js files 대상으로 모듈 export and import

5. 기본 모듈 설치: react, react-dom, react-scripts

실습1-2: 빈 프로젝트 생성하여 famous-location-02 앱 빌드 (계속)

```
/* famous-location-02/src/index.js */
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';

const root =
  ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <App />
);
```

```
/* famous-location-02/src/App.js */
import data from "./data/famous-location.json";
import Famous from "./components/Famous";

function App() {
  return (
    <Famous famous={data} title="Famous places" />
  );
}

export default App;
```

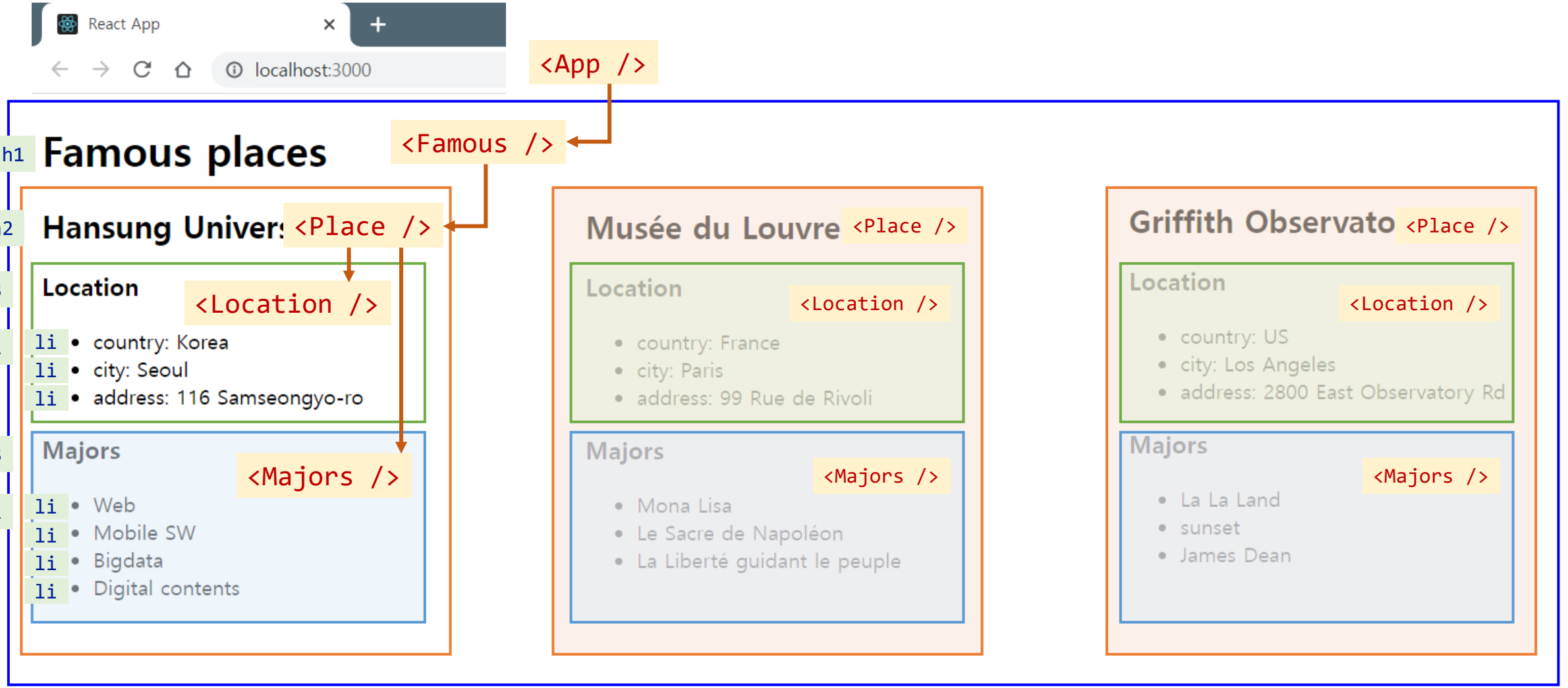
```
/* famous-location-02/src/components/Famous.js */
import Place from "./Place";
const Famous = props => (
    <Place ... />
);
export default Famous;
```

```
/* famous-location-02/src/components/Place.js */
import Location from "./Location";
import Majors from "./Majors";
const Place = ({ name, locations, majors }) => (
    <Location ... />
    <Majors ... />
);
export default Place;
```

```
/* famous-location-02/src/components/Location.js */
const Location = ({ locations }) => (
  <div>...</div>
);
export default Location;
```

```
/* famous-location-02/src/components/Majors.js */
const Majors = ({ majors }) => (
  <>...</>
);
export default Majors;
```

