7장 Javascript의 객체지향

1. Javascript에서 객체의 의미

Javascript는 Prototype 기반의 객체지향 언어이다. Javascript에서는 객체를 Prototype 형태로 생성하고 그 안에 기능을 위한 함수나 변수를 추가하는 방법으로 그룹화하는 개념이다.

1) Prototype 기반 언어에서 객체 생성 방법

Prototype 기반 언어에서 객체를 생성하는 방법은 크게 두 가지가 있다. 첫 번째 방법은 Class를 작성하고, 이 Class를 기반으로 객체를 생성하는 방법이고, 두 번째는 아무것도 들어있지 않은 빈 객체를 생성한 후에 이 객체의 기능을 점차 확장해가는 방법이다.

여기서는 두 번째 방법으로 소개한다. 두 번째 방법이 Prototype 기반의 언어를 이해하는데 가장 효과적인 방법이다.

- 빈 객체의 생성 : 아무런 기능이 없는 상태의 빈 객체를 생성한다. 이 상태가 Prototype 이다.

- 변수의 추가 : 빈 객체안에 변수들을 추가한다. 일반적으로 용도에 따라 객체를 생성하고, 변수를 그룹화하기 위해 사용된다.

- 함수의 추가 : 빈 객체에 함수들을 추가한다. 기능은 서로 다르지만, 용도가 비슷한 함수들을 하나의 그룹으로 묶기 위한 단위가 객체이다.

2. Javascript에서 객체를 만드는 방법

1) 빈 객체의 생성

빈 객체를 만드는 것은 비어 있는 블록 괄호 '{ }'를 지정하는 것으로 표현한다.

```
var 객체명 = { }
```

2) 변수의 추가

객체안에 추가되는 변수를 멤버 변수 또는 프로퍼티라고 하며, 변수를 추가하기 위해서는 다음과 같은 형식을 사용한다.

```
객체이름.변수명 = "값";
```

simpleObject1.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			/**
			 * Javascript의 객체 지향
			 * ---------------------------------
			 * 클래스라는 객체의 설계도를 먼저 작성하고, 클래스에서 객체를 생산해내는 Java언어와 다르게,
			 * Javascript는 빈 깡통과 같은 객체를 생성하고, 그 안에 변수나 함수를 추가하는 과정으로 객체 지향을 구현한다.
			 * ---------------------------------
			 * Javascript에서 객체의 의미.
			* --> 변수와 함수를 개발자의 판단에 따라 하나의 그릇에 담아내는 그룹단위
			 * --> 객체 안에 포함된 변수 = 멤버변수
			 * --> 객체 안에 포함된 함수 = 메서드
			 */
			// 비어 있는 객체의 생성
			var people = { };

			// 객체 안에 변수를 포함시키기
			people.name = "홍길동";
			people.gender = "남자";

			// 프로퍼티(=멤버변수)의 사용
		document.write("<h1>" + people.name + "님은 " + people.gender + "입니다.</h1>");
		</script>
	</head>
	<body></body>
</html>
```

3) 함수의 다른 생성 방법

Javascript의 변수는 선언시에 데이터형이 결정되지 않고, 할당할 때 결정되기 때문에 함수를 직접 대입하는 것이 가능하다. 이렇게 생성된 변수는 함수와 동일하게 사용할 수 있다.

valueFunction1.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			// 일반적인 함수의 선언
			function myFunction() {
				document.write("<h1>Hello. Javascript</h1>");
			}

			// Javascript는 함수와 변수가 동급이다.
			// 따라서 함수의 이름을 변수에 대입할 수 있다.
			var value = myFunction;

			// 함수가 대입된 변수는 그 자신이 함수처럼 동작한다.
			value();
		</script>
	</head>
	<body></body>
</html>
```

valueFunction2.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			// 함수를 변수에 대입할 수 있기 때문에,
			// 처음부터 변수에 함수의 내용을 대입하는 것이 가능하다.
			var value = function() {
				document.write("<h1>Hello. Javascript</h1>");
			};
			value();
		</script>
	</head>
	<body></body>
</html>
```

4) 객체에 함수 추가(메서드)

빈 객체에게 프로퍼티를 추가하면서 값 대신 함수를 대입하는 것이다. 이 함수를 메서드라고 한다.

```
객체이름.함수이름 = function(매개변수) {
        실행문;
        return 값;
}
```

5) 객체 안에서 객체의 자원 활용

객체안에 포함된 메서드에서 다른 메서드를 호출하거나, 프로퍼티를 활용하고자 하는 경우 this 키워드를 사용한다. 하지만 메서드 안에서만 유효한 변수는 기존의 방식대로 var 키워드를 사용하여 생성할 수 있다. 멤버변수는 모든 메서드 안에서 서로 고하는 것이 가능하므로 전역 변수라고 한다. 반대로 메서드 안에서 var 키워드를 사용하여 생성되는 변수는 그 메서드 안에서만 유효하므로 지역 변수라고 한다.

```
객체이름.함수이름 = function(매개변수) {
        this.변수이름 = 값;
        var 변수이름 = this.함수이름(매개변수);
        return 값;
}
```

객체안에 멤버 변수와 메서드가 모두 포함시킨 것이다.

simpleObject2.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        /**
                         * 객체 안에 멤버변수와 함수를 포함시키기
                         */
                        // 비어 있는 객체의 생성
                        var people = {};

                        // 객체 안에 변수를 포함시키기
                        people.name = "홍길동";
                        people.gender = "남자";
```

```
                        // 객체 안에 함수를 포함시키기
                        people.sayName = function() {
                                // 객체 안에 포함된 함수에서, 멤버변수에 접근하기 위해서는
                                // 반드시 "this."이라는 특수 예약어를 사용해야 한다.
                                document.write("<h1>" + this.name + "</h1>");
                        };

                        people.sayGender = function() {
                                document.write("<h1>" + this.gender + "</h1>");
                        };

                        people.saySomething = function(msg) {
                                document.write("<h1>" + msg + "</h1>");
                        };

                        people.getName = function() {
                                return this.name;
                        };

                        people.getGender = function() {
                                return this.gender;
                        };

                        people.sayInfo = function() {
                                document.write("<h1>" + this.getName() + "님은 " +
this.getGender() + "입니다.</h1>");
                        };

                        // 객체 안의 메서드 호출예제
                        people.sayName();
                        people.sayGender();
                        people.saySomething("Hello Javascript");
                        people.sayInfo();
                </script>
        </head>
        <body></body>
</html>
```

3. 연습 예제 [간단한 계산기 만들기]

calc.js

```
// 빈 객체의 선언
var calc = {};

// 멤버변수의 추가
calc.x = 0;
calc.y = 0;

// 멤버변수에 값을 변경시키기 위한 메서드 추가
calc.setValue = function(p1, p2) {
	/** 파라미터 값을 멤버변수에게 대입한다. */
	this.x = p1;
	this.y = p2;
};

// 멤버변수 끼리의 덧샘결과를 리턴
calc.plus = function() {
	return this.x + this.y;
};

// 멤버변수 끼리의 뺄샘 결과를 리턴
calc.minus = function() {
	return this.x - this.y;
};

// 덧샘과 뺄샘의 결과를 출력하는 메서드
calc.result = function() {
	/** 어떤 메서드 안에서 같은 객체 안에 존재하는 다른 메서드를 호출하는 경우에도 "this."라는 예약어를 사용해야 한다. */
	var value1 = this.plus();
	var value2 = this.minus();

	document.write("<p>덧샘결과: " + value1 + "</p>");
	document.write("<p>뺄샘결과: " + value2 + "</p>");
};
```

