10

윈도우와 브라우저 관련 객체

강의 목표

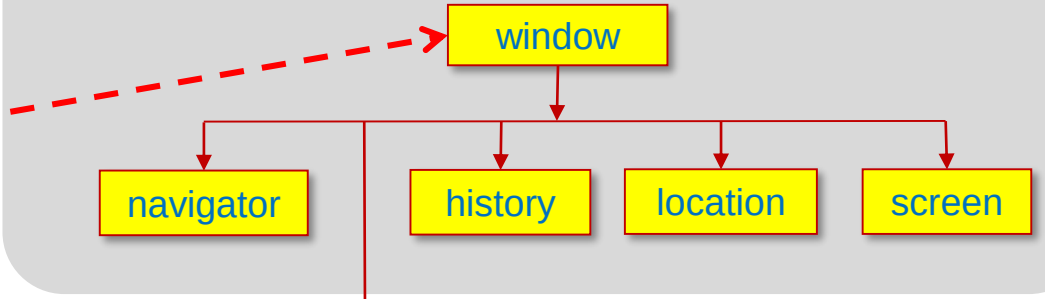
1. BOM, 즉 브라우저 관련 객체 종류를 안다.
2. window 객체를 이해하고 윈도우 열기, 닫기 등을 제어할 수 있다.
3. window 객체의 타이머 기능을 활용할 수 있다.
4. window 객체를 이용하여 프린트, 윈도우 움직이기 등 다양한 제어를 할 수 있다.
5. location 객체로 윈도우에 로드된 문서의 주소를 알고 새 문서를 로드할 수 있다.
6. navigator 객체를 통해 현재 브라우저의 관한 정보를 알아낼 수 있다.
7. screen 객체를 통해 현재 스크린 장치의 해상도를 알아 낼 수 있다.
8. history 객체를 이용하여 지금까지 윈도우에 로드된 웹 페이지로 이동할 수 있다.

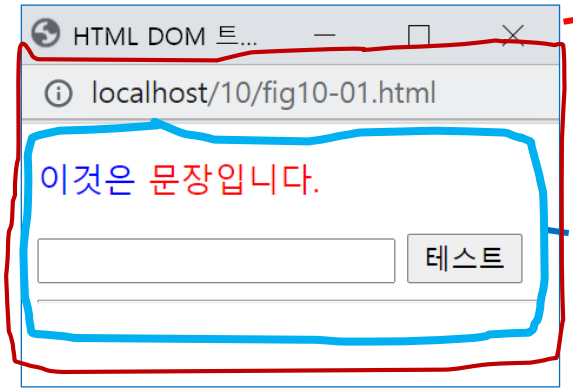
브라우저 관련 객체 개요

- BOM(Browser Object Model) 객체들
  - 자바스크립트로 브라우저를 제어하기 위해 지원되는 객체들
    - HTML 페이지의 내용과 관계없음
  - 브라우저 공통 BOM 객체들과 기능
    - window – 브라우저 윈도우 모양 제어. 새 윈도우 열기/닫기
    - navigator – 브라우저에 대한 다양한 정보 제공
    - history – 브라우저 윈도우에 로드한 URL 리스트의 히스토리 관리
    - location – 브라우저 윈도우에 로드된 HTML 페이지의 URL 관리
    - screen – 브라우저가 실행되고 있는 스크린 장치에 대한 정보 제공
- BOM의 국제 표준이 없다.
  - 브라우저마다 BOM 객체들이 조금씩 다름
  - 브라우저마다 이름이 같은 BOM 객체의 프로퍼티와 메소드 상이

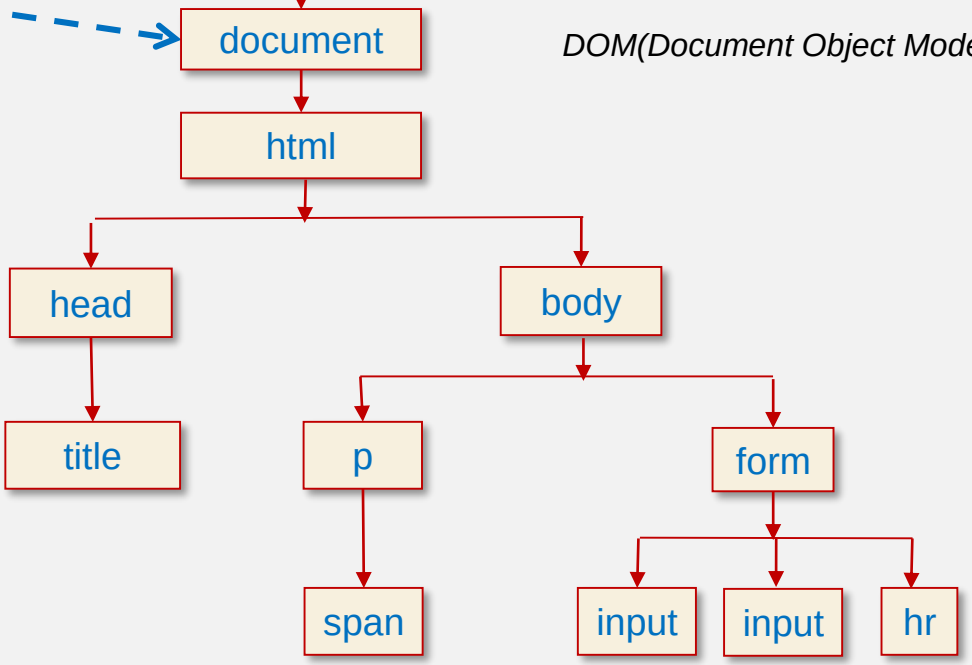
브라우저 관련 객체들

BOM(Browser Object Model)





DOM(Document Object Model)



