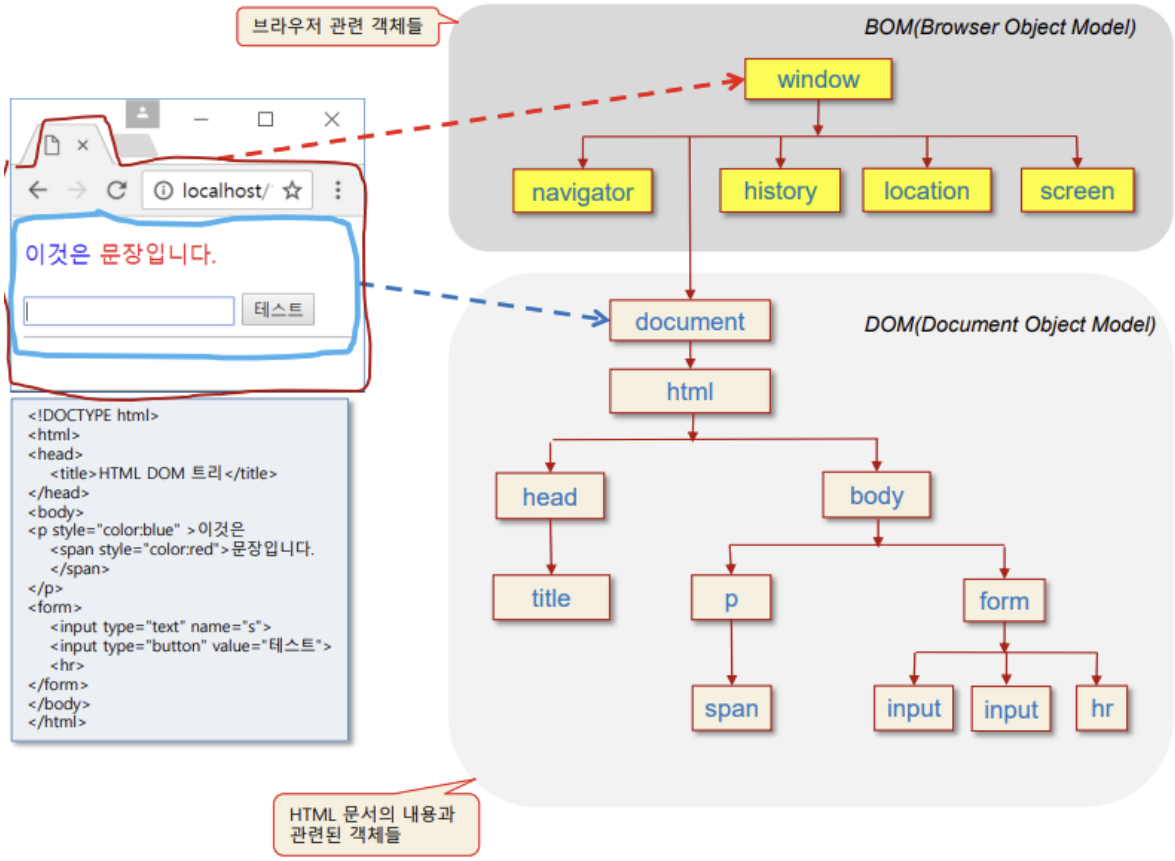
# 10장

## 윈도우와 브라우저 관련 객체

- 브라우저 관련 객체 개요

→ BOM 객체들 ⇒ 자바스크립트로 브라우저를 제어하기 위해 지원되는 객체들, 브라우저 공통 BOM 객체들과 기능

→ BOM에는 국제 표준이 없다.



