7장

자바스크립트 코어 객체와 배열

객체 개념

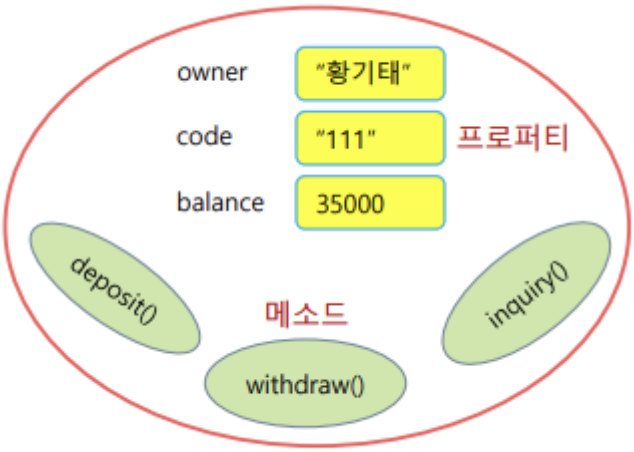
: 자신만의 고유한 구성 속성을 가진다

- 자바 스크립트 객체

: 여러 개의 property와 method로 구성

→ property : 객체의 고유한 속성(변수)

→ method : 함수



자바스크립트 객체 account

- 자바 스크립트 객체 종류

⇒ 자바스크립트는 객체 기반 언어

1. 코어 객체 : 자바스크립트 언어가 실행되는 어디서 나 사용 가능한 기본 객체
2. HTML DOM 객체 : HTML 문서에 작성된 각 HTML 태그들은 객체 화 한 것들
3. 브라우저 객체 : 자바스크립트로 브라우저를 제어하기 위해 제공되는 객체 (BOM)

- 코어 객체

→ 종류: Array ,Date, String ,Math

→ 생성: new 키워드 이용 (객체가 생성되면 객체 내부에 프로퍼티와 메소드들 존재)

→ 접근 : 객체와 멤버 사이에 점(.) 연산자 사용

ex) obj.프로퍼티 = 값;

- 자바스크립트 배열

=⇒ 배열 → 여러 개의 원소들을 연속적으로 저장 , 전체를 하나의 단위로 다루는 데이터 구조

ex) var cities = [”seoul”, “new york” , “paris”];

⇒ 인덱스를 이용하여 배열의 각 원소 접근

=⇒ 배열을 만드는 방법

1. [ ] 로 배열 만들기

ex) var week = [”월” , “화” , “수” …];

→ 배열 크기: 배열의 크기는 고정되지 않고, 마지막 원소 추가 시 늘어남.

2. Array 객체로 배열 만들기

→ 초기 값을 가진 배열 생성 : var week = new Array(”월”, “화”, “수”, ..);

→ 초기화되지 않은 배열 생성

: 일정 크기의 배열 생성 후 나중에 원소 값 저장 : var week = new Array(7);

→ 빈 배열 생성

: 원소 개수를 예상할 수 없는 경우

→ 배열의 크기 : Array 객체의 length 프로퍼티 (length 프로퍼티는 사용자가 임의로 값 변경 가능)

=⇒ 배열의 특징

: 배열은 Array 객체 (“[]”로 생성해도 Array 객체로 다루어짐)

: 배열에 여러 타입의 데이터 섞여 저장 가능

```
var any = new Array(5);      // 5개의 원소를 가진 배열 생성
any[0] = 0;
any[1] = 5.5;
any[2] = "이미지 벡터";      // 문자열 저장
any[3] = new Date();         // Date 객체 저장
any[4] = convertFunction;    // function convertFunction()의 주소 저장
```

concat → 배열과 배열을 연결해준다.

join("##") → 주어진 문자열을 합친다

reverse → 거꾸로 뒤집는다. (자체 변경)

slice → 필요한 구역을 자르기 (ex: `a.slice(1, 2)`는 문자열 `a`에서 인덱스 1부터 (인덱스 2 - 1)까지의 부분 문자열을 반환)

sort → 사전 순으로 정렬 (자체 변경)

toString → 원소 사이에 ","를 넣어 문자열 생성

- Date 객체

: 시간 정보를 담는 객체

ex)

var now = new Date(); // 현재 2017년 5월 15일 저녁 8시 48분이라면
var date = now.getDate(); // 오늘 날짜. date = 15
var hour = now.getHours(); // 지금 시간. hour = 20

vat day = now.getDay(); (0~6으로 표현 ;일 ~ 월)

var minute = now.getMinutes();

var second = now.getSeconds();

var millisecond = now.getMilliseconds();