simpleCalc.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
<!-- 객체는 자주 사용하게 될 기능들을 묶어서 외부의 스크립트 파일로 분리해 내는 용도로 주로 사용한다.
이렇게 코딩해 놓은 스크립트 파일은 여러개의 HTML파일에 의해서 참조하여 재사용할 수 있다. -->
		<script type="text/javascript" src="calc.js"></script>
		<script type="text/javascript">
			document.write("<h1>calc</h1>");
			calc.setValue(100, 50);
			calc.result();
		</script>
	</head>
	<body></body>
</html>
```

4. 내장 객체의 개요

내장 객체란 Javascript에서 제공하는 공통 기능들 활용하기 위해 Javascrip에 내장된 수 많은 내장 객체가 있다.

1) 내장 객체의 종류

① 값의 처리를 위한 내장 객체

내장 객체명	설 명
Date	시스템의 현재 날짜, 기각을 조회하거나 계산하기 위한 기능 제공
Array	같은 종류의 변수를 하나로 묶기 위한 배열에 관련된 기능 제공
String	하나의 문자열은 독립된 객체이다. 문자열 안에서 특정 글자가 시작되는 위치, 원하는 내용만 추출하는 등의 기능을 제공
Math	삼각함수, 지수, 로그 등 수학과 관련된 고급 함수 등을 제공

② 웹 브라우저 제어를 위한 내장 객체

내장 객체명	설 명
window	웹 브라우저 창에 대한 모든 상황을 제어하는 최상위 객체이다. 모든 웹 브라우저 제어에 관련된 내장 객체는 window 객체의 하위에 존재한다.
location	URL 정보를 제어하는 객체이다. 페이지 이동, 현재 URL주소 조회, 새로 고침 등의 기

	능를 제공한다.
history	웹 브라우저에 기록되어 있는 히스토리 정보를 제어한다.
navigator	웹 브라우저의 종류를 판단한다.
screen	웹 브라우저 화면에 대한 정보를 알려준다. 변수 값만 있으며 메서드는 포함하고 있지 않다.

③ HTML 문서를 제어하기 위한 내장 객체

내장 객체명	설 명
document	문서에 대한 정보, 즉 HTML 문서의 각 요소들을 제어하는 기능
image	<img> 태그에 대한 속성을 제어하는 기능
form	입력 양식 컴포넌트를 위한 개별 객체들을 포함한다.
frame	웹 페이지 안에 다른 웹 페이지를 포함하는 iframe을 제어하는 기능

5. 값의 처리를 위한 내장 객체

1) String 객체

변수로서 사용하는 문자열 값은 그 자체가 객체이다. 객체안에는 프로퍼티와 메서드들이 포함되어 있다.

① 프로퍼티

length	문자열의 길이를 조회한다.

② 메서드

리턴형 메서드이름(파라미터)	설 명
String(value)	문자열을 생성하기 위한 생성자
String charAt(int)	지정된 위치의 글자를 반환한다.
int indexOf(String)	문자열 앞에서부터 파라미터로 주어진 글자를 검색하여 위치를 알려준다. 없을 경우 -1을 반환한다.
int lastIndexOf(String)	문자열 뒤에서부터 파라미터로 주어진 글자를 검색하여 위치를 알려준다. 없을 경우 -1을 반환한다.
String substring(int, int)	문자열에서 첫 번째 위치부터 두 번째 위치가지 추출한다. 두 번째 파라미터가 없을 경우 끝까지 추출한다.
String toUpperCase()	대문자로 변환한다.
String toLowerCase()	소문자로 변환한다.

③ 생성자

생성자란 객체 생성을 위해 사용되는 특수한 형태의 메서드이다.

```
var 변수명 = new String("문자열");

var language = new String("Korean");
var language = "Korean";    // 위와 같은 기능을 하도록 허용.
```

④ 문자열 검색

특정 문자열이 처음 나타나는 위치는 문자열의 첫 번째 글자의 앞에 커서를 놓고 해당 문자열을 만나는 위치까지 오른쪽으로 커서를 이동시켜 확인할 수 있다.

```
var text = "자바스크립트";
var find = text.indexOf("스");  //  | 자 바 | 스 크 립 트    ->    결과는 2

var language = "javascript";
var find = text.indexOf("a");  // | j    a    v   | ascript     ->   결과는 3, 마지막으로 나타나는 a 의 위치
```

⑤ 문자열 잘라내기

substring(int, int) 메서드는 파라미터로 전달된 두 위치 사이의 문자열을 반환한다.

```
var language = "javascript";
var find = language.substring(0, 4);    // 결과 : java

var language = "javascript";
var find = language.substring(4);    // 결과 : script   지정된 위치부터 끝까지
```

string1.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        // 기능을 확인하기 위한 문자열의 선언
                        var url = "http://www.naver.com/";
                        document.write("<p>문자열 : " + url + "</p>");
```

```
            // 문자열의 글자 수를 리턴한다.
            var len = url.length;
            document.write("<p>문자열의 길이 : " + len + "</p>");

            // 파라미터로 설정된 위치의 글자를 리턴한다.
            var str2nd = url.charAt(2);
            document.write("<p>두번째 글자 : " + str2nd + "</p>");

            // 파라미터로 전달된 글자가 처음 나타나는 위치를 리턴한다.
            var position1 = url.indexOf("/");
            document.write("<p>`/`가 처음 나타나는 위치 : " + position1 + "</p>");

            // indexOf에 파라미터가 두 개인 경우,
        // 두 번째 숫자 값은 첫 번째 파라미터의 글자를 찾기 시작하는 위치를 의미한다.
            var position2 = url.indexOf("/", position1+1);
          document.write("<p>`/`가 두 번째로 나타나는 위치 : " + position2 + "</p>");

            // 파라미터로 전달된 글자가 마지막으로 나타나는 위치를 리턴한다.
            // 단 이 위치를 문자열의 끝에서 부터 세는 것이 아니라,
            // 문자열의 처음부터 센다.
            var position3 = url.lastIndexOf("/");
            document.write("<p>`/`의 마지막 위치 : " + position3 + "</p>");

            // 잘라내기 위한 시작 위치와 끝 위치를 파라미터로 설정한다.
            var substring1 = url.substring(0, 5);
            document.write("<p>문자열 자르기 : " + substring1 + "</p>");

            // 두 번째 파라미터가 없을 경우 7번째 부터 끝까지 자른다.
            var substring2 = url.substring(7);
            document.write("<p>문자열 자르기 : " + substring2 + "</p>");

            // 모든 글자를 대문자로 변환한다.
            var up = url.toUpperCase();
            document.write("<p>모든 글자의 대문자 변환 : " + up + "</p>");

            // 모든 글자를 소문자로 변환한다.
            var low = url.toLowerCase();
            document.write("<p>모든 글자의 소문자 변환 : " + low + "</p>");
        </script>
    </head>
```

```
        <body></body>
</html>
```

2) String 객체의 활용

string2.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        var url = "http://www.naver.com/index.html";
                        document.write("<h1>문자열 : " + url + "</h1>");

                        // 도메인 얻기
                        var p1 = url.indexOf("://");
                        var p2 = url.indexOf("/", p1+3);
                        var domain = url.substring(p1+3, p2);
                        document.write("<h1>도메인: " + domain + "</h1>");

                        // 파일이름 얻기
                        var p3 = url.lastIndexOf("/");
                        var p4 = url.lastIndexOf(".");
                        var file = url.substring(p3+1, p4);
                        document.write("<h1>파일이름: " + file + "</h1>");

                        // 확장자 얻기
                        var p5 = url.lastIndexOf(".");
                        var ext = url.substring(p5+1);
                        document.write("<h1>확장자: " + ext + "</h1>");
                </script>
        </head>
        <body></body>
</html>
```

2) Array(배열) 객체

① 배열 선언

```
var 배열명 = new Array(값1, 값2, … , 값n);
var 배열명 = [값1, 값2, … , 값n ];
```

```
var myArray = new Array("HTML", "CSS", "자바스크립트");
var myArray = ["HTML", "CSS", "자바스크립트"];
```

② 배열 사용

배열의 index는 0부터 시작한다.

