React Hook簡介

Yu Hsiang 2019/01/11

Hook

React官方原本是以class方式進行網站開發，而在近期逐漸轉向利用function進行開發，Hook讓使用者可以只利用**官方提供函式**去控制整個網頁的資料呈現。

參考：https://reactjs.org/docs/hooks-intro.html

前置作業 - 安裝

1.建立React專案

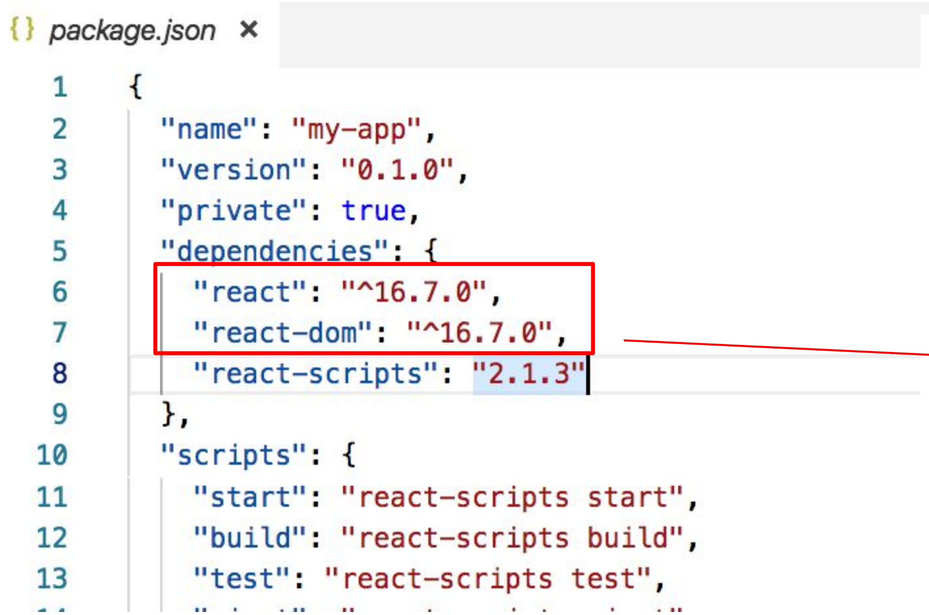
```
npx create-react-app my-app
```

網址 :

https://facebook.github.io/create-react-app/

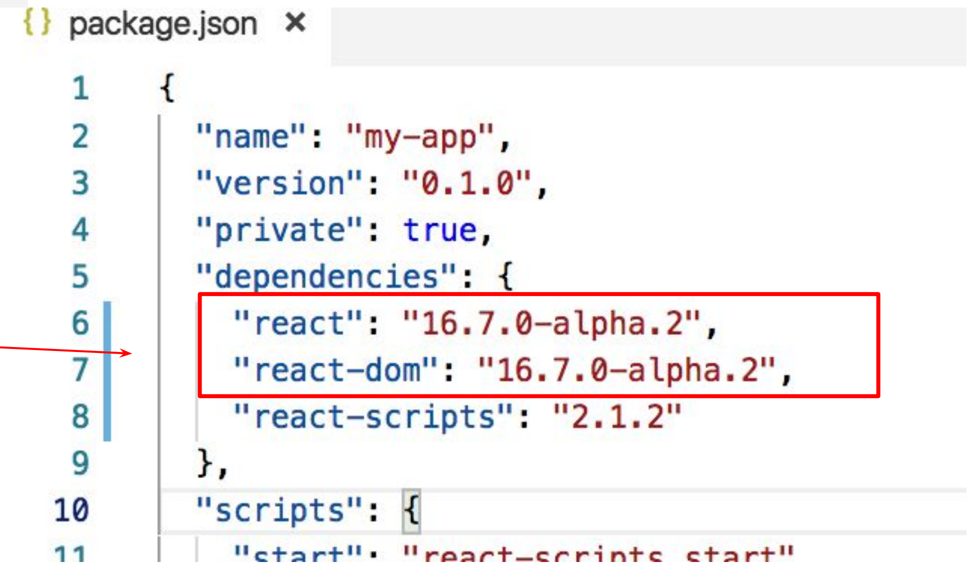
前置作業 - 改為支援Hook版本

2.更改為Hook支援版本 my-app/package.json



package.json

```
{
  "name": "my-app",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "^16.7.0",
    "react-dom": "^16.7.0",
    "react-scripts": "2.1.3"
  },
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
```



