當我們在 JavaScript 中透過 fetch 或 XMLHttpRequest 存取資源時，需要遵守 CORS (Cross-Origin Resource Sharing，跨來源資源共用)。瀏覽器在發送請求之前會先發送 preflight request (預檢請求)，確認伺服器端設定正確的 Access-Control-Allow-Methods、Access-Control-Allow-Headers 及 Access-Control-Allow-Origin 等 header，才會實際發送請求。使用 cookie 的情況下還需額外設定 Access-Control-Allow-Credentials header。
首先我們來認識瀏覽器的「同源政策」。 大家應該都有用過瀏覽器提供的 fetch API 或 XMLHttpRequest 等方式，讓我們透過 JavaScript 取得資源。常見的應用是向後端 API 拿取資料再呈現在前端。 需要注意的是，用 JavaScript 透過 fetch API 或 XMLHttpRequest 等方式發起 request，必須遵守同源政策 (same-origin policy)。 什麼是同源政策呢？簡單地說，用 JavaScript 存取資源時，如果是同源的情況下，存取不會受到限制； 然而，在同源政策下，非同源的 request 則會因為安全性的考量受到限制。瀏覽器會強制你遵守 CORS (Cross-Origin Resource Sharing，跨域資源存取) 的規範，否則瀏覽器會讓 request 失敗。
那什麼情況是同源，什麼情況不是呢？所謂的同源，必須滿足以下三個條件：
相同的通訊協定 (protocol)，即 http/https 相同的網域 (domain) 相同的通訊埠 (port)

簡單地說，CORS (Cross-Origin Resource Sharing) 是針對不同源的請求而定的規範，透過 JavaScript 存取非同源資源時，server 必須明確告知瀏覽器允許何種請求，只有 server 允許的請求能夠被瀏覽器實際發送，否則會失敗。

在 CORS 的規範裡面，跨來源請求有分兩種：「簡單」的請求和非「簡單」的請求。

接下來會分別解釋兩種請求的 CORS 分別如何運作。

所謂的「簡單」請求，必須符合下面兩個條件： 只能是 HTTP GET, POST or HEAD 方法 自訂的 request header 只能是 Accept、Accept-Language、Content-Language 或 Content-Type（值只能是 application/x-www-form-urlencoded、multipart/form-data 或 text/plain）。細節可以看 fetch spec。 不符合以上任一條件的請求就是非簡單請求。
舉個例子來說，下面這個請求不是一個簡單的請求：

```
const response = await fetch('https://othersite.com/data', {
  method: 'DELETE',
  headers: {
    'Content-Type': 'application/json',
```

```
    'X-CUSTOM-HEADER': '123'
  }
});
```

違反簡單請求的地方有三個，分別是：(1) 他是 http DELETE 方法；(2) 他的 Content-Type 是 application/json；(3) 他帶了不合規範的 X-CUSTOM-HEADER

非「簡單」的跨來源請求，例如：HTTP PUT/DELETE 方法，或是 Content-Type: application/json 等，瀏覽器在發送請求之前會先發送一個 「preflight request（預檢請求）」，其作用在於先問伺服器：你是否允許這樣的請求？真的允許的話，我才會把請求完整地送過去。

遇到 CORS 的問題，可以歸納出這樣的 SOP：

1. 先認清楚是否為「簡單」的跨來源請求，如果是，在後端 GET/POST/HEAD 方法本身加上 `Access-Control-Allow-Origin` header。
2. 如果非「簡單」跨來源請求，在後端 OPTIONS 加上 `Access-Control-Allow-Methods` 及 `Access-Control-Allow-Headers` header。另外，在後端方法本身加上 `Access-Control-Allow-Origin` header。
3. (Optional) 需要使用 cookie 的情況下，前端要加上 `credentials: 'include'` 或是 `withCredentials` 參數，後端要加上 `Access-Control-Allow-Credentials` header，而且 `Access-Control-Allow-Origin` header 不能用 `*`。