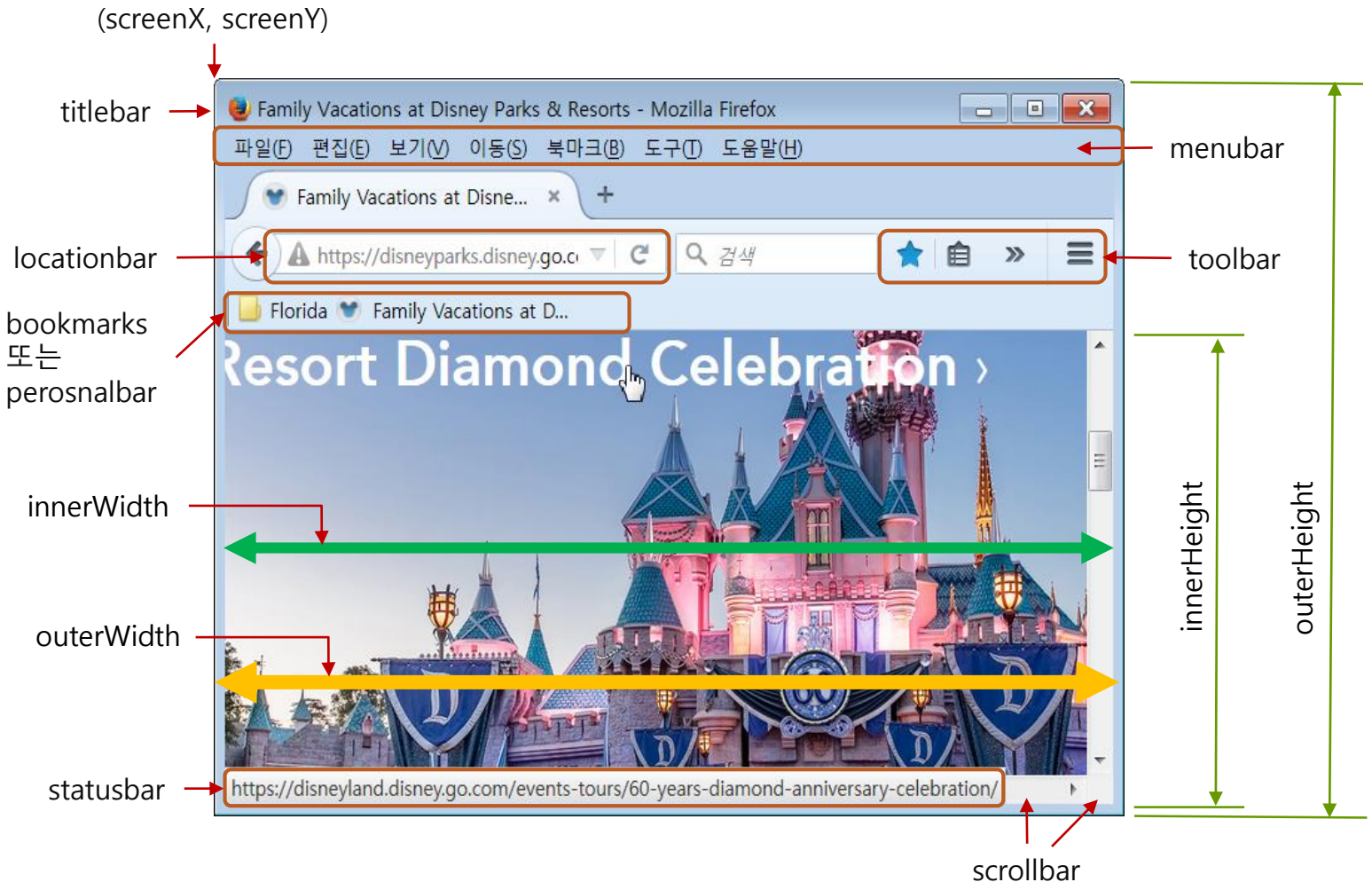

```
<!DOCTYPE html>
<html>
<head>
    <title>HTML DOM 트리</title>
</head>
<body>
<p style="color:blue" >이것은
    <span style="color:red">문장입니다.
    </span>
</p>
<form>
    <input type="text" name="s">
    <input type="button" value="테스트">
    <hr>
</form>
</body>
</html>
```

HTML 문서의 내용과 관련된 객체들

window 객체

- window 객체
  - 열려 있는 브라우저 윈도우나 탭 윈도우의 속성을 나타내는 객체
  - 브라우저 윈도우나 탭 윈도우마다 별도의 window 객체 생성
- window 객체의 생성
  - 3 가지 경우
    - 브라우저가 새로운 웹 페이지를 로드할 때
    - <iframe> 태그 당 하나의 window 객체 생성
    - 자바스크립트 코드로 윈도우 열기 시 window 객체 생성
      - *window.open("웹페이지 URL", "윈도우이름", "윈도우속성");*
- 자바스크립트 코드로 윈도우 객체에 대한 접근
  - window, 혹은 window.self, 혹은 self

윈도우 모양과 window 객체의 프로퍼티



윈도우 열기

- window.open()
  - 윈도우를 새로 열고 웹 페이지 출력
    - 예)

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

  - 3개의 매개변수를 가진 함수

window.open(sURL, sWindowName, sFeature)

- sURL : 윈도우에 출력할 웹 페이지 주소 문자열
- sWindowName : 새로 여는 윈도우의 이름 문자열로서 생략 가능
- sFeature : 윈도우의 모양, 크기 등의 속성들을 표현하는 문자열. 속성들은 빈칸 없이 콤마(‘,’)로 분리하여 작성하며 생략 가능

  - 윈도우 이름(sWindowName)

_blank :	이름 없는 새 윈도우를 열고, 웹 페이지 로드
_parent :	현재 윈도우(혹은 프레임)의 부모 윈도우에 웹 페이지 로드
_self :	현재 윈도우에 웹 페이지 로드
_top :	브라우저 윈도우에 웹 페이지 로드

윈도우 열기 사례

- myWin 이름에 툴바만 가지는 새 윈도우 열고 sample.html 출력

```
window.open("sample.html", "myWin", "toolbar=yes");
```

- 현재 윈도우에 sample.html 출력

```
window.open("sample.html", "_self");
```

- 이름 없는 새 윈도우에 sample.html 출력

```
window.open("sample.html", "_blank");
```

- (10, 10) 위치에 300x400 크기의 새 윈도우 열고 네이버 페이지 출력

```
window.open("http://www.naver.com", "myWin",
        "left=10,top=10,width=300,height=400");
```

- 이름과 속성이 없는 윈도우 열기

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

- 빈 윈도우 생성

```
window.open();
window.open("");
```

```
window.open("", "", "");
window.open("", null, null);
```

윈도우 이름과 윈도우 열기

- 이름 없는 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', '',
            'width=600,height=600')">새 윈도우 열기
</button>
```

  - 버튼을 클릭할 때마다 새 윈도우를 열고 네이버 사이트 출력

- 이름을 가진 윈도우 열기

```
<button onclick="window.open('http://www.naver.com', 'myWin',
            'width=600,height=600')">새 윈도우 열기
</button>
```

  - myWin 이름의 윈도우가 열려 있지 않는 경우
    - 버튼을 클릭하면, myWin이름의 새 윈도우 열고 네이버 출력
  - myWin 이름의 윈도우가 이미 열려 있는 경우
    - 버튼을 클릭하면, 이미 열려있는 myWin이름의 윈도우에 네이버 출력

윈도우 닫기

- 윈도우 닫기
  - 윈도우가 닫히면, 웹 페이지와 함께 윈도우는 사라진다.
  - 현재 윈도우를 닫는 코드