```
var language = myArray[0];

document.write("<h>" + myArray[1] + "</h>");

myArray[0] = "자바";
```

array1.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			/**
			 * 배열 : 하나의 값 안에 여러개의 변수를 그룹화 하는 단위
			* 배열이 생성되면 그 안의 값들은 0부터 순서대로 일련번호를 부여 받는다.
			 *	(배열의 인덱스)
			 */

			// 배열의 생성
			var myArray = new Array("홍길동", "자바스크립트", "학생");
			// 아래의 방법도 사용 가능합니다.
			// var myArray = ["홍길동", "자바스크립트", "학생"];

			// 배열에 저장된 값들을 읽기
			document.write("<h1>" + myArray[0] + "</h1>");
```

```
                document.write("<h1>" + myArray[1] + "</h1>");
                document.write("<h1>" + myArray[2] + "</h1>");

                // 배열에 저장된 값 변경(할당)
                myArray[0] = "Hong";
                myArray[1] = "Javascript";
                myArray[2] = "Student";

                document.write("<h1>" + myArray[0] + "</h1>");
                document.write("<h1>" + myArray[1] + "</h1>");
                document.write("<h1>" + myArray[2] + "</h1>");
            </script>
        </head>
        <body></body>
</html>
```

③ 빈 배열 만들기

Javascript에서는 배열의 크기를 지정하지 않으며 무한으로 데이터를 추가할 수 있다.

```
var 배열명 = new Array( );
var 배열명 = [ ];
```

```
var myArray = new Array( );
myArray[0] = "김유신";
myArray[1] = "이순신";
myArray[2] = "강감찬";
```

④ 배열의 크기 조회

```
배열객체이름.length;
```

```
var myArray = new Array( );
myArray[0] = "김유신";
myArray[1] = "이순신";
myArray[2] = "강감찬";

document.write("배열의 길이 : " + myArray.length);      // 결과 : 3
```

array2.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			// 빈 배열의 생성
			var numArray = new Array();
			// 아래의 방법도 사용 가능합니다.
			// var numArray = [ ];

			// 반복문을 통하여 배열의 칸을 확장하고 값을 저장
			for (var i=0; i<10; i++) {
				numArray[i] = i;
			}

			document.write("배열의 길이: " + numArray.length);

			// 반복문을 배열의 길이만큼 반복하도록 설정하고 값을 출력
			for (var i=0; i < numArray.length; i++) {
				document.write("<p>" + numArray[i] + "</p>");
			}
		</script>
	</head>
	<body></body>
</html>
```

⑤ 2차원 배열

가장 많이 사용되는 2차원 배열은 행과 열로 구성된다.

```
var 배열명 = new Array(
	new Array(값1, 값2, … , 값n),
	new Array(값1, 값2, … , 값n)
);
var 배열명 = [
	[값1, 값2, … , 값n ],
	[값1, 값2, … , 값n ]
];
```

⑥ 2차원 배열 사용

```
var generalArray = new Array(
        new Array("김유신", "이순신", "강감찬"),
        new Array("신라", "조선", "고려")
);

var general = generalArray[0][0];

document.write("<h>" + generalArray[1][1] + "</h>");

generalArray[1][0] = "통일신라";
```

⑦ 2차원 배열 크기 조회

2차원 배열 행의 크기

```
배열객체이름.length;
```

n번째 행에 대한 열의 크기

```
배열객체이름[n].length;
```

array3.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        // 2차 배열의 생성
                        var generalArray = [
                                ["김유신", "통일신라", "장군"],
                                ["이순신", "조선", "장군"]
                        ];

                        // myarray.length 는 2차 배열의 줄(행) 수
                        for (var i=0; i<generalArray.length; i++) {
                                // myarray[i].length 는 i번째 줄의 칸(열) 수
                                for (var j=0; j<generalArray[i].length; j++) {
```

```
                                        document.write("<h1>" + generalArray[i][j] + "</h1>");
                                }
                        }
                </script>
        </head>
        <body></body>
</html>
```

⑧ Array와 String간의 관계

String 객체의 split() 메서드는 파라미터로 전달된 문자열을 구분자로 삼아 문자열을 잘라내고, 그 결과를 배열로 변환하여 리턴한다.

```
var general = "김유신, 통일신라, 장군";
var data = general.split(“,”);
```

array4.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        var str = "김유신, 통일신라, 장군";

                        // 콤마(,)를 기준으로 문자열을 잘라내서, 배열로 리턴한다.
                        var data = str.split(",");

                        for (var i=0; i<data.length; i++) {
                                document.write("<h1>" + data[i] + "</h1>");
                        }
                </script>
        </head>
        <body></body>
</html>
```

3) Math 객체

① 프로퍼티

프로퍼티명	설 명
E	자연 로그 밑에 사용되는 오일러 상수값
PI	원주율(3.14159...)
LN10	밑수가 10인 싱용로그(2.302)
LN2	밑수가 2인 자연로그(0.693)
SQRT1_2	0.5의 제곱근(0.707)
SQRT2	2의 제곱근(1.414)

② 메서드

리턴형 메서드명(파라미터)	설 명
int abs(x)	x의 절대값
int exp(x)	E의 x승
int log(x)	로그 함수
int pow(x, y)	x의 y승
int sqrt(x)	x의 제곱근
int random(x)	0.0에서 1.0사이의 실수형 난수
int round(x)	반올림
int floor(x)	소수점 이하 값을 버림
int ceil(x)	소수점 이하 값을 올림
int max(x, y)	x, y 중 큰값
int min(x, y)	x, y 중 작은 값
int sin(int)	sin 값
int cos(int)	cos 값
int tan(int)	tan 값
int asin(int)	아크 sin 값
int acos(int)	아크 cos 값
int atan(int)	아크 tan 값

mathTest.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
```

```
                        /**(1) 두 수 중에서 최대값 */
                        var max = Math.max(100, 123);
                        document.write("<h1>최대값: " + max + "</h1>");

                        /**(2) 두 수 중에서 최소값 */
                        var min = Math.min(100, 123);
                        document.write("<h1>최소값: " + min + "</h1>");

                        /**(3) 원주율 */
                        document.write("<h1>최소값: " + Math.PI + "</h1>");

                        /** (4) 소수점 반올림 */
                        var num1 = 3.7146;
                        document.write("<h1>소수점 반올림: " + Math.round(num1) + "</h1>");

                        /** (5) 소수점 올림과 내림 */
                        document.write("<h1>소수점 올림: " + Math.ceil(num1) + "</h1>");
                        document.write("<h1>소수점 내림: " + Math.floor(num1) + "</h1>");

                        /** (6) 절대값을 반환 */
                        var num2 = -123;
                        document.write("<h1>절대값: " + Math.abs(num2) + "</h1>");

                        /** (7) 난수 발생 */
                        document.write("<h1>난수: " + Math.random() + "</h1>");
                </script>
        </head>
        <body></body>
</html>
```

③ 연습 예제(인증번호 생성)

mathRandom.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        /** 두 수 사이의 난수를 리턴하는 함수 */
```

```
                        function random(n1, n2) {
                                return parseInt(Math.random() * (n2 - n1 + 1)) + n1;
                        }

                        /** 함수의 결과 확인 */
                        var num = random(0, 9);
                        document.write("<h1>0~9 사이의 난수: " + num + "</h1>");

                        /** 함수의 응용 >> 5자리 인증번호 생성 */
                        var auth = "";
                        for (var i=0; i<5; i++) {
                                auth += random(0, 9);
                        }
                        document.write("<h1>인증번호: " + auth + "</h1>");
                </script>
        </head>
        <body></body>
</html>
```

4) Date 객체

① Date 객체 생성