package.json

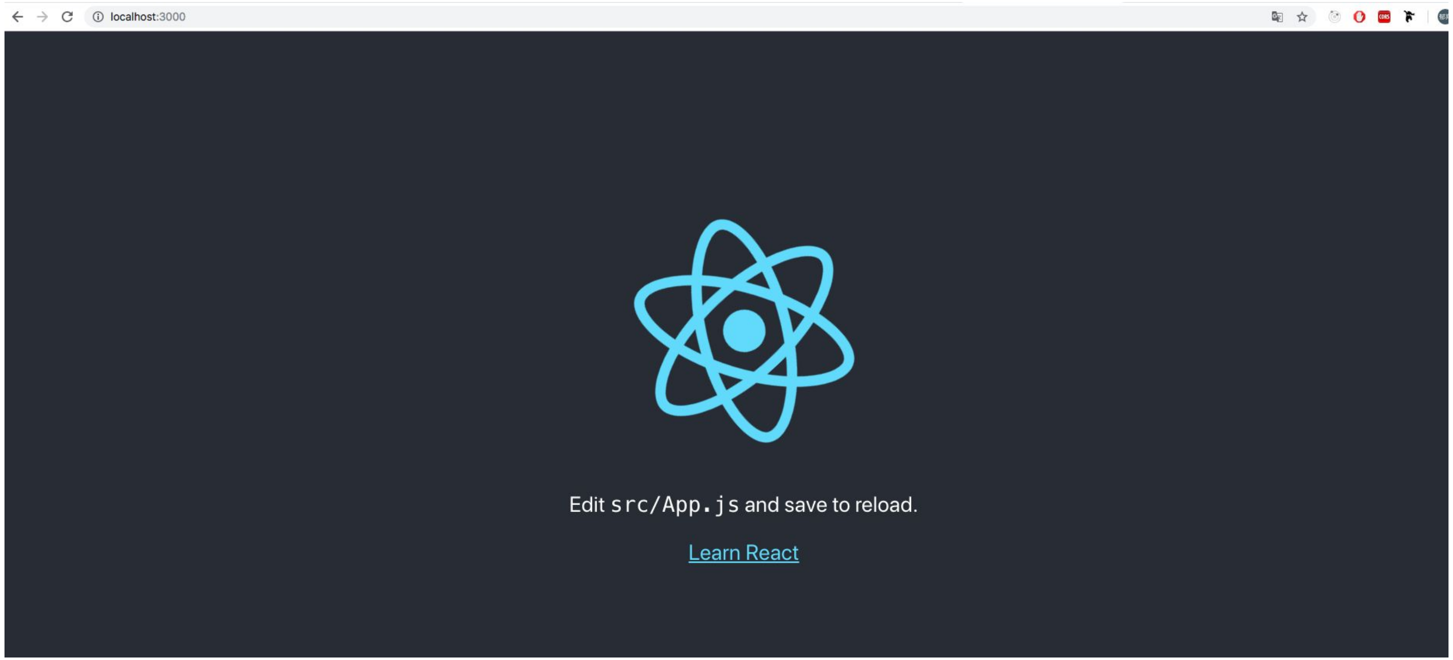
```
{
  "name": "my-app",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "react": "16.7.0-alpha.2",
    "react-dom": "16.7.0-alpha.2",
    "react-scripts": "2.1.2"
  },
  "scripts": {
    "start": "react-scripts start"
```

前置作業 - 安裝並運行

重新安裝:

```
npm install
```

```
npm start
```



Edit `src/App.js` and save to reload.