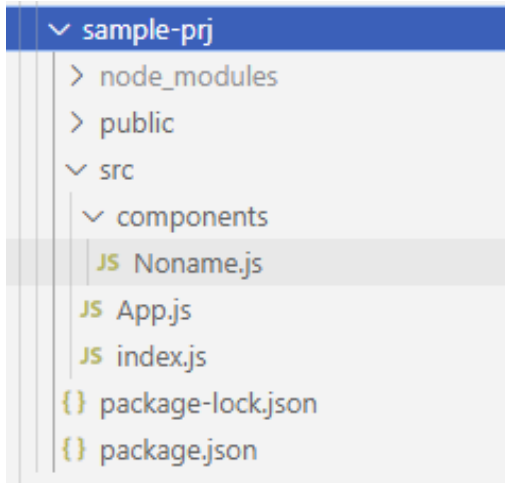
실습1-2: 컴포넌트 구조 확인

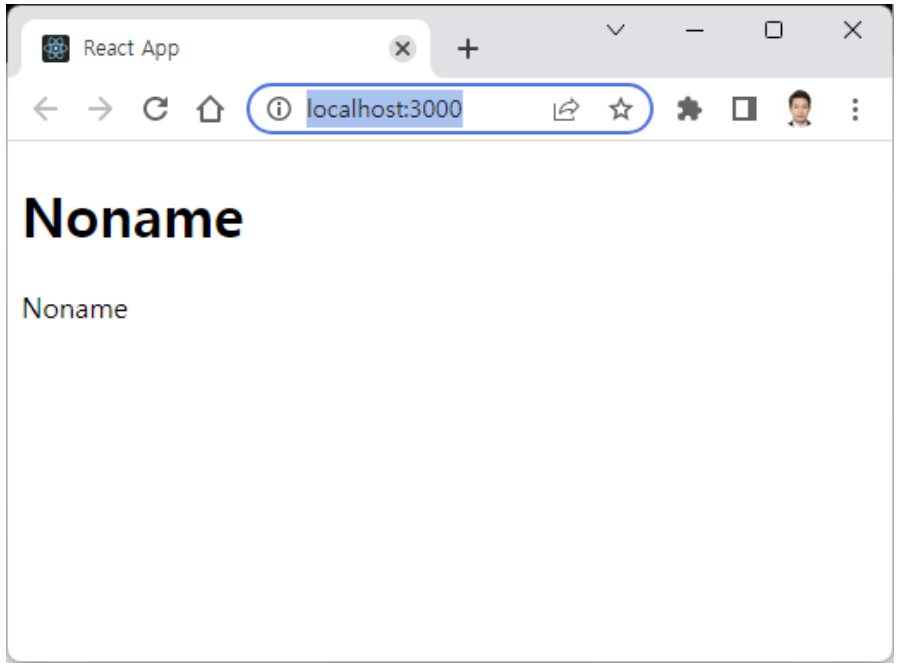


실습2-1: 별점 프로젝트 (단계1)

단계 1: sample-prj의 소스를 활용하여 ex-02-1 프로젝트 생성

```
> mkdir ex-02-1
> copy files from sample-prj/. to ex-02-1/.
> cd ex-02-1
> npm init
> npm install react react-dom react-scripts react-icons
> npm start
```





