Chapter 05-2

객체-2 (브라우저 객체 모델)-1

브라우저 객체 모델

BOM (Browser Object Model)

■ 브라우저 객체 모델 (BOM : Browser Object Model)

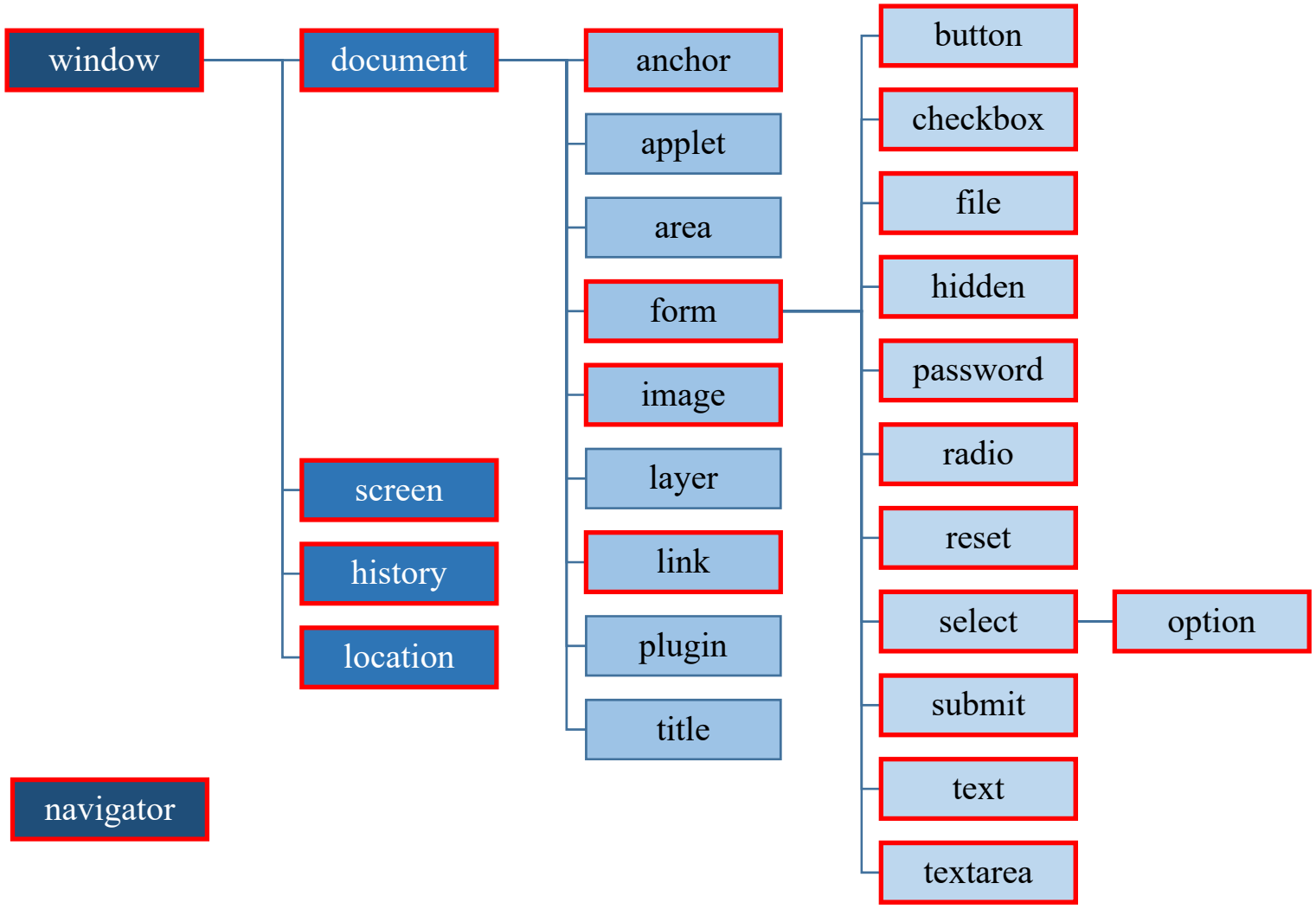
- 브라우저가 제공하는 객체
  - **자바스크립트에서 사용 가능**
    - 실제 자바스크립트에서는 자바스크립트 내장 객체보다 브라우저 객체가 더 많이 사용됨
  - **크게 Window 객체와 Navigator 객체로 구분**

- Window 객체
  - **브라우저 최상위 객체로 하위에 많은 객체를 포함하고 있음**
    - document 객체, screen 객체, location 객체, history 객체
  - **window 객체의 하위 객체**
    - document 객체 : 웹 문서와 그 내용에 대한 처리를 위한 객체 (가장 큰 객체)
    - screen 객체 : 모니터와 출력 영역의 정보를 추출하기 위한 객체
    - location 객체 : URL을 처리하기 위한 객체
    - history 객체 : 히스토리 정보를 유지하고 처리하기 위한 객체