Learn React

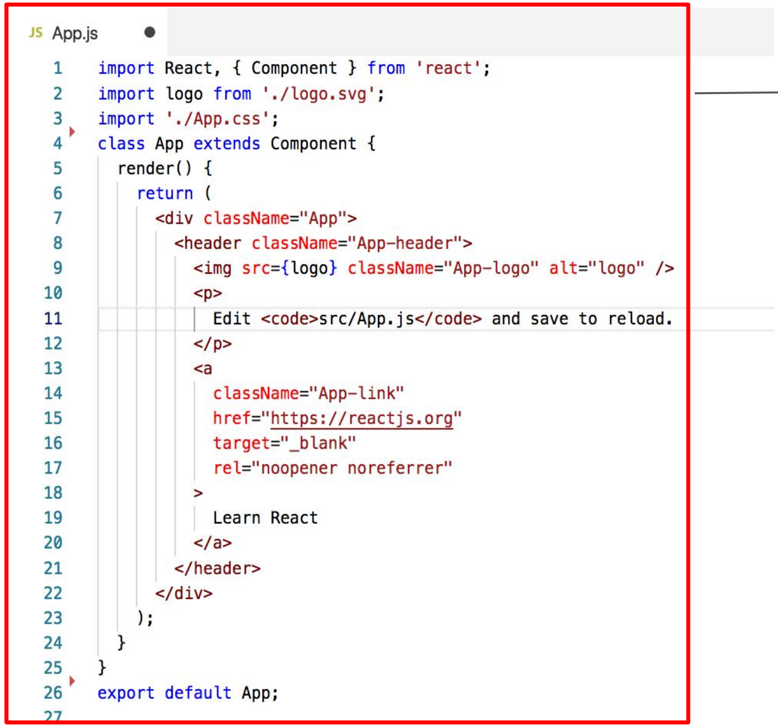
利用Hook撰寫頁面

以下範例參考 https://reactjs.org/docs/hooks-state.html

撰寫第一個頁面

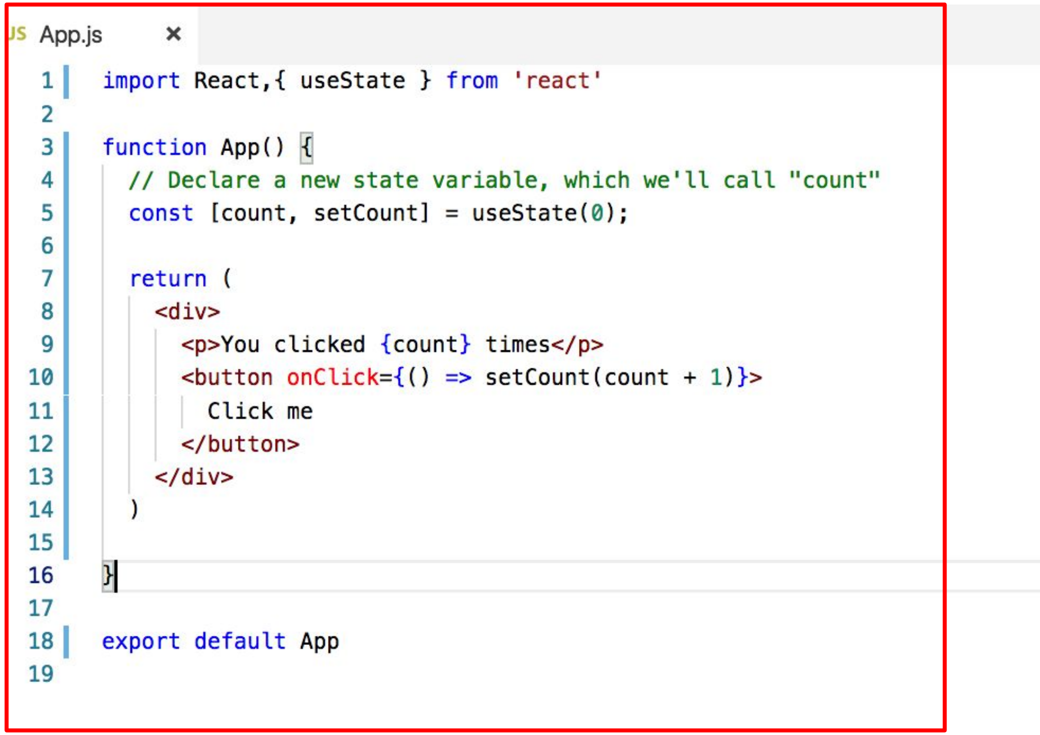
程式碼: http://codepad.org/4qh5qih1

編輯 my-app/src/App.js為新程式碼，並且儲存



App.js ●

```
import React, { Component } from 'react';
import logo from './logo.svg';
import './App.css';
class App extends Component {
  render() {
    return (
      <div className="App">
        <header className="App-header">
          <img src={logo} className="App-logo" alt="logo" />
          <p>
            Edit <code>src/App.js</code> and save to reload.
          </p>
          <a
            className="App-link"
            href="https://reactjs.org"
            target="_blank"
            rel="noopener noreferrer"
          >
            Learn React
          </a>
        </header>
      </div>
    );
  }
}
export default App;
```