```
window.close();
```

    - 브라우저에 따라 보안의 이유로 이 코드가 작동하지 않음

  - 자신을 만든 윈도우 닫기

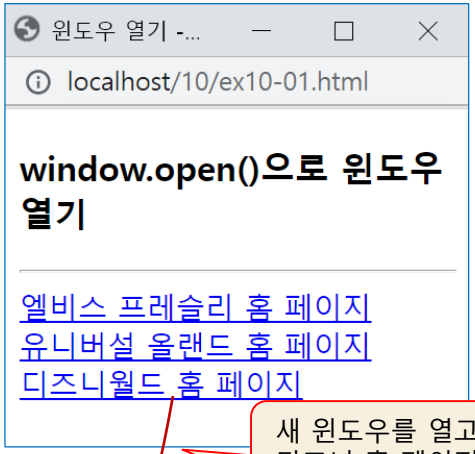
```
let win = window.open(); // 새 윈도우 생성
...
win.close(); // 자신이 만든 윈도우 닫기
```

    - 대부분의 브라우저는 자신이 만든 윈도우의 닫기를 허용

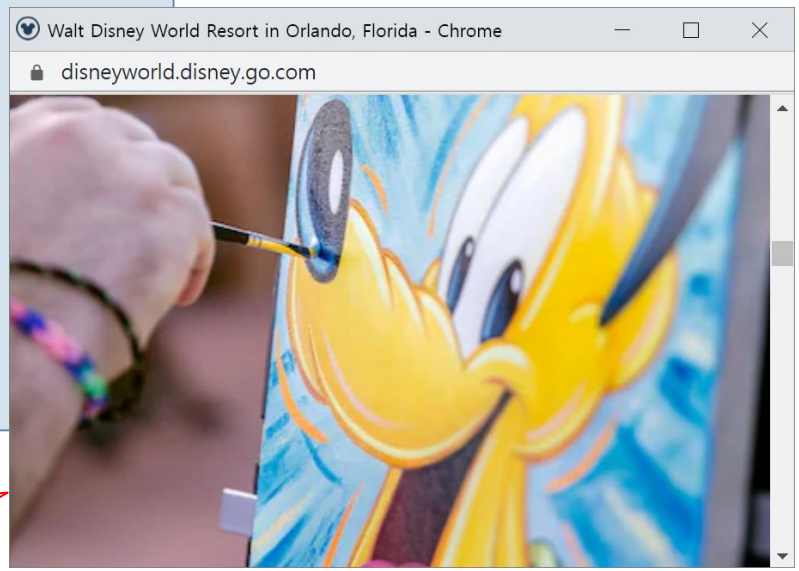
예제 10-1 window.open()으로 윈도우 열기

3개의 링크를 가진 웹 페이지를 작성하고, 각 링크를 클릭하면 myWin 이름의 새 윈도우를 열고 해당 사이트를 출력하라. myWin 윈도우는 공유된다. 새 윈도우는 스크린의 (300, 300) 위치에 400x300 크기로 출력된다.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>윈도우 열기</title>
<script>
function load(URL) {
   window.open(URL, "myWin", "left=300,top=300,width=400,height=300");
}
</script>
</head>
<body>
<h3>window.open()으로 윈도우 열기</h3>
<hr>
<a href="javascript:load('http://www.graceland.com')">
        엘비스 프레슬리 홈 페이지</a><br>
<a href="javascript:load('http://www.universalorlando.com')">
        유니버설 올랜드 홈 페이지</a><br>
<a href="javascript:load('http://www.disneyworld.com')">
        디즈니월드 홈 페이지</a><br>
</body>
</html>
```



새 윈도우를 열고
디즈니 홈 페이지 출력

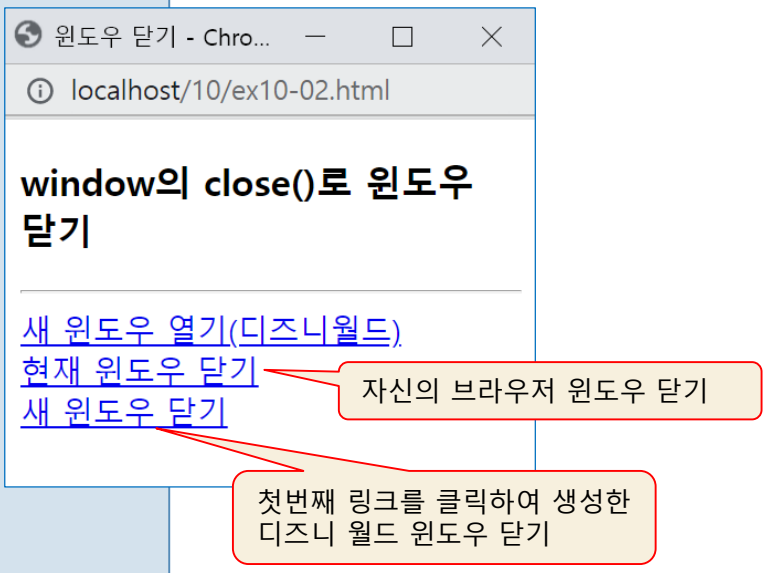


myWin 윈도우

예제 10-2 윈도우 닫기

윈도우를 스스로 닫는 경우와 자신이 생성한 윈도우를 닫는 사례를 보인다.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>윈도우 닫기</title>
<script>
let newWin=null; // 새로 연 윈도우 기억
function load(URL) {
    newWin = window.open(URL, "myWin", "left=300,top=300,width=400,height=300");
}
function closeNewWindow() {
    if(newWin==null || newWin.closed) // 윈도우가 열리지 않았거나 닫힌 경우
        return; // 윈도우가 없는 경우 그냥 리턴
    else
        newWin.close(); // 열어 놓은 윈도우 닫기
}
</script>
</head>
<body>
<h3>window의 close()로 윈도우 닫기</h3>
<hr>
<a href="javascript:load('http://www.disneyworld.com')">
        새 윈도우 열기(디즈니월드)</a><br>
<a href="javascript:window.close()">
        현재 윈도우 닫기</a><br>
<a href="javascript:closeNewWindow()">
        새 윈도우 닫기</a>
</body>
</html>
```



window 객체의 타이머 활용

- window 객체의 타이머 기능 2 가지
  - 타임아웃 코드 1회 호출
    - setTimeout()/clearTimeout() 메소드
  - 타임아웃 코드 반복 호출
    - setInterval()/clearInterval() 메소드

setTimeout()/clearTimeout()

- setTimeout() : 타임아웃 코드 1회 실행

```
let timerID = setTimeout("timeOutCode", msec)
clearTimeout(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setTimeout()은 msec 후에 timeOutCode를 1회 실행하도록 타이머를 설정하고, 타이머 ID를 리턴한다.
clearTimeout()은 작동 중인 timerID의 타이머를 해제한다.

예) 3초 후 경고창 출력

```
function myAlert(time) {
        alert(time + "초 지났습니다");
}
let timerID = setTimeout("myAlert(3)", 3000); // 3초 후 myAlert('3') 호출
```

예) 3초가 되기 전에 타이머 해제

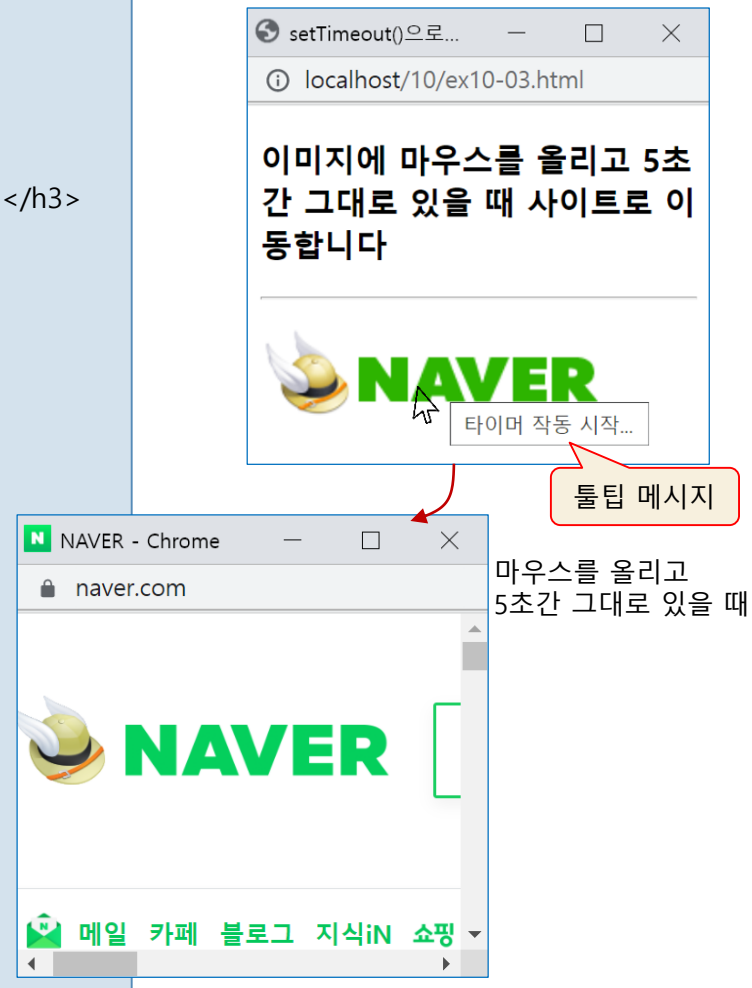
```
clearTimeout(timerID); // timerID의 타이머 해제
```

예제 10-3 setTimeout()로 웹 페이지 자동 연결

이미지 위에 마우스를 올린 상태로 5초가 지나면 네이버에 연결하며, 5초 전에 이미지를 벗어나면 타이머를 해제하는 코드를 작성하라.

