데이터 시각화(ECMA 6 기술)

Data visualization

ECMA 6 오브젝트

Object(오퍼레이션)

- ES6에서는 Object 오브젝트에 프로퍼티를 작성하고 제어하는 방법이 추가

- Object에서 같은 Key 사용

  var obj = {key: value}
  라는 형태를 가진 object에서 key 값이 같은 프로퍼티를 두 개 작성했을 때,,,

  ES3: key 값이 같더라도 따로 추가
  ES5: strict 모드에서만 에러 발생
  ES6: strict 모드에서도 에러가 발생하지 않으면 나중에 작성된 프로퍼티 값으로 대체

```
let sameKey = {one: 1, one: 2};
console.log(sameKey);
```

ECMA 6 오브젝트

Object(오퍼레이션)

- 변수 이름으로 값 설정
  변수 이름을 사용하여 Object 오브젝트의 프로퍼티 값을 설정 가능

```
let one = 1, two = 2;
let values = {one, two};
console.log(values);
```

- 변수 이름이 프로퍼티 이름이 되고 이에 대한 값이 프로퍼티 값으로 설정됨

- function 작성 방식 변경

```
let obj = {
   getTotal : function (param){
                  return param + 123;
    }
};
```

```
let obj = {
   function getTotal(param){
                  return param + 123;
    }
};
```

- 왼쪽 ES5, 오른쪽 ES6

ECMA 6 오브젝트

Object(오퍼레이션)

- 변수 이름으로 값 설정
  변수 이름을 사용하여 Object 오브젝트의 프로퍼티 값을 설정 가능

```
let one = 1, two = 2;
let values = {one, two};
console.log(values);
```

- 변수 이름이 프로퍼티 이름이 되고 이에 대한 값이 프로퍼티 값으로 설정됨

- function 작성 방식 변경

```
let obj = {
   getTotal : function (param){
                  return param + 123;
    }
};
```

```
let obj = {
   function getTotal(param){
                  return param + 123;
    }
};
```

- 왼쪽 ES5, 오른쪽 ES6

ECMA 6 오브젝트

Object(디스크립터)

- 디스크립터라는 기능은 ES5에서 제시되었음

```
Object.definePropery({}, “book”, {                    // 프로퍼티 이름
    value : 123,
    enumerable: true                                  //프로퍼티 디스크립터
}
```

타입	속성 이름	속성 값 형태	디폴트 값	개요
데이터	value	JavaScript 데이터 타입	undefined	프로퍼티 값으로 사용
	writable	true, false	false	false: 속성 값 변경 불가
엑세스	get	function, undefined	undefined	프로퍼티 getter 함수
	set	function, undefined	undefined	프로퍼티 setter 함수
공용	enumerable	true, false	false	false: for-in으로 열거 불가
	configurable	true, false	false	false: 프로퍼티 삭제 불가

ECMA 6 오브젝트

Object(디스크립터)

- 디스크립터라는 기능은 ES5에서 제시되었음

```
Object.definePropery({}, “book”, {                    // 프로퍼티 이름
    value : 123,
    enumerable: true                                  //프로퍼티 디스크립터
}
```

타입	속성 이름	속성 값 형태	디폴트 값	개요
데이터	value	JavaScript 데이터 타입	undefined	프로퍼티 값으로 사용
	writable	true, false	false	false: 속성 값 변경 불가
엑세스	get	function, undefined	undefined	프로퍼티 getter 함수
	set	function, undefined	undefined	프로퍼티 setter 함수
공용	enumerable	true, false	false	false: for-in으로 열거 불가
	configurable	true, false	false	false: 프로퍼티 삭제 불가

ECMA 6 오브젝트

Object(디스크립터 – get, set 속성 기능 ES5)

- get 속성은 getter 기능을 제공하고 set 속성은 setter 기능을 제공

```
var obj = {};
Object.defineProperty(obj, “book”, {
  get : function () {
   return “책”;
   }
});
console.log(obj.book);
```

- 해당 obj 오브젝트에서 book 프로퍼티 작성 여부를 체크
- 작성되어 있을 경우 get 속성 존재 여부를 자동으로 체크
- getter는 obj.book() 과 같이 함수를 호출하는 형태로 작성하지 않고 obj.book과 같이 함수 이름만 작성

```
var obj = {};
Object.defineProperty(obj, “item”, {
  set : function (param) {
    this.sports = param;
  }
});
obj.item = “야구”;
console.log(obj.sports);
```

ECMA 6 오브젝트

Object(디스크립터 – get, set 속성 기능 ES6)

- 좀 더 직관적으로 getter와 setter를 정의할 수 있음

```
let obj = {
   value : 123,
   get getValue(){
     return this.value;
   }
};
console.log(obj.getValue);
```

```
let obj = {
   set setValue(){
      this.value=value;
   }
};
obj.setValue = 123;
console.log(obj.value);
```

ECMA 6 오브젝트

Object(is())

- 값과 값 타입을 비교하여 같으면 true, 아니면 false를 반환

구분	타입	데이터(값)
형태		Object.is()
파라미터	Any	비교 대상 값
	Any	비교 대상 값
반환	Boolean	값과 값 타입이 같으면 true, 아니면 false

```
console.log("1:" , Object.is(1, "1"));
console.log("2:" , Object.is(NaN, NaN), NaN === NaN);

console.log("3:" , Object.is(0, -0), 0 === -0);
console.log("4:" , Object.is(-0, 0), -0 === 0);

console.log("5:" , Object.is(-0, -0), -0 === -0);
console.log("6:" , Object.is(0, -0), 0 === -0);

console.log("7:" , Object.is(undefined, null), undefined === null);
```

ECMA 6 오브젝트

Math

- ES6에 추가된 Math 함수

함수 이름	개요	함수 이름	개요
sinh()	쌍곡 사인	sign()	사인 값
asinh()	쌍곡 아크사인	trunc()	소수를 제외한 정수
cosh()	쌍곡 코사인	clz32()	32비트 값에서 0비트 수
acosh()	쌍곡 아크코사인	fround()	32비트 유동 소수 값
tanh()	쌍곡 탄젠트	expm1()	자연로그 상수(e)의 x승-1
atanh()	쌍곡 아크탄젠트	hypot()	제곱근
log1p()	log(1+파라미터 값)	cbrt()	세제곱근
log10()	10을 밑으로 한 로그 값	log2	2를 밑으로 한 로그 값

imul()	파라미터 값을 곱하고 결과를 32비트로 변환