跨來源資源共用（CORS）

B0929031 張宜茗

跨來源資源共用（CORS）是什麼呢？是一種使用額外 HTTP 標頭，使目前瀏覽網站的使用者代理在取得存取其他來源，像是網域等，伺服器特定資源權限的機制，就像是當使用者代理請求一個不同的文件來源，例如來自於不同網域、通訊協定或通訊埠的資源時，會建立一個跨來源 HTTP 請求；換句話說，就像是當我們在使用 JavaScrip 時，若想透過 fetch 或 XMLHttpRequest 存取資源時，就必須遵守跨來源資源共用（CORS），瀏覽器在發送請求前，會先發送預檢請求，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。

在預設的情況下，瀏覽器同源會遵守同源政策，並不允許跨域請求，因此在 server 未允許的情況下，會出現 「blocked by CORS policy」這樣的錯誤，所以當 client 想要取得跨域的資源，需要由 server 在 response header 中帶上 Access-Control-Allow-Origin 的欄位，而瀏覽器同源所遵守的「同源政策」又是什麼呢，「同源政策」和主題「跨來源資源共用」就是不一樣的東西了，同源政策中明確規範了哪些資源可以跨源存取，哪些會受到限制，一般來說跨來源

寫、跨來源嵌入是被允許的，而跨來源讀取是受限制的，也就是當你的來源，和你發送請求對象的來源是不同時，程式碼所發出的跨來源 HTTP 請求會受到限制。瀏覽器所發送的跨來源請求其實也不只上述提到的「預檢請求」，還有另一種叫做「簡單請求」，「簡單請求」有一些規範，程式中使用的一定要是 GET、HEAD、POST 方法，並且不能有客制的 header，僅允許特定的標頭和內容，如此才能算是簡單請求。兩者不同在於，如果是簡單請求而被 CORS 攔下來的話，實際上請求已經發送出去，只是被瀏覽器擋下來，但若是預檢請求，發送預檢請求後仍沒有通過，真正要發送的請求是不會發送出去的，也就是說「一旦預檢請求完成，真正的請求才會被送出」。

上面的說法或許有些模糊，以實際例子來看，我們第一次接觸到這個做法是在插入圖檔，當我們被要求在 HTML 頁面中插入圖片時，無可厚非，我們必須標記出圖片來源，http://domain-a.com HTML 頁面裡面一個 <img> 標籤的 src 屬性，載入來自 http://domain-b.com/image.jpg 的圖片，不僅如此，許多網路頁面所載入的資料來源，例如：CSS 樣式表、圖片影像、以及指令碼（script）都來自與所在位置分離的網域。