漫談無障礙網頁設計-1

內容目錄

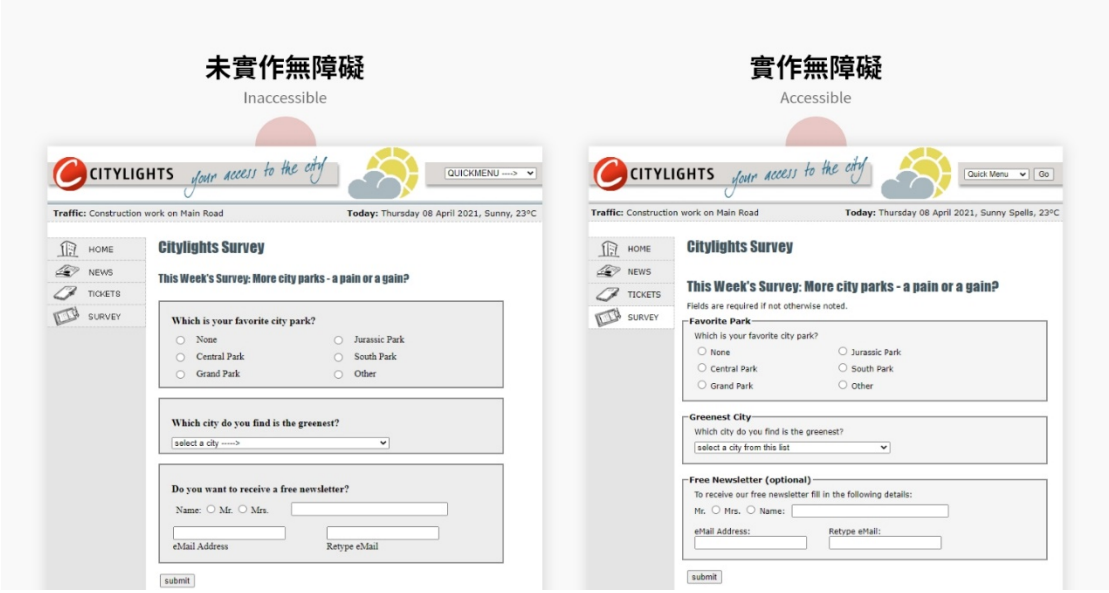
無障礙網頁設計是非常大的一個題目，雖然一開始可能看起來令人生畏，不過實際運用於網頁設計之中，除了能讓網站觸及更多使用族群之外，因為需要使用結構化語意，網頁的 SEO 也會因而一同被加強，進而使網頁更容易被一般使用者接觸到。

在這系列文章，筆者希望讓大家對於「無障礙網站」其核心目的、相關組織及具體規範等有較宏觀的認識，進而逐步介紹在設計、前端程式中具體有哪些準則應遵守。

無障礙網頁的目的

「無障礙網頁」的目的是讓任何人都能很容易地讀取網頁內容、使用網頁服務。在開發上，設計者不能僅僅以典型用戶的操作模式（使用螢幕、滑鼠、鍵盤、觸控螢幕……等）來規劃交互方式，而必須跳脫自己使用電腦和網頁的角度，考量身心障礙者的操作模式，例如色盲、使用輔助科技（如螢幕閱讀器）來瀏覽網站……等情境。

可能很多人提到「無障礙網頁」，就想像網頁上會出現許多額外的輔助資訊、或是設計簡陋不美觀，其實只要在設計階段靈活運用，滿足無障礙需求的同時也能兼顧視覺效果，而且一般使用者在瀏覽無障礙網頁時也同樣能受惠。



實作無障礙設計前後 demo 網頁比較（畫面截圖自 WAI）

從上面這個比較可以看出，有無實作無障礙設計並不會造成畫面設計上巨大的差異，很多項目反而更清楚了！會覺得這個設計看起來很古早，只是因為他真的是個將近十年前的範例網頁 XD [註 1]。

其實 UX 原則可說全都與增加網站可用性相關，瞭解 UX 重點、將這些考量融合於開發過程中，就能減少後期單純為了使網頁符合無障礙規範而進行耗費成本的修改。

往更廣的面向來說，其實還有許多在開發時就會預設好的情境典型，都可能令部分使用者——不只是身心障礙者——無法順利讀取網頁內容。以身在台灣的筆者自己來說，大多數專案開發時通常假設使用者都擁有高網速、瀏覽器支援 javascript 、沒有國家級防火牆等等，如果把這些都當成理所當然，就會排除掉部分低網速、使用手持裝置或是較舊版本瀏覽器的使用者，這也是廣義的無障礙網頁所希望避免的情況。

不過，下面我們主要還是針對狹義的無障礙網頁來作說明！

障礙的分群

Google Developers 和 MDN Web Docs 均將使用網頁的障礙分為四類 [註 2]，以利區分不同身心障礙者會遇到的使用情境，及他們可能會使用的輔助科技：

1. 視覺障礙：

   包含全盲、弱視、色盲、視力衰退等，這類使用者可能使用螢幕閱讀器、盲文顯示器、螢幕放大鏡、文字放大功能、高對比度模式、文字語音轉換等方式（甚至組合這些方式）來使用網頁。
2. 行動障礙：

   包含因為損傷、疼痛而不願使用滑鼠（例如網球肘、扳機指），以及身體癱瘓或身體某些部位活動範圍受限的使用者，這些障礙甚至可能是情境式的––像是滑鼠壞掉、或是在晃動的交通工具上使用裝置等。
3. 聽覺障礙：

   指毫無聽力或是聽力低弱的群體。與視覺一樣，聽力也會隨著年齡增長衰退。聽覺障礙者通常會使用助聽器等輔具，在網頁使用這方面，則有各種技術幫助將有聲內容轉化為替代文字，例如自動生成影片字幕等等。
4. 認知障礙：

   比較廣泛，認知疾病的類型有許多種，包含自閉症、思覺失調、ADHD 等等。這些疾病會影響使用者理解、記憶網站的操作方式。這些群體會使用的輔助工具與其他領域重疊，例如使用縮放功能來簡化閱讀或集中注意力。在開發上，比較難快速解決認知障礙造成的網頁使用問題，不過設計網站時，盡可能簡化並符合邏輯性、一致性會很有幫助。

現行規範很大一部分其實都是針對「視覺」這一塊的障礙在做檢測，主要也是由於網頁本身視覺化的內容通常較吃重，而流程、一致性這方面的要求又較難訂立統一的檢測標準之故。

障礙的分群

WCAG 與 WAI、WAI-ARIA

在研究無障礙網頁設計時一定會提到的幾個名詞縮寫，它們其實分別是組織與規範標準：

WAI（Web Accessibility Initiative）

W3C 為建立無障礙標準與準則而生的組織。WAI 的網站除了提供網頁設計、開發的建議，也提供許多背景知識，讓設計者、開發者能瞭解身心障礙人士如何使用網站，模擬出更完整的使用情境。

WCAG

WCAG 則為無障礙網頁訂出龐大、詳盡、可量化的準則（不過並未包含具體技術）。在 WCAG 2.0 中，定義了不同的評級： A 級 、 AA 級 、 AAA 級 ，許多國家立法制定的無障礙標準都奠基於 WCAG （如美國的 Section 508 of the Rehabilitation Act、台灣的無障礙網頁開發規範…等）。

WAI-ARIA（WAI – Accessible Rich Internet Applications）

WAI-ARIA 則是面向開發者的，這是一套由 W3C 編撰的規則，它定義了一系列額外的 HTML 屬性，能為 HTML 元件提供額外的語意。

自從 HTML5 導入了具有語意的標籤（ tag ），前端工程師在切版時不再只使用簡單的 `<div>` 標籤搭配 id 來呈現網頁上的各種元件，而會盡量使用有意義的標籤如 `<nav>` 、 `<footer>` 等等。然而，單純使用這些標籤仍有其限制，例如在頁面中有多個導覽列 `<nav>` 時，其各自究竟對應到哪個區塊，對螢幕閱讀器而言就難以分辨。

WAI-ARIA 定義的就是實作上應如何使用這套 HTML 屬性，如 `role` 、 `aria-label` 來標示各元件的用途、屬性、狀態，後續系列文章會更詳細地介紹。

無障礙網頁認證

在台灣，《身心障礙者權益保障法》規範政府機關、學校網站都應符合 A 級以上的檢測， 2017 年以後設計新版網頁或改版更需要達到 AA 級；而 2021 年 7 月開始即將實施的新一版（ 2.1 版）的「網站無障礙規範」，新版本的檢測範圍包含更多服務功能。

台灣規範的無障礙網站同樣有三種評級： A 級 、 AA 級 、 AAA 級 ，符合評定項目就能得到相應的等級，由 NCC 負責相關的申請與審查。

下面是 2.0 和 2.1 版的評級表格，可以看到 2.1 版新增了一個新的指引「輸入方式」，並在各細項做了調整 [註 3]：

4 原則	13 指引	A 等級	AA 等級	AAA 等級
1.可感知	1.替代文字	1.1		
	2.時序媒體	2.1 2.2 2.3	2.4 2.5	2.6 2.7 2.8 2.9
	3.可調適	3.1 3.2 3.3	3.4 3.5	3.6
	4.可辨識	4.1 4.2	4.3 4.4 4.5 4.10 4.11 4.12 4.13	4.6 4.7 4.8 4.9
2.可操作	1.鍵盤可操作	1.1 1.2 1.4		1.3
	2.充足時間	2.1 2.2		2.3 2.4 2.5 2.6
	3.防痙攣	3.1		3.2 3.3

4 原則	13 指引	A 等級	AA 等級	AAA 等級
	4.可導覽	4.1 4.2 4.3 4.4	4.5 4.6 4.7	4.8 4.9 4.10
	5.輸入方式	5.1 5.2 5.3 5.4		5.5 5.6
3.可理解	1.可讀性	1.1	1.2	1.3 1.4 1.5 1.6
	2.可預期性	2.1 2.2	2.3 2.4	2.5
	3.輸入協助	3.1 3.2	3.3 3.4	3.5 3.6
4.穩健性	1.相容性	1.1 1.2	1.3	

2.1 版無障礙規範索引（筆者自行整理），粗體為與 2.0 版不同之處

其中，4 原則和 13 指引大致內容如下：

4 原則[註 4]：

1. 可感知：資訊及使用者介面元件應以使用者能察覺之方式呈現（使用者一定要能察覺呈現出來的資訊，也就是資訊不能對使用者所有的感官均無形）
2. 可操作：使用者介面元件及導覽功能應具可操作性（使用者一定要能夠操作介面，介面不能要求使用者無法執行的互動方式）
3. 可理解：資訊及使用者介面之操作應具可理解性（使用者一定要能夠明白資訊及使用者介面的操作，亦即內容及操作皆不能超出使用者的理解）
4. 穩健性：網頁內容應可供身心障礙者以輔助工具讀取，並具有相容性（隨著科技進步，使用者一定要能取用內容，也就是說當科技及使用者代理演進後，內容仍應保有可及性）