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>setTimeout()으로 웹 페이지 자동 연결</title>
</head>
<body>
<h3>이미지에 마우스를 올리고 5초간 그대로 있을 때 사이트로 이동합니다</h3>
<hr>
<img id="img" src="media/naver.gif"
        onmouseover="startTimer(5000)"
        onmouseout="cancelTimer()">
<script>
let timerID=null;
function startTimer(time) {
    // 타이머 시작
    timerID = setTimeout("load('http://www.naver.com')", time);

    // 이미지에 마우스 올리면 나타내는 툴팁 메시지
    document.getElementById("img").title = "타이머 작동 시작...";
}
function cancelTimer() {
    if(timerID !=null)
        clearTimeout(timerID); // 타이머 중단
}
function load(url) {
    window.location = url; // 현재 윈도우에  url 사이트 로드
}
</script>
</body>
</html>
```



setInterval()/clearInterval()

- setInterval() : 타임아웃 코드 반복 실행

```
let timerID = setInterval("timeOutCode", msec)
clearInterval(timerID)
```

- timeOutCode : 타임아웃 자바스크립트 코드
- msec : 밀리초 단위의 정수로서, 타임아웃 지연 시간

setInterval()은 msec 주기로 timeOutCode를 무한 반복하도록 타이머를 설정하고, 타이머의 ID를 리턴한다. clearInterval()은 timerID의 타이머를 해제한다.

예) 1초 간격으로 f() 반복 호출

```
function f() {
        // 함수 코드
}
let timerID = setInterval("f()", 1000); // 1초 주기로 f()가 호출되도록 타이머 작동
```

예) 타이머 해제

```
clearInterval(timerID); // timerID의 타이머 해제
```

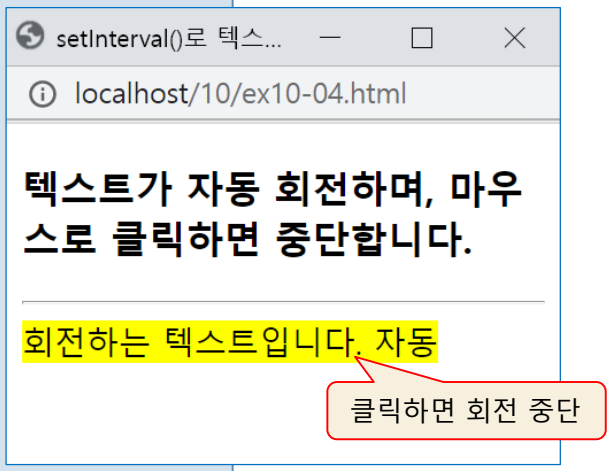
예제 10-4 setInterval()로 텍스트 회전

setInterval()을 이용하여 텍스트를 옆으로 반복 회전시키는 코드를 작성하라. 텍스트 위에 마우스를 클릭하면 회전이 중단된다.

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"><title>setInterval()로 텍스트 회전</title></head>
<body>
<h3>텍스트가 자동 회전하며, 마우스로 클릭하면 중단합니다.</h3>
<hr>
<div><span id="span" style="background-color:yellow">
                자동 회전하는 텍스트입니다.</span>
</div>
<script>
let span = document.getElementById("span");
let timerID = setInterval("doRotate()", 200); // 200밀리초 주기로 doRotate() 호출

span.onclick = function (e) { // 마우스 클릭 이벤트 리스너
    clearInterval(timerID); // 타이머 해제. 문자열 회전 중단
}

function doRotate() {
    let str = span.innerHTML;
    let firstChar = str.substr(0, 1);
    let remains = str.substr(1, str.length-1);
    str = remains + firstChar;
    span.innerHTML = str;
}
</script>
</body></html>
```



윈도우 위치 및 크기 조절

- 윈도우를 오른쪽으로 5픽셀, 아래로 10픽셀 이동

```
window.moveBy(5, 10);  혹은
moveBy(5, 10);
```

- 윈도우를 스크린의 (25, 10) 위치로 이동

```
window.moveTo(25, 10); 혹은 self.moveTo(25, 10);
```

- 윈도우 크기를 5 픽셀 좁게, 10픽셀 길게 조절

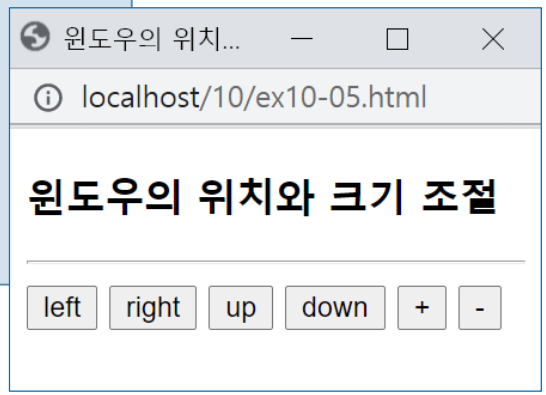
```
window.resizeBy(-5, 10); 혹은
resizeTo(self.outerWidth-5, self.outerHeight+10);
```

- 윈도우 크기를 200x300으로 조절

```
window.resizeTo(200, 300);
```

예제 10-5 윈도우의 위치와 크기 조절


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>윈도우의 위치와 크기 조절</title>
</head>
<body>
<h3>윈도우의 위치와 크기 조절</h3>
<hr>
<button onclick="window.moveBy(-10, 0)">left</button>
<button onclick="window.moveBy(10, 0)">right</button>
<button onclick="self.moveBy(0, -10)">up</button>
<button onclick="moveBy(0, 10)">down</button>
<button onclick="resizeBy(10, 10)">+</button>
<button onclick="resizeBy(-10, -10)">-</button>
</body>
</html>
```



- 이 예제는 브라우저에서 바로 실행시키면 보안의 이유로 실행되지 않지만, window.open()으로 생성한 윈도우 내에서는 잘 작동함. www.webprogramming.co.kr 사이트에서 실행시켜 보세요.

웹 페이지 스크롤

- 웹 페이지를 위로 10픽셀 스크롤(마우스 스크롤 다운)

```
window.scrollBy(0, 10); // 옆으로 0, 위로 10픽셀
```

- 웹 페이지를 왼쪽으로 10픽셀, 아래로 15픽셀 스크롤(마우스 스크롤 업)

```
window.scrollBy(10, -15);
```

- 웹 페이지의 (0, 200) 좌표 부분이 현재 윈도우의 왼쪽 상단 모서리에 출력되도록 스크롤

