2-2 CSS的寫法

選擇器、屬性、值

能夠掌握CSS中的「選擇器」、「屬性」、「值」三種用語相當地重要。「選擇器」是樣式的套用對象，「屬性」是樣式的種類，「值」則是樣式的具體內容或程度。例如，將段落（p元素）的文字顏色設定為紅色時，其指定是「p{ color:red; }」，此時，選擇器是「p」，屬性是「color」，值就是「red」。

選擇器、屬性、值

首先就以「h1」或「p」等XHTML的元素名稱來思考選擇器。除此之外，把ID和CLASS設為選擇器，或某元素所包含的子元素等寫法也都是可行的，稍候再詳細說明。

屬性和值是以冒號（:）做為區隔。整組的屬性和值則可以利用分號（;）做區隔，做出多筆組別的指定。舉例來說，就如同「p{ color: red; font-size: 120%; }」。但是，如果按照範例寫成1行，指定的屬性則會增加越多，閱讀上就會變得更不容易，因此，在每個屬性換行是比較普遍的做法。

指定多項屬性時

只要利用【Tab】鍵或兩格半形【空白】鍵，使換行的屬性縮排（indent），就會比較容易閱讀了。

註解的寫法

在CSS的程式碼中，寫上註解的做法是利用「/*」和「*/」將內容框起來。註解是製作者用來作記號或備忘用的，並不會影響到樣式的套用。例如，「/*套用標題的樣式*/」。註解中也可以進行換行，使長篇文章更容易閱讀。

另外，註解中沒辦法寫上註解（例如：「/*……/*……*/……*/」），這一點也請多加注意。

```
/* 套用標題的樣式 */
…
/*
  以下指定是調整Internet Explorer 6中的偏移之用。
  在選擇器的前面加上「* html」，進行樣式的套用。
  */
…
```

註解的寫法

可指定的值

在標準規格中，CSS的屬性各自都有「可指定的值」、「預設值」及「可套用的元素」。例如，指定排列背景圖像用的background-repeat屬性中，可指定「repeat」、「repeat-x」、「repeat-y」、「no-repeat」這些值，預設值是「repeat」，可套用的元素則是「所有的元素」。預設值就是當該屬性沒有指定時，會被自動設定的值。

也有可指定任意值的屬性。例如，指定字型大小用的font-size屬性就可以指定「120 %」、「10 pt」等任意值。

單位的省略

值指定為「0」時，可以省略單位（例如，「0 px」可寫成「0」）。

可使用的單位

在CSS中可利用各種不同的單位，代表性的單位如下。

長度的單位

以螢幕為前提的情況下，最常使用的是像素（px）、點（pt），以及把字型大小設定為1單位（em）等。

單位	說明	範例
pt	點，point（1/72 英寸）	12 pt
pc	皮卡，pica（1pica=12 pt）	1 pc
in	英吋，inch（2.54 cm）	3 in
cm	公分，Centimetre	1.5 cm
mm	公厘，Millimeter	10 mm
px	像素，pixel	16 px
em	以1為單位	1.2 em
ex	將小寫「x」的高度為單位	2 ex

小數點的使用

不只是整數（1、2、3等），小數點（1.2、2.5等）也可以使用，但若是太過細微，瀏覽器或裝置將無法呈現，普遍的做法是將小數點最多設定到第1位或第2位。

百分比

百分比（%）的指定也經常使用。

單位	說明	範例
%	百分比	150 %

顏色的單位

6碼的RGB值（#rrggbb）、3碼的RGB值（#rgb）、顏色的名稱（「red」等17色）都可以使用。

色彩名稱	顏色	RGB值
aqua	水藍色	#00ffff
black	黑色	#00000
blue	藍色	#0000ff
fuchsia	紫紅色	#ff00ff
gray	灰色	#808080
green	綠色	#008000
lime	黃綠色	#00ff00
maroon	褐紅色	#800000
navy	深藍色	#000080
olive	橄欖色	#808000
orange	橘黃色	#ffa500
purple	紫色	#800080
red	紅色	#0000ff
silver	銀色	#c0c0c0
teal	青綠色	#008080
white	白色	#ffffff
yellow	黃色	#0000ff

CSS 2.1中所認同的色彩顏色（17色）

URL

部分的屬性需指定圖像的路徑等URL。以「url（…）」的形式進行描述是較普遍的做法。

```
body { background-image: url(../images/body_bcg.gif); }
```

將值指定為「url」的寫法

文字列

在部分的屬性中可指定文字列。普遍的做法是採用「"…"」（以雙引號框住）的形式進行描述。

```
body:after { content: "頁面的內文到此為止"; }
```