13 指引：

1.1 替代文字

有意義的非文字內容需有替代文字來描述、裝飾性內容則要使輔助科技能夠忽略之

1.2 時序媒體

音訊、視訊、互動性影片等需具有替代文字、字幕、描述，甚至手語翻譯

1.3 可調適

程式的結構、順序需要與內容有關聯性，能夠讓使用者知道元件的用途

1.4 可辨識

資訊內容須與背景有足夠的對比，讓使用者易於辨識，主要關乎設計及 RWD

2.1 鍵盤可操作

針對「只有鍵盤」的使用情境提供額外的支援度，避免功能只依據特定的輸入方式才能使用

2.2 充足時間

有障礙的使用者通常會需要花更多時間來進行互動，須確保網頁中有計時、自動播放的內容能夠讓使用者能有足夠反應時間，或是可以控制內容暫停

2.3 預防痙攣和身體不適反應

減少畫面閃爍、使用者能自行終止互動性的動畫

2.4 可導覽

使用螢幕閱讀器等輔助工具，瀏覽網頁時是線性的，這項的功能在於如何讓使用者知道自己身處網頁中的何處、如何到達其他地方，與標題、標籤、鏈結、 HTML 標籤、焦點的設定有關

2.5 輸入方式

和鍵盤可操作類似，這邊主要是針對滑鼠、觸控螢幕、雷射指標等指標輸入設備的支援度

3.1 可讀性

確保使用者能理解網頁內容，針對特殊或艱深辭彙、縮寫、破音字等需有說明

3.2 可預期性

與可導覽類似，使用輔助工具時對頁面內容難以有全觀的瞭解，因此元件需具備一致性以讓使用者更容易瞭解掌握頁面內容

3.3 輸入協助

與 input 相關的提示說明，以及偵測錯誤、提出建議及錯誤預防機制

4.1 相容性

HTML 結構須完整，元件名稱、屬性、值、狀態等均需要實作於結構中

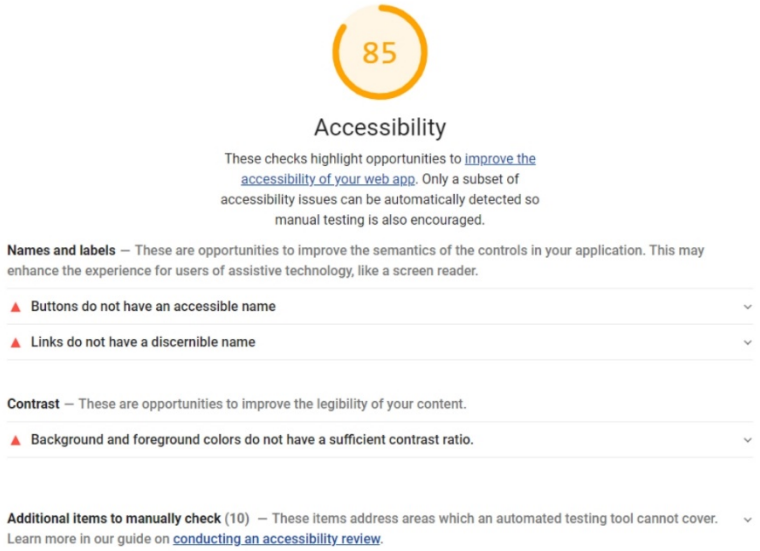
在專案中導入無障礙設計

若專案需求包含通過無障礙網頁認證，最好一開始就和團隊成員說明專案需通過哪個等級的認證，具體必須符合哪些規範，以避免專案最後階段因為無法通過認證而必須大幅改動的情形。例如， SPA（Single Page Application）[註 5] 其中一個硬傷就是 SEO 不佳，原因在於頁面一開始只有一個空容器或至少不是完整的網頁內容，必須額外實作 SSR（Server Side Render）來解決這個問題。而同樣的原因也會使無障礙標準難以達成，如果一開始不清楚就把頭洗下去，到最後階段才打掉重練絕對會欲哭無淚。以分工來說，團隊全員最好都知道大致有哪些規範，才有辦法實作於專案中並進行檢驗。

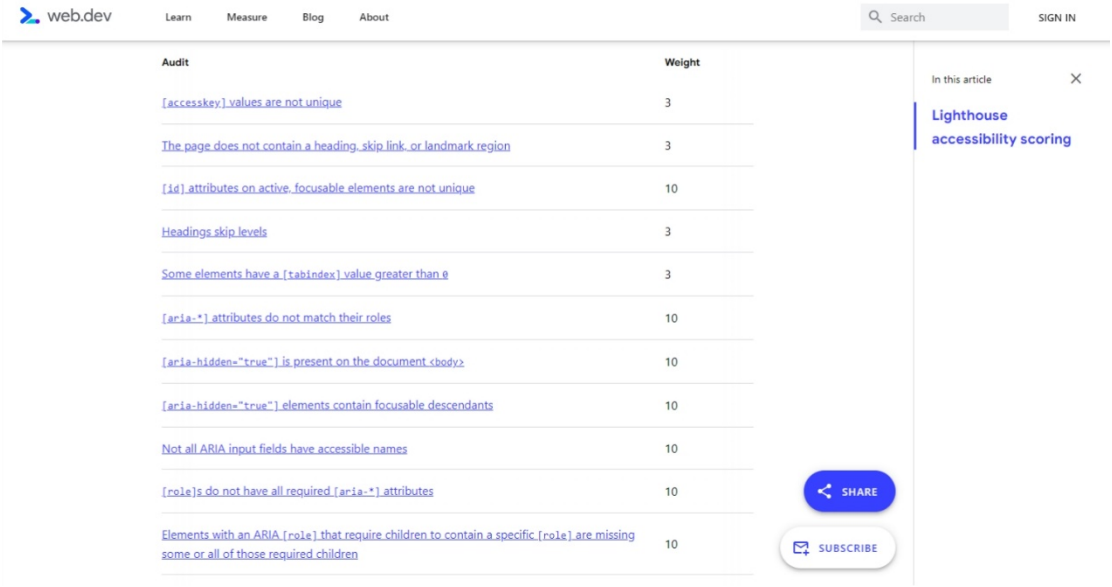
首先，設計師會需要額外考慮到顏色對比度、字級等具體規定（ UI ），並加強檢驗頁面流程及元件的一致性（ UX ）。設計師們可以著重閱讀以下幾個指引：

- 1.4 可辨識
- 2.3 預防痙攣和身體不適反應（針對動畫、動態設計）
- 3.2 可預期性（與 UX 相關）

再來，對前端工程師而言，由於大部分技術面的規範都屬於頁面結構、HTML 標籤及屬性的範疇，所以擔子會更吃重一些。不過！前端工程師如果熟悉 Google 提供的 web.dev 工具（或 Lighthouse ），那應該已經對 A 級的大部分規範都不陌生了，其中 Accessibility 這塊就是無障礙相關的檢測。



Lighthouse 檢測結果



Lighthouse 的 Accessibility 相關檢測項目

於是前端工程師應該著重閱讀那些規範呢？可以的話就全部都看一下吧……（炸）因為負責實作的就是前端工程師嘛……。總之這邊還是列一些偏技術性的內容供參考：

- 1.3 可調適
- 2.1 鍵盤可操作
- 2.4 可導覽
- 2.5 輸入方式

- 3.3 輸入協助
- 4.1 相容性

至於為什麼後端工程師也應該要瞭解呢？例如網頁中若有編輯器產生的區塊，如何確保後台使用者輸入的內容也符合規範，這部分的防呆機制和過濾就會落在後端職責了。

最後，無障礙規範其實有非常多與網站內容有關，例如媒體均需有文字替代等。因此，針對額外需要產出的內容與客戶溝通，以及因無障礙開發而產生的時間成本、風險都會是 PM 額外需要考量的。

PM、企劃、文案可以多瞭解與內容產出相關的規範：

- 1.1 替代文字
- 1.2 時序媒體
- 2.2 充足時間（列入這點是因為計時長度可能需要協調）
- 3.1 可讀性

別以為無障礙網頁認證是額外做一個申請動作就能拿到的標章！事實上，根據網站內容複雜度，開發時間可能會增加不只一倍。開發時間的增加反映在兩個地方：

1. 無障礙開發本身
2. 檢測後進行反覆修改

如果網站內容龐大，且包含影音資料，那麼開發本身的複雜度會倍增，檢測點也會變多。若工程師熟悉無障礙規範，雖然開發本身會較耗時，但檢測可能會比較快通過、需要較少來回修改的時間；反之，更常見的則是先進行一般開發，送審後再根據改善建議進行調整––對 PM 而言，最好一開始就預期會有不少時間卡在審核與修改，尤其在需要 A 以上評級的情形，光是要熬到過審就可能曠日廢時。

[註 1]　可以來玩玩看這個 WAI 提供的 DEMO 網站：Before and After Demonstration

[註 2]　可參閱 Google Developers 和 MDN Web Docs 針對障礙者的說明頁面。

[註 3]　可參閱 網站無障礙規範。

[註 4]　整理自 網站無障礙規範。

[註 5]　SPA 是指透過 JavaScript 來變更畫面內容，讓頁面不需要一直全部重新的技術， Gmail 就是經典的例子。

漫談無障礙網頁設計-2

內容目錄

無障礙專案的開始通常有兩種情形，一種是在需求階段就提出網站要符合無障礙標準，另一種則是既有網站想要更新符合無障礙標準。前者的成本相對較低，而在後者的情況下，如果有流程或邏輯上的問題需要調整，那會要改動的範圍就會變得廣泛許多，可以考慮區分階段進行。

不過，即使專案並不需求取得無障礙標章，瞭解這些注意要點也能使你的網頁設計能為更多人所使用。實作無障礙網站，從設計開始就有許多注意要點，這篇文章希望能藉由提供一些範例讓這些要點更具體些，期待更多人能嘗試將無障礙設計應用在自己的專案中。

接下來會提到不少實際的無障礙規範內容，會引述剛出爐熱騰騰的 網站無障礙規範（110.07）十三指引（左邊目錄竟然都沒有自己的獨立連結我快瘋掉，希望之後會有改進）。

在設計上實作無障礙網頁的大方向

上一篇提到無障礙的四大原則及十三項指引，接下來就由抽象至具體，以實例來說明設計上有那些需要注意的地方以符合這些大原則。

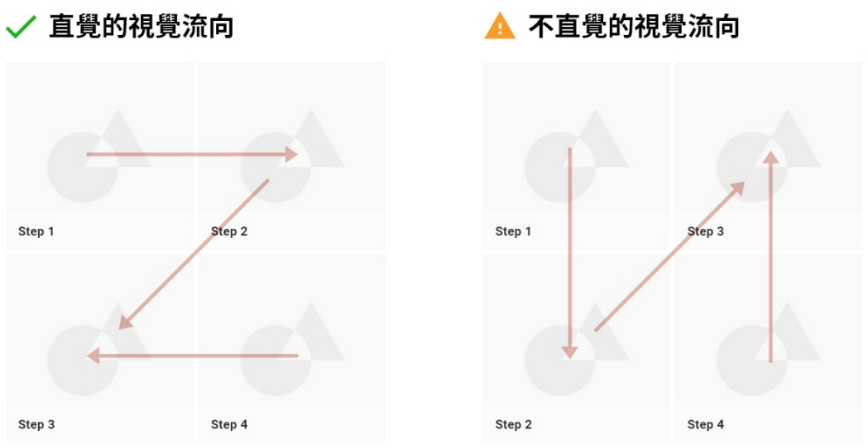
TL;DR

- 遵循易於理解的流程邏輯、並保持頁面結構具有一致性
- 相關的內容應組成群組，並且具有層級性，重要性高的內容靠前
- 同樣功能的元件，結構和外觀應有一致性

那麼就開始吧！

流程邏輯與頁面結構的一致性

多數使用者不會特別注意自己瀏覽、操作網站時遵循某些既有的順序與邏輯，像是閱讀時會由左至右、由上至下閱讀（ Z 字型的瀏覽方式）。



瀏覽順序比較

遭遇跳脫既有邏輯的設計時，一般使用者可能會只會感覺到使用網站時突然卡了一下，需要多花幾秒理解這個地方是怎麼回事，但是對於有認知障礙困擾的使用者，這種「小問題」需要花費好幾倍的心力來重新理解，容易產生挫折感並因此中斷瀏覽和操作。