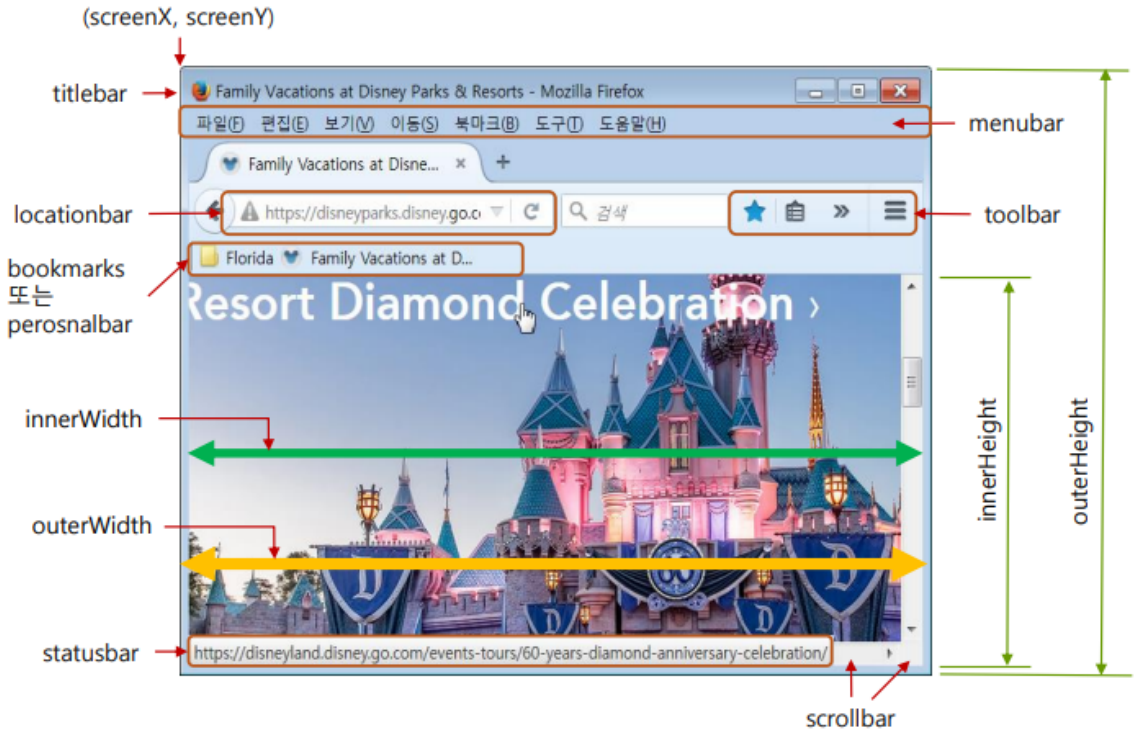

- window 객체

:열려 있는 브라우저 윈도우나 탭 윈도우의 속성을 나타내는 객체

: 브라읓 윈도우나 탭 윈도우마다 별도의 window 객체 생성

=⇒ 윈도우 객체 생성 : 1. 브라우저가 새로운 웹 페이지를 로드할 때 2. <iframe> 태그 당 하나의 window 객체 생성 3. 자바스크립트 코드로 윈도우 열기 시 window 객체 생성

=⇒ 자바스크립트 코드로 윈도우 객체에 대한 접근 : window, window.self , self




<윈도우 모양>

- 윈도우 열기

: window.open() → 윈도우를 새로 열고 웹 페이지 출력

: 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

| | |
|---|---|
| _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);
```

→ 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 않은 경우 : 버튼을 클릭하면 myWin 이름의 새 윈도우를 열고 네이버  출력

⇒ myWin 이름의 윈도우가 이미 열려 있는 경우 : 버튼을 클릭하면 이미 열려있는 myWin 이름의 윈도우에 네이버 출력

- 윈도우 객체의 타이머 활용

1. 타임아웃 코드 1회 호출 : setTimeout() / clearTimeout() 메소드
2. 타임아웃 코드 반복 호출 : setInterval() / clearInternal() 메소드

- setTimeout()

: 타임 아웃 코드 1회 실행

```
var 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.

- setInterval()

: 타임아웃 코드 반복 실행

```
var 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.

- 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);
```

- 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);
```

- 웹 페이지 프린트 : window.print()

- onbeforeprint 와 onafterprint

<웹 페이지 프린트 과정>

1. window 객체에 onbeforeprint 리스너 호출 : 회사 로그 이미지를 보이도록 CSS3 스타일 설정
2. 웹 페이지 프린트
3. window 객체에 onafterprint 리스너 호출 : 회사 로그 이미지를 보이지 않도록 CSS3 스타일 설정

- location 객체

: 윈도우에 로드된 웹 페이지의 url 정보를 나타내는 객체

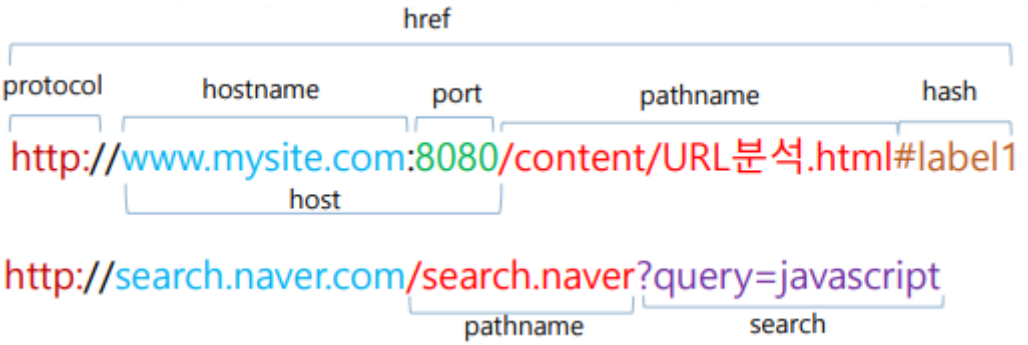
: location 객체로 현재 윈도우에 웹페이지 열기

```
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

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

: location 객체의 프로퍼티와 URL 의 구성 요소와의 관계




- navigator 객체

: 현재 작동 중인 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 |

- screen 객체

: 브라우저가 실행되는 스크린 장치에 관한 정보를 담고 있는 객

| 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 |

- 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 |

예시 )

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