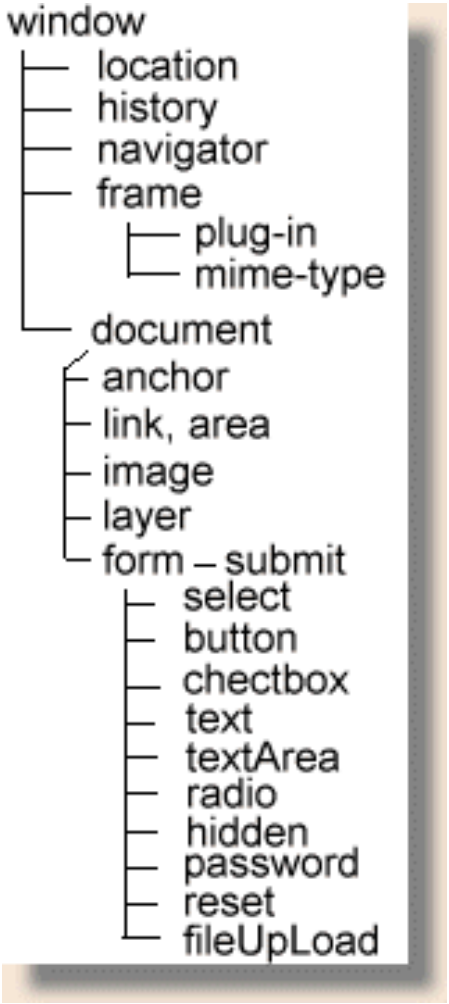
브라우저 내장 객체 1

브라우저 객체 조직도



window 객체

window 객체는 브라우저 내장 객체들의 계층 구조 중에서 가장 상위에 있는 객체로, 우리가 자바스크립트로 하는 모든 작업들이 window 객체 안에서 이뤄지게 된다.

window 객체		
속성	screenX, screenY	넷스케이프에서 창의 위치를 지정
	left, top	익스플로어에서 창의 위치를 지정
	status	브라우저 상태선에 나타날 문자열
	defaultStatus	status 에 지정된 문자열이 없을때 나타날 문자열
	self	자기 자신 객체
	parent	window 객체 간의 상위 객체
	top	window 객체간의 최상위 객체
	frames	window 객체 안에 들어간 프레임들
	opener	open()메소드의 윈도우를 연 문서가 있는 윈도우
메소드	open()	윈도우 하나 열기
	close()	윈도우 닫기
	alert()	메시지를 전달하기 위한 대화상자
	confirm()	확인을 박기 위한 대화상자
	prompt()	사용자로부터 문자열을 입력받기 위한 대화상자
	setTimeout()	지정된 시간동안 기다린후 명령 실행
	clearTimeout()	setTimeout() 해제
	scroll()	행, 열 값을 지정하여 스크롤바 이동
이벤트 핸들러	onLoad	브라우저에서 문서를 읽은후 실행할 명령 지정
	onUnload	현재 문서를 모두 지운후 실행할 명령 지정
	onError	문서를 읽던 중 에러가 발생했을 때
	onBlur	브라우저에서 focus 를 잃었을때
	onFocus	브라우저에서 focus 를 얻었을때

window 객체의 속성

self

self 특성은 말 그대로 자기 자신을 가리키는 것이다. 즉, window 객체가 된다. 이 특성은 한 윈도우 안에서 여러 개의 프레임을 사용할 경우 유용하게 사용된다.

예로 document.wirte() , window.document.write(), self.document.write()는 모두 같은 의미이다.

parent/top

parent 와 top 은 self 특성처럼 하나의 윈도우에 하나의 문서가 읽혀지는 곳에서는 별다른 의미가 없다. 대신 어러개의 프레임이 있는 경우에 유용하다.

opener

opener 특성은 open() 메소드로 윈도우를 연 문서가 포함되어 있는 윈도우를 가리키는 변수이다. 이 특성은 새로 만들어진 윈도우에서 원래의 윈도우에 문자열을 출력하거나 특성을 수정하고자 할때 주로 사용한다.

window 객체의 메소드

open()

윈도우의 모양은 현재 우리가 보고 있는 브라우저의 모양과 동일한 것이 나오게 된다. 물론 툴바, 상태선, 스크롤바 등의 요소들도 마음대로 나타내거나 없을 수 있다.

메소드 원형	**open("URL" , "윈도우명" , "윈도우특성")**
매개 변수	URL : 윈도우에 보여줄 문서의 위치 윈도우명 : 윈도우의 이름,타이틀 윈도우특성 : 윈도우의 특정 지정
리턴값	새로 만들어지는 윈도우 객체

다음은 윈도우의 특성들이다.