將值指定為「文字列」的寫法

2-3 CSS的套用方法

在XHTML中套用CSS的方法

在XHTML中套用CSS的方法有下列四種。

1. 利用link元素讀取CSS檔

```
<link rel="stylesheet" type="text/css" href="css/styles.css" />
```

2. 利用@import讀取CSS檔

```
<style type="text/css">
@import "css/styles.css";
</style>
```

3. 利用style元素指定CSS來源

```
<style type="text/css">
p { margin: 15px; }
</style>
```

4. 利用style屬性指定CSS來源

```
<p style="margin: 15px;">內文</p>
```

以上列出的方法當中，最為理想的是 1. 和 2. 讀取CSS檔的方法，像 3. 和 4. 直接把CSS描述在XHTML中的方法並不理想。因為與結構毫無關係的CSS內容一旦寫入XHTML中，就無法徹底區分結構和外觀設計。另外，3. 和 4. 的樣式只能在撰寫的該頁面發生作用，如此一來，就無法統一管理整個網站的CSS，此為降低維護效能的一大缺點。建議僅用來作為暫時性、限定性的利用。

文字編碼在CSS中的指定

以 1. 或 2. 的方法所讀取的CSS檔的第1行中，通常都會預先指定文字編碼「@charset"…";」。

```
@charset "UTF-8";
```

在CSS檔中指定文字編碼（一定要在第1行）

在日文頁面中會使用「UTF-8」、「Shift-JIS」、「EUC-JP」等文字編碼（繁體中文頁面中，則是使用BIG5的文字編碼）。雖然這是在CSS檔中所使用的文字編碼，與XHTML的文字編碼無關，但若沒有特別的理由，就指定為相同的文字編碼。

另外，「@charset "…";」的前面不能撰寫選擇器或屬性、註解等，請務必記住，一定要寫在第1行。

套用CSS的媒體種類

也可以指定套用CSS的媒體（裝置或軟體）。例如，「screen」是把CSS套用於電腦螢幕的值，「print」是套用於印表機等列印媒體的值，「handheld」則是套用於行動裝置的值。

將CSS套用於電腦螢幕或投影機

```
<link rel="stylesheet" type="text/css" href="css/screen.css" media="screen, projection" />
```

將CSS套用於電視、電視遊樂器

```
<link rel="stylesheet" type="text/css" href="css/tv.css" media="tv" />
```

將CSS套用於印表機等列印媒體

```
<link rel="stylesheet" type="text/css" href="css/print.css" media="print" />
```

將CSS套用於行動裝置

```
<link rel="stylesheet" type="text/css" href="css/handheld.css" media="handheld" />
```

指定套用CSS的媒體

這些值稱為「媒體類型（media types）」。在讀取CSS的link元素中加上media屬性，以指定這些媒體類型作為值的方法，這是最普遍的做法，就像範例中的「media="screen"」。也可以利用逗號區隔值，同時指定多項媒體類型，例如，指定「media="screen, projection"」。@import和@media也可以指定媒體類型，詳細會在第7章進行解說（p.216）。

就實際應用而言，要讓網站對應何種媒體，全憑目標使用者的便利性作為考量。如果使用自家或公司的電腦存取的使用者佔大多數，就只對應「sereen, projection」或「print」，這樣的判斷也是可行的。如果在戶外使用行動電話存取的使用者較多，只要預先對應「handheld」，便利性便會提高許多。另外，Nintendo Wii等電視遊樂器也可以安裝瀏覽器，有媒體類型「tv」可對應，所以只要準備電視畫面專用的CSS，就能夠提供這類使用者更容易使用的頁面。

正確的XHTML+CSS網頁，原本就不分媒體。就算不套用CSS，無論在任何裝置或軟體上都可以顯示內容、取得情報，這是理所當然的事。可是，正如前面所陳述的，因為外觀對優使性有很大的影響，所以製作出最適合每個裝置的CSS，就能夠讓使用者在利用該網站的時候，變得更加容易。

媒體類型	裝置或軟體
all	所有的媒體
screen	一般的電腦螢幕
projection	投影機、電影放映機等
tv	電視、電視遊樂器等
handheld	行動裝置（PDA、行動電話等）
tty	以固定寬度字型顯示的裝置
print	印表機（含列印預覽等）
speech	聲音朗讀（聲音瀏覽器等）
braille	點字顯示器等
embossed	點字印表機

