設計
Design
×
制度
Systems

「設計制度」其實是單一一項專業，但在這個講章中會把「設計」和「制度」分開來講，以來顯示他們兩者之間是如何的息息相關。

設計

產品設計的歷史比網頁設計比起來悠久相當多。產品設計擁有穩固的產品研究期、收集資料、使用者經驗彙整期這樣的前置作業。相較於網頁設計，通常大家容易去直接開始製作開發，忽略了許多前置期需要考慮的作業。甚至有在前置期在放資源在不適合的事情上面。這一切都要看今天所有著手製作的計畫來做規劃。

好比產品設計時不但有畫圖、還有建模、材料研究、材料選擇、材料測試、結構測試一樣，在做網站的同時，我也需要考慮到一樣的步驟。

以韌性為重，所以我們將特別著重於建置網站時所使用的技術和方法。

椅子	網站表單
可以坐	可以輸入
平面	`<input>`
需要長得像椅子	長得像 `<input>`
有高度	有欄位名稱
有平面	有 cursor |
不太大也不太小	有邊框
需要符合人體工學	符合 `<input>` 的基本行為
適當的高度	可以被 focus
腳可以碰到地板	可以按 enter 送出
有椅背	可以斷行

假設今天的作業是要設計家具，第一個步驟就是決定要設計什麼家具。我們用最經典的椅子來舉例。全世界的文化幾乎都有某種型態的椅子的發明，就算只是顆能夠坐的石頭。

我們大家對椅子都有一些既定的想像，就像經常使用網站的我們對於網頁元件會有一些既定的想像。

椅子要有一個平面，而文字表單欄位應該要有文字告知我這個欄位收什麼資訊。

當我們坐到椅子上面的時候我們期待他不會晃動，相同的當我們在文字表單欄位按下 Enter 鍵時應該就要可以送出表單。

制度

產品設計又和建築很像，有很多必要的測試以及規則，以確保設計出來是安全且可用的。在這裡網頁設計製作中又有許多可以跟建築借鏡的地方。

像是我們都知道蓋房子時要確認土壤下層的構成，知道使用海砂蓋的房屋不耐久。這些種種都是有規定和檢測的。網站也是一樣。

椅子	網站表單
需要承受重量	需要接收 input
承受多少重量？	承受哪種 input？
承受多久這個重量？	多少 input?
設計結構需要穩固	設計結構需要穩固
需要多少維修？	需要多少維修？
汰換率多高？	汰換率多高？
材料需要持久	材料需要持久
可以使用多久？	可以使用多久？

可以發現對於清楚定義網站所需要達成的目標非常重要。

拿椅子來舉例，戶外使用的椅子和室內使用的椅子就有相當大的差別：

可以使用怎麼樣的材料？會不會遭受到風雨侵蝕？這個材料可不可靠？我們知道是否已經被壓力測試過了嗎？如果材料出了問題該找誰、能夠怎麼維修呢？

拿網站來舉例，元件被使用者點下去之後也有相當的大的差別：

點下去後是帶使用者去了其他地方呢、還是有新的內容跳了出來？這關乎於我們要使用 `<a>` 超連結還是 `<button>` 按鈕。在使用前端元件時，使用的第三方程式可靠嗎？有多少人在維護？是否適合我們這個網站的用途？例如 Facebook 使用 React 製作一整個社群媒體網站，包含管理員機制。如果我們要做的是一頁式網站，有需要用到這樣的架構嗎？

前端

HTML

CSS

JavaScript

前端也就是所謂的客戶端，也可以想像就是我們在和一個網站伺服器溝通所使用的介面。

例如遙控器是我們跟電視機溝通的介面。而這個介面由三個技術所組成。HTML、CSS、和 JavaScript。

HTML

定義這是一張椅子

在這個舉例中，椅子就像是 `<form>` 表單一樣，是 HTML 固定的元件。就像每一個人的家中都會有椅子，每一個網站也幾乎都會有表單。

HTML 是前端架設的最基礎層。不論是瀏覽器引擎、驗證網站程式碼的工具、或是視障者所使用的螢幕閱讀軟體，都一定能看得懂 HTML。就像是我們生活中看到不同文化發明出來的家具，大多都可以輕鬆猜出哪個是椅子、哪個是桌子。