同樣的，頁面結構具有一致性，能幫助使用者預測內容的位置。例如導航欄都位於頁面標題之下，那麼在同個網站中，無論到哪個頁面，使用者都能預期在頁面標題下找到導航欄；反之，相同用途的區塊在各頁面中處於不同位置，會造成認知上的困難。

相關指引及準則：

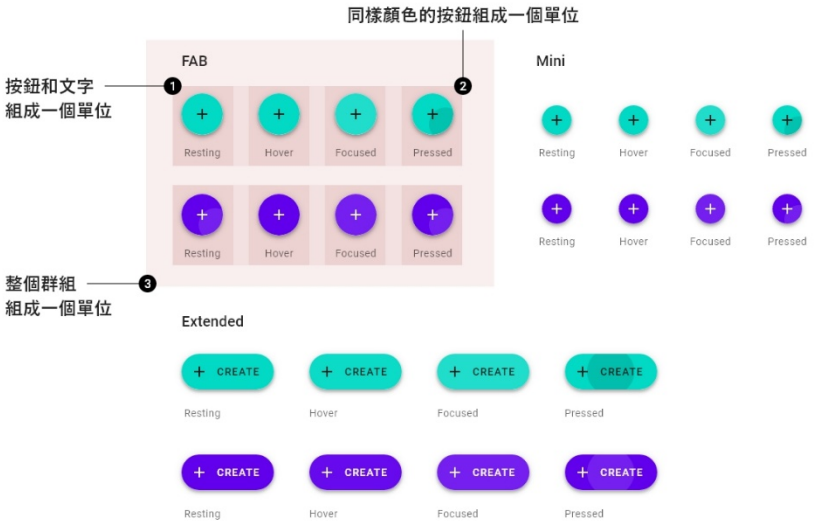
指引 3.2：可預期性
讓網頁以可預期的方式來呈現及運作

成功準則 3.2.3：一致的導覽 (檢測等級 AA)

除非使用者做出變更，否則在一組網頁中，反覆出現的導覽機制每次都要有相同的相對順序。

內容的層級與順序

在介面設計中，完形心理學[1] 的接近性法則有很重要的影響力——人們傾向將互相靠近的物件視為一體，而非僅是相似的物件——這能幫助引導使用者理解所要呈現的內容。



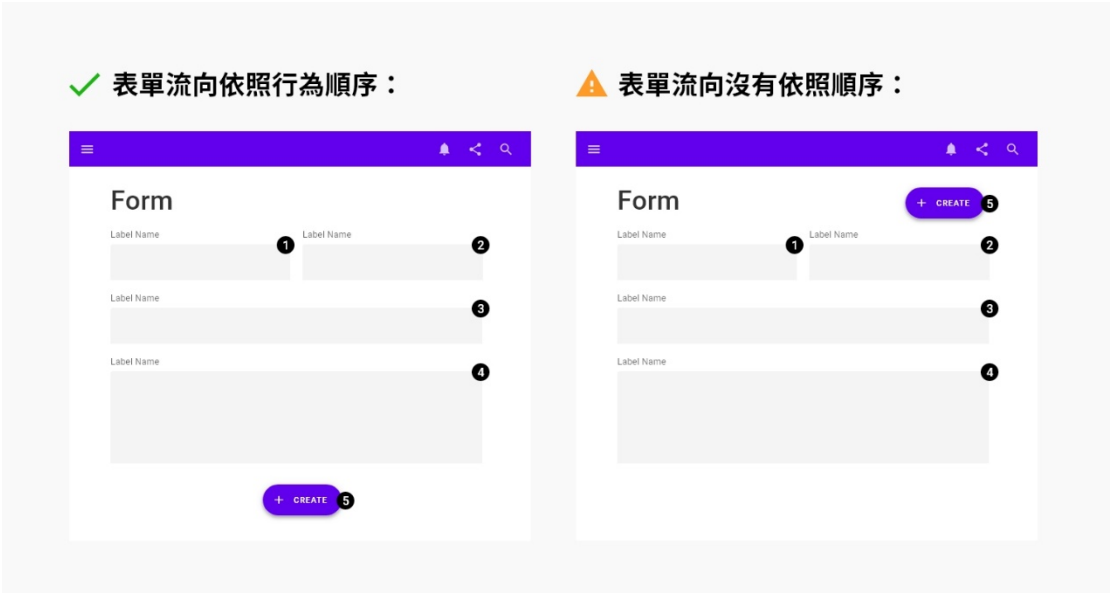
接近性法則讓我們將這個介面裡互相靠近的物件劃成群組

瀏覽畫面時，會預期同類型或相關的內容會組成一個視覺群組、並與其他內容有一定的間隔。但使用螢幕閱讀器的時候又如何呢？

以視覺瀏覽網頁時，很容易得到網頁整體的全局觀，使用者只要大略掃過整個畫面，就能大致知道網頁整體結構，要找某個特定按鈕只需上下捲動畫面就行了；然而操作螢幕閱讀器時，瀏覽、記憶網站內容的方式是線性的，因此如何幫助使用者理解自己在頁面中的位置至關重要，需要靠有規則的層級與順序來達成（還有導覽功能，不過這個之後再提）[2]。

規劃 wireframe 的時候，應考量內容的重要性來安排順序，重要性由上至下，並盡量將具相關性的元件依序放在靠近的位置，避免使用者需要來回切換至不同的地方閱讀內容。

如果內容沒有依順序排列，例如下圖右邊這個不良範例——表單的送出按鈕位於表單最上方。此時，使用螢幕閱讀器就要額外記憶「填完表格要回到前面某某位置」才能將表單送出。

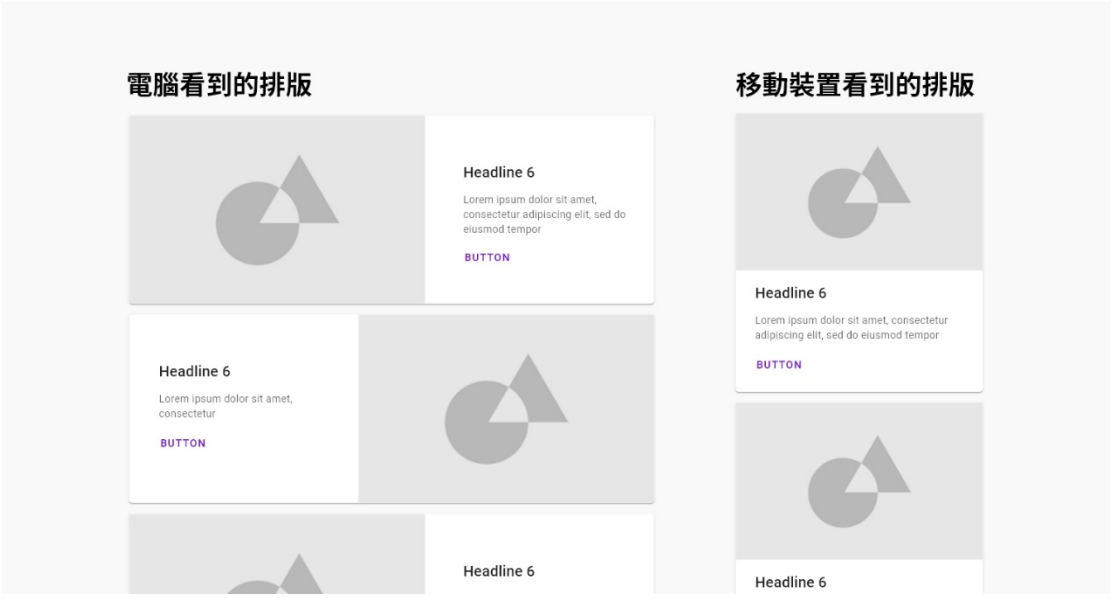


表單按鈕位置比較

這些屬於大方向的設計考量，到建置後期才改善會需要很高的成本，可以先根據 wireframe 預想操作流程，避免上述問題。

延伸討論：RWD 的情形

內容的層級與順序在不同尺寸的裝置上也應該具有一致性。常見的情形是在桌面版設計採取圖文排列時，為使畫面重量平均，會將左圖右文、左文右圖交錯排列，這樣的設計到了手機上，順序應統一調整為圖上字下或圖下字上其一，否則閱讀的流向反反覆覆會造成使用者的混亂。（注意這種設計在程式上應使用 CSS 來達成， HTML 結構要維持一致，以便螢幕閱讀器的使用者也能以正確的順序閱讀，後續文章會再深入探討）



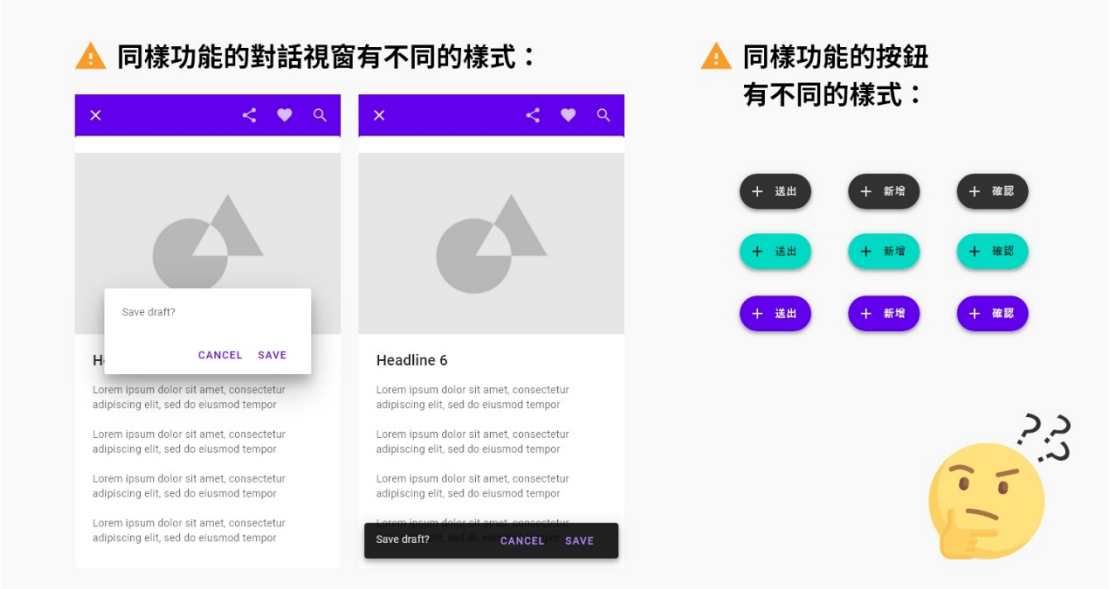
桌面與手機版的排列流向差異

相關指引及準則：

成功準則 1.3.2：有意義的序列 (檢測等級 A)
當內容中的呈現順序會影響其意義時，應該要能以程式化的方式，判讀正確的閱讀序列。

元件具有一致性

和前面提到的頁面結構應具備一致性同樣的理由，相同功能的元件也應該具有一致性。如果某按鈕在多個頁面中有不同描述、顏色、形狀，就容易造成使用者的混淆，以為這是一個全新的、具有不同功能的按鈕。