윈도우 특징	입력값	설 명
toolbar	Boolean	툴바 메뉴(Back,Forward...)
location	Boolean	문서위치정보(URL)
directories	Boolean	디렉토리 메뉴(what's new...)

status	Boolean	상태선
menubar	Boolean	메뉴
scrollbars	Boolean	스크롤바
resizable	Boolean	윈도우크기 조절 기능
copyhistory	Boolean	히스토리 정보 복사
width	픽셀수	윈도우 바깥 너비
height	픽셀수	윈도우 바깥 높이
fullscreen	Boolean	전체화면으로

close()

close() 는 open() 메소드를 통해 만든 윈도우를 닫는 역할을 한다.

setTimeout()

window 객체의 setTimeout() 메소드는 지정된 시간동안 기다린후 지정된 명령을 실행시키는 기능을 가지고 있다. 예를 들어 주기적으로 시간을 체크하는 시계프로그램, 잠시 안내 메시지를 보여준후에 본래 홈페이지를 보여준다든가 하는 효과들은 이 함수로 구현된 것들이다

> setTimeout(function,time)
> function : 지정된 시간이 지난후 실행할 함수나 명령
> time : 기다릴 시간(단위 : ms, 1/1000 초)
> 리턴값 : 식별자(나중에 setTimeout 상태를 해제할 때 사용)

단위가 ms 이므로 1 초를 기다리게 하려면 1000 을 입력하면 된다.

clearTimeout()

setTimeout() 메소드로 지정한 것을 해제시키는 역할을 한다. 특히 브라우저가 실행되는 동안에는 setTimeout()으로 설정한 것이 유효하기 때문에 다른 페이지로 이동한 후에도 실행될 수가 있다. 그러므로 사용하지 않을때에는 해제시켜주는 것이 좋다.

> clearTimeout(setTimeout_id)
> setTimeout_id : setTimeout() 실행시 리턴된 값(식별자)

3. window 객체의 이벤트

윈도우 객체의 다양한 예제를 통하여 이해하여 보자

open() 예제
창열기 1~6 번까지 눌러보세요

[FrontPage 결과 저장 구성 요소]

```
<html>
<head>
<script language="javascript">
function newwindow() {
  win= open()
  win.document.write("새로운 window")
  win.document.write("<img src='images/1.gif'>")
}
</script>
</head>
<body>
<form>

// 빈문서인 창 열기
<input type="button" value = "창열기 1"
      onclick = "window.open () " >

// js9-2-1.html 문서를 열고 메뉴바를 나타내기
<input type="button"  value = "창열기 2"
      onclick = "window.open  ('exam/newwin1.html' , '' , 'menubar=yes') " >

// svs.or.kr 사이트문서를 열기
<input type="button" value = "창열기 3"
      onclick = "window.open ('http://svs.or.kr') " >

// 창의 크기기정하기
<input type="button" value = "창열기 4"
```

```
      onclick = "window.open   ('','','width=200,height=200, toolbar=no,
location=no,directories=no,status=no,menubar=no') " >

// 윈도우객체 이용하기
<input type="button" value = "창열기 5"
      onclick = "newwindow()">

// 윈도우의 위치 이용하기
<input type="button" value = "창열기 6"
      onclick = "window.open('','',
      'width=200,height=200,top=300,left=200')">
</form>

</body></html>
```

윈도우의 close()예제

close() 예제
- 문서를 불러올때 새로운 창을 열어 새로운창에서 창닫기 버튼을 만들어보자

[FrontPage 결과 저장 구성 요소]

```
<html>
<body onload="window.open('exam/newwin_close.htm')">
새로운 창을 여는 예제입니다.
</body>
</html>
```

newwin_close.htm

```
<html>
<body>
알림판
환영합니다. 웹마스터반
<form>
<input type="button" value="확인" onclick="window.close()">
```

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

윈도우의 스크롤 기능

문서 자동으로 내려가기
- window.scroll(가로, 세로) : 윈도우 스크롤바의 이동

```
<html>
<head>
<script language="javascript" >
   function sl(){
        for(i=1; i<=1000; i++)
             window.scroll(0, i)
   }
</script>
</head>

<body >
<br><br>
<form >
<input type="button" value="내려가기" onClick="sl()">
</form>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<br>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<br>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<br>
</body>
</html>
```

Location 객체

HTML 문서의 주소, 즉 url 을 담고 있는 객체이다. 새로운 값을 할당함으로써 다른 웹페이지를 불러들일 수 있게 되는 것이다.

