강의 내용 정리 (1 주 ~ 7 주)

참조 파일:

(1) 21-webProg-midTest.pdf (2 학년 2 학기, 웹 프로그래밍 중간 고사)
(2) object / objec1.js, object2.js, object3.js, object5.js
(3) fac.js
(4) closureEx.js
(5) removeRestore.html
(6) tagByJS/ ex82-createNode.html, Ex82multiComp.jsx, Ex82.jsx, Ex82useState.jsx
(7) JSX2.jsx
(8) rendering / IncDec2.jsx
(9) NumberList2.jsx
(10) functional / array-api.js, array.js, destructuring.js, qsort.js

1. HTML / CSS
- 위의 중간고사 문제 참고

2. JavaScript 의 객체 정의하는 방법 (파일 : prototypeBased.pdf)
- 리터럴 (파일 : object1.js)
- 동적으로 프로퍼티 추가 (파일 : object2.js)
- 생성자 함수 이용 (파일 : object3.js)
- Class 이용 (파일 : object5.js)

3. Factorial 함수를 정의하는 여러 가지 방법 (파일 : fac.js)
- 전역변수 및 Loop 문 사용 : 명령형 프로그래밍
- 재귀함수로 표현
- if 대신 3 항 연산식 사용
- 람다 함수로 표현
- 중괄호 및 return 이 생략된 람다 함수로 표현 : 함수형 프로그래밍

4. 한 태그에 onclick 이벤트를 추가하는 5 가지 방법 (파일: onclick.html)
- addEventListener 내에서 핸들러를 직접 구현
- addEventListener 내에서 핸들러를 호출
- 객체에 onclick 프로퍼티를 추가함
- 객체에 onclick 프로퍼티의 함수를 화살표 함수로 표현 (this 대신 객체로 표현)
- 객체에 onclick 프로퍼티의 함수를 다른 함수를 호출하도록 함 (위의 2 번과 유사함)

5. Closure (파일 : closureEx.js)

- 함수 내에 함수를 내포할 수 있으며, 내포된 함수는 함수 밖에서 선언된 변수를 사용할 수 있음. 결과적으로 선언된 변수의 scoping 의 이슈가 됨.

6. DOM 트리에서 노드 remove/append 하기

(파일: removeRestore.html, hw-web-domTree.pdf)

DOM 트리에서 parent/child 관계는 parent 노드에서부터 children 노드 사이의 링크(link, pointer) 로서 표현된다. 메소드 removeChild 와 appendChild 는 이 둘 사이의 링크를 끊거나/붙이는 기능을 한다. body.removeChild(p) 는 p 노드를 삭제하는 것이 아니라, body 노드로부터 단지 링크를 끊을 뿐이며, 그 내용은 그대로 남아 있다. 이 상황에서 body.appendChild(p) 를 수행하면 원 상태로 회복된다.

7. 이벤트 (파일 : ch09.pdf)

- 주요 이벤트 : load, click, change 등
- 이벤트 객체 (event object)
- preventDefault()

8. HTML 대신 JavaScript 로 표현하기

- (파일 : ex82-createNode.html, Ex82multiComp.jsx, Ex82.jsx, Ex82useState.jsx)

<body> 태그 내에 태그를 사용하여 웹 페이지를 구성하는 대신, JavaScript 로 이를 표현하는 방법.

- ex82-createNode.html : createElement, appendChild 메소드를 이용하여 트리 완성
- Ex82multiComp.jsx : 위의 html 과 유사한 형태의 react 코드. getElementById 메소드와 할당문에 의한 상태 변경이 이루어지고 있으므로 react 환경을 이용하고 있기는 하지만, 기존 vanilla JS 코드 (즉, DOM 프로그래밍) 기법을 일부 사용됨
- Ex82.jsx : 위의 jsx 파일의 코드에서 컴포넌트의 수를 좀 더 줄이고, JSX 표현을 사용하여 간결하기 표현함. 아직도 getElementById 와 할당문에 의한 상태 변경을 이용하고 있음
- Ex82useState.jsx : 함수형 컴포넌트와 useState 를 이용한 제대로 된 React 코드.

9. JSX (파일 : JSX2.jsx, Ex82useState.jsx)

- JSX2.jsx : element1 과 element2 는 동일한 효과를 얻는다. 즉, JSX 태그는 단순히 javascript 객체를 단순히 표현한 것으로서, greeting 함수에서 보듯이 jsx 는 일반 javascript 의 expression 처럼 자유롭게 사용될 수 있다.