```
var 날짜객체명 = new Date();
```

특정 날짜 생성

```
var 날짜객체명 = new Date(년, 월-1, 일);
```

②주요 메서드

리턴형 메서드명(파라미터)	설 명
int getFullYear()	년도를 반환
int getMonth()	월을 반환(0->1월 , 1->2월, … , 11->12월)
int getDate()	일을 반환
int getDay()	요일을 반환(0->일요일, 1->월요일, … , 6->토요일)
int getHours()	시간을 반환
int getMinutes()	분을 반환
int getSeconds()	초를 반환
int getTime()	1970년 1월 1일 0시 0분 0초부터 현재까지의 시간을 1/1000초 단위로

	반환(TimeStamp)
void setYear(int)	1970년 이상의 년도를 설정
void setMonth(int)	월을 설정(0 ~ 11)
void setDate(int)	일을 설정
void setDay(int)	요일을 설정(0 ~ 6)
void setHours(int)	시를 설정
void setMinutes(int)	분을 설정
void setSeconds(int)	초를 설정
void setTime(int)	TimeStamp값으로 시각을 설정

③ 현재 날짜를 출력

```
var mydate = new Date();

// 년,월,일.시간,분,초를 리턴받기
var yy = mydate.getFullYear();
// 월은 0이 1월 11이 12월을 의미한다. 그러므로 1 증가시켜준다.
var mm = mydate.getMonth() + 1;
var dd = mydate.getDate();
```

④ 현재 요일을 출력

```
// 요일의 이름을 저장하고 있는 배열의 생성
var days = ["일", "월", "화", "수", "목", "금", "토"];

// Date 객체의 생성 --> 이 객체 안에는 기본적으로 현재 시각이 저장되어 있다.
var mydate = new Date();

// 0=일요일~6=토요일의 값이 리턴된다.
var i = mydate.getDay();
var day = days[i];
```

⑤ 현재 시간을 출력

```
var mydate = new Date();

var hh = mydate.getHours();
var mi = mydate.getMinutes();
var ss = mydate.getSeconds();
```

date1.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			// 요일의 이름을 저장하고 있는 배열의 생성
			var days = ["일", "월", "화", "수", "목", "금", "토"];

		   // Date 객체의 생성 --> 이 객체 안에는 기본적으로 현재 시각이 저장되어 있다.
			var mydate = new Date();

			// 년,월,일.시간,분,초를 리턴받기
			var yy = mydate.getFullYear();
			// 월은 0이 1월 11이 12월을 의미한다. 그러므로 1 증가시켜준다.
			var mm = mydate.getMonth() + 1;
			var dd = mydate.getDate();

			// 0=일요일~6=토요일의 값이 리턴된다.
			var i = mydate.getDay();
			var day = days[i];

			var hh = mydate.getHours();
			var mi = mydate.getMinutes();
			var ss = mydate.getSeconds();

		var result = yy + "-" + mm + "-" + dd + " " + day + "요일 " + hh + ":" + mi + ":" + ss;
			document.write("<h1>" + result + "</h1>");
		  </script>
	</head>
	<body></body>
</html>
```

⑥ Date 객체에게 임의의 날짜, 시간을 저장

date2.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
```

```
                .. 생략 ..
                <script type="text/javascript">
                        // 요일의 이름을 저장하고 있는 배열의 생성
                        var days = ["일", "월", "화", "수", "목", "금", "토"];

                    // Date 객체의 생성 --> 이 객체 안에는 기본적으로 현재 시각이 저장되어 있다.
                        var mydate = new Date();

                      // Date객체 안에 저장된 시각을 임의의 날짜로 변경하기 -> 2016년 2월 24일
                        mydate.setYear(2016);
                        mydate.setMonth(2);      // 0부터 시작하므로 2월을 위해서는 1로 설정한다.
                        mydate.setDate(24);
                        mydate.setHours(10);
                        mydate.setMinutes(16);
                        mydate.setSeconds(30);

                        // 년,월,일.시간,분,초를 리턴받기
                        var yy = mydate.getFullYear();
                        // 월은 0이 1월 11이 12월을 의미한다. 그러므로 1 증가시켜준다.
                        var mm = mydate.getMonth() + 1;
                        var dd = mydate.getDate();

                        // 0=일요일~6=토요일의 값이 리턴된다.
                        var i = mydate.getDay();
                        var day = days[i];

                        var hh = mydate.getHours();
                        var mi = mydate.getMinutes();
                        var ss = mydate.getSeconds();

              var result = yy + "-" + mm + "-" + dd + " " + day + "요일 " + hh + ":" + mi + ":" + ss;
                        document.write("<h1>" + result + "</h1>");
                </script>
        </head>
        <body></body>
</html>
```

⑦ 두 날짜의 차이 – TimeStamp값 사용

TimeStamp란 1970년 1월 1일 자정부터 현재까지 지난 시간을 초 단위로 바꾼 값이다. getTime() 메소드를 통해 Date 객체가 담고 있는 시각을 1/1000초 단위의 TimeStamp 형태로 변환하여 리턴한다.

date3.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			/** 날짜 객체 */
			var theday = new Date(2016, 1, 1);
			var today = new Date();
			var cnt = today.getTime() - theday.getTime();
			// Math.floor는 소수점 이하를 절단하는 함수이다.
			var day = Math.floor(cnt / (24 * 60 * 60 * 1000));
			document.write("<h1>올해는 " + day + "일 지났습니다.</h1>");
		</script>
	</head>
	<body></body>
</html>
```

⑧ setInterval() 메서드를 활용한 타이머 예제

setInterval() 메서드는 파라미터로 다른 함수명과 1/1000초 단위의 시간값으로 설정하여 정해진 시간에 한번씩 파라미터로 전달된 함수를 자동 반복하는 메서드이다.

```
setInterval(함수명, 1000);   // 1초에 한번씩 반복
```

interval-Test.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			function printTime() {
				// 현재 시각을 구한다.
				var days = ["일", "월", "화", "수", "목", "금", "토"];
```

```
                                var mydate = new Date();

                                var yy = mydate.getFullYear();
                                var mm = mydate.getMonth() + 1;
                                var dd = mydate.getDate();
                                var i = mydate.getDay();
                                var day = days[i];
                                var hh = mydate.getHours();
                                var mi = mydate.getMinutes();
                                var ss = mydate.getSeconds();

                                // 완성된 현재 시각
            var result = yy + "-" + mm + "-" + dd + " " + day + "요일 " + hh + ":" + mi + ":" + ss;

                                // id속성값이 timer인 요소에게 결과를 출력한다.
                                document.getElementById("timer").innerHTML = result;
                        }

                        function startTimer() {
                                // printTime 함수를 1초에 한번씩 반복해서 자동 호출한다.
                                setInterval(printTime, 1000);
                        }
                </script>
        </head>
        <body onload="startTimer()">
                <h1 id="timer"></h1>
        </body>
</html>
```

⑨ 날짜 연산 기능을 활용하여 D'day를 계산

d-day.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                /** 오늘부터 파라미터로 전달받은 날짜에 대한 d'day를 계산하여 리턴한다. */
                        function getDday(y, m, d) {
```

```
                                // 오늘 날짜
                                var today = new Date();
                                // 파라미터로 받은 날짜
            // 정상적인 날짜값을 받은 경우, Javascript의 객체에 전달할 때는 "-1"처리해 주어야 한다.
                                var dday = new Date(y, m-1, d);
                                // 두 날짜간의 차이를 구한다.
                                var cnt = dday.getTime() - today.getTime();
                        // 남은 날짜는 1시간이라도 1일로 계산해야 하므로, 연산결과를 올림한다.
                                var n = Math.ceil(cnt / (24 * 60 * 60 * 1000));
                                return n;
                        }

                        // 매년 12월 31일까지의 남은 날짜 계산하기
                        var date = new Date();
                        var y = date.getFullYear();
                        // "-1"을 하지 않은 정상적인 날짜를 전달하도록 구현하자.
                        var dday = getDday(y, 12, 31);

                        document.write("<h1>올해는 앞으로 " + dday + "일 남았습니다.</h1>");
                </script>
        </head>
        <body>
        </body>
</html>
```

6. 웹 브라우저 제어를 위한 내장 객체

Javascript를 통해서 할 수 있는 처리를 크게 두가지로 나눈다면, 첫 번째는 웹 브라우저 자체를 제어하는 것이고, 두 번째는 웹 브라우저 안에 표시되는 HTML 컨텐츠를 제어하는 것이다.