window.location.href = url

location 예제
단추를 눌러보세요 5 초후 다른 페이지로 이동합니다.

```
<html>
<head>
<script language="javascript">
function nextwin() {
    location.href = "http://svs.or.kr"
}
</script>
</head>

<body onload = "setTimeout('nextwin()',5000)"  >
5 초간만 기다리십시요. 다른 페이지로 이동합니다.
</body></html>
```

frame 객체

프레임을 생성하기위해서는 <frameset>과 <frame> 두가지의 태그를 알아야 한다. 웹페이지를 두개의 프레임으로 나눈다고 하면 아래와 같은 코드가 될 것이다.

```
<html>
<frameset rows="50%,50%">
<frame src="page1.html" name="frame1">
<frame src="page2 html" name="frame2">
```

</frameset>
</html>

프레임과 자바스크립트

계층구조의 최상위 브라우저가 부모 윈도우이고 두개의 프레임이 바로 자식 윈도우가 되는 것이다. 부모 윈도우에서 스크립트를 이용하여 자식 윈도우에 접근하고자 한다면 다음과 같이 쓸 수 있다.

frame2.document.write("부모 윈도우에서 자식 윈도우로 보내는 메세지입니다.");

하나의 자식 프레임에서 또 다른 자식 프레임에 접근할 수 있을까 ? 부모 윈도우는 'parent'가 된다. 부모윈도우를 통해 우회하는 방법을 통해 'frame1' 윈도우에서 'frame2'윈도우로 접근할 수 있는 것이다.

parent.frame2.document.write("안녕? 나 frame 1 이야!");

다양한 프레임을 이용한 링크 적용 예

self.location.href : 자기자신이 속해있는 프레임으로 링크
top.location.href : 프레임과 무관하게 전체 화면이 모두 바뀜
parent.location.href : 자기자신을 포함하고 있는 부모 프레임이 바뀜
parent.framename.location.href : 부모프레임중 선택된 이름을 가진 자식프레임으로 링크

다음의 프레임 예제는 위에서 설명한 예를 완성한것이다.

프레임 예제 **단추를 눌러보세요. 위의 예제입니다.**
main.html <html> <frameset cols="50%,50%">

```
<frame name="frame1" src="page1.htm">
<frame name="frame2" src="page2.htm">
</frameset>
</html>
```

page1.htm

```
<html><body>
저는 프레임 1(page1)입니다.
<form>
<input type="button"  value="프레임 1"
   onclick="parent.frame2.document.write('안녕?나 프레임 1 이야!')">
</form>
</body></html>
```

page2.htm

```
<html><body>
저는 프레임 2(page2)입니다.
<form>
<input type="button"  value="프레임 2"
   onclick="parent.frame1.document.write('안녕?나 프레임 2 야!')">
</form>
</body></html>
```

close()와 open()메소드 적용

도큐먼트에 데이터를 쓰고 나서는 다음과 같이 그 도큐먼트를 닫는다.

```
myWin.document.open();
myWin.document.close();
```

프레임에 새로운 도큐먼트를 만들고 싶다면

```
parent.frame2.document.open();
parent.frame2.document.write("도큐먼트에 쓰는 데이터입니다.");
parent.frame2.document.close();
```

브라우저 내장객체 II

document 객체

document 객체는 window 객체 바로 아래에 있는 것으로 BODY 택 안에 있는 내용들과 직접적으로 연결되어 있다. 그렇기 때문에 document 객체에서는 툴바,상태선,문서위치 정보 등에는 접근할 수가 없다. 그런데 여기서 BODY 택이라고 하지만 정확히 말하면 꼭 그렇다고 볼 수는 없다. 왜냐면 HEAD 안에 있는 TITLE 택의 정보에도 접근할 수 있으며, 문서의 변경 날짜 정보에도 접근할 수 있기 때문이다.

document 객체		
속성	title	문서의 제목
	location	문서의 URL 위치
	lastModified	문서를 마지막으로 수정한 날짜
	referrer	링크로 현재 문서에 왔을 때 이전 문서의 URL 위치
	bgColor	문서의 배경색
	fgColor	문서의 전경색
	linkColor	문서에서 링크를 표시하는 색
	alinkColor	링크를 클릭했을 때 나타나는 색
	vlinkColor	이전에 방문했던 링크를 표시하는 색
	anchors	문서에 있는 표식들의 배열
	forms	문서에 있는 입력 양식들의 배열
	links	문서에 있는 링크들의 배열
	images	문서에 있는 이미지들이 배열
	applets	문서에 있는 자바 애플릿들의 배열
	embeds	문서에 있는 플러그인들의 배열
	cookie	클라이언트 쪽의 PC 에 저장한 정보
메소드	open()	문서에 데이터를 출력하기 위해 준비시키는 것