실습2-1: 별점 프로젝트 (단계2)

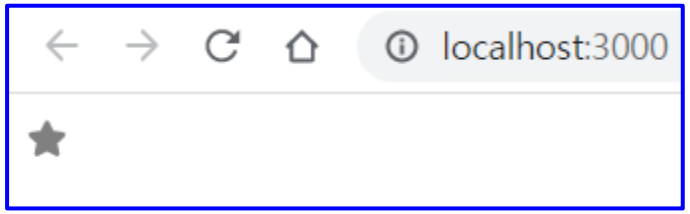
단계 2: Star 컴포넌트에서 FaStar 라이브러리를 이용하여 별 1개 렌더링

- App 컴포넌트에서 Star 컴포넌트로 전달하는 selected 프로퍼티가 true이면 붉은 색, false이면 회색의 별을 렌더링
- App 컴포넌트에서 호출하는 Noname 컴포넌트는 더 이상 사용하지 않으므로, 이 컴포넌트 이름을 변경하여 사용

```
/* ./src/components/Star.js */

// import module FaStar from react-icons/fa
import { FaStar } from "react-icons/fa";

const Star = ({ selected = false }) =>
  <FaStar color={selected ? "red" : "grey"} />

// export module
export default Star;
```

react-icons 설치 필요

```
/* ./src/App.js */

import Star from "./components/Star";

function App() {
  return (
    <Star selected={false} />
  );
}
```



실습2-1: 별점 프로젝트 (단계3)

단계 3: StarRating 컴포넌트를 생성하고, Star 컴포넌트를 활용하여 별 5개 그리기

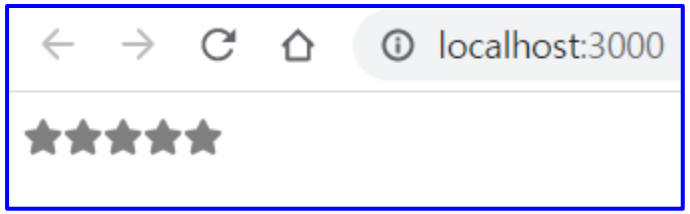
- App 컴포넌트에서 Star 컴포넌트로 전달하는 selected 프로퍼티가 true이면 붉은 색, false이면 회색의 별을 렌더링
- App 컴포넌트에서 호출하는 Noname 컴포넌트는 더 이상 사용하지 않으므로, 이 컴포넌트 이름을 변경하여 사용

```
/* ./src/App.js */
import StarRating from "./components/StarRating";

function App() {
  return (
    <StarRating totalStars={5}/>
  );
}

// export module
export default App;
```

새로운 컴포넌트 렌더링

```
/* ./src/components/StarRating.js */
import Star from "./Star";
const createArray = length => [...Array(length)];

const StarRating = ({ totalStars = 5 }) =>
  createArray(totalStars).map((n, i) =>
    <Star key={i} />
  )

export default StarRating;
```

컴포넌트 생성



실습2-1: 별점 프로젝트 (단계4)

단계 4: StarRating 컴포넌트에서, 상태변수 selectedStars를 생성하고 초기 값 지정

- 지정된 수 만큼만 붉은 별, 나머지는 회색 별을 그리고
- 상태변수 값을 화면에 렌더링

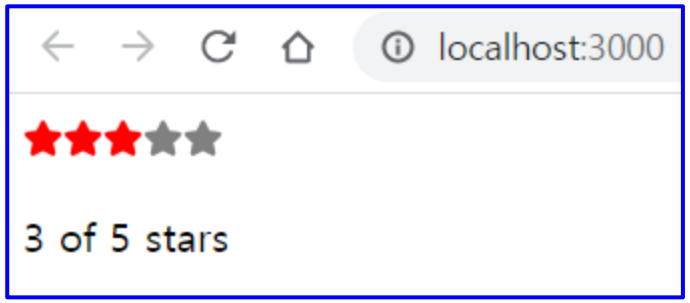
```
/* ./src/components/StarRating.js */

import { useState } from "react";
import Star from "./Star";
const createArray = length => [...Array(length)];

const StarRating = ({ totalStars = 5 }) => {
  const [selectedStars] = useState(3);
```