1) window 객체

① Javascript의 최상위 객체

window 객체는 Javascript에서 사용되는 모든 객체의 최상위 객체이다. 사용자 정의 객체들도 자동으로 window 객체의 하위객체가 된다.

```
window.내장객체명.함수명([파라미터]);
```

window 객체명을 생략 가능하다.

```
내장객체명.함수명([파라미터]);
```

```
window.document.getElementById("id 속성값");    // window 생략 가능
document.getElementById("id 속성값");
```

② 웹 브라우저의 창 제어

▶ 새로운 창 열기

```
window.open("페이지 URL");
```

```
window.open("페이지 URL", "창 이름", "옵션");
```

▶ 현재 창 닫기

```
window.close();  또는  self.close();
```

```
window.open("http://www.naver.com/", "newWindow", "팝업창 옵션");
```

창의 이름을 지정한 경우는 open() 함수를 여러 번 호출(클릭)하여 열기를 하더라도 단 하나의 창에서 새로 고침만 일어난다.

```
window.open("http://www.naver.com/", "", "팝업창 옵션");
```

창의 이름을 공백으로 두면 open() 함수를 여러 번 호출(클릭)하면 호출한 횟수만큼 창이 열린다.

팝업창 옵션

옵션값	값 지정 방법	설명
toolbar	yes / no	툴바 아이콘의 표시여부 결정
location		주소 표시줄의 표시 여부 결정
status		상태바의 표시 여부 결정
menubar		메뉴 표시줄의 표시 여부 결정
scrollbars		스크롤바의 표시 여부 결정
resizable		창의 크기를 조절 가능하게 할지 여부 결정
idth	픽셀값	창의 폭을 지정
height		창의 높이를 지정

가로폭 300px, 높이 500px, 스크롤바 표시 안함, 툴바 표시 안함, 메뉴바 표시 안함, 상태바 표시 안함, 주소표시줄 표시 안함에 대한 옵션을 지정한다.

```
window.open("http://www.naver.com/", "newWindow", "width=300, height=500, scrollbars=no, toolbar=no, menubar=no, status=no, location=no");
```

③ <a> 태그에 대한 onclick 이벤트 처리

▶href 속성과 onclick 이벤트의 충돌 방지

onclick 속성에서 이벤트 함수를 호출한 뒤, "return false;" 구문을 추가하여 실행을 중단하도록 처리해야 한다.

```
<a href="#" onclick="newOpen();  return false;"> 새 창 열기 </a>
```

▶이벤트 처리의 파라미터 전달

함수에 파라미터가 정의되어 있다면, 이벤트를 통해 함수를 호출하면서 파라미터 값을 전달할 수 있다.

```
function newOpen(msg) {
	alert(msg);
}
```

```
<a href="#" onclick="newOpen("오픈기념 행사");  return false;"> 오픈기념 이벤트 </a>
// 위는 문법 충돌 발생
<a href="#" onclick="newOpen('오픈기념 행사');  return false;"> 오픈기념 이벤트 </a>
```

open.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
	</head>
	<body>
		<h1>새로 열린 창</h1>
		<div>
			<!-- 창 닫기 -->
			<a href="#" onclick="window.close(); return false;">창 닫기</a>
		</div>
	</body>
</html>
```

windowTest.html

```
<!DOCTYPE html>
<html lang="ko">
```

```
	<head>
		.. 생략 ..
		<script type="text/javascript">
			function open1() {
				/** 새 창(혹은 탭) 띄우기 */
				window.open('open.html');
			}

			function open2() {
				/** 클릭할 때 마다 창이 새로 열리는 팝업창 */
				window.open('open.html', ' ', 'width=300, height=500, scrollbars=no,
toolbar=no, menubar=no, status=no, location=no');
			}

			function open3(url) {
				/** 한번 생성된 팝업창을 지속적으로 재 사용하는 팝업창 */
				// 팝업의 주소를 파라미터로 전달받는다.
				window.open(url,'mywin','width=500, height=300, scrollbars=no,
toolbar=no, menubar=no, status=no, location=no');
			}
		</script>
	</head>
	<body>
		<h1>window 객체</h1>
		<h3>open 메소드 확인</h3>
		<div>
			<a href="#" onclick="open1(); return false;">새 창 열기</a>
			<br/>
			<a href="#" onclick="open2(); return false;">팝업 창 열기(1)</a>
			<br/>
			<!-- 파라미터를 포함한 함수의 호출 -->
		       <a href="#" onclick="open3('http://www.naver.com'); return false;">네이버</a>
			<br/>
			<a href="#" onclick="open3('http://www.daum.net'); return false;">다 음</a>
			<br/>
		</div>
	</body>
</html>
```

2) location 객체

location 객체는 웹 브라우저의 주소 표시줄을 제어한다. 주소 표시줄의 현재 주소 혹은 그 일부를 조회할 수 있으며, 변경할 수도 있다. 이 때 주소가 변경되면, 웹 브라우저는 페이지를 이동시킨다.

① location 객체의 기본 속성

location 객체는 웹 브라우저의 주소 표시줄에 표시되는 URL로부터 다양한 정보를 추출할 수 있는 속성 값들을 가지고 있다.

속성 이름	설명
href	문서의 URL 주소
host	주소 표시줄의 URL 주소에서 호스트 이름과 포트를 조회한다.
hostname	호스트 이름이나 도메인을 조회한다.
hash	앵커 이름을 조회한다. URL에 "#"기호와 함께 표시되는 단어를 의미한다.
pathname	경로를 조회한다.
port	포트 번호를 조회한다.
protocol	프로토콜의 종류를 조회한다.
search	URL에 포함된 파라미터를 조회한다.

https://search.naver.com/search.naver**?**where=nexearch**&**query=**java**&sm=top_hty&fbm=1&ie=utf8

http://localhost:80/index.html?변수명=값&변수명=값&변수명=값#앵커명

locationTest.html(웹 서버에서 실행해야 한다.)

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			document.write("<p>문서의 URL주소: " + location.href + "</p>");
			document.write("<p>호스트 이름과 포트: " + location.host + "</p>");
			document.write("<p>호스트 컴퓨터 이름: " + location.hostname + "</p>");
			document.write("<p>앵커이름: " + location.hash + "</p>");
			document.write("<p>디렉토리 이하 경로: " + location.pathname + "</p>");
			document.write("<p>포트번호 부분: " + location.port + "</p>");
			document.write("<p>프로토콜 종류: " + location.protocol + "</p>");
			document.write("<p>URL 조회부분: " + location.search + "</p>");
		</script>
	</head>
	<body>
```

```
                <a href="locationTest.html?a=1&b=2#top">이 링크로 다시 실행하세요.</a>
        </body>
</html>
```

② 다른 페이지로 이동

location.href 값에 다른 페이지 URL을 대입하면 페이지가 이동한다.

pagemove.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        function goNaver() {
                                if (confirm("정말 네이버로 이동하시겠습니까?")) {
                                        location.href= "http://www.naver.com";
                                }
                        }
                </script>
        </head>
        <body>
                <input type="button" value="네이버로 이동하기" onclick="goNaver()" />
        </body>
</html>
```

③ 페이지 새로 고침

location 객체의 reload() 함수는 현재 페이지를 새로 고침 한다.

임의로 생성된 5자리 인증번호를 새로 고침으로 다시 받는다.

refresh.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
                .. 생략 ..
                <script type="text/javascript">
                        /** 두 수 사이의 난수를 리턴하는 함수 */
                        function random(n1, n2) {
```

```
                return parseInt(Math.random() * (n2 - n1 + 1)) + n1;
            }

            /** 5자리의 인증번호를 id값이 "auth"인 요소에게 출력 */
            function authNo() {
                var value = "";
                for (var i = 0; i < 5; i++) {
                    value += random(0, 9);
                }

                document.getElementById("auth").innerHTML = value;
            }

            /** 페이지 새로 고침 */
            function refresh() {
                location.reload();
            }
        </script>
    </head>
    <!-- 페이지 최초 로딩시, authNo() 함수 호출 -->
    <body onload="authNo()">
        <p>
            <!-- strong 태그 안이 자바스크립트 출력 부분 -->
            고객님의 인증번호는 <strong id="auth">00000</strong>입니다.
        </p>
        <!-- 페이지 새로고침 이벤트 호출 -->
        <input type="button" value="인증번호 새로 받기" onclick="refresh()"/>
    </body>
</html>
```