```
window.scrollTo(0, 200);
```

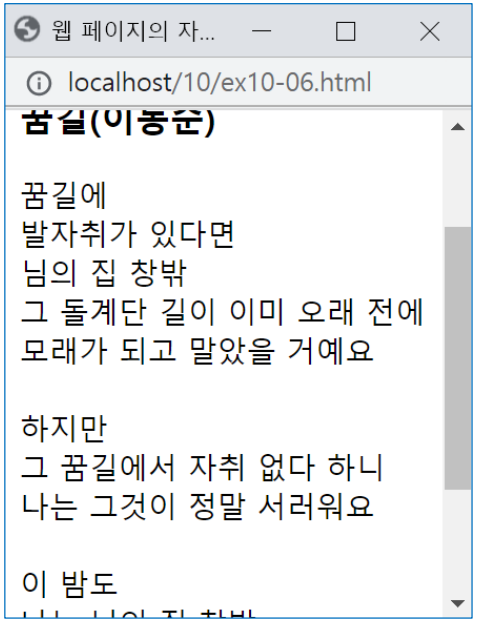
* 스크롤 다운(scroll down)은 스크롤 바를 내리는 작동이며, 이에 따라 웹 페이지는 위로 이동한다.

예제 10-6 1초마다 10픽셀씩 자동 스크롤

웹 페이지가 로드되자 마자 자동으로 1초에 10픽셀씩 웹 페이지가 올라가도록 작성하라.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>웹 페이지의 자동 스크롤</title>
<script>
function startScroll(interval) {
    setInterval("autoScroll()", interval);
}

function autoScroll() {
    window.scrollBy(0,10); // 10픽셀 위로 이동
}
</script>
</head>
<body onload="startScroll(1000)">
<h3>자동 스크롤 페이지</h3>
<hr>
<h3>꿈길(이동순)</h3>
꿈길에<br>
발자취가 있다면<br>
님의 집 창밖<br>
그 돌계단 길이 이미 오래 전에<br>
모래가 되고 말았을 거예요<br><br>
하지만<br>
그 꿈길에서 자취 없다 하니<br>
나는 그것이 정말 서러워요<br><br>
이 밤도<br>
나는 님의 집 창밖<br>
그 돌계단 위에 홀로 서서<br>
혹시라도 님의 목소리가 들려올까<br>
고개 숙이고 엿들어요<br>
</body>
</html>
```