상태 변수 지정

```
  return (
    <>
      {createArray(totalStars).map((n, i) =>
        <Star
          key={i}
          selected={selectedStars > i}
        />)}
      <p>{selectedStars} of {totalStars} stars</p>
    </>
  );
}

export default StarRating;
```

선택된 별의 true/false 결정

상태 변수 값 화면에 렌더링



실습2-1: 별점 프로젝트 (단계5)

단계 5: click 이벤트 처리: StarRating 컴포넌트에서, 상태 변경함수 setSelectedStars 등록

- FaStar 컴포넌트에 onClick 이벤트 리스너 등록하고, click 이벤트 처리 → 이벤트는 StarRating 컴포넌트에서 상태변경 함수 호출
- 상태 변경함수 setSelectedStars가 선택된 별 만큼 red로 변경함에 따라 다시 렌더링
  . 5개의 Star 중 하나를 클릭하면 선택된 Star가 몇 번째 Star인지 계산 (index+1)해서 상태변수 seletecStars의 값을 업데이트

```
/* ./src/components/StarRating.js */

import { useState } from "react";
...

const StarRating = ({ totalStars = 5 }) => {
  const [selectedStars, setSelectedStars] =useState(3);

  return (
    <>
      {createArray(totalStars).map((n, i) =>
        <Star key={i} selected={selectedStars > i}
          onSelect={() => setSelectedStars(i+1)} />)}
      <p>{selectedStars} of {totalStars} stars</p>
    </>
  );
}
...
```

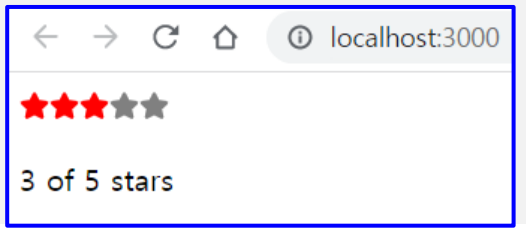
상태 변경함수

이벤트 처리

```
/* ./src/components/Star.js */

import { FaStar } from "react-icons/fa";

const Star = ({
  selected = false,
  onSelect = f => f
}) =>
  <FaStar
    color={selected ? "red" : "grey"}
    onClick={onSelect}
  />

export default Star;
```

이벤트 리스너

클릭: FaStar 컴포넌트에서 발생
별점 변경: StarRating 컴포넌트에서 처리
> 값 변경 후 다시 렌더링



실습2-2: 별점 프로젝트 (단계1)

단계 1: 데이터를 json 파일로 import 하고, 데이터의 구성에 따라 정보를 렌더링하는 Color 컴포넌트 생성

- 화면에 표시할 정보는 title, color, 별점 정보 (실습2: StarRating + Star 컴포넌트)

```
/* ./src/App.js */
import Color from "./components/Color";
import colorData from "./data/color-data.json";

function App() {
  return (
    <Color colors={colorData} />
  );
}

export default App;
```

새로운 컴포넌트 렌더링

```
{
  "id": "0175d1f0-a8c6-41bf-8d02-df5734d829a4",
  "title": "ocean at dusk",
  "color": "#00c4e2",
  "rating": 5
}
```