3) history 객체

history 객체는 웹 브라우저의 뒤로 가기, 앞으로 가기 버튼의 기능을 수행하는 back() 함수와 forward() 함수가 있다.

```
history.back();
history.forward();
```

history1.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
	</head>
	<body>
		<h1>History 객체</h1>
		<a href="history1.html">페이지 이동</a><br />
		<a href="#" onclick="history.forward(); return false;">앞 페이지로 이동</a>
	</body>
</html>
```

history2.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
	</head>
	<body>
		<h1>History 객체</h1>
		<a href="#" onclick="history.back(); return false;">이전 페이지로 이동</a>
	</body>
</html>
```

4) navigator 객체

navigator 객체는 웹 브라우저의 정보를 조회하는 속성을 가지고 있다.

속성 이름	설명
appName	웹 브라우저 이름
appCodeName	웹 브라우저 코드명
platform	웹 브라우저가 설치된 시스템의 환경
userAgent	웹 브라우저의 종류와 버전
appVersion	웹 브라우저의 종류

navigator.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
			var info = "<h1>웹 브라우저 정보 확인</h1>";
			info += "<p>브라우저 이름 : " + navigator.appName + "</p>";
			info += "<p>브라우저 코드명 : " + navigator.appCodeName + "</p>";
			info += "<p>플랫폼 정보 : " + navigator.platform + "</p>";
			info += "<p>사용자 정보 : " + navigator.userAgent + "</p>";
			info += "<p>브라우저 버전 : " + navigator.appVersion + "</p>";
			document.write(info);
		</script>
	</head>
	<body></body>
</html>
```

① navigator 객체의 응용

모바일 용과 PC용 브라우저 구분과 브라우저 종류 구분

ismobileTest.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<script type="text/javascript">
		/** 모바일 브라우저이면 true, 그렇지 않으면 false를 리턴하는 사용자 정의 함수 */
			function isMobile() {
				var tmpUser = navigator.userAgent;
				var isMobile = false;

  // userAgent값에 iPhone, iPad, iPod, Android 라는 문자열이 하나라도 검색되면, 모바일로 간주한다.
				if (tmpUser.indexOf("iPhone") > 0 || tmpUser.indexOf("iPad") > 0 ||
tmpUser.indexOf("iPod") > 0 || tmpUser.indexOf("Android ") > 0) {
					isMobile = true;
				}
				return isMobile;
```

```
                }

                var isMobileWeb = isMobile();

                if (isMobileWeb) {
                        document.write("<h1>모바일 웹 브라우저로 접속하셨습니다.</h1>");
                } else {
                        document.write("<h1>PC용 웹 브라우저로 접속하셨습니다.</h1>");
                }
        </script>
    </head>
    <body></body>
</html>
```

▶ 웹 브라우저 종류

browserTest.html

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        .. 생략 ..
        <script type="text/javascript">
            /** 브라우저의 이름을 리턴하는 함수 */
            function getWebBrowserName() {
                // userAgent값을 모두 소문자로 변환하여 변수에 대입
                var agt = navigator.userAgent.toLowerCase();

                // 특정 브라우저의 이름이 검색되는지 여부를 판별하여
              // 특정 이름이 검색될 경우, 브라우저 이름을 의미하는 문자열을 리턴
                if (agt.indexOf("chrome") != -1) {
                    return 'Chrome';
                } else if (agt.indexOf("opera") != -1) {
                    return 'Opera';
                } else if (agt.indexOf("firefox") != -1) {
                    return 'Firefox';
                } else if (agt.indexOf("safari") != -1) {
                    return 'Safari';
                } else if (agt.indexOf("skipstone") != -1) {
                    return 'SkipStone';
```

```
                        } else if (agt.indexOf("msie") != -1 || agt.indexOf("trident") != -1) {
                                return 'Internet Explorer';
                        } else if (agt.indexOf("netscape") != -1) {
                                return 'Netscape';
                        } else {
                                return "Unknown";
                        }
                }
                document.write("<h1>" + getWebBrowserName() + "</h1>");
        </script>
    </head>
    <body></body>
</html>
```

IE 9 이상 버전에서는 Trident 라는 문자열을 내장하고 있다. Microsoft Edge는 Netscape로 확인된다.

5) screen 객체

screen 객체는 장치의 디스플레이 정보를 조회할 수 있는 속성이 있다. 모니터의 해상도 크기

속성 이름	설 명
availHeight	스크린 높이
availWidth	스크린 너비
colorDepth	스크린 색상 수
height	스크린 필셀당 높이
width	스크린 픽셀당 너비
pixelDepth	스크린 픽셀당 비트수(IE 9 미만은 지원안함)

screenTest.html

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        .. 생략 ..
        <script type="text/javascript">
            /** screen 객체의 정보 조회 */
            document.write("<h1>화면 높이 : " + screen.availHeight + "</h1>");
            document.write("<h1>화면 너비 : " + screen.availWidth + "</h1>");
            document.write("<h1>색상  수  : " + screen.colorDepth + "</h1>");
            document.write("<h1>픽셀당 높이 : " + screen.height + "</h1>");
```

```
                document.write("<h1>픽셀당 너비 : " + screen.width + "</h1>");
            document.write("<h1>픽셀당 비트수(IE는 지원안함) : " + screen.pixelDepth + "</h1>");
            </script>
        </head>
        <body></body>
</html>
```

screen 객체를 사용하여 해상도와 관계없이 중앙 좌표를 얻어낸다.

screenCenter.html

```
<!DOCTYPE html>
<html lang="ko">
        <head>
            .. 생략 ..
            <script type="text/javascript">
                /** 화면의 중심 좌표를 가로/세로 형태의 배열로 리턴하는 함수 */
                function getCenterPixcel() {
                    var center = new Array(
                        parseInt(screen.availWidth/2), parseInt(screen.availHeight/2)
                    );
                    return center;
                }

                var screenCenter = getCenterPixcel();
                document.write("<h1>모니터 화면의 중심 좌표: x=" + screenCenter[0] + "px, y=" + screenCenter[1] + "px</h1>");
            </script>
        </head>
        <body></body>
</html>
```

7. HTML 문서를 제어하기 위한 내장 객체

1) document 객체

document 객체는 Javascript의 가장 핵심적인 객체이다. HTML 문서의 구조나 내용을 제어하기 위한 기본 기능이 있고, image, form 등의 모든 객체들이 document 객체의 하위 객체이다.