- Ex82useState.jsx : jsx 의 태그 내에 표현되는 것은 스트링이다. 스트링이 아닌 JavaScript 의 expression 으로 표현하고 싶을 때는 중괄호 { } 를 사용해야 한다. style 은 객체이므로 { … } 형태로 표현되는데, jsx 내에서 이들은 다시 중괄호로서 안에 내포되어야 하므로 {{ … }} 형태로서 표현된다.

10. React 개념 이해하기

(파일 : IncDec2.jsx)

- 상태는(state) 는 어떤 상황에서 발생하는가? Mutable 변수를 사용할 때, 한 변수에 값이 2 번 이상 여려 번 동적으로 (dynamically) 변화함으로써 발생함. immutable 변수만을 사용하는 순수 함수형 프로그래밍에서는 상태가 발생하지 않는다.
- IncDec2.jsx : 웹의 이벤트 핸들러의 대부분은 필수적으로 상태 변환을 해야 한다. 따라서 React 에서 함수형 프로그래밍 기법을 적용할 때는 상태 변환을 위해서 특별한 함수들을 사용한다. 이들을 hooks 함수라고 하는데, 대표적으로 useState, useEffect, … 등이 있다. 여기서 유의해야 할 점은 UI 를 위한 rendering 용 변수의 상태 변환은 할당문을 사용하지 않고 useState 에서 지정한 변수 및 set 함수를 이용한다. 그렇지 않은 경우 상태 변환된 변수 값이 rendering 되지 않는다. (강의자료 W6.pptx 의 8 쪽)
- React 컴포넌트란 무엇인가? 컴포넌트는 JavaScript 코드 (JSX 와 함께)로 표현된, 모듈화된 프로그램의 일부분이다. 웹의 프로그램 코드는 MVC (Model, Visual, Control) 을 표현해야 하는데, 한 컴포넌트 내에 필요한 이 세 요소들이 캡슐화되어 표현된 독립적으로 동작할 수 있어야 한다. 컴포넌트가 선언적(declarative) 하다고 하는 것은 한 컴포넌트 호출 결과가 문맥에 상관없이 언제나 일정한 결과 (jsx) 를 갖게 됨을 의미한다. 이렇게 하면, 큰 규모의 웹 프로그램을 레고 블록을 조립하듯이, 컴포넌트들을 호출함으로써 구성할 수 있다. 그리고, 이 컴포넌트들은 여러 프로그램에 사용될 수 있으므로 프로그램 생산성을 향상 시킨다.

11.클래스 컴포넌트를 함수형 컴포넌트로 변환하기

- 모든 클래스 컴포넌트들은 함수형 컴포넌트들로 표현될 수 있으며, 두 가지 형태의 컴포넌트들이 한 프로그램 내에서 공존할 수 있다.
- 함수형 컴포넌트 호출 시 파라메터 (props 라고도 함)을 전달하는 방법 (W6.pptx, 4 쪽)
- 클래스 컴포넌트에서 사용되는 생명 주기 함수 (lifecycle 메소드) 들 componentDidMount, componentDidUpdate, componentWillUnmount 이 호출되는 시점 및 사용 방법에 대해서 설명하라.

12. Functional 컴포넌트와 일반 JavaScript 함수의 비교

- Functional 컴포넌트와 JavaScript 의 보통 함수와의 차이점을 비교하라 (반드시 시험에 출제됨).
 함수 이름, 함수 호출의 표현식, 파라메터(props) 전송 표현식, 컴포넌트 함수 호출이 top-down 방식으로 진행됨, return 값의 특성 등

13. 배열을 <li> 로 연관시켜 표현할 때의 key (파일 : NumberList2.jsx)
- 각 item 마다 unique 한 key 값이 주어져야 함
- 이 key 값이 할당되는 곳은 map 함수 내 (혹은 배열처리를 위한 loop 문 내)이다.

14. 함수형 프로그래밍
(파일 : functional / array-api.js, array.js, destructuring.js, qsort.js)
- 배열을 처리하는 함수 (loop 대신) : map, filter, forEach, find, ….
- 배열 및 객체에 대한 destrcutring

15. 기타 강의 자료 참고