- Navigator 객체
  - **브라우저의 정보를 유지하는 객체**

BOM (Browser Object Model)

■ 브라우저 객체의 계층구조



window 객체

window 객체

■ window 객체

- 객체의 계층 구조 중 최 상위 객체
  - 하위에 document 객체, screen 객체, location 객체, history 객체를 포함하고 있음
- 윈도우에 대한 정보 제공 및 제어 수행
  - 윈도우 객체에서 사용할 수 있는 속성과 메소드를 제공
- 단독으로 사용되기 보다는 주로 파생 객체와 함께 사용
  - 파생 객체와 함께 사용될 경우 window 객체는 생략 가능
  - [예] window.document.write() = document.write()
  - (주의) 파생 객체와 함께 사용되지 않을 경우에는 window 객체 생략 불가능

window 객체

■ window 객체의 메서드

메서드	설명
open()	새로운 윈도를 생성
close()	기존 윈도우를 닫음
alert()	특정 문자열과 [확인] 버튼을 가진 대화상자 출력
prompt()	특정 문자열과 데이터 입력창, [확인]과 [취소] 버튼을 가진 대화상자 출력
confirm()	특정 문자열과 [확인]과 [취소] 버튼을 가진 대화상자 출력
moveTo()/ moveBy()	윈도우를 절대적(moveTo()) 또는 상대적인 위치 (moveBy())로 이동
resizeTo()/ resizeBy()	윈도우를 절대적(resizeTo()) 또는 상대적인 크기 (resizeBy())로 재지정

window 객체 (메서드)

■ open() 메서드

- 새로운 윈도우를 생성하는 메서드

```
open( "열릴 문서", "윈도우 이름", "윈도우 속성" )
```

- **열릴 문서**
  - 생성되는 새로운 윈도우에 포함될 웹 문서의 이름이나 사이트의 주소

- **윈도우 이름**
  - 생성되는 고유한 윈도우 이름 (여러 윈도우를 식별하기 위해 사용)

- **윈도우 속성**
  - 생성되는 윈도우가 가지는 속성을 지정(메뉴바, 툴바, 상태표시줄 등)
  - 윈도우를 구성하는 요소의 출력 여부를 yes나 no로 지정
  - 윈도우의 위치나 크기 등을 픽셀 단위로 지정

window 객체 (메서드)

- 윈도우 속성
  - open() 메서드의 세 번째 인자

메서드	값	설명
width/height	pixel	윈도우의 너비(width)와 높이(height) 지정
left/top	pixel	생성되는 윈도우의 좌측 상단 좌측 좌표(left)와 우측 좌표(top) 지정
location	yes/no	주소 입력 줄 표시 여부 지정 (기본값 : no)
status	yes/no	상태표시줄 표시 여부 지정 (기본값 : no)
scrollbars	yes/no	스크롤 바 표시 여부 지정 (기본값 : no) (브라우저 버전에 따라 출력되지 않을 수 있음)
toolbar	yes/no	툴 바의 표시 여부 지정 (기본값 : no)
resizable	yes/no	윈도우 크기 조절 가능 여부 지정 (기본값 : no)

  - 브라우저마다 특성을 따를 수 있음

window 객체 (메서드)

■ close() 메서드

- 생성되어 있는 윈도우를 닫기 위한 메서드

```
윈도우이름.close()
```

- 다른 윈도우를 닫을 수 있으며, 자기 자신의 윈도우 닫을 수도 있음
  - 자신이 아닌 다른 윈도우를 닫을 경우

```
닫을윈도우이름.close()
```