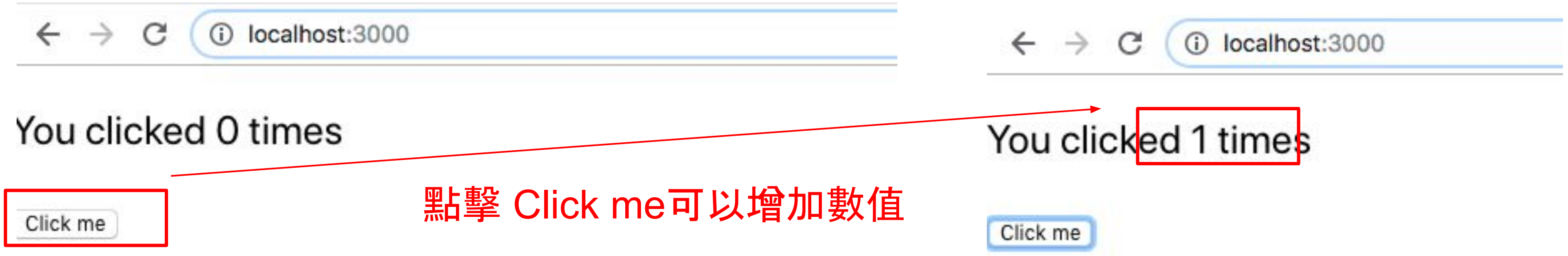


App.js ×

```
import React,{ useState } from 'react'

function App() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  )

}

export default App

```

回到網站，網頁被更改了

http://localhost:3000/

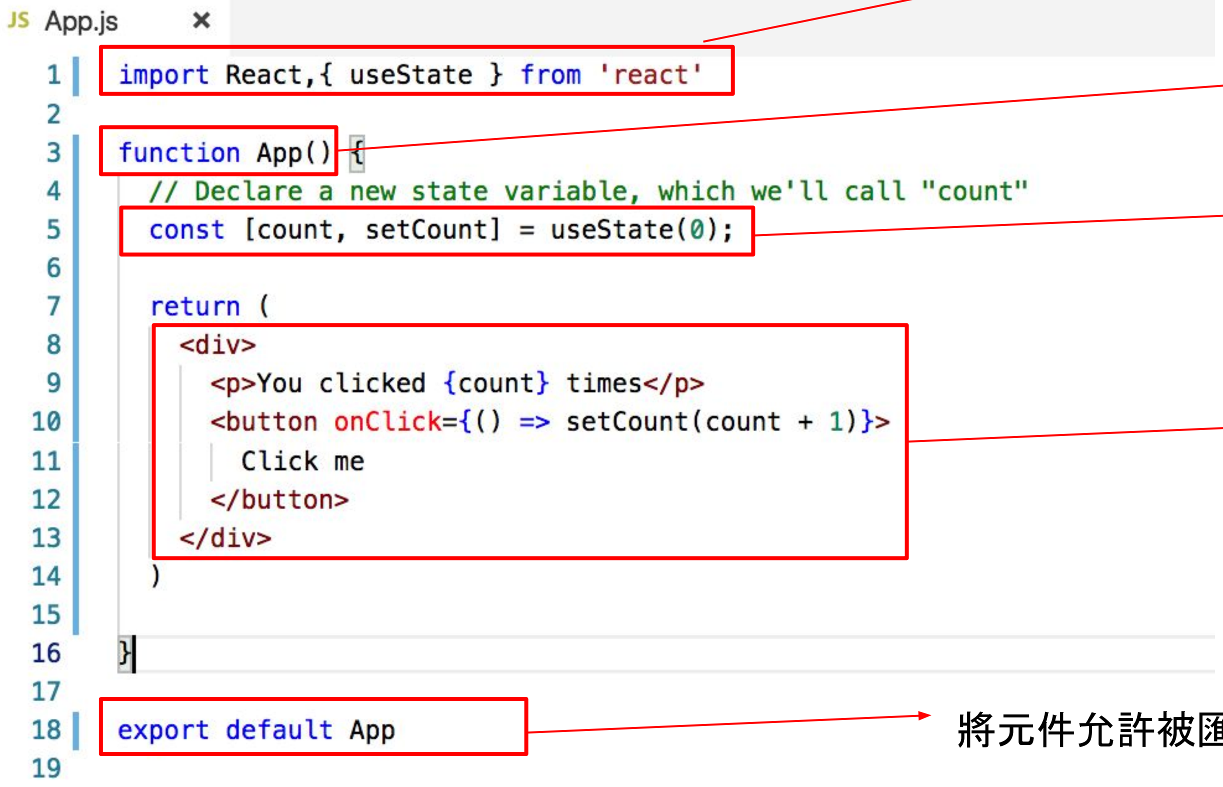


You clicked 0 times

點擊 Click me可以增加數值

You clicked 1 times

程式碼檢視


App.js

```
import React,{ useState } from 'react'

function App() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  )

}

export default App

