**클래스**

-클래스가 도입됨.

-기초 문법

class test {

constructor(name) {

this.name = name;

}

hello() {

return ‘Hello World!’

}

}

-상속 : 자바랑 비슷

**let과 const로 변수 선언하기**

**-**둘 다 지역변수임

-const는 변경 불가, let은 변경 가능

**화살표 함수**

-이전버전과 달리 함수선언을 => 로 하도록 함

//이전

const testFunction = function() {

…

}

//es6

const testFunction = () => {

//

}

-매개변수 하나일 시 : const test = name => { … } 처럼 괄호 생략가능

-매개변수 두개 이상 :const test = (name, age) => {…} 처럼 쓴다.

**암묵적 반환(implicit return)**

-화살표 함수의 내용이 한 줄로 구성되어있다면, 몸체를 감싸는 {}와 return을 생략할 수 있다.

const testFunction = () => ‘hello hihi.’;

**구조분해 할당(destructuring assignment)**

-객체나 배열의 일부를 쉽게 변수로 해체할 수 있게 하는 것.

const developer = {

firstName: ’seongil’,

lastName: ’shin’,

developer : true;

age:23,

}

const {firstName, lastName} = developer; // developer의 firstName, lastName이 새 변수에 들어감.

console.log(fisrtName);

console.log(lastName);

//이름을 바꾸고 싶을 시

const {firstName:name } = developer; //developer의 firstName이 새 변수 name에 들어감

//배열 사용시

const [one, two] = numbers; // 첫번째 두번째 요소 들어감

const [one, , three, ,five] = numbers; //첫번째, 세번째, 다섯번째 요소 들어감.

**맵과 필터**

-ES5 배열 메서드지만, react에서 많이 쓰이기에 저술

-데이터를 가공할 때 유용하게 쓰인다.

-API를 통해 받아온 데이터가 JSON 배열이라 가정

render() {

const users = [

{ name:’Nathan’, age:25},

{ name:’Jack’, age:30}.

{ name:’Joe’, age:28}.

];

return (

<ul>

{users

//데이터를 필터링해서 사용할 수 있다.

.filter(user => user.age > 26)

//배열의 각 요소를 map을 이용해 뽑아내 render()에서 사용할 수 있다.

.map(user => <li>{user.name}<\li>)

}

</ul>

)

}

**ES6 모듈시스템**

-자바스크립트가 파일을 호출(import)하고, 선언하여 내보낼 수 있게 함(export)

-모듈은 export 를 통해 하나 이상의 값(객체, 함수 or 변수)를 선언하여 외부로 내보낼 수 있게함

//default export 딱 하나만 사용가능.

export default function times(x) {

return x\*x;

}

//여러 개의 named expeort 선언가능

export function timesSecond(x) {

return x\*x;

}

export function plusTwo(number) {

return number + 2;

}

-불러올땐 다음과 같이 import로 불러옴. default export 한 건 { } 를 안사용해도 자동으로 import됨.

import {timesSecond, plusTwo } from ‘./util.js’;

-다음과 같은 as를 통해 다른 이름을 지어줄수도 있다.

import {timesSecond as mult, plusTwo as plus2 } from ‘./util.js’;

-다음과 같이 절대이름으로 import도 가능하지만, 이때는 상응하는 패키지 이름을 node\_modules 에서

검색하므로, local에 저장된 파일을 import할 때는 파일의 위치를 제대로 입력해야한다.