

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。

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。

*The **same-origin policy** is a critical security mechanism that restricts how a document or script loaded by one origin can interact with a resource from another origin.* 同源政策是網站安全的基礎。<https://domain-a.com> 只能存取自己網站裡的資源(圖片、影片、程式碼等)，不允許網站 <https://domain-b.com> 來存取。想要存取跨來源資源必須在某些特定情況下才被允許。

Same Origin Policy 雖然不錯，因為他防止了一些惡意的 script 攻擊，但總不會每一個跨網域都是惡意的；也不可能一間公司擁有所有的資源，有時還是必須串接第三方資源，例如 Facebook API、Google Map、政府釋出的公開 API 等。

怎麼實作 實作 CORS 相當容易，它其實只是 HTTP-Header 而已，這些設定基本上都是在後端，所以前端只需知道概念跟怎麼看 Response Header 即可。當前端用 fetch 或 XMLHttpRequest 要存取資源時，在 request 之前都會先發送 preflight request 確定 server 端有設定正確的相關 Http-Header，若檢查通過，才會實際發出 request。當然後端可以把權限開到最大讓任何人都可以讀取，不受同源政策的限制

我覺得CORS是必要存在的協定，在後端實現上非常有幫助，前端也就不用了解後端的細節，並且資源利用實用性還很高。

