4/01 學習 React 概念

React 是一個 javaScript 函式庫，  
常使用JSX (一種JavaScript語法擴充) 描述介面外觀。JSX語法類似:





 \* 類似這種標籤語法  
(不要求但大部分人覺得很好: 視覺輔助、顯示更有用的警告訊息)

React將其拆分為包含UI和邏輯的 Component，彼此間很少互相依賴。

-

React優點：

1. 使用component組合創建完整的UI，減少建置UI出現錯誤率
2. 元件封裝並能夠重複使用，簡化程式碼且容易維護
3. 使用JSX語法編寫UI，不需要寫太長的程式碼(create…)去建置UI
4. 簡單編寫出即時渲染介面的程式

JSX 簡介

1. JSX中嵌入Expression：（在大括號{}中寫入JavaScript expression）

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

2. Expression為function範例 :

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

自動產生的描述 一張含有 字型, 文字, 印刷術, 圖形 的圖片

自動產生的描述

3. JSX作為function的回傳：一張含有 文字, 螢幕擷取畫面, 字型, 行 的圖片

自動產生的描述

4. JSX防範注入攻擊：  
React DOM會在render前將變數轉為字串，避免XSS(跨網站指令碼) 攻擊。

一張含有 文字, 字型, 螢幕擷取畫面, 行 的圖片

自動產生的描述

5. Babel語法，將JSX編譯為呼叫React.createElement() 的程式。一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

React.createElement() 會產生類似這種物件，稱為「React element」：  
一張含有 文字, 字型, 螢幕擷取畫面, 設計 的圖片

自動產生的描述

Render element

1. 一個Root DOM node



1. Render 元素到root DOM node：

一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

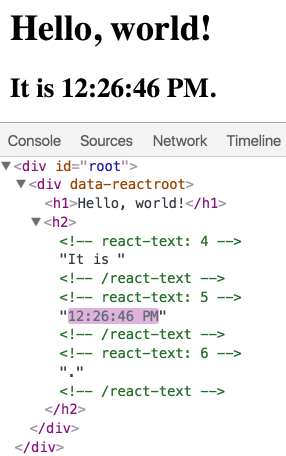
* + 1. 創建root  
    const root = ReactDOM.createRoot( document.getElementById('root') );
  + 2. 創建element  
    const element = <h1>Hello, world!</h1>;
  + 3. Render element to root DOM node  
    root.render( element ); // 大部分React應用只呼叫一次root.render()

1. React element是不可變(固定)的，  
   若要動態更新UI則使用new建立新的element：

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

(setInterval()為每1秒呼叫tick()，每次呼叫會new新的element)達成如下效果：

 # 只有內容更改處才會被React DOM更新

Component 與 Props \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Component可分為function (字首大寫，並回傳React element), class。一張含有 文字, 字型, 螢幕擷取畫面, 行 的圖片

自動產生的描述

( React Component 都必須像 Pure function一般保護他的props #參數不改)

Props是一個物件，可以是一個自定義的element。一張含有 文字, 螢幕擷取畫面, 字型, 行 的圖片

自動產生的描述

* 1. 呼叫root.render(element)
* 2. 以 { name: ‘Sara’} 作為props 傳入 Welcome這個component並呼叫
* 3. Welcome 回傳 <h1>Hello, Sara</h1>
* 4. React DOM將DOM更新為 <h1>Hello, Sara</h1>

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述一張含有 文字, 字型, 螢幕擷取畫面, 印刷術 的圖片

自動產生的描述

此例中，App()為最高層級的component (每個React應用中有一個)，  
Welcome是小型component (有多個)。

State 與 生命週期 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

讓Clock這個function變成一個擁有State的class。

1.原function:一張含有 文字, 螢幕擷取畫面, 軟體, 字型 的圖片

自動產生的描述

2.轉換成class: ( function內容搬到render(){ }, 加上this)

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

3.加入Local State ( 建一個constructor(props){} 初始化state )

一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

自動產生的描述

1. 加入生命週期方法 (設定timer)

分為mount(設定timer)、unmount(清除timer)。一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

1. 加入tick() 讓 Clock component每秒執行 (必須setState())一張含有 文字, 螢幕擷取畫面, 軟體 的圖片

   自動產生的描述

步驟1. root.render(<Clock />)呼叫Clock組件 並進行constructor初始化

步驟2. 呼叫render()方法，插入DOM

步驟3. 每當插入DOM，會呼叫componentDidMount() 產生生命週期

步驟4. 週期中每秒呼叫tick()，由於呼叫tick(),透過setState()，React知道state改變了，再次呼叫render()去更新DOM。  
\* 不可直接改變state的值，必須使用setState()。

步驟5. 直到Clock 組件從DOM被移除，呼叫componentWillUnmount()

　　　　清除timer。

互動

標籤中使用onClick屬性:

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

列表(li)中的key

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

當一個列表被重新render時，  
React根據每個項目的key搜尋上一個列表的項目，尋找相對應的key，

讓 React 能夠在每次重新 render 之間維持 state。

React 會自動使用 key 來決定哪一個 component 需要被更新：

* 如果目前列表多了一個之前不存在的key：建立一個component
* 如果目前列表缺少一個之前有的 key：將上一個 component 摧毀
* 如果兩個 key 符合：相對應的 component 就會被移動
  + 推薦在建立動態列表時一律指定適當的 key