L7-javascript

예제 실습 및 보고서 작성



학 과 : 컴퓨터공학전공

과 목 : 웹프로그래밍

교수님 : 한영선 교수님

학 번 : 202030494

이 름 : 윤원재

제출일 : 2023년 04월 22일

1. **예제 7-1 자바스크립트 객체 생성 및 활용**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**객체 생성 및 활용**</title>**

**</head>**

**<body>**

**<h3>**객체 생성 및 활용**</h3>**

**<hr>**

**<script>**

var today = new Date();

document.write('현재 시간 : ' + today.toLocaleString() + '<br>');

var mystr = new String('자바스트립트 공부하기');

document.write('mystr의 내용 : ' + mystr + '<br>');

document.write('mystr의 길이 : ' + mystr.length + '<br>');

**</script>**

**</body>**

**</html>**

****

<그림 1-1> 예제 7-1 결과

자바스크립트는 객체 지향 언어가 아닌 객체 기반 언어로 객체를 지원하고 있다. 자바스크립트 객체의 유형으로 코어 객체, HTML DOM 객체, 브라우저 객체가 있다. 코어 객체는 자바스크립트 언어가 실행되는 어디서나 사용 가능한 객체로 기본 객체이다. new 키워드를 이용하여 생성하며 객체가 생성되면 객체 내부의 속성과 메소드를 사용할 수 있다. 객체에 접근하기 위해선 ‘.’을 사용한다. <그림 1-1>에서 new 키워드를 사용하여 Date타입의 today 객체를 생성하고 ‘.’을 사용하여 toLocaleString() 메소드를 사용한 것을 확인할 수 있다. 마찬가지로 String 타입의 mysyr 객체를 생성한 것을 확인할 수 있다.

1. **예제 7-2 []로 배열 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**[]로 배열 만들기**</title>**

**</head>**

**<body>**

**<h3>**[]로 배열 만들기**</h3>**

**<hr>**

**<script>**

var plots = [20, 5, 8, 15, 20];

document.write('var plots = [20, 5, 8, 15, 20]<br>');

for(i = 0; i < plots.length; i++) {

var size = plots[i];

while (size > 0) {

document.write('\*');

size--;

}

document.write(plots[i]+'<br>');

}

**</script>**

**</body>**

**</html>**

****

<그림 2-1> 예제 7-2 결과

자바스크립트는 C언어의 문법과 매우 유사하여 []를 사용하여 배열을 만들 수 있다. []내에는 배열의 원소들의 초기값을 나열한다. 이렇게 만든 배열은 크기가 고정되어 있지 않아 원소를 추가하면 늘어난다. 또 현재 배열 길이보다 더 큰 인덱스에 원소를 추가하면 그 사이는 모두 undefined값으로 채워진다. <그림 2-1>에서 []을 사용하여 배열을 만들고나서 각 원소의 값만큼 ‘\*’을 출력한 것을 확인할 수 있다.

1. **예제 7-3 Array 객체로 배열 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**Array 객체로 배열 만들기**</title>**

**</head>**

**<body>**

**<h3>**Array 객체로 배열 만들기**</h3>**

**<hr>**

**<script>**

var degrees= new Array();

degrees[0] = 15.1;

degrees[1] = 15.4;

degrees[2] = 16.1;

degrees[3] = 17.5;

degrees[4] = 19.2;

degrees[5] = 21.4;

var sum = 0;

for (var i = 0; i < degrees.length; i++)

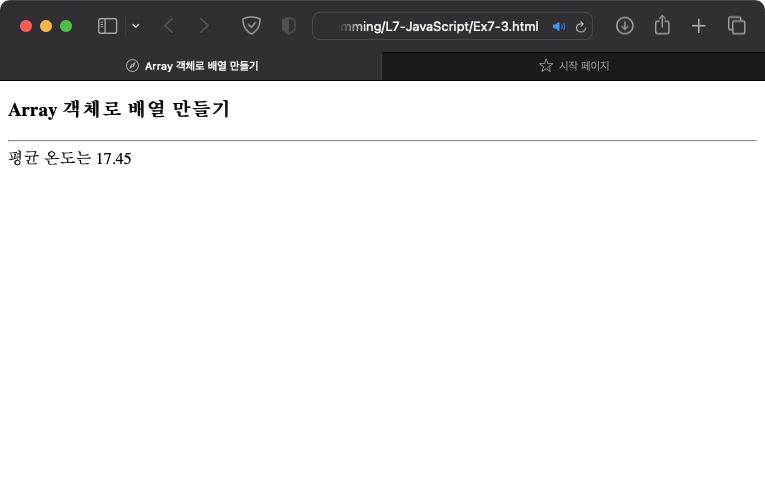
sum += degrees[i];

document.write('평균 온도는 '+ sum / degrees.length + '<br>')

