BOM

location对象

提供当前窗口加载文档的信息，导航功能
既是window属性，又是document属性，即window.location和document.location指向同一个对象
保存着URL解析为离散片段后能够通过属性访问的信息

假设浏览器当前加载的URL是http://foouser:barpassword@www.wrox.com:80/WileyCDA/?q=javascript#contents，location对象的内容如下表所示。

属性	值	说明
location.hash	"#contents"	URL散列值（井号后跟零或多个字符），如果没有则为空字符串
location.host	"www.wrox.com:80"	服务器名及端口号
location.hostname	"www.wrox.com"	服务器名
location.href	"http://www.wrox.com:80/WileyCDA/?q=javascript#contents"	当前加载页面的完整URL。location的toString()方法返回这个值
location.pathname	"/WileyCDA/"	URL中的路径和（或）文件名
location.port	"80"	请求的端口。如果URL中没有端口，则返回空字符串
location.protocol	"http:"	页面使用的协议。通常是"http:"或"https:"
location.search	"?q=javascript"	URL的查询字符串。这个字符串以问号开头
location.username	"foouser"	域名前指定的用户名
location.password	"barpassword"	域名前指定的密码
location.origin	"http://www.wrox.com"	URL的源地址。只读

查询字符串

1. location.search获取 ?以后 的内容 location.search.substring(1) ; 若空则返回""
2. & 切割字符串
3. decodeURIComponent转义URL编码后的字符串

URLSearchParams get()，set()，delete()等方法

```
let qs = "?q=javascript&num=10";
let searchParams = new URLSearchParams(qs);
alert(searchParams.toString()); // " q=javascript&num=10"
searchParams.has("num"); // true
searchParams.get("num"); // 10
searchParams.set("page", "3");
alert(searchParams.toString()); // " q=javascript&num=10&page=3"
searchParams.delete("q");
alert(searchParams.toString()); // " num=10&page=3"
```

支持迭代

```
let qs = "?q=javascript&num=10";
let searchParams = new URLSearchParams(qs);
for (let param of searchParams) {
 console.log(param);
}
// ["q", "javascript"]
// ["num", "10"]
```

操作地址

location.assign(URL)
立即导航到新的URL，浏览器历史记录增加一条记录
以下代码与location.assign()效果一样

```
window.location = "http://www.wrox.com";
location.href = "http://www.wrox.com";
```

location.replace(URL) 不会新增历史记录，但是用户不能回到前一页

location.reload()重新加载当前页面
不传参，可能有缓存，传参true表示强制刷新

```
location.reload(); // 重新加载，可能是从缓存加载
location.reload(true); // 重新加载，从服务器加载
```

navigator对象

客户端表示浏览器标准

属性/方法	说　明
activeVrDisplays	返回数组，包含 ispresenting 属性为 true 的 VRDisplay 实例
appCodeName	即使在非 Mozilla 浏览器中也会返回"Mozilla"
appName	浏览器全名
appVersion	浏览器版本。通常与实际的浏览器版本不一致
battery	返回暴露 Battery Status API 的 BatteryManager 对象
buildId	浏览器的构建编号
connection	返回暴露 Network Information API 的 NetworkInformation 对象
cookieEnabled	返回布尔值，表示是否启用了 cookie
credentials	返回暴露 Credentials Management API 的 CredentialsContainer 对象
deviceMemory	返回单位为 GB 的设备内存容量
doNotTrack	返回用户的“不跟踪”（do-not-track）设置
geolocation	返回暴露 Geolocation API 的 Geolocation 对象
getVRDisplays()	返回数组，包含可用的每个 VRDisplay 实例
getUserMedia()	返回与可用媒体设备硬件关联的流
hardwareConcurrency	返回设备的处理器核心数量
javaEnabled	返回布尔值，表示浏览器是否启用了 Java
language	返回浏览器的主语言
languages	返回浏览器偏好的语言数组

属性/方法	说　明
locks	返回暴露 Web Locks API 的 LockManager 对象
mediaCapabilities	返回暴露 Media Capabilities API 的 MediaCapabilities 对象
mediaDevices	返回可用的媒体设备
maxTouchPoints	返回设备触摸屏支持的最大触点数
mimeTypes	返回浏览器中注册的 MIME 类型数组
onLine	返回布尔值，表示浏览器是否联网
oscpu	返回浏览器运行设备的操作系统和（或）CPU
permissions	返回暴露 Permissions API 的 Permissions 对象
platform	返回浏览器运行的系统平台
plugins	返回浏览器安装的插件数组。在 IE 中，这个数组包含页面中所有<embed>元素
product	返回产品名称（通常是"Gecko"）
productSub	返回产品的额外信息（通常是 Gecko 的版本）
registerProtocolHandler()	将一个网站注册为特定协议的处理程序
requestMediaKeySystemAccess()	返回一个期约，解决为 MediaKeySystemAccess 对象
sendBeacon()	异步传输一些小数据
serviceWorker	返回用来与 ServiceWorker 实例交互的 ServiceWorkerContainer
share()	返回当前平台的原生共享机制
storage	返回暴露 Storage API 的 StorageManager 对象
userAgent	返回浏览器的用户代理字符串
vendor	返回浏览器的厂商名称
vendorSub	返回浏览器厂商的更多信息
vibrate()	触发设备振动
webdriver	返回浏览器当前是否被自动化程序控制