① HTML 문서의 내용을 제어하는 객체

특정 HTML 요소를 객체 형태로 가져오기

```
var some_object = document.getElementById("Id 속성값");
```

HTML 요소의 제어

```
some_object.innerHTML = "<h1> 문서 객체 </h1>";
var content = some_object.innerHTML;
```

② HTML 요소의 CSS 제어

Javascript에 의해 객체화 된 HTML 요소는 style이라는 하위 객체에 저장한다. style 객체는 CSS 속성에 대응하는 프로퍼티들을 가지고 있으며, 이 프로퍼티에 특정 값을 대입하는 것으로 Javascript를 통한 특정 요소의 CSS제어가 가능하다.

```
var some_object = document.getElementById("Id 속성값");
some_object.style.CSS속성 = "CSS값";
또는
document.getElementById("Id 속성값").style.CSS속성 = "CSS값";
```

Javascript의 CSS 속성에 대한 프로퍼티 종류

배경 속성

Javascript 속성	CSS 속성	설명
backgroundColor	background-color	배경 색상 설정
backgroundImage	background-image	배경 이미지 경로 설정
backgroundPosition	background-position	배경 이미지 위치 설정
backgroundRepeat	background-repeat	배경 이미지 반복 속성 설정

폰트 속성

Javascript 속성	CSS 속성	설명
color	color	글자 색상 설정
font	font	font-style, font-weight, font-size, font-family 순서대로 공백으로 구분하여 설정
fontFamily	font-family	글꼴 이름 설정
fontSize	font-size	글자 크기를 px, pt 등의 단위로 설정
fontStyle	font-style	기울림 여부 설정(normal, italic)

fontWeight	font-weight	굵게 표시 여부 설정(normal, bold)

크기 속성

Javascript 속성	CSS 속성	설명
width	width	요소의 가로 폭 설정
height	height	요소의 세로 높이 설정

형태 속성

Javascript 속성	CSS 속성	설명
display	display	요소의 표시 형태를 block, inline, inline-block, none 값으로 설정
textAlign	text-align	텍스트 정렬 설정(left, center, right)

테두리 속성

Javascript 속성	CSS 속성	설명
borderWidth	border-width	테두리 굵기 설정
borderStyle	border-style	테두리 종류 설정
borderColor	border-color	테두리 색상 설정
border	border	border-width, border-style, border-color 값을 공백으로 설정

여백 속성

Javascript 속성	CSS 속성	설명
paddingLeft	padding-left	박스의 좌측 안쪽 여백 설정
paddingRight	padding-right	박스의 우측 안쪽 여백 설정
paddingTop	padding-top	박스의 상단 안쪽 여백 설정
paddingBottom	padding-bottom	박스의 하단 안쪽 여백 설정
padding	padding	박스의 상단부터 시계방향으로 안쪽 여백 값을 공백으로 구분하여 설정, 값이 하나이면 모든 방향이 공통적으로 적용

위치 속성

Javascript 속성	CSS 속성	설명
marginLift	margin-lift	박스의 좌측 바깥 여백 설정

marginRight	margin-right	박스의 우측 바깥 여백 설정
marginTop	margin-top	박스의 상단 바깥 여백 설정
marginBottom	margin-bottom	박스의 하단 바깥 여백 설정
margin	margin	박스의 상단부터 시계방향으로 바깥 여백 값을 공백으로 구분하여 설정, 값이 하나이면 모든 방향이 공통적으로 적용

③ HTML 요소의 CSS 클래스 제어

특정 element 객체에 포함되어 있는 className 속성은 해당 요소의 CSS 클래스를 설정할 수 있다.

```
var some_object = document.getElementById("Id 속성값");
some_object.className = "CSS 클래스명";
또는
document.getElementById("Id 속성값").className = "CSS 클래스명";
```

document.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<style type="text/css">
			.box1 {
				margin: 10px auto;
				border: 5px solid #ccc;
				padding: 30px;
				text-align: center;
			}

			.box2 {
				margin: 10px auto;
				border: 10px solid #ff00ff;
				background-color: #ff0;
				padding: 25px;
				text-align: left;
			}
		</style>

		<script type="text/javascript">
```

```
                function setFontColor(color) {
                        /** id값이 box인 요소 */
                        var box = document.getElementById("box");
                        // 글자 색상을 파라미터로 전달된 값으로 설정함
                        box.style.color = color;
                }

                function setBgColor(color) {
                // id값이 box인 요소의 CSS background-color 속성을 파라미터값으로 설정
                        document.getElementById("box").style.backgroundColor = color;
                }

                function setWidth(width) {
                        // id값이 box인 요소의 CSS width 속성을 파라미터값으로 설정
                        document.getElementById("box").style.width = width;
                }

                function changeClass(cls) {
                        // id값이 box인 요소의 클래스를 파라미터값으로 적용
                        document.getElementById("box").className = cls;
                }
        </script>
</head>
<body>
        <h1>CSS제어하기</h1>
        <div id="box" class="box1"><h1>테스트 영역 입니다.</h1></div>
        <input type="button" value="(폰트) red" onclick="setFontColor('#f00')" />
        <input type="button" value="(폰트) green" onclick="setFontColor('#0f0')" />
        <input type="button" value="(폰트) blue" onclick="setFontColor('#00f')" />
        <input type="button" value="(배경) red" onclick="setBgColor('#f00')" />
        <input type="button" value="(배경) green" onclick="setBgColor('#0f0')" />
        <input type="button" value="(배경) blue" onclick="setBgColor('#00f')" />
        <input type="button" value="width=50%" onclick="setWidth('50%')" />
        <input type="button" value="width=auto" onclick="setWidth('auto')" />
        <input type="button" value="box1 클래스 적용" onclick="changeClass('box1')" />
        <input type="button" value="box2 클래스 적용" onclick="changeClass('box2')" />
</body>
</html>
```

2) image 객체

image 객체의 기본 프로퍼티

<img id="id 속성 값"src="이미지파일 경로 값" width="이미지파일 가로 넓이 값" height="이미지파일 가로 높이 값" >

```
var some_object = document.getElementById("Id 속성값");
some_object.src = "이미지 경로명";
some_object.width = "300px";
some_object.height = "200px";
```

image.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<style type="text/css">
			/** 목록 정의 초기화 및 목록 박스 좌측 배치 */
			.thumbnail {
				padding: 0; margin: 0;
				list-style: none;
				width: 120px; float: left;
			}

			/** 목록의 각 항목에 대한 크기 및 여백 설정 */
			.thumbnail li {
				width: 100px; height: 100px; padding: 5px 10px;
			}

			/** 썸네일 이미지의 크기 설정 */
			.thumbnail img {
				width: 100px; height: 100px;
			}

			/** 큰 이미지 영역의 배치와 크기, 여백 설정 */
			.view {
				float: left; width: 500px; height: 320px; padding: 5px 0;
			}
```

```
                        /** 큰 이미지의 크기 설정 */
                        .view img {
                                width: 500px; height: 320px;
                        }
                </style>

                <script type="text/javascript">
                        /** 링크에 의해서 호출될 함수 */
                        function setImage(index) {
                                // 이미지의 경로를 담고 있는 배열
                                var image_list = [
                                        'img/1.jpg',
                                        'img/2.jpg',
                                        'img/3.jpg'
                                ];

                                // 이미지 요소의 객체화
                                var image = document.getElementById("target");

                // 객체의 src속성에 배열의 값들 중에서 파라미터로 전달된 위치의 값을 설정한다.
                                image.src = image_list[index];
                        }
                </script>
        </head>
        <body>
                <!-- 각 이미지를 포함하는 링크를 클릭했을 경우, setImage 함수 호출 -->
                <ul class="thumbnail">
                        <li><a href="#" onclick="setImage(0); return false;"><img src="img/1.jpg"
/></a></li>
                        <li><a href="#" onclick="setImage(1); return false;"><img src="img/2.jpg"
/></a></li>
                        <li><a href="#" onclick="setImage(2); return false;"><img src="img/3.jpg"
/></a></li>
                </ul>
                <div class="view">
                        <img src="img/1.jpg" id="target" />
                </div>
        </body>
</html>
```

롤 오버 메뉴(onmouseover, onmouseout 이벤트 사용)

rollover.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<style type="text/css">
			/** 목록정의 요소의 여백 제거, 넓이지정, 기호 제거 */
			ul { padding: 0;   margin: 0;   width: 670px;   list-style: none; }

			/** 목록의 가로 배치 */
			li { float: left; }

			/** float 속성 해제 */
			ul:after { content: '';   display: block;   float: none;   clear: both; }