**</script>**

**</body>**

**</html>**

****

<그림 3-1> 예제 7-3 결과

코어 객체 중 하나인 Array 객체를 사용해서 배열을 만들 수 있다. 배열을 생성할 때 Array()의 괄호 안에 초기값이나 배열의 크기를 넣을 수 있다. 원소 개수를 예상할 수 없는 경우 빈 배열을 생성하여 사용한다. <그림 3-1>에서 Array 객체를 사용해서 빈 배열 degrees를 만들었다. 그런 다음 원소를 추가하여 자동으로 배열의 크기가 늘어나는 것을 확인할 수 있다.

Array 객체에 있는 length 프로퍼티로 배열의 길이를 받을 수 있으며 값을 변경하여 배열의 크기를 늘리거나 줄일 수 있다.

1. **예제 7-4 Array 객체의 메소드 활용**

c = a.join('##');

pr('c = a.join(\'##\' 후 c = ', c);

pr('c = a.join(\'##\' 후 a = ', a);

c = a.reverse();

pr('c = a.reverse() 후 c = ', c);

pr('c = a.reverse() 후 a = ', a);

c = a.slice(1, 2);

pr('c = a.slice(1, 2) 후 c = ', c);

pr('c = a.slice(1, 2) 후 a = ', a);

c = a.sort();

pr('c = a.sort() 후 c = ', c);

pr('c = a.sort() 후 a = ', a);

**</script>**

**</body>**

**</html>**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**Array 객체의 메소드 활용**</title>**

**<script>**

function pr(msg, arr){

document.write(msg + arr.toString());

document.write('<br>');

}

**</script>**

**</head>**

**<body>**

**<h3>**Array 객체의 메소드 활용**</h3><hr>**

**<script>**

var a = new Array('1', '2', '3');

var b = new Array('4');

var c;

pr('배열 a = ', a);

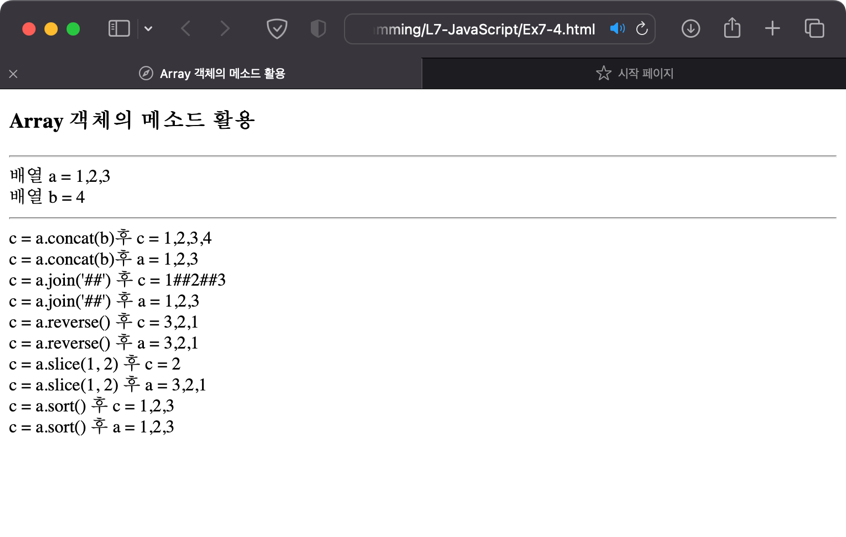
pr('배열 b = ', b);

document.write('<hr>');

c = a.concat(b);

pr('c = a.concat(b)후 c = ', c);

pr('c = a.concat(b)후 a = ', a);

****

<그림 4-1> 예제 7-4 결과

Array 객체에는 다양한 메소드가 있다. <그림 4-1>에서 concat() 메소드를 사용하면 a배열과 b배열을 연결할 수 있다. join() 메소드를 사용하면 a배열의 원소들을 연결할 수 있다. 이때 괄호 안의 내용을 각 원소 사이에 넣을 수 있다. reverse() 메소드를 사용하면 a배열의 원소들의 순서를 역으로 정렬한다. slice() 메소드를 사용하면 배열을 잘라 출력할 수 있다. 시작 index와 종료 index를 지정하여 사용하며 종료 index는 잘라낼 배열에 포함되지 않는다. sort() 메소드를 사용하면 배열 내 원소들을 정렬할 수 있다.