```

1.用import匯入React套件
類似C#的using

2.元件名稱 App

3.變數容器
變數 count
以及 setter setCount
初始值 0

4.HTML主體
按鈕的 onClick 上綁定了
setter setCount
因此，setCount(count+1)
讓數值加 1

將元件允許被匯出

讓程式更簡潔些
ES6 arrow function

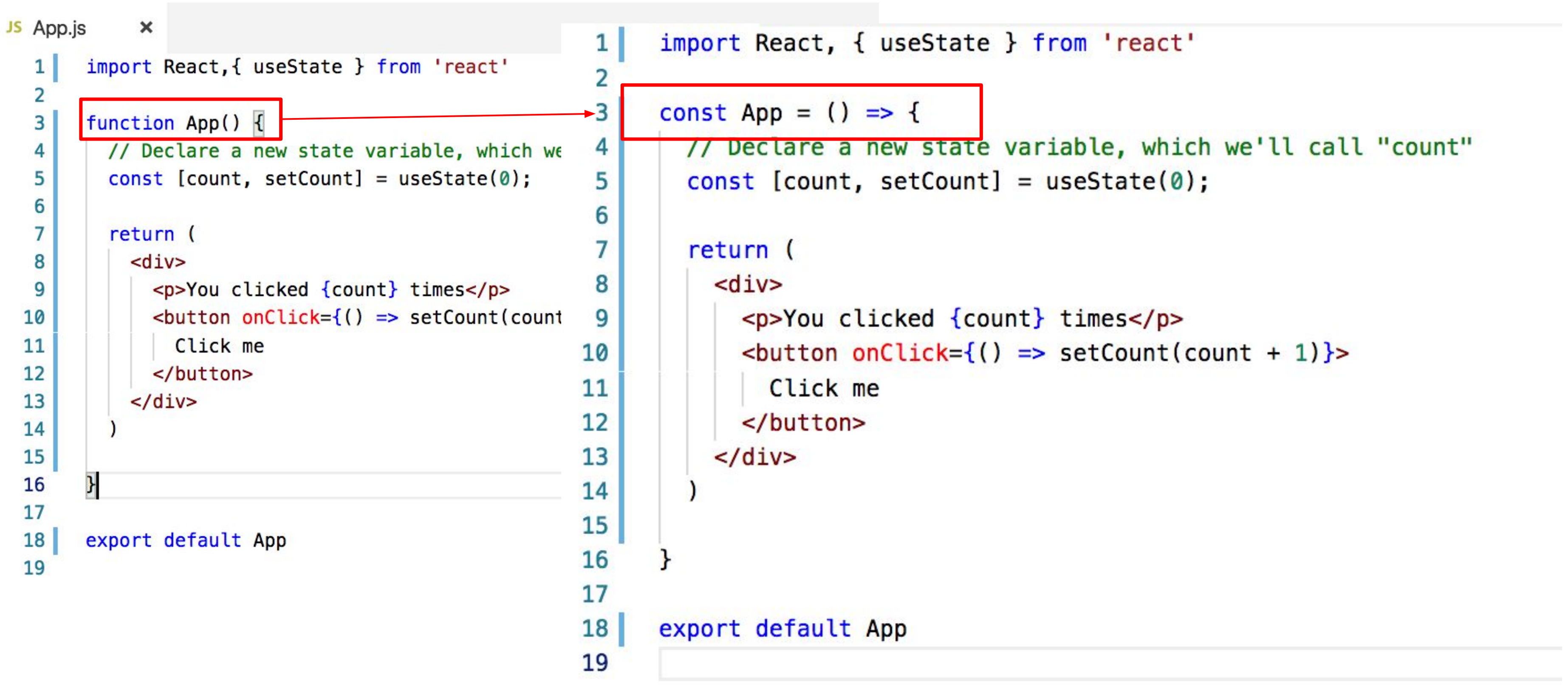
參考
：https://developer.mozilla.org/zh-TW/docs/Web/JavaScript/Reference/Functions/Arrow_functions

```
function App (){

}
```

可改為**匿名函式**的寫法

```
const App = () =>{

}
```

於是變成



```
import React,{ useState } from 'react'

function App() {
  // Declare a new state variable, which we
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count
        Click me
      </button>
    </div>
  )

}

export default App
```

