JavaScript

목표

Javascript의 Object, Array 를 다룰 줄 안다.

HTML 태그를 동적으로 추가할 수 있다.

Object, Array ?

오늘 배우는 걸 어디에 써먹나?

대부분의 콘텐츠는 리스트형태이며,
자주 변경거나 업데이트 된다.

즉,
서버로부터 데이터를 받고
필요시 데이터를 가공해서
본문에 추가하거나 업데이트 한다.

변 수

JavaScript의 타입에는
primitive type과 reference type이 있음

primitive type, reference type

primitive type

null, undefined, boolean, Number, String

typeof 연산자로 primitive 타입인지 확인 할 수 있다 (null은 Object로 나옴).

reference type

object, array..

instanceof로 타입을 확인할 수 있다.

primitive , reference

primitive

```
var str = “i love NEXT”;
var copyStr = str;
 str = “i love H-Square”;
console.log(copyStr);
```

값이 **복사**될 뿐이다.

reference

```
var str = new Object();
var copyStr = str;
str.name = “next”;
console.log(copyStr.name);
```

값이 같이 **참조**(공유)되고 있다

Object

```
var myCar = new Object();
```

```
myCar["make"] = "Ford";
myCar["model"] = "Mustang";
myCar["year"] = 1969;
```

```
myCar.make = "Ford";
myCar.model = "Mustang";
myCar.year = 1969;
```

소스코드 참고 :
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects?redirectlocale=en-US&redirectslug=JavaScript%2FGuide%2FWorking_with_Objects

자주쓰는 똑같은 방식.

```
var myCar = {};
```

```
myCar["make"] = "Ford";
myCar["model"] = "Mustang";
myCar["year"] = 1969;
```

```
myCar.make = "Ford";
myCar.model = "Mustang";
myCar.year = 1969;
```

소스코드 참고 :
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects?redirectlocale=en-US&redirectslug=JavaScript%2FGuide%2FWorking_with_Objects

`for in` (Iterating)

```
for ( var i in myCar) {
   console.log('key : ' + i + ', ' + 'value : ' + myCar[i]);
}
```

```
//result
key : make, value : Ford
key : model, value : Mustang
```

참고.

hasOwnProperty() 메소드를 통해서 좀더 효율적인 객체의 원소를 순회할 수 있습니다.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/hasOwnProperty

delete

```
myCar.make = null;          ( 틀린 방법 )
myCar.make = undefined;     ( 틀린 방법 )

delete myCar.make;          ( 올바른 방법 )
```

숙제5

다음 object를 인자로 받아 object의 모든 값을 출력하는 function을 만드세요.

```
obj = {
        name : ‘javascript’,
  age  : 14,
  type : ‘dynamic’
};
```

Array

```
var myCar = new Array();

var myCar = Array();

var myCar = new Array(2);

var myCar = Array(2);
```

그러나.. 아래 두 개는 완전히 다르다

```
var myCar = new Array(2);
var myCar = new Array('2');
```

자주쓰는 똑같은 방식.

```
var myCar = [];
var myCar = [1,2,3,‘apple’];
```

Javascript Array의 크기는 **동적으로 변한다**.