1. **예제 7-5 Date 객체 생성 및 활용**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**Date 객체로 현재 시간 알아내기**</title>**

**</head>**

**<body>**

**<h3>**Date 객체로 현재 시간 알아내기**</h3>**

**<hr>**

**<script>**

var now = new Date();

document.write('현재 시간 : ' + now.toUTCString() + '<br><hr>');

document.write(now.getFullYear() + '년도<br>');

document.write(now.getMonth() + 1 + '월<br>');

document.write(now.getDate() + '일<br>');

document.write(now.getHours() + '시<br>');

document.write(now.getMinutes() + '분<br>');

document.write(now.getSeconds() + '초<br>');

document.write(now.getMilliseconds() + 1 + '밀리초<br><hr>');

var next = new Date(2017, 7, 15, 12, 12, 12);

document.write('next.toLocaleString() : '+ next.toLocaleString() + '<br>');

**</script>**

**</body>**

**</html>**

****

<그림 5-1> 예제 7-5 결과

Data 객체는 시간 정보를 담는 객체이다. ‘Date(연, 월, 일, 시, 분, 초)로 초기화 하여 객체를 생성하며 아무런 값이 없을 경우 현재 시간 정보를 가진 객체가 생성된다. Date 객체의 메소드를 사용하면 연, 월, 일 등의 값을 출력할 수 있다. <그림 5-1>에서 Date 객체를 사용하여 시간 정보를 출력한 것을 확인할 수 있다.

1. **예제 7-6 방문 시간에 따라 변하는 배경색 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8">**

**<title>**방문 시간에 따라 변하는 배경색**</title>**

**</head>**

**<body>**

**<h3>**페이지 방문 초시간이 짝수이면 violet, 홀수이면 lightskyblue 배경**</h3>**

**<hr>**

**<script>**

var current = new Date();

if (current.getSeconds() % 2 == 0)

document.body.style.backgroundColor = 'violet';

else

document.body.style.backgroundColor = 'lightskyblue';

document.write('현재 시간 : ');

document.write(current.getHours() + '시, ');

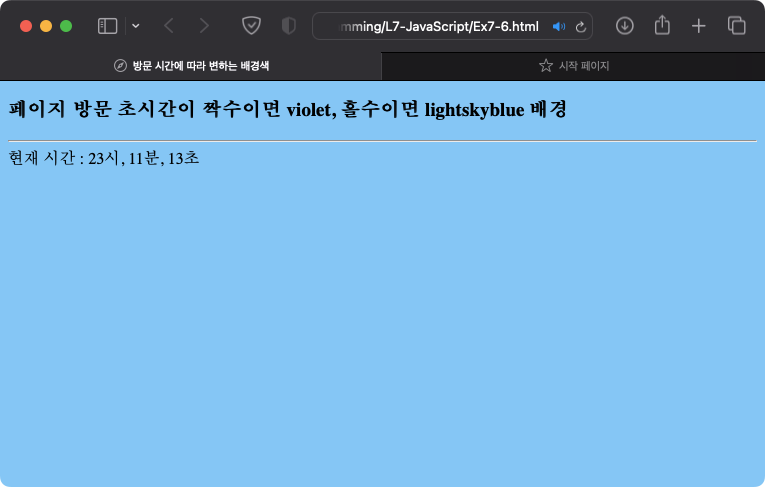
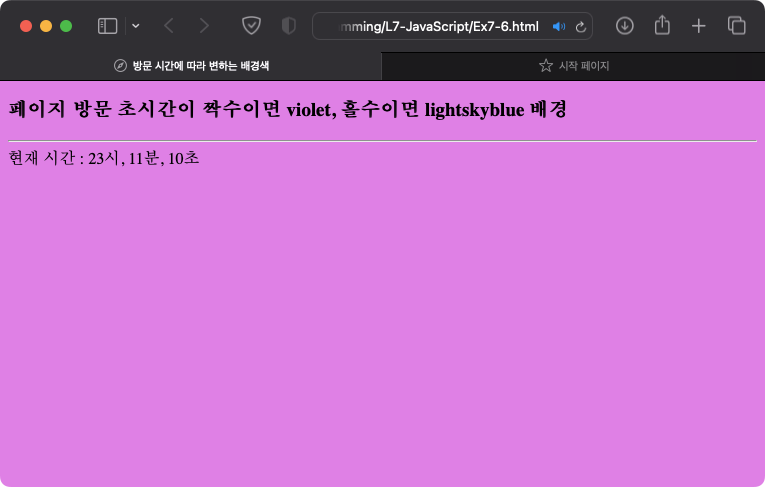
document.write(current.getMinutes() + '분, ');