*toLocalString ⇒ 숫자 , 날짜, 시간을 현지화 된 형식의 문자열로 변환하는데 사용되는 method

- String 객체

: 문자열을 담기 위한 객체

ex)

```
var hello = new String(“Hello”);
var hello = “Hello”;
```

→ String 객체는 일단 생성되면 수정 불가능

=⇒ 특징

→ 문자열 길이: length 프로퍼티 사용

→ 문자열을 배열처럼 사용

[] 연산자를 사용하여 각 문자 접근

```
var hello = new String("Hello");
var c = hello[0];      // c = "H". 문자 H가 아니라 문자열 “H”
```

*indexOf: 문자의 인덱스 번호 추출

*substr(5,3) : 인덱스 5번에서부터 3개의 글자 추출

*.trim(): 문자열의 양 끝에서 공백을 제거하는 메서드

*split() : 빈칸으로 분리

- Math 객체

: 수학 계산을 위한 프로퍼티와 메서드 제공

```
var sq = Math.sqrt(4);        // 4의 제곱근을 구하면 2
var area = Math.PI*2*2;       // 반지름이 2인 원의 면적
```

→ 난수 발생

=⇒ Math.random() : 0~1보다 작은 랜덤한 실수 리턴

=⇒ Math.floor(m) : m의 소수점 이하를 제거한 정수 리턴

- 사용자 객체 만들기

→ 객체 작성 방법

1. 직접 객체 만들기

=⇒ new Object() 이용 , 리터럴 표기법 이용

2. 객체의 틀(프로토 타입)을 만들고 객체 생성하기

→ 과정 (구체적)

1. new Object()로 객체 만들기

< new Object() 로 빈 객체 생성, 빈 객체에 프로퍼티 추가, 빈 객체에 메소드 추가 >

```
var account  = new Object();
account.owner = "황기태";       // 계좌 주인 프로퍼티 생성 및 초기화
account.code = "111";           // 코드 프로퍼티 생성 및 초기화
account.balance = 35000;        // 잔액 프로퍼티 생성 및 초기화
account.inquiry = inquiry;      // 메소드 작성
account.deposit = deposit;      // 메소드 작성
account.withdraw = withdraw; // 메소드 작성
```

2. 리터럴 표기법으로 만들기

<중괄호를 이용하여 객체의 프로퍼티와 메소드 지정>

```
var account = {
  // 프로퍼티 생성 및 초기화
  owner : "황기태",    // 계좌 주인 프로퍼티 추가
  code : "111",        // 계좌 코드 프로퍼티 추가
  balance : 35000,     // 잔액 프로퍼티 추가

  // 메소드 작성
  inquiry : function () { return this.balance; }, // 잔금 조회
  deposit : function(money) { this.balance += money; }, // 저금. money 만큼 저금
  withdraw : function (money) { // 예금 인출, money는 인출하고자 하는 액수
    // money가 balance보다 작다고 가정
    this.balance -= money;
    return money;
  }
};
```

3. 프로토 타입으로 만들기

→ 프로토 타입이란? : 객체 모양을 가진 틀

<프로토 타입은 함수로 만든다, new 연산자로 객체를 생성한다.>

*프로토 타입 함수 = 생성자 함수

```
// 프로토타입 Student 작성
function Student(name, score) {
    this.univ = "한국대학"; // this.univ을 이용하여 univ 프로퍼티 작성
    this.name = name; // this.name을 이용하여 name 프로퍼티 작성
    this.score = score; // this.score를 이용하여 score 프로퍼티 작성
    this.getGrade = function () { // getGrade() 메소드 작성
            if(this.score > 80) return "A";
            else if(this.score > 60) return "B";
            else return "F";
    }
}
```

```
var kitae = new Student("황기태", 75);          // Student 객체 생성
var jaemoon = new Student("이재문", 93); // Student 객체 생성
document.write(kitae.univ + ", " + kitae.name + "의 학점은 " + kitae.getGrade() + "<br>");
document.write(jaemoon.univ + ", " + jaemoon.name + "의 학점은 " + jaemoon.getGrade() + "<br>")
```

—> new 연산자로 객체를 생성