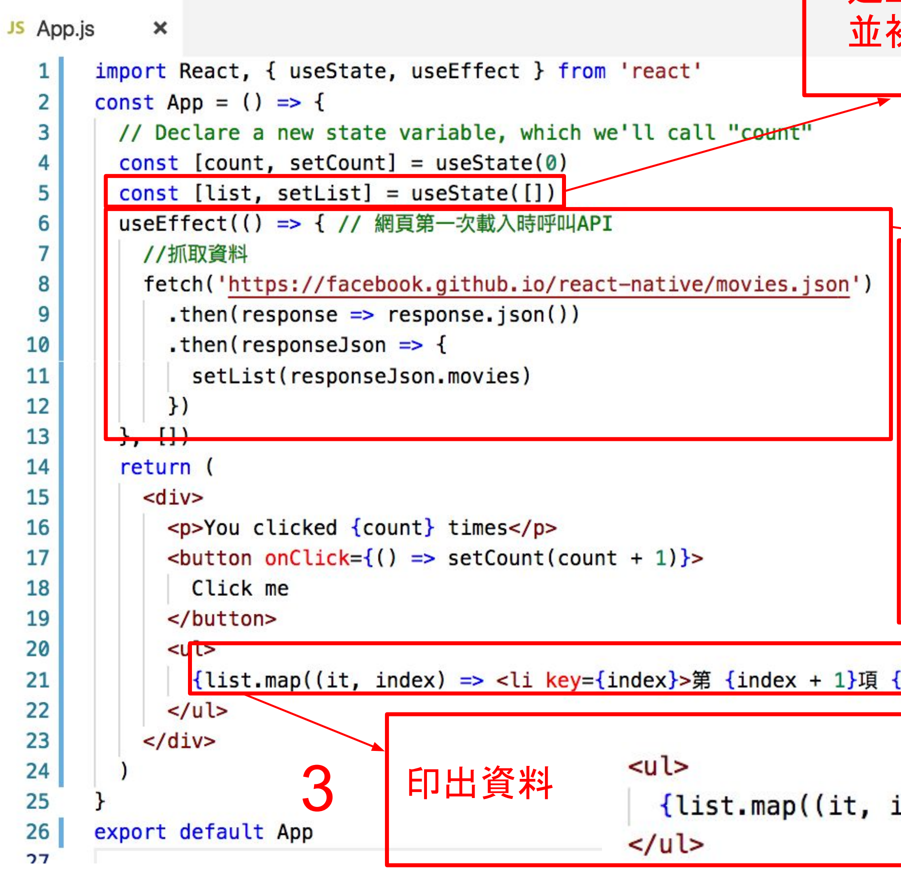
```
import React, { useState } from 'react'

const App = () => {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  )

}

export default App
```

什麼是Hook？
React 提供在函式內管理UI上資料的方法，而基本上只會常用到紅字的方法

基本：
useState - 管理UI上的資料
useEffect - 用於載入頁面時與API拿資料
useContext

進階：
useReducer
useCallback
useMemo
useRef
useImperativeMethods
useLayoutEffect

抓取API資料並顯示



App.js

```
import React, { useState, useEffect } from 'react'
const App = () => {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0)
  const [list, setList] = useState([])
  useEffect(() => { // 網頁第一次載入時呼叫API
    //抓取資料
    fetch('https://facebook.github.io/react-native/movies.json')
      .then(response => response.json())
      .then(responseJson => {
        setList(responseJson.movies)
      })
  }, [])
  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
      <ul>
        {list.map((it, index) => <li key={index}>第 {index + 1}項 {it.title}</li>)}
      </ul>
    </div>
  )
}
export default App
```

1

```
const [list, setList] = useState([])
```

建立一個list變數與setter setList
並初始化為 [ ] 空陣列

2

```
useEffect(() => { // 網頁第一次載入時呼叫API
  //抓取資料
  fetch('https://facebook.github.io/react-native/movies.json')
    .then(response => response.json())
    .then(responseJson => {
      setList(responseJson.movies)
    })
}, [])
```

