**1. FLEX레이아웃과 GRID 레이아웃을 설명하고 예제를 만드시오.**

- Flex는 요소의 크기가 불분명하거나 동적인 경우에도, 각 요소를 정렬할 수 있는 효율적인 방법을 제공합니다.

우선 Flex는 2개의 개념으로 나뉩니다.

1. Container : Container는 Items를 감싸는 부모 요소이며, 각 Item을 정렬하기 위해선 Container가 필수입니다. display, flex-flow, justify-content 등의 속성을 사용할 수 있습니다.

2. Items : order, flex, align-self 등의 속성을 사용 할 수 있습니다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

예제)

스크린샷이(가) 표시된 사진

자동 생성된 설명

- GRID는 예전부터 핵(Hack)으로 불린 다양한 레이아웃 대체 방식들을 해결하기 위해 만들어진 특별한 CSS 모듈입니다.

CSS Flex와 같이 Container(컨테이너)와 Item(아이템)이라는 두 가지 개념으로 구분되어 있습니다.

Container는 Items를 감싸는 부모 요소이며, 그 안에서 각 Item을 배치할 수 있습니다.

예제)

스크린샷이(가) 표시된 사진

자동 생성된 설명

**2. ES6문법을 10가지이상 설명하고 각 예제를 만드시오.**

**- 변수 let 과 const**

ES6 이전에는 변수를 var을 사용했는데 이제 let (변수) 과 const (상수) 로 나누어진다.

var 과의 차이점은 var은 전역 범위인 반면 let 과 const 는 선언한 { } 내부 블록에서만 유효하다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

**- 템플릿 리터럴**

ES6 에서는 백틱 ( ` ) 을 이용하여 문자열을 + 기호 없이 간단히 처리할 수 있다.

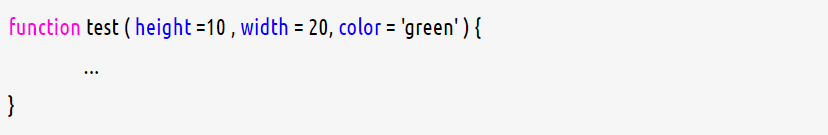
또한 백틱 ( ` ) 안에서는 멀티 라인도 자유롭게 사용할 수 있다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

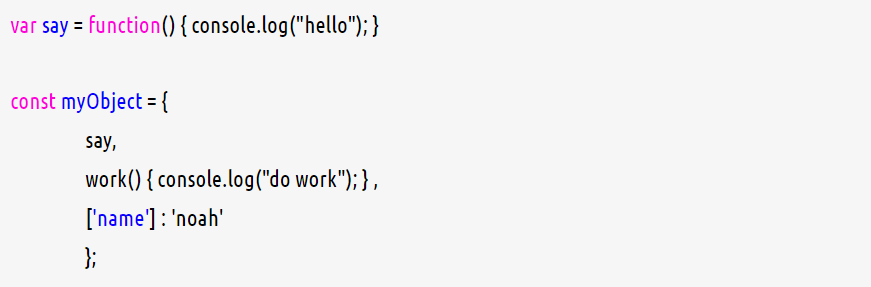
**- 함수의 기본 매개 변수값 지정**

ES6 의 함수는 매개변수에 기본값 (default) 을 지정해 줄 수 있다.



**- 객체 리터럴**

ES6 에서는 객체 리터럴의 사용성이 좀 더 간편화 되었다.



**- 화살표 함수**

ES6 에서는 축약형 함수인 화살표 함수를 사용할 수 있다.

