Cookie 是存儲在您計算機上的小型文本文件中的數據。

當 Web 服務器將網頁發送到瀏覽器時，連接會關閉，服務器會忘記有關用戶的所有信息。

發明 Cookie 是為了解決“如何記住用戶信息”的問題：

當用戶訪問網頁時，他/她的姓名可以存儲在 cookie 中。

下次用戶訪問該頁面時，cookie 會“記住”他/她的名字。

當瀏覽器從服務器請求網頁時，屬於該頁面的 cookie 會添加到請求中。通過這種方式，服務器獲取必要的數據來“記住”有關用戶的信息。

Document.cookie

獲取並設置與當前文檔相關聯的 cookie。可以把它當成一個 getter and setter

讀取所有可從此位置訪問的 COOKIE

```
allCookies = document.cookie;
```

在上面的代碼中，allCookies 被賦值為一個字符串，該字符串包含所有的 Cookie，每條 cookie 以"分號和空格(;)"分隔(即, 鍵值對)。key=value

寫一個新 COOKIE

```
document.cookie = newCookie;
```

newCookie 是一个键值对形式的字符串。需要注意的是，用这个方法一次只能对一个 cookie 进行设置或更新。

以下可選的 cookie 屬性值可以跟在鍵值對後，用來具體化對 cookie 的設定/更新，使用分號以作分隔：

- ;path=path(例如 '/', '/mydir') 如果沒有定義，默認為當前文檔位置的路徑。
- ;domain=domain(例如'example.com'，'subdomain.example.com') 如果沒有定義，默認為當前文檔位置的路徑的域名部分。與早期規範相反的是，在域名前面加. 符將會被忽視，因為瀏覽器也許會拒絕設置這樣的 cookie。如果指定了一個域，那麼子域也包含在內。
- ;max-age=max-age-in-seconds(例如一年為 60*60*24*365)
- ;expires=date-in-GMTString-format 如果沒有定義，cookie 會在對話結束時過期
- 這個值的格式參見 Date.toUTCString()
- ;secure(cookie 只通過 https 協議傳輸)

cookie 的值字符串可以用 encodeURIComponent()來保證它不包含任何逗號、分號或空格(cookie 值中禁止使用這些值).

路徑限制並**不能**阻止從其他路徑訪問 cookie. 使用簡單的 DOM 即可輕易地繞過限制(比如創建一個指向限制路徑的, 隱藏的 iframe , 然後訪問其 contentDocument.cookie 屬性). 保護 cookie 不被非法訪問的唯一方法是將它放在另一個域名/子域名之下, 利用同源策略保護其不被讀取.

Web 應用程序通常使用 cookies 來標識用戶身份及他們的登錄會話. 因此通過竊聽這些 cookie, 就可以劫持已登錄用戶的會話. 竊聽的 cookie 的常見方法包括社會工程和 XSS 攻擊

HttpOnly 屬性可以阻止通過 javascript 訪問 cookie, 從而一定程度上遏制這類攻擊