CSS

這張椅子長什麼樣子

網站的排版、顏色，元件的形狀、互動時的給使用者的回饋。例如滑鼠移過按鈕的時候按鈕會浮起來、移過連結時游標會變成手指的形狀。

JavaScript

這張椅子需要什麼額外的功能

例如這張椅子的腳上要加上滾輪、滾輪要可以被固定住。不應該使用 JavaScript 建立椅子最基本的功能。

漸進增強 →

Progressive Enhancement

在沒有附加功能的狀態下，建設的最基礎的功能仍要能夠運作。

當我們建設了電扶梯，在停電的時候人們還是可以在樓層中移動。但如果唯一的移動方式是電梯，不但在危急狀況下可能沒有辦法逃離，在短暫停電時，甚至可能受困。

網站也是一樣。如果盡量使用 HTML 元件作為基底功能的架設，在 JavaScript 沒有被讀取成功（流量、駭客攻擊），使用者仍然可以繼續使用網站。

前端設計與架構韌性

速度、穩定性

易接手、易維護

易用性

無障礙、親和力

速度、穩定性

網站包含多少內容

內容是否有被壓縮後再上傳

網站是否有不必要的功能或特效

網站不應該為了看起來豐富而增加內容，不必要的拖慢網站的速度。

任何媒材（圖片、影片）在不必要時不應該被讀取。

政府部門所提供的服務中最重要的是可以讓使用者達成某種任務，是很目標取向的服務。例如 6000 網站，不需要華麗的按鈕動畫及插圖，而應該把資源（電腦計算資源、網頁讀取的檔案大小）投注於讓使用者拿到 6000 元的這個流程。

易接手、易維護

網站使用多少特殊設計的的互動模式

網站使用多少第三方元件

網站使用的技術是否常見且容易學習

因為安全性及易維護性的考量，網站應該減少對第三方元件的依賴。乾淨且有組織的程式碼是最重要的。首要使用通用的 HTML 元件、次要使用政府提供的設計系統、最後再考慮是否使用可靠且洽當的第三方元件。減少私人開發的元件、避免之後需要接手的人需要摸黑學習未知、沒有教材的技術。

易用性

符合使用者期待的互動

清楚明瞭的資訊架構

直觀的導航是確保訪問者能夠輕鬆找到所需資訊的關鍵。清晰簡明的選單結構、邏輯的頁面層次結構和搜尋功能有助於提供良好的使用者體驗。此外，採用響應式設計原則確保網站能夠在不同設備和螢幕尺寸上無縫適應，提升可達性和可用性。

易讀且清晰的排版在可用性方面起著重要作用。選擇適當的字體風格、大小和間距可以提高內容的可讀性。保持文字與背景色彩之間足夠的對比度，以確保內容易於區分。在設計元素（如按鈕、圖示和表單）中保持一致性，有助於在網站的不同部分中提供熟悉和易用的使用者介面。

使用圖像、視頻和信息圖表等視覺元素可以增強使用者的參與和理解。然而，重要的是要優化文件大小，以實現快速加載時間，同時不影響品質。為視覺內容提供替代文本描述，以確保視覺障礙使用者的可訪問性。

網頁親和力

多用 HTML 元件

多用文字而非圖像來表示互動元件

避免不必要的華麗視覺效果

對於螢幕閱讀軟體來說，HTML 有如一張地圖。適當地使用它可以讓螢幕閱讀軟體的使用者有效且快速的找到需要的內容。一個網站當有越多功能不是使用 HTML 所定義，就代表地圖上的資料越少，那麼地圖就對於使用者來說更加沒有用處。

除了滑鼠、鍵盤以外，甚至有使用者是使用語言瀏覽網站。這時候讓使用者可以看到網站就知道怎麼樣念出某個按鈕的名稱就非常重要。

總結

Do less

英國的政府設計指導方針中第二點是「Do less」，也就是「少做點事情」，我覺得就是我們在做任何關於前端數位韌性決定時應該要努力的目標。

當我們找到了一種有效、且簡單就能達成目標的方法（例如使用 HTML 元件），就應該使重複使用、並盡量分享，而不是每次都重新發明一個新的做法。用最簡單的方式去達成一個網站所需要的功能。

簡單一句「Do less」這樣簡單的原則，就可以幫助我們達成

1. 速度、穩定性
2. 易接手、易維護
3. 易用性
4. 無障礙、親和力

因為 Web platform 已經有超過 25 年的歷史，在大部分的時候，我們要解決的問題都已經被解決過了。