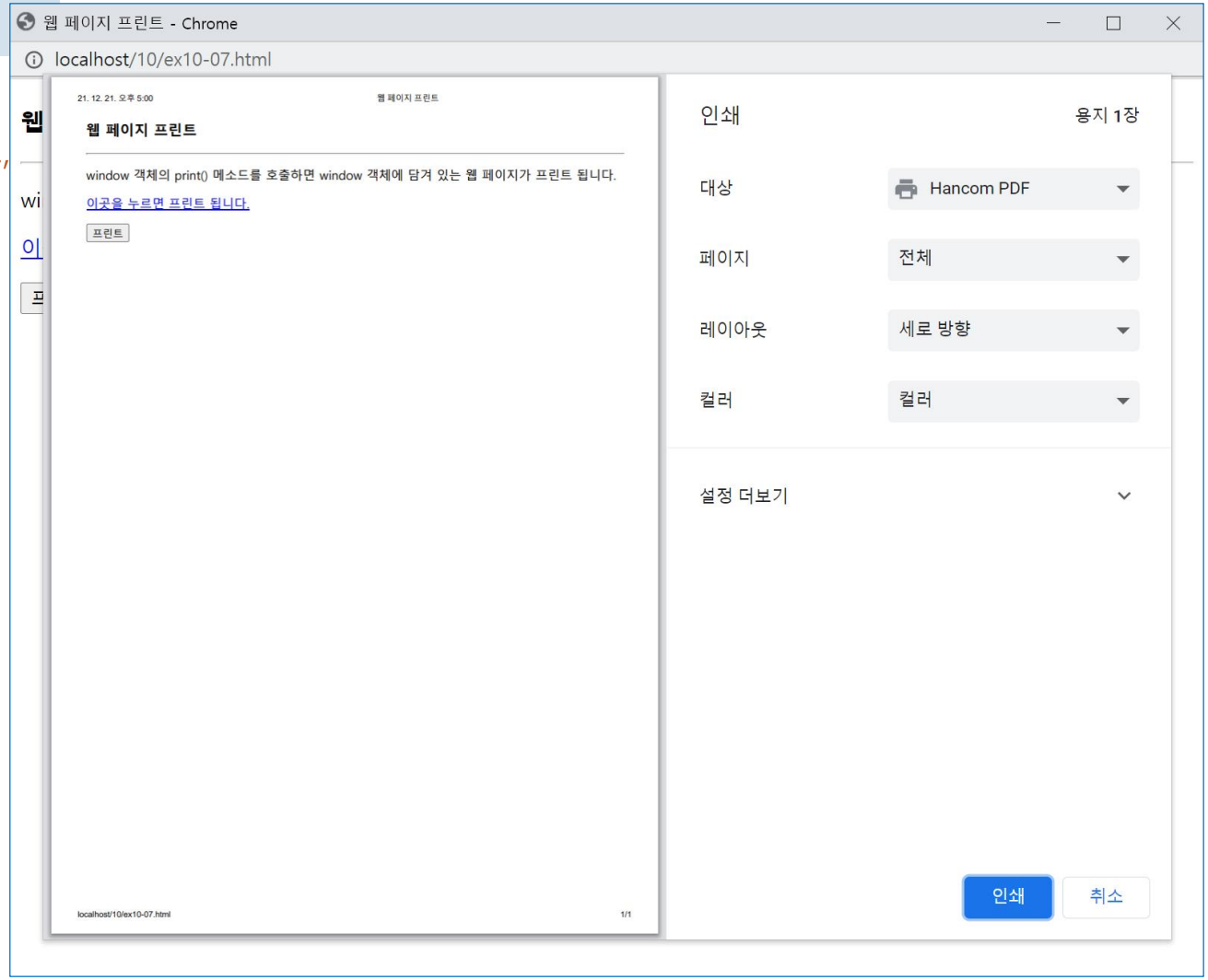


웹 페이지 프린트

웹 페이지 프린트

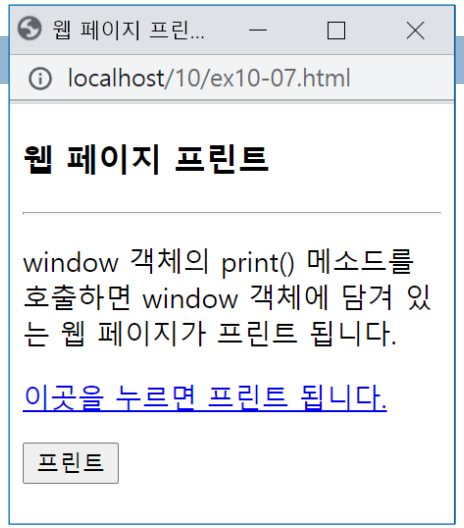
```
window.print();
```

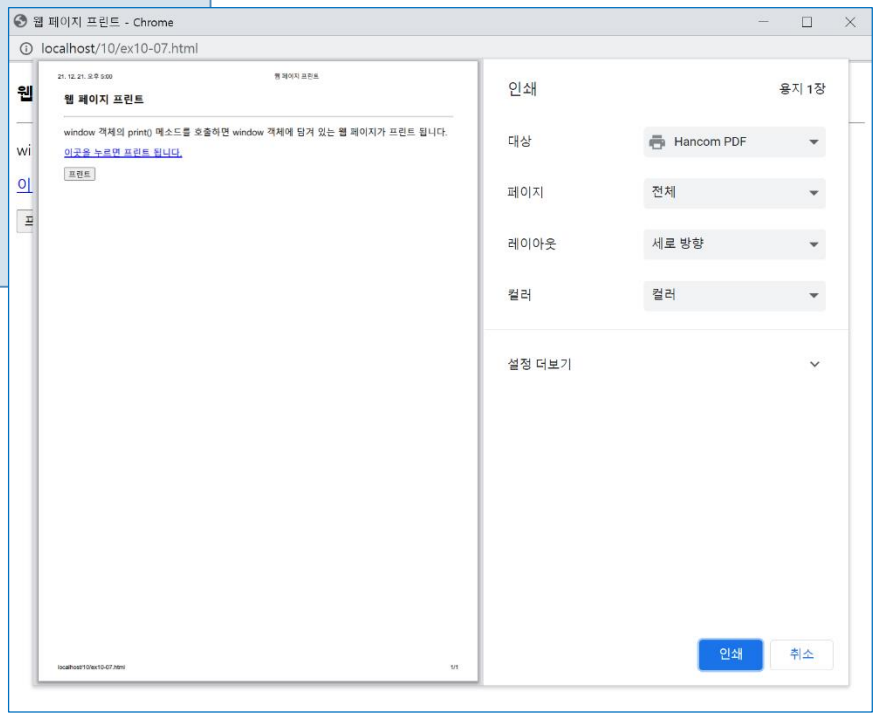
이 코드가 실행되면
인쇄 다이얼로그가 열리고,
'확인' 버튼을 누르면
인쇄가 이루어진다.



예제 10-7 웹 페이지 프린트


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>웹 페이지 프린트</title>
</head>
<body>
<h3>웹 페이지 프린트</h3>
<hr>
<p>window 객체의 print() 메소드를 호출하면
window 객체에 담겨 있는 웹 페이지가 프린트 됩니다.
<p>
<a href="javascript:window.print()">
   이곳을 누르면 프린트 됩니다.</a><p>
<input type="button" value="프린트"
        onclick="window.print()">
</body>
</html>
```



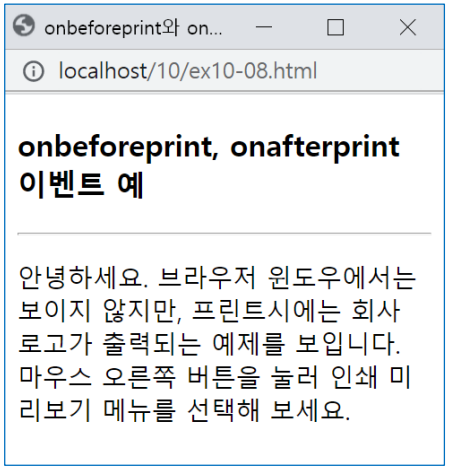


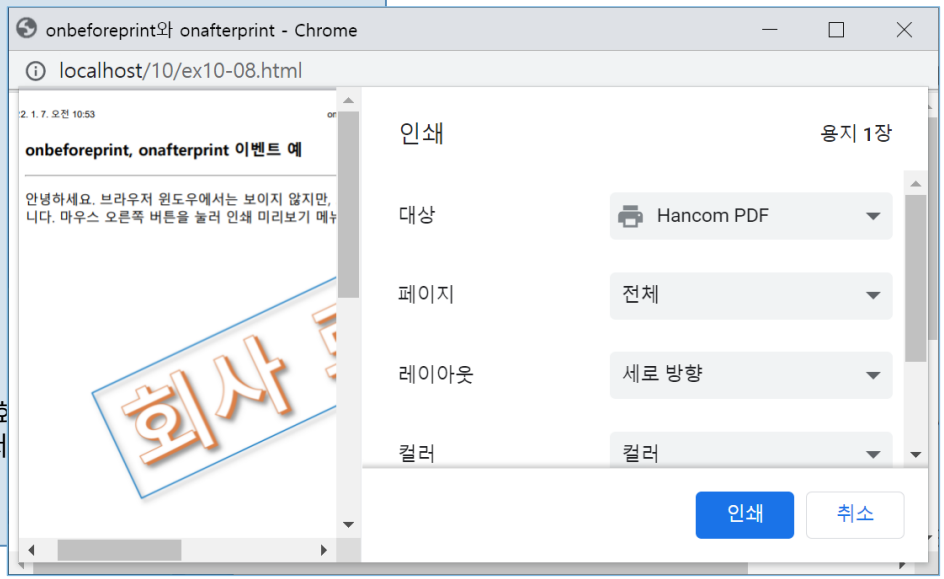
onbeforeprint와 onafterprint

- 웹 페이지의 프린트 과정
  1. window 객체에 onbeforeprint 리스너 호출
  2. 웹 페이지 프린트
     - 브라우저가 웹 페이지를 이미지로 만들어 프린터로 전송
  3. window 객체에 onafterprint 리스너 호출
- onbeforeprint와 onafterprint 활용
  - 웹 페이지에는 보이지 않는 회사 로고를 프린트 시 종이에 출력
  - onbeforeprint
    - 회사 로그 이미지를 보이도록 CSS3 스타일 설정
  - onafterprintt
    - 회사 로그 이미지를 보이지 않도록 CSS3 스타일 설정

예제 10-8 onbeforeprint와 onafterprint 이벤트 활용


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>onbeforeprint와 onafterprint</title>
<style>
#logoDiv {
    display : none;
    position : absolute; left : 0; top : 0;
    width : 100%; height : 100%;
    z-index : -1; /* 로고 이미지를 문서의 밑바닥에 배치 */
}
</style>
<script>
window.onbeforeprint=function (e) {
    let logoDiv = document.getElementById( " logoDiv " );
    logoDiv.style.display =  " block " ; // block으로 변경. 로고가 화면에 나타나게 함
}
window.onafterprint=hideLogo;
function hideLogo() {
    let logoDiv = document.getElementById( " logoDiv " );
    logoDiv.style.display =  " none " ; // 로고를 보이지 않게 함
}
</script></head>
<body>
<h3>onbeforeprint, onafterprint 이벤트 예</h3>
<hr>
<div id="logoDiv">
    <img src="media/logo.png" alt="이미지 없습니다.">
</div>
<p>안녕하세요. 브라우저 윈도우에서는 보이지 않지만, 프린트시에는 회
보입니다. 마우스 오른쪽 버튼을 눌러 인쇄 미리보기 메뉴를 선택해 보세
</body>
</html>
```





location 객체

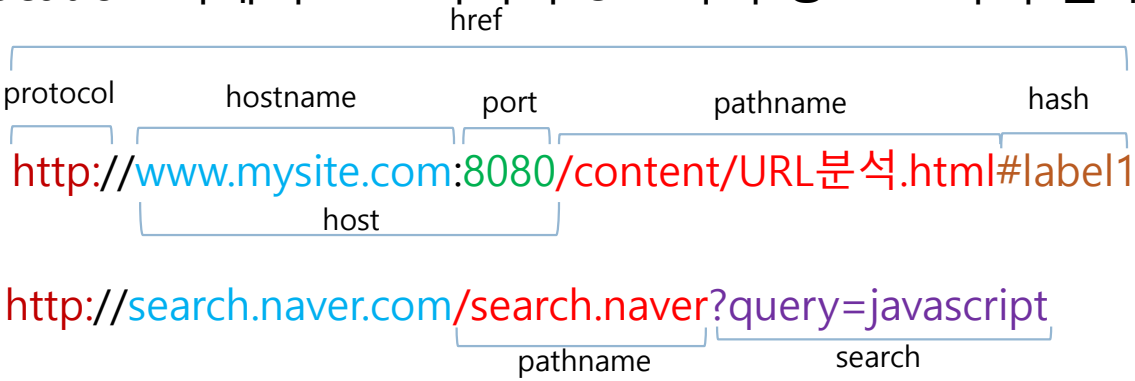
- location 객체
  - 윈도우에 로드된 웹 페이지의 URL 정보를 나타내는 객체
  - location 객체로 현재 윈도우에 웹 페이지 열기

```
window.location = "http://www.naver.com";
window.location.href = = "http://www.naver.com";
window.location.assign("http://www.naver.com");
window.location.replace("http://www.naver.com");
```