检测插件

通过plugins数组

- name 插件名称
- description 插件介绍
- filename 插件的文件名
- length 当前插件处理的MIME类型数量

```
let hasPlugin = function (name) {
    name = name.toLowerCase();
    for (let plugin of window.navigator.plugins) {
        if (plugin.name.toLowerCase().indexOf(name) > -1) {
            return true;
        }
    }
    return false;
}
// 检测 Flash
alert(hasPlugin("Flash"));
// 检测 QuickTime
alert(hasPlugin("QuickTime"));
```

旧版本IE中的插件检测

IE10更低版本中插件是COM对象，检测COM标识符即可

```
// 在旧版本 IE 中检测插件
function hasIEPlugin(name) {
 try {
 new ActiveXObject(name);
 return true;
 } catch (ex) {
 return false;
 }
}
// 检测 Flash
alert(hasIEPlugin("ShockwaveFlash.ShockwaveFlash"));
// 检测 QuickTime
alert(hasIEPlugin("QuickTime.QuickTime"));
```

所以所有浏览器中检测插件应该是特定的，无法通用

```
// 在所有浏览器中检测 Flash
function hasFlash() {
 var result = hasPlugin("Flash");
 if (!result){
 result = hasIEPlugin("ShockwaveFlash.ShockwaveFlash");
 }
 return result;
}
// 在所有浏览器中检测 QuickTime
function hasQuickTime() {
 var result = hasPlugin("QuickTime");
```

```
 if (!result){
 result = hasIEPlugin("QuickTime.QuickTime");
 }
 return result;
}
// 检测 Flash
alert(hasFlash());
// 检测 QuickTime
alert(hasQuickTime());
```

注册处理程序

navigator.registerProtocolHandler()
把一个网站注册为处理某种特定类型信息应用程序
参数：

- 要处理的协议
- 处理该协议的URL
- 应用名称

```
navigator.registerProtocolHandler("mailto",
 "http://www.somemailclient.com?cmd=%s",
 "Some Mail Client");
```

screen对象

window的属性
保存客户端能力信息

属　性	说　明
availHeight	屏幕像素高度减去系统组件高度（只读）
availLeft	没有被系统组件占用的屏幕的最左侧像素（只读）
availTop	没有被系统组件占用的屏幕的最顶端像素（只读）
availWidth	屏幕像素宽度减去系统组件宽度（只读）
colorDepth	表示屏幕颜色的位数；多数系统是 32（只读）
height	屏幕像素高度
left	当前屏幕左边的像素距离
pixelDepth	屏幕的位深（只读）
top	当前屏幕顶端的像素距离
width	屏幕像素宽度
orientation	返回 Screen Orientation API 中屏幕的朝向

history对象

window的属性
当前窗口首次使用以来用户的导航历史记录
安全考虑，不会暴露访问过的URL，但是可以前进与后退

导航

go() 前进/后退多少步

```
// 后退一页
history.go(-1);
// 前进一页
history.go(1);
// 前进两页
history.go(2);
```

旧版本，参数也可以是字符串，导航到包含该字符串最近的位置，没有匹配项则什么都不做

简写 back(),forward()

```
// 后退一页
history.back();
// 前进一页
history.forward();
```

length属性，表示历史记录的条目

如果页面URL发生变化，则历史记录中生成一个新条目

历史状态管理

hashchange事件
页面URL散列变化时被触发，可以使用状态管理API来决定是否加载新页面

pushState()会创建新的历史记录
参数：state对象，新状态的标题，相对URL（可选）

replaceState()不会创建新的历史记录，只覆盖当前状态

总结

浏览器对象模型（BOM,Browser Object Model）是以window对象为基础，代表了浏览器窗口和页面可见区域
window 对象也被复用为 ECMAScript 的 Global 对象，因此所有全局变量和函数都是它的属性，所有原生类型的构造函数和普通函数也都从一开始就存在于这个对象之上

- 要引用其他 window 对象，可以使用几个不同的窗口指针
- 通过 location 对象可以以编程方式操纵浏览器的导航系统。通过设置这个对象上的属性，可以改变浏览器 URL 中的某一部分或全部
- 使用 replace()方法可以替换浏览器历史记录中当前显示的页面，并导航到新 URL
- navigator 对象提供关于浏览器的信息。提供的信息类型取决于浏览器，不过有些属性如userAgent 是所有浏览器都支持的

BOM 中的另外两个对象也提供了一些功能

- screen对象中保存着客户端显示器的信息。这些信息通常用于评估浏览网站的设备信息
- history 对象提供了操纵浏览器历史记录的能力，开发者可以确定历史记录中包含多少个条目，并以编程方式实现在历史记录中导航，而且也可以修改历史记录