容易造成混淆的元件

相關指引及準則：

成功準則 3.2.4：一致的識別 (檢測等級 AA)
在一組網頁中，具有相同功能性的元件，就要有一致的識別。

大方向看完，是否覺得其實也沒有很困難呢？希望這篇文章能讓大家注意到一些平常使用網站時不會發現的眉眉角角！下一篇文章將開始討論設計

細節，各種元件有哪些注意事項、常見的文字與背景顏色對比度問題，都將以實例說明～

[註 1]　延伸閱讀：《用「完形心理學」，增加介面設計有感度!》
[註 2]　《Content reordering》這篇文章中有實際的範例，可以用 tab 鍵操作看看無序的內容

參考文章及素材

- How To Make Your Website Accessible to People with Disabilities
- 我們都應該了解的 無障礙網頁 概念（上面這篇的翻譯）
- Material Design: Accessibility
- MDN Web Doc: 無障礙網頁
- 以設計層面來思考無障礙網站
- 介面素材使用 Material Design UI Kit
- 疑惑臉 emoji 來自（Emoji Icon made by）: Freepik from www.flaticon.com

漫談無障礙網頁設計-3

內容目錄

上一篇講了無障礙網頁在設計上的大方向，這一篇將從各種元件的細節來討論，從尺寸、色彩、文案、影音……究竟設計上還有什麼值得注意的地方？

TL;DR

- 標題與內文的樣式應能有所區別，內文的預設文字大小通常是 16px
- 可點擊元件的尺寸寬高不應小於 44px
- 文字與背景顏色需要達到一定的對比度，大尺寸文字的對比度可以比小字低一些
- 不僅只以顏色傳達狀態，加入符號或文字，讓無法辨認顏色的使用者也能順利理解
- 在圖片有附圖說的情形下，圖說應該偏向補充圖片中沒有的額外、衍伸訊息
- 連結、提示文字要能讓使用者在任何情況下都能明確知道目的
- 會自動播放的動態內容、影片或音訊，需要有暫停的功能
- 網頁中不能有任何一秒內會閃爍三次以上的元件

文字、元件的尺寸

標題及內文尺寸

不同層級的標題、內文文字的樣式應能有所區別，例如較重要的文字有突出的文字大小、粗細等等。

Material Design 的標題層級設計

內文文字不應過小–– 16px 是一個不錯的標準，這是各瀏覽器的預設文字尺寸，也是一般而言建議的內文字尺寸。不過，這個數值並沒有硬性規定，由於有需要的使用者可以自行調整畫面和文字的縮放比例，因此條文僅要求頁面內容應可被放大至 200% ，且在放大或調整文字段落樣式的情形下，都能被正常地閱讀（這部分牽涉到前端程式的設定，會在後續的文章說明）。

段落樣式也值得注意，盡量能有：

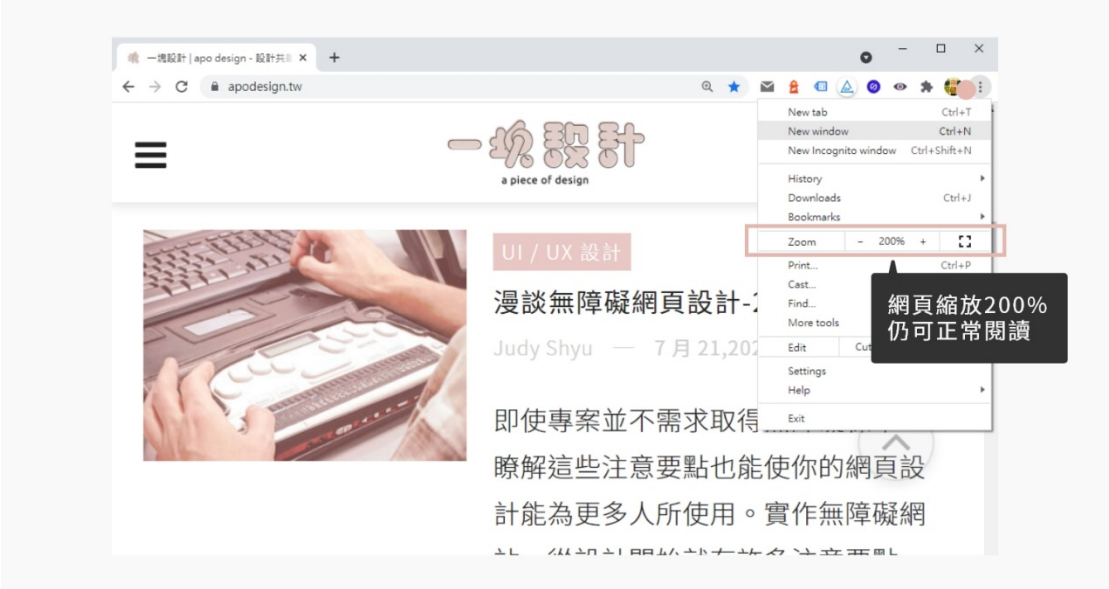
1. 充足的行高
2. 充足的段落間距（應該比行高高）
3. 充足的字元間距
4. 合理的文字對齊（尤其是拉丁文字應避免左右對齊（ `text-align: justify` ），因為斷行的關係，會使閱讀性下降）
5. 充足的欄位寬度

有興趣瞭解更多，可以參考之前與字型相關的文章《To 設計師：入門網頁字型指南 -5（設計彈性與多語系網站）》。

相關指引及準則：

成功準則 1.4.4：調整文字尺寸 (檢測等級 AA)
除字幕及影像文字外，文字在沒有額外輔助科技的情況下，要能夠放大至百分之兩百，而不會失去內容或功能性

成功準則 1.4.12：文字間距 (檢測等級 AA)
使用支援以下文字樣式屬性的標記語言實現的內容中，透過設置以下所有內容且在不更改其他樣式屬性下，不會喪失任何內容或功能性：
- 行高至少為字體大小的 1.5 倍；
- 段落間距至少是字體大小的 2 倍；
- 字元間距至少為字體大小的 0.12 倍；中文字元 0.14 倍。
- 字間距至少為字體大小的 0.16 倍。
例外：在書面文字中並未使用一個或多個這些文字樣式屬性的人類語言和腳本，可以使用專門對應該語言和腳本組合的屬性值。
註：中文內容的文字間距可以採用上述的行高和段落間距要求，字距則可參照一般中文出版業的要求。



網頁內容放大 200%後仍能正常閱讀

可點擊元件的尺寸

可點擊元件應注意寬高不小於 44px ，這項規範是為了確保手指點擊容易被觸發（理解一下有香腸手的痛苦！），WCAG 2.1 [1]和台灣的網站無障礙規範[2]規範的大小是 44px，業界也很常見寬高不小於 48px 的標準[3]。
列舉幾個常見點擊範圍不足的情形：

- 只有 icon 的按鈕（例如社群媒體按鈕）

- 麵包屑文字（通常行高不足）
- Footer 內的連結

如果專案的設計風格較精緻、想使用小 icon 來做為按鈕，這時候可以預留周圍空間作為點擊範圍。

按鈕視覺大小 v.s. 按鈕實際可點擊大小

相關指引及準則：

成功準則 2.5.5：目標尺寸 (檢測等級 AAA)
除以下條件外，指標輸入的目標尺寸至少為 44 乘 44 CSS 像素：
- 等效：目標可透過同一網頁上等效的鏈結或控制元件獲得，該目標尺寸至少須為 44 乘 44 CSS 像素；
- 行內：指標操作目標位於句子或文字區塊內；
- 使用者代理控制：指標操作目標的外觀是由使用者代理決定而非由網頁作者修改；
- 必要性：指標操作目標特定的呈現方式對於資訊的傳遞有其必要性。

顏色及對比度

文字與背景的對比度

無障礙規範裡和畫面也非常直接相關的一項，就是文字與背景的對比度。這裡提到的不只限於真正的文字和網頁背景色，也包含圖片中有提示意義的文字與其背景（圖片上的文字我們還是應該盡量避免）。

規範上，不同大小、粗細的文字有不同要求，小字需要的對比度較高； AA 級 和 AAA 級的對比度要求也不同，AAA 級要求的對比度較高。 AA 級要求 4.5:1 、大尺寸文字 3:1 ； AAA 級則要求 7:1 、大尺寸文字 4.5:1 。其中，對比度的小數點不能四捨五入，必須大於等於規範的對比度才算符合要求。

另外，雖然沒有明文規範，但功能性的 icon 也應該盡量具備與文字同樣的色彩對比度[4]。
每次看到這些數值頭都要暈啦！首先說明一下文字尺寸的差別，此處的文字尺寸都是指使用者沒有進行縮放下的尺寸，也就是原始的設計尺寸。

大尺寸文字的定義（根據 WCAG 2.1 [5]）：

最少 18pt 或粗體的 14pt 。這邊的單位不是 px 而是 pt，大約等於預設情形下的 1.2em 和 1.5em（約莫是 19.2px 和 24px）；不過這還要根據實際使用的字體及語言來做判斷——同樣的設定，不同字體看起來的大小不一定相同，而且中文、日文、韓文要達到可清晰辨認的尺寸又較羅馬文字大些。