			/** 이미지 테두리 제거 */
			img { border: none; }
		</style>

		<!-- 사용자 스크립트 블록 -->
		<script type="text/javascript">
		    // <a>태그에서 정의된 onmouseover와 onmouseout에 의해서 호출되는 함수
		    // 이미지 파일의 경로와 처리할 대상 요소의 id값을 파라미터로 전달 받는다.
			function changeImage(image, id) {
				// 파라미터로 전달받은 요소의 id값을 사용하여 image객체를 만든다.
				// 그 객체의 src 속성값에 파리미터로 전달된 이미지 경로를 대입하여,
				// 표시 이미지를 변경한다.
				    document.getElementById(id).src = image;
			}
		</script>
	</head>
	<body>
		<ul>
			<li>
				<a href="#" onmouseover="changeImage('img/jquery_on.jpg',
'jquery');" onmouseout="changeImage('img/jquery.jpg', 'jquery');">
<img id="jquery" src="img/jquery.jpg"/></a>
			</li>
```

```
                        <li>
                                <a href="#" onmouseover="changeImage('img/javascript_on.jpg',
'js');" onmouseout="changeImage('img/javascript.jpg', 'js');">
<img id="js" src="img/javascript.jpg"/></a>
                        </li>
                        <li>
                                <a href="#" onmouseover="changeImage('img/css_on.jpg', 'css');"
onmouseout="changeImage('img/css.jpg', 'css');">
<img id="css" src="img/css.jpg"/></a>
                        </li>
                        <li>
                                <a href="#" onmouseover="changeImage('img/html_on.jpg', 'html');"
onmouseout="changeImage('img/html.jpg', 'html');">
<img id="html" src="img/html.jpg"/></a>
                        </li>
                </ul>
        </body>
</html>
```

3) form 객체

<form> 태그는 모든 사용자에게 입력을 받기 위한 태그이다. 반드시 입력해야 하는 항목에 입력되지 않으면 에러가 발생한다. 그래서 입력 여부에 대한 유효성 검사를 해야 한다.

① 입력 양식의 객체 획득

```
<form name = "joinData">
    <input type="text" name="user_name" id="user_name" />
</form>
```

id 값에 의한 객체 획득

```
var input = document.getElementById("user_name");
```

name 값에 의한 객체 획득

```
var inputForm = document.joinData;
var user_name = inputForm.user_name;
또는
var user_name = document.joinData.user_name;
```

② 입력 값 처리

```
var inputForm = document.joinData;
var user_name = inputForm.user_name;
var user_name_value = user_name.value;
또는
var user_name = document.joinData.user_name.value;
```

입력 값 설정

```
var inputForm = document.joinData;
var user_name = inputForm.user_name;
user_name.value = "홍길동";
또는
document.joinData.user_name.value = "홍길동";
```

입력 여부 검사

```
var inputForm = document.joinData;
var user_name = inputForm.user_name;

if(!user_name.value) {
   // 이름이 입력되지 않았을 때의 처리
}
또는
if(!document.joinData.user_name.value) {
   // 이름이 입력되지 않았을 때의 처리
}
```

③ 선택 항목 처리

```
<form name = "selectForm" >
    <select name="choice">
        <option> -- 선택하세요 -- </option>
        <option value="item1"> 항목1 </option>
        <option value="item2"> 항목2 </option>
    </select>
</form>
```

```
var user_select = document.selectForm;
var select = user_select.choice.selectedIndex;
```

선택 위치 설정

```
// 선택 항목의 index 번호는 0번부터 시작
document.selectForm.choice.selectedIndex = 1;
```

선택 항목 초기값 조회

```
document.selectForm.choice[document.selectForm.choice.selectedIndex].value;
```

선택 여부 검사

```
if(selectForm.choice.selectedIndex < 1) {
    // 선택되지 않았을 때의 처리
}
```

④ 체크된 항목 처리

```
<form name = "selectForm" >
      <input type="radio" name="gender" value="M"> 남자
      <input type="radio" name="gender" value="F"> 여자
</form>
```

```
// radio type의 항목은 배열 인덱스로 접근할 수 있다.
var gender = document.selectForm.gender[0].value;
```

라디오 버튼과 체크 박스의 상태 확인

```
if(!document.selectForm.gender[0].checked) {
    // 선택되지 않았을 때의 처리
}

document.selectForm.gender[1].checked = true;
```

체크 여부 검사

```
// 하나라도 체크되었다면 이 값이 true로 변경된다.
var is_check = false;

// 체크 박스의 배열 길이 만큼 반복
for(var i = 0; i < document.selectForm.gender.length; i++) {
    // i번째 항목이 체크되면
    if(document.selectForm.gender[i].checked) {
        is_check = true;
        break;
    }
}
if(!is_check) {
    // 하나도 체크되지 않았을 때의 처리
}
```

⑤ focus의 지정

focus는 입력 컴포넌트에 입력을 위한 포커스가 지정된 상태를 말한다.

```
document.formName.inputName.focus();
```

⑥ 입력 내용 처리

입력된 내용을 지우기

<input type="reset">

```
document.formName.reset();
```

입력된 내용 전송하기

<input type="submit">

```
document.formName.submit();
```

작성한 내용 전송하기 전에 검사하기

```
<form name="inputJoin" onsubmit="함수명(); return false;" >
```

formTest.html

```
<!DOCTYPE html>
<html lang="ko">
	<head>
		.. 생략 ..
		<style type="text/css">
			/** 하나의 입력 영역을 정의하는 <div>태그 */
			.input_group {
				height: 42px;
				border-bottom: 1px dotted #ccc;
				font: 1em/40px '돋움', 'Helvetica';
			}

			/** 입력양식의 제목을 볼 수 있게 하는 태그 */
			.input_group > label {
				width: 80px; display: inline-block;
			}
		</style>
		<script type="text/javascript">
			function doSubmit() {
				// 폼 객체
				var frm = document.form1;

				// text 요소의 입력여부 검사
				if (!frm.user_name.value) {
					alert("이름을 입력해 주세요.");
					frm.user_name.focus();
					return false;
				}

				// 라디오 버튼의 입력여부 검사.
				// 항목이 두 개 밖에 없으므로, 굳이 반복문을 처리하지는 않았다.
				if (!frm.gender[0].checked && !frm.gender[1].checked) {
					alert("성별을 선택해 주세요.");
					frm.gender[0].focus();
					return false;
				}

				// select 요소에 대한 선택위치 검사
```

```
                        if (frm.job.selectedIndex < 1) {
                                alert("직업을 선택해 주세요.");
                                frm.job.focus();
                                return false;
                        }

                        // 체크박스의 선택여부 검사
                        var chk = false;
                        for (var i = 0; i < frm.hobby.length; i++) {
                                if (frm.hobby[i].checked) {
                                        chk = true;
                                        break;
                                }
                        }

                        if (!chk) {
                                alert("취미를 선택해 주세요.");
                                frm.hobby[0].focus();
                                return false;
                        }

                        // 입력확인하기
                        if (confirm("입력하신 내용이 맞습니까?")) {
                                frm.submit();
                        }
                }
        </script>
</head>
<body>
        <form name="form1" onsubmit="doSubmit(); return false;">
                <fieldset>
                        <legend>회원가입</legend>
                        <div class="input_group first">
                                <label>이름</label>
                                <input type="text" name="user_name" />
                        </div>
                        <div class="input_group">
                                <label>성별</label>
                                <label><input type="radio" name="gender" value="M"> 남
자</label>
```

```
                                        <label><input type="radio" name="gender" value="F"> 여
자</label>
                                </div>
                                <div class="input_group">
                                        <label>직업</label>
                                        <select name="job">
                                                <option>----- 선택하세요 -----</option>
                                                <option value="dev">프로그래머</option>
                                                <option value="pub">퍼블리셔</option>
                                        </select>
                                </div>
                                <div class="input_group">
                                        <label>취미</label>
                                        <label><input type="checkbox" value="축구"
name="hobby"> 축구</label>
                                        <label><input type="checkbox" value="농구"
name="hobby"> 농구</label>
                                        <label><input type="checkbox" value="야구"
name="hobby"> 야구</label>
                                </div>
                                <div class="input_group">
                                        <label> </label>
                                        <input type="submit" name="button" value="제출" />
                                        <input type="reset" name="button2" value="리셋" />
                                </div>
                        </fieldset>
                </form>
        </body>
</html>
```