載入網頁時跟API拿資料後利用setter setList設定

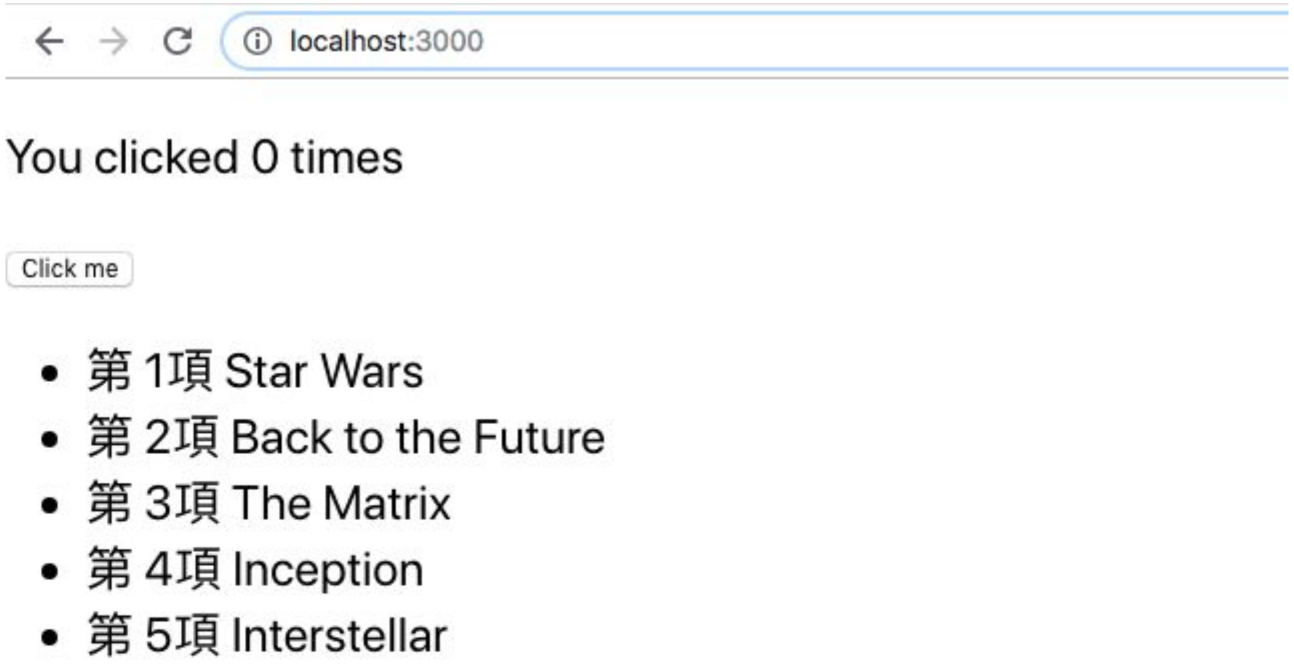
3

印出資料

```
<ul>
  {list.map((it, index) => <li key={index}>第 {index + 1}項 {it.title}</li>)}
</ul>
```

回到網站，呈現出 API資料

程式碼：http://codepad.org/gOE2whSD



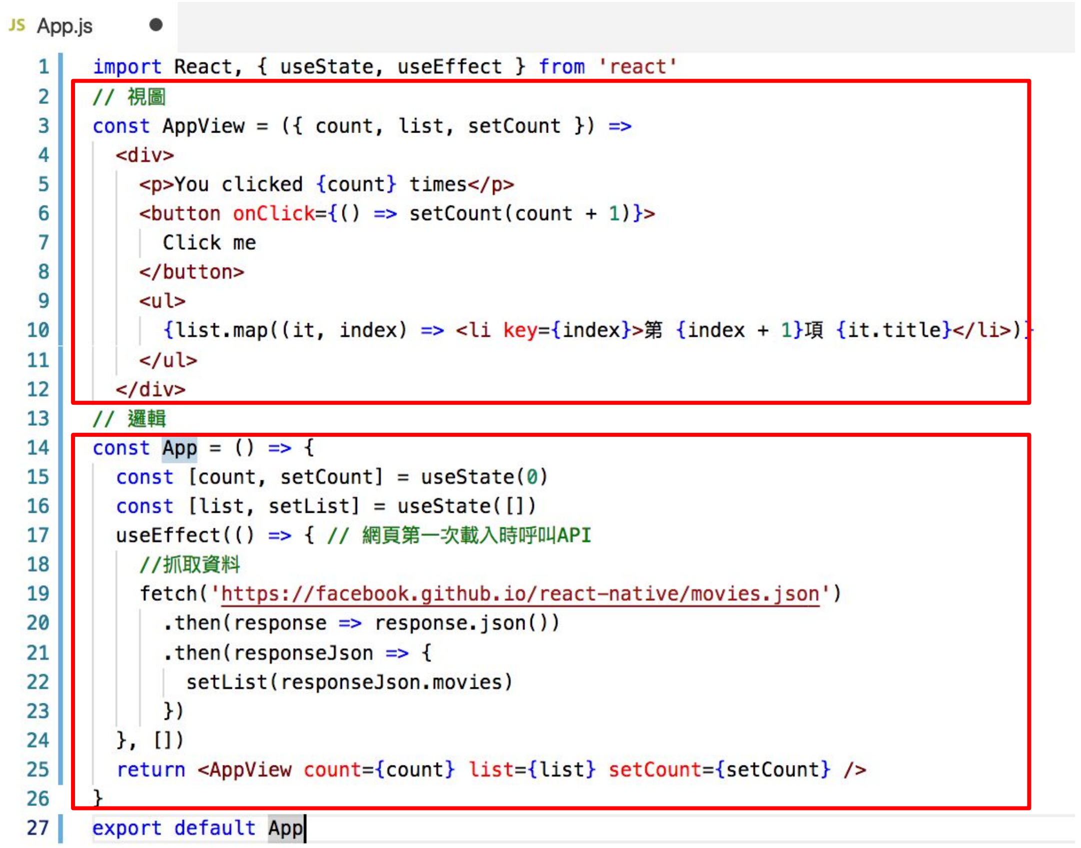
You clicked 0 times

- 第 1項 Star Wars
- 第 2項 Back to the Future
- 第 3項 The Matrix
- 第 4項 Inception
- 第 5項 Interstellar