這邊以純白背景、純黑背景兩個實際範例說明：

灰階文字與純白色背景對比度比較表

灰階文字與純黑色背景對比度比較表

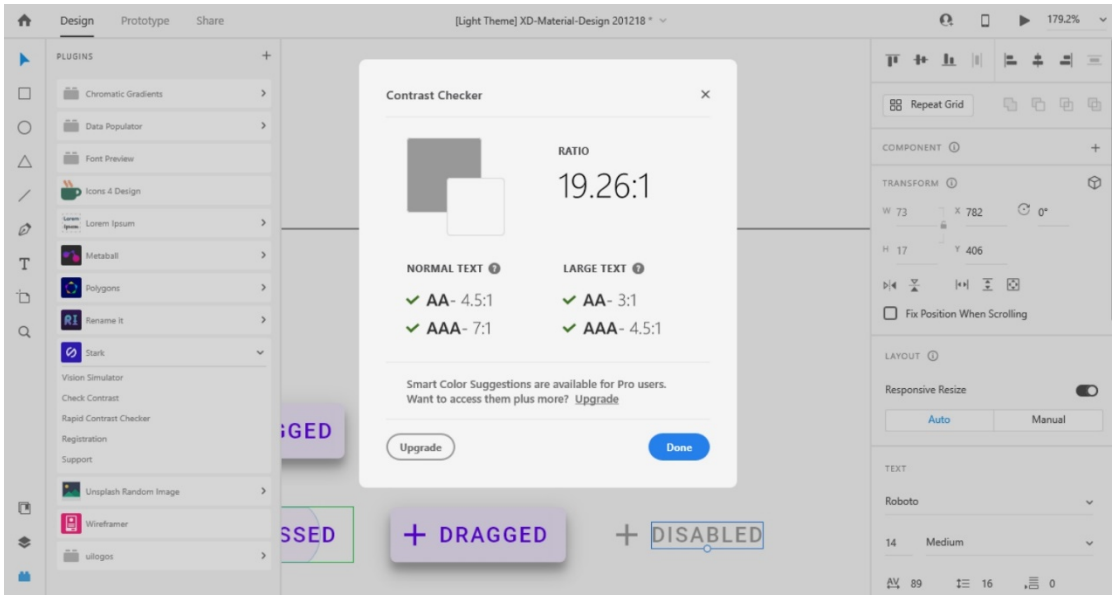
這個對比度又是怎麼算的呢？

實用計算工具：

實務上我們不會自己去算對比度，別擔心！單純想要線上計算顏色對比度的話，這個幾個網址都很好用：

- WebAIM: Contrast Checker：該有的功能都有，雖然畫面比較傳統，但可以直接調整顏色亮度和明度相當實用，下方的範例也很清楚，使用起來快狠準。
- Color Contrast Checker – Coolors：畫面清晰美觀、有多種調色盤模式可以直接選色。
- contrast ratio：簡單粗暴，輸入色碼，給你對比度，而且支援透明色。

對設計師來說，則有這個超級實用的套件 STARK ，支援 Figma、Adobe XD、Sketch，也有 Chrome 套件 ，非常便利！



STARK 在 Adobe Xd 中的截圖

若不想額外裝插件，Chrome 的 devTool 本身也有判斷對比度的功能，只是容易遇到程式無法判讀的情形，像是圖片背景、偽元素背景……等等，devTool 目前也還沒辦法直接吸色做判斷，只能麻煩一點先找個能夠判讀的元件，再用這個元件來吸色。

相關指引及準則：

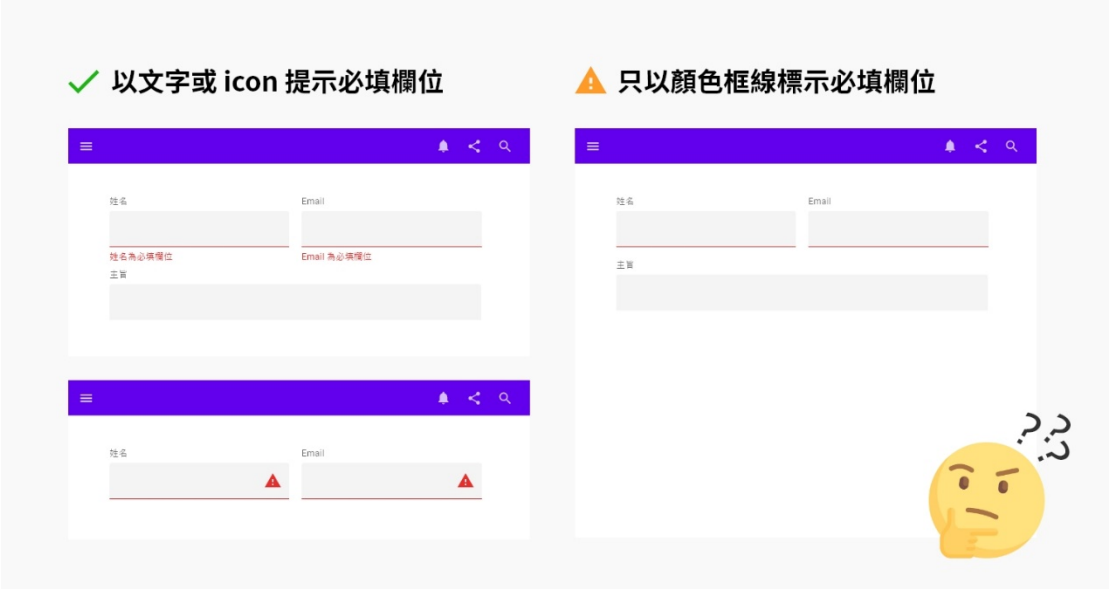
成功準則 1.4.3：對比值(最小) (檢測等級 AA)

除非是下列各款中的例外情形，否則文字及影像文字的視覺呈現，至少要有 4.5:1 的對比值：

1. 大尺寸的文字及大尺寸的影像文字至少要有 3:1 的對比值。
2. 閒置中的使用介面元件上的、純裝飾用的、任何人都看不到的文字或影像文字，或者只是另一張圖片的局部且該圖片顯然還有其他視覺內容，都毋須要求對比值。
3. 標識或商標名稱上的字樣沒有最小對比值的要求。

不僅只以顏色傳達狀態

顏色在設計上是很重要的溝通手段，可以呈現重要性、狀態、情緒調性等等重要的訊息。不過，加入無障礙考量後，為使色盲、色弱[6]、弱視者也能順利理解網頁內容，應注意不要將顏色作為傳遞訊息的唯一方式，加入提示文字或是其他足以區辨差異的符號或紋路吧！



不僅只以顏色作為唯一的訊息傳達方式的表單提示範例

Chrome 也有一些套件可以模擬不同類型色盲閱讀網頁時看到的樣子，例如：

- Colorblindly：不用做太多設定，點開、選特定的色盲模式即可預覽當下的頁面，個人推薦~
- Colorblind – Dalton for Google Chrome：有比較多細節可以設定，選完後按 Reset 就能套用到當下的頁面

相關指引及準則：

成功準則 1.4.1：色彩使用 (檢測等級 A)

色彩不可當做唯一能傳達資訊、提出動作、提請回應或區別視覺元件的視覺手段來使用。

提示文字及圖片描述

這個有點偏向內容產生的工作，不過因為設計上有時也會運用到「圖說」（caption，可以使用 HTML tag `figcaption`），所以一起在這邊帶到。

圖片描述

一般而言，設計應盡量避免在圖片中放資訊性的文字，因為螢幕閱讀器無法讀取圖片。螢幕閱讀器呈現圖片的方式是朗讀圖片的 `alt` 屬性，不過其實 `alt` 的使用範圍除此之外還有許多，例如爬蟲讀取網頁的時候，也是

靠 alt 來辨認圖片內容；另外，在沒有特別設計替代圖片的情況，當圖片路徑失效時，瀏覽也會顯示 alt 的文字作為代替。

圖片失效看到 alt 的情形

在圖片有附圖說的情形下，應注意圖說與 alt 的內容不要重複，且 alt 要在能清楚描述圖片內容的前提下盡量精簡，避免過於寬泛或冗長。因為螢幕閱讀器會將兩者都朗讀出來，考慮閱讀時間，這對使用者來說反而是種負擔。

我們可以思考一下兩者目的的不同：alt 忠實描述圖片視覺上能看到的內容，而圖說則用以補充額外的、衍伸的訊息。以下面這個範例來說明：

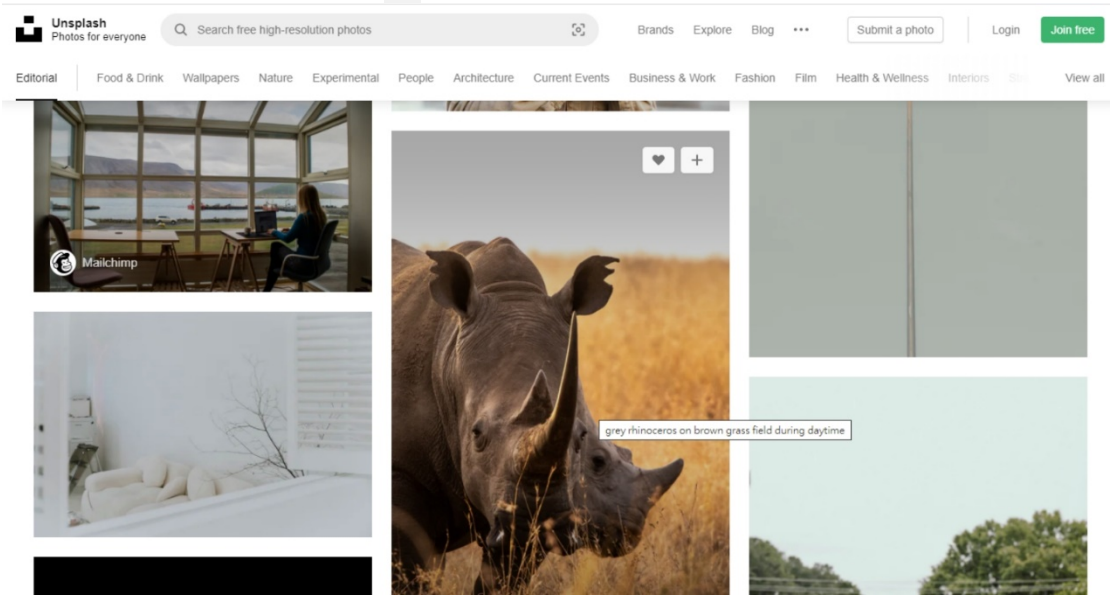
 奶泡的流量和速度是拉花技巧中重要的一環，倒得太多或太快都會使圖案變形難以控制。

```
<figure>
<img src="..." alt="牛奶倒入杯中製作拉花圖案的特寫" />
<figcaption>奶泡的流量和速度是拉花技巧中重要的一環，倒得太多或太快都會使圖案變形難以控制。</figcaption>
```

</figure>

可以看到 alt 單純敘述了照片的內容，而 figcaption 則撰寫了與上下文較相關的延伸內容。

順帶一提，素材網站 upslash 裡，可以看到有不少圖片的標題都是直白描述照片內容，是很不錯的 alt 撰寫方式參考！



upslash 截圖，可以看到圖片名稱精準描述圖片內容

這裡就先不提 alt 的各種寫法建議，設計師們可以先專注於「圖說」內容，盡量達到與圖片本身的內容有所差異即可。[7]

文字連結

清楚地描述連結讓使用者得以預測連結內容，進而能自由決定是否跳過他們不感興趣的頁面，節省瀏覽整個網站的時間。這對有運動障礙或認知障礙的使用者來說相當重要，有些輔助閱讀裝置甚至能直接列出網頁中所有的連結清單，這時，明確的描述就能讓使用者馬上找到自己想看的頁面。

不過也有例外情況，如果連結的目的就是想讓使用者無法猜到點開來會導向什麼頁面，例如連結猜謎或是有隨機支線的互動遊戲，那這個連結本身在畫面上只有模糊的說明就沒有問題。