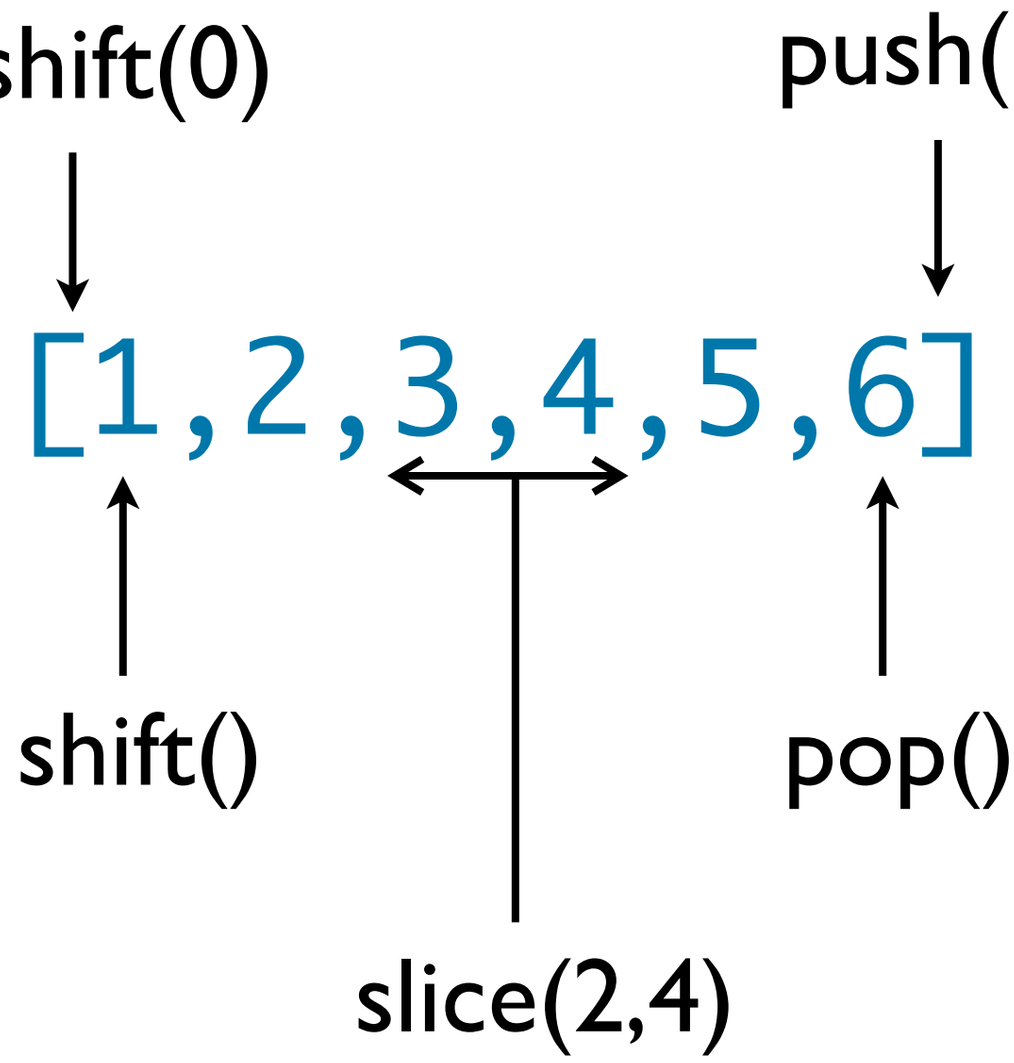
```
var myCar = [];

myCar.length = 10000;

console.log(myCar.length);
```

Array methods.

반환값을 확인해보자.



splice는 막강하며 **원본 배열의 값을 쉽게 변경할 수 있다**

```
splice(index, count_to_remove, addelement1, addelement2, ...)
```

```
[1,2,3,4,5,6]
```

```
myNum.splice(3,2,33,55);
```

Array Iterating

```
var colors = ['red', 'green', 'blue'];

for (var i = 0; i < colors.length; i++) {
  console.log(colors[i]);
}
```

소스코드:
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Predefined_Core_Objects?redirectlocale=en-US&redirectslug=JavaScript%2FGuide%2FPredefined_Core_Objects

easy Iterating – **forEach**

array를 순회하면서 **무언가**를 한다

```
var a = ['a', 'b', 'c'];
a.forEach(function(item){
            console.log(item);
          });
```

easy Iterating – **map**

array를 순회하면서 **값을 조작하여 새로운 array**를 반환한다.

```
var a = ['a', 'b', 'c'];
var result = a.map(function(item){
      return item+item;
    });

console.log(result);
```

easy Iterating – **filter**

array를 순회하면서 값을 **특정조건에 맞는** 새로운 array를 반환한다.

```
var a = [123,421,3,31,24,55,32];
var result = a.filter(function(item){
    return item % 2 ===0;
});

console.log(result);
```

숙제6

array 중에 짝수 숫자타입 원소만으로 구성된 배열을 만드는 userFunction 함수를 만드세요.

```
var arr = [23,-2,23.4 , “324”, 333, 34, 1000,20];
userFunction(arr);  // [-2, 34, 1000, 20]
```

End

;-D