【名詞解釋】【JS】Tree-Shaking (搖樹優化)

Tree-Shaking (搖樹優化) 是一種在前端開發中非常重要的**程式碼優化技術**，它主要用於 **JavaScript 模組**。它的核心思想是**消除那些最終應用程式中沒有使用的程式碼**，從而減少打包後的檔案大小，提升網頁加載速度和執行效率。

什麼是 Tree-Shaking？

你可以把 Tree-Shaking 想像成一棵程式碼樹。你的專案代碼就是這棵樹的主幹和分枝，而你從各個模組中引入的函式、變數等，就是這棵樹上的葉子。

Tree-Shaking 的目的就是「搖掉」那些從未被使用的「葉子」。

在傳統的 JavaScript 模組中，即使你只引入了一個庫中的一個函式，整個庫的程式碼也可能會被打包進來。但有了 Tree-Shaking，打包工具能夠分析你的程式碼，識別出哪些部分實際被使用了（被「引用」或「匯入」），哪些部分從未被使用。最終，那些未被使用的程式碼就會在打包時被移除，就像秋天樹葉被搖落一樣。

Tree-Shaking 的運作原理

Tree-Shaking 的實現依賴於 **ES Modules (ESM)** 的靜態分析特性。

1. 靜態分析 (Static Analysis)：
   ES Modules (使用 import 和 export 語法) 允許打包工具在編譯時期（而不是執行時期）分析模組之間的依賴關係。這意味著打包工具能夠清楚地知道每個模組輸出了什麼，以及你的程式碼從其他模組導入了什麼。
2. 標記 (Marking)：
   打包工具（例如 Webpack、Rollup）會從你的應用程式的入口點開始，追蹤所有 import 和 export 語句。它會遍歷所有的程式碼，將實際被使用到的程式碼片段標記為「已使用」(used)。
3. 清除 (Sweeping)：
   在標記完成後，打包工具會將所有未被標記為「已使用」的程式碼片段從最終的打包檔案中移除。這些未被使用的程式碼被視為「死程式碼」(dead code)。

重點： Tree-Shaking 只有在你的程式碼是 **ES Modules 語法**時才能有效運作。這是因為 CommonJS (使用 `require` 和 `module.exports`) 等其他模組系統是動態載入的，打包工具難以在編譯時確定哪些程式碼會被使用。

誰在做 Tree-Shaking？

Tree-Shaking 主要由**現代化的模組打包工具 (Module Bundlers)** 來實現，其中最著名的就是：

- **Webpack (版本 2 及以上)**：Webpack 預設支援 Tree-Shaking，但在實際使用中，為了達到最佳效果，你可能還需要做一些配置，例如啟用 `optimization.usedExports` 和確保你的 Babel 配置不會將 ES Modules 轉換為 CommonJS。
- **Rollup**：Rollup 在設計之初就強調了 Tree-Shaking 的能力，它通常被認為在庫的打包方面效果更好，因為它能生成更小、更扁平的程式碼。
- **Vite**：Vite 在開發模式下利用瀏覽器的 ESM 原生支援，在生產環境則使用 Rollup 進行打包，因此也具備出色的 Tree-Shaking 能力。

Tree-Shaking 的好處

- **更小的檔案大小：** 這是最直接的好處，程式碼量減少，下載所需時間也減少。
- **更快的加載速度：** 瀏覽器需要下載和解析的 JavaScript 檔案更小，網頁加載速度自然更快。

- **更低的執行成本：** 瀏覽器需要執行的程式碼更少，降低了 CPU 和記憶體的使用。
- **更好的使用者體驗：** 網站響應更快，用戶體驗更流暢。

實際應用中的注意事項

為了確保 Tree-Shaking 能夠發揮最大效果，你需要注意以下幾點：

1. **使用 ES Modules 語法：** 確保你的專案程式碼和引用的第三方庫都使用 `import` 和 `export` 語法。
2. **配置 Babel (或 TypeScript)：** 如果你使用 Babel 或 TypeScript 轉換程式碼，請確保它們**不會將 ES Modules 轉換為 CommonJS 模組**，否則打包工具將無法進行靜態分析。在 Babel 中，這通常意味著將 `@babel/preset-env` 中的 `modules` 選項設定為 `false`。
3. **確認第三方庫是否支援 Tree-Shaking：** 有些第三方庫可能沒有設計成 Tree-Shaking 友好的形式。理想情況下，庫會在其 `package.json` 中設定 `sideEffects: false`，表明其程式碼沒有副作用，可以安全地進行 Tree-Shaking。
4. **避免副作用 (Side Effects)：** 包含副作用的程式碼可能會阻止 Tree-Shaking。例如，直接在模組的頂層執行一個函式可能會被打包工具視為副作用，即使這個函式本身沒有被顯式引用，也可能不會被移除。

總之，Tree-Shaking 是一種利用 ES Modules 靜態分析特性，移除未使用的程式碼，從而優化前端應用程式效能的關鍵技術。理解並正確配置它，對於構建高效能的網頁應用程式至關重要。

修訂版本 #1
★由 treeman 建立於 8 C 2025 15:07:59
由 treeman 更新於 8 C 2025 15:09:58