媒體類型

2-4 選擇器

決定樣式套用對象的是「選擇器」。確實了解選擇器的種類及特徵是有效製作CSS的第一步。

選擇器的種類

下列是CSS可利用的選擇器中，使用頻率比較高的種類。

萬用選擇器Universal selector（*）

一旦指定星號（*）為選擇器，樣式便會套用於所有的元素。也統稱為選擇器。

類型選擇器Type selector（E）

使用元素名的選擇器。把樣式套用於所有與該名稱相同的元素中，例如指定「body」或「p」。

E
以「E」表示元素的選擇器，「E」是「Element」的意思。

後代選擇器Descendant selectors（E F）

這個選擇器是把樣式套用在元素中的子元素。以【空白】鍵區隔連接選擇器。例如，指定「div p」時，樣式便會套用在「div」所包含的子元素「p」中。也可以用「div p em span」方式，篩選縮小樣式的套用對象。

名稱	選擇器	說明	範例
萬用選擇器	*	所有的元素	* { color: #000000; }
類型選擇器	E	該元素（E）	P { color: #222222; }
後代選擇器	E F	父元素（E）所包含的子元素（F）	p em { color: #cc0000; }
子選擇器	E > F	父元素（E）所包含的直接子元素（F）	p > em { color: #cc0000; }
相鄰選擇器	E + F	兄元素（E）之後的弟元素（F）	h1 + p { color: #222222; }
CLASS選擇器	.classname	設定class屬性值會成為「classname」的元素	.note { color: #ff0000; }
ID選擇器	#idname	設定id屬性值會成為「idname」的元素	#header { color: 006600; }
屬性選擇器	E[attr]	元素（E）有該屬性（attr）時	input[type] { padding: 2px; }
	E[attr="value"]	元素（E）有該屬性（attr）和值（value）時	input[type="text"] { padding: 2px; }
	E[attr~="value"]	元素（E）有該屬性，可利用空白字元 區隔指定多項屬性值，包含該屬性值（value）時	p[class~="moveto"] { color: #000066; }
	E[attr|="value"]	元素（E）有該屬性，可利用連字號區隔指定多項屬性值，包含與前方一致的該屬性值（value）時	div[lang|="en"] { font-size: small; }

選擇器一覽

:first-letter虛擬元素（:first-letter）

只把樣式套用於元素的第1個文字就是「:first-letter」。例如，只把段落（p元素）的第1個文字放大，使其變得醒目，讓後續的內文緊密貼合時，就是利用這種元素（此視覺效果又可稱為首字放大（drop cap）或字首大寫（initial cap））。

:before/:after虛擬元素（:before／:after）

為了在元素的前後產生某些內容，就要利用「:before」、「:after」，兩種都是利用content屬性指定產生內容。例如，在「p.note:before { content:"注意！"; }」樣式中，被指定為「class="note"」的p元素的前面，便會產生「注意！」的內文。

名稱	選擇器	說明	範例
類別虛擬類別	:link	未瀏覽的連結	a:link { color: #000080; }
	: visited	已瀏覽的連結	a:visited { color: #800080; }
動態虛擬類別	:hover	當滑鼠移動至連結時	a:hover { color: #ff0000; }
	:focus	當元素成為聚焦狀態時	a:focus { color: #ff0000; }
	:active	元素為作用中的狀態時	a:active { color: #ff0000; }
言語虛擬類別	:lang()	元素被指定為該語言編碼時	div:lang(en) { font-size: small; }
:first-child虛擬類別	:first-child	元素中的第一個子元素	div:first-child {font-style: italic; }
:first-line虛擬元素	:first-line	只有元素的第1行	h2:first-letter {font-size: large; }
:before/:after虛擬元素	:before	在元素的前面產生內容	#main:before { content: "內文從這裡開始"; }
	:after	在元素的後面產生內容	#main:after { content: "內文到這裡結束"; }

虛擬類別、虛擬元素一覽

選擇器的組合

選擇器可以組合利用。例如，指定「#header h1」時，樣式便能夠套用在「id="header"」中，作為後代元素所包含的h1元素。以顯示這種結構的ID及類別為基礎，利用後代選擇器縮小樣式的套用對象，經常被套用在實際的製作上。

```
#header {
  height: 66px;
  background: #33cccc url(../images/bcg_header.jpg);
}
#header h1 {
  padding-top: 17px;
  padding-left: 12px;
  color: #ffffff;
}
#header h1 a {
  color: #0000ff;
}
...
```

以顯示內容範圍的ID名稱及類別為基礎，利用後代選擇器套用樣式

以下是使用了類別及虛擬類別，稍微複雜的後代選擇器範例。

```
#main .highlight ul li a:link {
  color: #000080;
}
#main .highlight ul li a:visited {
  color: #800080;
}
...
```