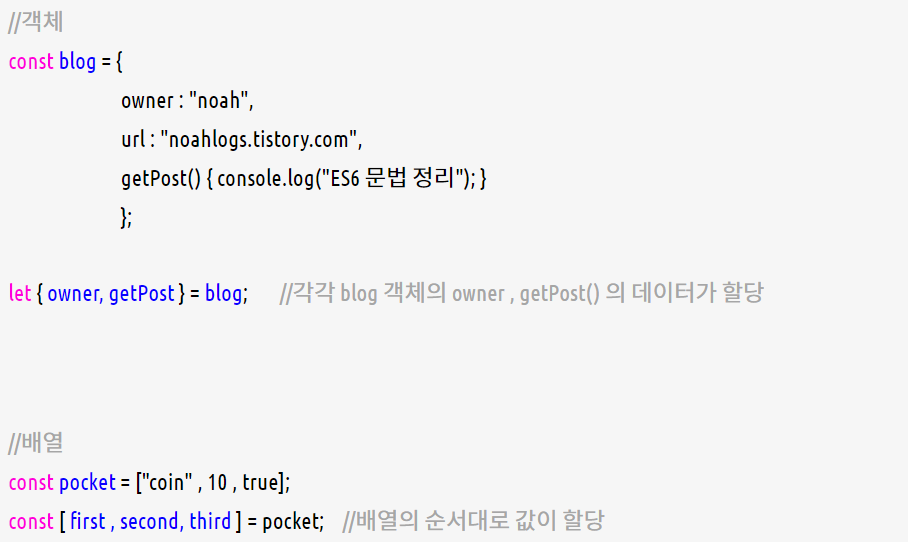
특이한 점이 있다면 화살표 함수에서 this 는 함수 자신을 호출하는 객체가 아닌

화살표 함수의 상위 스코프 (scope) 를 호출한다.



**- 비구조화 할당 (destructuring Assignment)**

비구조화 할당은 배열이나 객체에서 데이터를 편리하게 가져올 수 있게 해준다.

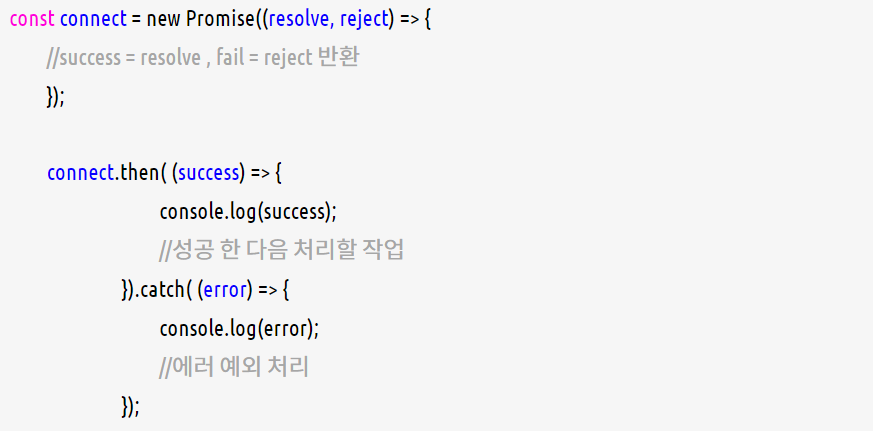


**- 프로미스 (Promise)**

프로미스는 자바스크립트의 비동기 처리에 사용되는 객체이다.

콜백 지옥에서 벗어나게 해주는 유용한 기능이다 :)

ES8 (ECMAScript 2017) 에서 async/await 이 생겨나면서 프로미스에서 처리할 수 없던 순서 처리를 할 수 있게 되었다. 필요할 때 사용하면 될 것 같다.



출처: https://noahlogs.tistory.com/26 [인생의 로그캣]

**3. props 와 state는 언제 쓰는가.**

- props : 부모 객체에서는 자식객체에 데이터를 props 형태로 전달 해 줄 때,

- state : Component 내부에서 생성된 데이터를 변경할 때,

선언된 state 값을 변경하기 위해서는 무조건 this.setState를 사용해야 합니다. React 내부에서 setState를 호출하게 되면 자동으로 Component를 리렌더링하게 설계되어 있습니다.