  - 자신의 윈도우를 닫을 경우

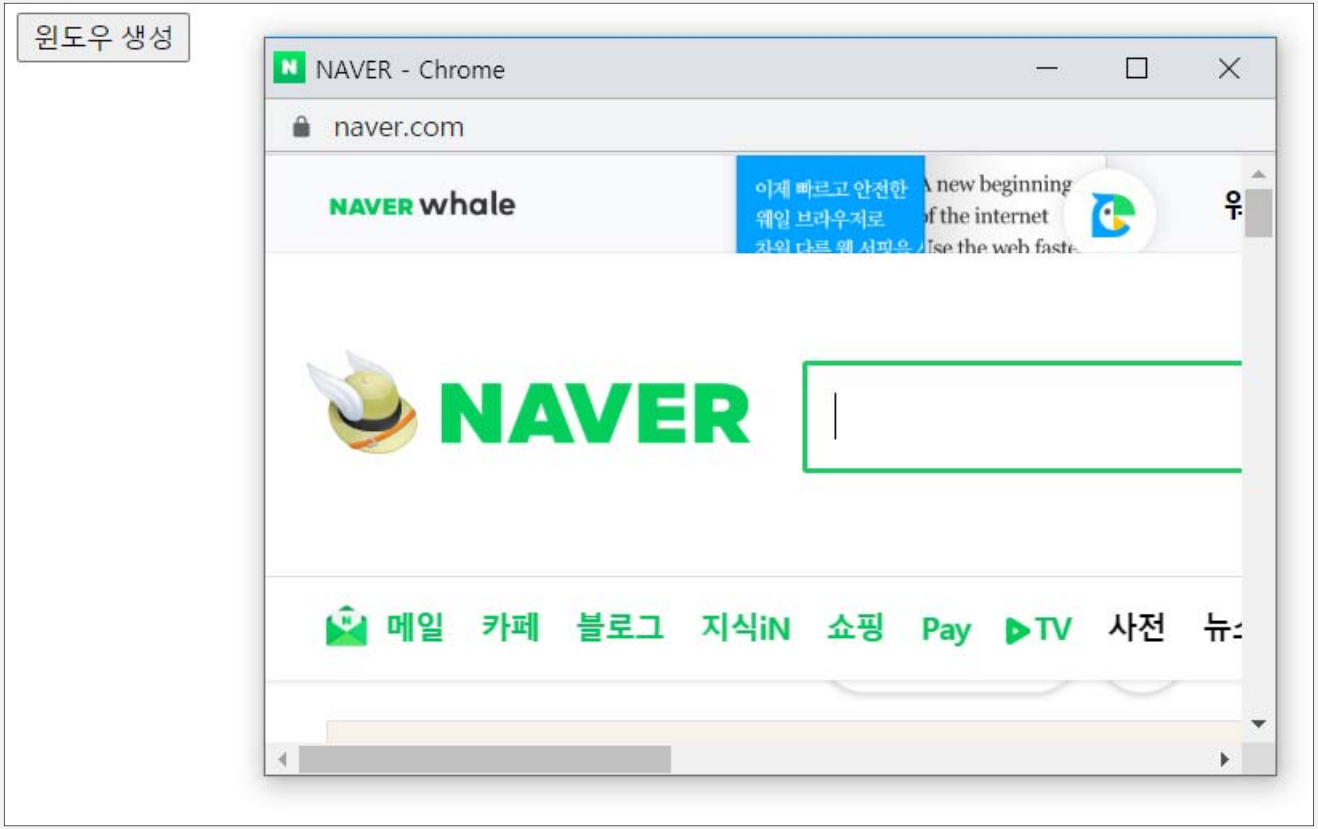
```
window.close()  또는  self.close()
```

window 객체 (메서드)

5-window-open.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function open_win() {
			window.open('https://www.naver.com',  'win1',  'width=500,  height=300,  location=yes,
resizable=no');
		}

	</script>
</head>
<body>
	<input type=BUTTON value="윈도우 생성" onClick="javascript:open_win();">
</body>
</html>
```

window 객체 (메서드)



window 객체 (메서드)

■ open() 메서드와 opener 속성