데이터 파일

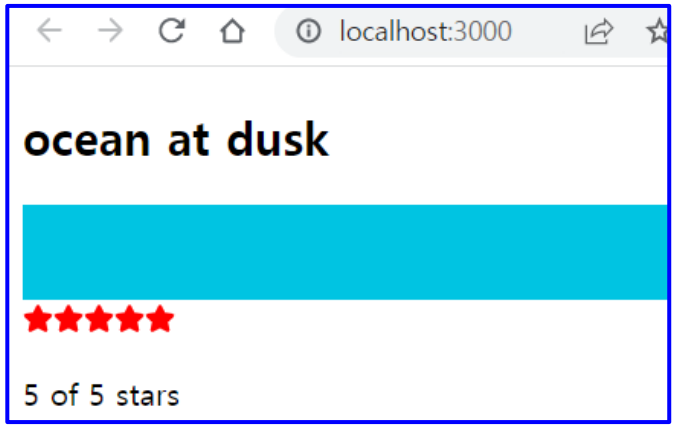
```
/* ./src/components/Color.js */

import StarRating from "./StarRating";

const Color = ({ colors }) =>
  <section>
    <h1>{colors.title}</h1>
    <div style={{ backgroundColor: colors.color, height: 50 }} />
    <StarRating selectedStars={colors.rating} />
  </section>

export default Color;
```

컴포넌트 생성

실습2-2: 별점 프로젝트 (단계1, 계속)

```
/* ./src/components/StarRating.js */

// import { useState } from "react";
import Star from "./Star";
const createArray = length => [...Array(length)];

const StarRating = ({ totalStars = 5, selectedStars }) => {
  // const [selectedStars, setSelectedStars] = useState(3);

  return (
    <>
      {createArray(totalStars).map((n, i) =>
        <Star
          key={i} selected={selectedStars > i}
          // onSelect={() => setSelectedStars(i+1)}
        />)}
      <p>{selectedStars} of {totalStars} stars</p>
    </>
  );
}
export default StarRating;
```

별점 클릭 이벤트 처리는 임시로 주석처리



실습2-2: 별점 프로젝트 (단계2)

단계 2: 3개의 데이터를 array를 json 파일로 import 하고, 데이터의 구성을 표현하는 ColorList 컴포넌트 생성

```
/* ./src/App.js */
import ColorList from "./components/ColorList";
import colorData from "./data/color-data.json";

function App() {
  return (
    <ColorList colors={colorData} />
  );
}

export default App;
```

새로운 컴포넌트 렌더링

```
[
  { "id": "...", "title": "ocean at dusk",
    "color": "#00c4e2", "rating": 5 },
  { "id": "...", "title": "lawn",
    "color": "#26ac56", "rating": 3 },
  { "id": "...", "title": "bright red",
    "color": "#ff0000", "rating": 0 }
]
```

데이터 확장

```
/* ./src/components/ColorList.js */
import Color from "./Color";

const ColorList = ({ colors = [] }) =>
  <div>
    {colors.map( color  =>
      <Color key={color.id} {...color} />)}
  </div>
export default ColorList;
```

컴포넌트 생성

```
/* ./src/components/Color.js */
import StarRating from "./StarRating";

const Color = ({ title, color, rating }) =>
  <section>
    <h1>{title}</h1>
    <div style={{ backgroundColor: color, height: 50 }} />
    <StarRating selectedStars={rating} />
  </section>
export default Color;
```

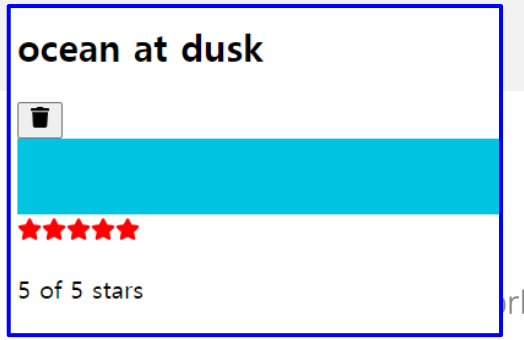
실습2-2: 별점 프로젝트 (단계3)

단계 3: 아이템 삭제 – Color 컴포넌트 내부에 버튼을 추가하고, 버튼이 클릭되면 해당 아이템을 삭제하도록 구현

- 버튼과 이벤트 (Color 컴포넌트), 리스트 삭제 (ColorList 컴포넌트), 데이터 반영 (App 컴포넌트)

```
/* ./src/components/Color.js */
import StarRating from "./StarRating";
import { FaTrash } from "react-icons/fa";

const Color = ({ id, title, color, rating,
  onRemove = f => f }) =>
  <section>
    <h1>{title}</h1>
    <button onClick={() => onRemove(id)}>
      <FaTrash />
    </button>
    <div style={{ backgroundColor: color, height: 50 }} />
    <StarRating selectedStars={rating} />
  </section>

export default Color;
```

