

05

## 객체(object)

## ❖ 객체(object)

- 하나의 변수에 다양한 정보를 담기 위해 사용하는 자료형
- 객체를 변수에 저장하면 객체가 저장된 참조(reference) 를 저장

객체 : `let obj = { }`

`let book = {`

`title : 'javascript',`

`author : '홍길동',`

`pages : 500,` → 속성 이름 : 속성 값, key : value

`price : 15000`

메서드 `info : function(){`

`alert(this.title + '책의 분량은 ' + this.pages + '쪽입니다');`

`}`

`};`

`> book.pages → 500`

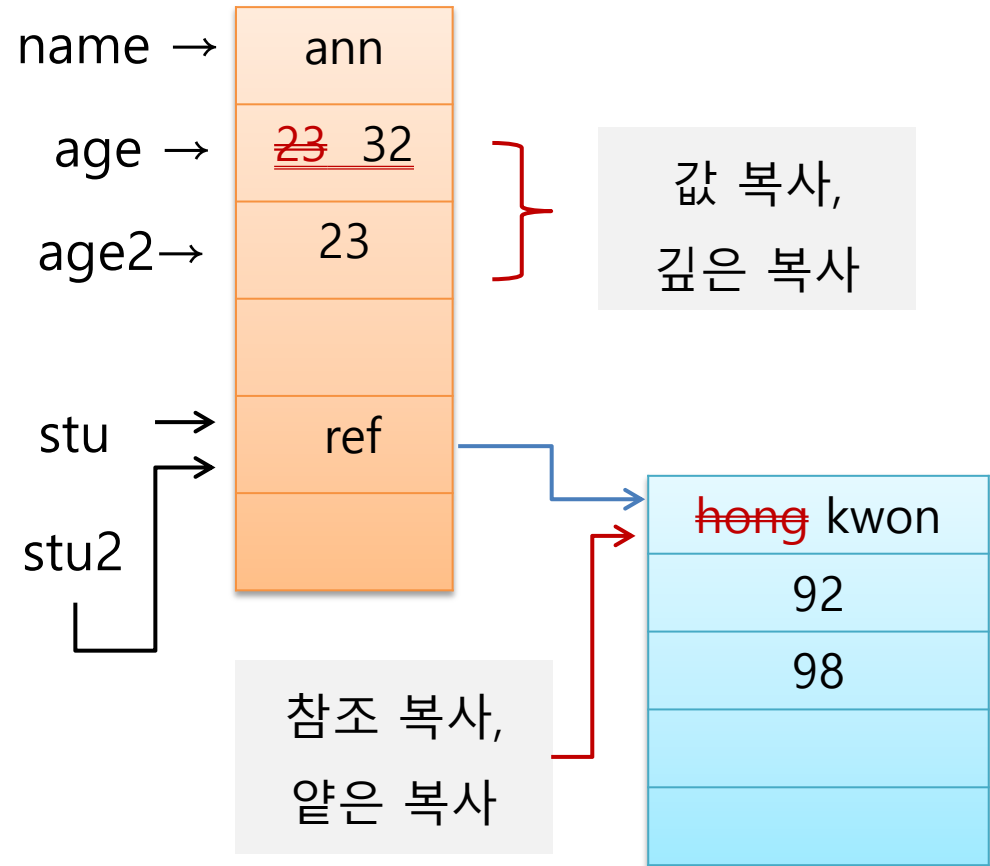
`> book.info()`

• : object access operator

## ❖ 객체(object) 참조 복사

```
let name='ann'
let age=23
let age2= age
age = 32
```

```
let stu = {
  name : 'hong',
  kor : 92, eng : 98,
};
let stu2 = stu
stu.name = 'kwon'
```



## ❖ 객체의 깊은 복사(clone)

```
function clone(obj){  
  let output = {};  
  for (let i in obj){  
    output[i] = obj[i];  
  }  
  return output;  
}  
  
let original = {a: 10, b: 20};  
let referenced = original;  
let cloned = clone(original);  
original.a = 88;
```

```
console.log(original);  
console.log(referenced);  
console.log(cloned);
```

```
▶ {a: 88, b: 20}
```

```
▶ {a: 88, b: 20}
```

```
▶ {a: 10, b: 20}
```

## ❖ 속성과 메서드

```
let object = {  
  number : 273,  
  string : 'text',  
  boolean : true,  
  array : [1, 2, 3, 4],  
  func : function(food){ }  
};  
  
object.func();
```

객체 내부 값 : 속성(property)

-- 객체 내부 함수 : 메서드(method)

---- 메서드 호출

## ❖ for in 반복문 : 객체의 요소 개수만큼 반복문 실행

```
let object = {  
  number : 273,  
  string : 'text',  
  boolean : true,  
  array : [1, 2, 3, 4],  
  func : function(){}  
}
```

```
let output = '';
```

```
for ( let key in object) {  
  output += ' - ' + key + ' : ' + object[key] + '\n';  
}
```

```
console.log(output);
```

```
- number : 273  
- string : text  
- boolean : true  
- array : 1,2,3,4  
- func : function(){}  