- open() 메서드를 사용해 윈도우를 생성하면 부모(parent)와 자식(child)의 관계가 성립됨
- 부모(parent) 윈도우
  - open() 메서드를 수행한 문서가 열려 있는 현재 윈도우
- 자식(child) 윈도우
  - open() 메서드에 의해 새로 생성된 윈도우
- 자식 윈도우는 opener 속성을 사용할 수 있음
  - opener 속성은 자식 윈도우 입장에서 자신을 생성한 윈도우(부모 윈도우)를 지칭함

window 객체 (메서드)

5-window-open-parent.htm

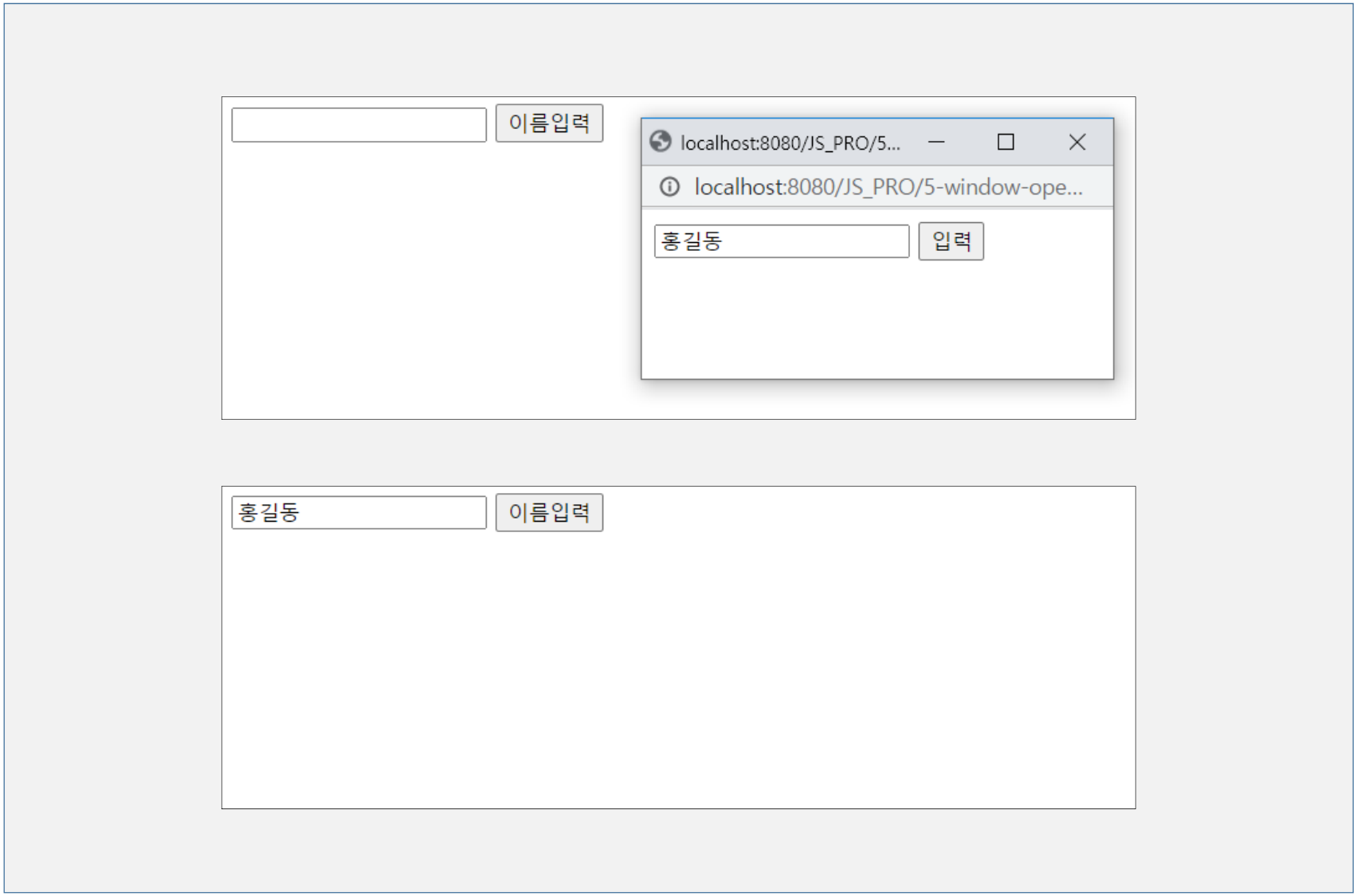
```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function open_win() {
			open("5-window-open-child.htm", "win2", "width=300 height=50");
		}