버튼 클릭 이벤트 리스너




```
/* ./src/components/ColorList.js */
const ColorList = ({ colors = [],
  onRemoveColor = f => f }) =>
  <div>
    {colors.map( color  =>
      <Color key={color.id} {...color}
        onRemove={onRemoveColor} />)}
  </div>
```

```
/* ./src/App.js */
import { useState } from "react";
function App() {
  const [colors, setColors] = useState(colorData);
  return (
    <ColorList colors={colors}
      onRemoveColor={id => {
        const newColor = colors.filter(
          color => color.id !== id)
        setColors(newColor);
      }} />
  );
}
```

클릭 이벤트 처리
→ onRemoveColor()

실습2-2: 별점 프로젝트 (단계4)

단계 4: 별점변경 이벤트 처리 – FaStar 컴포넌트 클릭에 따라 별점을 변경하도록 이벤트 처리

- 버튼과 이벤트 (FaStar 컴포넌트), 값 변경 (StarRating 컴포넌트), 데이터 반영 (App 컴포넌트)

```
/* ./src/components/Star.js */

import { FaStar } from "react-icons/fa";

const Star = ({
  selected = false,
  onSelect = f => f
}) =>
  <FaStar
    color={selected ? "red" : "grey"}
    onClick={onSelect}
  />

export default Star;
```

별점 클릭 이벤트 리스너

```
/* ./src/components/StarRating.js */
import Star from "./Star";
const createArray = length => [...Array(length)];

const StarRating = ({ totalStars = 5, selectedStars,
  onRate = f => f }) => (
    <>
      {createArray(totalStars).map((n, i) =>
        <Star key={i} selected={selectedStars > i}
          onSelect={() => onRate(i+1)}
        />)}
      <p>{selectedStars} of {totalStars} stars</p>
    </>
  );

export default StarRating;
```

이벤트 처리를 위한 메서드 전달

실습2-2: 별점 프로젝트 (단계4, 계속)

```
/* ./src/components/Color.js */
import StarRating from "./StarRating";
import { FaTrash } from "react-icons/fa";
const Color = ({ id, title, color, rating,
  onRemove = f => f, onRate = f => f }) =>
  <section>
    <h1>{title}</h1>
    <button onClick={() => onRemove(id)}><FaTrash />
    </button>
    <div style={{ backgroundColor: color, height: 50 }} />
    <StarRating selectedStars={rating}
      onRate={rating => onRate(id, rating)}
    />
  </section>
export default Color;
```

이벤트 처리를 위한 메서드 전달

```
const removeColor = id => {
  const newColor = colors.filter(color => color.id !== id);
  setColors(newColor);
}
const rateColor = (id, rating) => {
  const newColors = colors.map(color => color.id === id ?
    { ...color, rating } : color);
  setColors(newColors);
}
```

별점 변경 필요한 id이면 별점 변경 아니면 유지

```
/* ./src/components/ColorList.js */
import Color from "./Color";
const ColorList = ({ colors = [],
  onRemoveColor = f => f, onRateColor = f => f }) =>
  <div>
    {colors.map( color  =>
      <Color key={color.id} {...color}
        onRemove={onRemoveColor}
        onRate={onRateColor} />)}
  </div>
export default ColorList;
```

이벤트 처리를 위한 메서드 전달

```
/* ./src/App.js */
import { useState } from "react";
function App() {
  const [colors, setColors] = useState(colorData);
  const removeColor = ...;
  const rateColor = ...;
  return (
    <ColorList
      colors={colors} onRemoveColor={removeColor}
      onRateColor={rateColor} />
  );
}
export default App;
```