相關指引及準則：

成功準則 2.4.4：鏈結目的(脈絡) (檢測等級 A)

每一個鏈結的目的可以透過鏈結文字本身或以鏈結文字加上能以程式化判定的鏈結脈絡予以確認，除非鏈接的目的對整體使用者來說均不明確。

提示文字

在設計較複雜的介面時，有時會使用提示文字來引導使用者進行操作，此時要注意提示文字、替代文字的精準度。

一個常見的情形就是，在一個有各種類型內容的網站，其首頁可能各有幾篇不同類型的文章預覽，例如最新消息、精選案例、讀者投稿……等等。這時，與其每個區塊都放一個「閱讀更多」按鈕，不如更精準地寫「更多最新消息」、「更多精選案例」…… 能讓按鈕的功能更加明確。

另外一個值得討論的情形則是具有方向性的提示文字。比較常見的例如：導航列在桌面版位於左側、但手機版改為在上方的情形，這時若有提示文字，可以避免使用「回到左方導覽列」這種有具體方向描述的寫法，單純寫「回到導覽列」即可。

動態內容、會閃爍的內容及音訊

動態內容

網頁中有持續性的動態內容，除非這個動態是頁面的必要元素（比如說點擊標靶的遊戲），否則都要可以由使用者操作暫停。

比較常見的就是會自動播放的影片了，需要給它一個暫停 / 重新播放或乾脆隱藏整個影片的按鈕；另外過去很流行的捲動元素（marquee）[8]也都屬於此範疇。

相關指引及準則：

成功準則 2.2.2：暫停、停止和隱藏 (檢測等級 A)
對於會移動、閃動、捲動或自動更新的資訊來說，下列各款全部都要做到：
1. 對於任何會移動、閃動或捲動，且為(1)會自動開始、(2)維持超過 5 秒鐘、(3)與其他內容平行呈現的資訊來說，除非這種移動、閃動或捲動乃是活動的一部分且不可或缺，否則就要有個機制來讓使用者可以暫停、中止或加以隱藏。

2. 對於任何會自動更新，且為(1)會自動開始、(2)與其他內容平行呈現的資訊來說，除非這種自動更新是內容的一部分且不可或缺，否則就要有個機制來讓使用者可以暫停、中止或加以隱藏，或能控制更新的頻率。

會閃爍的內容

禁止會一直閃爍的畫面效果，因為快速閃動的畫面有可能引發癲癇（可以瞭解一下 3D 龍事件[9]）。這項規範有非常明確的數值––不可以有任何元件在一秒內會閃爍三次以上。這種閃爍就算只持續一秒也不行，不像上面提到的動態內容，不能藉由提供暫停或關閉的選項來符合規範。
如果你的專案需求有包含必須閃爍的元件（例如遊戲產業常見的閃電、槍焰等等），那麼降低閃爍的速度是你的好選擇，更進一步，也可以試著降低閃爍的範圍或對比度來避免達到閾值。

相關指引及準則：

成功準則 2.3.1：閃爍三次或低於閾值 (檢測等級 A)
網頁上不可含有任何一個元件，其在任何 1 秒鐘之內，會閃爍超過 3 次，或者閃爍低於一般閃爍以及紅閃爍閾值。

音訊

與自動播放的影片類似，音訊也要能夠被終止或能夠調整音量，常見的處理方式是提供靜音 / 取消靜音按鈕。

靜音按鈕範例（畫面擷取自 moooi ）

相關指引及準則：

成功準則 1.4.2：音訊控制 (檢測等級 A)
如果網頁上有任何音訊會自動播放達 3 秒鐘以上，應提供一套機制來暫停或中止音訊播放，或者要能在整體系統音量設定外，另外提供控制音量的機制。

以上就是與設計相關的無障礙規範元件實例與介紹，這些細節方面的規範在實務上超級容易遇到的！雖然一路看下來可能會覺得很複雜，但其實不少內容都可以靠插件工具檢測，馬上檢視一下自己過去的專案有沒有設計方面的錯誤提示（檢測結果通常大部分都是前端要改的東西，請過濾一下……　　），就很容易能知道有哪些地方可以改進啦！

不需要給自己太大的壓力，除非是手上的專案有取得無障礙標章的需求，否則並不用追求自己的設計馬上就要完美達成所有規範。期望大家讀完這些內容後，逐漸內化到自己設計思路之中，讓設計的網頁能被更多人所使用！

[註 1]　WCAG 的第 2.5.5 項指引 Target Size 《Understanding Success Criterion 2.5.5: Target Size》

[註 2] 第 2.5.5 項指引：目標尺寸（檢測等級 AAA）

[註 3] 例如 Google ：《Accessible tap targets》（這篇也有實際範例可以玩玩看）和 Android 的 Material Design：《Layout and typography》

[註 4] Material Design: Contrast ratios

[註 5] WCAG 2.1: Definition of large scale

[註 6] 你知道嗎？男性 12 人中就有 1 人（8%）、女性 200 人中就有 1 人（0.5%）為色弱所困擾。延伸閱讀：每一个色盲玩家都被泡泡龙和祖玛虐哭过 這篇文章深刻地表達了紅綠色盲（色弱）者在使用純以顏色導向的設計時所面臨的痛苦。

[註 7] 延伸閱讀：《Alt vs Figcaption》

[註 8] <marquee>: The Marquee element （注意！這是已經從標準中移除的 HTML tag，請不要在正式專案使用他）

[註 9] 3D 龍事件

參考文章及素材

- web.dev: Color and contrast accessibility
- WCAG 2.1 Checklist-Guide to Accessibility
- Understanding Success Criterion 2.4.4: Link Purpose (In Context)
- Understanding Success Criterion 2.3.1
- Images | Understanding Web Accessibility | Web Accessibility | Brandeis University
- 疑惑臉 emoji 來自（Emoji Icon made by）: Freepik from www.flaticon.com
- 介面素材使用 Material Design UI Kit

漫談無障礙網頁設計-4

內容目錄

在前端實作無障礙網頁不可不知的 HTML

這個系列的前三篇文章大致描述了無障礙網頁的基本概念，以及設計師們可以注意的大方向和細節，接下來將從前端實作的角度討論網頁結構有哪些值得注意的地方，HTML 要怎麼撰寫才能符合無障礙規範。這篇文章將聚焦介紹常用的與頁面結構相關的 HTML 元素，至於互動性的元素以及 ARIA 屬性則會在下一篇提及。

大前提：結構的正確性

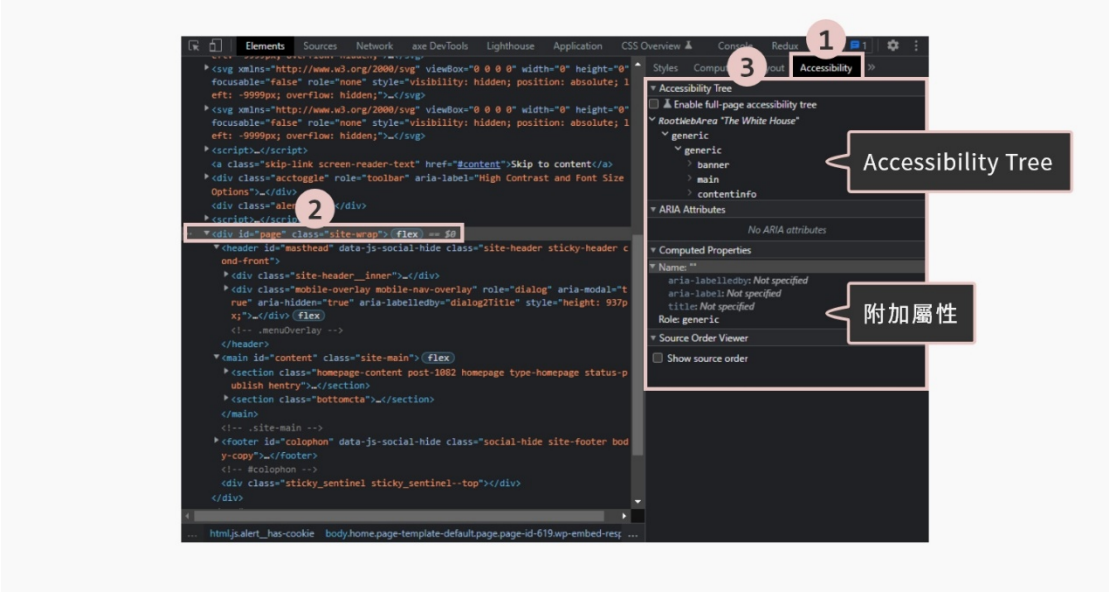
諸如 HTML tag （ HTML 標籤[1] ）缺少對應的頭尾、屬性缺少引號、大小寫不符合規範、屬性之間缺少空格……等等都可能導致畫面呈現錯誤，HTML 結構的正確性不只是為了符合無障礙，也是為了整個頁面架構的正確性，可說是撰寫 HTML 時最需要注意的大前提。

如果使用預處理器（ preprocessor ）通常不會有這類問題，各大編輯器也都有原生功能或套件可以協助做 HTML Lint；如果想要看更詳細的檢測報告，也有 線上服務 可以協助檢查 HTML 是否符合規範。

使用具備語義的 HTML tag

如果說視覺化地認知網站是藉由元素的排列和樣式來理解各區塊和元件的群組、位置和功能，那麼螢幕閱讀器使用者就是靠著語義化的 HTML tag 來幫助理解當前這個元件在哪裡、可以做什麼、應該如何互動。

瀏覽器會根據頁面的 HTML 建立 DOM Tree，並依據 DOM 元件上的資訊（ ARIA 屬性、title …… ）建構出 Accessibility Tree [2]。螢幕閱讀器則會根據 Accessibility Tree 進行朗讀，使用 Chrome 的 devTool 可以檢視頁面的 Accessibility Tree ：



利用 Chrome 的 devTool 檢視 Accessibility Tree

Chrome 目前版本（ 103.0 ） devTool 有一個實驗性的功能，可以直接把 Accessibility Tree 顯示在左邊，更方便檢視：

新的實驗功能讓我們能更容易檢視 Accessibility Tree

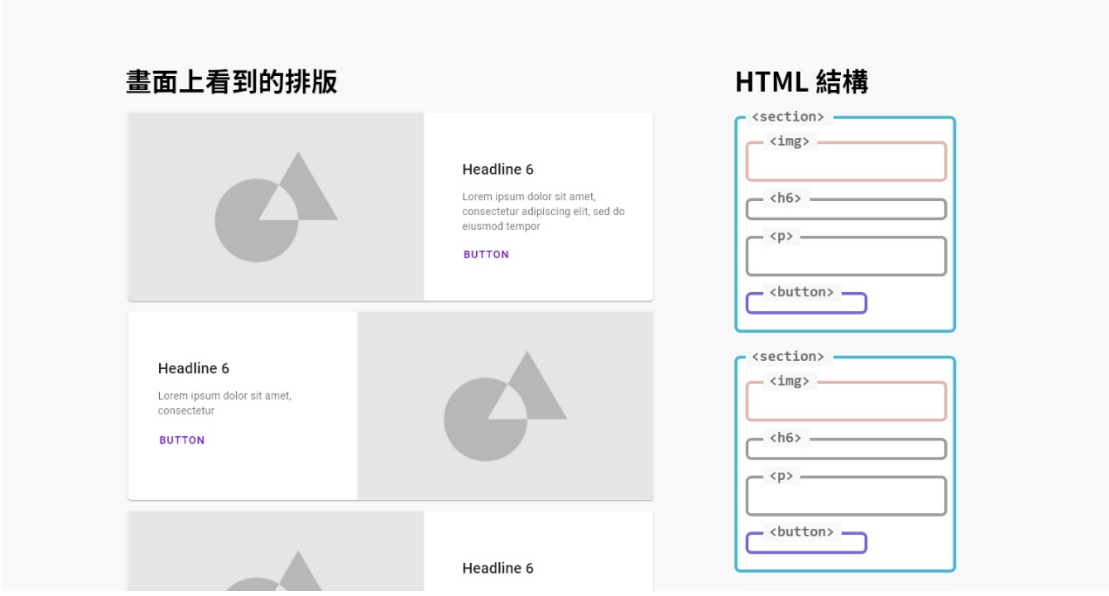
2014 年推出的 HTML5 標準，新增了非常多語義化標籤（Semantic HTML Element），讓搜尋引擎爬蟲和螢幕閱讀器都能更容易辨識網頁內容。