document.write(current.getSeconds() + '초<br>');

**</script>**

**</body>**

**</html>**

****

<그림 6-1> 예제 7-6 결과

Date 객체의 메소드인 getSeconds()를 사용하여 배경색을 바꾸는 예제이다. 초시간이 짝수일 경우 ‘violet’ 색으로, 홀수일 경우 ‘lightskyblue’ 색으로 바뀐다.

1. **예제 7-7 String 객체의 메소드 활용**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>**String 객체의 메소드 활용**</title>**

**</head>**

**<body>**

**<script>**

var a = new String('Boys and Girls');

var b = '!!';

document.write('a : ' + a + '<br>');

document.write('b : ' + b + '<br><hr>');

document.write(a.charAt(0) + '<br>');

document.write(a.concat(b, '입니다') + '<br>');

document.write(a.indexOf('s') + '<br>');

document.write(a.indexOf('And') + '<br>');

document.write(a.slice(5, 8) + '<br>');

document.write(a.substr(5, 3) + '<br>');

document.write(a.toUpperCase() + '<br>');

document.write(a.replace('and', 'or') + '<br>');

document.write(' kitae '.trim() + '<br><hr>');

var sub = a.split(' ');

document.write('a를 빈칸으로 분리<br>');

for(var i=0; i<sub.length; i++)

document.write('sub' + i + '=' + sub[i] + '<br>');

document.write('<hr>String 메소드를 실행 후 a와 b 변함 없음<br>');

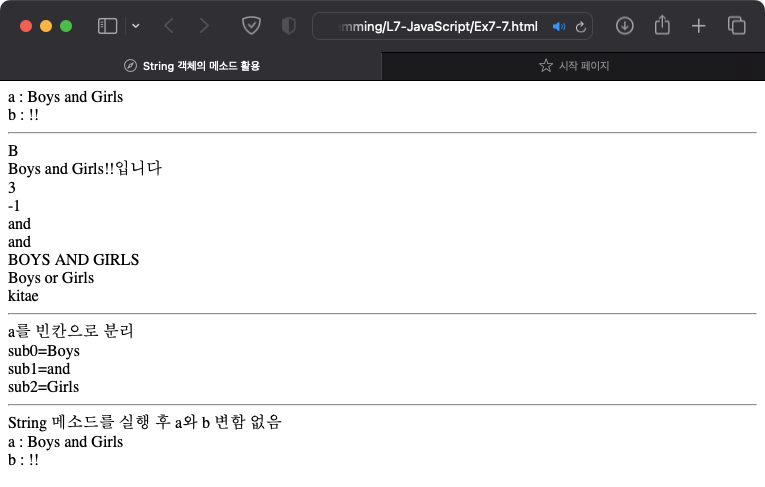
document.write('a : '+ a +'<br>');

document.write('b : '+ b +'<br>');

**</script>**

**</body>**

**</html>**

****

<그림 7-1> 예제 7-7 결과

String 객체를 사용하여 문자열을 담을 수 있다. String 객체는 생성되면 수정이 불가능하며 length로 객체 길이를 출력할 수 있다. 또한 []를 사용하여 배열처럼 사용할 수 있다. String 객체는 다양한 메소드를 가지고 있다. <그림 7-1>에서 charAt() 메소드는 해당 인덱스의 문자를 출력한다. concat()는 문자열을 합쳐 출력하는 메소드이다. indexOf()는 해당 문자의 index를 출력하며 없을 경우 -1을 출력한다. slice()메소드는 Array의 메소드와 마찬가지로 문자열을 잘라 출력한다. substr()메소드는 slice()와 같은 작업을 하지만 종료 index가 아닌 출력하고자 하는 범위를 작성한다. toUpperCase()메소드는 문자열을 대문자로 출력한다. replace()는 특정 문자를 변경하는 메소드이며 trim()메소드는 앞뒤 여백을 없애는 메소드이다. split()메소드를 사용하면 지정한 문자로 문자열을 분리할 수 있다. String 메소드를 실행하여도 각 문자열의 값은 변화가 없다.