	</script>
</head>
<body>

	<form NAME="parent">
		<input type=text name="parent_data" size=20 readonly onClick="alert('버튼을 클릭하세요.')">
		<input type=button value="이름입력" onClick="javascript:open_win()">
	</form>

</body>
</html>
```

window 객체 (메서드)

5-window-open-child.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function open_win2() {
			var yourname = eval("document.child.child_data.value");
			opener.document.parent.parent_data.value = yourname;
			self.close();
		}

	</script>

</head>
<body>

	<form NAME="child">
		<input type=text name="child_data" size=20>
		<input type=button value="입력" onClick="javascript:open_win2()">
	</form>

</body>
</html>
```

window 객체 (메서드)



window 객체 (메서드)

■ alert() 메서드

alert (“대화상자에 출력할 문자열”)

- 문자열과 함께 [확인] 버튼을 가진 대화상자를 출력
  - 확인 버튼을 클릭할 때까지 이하의 스크립트는 실행되지 않고 대기함
  - 사용자에게 특정 메시지를 전달하거나 오류의 내용을 통지할 때 사용

5-window-alert.htm

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8">
        <script type="text/javascript">

                alert("안녕하세요");

        </script>

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

window 객체 (메서드)

■ prompt() 메서드

```
변수 = prompt("대화상자에 출력할 문자열", [기본값]);
```

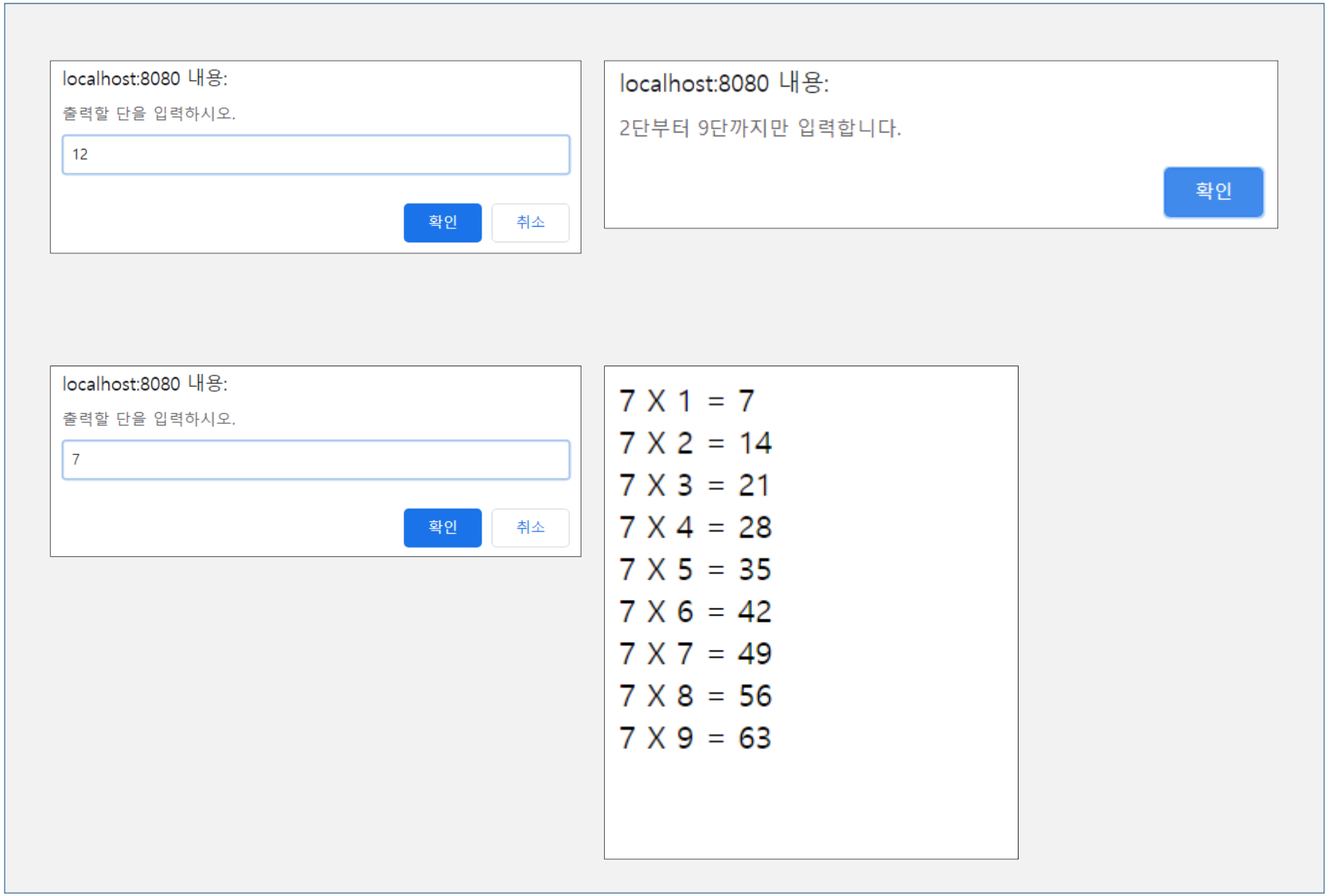
- 사용자로부터 숫자나 문자(열)를 받아 들이기 위해 사용
  - 한 줄 입력 상자, 문자열, [확인] 버튼, [취소] 버튼을 가진 대화상자를 출력
  - [기본값]은 입력 상자에 출력될 기본 값을 의미
- [확인]이나 [취소] 버튼을 클릭할 때까지 이하의 스크립트는 실행되지 않고 대기함
  - [확인] 버튼을 클릭할 경우 입력된 값이 변수에 저장됨
  - [취소] 버튼을 클릭할 경우 대화상자가 사라지고 prompt 함수의 실행이 종료됨

window 객체 (메서드)

5-window-prompt.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		var input=prompt("출력할 단을 입력하시오.", "");
		if(isNaN(input)) {
			alert("숫자를 입력해야합니다");
			exit;
		}

		dan=parseInt(input,10);
		if(dan<2 || dan>9) {
			alert("2단부터 9단까지만 입력합니다.");
			exit;
		}

		for(var i=1; i<=9; i++){
			document.write(dan, " X ", i, " = ", dan*i, "<br>");
		}

	</script>

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

window 객체 (메서드)



window 객체 (메서드)

■ confirm() 메서드

```
변수 = confirm("대화상자에 출력할 문자열");
```

- 문자열과 함께 [확인] 버튼과 [취소] 버튼을 가진 대화상자 출력
  - [확인]이나 [취소] 버튼을 클릭할 때까지 이하의 스크립트는 실행되지 않고 대기함
- confirm()은 항상 특정 변수와 함께 사용
  - [확인] 버튼을 클릭하면 해당 변수에 TRUE가 전달
  - [취소] 버튼을 클릭하면 해당 변수에 FALSE가 전달
- 주로 IF 문과 함께 사용

window 객체 (메서드)

5-window-confirm.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function confirmTest(message) {
			var ok;
			ok = confirm(message);

			if(ok == true)
				alert("회원 가입을 환영합니다.");
			else
				alert("다음에는 가입해주세요.");
		}

	</script>

</head>
	<body onLoad="javascript:confirmTest('가입을 원하시면 확인을 누르세요.')">
	</body>
</html>
```

window 객체 (메서드)

window 객체 (메서드)

■ moveBy() 메서드

```
window.moveBy( X, Y )                    //현재 크롬에서는 수행되지 않음
```

- 상대적 기준으로 윈도우의 이동
  - **윈도우의 위치를 현재 위치를 기준으로 가로 세로 각각 x, y 픽셀만큼 이동**
    - X, Y가 양수인 경우 : 각각 오른쪽과 아래쪽으로 이동
    - X, Y가 음수인 경우 : 각각 왼쪽과 위쪽으로 이동

■ moveTo() 메서드