  - 새 윈도우에 웹 페이지 열기

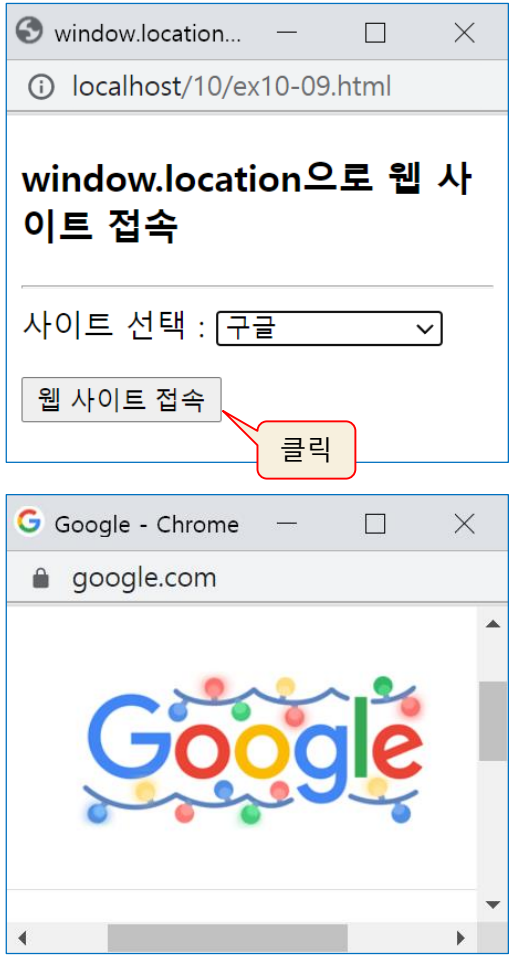
```
let win=window.open();                          // 빈 윈도우 열기
win.location="http://www.naver.com";            // 네이버 페이지 로드
```

  - location 객체의 프로퍼티와 URL의 구성 요소와의 관계



예제 10-9 location 객체로 웹 사이트 접속


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>window.location으로 웹 사이트 접속</title>
<script>
function load() {
    let select = document.getElementById("site");
    window.location=select.options[select.selectedIndex].value;
}
</script>
</head>
<body>
<h3>window.location으로 웹 사이트 접속</h3>
<hr>
사이트 선택 :
<select id="site">
    <option value="http://www.naver.com" selected>네이버
    <option value="http://www.google.com">구글
    <option value="http://www.microsoft.com">마이크로소프트
</select>
<p>
<button onclick="load()">웹 사이트 접속</button>
</body>
</html>
```



navigator 객체

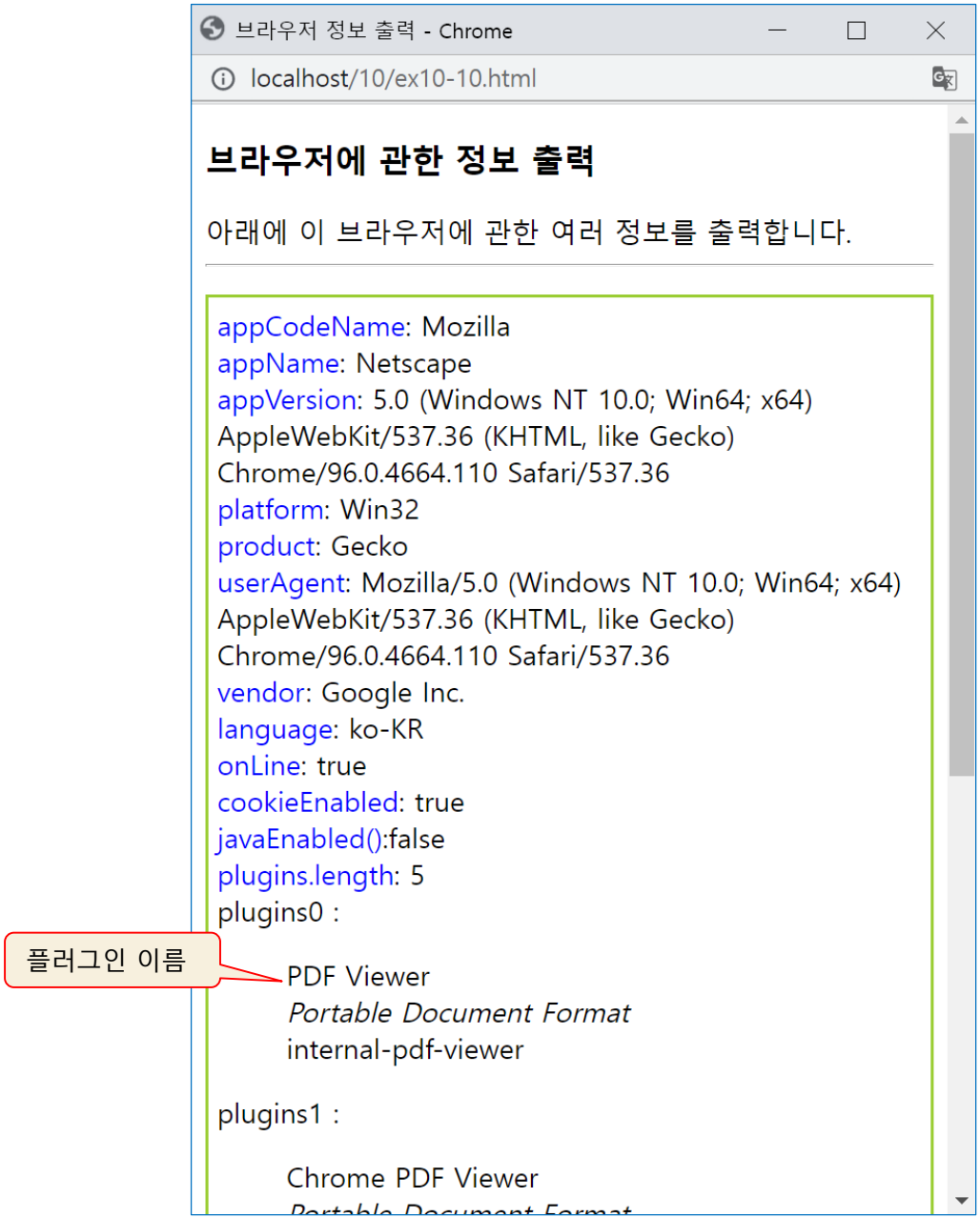
- navigator 객체
  - 현재 작동중인 브라우저에 대한 다양한 정보를 나타내는 객체

프로퍼티	설명	r/w
appCodeName	브라우저의 코드 이름을 가진 문자열	r
appName	브라우저 이름 문자열	r
appVersion	브라우저의 플랫폼과 버전에 관한 문자열	r
platform	운영체제 플랫폼의 이름	r
product	브라우저 엔진의 이름	r
userAgent	브라우저가 웹 서버로 데이터를 전송할 때, HTTP 헤더 속의 user-agent 필드에 저장하는 문자열로서 웹 서버가 클라이언트를 인식하기 위한 목적	r
vendor	브라우저 제작 회사의 이름 문자열	r
language	브라우저의 언어를 나타내는 문자열로서, 영어는 "en-US", "ko-KR"	r
onLine	브라우저가 현재 온라인 작동중이면 true, 아니면 false	r
plugins	브라우저에 설치된 플러그인(plugin 객체)에 대한 컬렉션	r
cookieEnabled	브라우저에 쿠키를 사용할 수 있는 상태이면 true, 아니면 false	r
geolocation	위치 정보를 제공하는 geolocation 객체에 대한 레퍼런스	r

예제 10–10 navigator로 브라우저 정보 출력


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"><title>브라우저 정보 출력</title>
<style>
span { color : red; }
div {
    border-color : yellowgreen;
    border-style : solid;
    padding : 5px;
}
</style>
<script>
function printNavigator() {
    let text = "<span>appCodeName</span>: " + navigator.appCodeName + "<br>";
        text += "<span>appName</span>: " + navigator.appName + "<br>";
        text += "<span>appVersion</span>: " + navigator.appVersion + "<br>";
        text += "<span>platform</span>: " + navigator.platform + "<br>";
        text += "<span>product</span>: " + navigator.product + "<br>";
        text += "<span>userAgent</span>: " + navigator.userAgent +"<br>";
        text += "<span>vendor</span>: " + navigator.vendor +"<br>";
        text += "<span>language</span>: " + navigator.language + "<br>";
        text += "<span>onLine</span>: " + navigator.onLine + "<br>";
        text += "<span>cookieEnabled</span>: " + navigator.cookieEnabled + "<br>";
        text += "<span>javaEnabled()</span>:" + navigator.javaEnabled() + "<br>";
        text += "<span>plugins.length</span>: " + navigator.plugins.length + "<br>";
        for(let j=0; j<navigator.plugins.length; j++) {
            text += "plugins" + j + " : <blockquote>";
            text += navigator.plugins[j].name + "<br>";
            text += "<i>" + navigator.plugins[j].description + "</i><br>";
            text += navigator.plugins[j].filename + "</blockquote>";
        }

    // div 태그에 출력
    let div = document.getElementById("div");
    div.innerHTML = text;
}
</script>
</head>
<body onload="printNavigator()">
<h3>브라우저에 관한 정보 출력</h3>
아래에 이 브라우저에 관한 여러 정보를 출력합니다.
<hr>
<p>
<div id="div"></div>
</body>
</html>
```