1. **예제 7-8 Math를 이용한 구구단 연습**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>**Math를 활용한 구구단 연습**</title>**

**<script>**

function randomInt() {

return Math.floor(Math.random() \* 9) + 1;

}

**</script>**

**</head>**

**<body>**

**<script>**

var ques = randomInt() + '\*' + randomInt();

var user = prompt(ques + ' 값은 얼마입니까?', 0);

if (user == null) {

document.write('구구단 연습을 종료합니다.');

} else {

var ans = eval(ques);

if (ans == user)

document.write('정답! ');

else document.write('아니오! ');

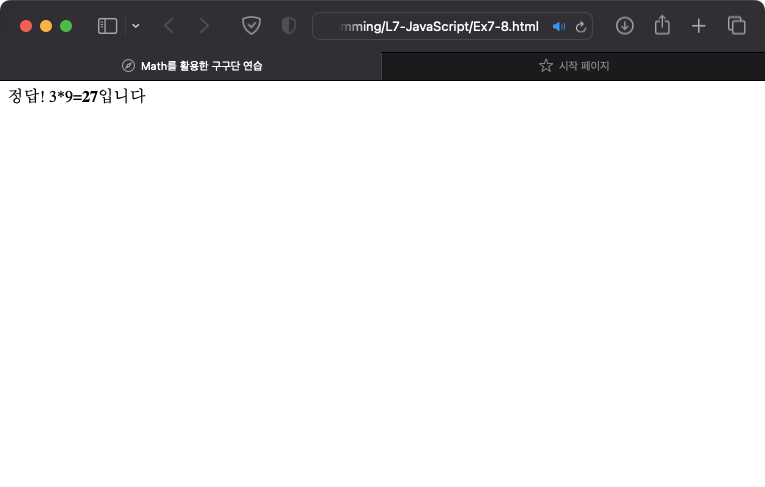
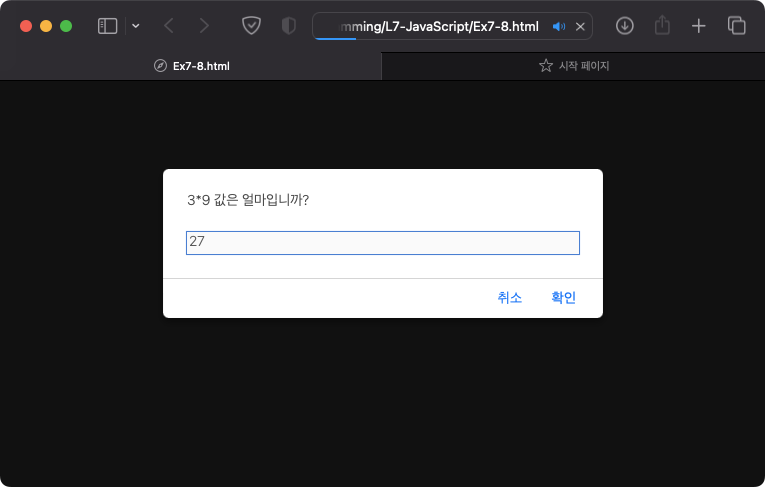
document.write(ques + '=' + '<strong>' + ans + '</strong>입니다<br>');

}

**</script>**

**</body>**

**</html>**

****

<그림 8-1> 예제 7-8 결과

Math 객체는 수학 계산을 위한 속성과 메소드를 제공하며 new 연산자를 사용하여 객체를 생성하지 않고 사용할 수 있다. random() 메소드를 사용하여 0~1사이의 실수를 무작위로 반환한다. sqrt() 메소드를 사용하면 제곱근을 구할 수 있으며 floor() 메소드를 사용하면 소수점 이하를 제거한 정수를 반환한다. <그림 8-1>은 Math 객체를 사용하여 구구단 문제를 내는 것을 확인할 수 있다.