```
window.moveTo( X, Y )                    //현재 크롬에서는 수행되지 않음
```

- 절대적 기준으로 윈도우의 이동
  - **현재 위치가 아닌 브라우저의 왼쪽 상단을 기준으로 가로 세로 각각 x, y 픽셀만큼 이동**
    - X, Y가 양수인 경우 : 각각 오른쪽과 아래쪽으로 이동
    - X, Y가 음수인 경우 : 잘림

window 객체 (메서드)

5-window-move-to-by.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

	    function open_1() {
	       window.moveBy(100,100);
	     }
	     function open_2() {
	       window.moveTo(100,100);
	     }

	</script>

</head>
<body>
	<input type=button value="가로 세로 100 픽셀 이동" onClick="javascript:open_1();"><br>
	<input type=button value="100,100으로 이동" onClick="javascript:open_2(); return true;"><br>
</body>
</html>
```

window 객체 (메서드)

■ resizeBy() 메서드

```
window.resizeBy( X, Y )                    //현재 크롬에서는 수행되지 않음
```

- 상대적 기준으로 윈도우의 크기를 조절
  - **윈도우의 위치를 현재 위치를 기준으로 가로 세로 각각 x, y 픽셀만큼 조절**
    - X, Y가 양수인 경우 : 현재 크기를 기준으로 지정된 크기만큼 확대
    - X, Y가 음수인 경우 : 현재 크기를 기준으로 지정된 크기만큼 축소

■ resizeTo() 메서드

```
window.resizeTo( X, Y )                    //현재 크롬에서는 수행되지 않음
```

- 절대적 기준으로 윈도우의 크기를 지정
  - **현재 위치가 아닌 브라우저의 왼쪽 상단을 기준으로 가로 세로 각각 x, y 픽셀만큼 크기를 지정**
    - X, Y가 양수인 경우 : 윈도우의 크기를 지정된 크기로 지정
    - X, Y가 음수일 수는 없음

window 객체 (메서드)

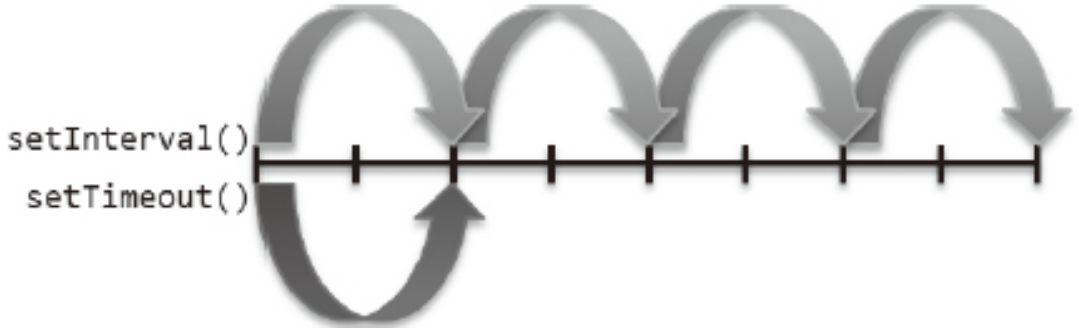
5-window-resize-to-by.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

	    function open_1() {
	       window.resizeBy(50,-50);
	     }
	     function open_2() {
	       window.resizeTo(600, 500);
	     }

	</script>

</head>
<body>
	<input type=button value="가로 50 확대 세로 20 축소" onClick="javascript:open_1();"><br>
	<input type=button value="가로  600 세로 500으로 지정" onClick="javascript:open_2();"><br>
</body>
</html>
```

window 객체

■ window 객체의 타이머 관련 메서드

메서드	설명
setIntervel()	일정 간격으로 특정 명령을 반복 수행
clearInterval()	setIntervel()을 사용해 지정된 반복 작업을 해제
setTimeout()	일정 시간 후 특정 명령을 수행
clearTimeOut()	setTimeout()을 사용해 지정된 작업을 해제



window 객체 (메서드)

■ setTimeout()과 clearTimeout() 메서드

• setTimeout() 메서드

```
시간아이디 = setTimeout( 명령문, 시간 )
```

- 인자로 가지는 명령문을 지정된 시간 후에 한번 실행함 (예약 실행의 개념)
- 시간은 1/1,000ch를 사용 (1초일 경우 1,000을 지정)
  - [예] now = setTimeout("check()", 1000)

• clearTimeout() 메서드