針對前端切版的初學者，推薦閱讀一遍 W3C School 這篇按照類別排序的 HTML 標籤列表，先大致瞭解有哪些標籤可以使用。閱讀的時候也不需要把全部標籤都背起來，只要在腦中建立印象，對實際撰寫 HTML 就很有幫助，不至於因為不知道有特定標籤能用而直接寫 `<div>` ，遇到不熟悉的類型只要回頭查閱就可以了。

HTML 結構順序

螢幕閱讀器會依照 Accessibility Tree 的順序將內容提供給使用者，所以在前面文章[3] 提過的流程邏輯與頁面結構的一致性、內容的層級與順序，在 HTML 的實作也一樣需要注意。

設計上為考量使用者觀看網頁的節奏，有時會將畫面的結構順序做調換，遇到這種情形，撰寫 HTML 結構時仍應該維持有意義、有邏輯的順序。善用 CSS 選擇器，配合 `float` 、 `order` 或 `grid` 等等方式做出變化。



畫面呈現與實際的 HTML 流向

HTML tag 類型和 `role` 屬性

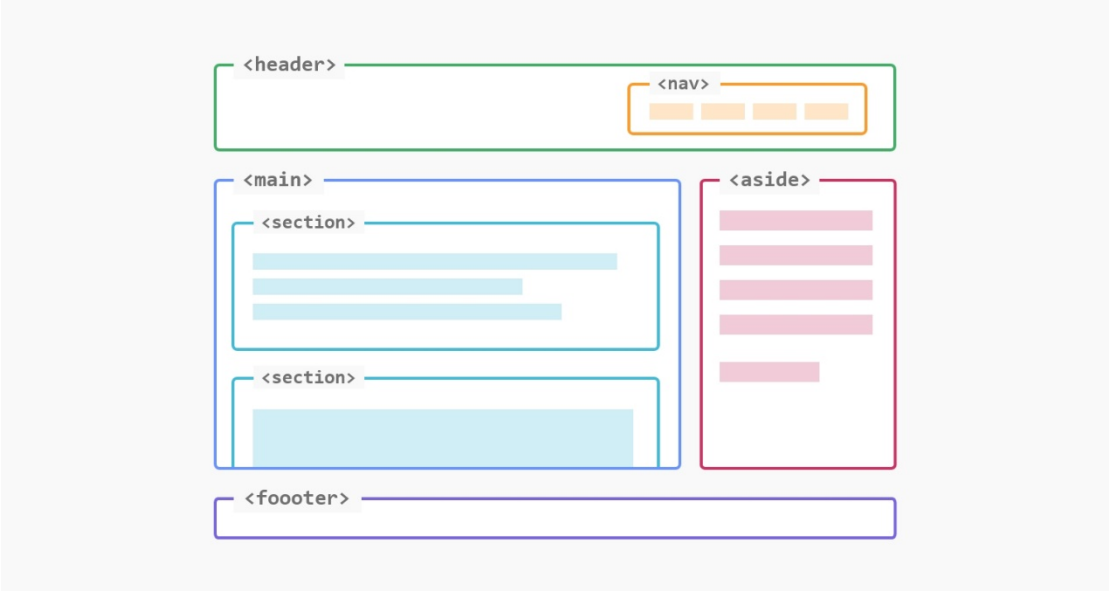
過去沒有這麼多語義化標籤時，會使用 `role` 屬性來補充描述元件的角色功能， HTML5 推出的語義化標籤吸納了不少原本需要寫落落長屬性才能表達的含意。

在實作上，我們應該優先使用語義化 HTML tag ，沒有可採用的 HTML tag 時才使用 `role` 作為補充；因為原生的 HTML tag 在瀏覽器中通常都具備鍵盤可訪問性，這類元件如果選擇使用 `<div>` 搭配 `role` 來呈現，就必須自行撰寫 `tab-index` 和額外的 javascript 來模仿瀏覽器行為。
在一頭栽進 ARIA 的世界之前，下面會按照類型先介紹一些常用的 HTML tag 。

Landmark

網頁的 landmark 目前好像沒有正式的翻譯[4] ，不過觀察下面幾個標籤，就會發現它們都是定義網頁主要架構、上層的元件，如同單字本身的意思，它可說是網頁的「地標」或「最主要的結構區塊」。
在無障礙的實現上，landmark 可以說是最重要也是最容易實作的類型，有下面這幾種 HTML tag 可以使用[5] ：

- `<header>`
- `<nav>`
- `<main>`
- `<aside>`
- `<section>`
- `<form>`
- `<footer>`



常見的網頁 Landmark 結構

landmark 可以輔助導航功能，能讓螢幕閱讀器的使用者跳到特定的區塊進行閱覽，這也是為什麼 WCAG 建議所有的頁面內容都應該被放在 landmark 結構中。這同時意味著 landmark 的使用應該盡量精簡，以免讓使用者難以理解網頁主結構。

原則上，一個頁面應該只有一個 `<main>`，其他 landmark 在有複數個同時存在時，需利用 `aria-label` 來區別其目的。例如頁面常見
在 `<header>` 、 `<footer>` 裡有多個 `<nav>` ，此時就可以針對內容，標示 `<nav aria-label="Header">` 、 `<nav aria-label="Footer ">` 、或是 `<nav aria-label="Social">`。
要注意， `<header>` 、 `<footer>` 常見被作為子區塊結構（例如獨立的文章有自己的標題和結尾區塊），這時它們不會被認定是 landmark role；只有在位於第一層（直屬於 `<body>`）的情況下， landmark HTML 元素才會被認定為對應的 landmark role。 `<section>` 、 `<form>` 則是需要有可供辨識的名稱如 `aria-labelledby` 、 `aria-label` 或 `title` 屬性，才會被認定為 landmark role。關於 ARIA role 的類型會在後面的文章介紹。

標題

標題是讓使用者（無論是否借助螢幕閱讀器）快速瞭解頁面結構的重要元件，標題的 HTML tag 就是單純的 `<h1>` ~ `<h6>` 。撰寫標題結構要注意：

1. 由`<h1>` 到 `<h6>` ，按照層級順序撰寫；
2. 下層內容與其所屬的上層內容相關；
3. 須從 `<h1>` 開始，且不可以跳層。

標題順序的 DOs & DON’Ts

段落和內容

- `<article>`
- `<blockquote>`

- `<ul>` 、 `<ol>`
  - 兩者的差別在於`<ul>` 是無序、`<ol>` 則是有序的列表。
- `<li>`
  必須被直接放在 `<ul>` 或 `<ol>` 內。
- `<p>`
- `<br>`
  - 避免使用 `<br>` 來呈現 CSS 可以達成的間距，例如將兩個段落的文字內容都放在一個 `<p>` 裡面，中間加一堆 `<br>` 來做出間隔，「兩個段落」顧名思義就是兩個 `<p>`，別再偷懶啦！
- `<del>`
- `<em>`
  - 相較單純只呈現樣式的 `<i>` （斜體字），使用 `<em>` 可以讓螢幕閱讀器使用者也接收到「這段內容可能有不同解讀」的語義 [5]。
- `<strong>`
  - 相較單純只呈現樣式的 `<b>` （粗體字），使用 `<strong>` 可以讓螢幕閱讀器使用者也接收到「重要內容」的語義。
- `<ins>`
- `<sub>`、`<sup>`
- `<hr>`
- `<code>`
- `<time>`

圖片

- `<figure>`
- `<figcaption>`
  - `<figcaption>` 必須是 `<figure>` 內第一個或最後一個元件。
- `<img>`
  - 除非是裝飾性的圖片、或相鄰元件就具備描述性的文字，否則 `<img>` 都應該要有能描述圖片內容的 `alt`，讓螢幕閱讀器使用者能讀取圖片。

常見的圖片結構

表格

- `<table>`
- `<caption>`
  - `<caption>` 必須是 `<table>` 內的第一個元件。
- `<tbody>` 、 `<thead>` 、 `<tfoot>`
- `<tr>`
- `<th>`
- `<td>`

由於 `<table>` 本身的樣式特性，有時用 `<table>` 難以達成特殊的設計排版，這時用 `<div>` 配合 `role` 就可以表達 table 的語義結構。

下一篇會接著深入介紹表格元件。

[註 1]　為避免和 `label` （標籤屬性）、 `aria-label` （ARIA 標籤）混淆，這系列的文章會一律以 HTML tag 表示。

[註 2]　web.dev: The Accessibility Tree

[註 3]　延伸閱讀：《漫談無障礙網頁設計-2》

[註 4]　本文提到 landmark 大多會使用原文，少數段落使用「地標」作為中文翻譯（參考 MDN Web Docs 的中文翻譯）

[註 5]　定義來自 HTML5 draft。實務上 `em` 也被用來強調段落中的文字，不過強度比 `strong` 低。

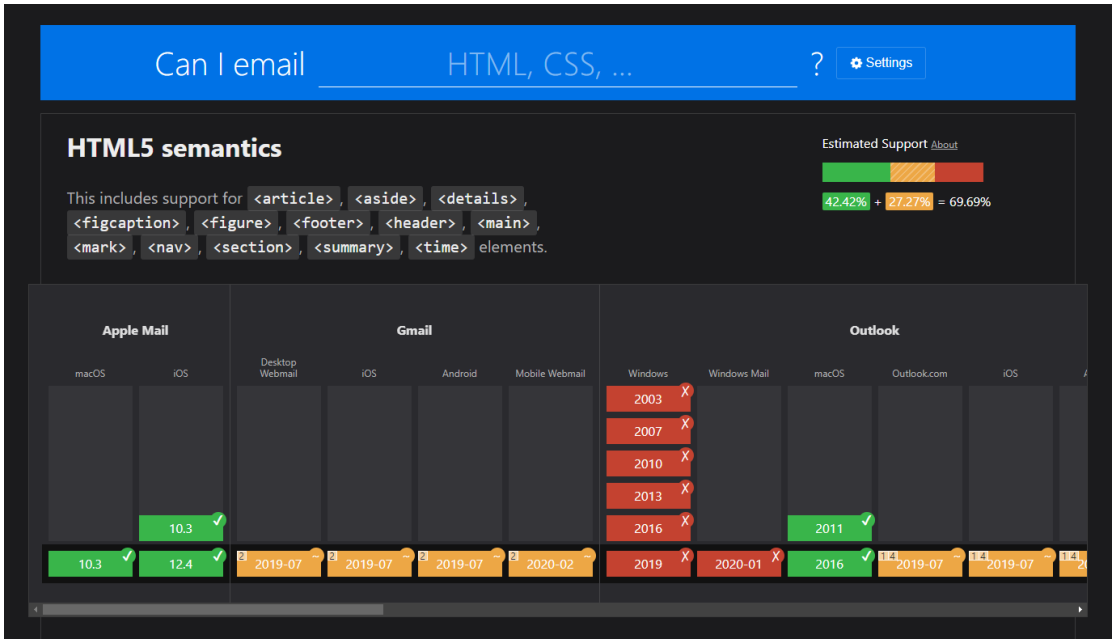
漫談無障礙網頁設計-5

內容目錄

上一篇文章最後提到表格使用的 HTML tag `<table>`，實務上 `<table>` 可說是應用方式相對複雜一種元件類型，因此這裡會再花一點篇幅來深入說明。