```

## ❖ 객체 관련 키워드

### in 키워드 ( 'key' in 객체 )

- 해당키가 객체 안에 있는지 확인

```
let check = "";
let student = {
  이름 : '홍길동',
  국어 : 92, 수학 : 98,
  영어 : 96, 과학 : 98
};
// in keyword
check += "'이름' in student : " + ('이름' in student) + '\n';
check += "'성별' in student : " + ('성별' in student);
console.log(check);
```

```
'이름' in student : true
'성별' in student : false
```

## ❖ 객체 관련 키워드

**with 키워드 : with (객체) { 코드 }**

- 코드를 짧게 줄여주는 키워드

```
let check = '';  
let student = {  
  이름 : '홍길동',  
  국어 : 92, 수학 : 98,  
  영어 : 96, 과학 : 98  
};
```

```
with(student){  
  sungJuk += '이름 : ' + 이름 + '\n';  
  ...  
  sungJuk += '총점 : ' + ( 국어 + 수학 + 영어 + 과학) ;  
}
```

```
let sungJuk = '';  
  
sungJuk += '이름 : ' + student.이름 + '\n';  
sungJuk += '국어 : ' + student.국어 + '\n';  
sungJuk += '수학 : ' + student.수학 + '\n';  
sungJuk += '영어 : ' + student.영어 + '\n';  
sungJuk += '과학 : ' + student.과학 + '\n';  
sungJuk += '총점 : ' + ( student.국어 + student.수학 + student.영어 + student.과학) ;  
console.log(sungJuk);
```



## ❖ 객체의 속성 추가 제거(delete)

```

student.미술 = 100;          ---- 속성 추가
student.toString = function(){  ---- 메서드 추가
    let output = "";
    for (let key in this){
        if (key !== 'toString'){
            output += key + ' : ' + this[key] + '\n' }
        }
    return output;
}
console.log(student.toString());

delete (student.미술); ----- 속성 제거
console.log(student.toString());

```

이름	:	홍길동
국어	:	92
수학	:	98
영어	:	96
과학	:	98

미술	:	100
----	---	-----

이름	:	홍길동
국어	:	92
수학	:	98
영어	:	96
과학	:	98

## ❖ 생성자 함수(constructor)

### 생성자 함수와 객체 생성

- 객체를 생성할 때 사용하는 함수
- 함수의 이름은 대문자로 시작

```
function Student(name, korean, math, english, science){
```

```
    this.name = name;
```

```
    this.korean = korean;
```

```
    this.math = math;
```

```
    this.english = english;
```

```
    this.science = science;
```

} 속성 생성

```
}
```

인스턴스 : 생성자 함수로 생성된 객체

```
let student = new Student('홍길동', 96, 98, 92, 98);
```

객체 생성 키워드

→ 생성자 함수

## ❖ 생성자 함수

### 메서드 생성

```
function Student(name, korean, math, english, science){
    this.name = name; this.korean = korean; this.math = math; this.english = english;
    this.science = science;

    this.getSum = function(){
        return this.korean + this.math + this.english + this.science;
    };
    this.getAvg = function(){
        return this.getSum()/4;
    };
    this.toString = function(){
        return this.name + ' ₩t' + this.getSum() + ' ₩t ' + this.getAvg();
    };
}
let student = new Student('홍길동', 96, 98, 92, 98);
console.log(`이름₩t총점₩t평균`);
console.log(student.toString());
```

## ❖ 생성자 함수

### 프로토타입(prototype)

- 생성자 함수로 생성된 객체가 공통으로 가지는 공간
- 일반적으로 메서드를 선언
  - 속성은 모든 객체가 다른 값, 메서드는 같은 값을 가짐
- 메모리 공간 효율적 사용
- 프로토타입(prototype)도 객체
- 프로토타입을 사용해서 기존 객체에 메서드 추가

```
Rectangle.prototype.getArea = function(){  
    return this.width * this.height;  
};
```

## ❖ 생성자 함수 관련 키워드

### new 키워드

- 일반적으로 this는 window 객체를 의미
- new 키워드로 함수를 호출하면
  - 객체를 위한 공간을 만들고
  - this는 해당공간을 의미

### instanceof 키워드

- 해당 객체의 생성자 함수 확인

```
console.log(student instanceof Student);    // true  
console.log(student instanceof Number);    // false
```

## ❖ 도전! 문제

### 문제1

다음과 같은 객체를 생성할 수 있는 생성자 함수를 작성하여 가격을 출력해 보세요.

- 생성자 함수명 : Product

키(key)	값(value)	설명
이름	삼겹살	판매가격 책정 객체 만들기
무게	100g	
가격	1690원	
메소드	기능	
calculate	구매가격 계산	500g 구매 시 가격은?