브라우저 정보 출력 - Chrome
localhost/10/ex10-10.html
브라우저에 관한 정보 출력
아래에 이 브라우저에 관한 여러 정보를 출력합니다.
appCodeName: Mozilla
appName: Netscape
appVersion: 5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/96.0.4664.110 Safari/537.36
platform: Win32
product: Gecko
userAgent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/96.0.4664.110 Safari/537.36
vendor: Google Inc.
language: ko-KR
onLine: true
cookieEnabled: true
javaEnabled():false
plugins.length: 5
plugins0 :
PDF Viewer
Portable Document Format
internal-pdf-viewer
plugins1 :
Chrome PDF Viewer
플러그인 이름

screen 객체

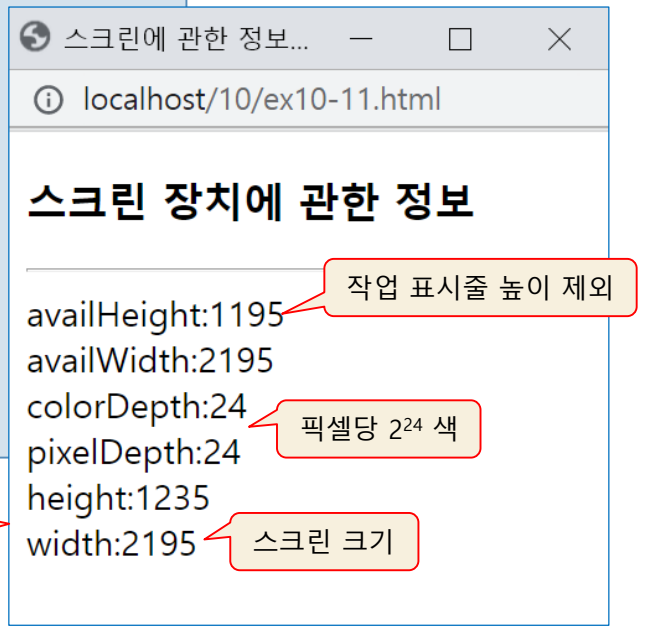
- screen
  - 브라우저가 실행되는 스크린 장치에 관한 정보를 담고 있는 객체

프로퍼티	설명	r/w
availHeight	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 높이	r
availWidth	작업 표시줄 등을 제외하고 브라우저가 출력 가능한 영역의 폭	r
pixelDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수	r
colorDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수로서 pixelDepth와 동일. 대부분의 브라우저에서 지원되므로 pixelDepth보다 colorDepth를 사용할 것을 권함	r
height	스크린의 수직 픽셀 수	r
width	스크린의 수평 픽셀 수	r

예제 10-11 스크린 장치에 관한 정보 출력


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>스크린 장치에 관한 정보 출력</title>
<script>
function printScreen() {
    let text = "availHeight:".fontcolor('blue') + screen.availHeight + "<br>";
        text += "availWidth:".fontcolor('blue') + screen.availWidth + "<br>";
        text += "colorDepth:".fontcolor('blue') + screen.colorDepth + "<br>";
        text += "pixelDepth:".fontcolor('blue')+ screen.pixelDepth + "<br>";
        text += "height:".fontcolor('blue') + screen.height + "<br>";
        text += "width:".fontcolor('blue') + screen.width + "<br>";

    document.getElementById("div").innerHTML = text;
}
</script>
</head>
<body onload="printScreen()">
<h3>스크린 장치에 관한 정보</h3>
<hr>
<div id="div"></div>
</body>
</html>
```



height와 width는 브라우저의 설정에서 확대/축소 값을 100%로 해야 정확한 값으로 출력됨

history 객체

- history 객체
  - 윈도우에서 방문한 웹 페이지 리스트(히스토리)를 나타내는 객체

프로퍼티	설명	r/w
length	히스토리 리스트에 있는 엔트리 수	r

메소드	설명
back()	히스토리에 있는 이전 웹 페이지로 이동. 브라우저의 <back> 버튼과 동일
forward()	히스토리에 있는 다음 웹 페이지로 이동. 브라우저의 <forward> 버튼과 동일
go(n)	히스토리에서 현재 웹 페이지에서 n 만큼 상대적인 웹 페이지로 이동

  - history 객체를 이용하여 웹 페이지를 이동하는 코드 사례

```
history.back();        // 이전 페이지로 이동
history.go(-1);        // 이전 페이지로 이동
history.forward();     // 다음 페이지로 이동
history.go(1);         // 다음 페이지로 이동
```

예제 10-12 history 객체 활용


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>history 객체 활용</title>
</head>
<body>
<h3>history 객체 활용</h3>
<hr>
<button onclick="history.back()">back()</button>
<button onclick="history.forward()">forward()</button>
<button onclick="history.go(-1)">go(-1)</button>
</body>
</html>
```