```
clearTimeout( 시간아이디 )
```

- setTimeout() 함수에 의해 수행되는 명령문을 중지시킴
- 중지시킬 명령문은 setTimeout() 함수에서 지정한 시간 ID를 사용
  - [예] clearTimeout(now)
- 한 번만 수행되므로 setTimeout()에 의해 수행되기 전에 취소할 경우 사용

window 객체 (메서드)

5-window-setTimeout.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function start() {
			time_id = setTimeout("alert('Hello')", 2000);
		}

		function stop() {
			clearTimeout(time_id);
		}

	</script>
</head>
<body>
	<form>
		<input type=button  onClick="start()" value="시작">
		<input type=button  onClick="stop()" value="종료" >
	</form>
</body>
</html>
```

window 객체 (메서드)

■ setInterval()과 clearInterval() 메서드

• setInterval() 메서드

```
시간아이디 = setInterval( 명령문, 시간 )
```

- 인자로 가지는 명령문을 지정된 시간마다 반복하여 실행함 (순환의 개념)
- 시간은 1/1,000ch를 사용 (1초일 경우 1,000을 지정)
  - [예] now = setInterval("check()", 1000

• clearInterval() 메서드

```
clearInterval( 시간아이디 )
```

- setInterval() 함수에 의해 수행되는 명령문을 중지시킴
- 중지시킬 명령문은 setInterval() 함수에서 지정한 시간 ID를 사용
  - [예] clearInterval(now)

window 객체 (메서드)

5-window-setInterval-1.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		function start() {
			time_id = setInterval("alert('Hello')", 5000);
		 }

		function stop() {
			clearInterval(time_id);
		}

	</script>
</head>
<body>
	<form>
		<input type=button  onClick="start()" value="시작">
		<input type=button  onClick="stop()" value="종료" >
	</form>
</body>
</html>
```

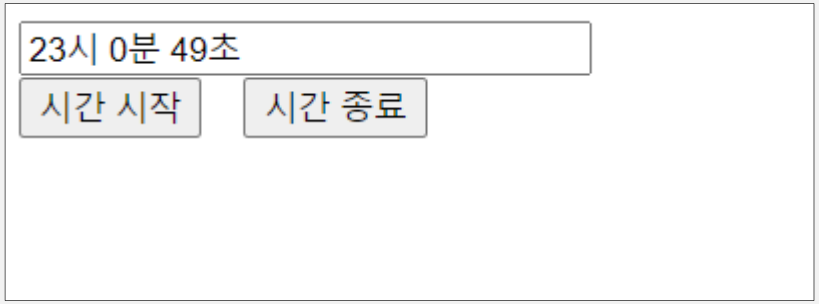
window 객체 (메서드)

5-window-setInterval-2.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

	    function time_start() {
	        today = new Date();
	        now = today.getHours()+"시 " + today.getMinutes()+"분 " + today.getSeconds()+"초";
	        document.myform.timebar.value=now;
	     }

	    function start() {
	          time_id = setInterval("time_start()", 1000);
	    }

	    function stop() {
	        clearInterval(time_id);
	     }

	</script>
</head>
```

window 객체 (메서드)

```
<body>

        <form name="myform">

                <input type=text name=timebar SIZE=30><br>
                <input type=button value="시간 시작" onClick="javascript:start()">  
                <input type=button VALUE="시간 종료" onClick="javascript:stop()">

        </form>

</body>
</html>
```



screen 객체

screen 객체

■ screen 객체

- 사용자의 모니터 정보를 제공하는 객체
  - 속성은 read-only임 (속성값은 수정이 불가능함)
  - 메서드는 존재하지 않음

- screen 객체의 속성

속성	설명
screen.width	전체 화면의 너비 정보를 출력
screen.height	전체 화면의 높이 정보를 출력
screen.availWidth	작업표시줄을 제외한 화면의 너비 정보를 출력
screen.availHeight	작업표시줄을 제외한 화면의 높이 정보를 출력
screen.colorDepth	현재의 모니터가 표현 가능한 컬러의 비트 수를 반환

screen 객체

5-screen.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<script type="text/javascript">

		document.write("width: ", screen.width, "<br>");
		document.write("height: ", screen.height, "<br>");
		document.write("availWidth: ", screen.availWidth, "<br>");
		document.write("availHeight: ", screen.availHeight, "<br>");
		document.write("colorDepth: ", screen.colorDepth);

	</script>
</head>
<body></body>
</html>
```

```
width: 1536
height: 864
availWidth: 1536
availHeight: 834
colorDepth: 24
```

location 객체

location 객체

■ location 객체

- 브라우저 주소 표시줄에 입력된 주소 정보를 추출하거나 다른 문서나 사이트로의 이동을 위한 객체
- 메서드
  - 다른 문서나 사이트로 이동하기 위해 사용
- 속성
  - 웹 브라우저의 주소표시줄에 입력된 주소에 대한 정보를 추출
    - 주소 표시줄의 프로토콜, 호스트 이름, 문서의 위치 등의 정보를 추출할 수 있음
  - 실제 인터넷을 통해 수행되는 경우에만 모든 정보 확인 가능

location 객체 (메서드)

location 객체의 메서드

메서드	설명
reload()	현재 URL에 대한 새로 고침 수행
replace(URL)	인자로 지정된 URL로 이동함
assign(URL)	인자로 지정된 URL로 이동함

location 객체 (메서드)

■ reload() 메서드