1. **예제 7-9 new Object()로 account 객체 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8" >**

**<title>**new Object()로 사용자 객체 만들기**</title>**

**<script>**

function inquiry() {

return this.balance;

}

function deposit(money) {

this.balance += money;

}

function withdraw(money) {

this.balance -= money;

}

**</script>**

**</head>**

**<body>**

**<h3>**new Object()로 사용자 객체 만들기**</h3><hr>**

**<script>**

var account = new Object();

account.owner = '홍길동';

account.code = '111';

account.balance = 35000;

account.inquiry = inquiry;

account.deposit = deposit;

account.withdraw = withdraw;

document.write('account : ');

document.write(account.owner + ', ');

document.write(account.code + ', ');

document.write(account.balance + '원<br>');

account.deposit(10000);

document.write('10000원 저금 후 잔액은 ' + account.inquiry() + '원<br>');

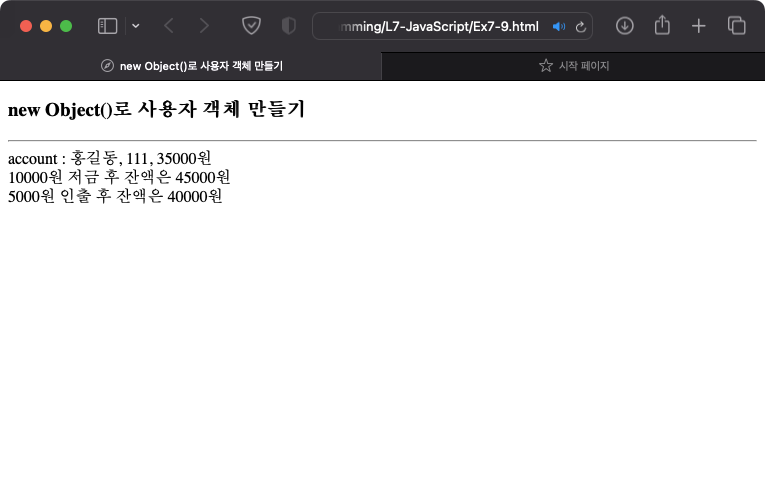
account.withdraw(5000);

document.write('5000원 인출 후 잔액은 '+ account.inquiry() + '원<br>');

**</script>**

**</body>**

**</html>**

****

<그림 9-1> 예제 7-9 결과

사용자가 새로운 타입의 객체를 작성할 수 있다. 객체를 만드는 방법으로 new Object() 이용, 리터럴 표기법 이용, 프로토타입을 만들고 객체 생성의 3가지 방법이 존재한다. 우선 new Object()로 객체를 만들어보면, new Object()로 빈 객체를 생성한다. 이 빈 객체에 프로퍼티를 추가하고 메소드를 추가한다. 이때 메소드는 미리 함수로 만들어 놓는다. <그림 9-1>에서 new Object()를 사용하여 account 객체를 만들고 프로퍼티로 ‘owner’, ‘code’, ‘balance’를 지정하고 메소드로 ‘inquiry’, ‘deposit’, ‘withdraw’를 지정하였다. 지정된 메소드는 <head>태그 내에서 함수로 미리 지정된 것을 확인할 수 있다.

1. **예제 7-10 리터럴 표기법으로 account 객체 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8" >**

**<title>**리터럴 표기법으로 객체 만들기**</title>**

**<script>**

var account = {

owner: '홍길동',

code: '111',

balance: 35000,

inquiry: function () {

return this.balance;

},

deposit: function (money) {

this.balance += money;

},

withdraw: function (money) {

this.balance -= money;

return money;

},

};

**</script>**

**</head>**

**<body>**

**<h3>**리터럴 표기법으로 사용자 객체 만들기**</h3><hr>**

**<script>**

document.write('account : ');

document.write(account.owner + ', ');

document.write(account.code + ', ');

document.write(account.balance + '<br>');

account.deposit(10000);

document.write('10000원 저금 후 잔액은 ' + account.inquiry() + '<br>');

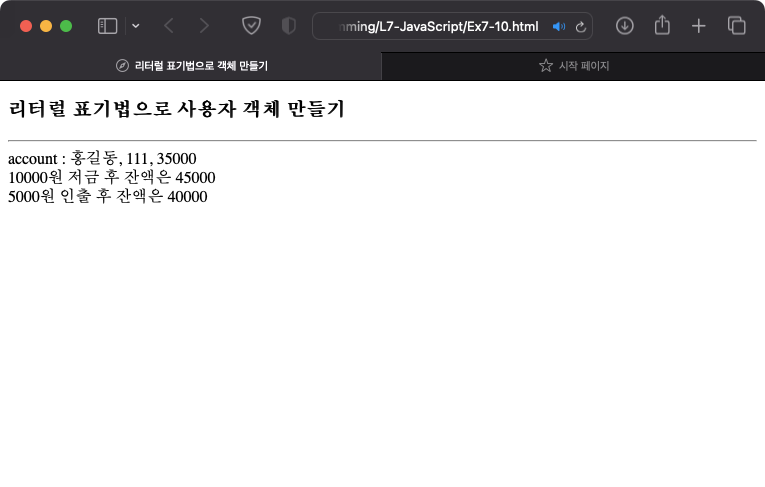
account.withdraw(5000);