表格的使用

網際網路剛推出，網站內容還沒有這麼多豐富變化，甚至 CSS 還沒推出之前[1]，`<table>` 是用來形塑網站結構的主要工具，有接觸過 Frontpage 、Dreamweaver 等工具的讀者大概都不陌生，過去網站所有的內容可以用一層一層的 table 架構完畢（現存的經典案例必須要提 阿部寬的官網 XD）。如果有在處理 EDM （Electronic Direct Mail，也就是電子報），也不可避免需要大量使用 `<table>` 。這是因為 email 的用戶端非常多–– Outlook 就有好幾版，iOS、iPadOS、Apple Mail、Yahoo! Mail、Gmail……族繁不及備載，而這些用戶端對 HTML、CSS 的支援度幾乎都比瀏覽器差，針對 email 甚至有一個獨立的 Can I email… 網站，可見一斑。



語義化 HTML tag 在 email 用戶端的支援度（2022 年 9 月）

從上圖可以看到幾個主流用戶端對語義化 HTML tag 的支援度都未臻完整，更別提右邊畫面外還有超過十個以上的其他用戶端支援度參差不一。

說了這麼多，就是要表達瞭解如何使用 `<table>` 也是很重要的！關於表格，有三種常見的使用情境：

1. Data Table 資料表格
    - 使用 `<table>` 的 HTML 結構來呈現二維資料，即原生的寫法。
2. Layout Table 排版表格
    - 因為排版需求，用 `<table>` 來排版，實際內容並非表格資料；如上面提到 EDM、維護更新古早網站或任何用 `<table>` 切版較快的情形。
3. 模擬表格
    - 與資料表格接近，有表格的外觀和二維特性，但用 `<div>` 或 `<ul>` 、 `<li>` 搭配 CSS 來呈現表格資料。

下面分別介紹這三種情境的實作方式：

1. Data Table 資料表格

上一篇有提到語義化表格會使用到的 HTML tag：

- `<table>`

- <caption>
- <tbody> 、 <thead> 、 <tfoot>
- <tr>
- <th>
- <td>

一個完整使用到所有結構的表格大概會像這裡的範例一：

如果已經有一定的 HTML 基礎，應該都對如何使用表格結構不陌生，初學者則可以漸進式地加強你的表格結構：

A. 正確使用 <tr> 、 <th> 、 <td>

<tr> 代表的是 table row，<th> 代表的是 table header，<td> 則是 table data 。表格是二維的資料呈現方式，基本的表格由橫向的 row 和直行的 column 構成，橫向的 row 由 <tr> 來表示，而表格的每格單位則由 <th> 和 <td> 組成。所有的 <th> 、 <td> 都必須被包含在 <tr> 中。
正確使用 <th> 相當重要，因為在沒有其他額外屬性標示的情況下，螢幕閱讀器只能藉由 HTML 結構判斷哪些 table data 屬於哪個 table header。

B. 加入<caption> （和摘要 summary）

無障礙表格可以帶有標題，這裡指的標題並不是 <h1> ~ <h6> ，而是專門用來描述表格標題的 <caption> 。與提供 <figure> 標題的 <figcaption> 用法類似，不同的是 <caption> 必須放在 <table> 內的第一個元件，不能是最後一個（視覺上則可以藉由 CSS 來把標題的位置移到表格下方）。
若表格內容複雜，可以補充摘要（summary）來加強說明表格內容，摘要的寫法比較靈活，可以：

1. 直接在 <caption> 裡塞一個子元素（ …等）來放說明文字；
2. 在 <table> 鄰近用別的元素撰寫摘要 （要帶 id)，並在 <table> 上加入 aria-describedby="{id}" 來關聯兩者；
3. 把 <table> 包在 <figure> 裡，利用 <figcaption> 來撰寫摘要。

※ 注意不要使用 summary 屬性，這個已經不支援了。

C. 加入 <tbody> 、 <thead> 、 <tfoot>

其實就算程式碼裡沒有寫 <tbody> ，瀏覽器在產生 <table> 的 DOM 時也會加上。

如果你的表格結構帶有表頭 row 、匯總 row ，那麼就可以用 `<thead>` 、 `<tfoot>` 將 `<tr>` 包起來，清楚表達結構。不過這幾個 HTML tag 有比較多使用順序上的規範：

1. 一個 `<table>` 中， `<thead>` 、 `<tfoot>` 只能有一個，`<tbody>` 則可以有多個，用來區別複雜資料的區段；
2. 如果自行撰寫 `<tbody>` ， `<table>` 內就不能有直屬的 `<tr>` 元素，所有的 `<tr>` 都必須被放在 `<tbody>` 、 `<thead>` 或 `<tfoot>` 中；
3. `<tbody>` 、 `<thead>` 、 `<tfoot>` 內都必須有至少一個 `<tr>` ；
4. `<thead>` 必須放在 `<table>` 內的 `<caption>` 或 `<colgroup>` 之後，並在 `<tbody>` 、 `<tfoot>` 之前；
5. 相反地， `<tfoot>` 必須放在 `<caption>` 、 `<colgroup>` 、 `<thead>` 、 `<tbody>` 之後。

D. 加入屬性

加入屬性的目的是闡明資料之間的關聯，能讓螢幕閱讀器知道應該用怎樣的順序來閱讀表格，其中最重要也比較容易實作的屬性有[2]：

- `scope`：
  - 只能用在 `<th>` ，用來定義與其相關的資料內容的方向性， `scope` 可以有的值有：
    - `row`：表示同一個橫向 row 的 `td` 與其相關
    - `col`：表示同一個直排 column 的 `td` 與其相關
    - `rowgroup`：表示多個 row 的 `td` 與其相關，通常會搭配排版使用的 `rowspan` 屬性使用
    - `colgroup`：表示多個 column 的 `td` 與其相關，通常會搭配排版使用的 `colspan` 屬性使用
- `headers`：
  - 可用在 `<th>` 、 `<td>` ，表達這個元素屬於哪個 `<th>` ，其值為所屬 `<th>` 的 `id` （只能是 `<th>` 的 `id` ，不過可以同時有多個，用空白區隔）

看看實際的應用範例能更快理解（範例二及三）：

上面的範例為求清楚，將兩種屬性分開成兩個表格舉例，實際使用時 `scope` 和 `headers` 可以同時並存[3]。

E. 優化複雜表格

表格內容太多時，經常會有超出螢幕畫面不滾動就閱讀不到的情形，雖然直接把格子間距或文字縮小很直覺，但這樣反而會導致閱讀性不佳，遇到這種情況，複雜表格可以藉由幾種方式處理：

1. 精簡表格內容
2. 拆分成多個表格
3. 根據表格類型，甚至可以直接將表格內容整個設計成互動性的查詢功能頁

其中，要注意表格內容在簡化或拆分後是否還能正確表達原本要呈現的資訊。而第三種方式雖然整體成本較高，但對使用者來說會方便很多，例如下面這個郵資一覽表：

－ 信函

民國108年11月1日起實施 單位：新臺幣元

重量(公克)	普通	限時	掛號	限時掛號	掛號附回執	限時掛號附回執
不逾20	8	15	28	35	43	50
21-50	16	23	36	43	51	58
51-100	24	31	44	51	59	66
101-250	40	47	60	67	75	82
251-500	72	79	92	99	107	114
501-1000	112	119	132	139	147	154
1001-2000	160	167	180	187	195	202

備註：每件限重不逾20公斤，逾2公斤，每續重1公斤加收48元。

普通資費表：信函

觀察其內容其實是有公式可循的，使用者使用這張表的目的是查詢資費，那麼使用查詢表式就更直覺了：

國內函件資費查詢

2. Layout Table 排版表格

單純使用表格當作排版工具是非常不建議的，除了前面提到歷史因素和 EDM 的狀況以外都應該盡量避免。若因不可抗力必須要用 `<table>` 來排版，也應該注意：

1. `<table>` 內除了 `<td>` 以外不要使用任何其他上述的 HTML tag，以免造成更多語義上的混亂；
2. 在 `<table>` 加上 `role="presentation"` ，表明這是裝飾性的內容，不需要被讀出；
3. 不要把這種表格放在資料表格內，會造成資料表格的閱讀困難。

3. 模擬表格

如非必要，也應避免使用這種方式來撰寫表格。與其使用一大堆 ARIA 標籤，不如盡量使用原生的語義化 HTML tag ，對使用者更友善。`role="table"` 算是較新的 ARIA 內容，要注意瀏覽器、輔具，尤其是觸控裝置對它的支援度尚未完全普及。

模擬表格有資料表格的外觀和陣列特性，但因為排版需求不使用 `<table>` 的 HTML 結構，而是採用 `<div>` 或 `<ul>` 、 `<li>` 搭配 CSS 來呈現表格資料，這種情形需要搭配 `role` 來表明其表格結構；通常用在有特殊排版或 RWD 不同尺寸樣式變化巨大的表格。

模擬表格的整體結構會跟原生 `<table>` 差不多，藉由利用 role 屬性來標示對應的角色，能使用的 role 屬性和 HTML tag 其實都有對應，觀察下面的撰寫範例，就能發現基本的 role 屬性都是能夠對應 HTML tag 的：

[註 1]　講古小知識！HTML 在 1990 問世，當時的樣式表（style sheet）還沒有統一規範，直到 1996 年 CSS 才推出，而且一開始還不怎麼流行。`<div>` + CSS 的排版方式是 2000 年後才逐漸把 `<table>` 排版取代的。

[註 2]　參照 MDN Web Docs

[註 3]　更多範例可以參考：Tables Tutorial | Web Accessibility Initiative (WAI) | W3C

[註 4]　HTML spec；這篇的討論也有相當詳細的說明 Valid to use <a> (anchor tag) without href attribute?