	close()	문서에 데이터 출력하는 것을 마무리
	clear()	브라어저에서 문서 지우기
	write()	문서에 데이터 출력
	writeln()	문서에 데이터 출력(줄바꾸기 포함)
이벤트	onFocus	문서가 focus 를 얻었을때
핸들러	onBlur	문서가 focus 를 잃었을때

history 객체

window 객체 바로 아래에 있는 것으로, 브라우저의 히스토리 리스트 정보를 저장해 두는 곳이다. history 객체에서 제공하는 여러 메소드들을 이용하여 방금 지나왔던 페이지로 이동할 수 있게 된다.

history 객체		
속성	length	히스토리 리스트에 포함되어 있는 URL 주소의 개수
메소드	back()	히스토리 리스트에서 한단계 앞으로 이동
	forward()	히스토리 리스트에서 한단계 뒤로 이동
	go()	히스토리 리스트에서 임의의 위치로 이동

```
<form>
<input type="button" value = "이전의 문서나 url 은 무엇일까요?
   onclick="history.go(-1)" >
</fomr>
```

navigator 객체

navigator 객체		
속 성	appName	애플리케이션 이름, 예:Netscape
	appVersion	브라우저 현재 버전, 예:2.0 (Win16;I)
	appCodeName	브라우저 현재 코드 이름, 예:Mozilla
	userAgent	브라우저의 현재 User Agent
	mimeTypes	브라우저에서 지원하고 있는 MIME 타입들
	plugins	현재 브라우저에 설치된 플러그인 종류
메소드	javaEnabled()	현재 브라우저에서 자바를 지원하고 있는지 체크

다음의 예제를 실행하여 보자

```
document.write(navigator.appName)
document.write(navigator.appVersion)
document.write(navigator.appCodeName)
document.write(navigator.userAgent)
document.write( navigator.appName + " " + navigator.appVersion)
```

위의 실행된 결과이다.

애플리케이션 이름: Netscape
브라우저 버전: 5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.109 Safari/537.36
브라우저 코드 이름: Mozilla
User Agent : Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/48.0.2564.109 Safari/537.36 애플리케이션 이름: Microsoft Internet Explorer
브라우저 버전: 4.0 (compatible; MSIE 5.0; Windows NT; DigExt)
브라우저 코드 이름: Mozilla
User Agent : Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)

Image 객체

Image 객체는 넷스케이프 3.0 버전부터 추가된 것이기 때문에 그 이하 버전에서는 사용할 수 없다.

Image 객체		
속성	name	image 객체의 이름
	src	이미지 파일의 위치
	lowsrc	이미지 파일의 위치
	border	테두리선 굵기
	height	이미지의 세로길이
	width	이미지의 가로길이
	hspace	이미지의 가로여백
	vspace	이미지의 세로여백
	complete	이미지 전송이 끝났는지 여부
	prototype	image 객체에 특성을 추가하기 위한 것
이벤트 핸들러	onAbort	이미지 전송을 중단시켰을 때
	onError	이미지 전송중 에러가 발생했을 때
	onLoad	이미지가 브라우저에 나타날 때

다음의 이미지 속성을 나타내 보자


image 객체 활용

사용자의 요구가 있는 시점에서 이미지를 읽어들이지 않고 그 전에 모든 이미지를 미리 읽어들임으로써 바로바로 이미지의 치환이 가능하도록 하는 것이다. 이러한 것을 이미지를 미리 읽어들인다하여 Image Preloading 방법이라고 한다. 다음의 코드를 보자.

```
image1 = new Image()
image1.src = "images/image1.gif"
```

이제 이미지를 바꿀 때에는 다음과 같은 코드를 적어주면 된다.

```
document.img.src = image1.src
```

이제 이미지는 브라우저의 캐쉬메모리로부터 가져와 바로 보여줄 수 있다. 즉, 이미지를 미리 읽어들인 것이다.

여기위로 마우스를 올려보세요

```
<html> <head>

<script language="JavaScript">

image1over = new Image();
image1over.src = "images/image1over.gif";
image1out = new Image();
image1out.src = "images/image1out.gif";

</script></head>

<body>
<img src="images/image1out.gif" name="image1"
  onmouseover="document.image1.src=image1over.src"
    onmouseout="document.image1.src=image1out.src" >

<a href="#" onmouseover="document.image1.src=image1over.src"
```

```
   onmouseout="document.image1.src=image1out.src" >
   여기위로 마우스를 올려보세요 </a>

</body>  </html>
```