```
location.reload();
```

- 현재 페이지에 대해 [새로 고침]을 수행하는 메서드
  - 브라우저에서 reload 기능과 동일한 역할 수행

■ replace() 메서드

```
location.replace('이동할 문서(사이트)');
```

- 인자로 지정한 문서나 사이트로 이동하기 위한 메서드
- location 객체의 href 속성과 동일한 기능 수행
  - [예] `location.replace('https://www.ut.ac.kr')` = `location.href = 'https://www.ut.ac.kr'`

location 객체 (메서드)

■ assign() 메서드

location.assign('이동할 문서(사이트)');

- 인자로 지정한 문서나 사이트로 이동하기 위한 메서드

5-location-1.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
</head>
<body>

	<input type=button value="새로고침" onClick="location.reload()"><br>
	<input type=button value="네이버" onClick="location.href='https://www.naver.com'"><br>
	<input type=button value="한국교통대학교" onClick="location.replace('https://www.ut.ac.kr')"><br>
	<input type=button value="다음" onClick="location.assign('http://www.daum.net')"><br>

</body>
</html>
```

크롬의 경우 replace() 메서드는 브라우저에서 [뒤로 가기] 버튼을 활성화하지 않음

location 객체 (메서드)

■ location 객체의 속성

속성	설명
location.hash	URL에 표시된 해시정보(서버이름 다음의 #이후 문자열)을 반환
location.hostname	URL에 표시된 호스트의 이름을 반환
location.host	URL에 표시된 호스트의 이름과 포르 번호를 반환
location.port	URL에 표시된 포트 번호를 반환
location.pathname	URL에 표시된 서버 이하의 디렉터리 경로를 반환
location.protocol	URL에 표시된 프로토콜을 반환
location.search	URL에 표시된 쿼리 스트링(?이하의 문자)을 반환
location.href	인자를 가지지 않는 경우 : 표시되어 있는 URL의 전체 정보를 반환 URL을 인자로 가지는 경우 : 인자로 지정된 URL로 이동함

location 객체 (메서드)

5-location-2.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">

	<script type="text/javascript">

		document.write("URL 정보 : ", location.href, "<br>");
		document.write("앵커의 이름 : ", location.hash, "<br>");
		document.write("호스트의 이름과 포트 :", location.host, "<br>");
		document.write("호스트의 이름 : ", location.host, "<br>");
		document.write("포트의 이름 : ", location.port, "<br>");
		document.write("디렉토리 이하의 경로 : ", location.pathname, "<br>");
		document.write("프로토콜 이름 : ", location.protocol, "<br>");
		document.write("URL 쿼리 정보 : ", location.search);

	</script>
</head>
<body> </body>
</html>
```

URL 정보 : http://localhost:8080/JS_PRO/5-location-2.htm
앵커의 이름 :
호스트의 이름과 포트 :localhost:8080
호스트의 이름 : localhost:8080
포트의 이름 : 8080
디렉토리 이하의 경로 : /JS_PRO/5-location-2.htm
프로토콜 이름 : http:
URL 쿼리 정보 :

history 객체

history 객체

■ history 객체

- 사용자가 방문한 사이트의 히스토리 정보를 이용해 이동을 수행하기 위한 객체

- history객체의 속성과 메서드

구분		설명
속성	history.length	저장되어 있는 히스토리 정보의 수를 반환
메서드	history.back()	히스토리 정보 중 현재 위치를 기준으로 한 단계 이전 페이지로 이동
	history.forward()	히스토리 정보 중 현재 위치를 기준으로 한 단계 다음 페이지로 이동
	history.go(n)	히스토리 정보 중 현재 위치를 기준으로 인자로 가지는 n단계 만큼 이동 - n이 양수일 경우 : n단계 만큼 다음 페이지 출력 - n이 음수일 경우 : n단계 만큼 이전 페이지 출력 - history.go(1) = history.forward() - history.go(-1) = history.back() - history.go(0) = location.reload() // 새로 고침

history 객체

5-history-1.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
</head>
<body>
	<h3>1 페이지</h3>
	<a href="5-history-2.htm">2 페이지 이동</a>
</body>
</html>
```

5-history-2.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
</head>
<body>
	<h3>2 페이지</h3>
	<a href="5-history-3.htm">3페이지 이동</a>
	<p>
		<button id="prevBtn" onclick="history.back();">이전 페이지</button>
		<button id="nextBtn" onclick="history.forward();">다음 페이지</button>
	</p>
</body>
</html>
```

history 객체

5-history-3.htm

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
</head>
<body>
	<h3>3 페이지</h3>
	<p>
		<button onclick="history.go(-1)">1단계 이전 페이지</button>
		<button onclick="history.go(-2)">2단계 이전 페이지</button>
	</p>

	<script type="text/javascript">
		document.write("히스토리 내 페이지의 수 : " + history.length);
	</script>
</body>
</html>
```

수고하셨습니다