document.write('5000원 인출 후 잔액은 '+ account.inquiry() + '<br>');

**</script>**

**</body>**

**</html>**

****

<그림 10-1> 예제 7-10 결과

리터럴 표기법으로 객체를 생성할 수 있으며 중괄호를 이용하여 객체의 프로퍼티와 메소드를 지정한다. <그림 10-1>에서 <head>태그 내에 ‘owner’, ‘code’, ‘balance’ 프로퍼티와 ‘inquiry’, ‘deposit’, ‘withdraw’ 메소드를 중괄호로 묶어 account 객체에 넣고 사용하는 것을 확인할 수 있다.

1. **예제 7-11 프로토타입으로 객체 만들기**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<meta charset="UTF-8" >**

**<title>**프로토타입으로 객체 만들기**</title>**

**<script>**

function Account(owner, code, balance) {

this.owner = owner;

this.code = code;

this.balance = balance;

this.inquiry = function () {

return this.balance;

};

this.deposit = function (money) {

this.balance += money;

};

this.withdraw = function (money) {

this.balance -= money;

return money;

};

}

**</script>**

**</head>**

**<body>**

**<h3>**Account 프로토타입 만들기**</h3><hr>**

**<script>**

var account = new Account('홍길동', '111', 35000);

document.write('account : ');

document.write(account.owner + ', ');

document.write(account.code + ', ');

document.write(account.balance + '<br>');

account.deposit(10000);

document.write('10000원 저금 후 잔액은 ' + account.inquiry() + '<br>');

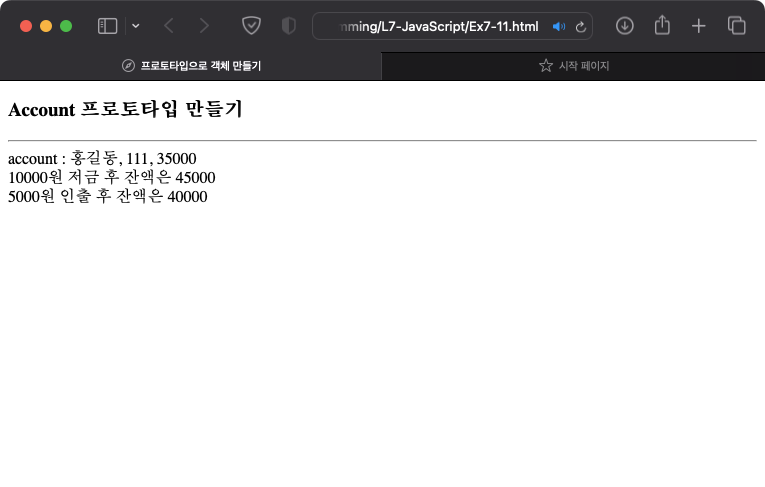
account.withdraw(5000);

document.write('5000원 인출 후 잔액은 '+ account.inquiry() + '<br>');

**</script>**

**</body>**

**</html>**

****

<그림 11-1> 예제 7-11 결과

프로토타입으로 객체를 만드는 것은 C++, JAVA에서 클래스를 만드는 것과 동일하다. <head>태그 내에서 function을 사용하여 생성자를 만들고 생성자 내에서 ‘this’ 키워드를 사용하여 프로퍼티와 메소드를 지정한다. 이후 <body>태그 내에서 new 연산자를 사용하여 객체를 생성한다. <그림 11-1>에서 Account 생성자를 만들고 new Account()로 객체를 생성하는 것을 확인할 수 있다.

1. **참고문헌**

* W3schools, (n.d.), “JavaScript Tutorial”, <https://www.w3schools.com/js/default.asp>
* TCPSCHOOL, (n.d.), “자바스크립트”, <http://www.tcpschool.com/javascript/intro>