概念上就是把資料整理成顯示的模式,
然後在視圖上印出，不直接控制DOM

說到視圖，可以將 邏輯 與 視圖 拆分

邏輯 與 視圖 拆分

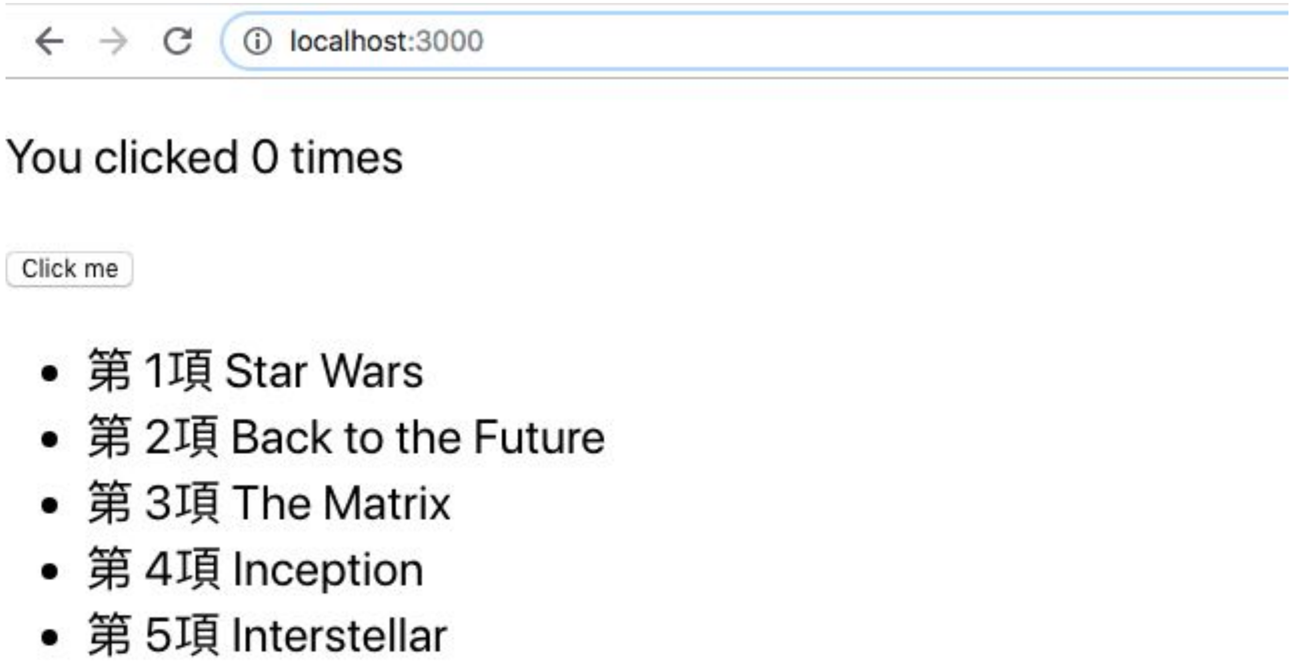


App.js

```
import React, { useState, useEffect } from 'react'
// 視圖
const AppView = ({ count, list, setCount }) =>
  <div>
    <p>You clicked {count} times</p>
    <button onClick={() => setCount(count + 1)}>
      Click me
    </button>
    <ul>
      {list.map((it, index) => <li key={index}>第 {index + 1}項 {it.title}</li>)}
    </ul>
  </div>
// 邏輯
const App = () => {
  const [count, setCount] = useState(0)
  const [list, setList] = useState([])
  useEffect(() => { // 網頁第一次載入時呼叫API
    //抓取資料
    fetch('https://facebook.github.io/react-native/movies.json')
      .then(response => response.json())
      .then(responseJson => {
        setList(responseJson.movies)
      })
  }, [])
  return <AppView count={count} list={list} setCount={setCount} />
}
export default App
```

回到網站，畫面還是一樣，但程式碼抽離了

程式碼：http://codepad.org/ND5sHxCM



localhost:3000

You clicked 0 times

Click me

- 第 1項 Star Wars
- 第 2項 Back to the Future
- 第 3項 The Matrix
- 第 4項 Inception
- 第 5項 Interstellar

就這簡單的操作，便可以控制整個頁面上的邏輯了

END.