稍微複雜的後代選擇器的範例

範例中的寫法混合了ID選擇器及CLASS選擇器、類型選擇器、虛擬類別，只要確實了解XHTML的樹狀結構（元素的繼承關係），就可以如同上例，把樣式只套用於最小限度的ID及類別之中。

雖然ID及類別經常能輕而易舉地增加，但增加得越多，也只會讓管理變得更加困難而已。不要完全仰賴ID及類別，使用後代選擇器縮小樣式的套用對象，是有效製作CSS所不可欠缺的方法。

另外，在顯示結構的ID及類別中，不加上類型選擇器（元素名）的情況也很多（表示不是指定「div#header」，而是「#header」）。因為連接後代選擇器時，「div」若是不斷地出現，會使程式碼變得不易閱讀。

謹慎使用後代選擇器

在後代選擇器中，請盡可能地仔細撰寫樹狀結構（代表不是指定「#header img」，而是指定「#header h1 a img」）。因為單是看到選擇器，就可以掌握XHTML的結構，閱讀上較為方便。也就是說，「把後代選擇器作為子選擇器是比較好的」（子選擇器本身在IE 6等瀏覽器中是沒有支援的，所以通常都不會使用）。

選擇器的群組化

以逗號區隔選擇器，進行群組化，就能夠同時套用相同的樣式。例如指定「h1, h2, h3」時，相同的樣式便會分別套用在h1元素、h2元素、h3元素當中。最後的選擇器請不要加上逗號。

```
h1, h2, h3 {
  color: #ff6600;
}
```

利用逗號區隔選擇器，進行群組化

利用後代選擇器把較長的選擇器群組化時，只要在每個逗號後面換行，就會比較容易閱讀了。

```
#utility-nav ul li a:link,
#global-nav ul li a:link,
#local-nav ul li a:link {
  color: #0000cc;
}
#utility-nav ul li a:visited,
#global-nav ul li a:visited,
#local-nav ul li a:visited {
  color: #cc0000;
}
```

把較長的選擇器群組化時，只要在每個逗號後換行，就會比較容易閱讀

2-5 屬性

屬性就是樣式的種類，也就是要賦予何種效果，然後，把效果的具體內容及程度作為「值」來進行指定。下列將針對使用頻率較高的屬性做說明。

熟悉了屬性之後…

雖然主要的屬性及值在使用順手的同時便會記得，但其實，只要在一開始就試著把它記下來就好了。

實際上，對於細微解釋及判斷感到迷惘，而逐一確認CCS規格（p.046）的情況並不少。但是，與其仔細地調查，不如預先了解原則的例外，以此做參考的情況很多，所以先掌握基本的觀念及結構才是最重要的。

方塊模型的屬性

XHTML的元素各自有自己的領域。領域不是圓形也不是三角形，而是「四方形」，這個四方形的領域稱為「方塊」（box）。而方塊是由下列四種領域所構成。

- 內容（content）：包含元素的內文或圖像等內容。
- 留白（padding）：內容與外框之間的空白。
- 外框（border）：框線。
- 邊界（margin）：外框與其他元素之間的間隔。

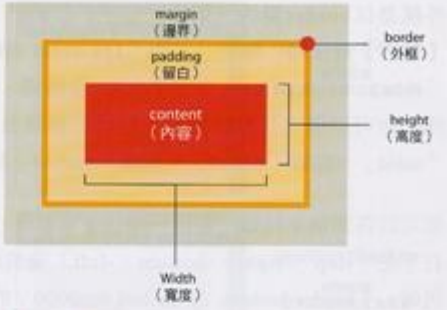
這樣的結構稱為「方塊模型（box model）」，在決定CSS頁面外觀上，這是非常重要的觀念。與方塊模型有關的是下列屬性。

屬性		效果	可指定的值
邊界	margin	邊界的統一指定	長度、%、auto
	margin-top, margin-right margin-bottom, margin-left	上下左右邊界	長度、%、auto
外框	border	外框的統一指定	粗細、形狀、以【空白】鍵區隔顏色
	border-width	外框的粗細	長度、thin、medium、thick
	border-style	外框的形狀	solid、dotted、dashed、none等
	border-color	外框的顏色	#rrggbb、#rgb、顏色名稱、transparent等
	border-top, border-right border-bottom, border-left	上下左右外框	粗細、形狀、以【空白】鍵區隔顏色
留白	padding	留白的統一指定	長度、%
	padding-top,padding-right padding-bottom,padding-left	上下左右的留白	長度、%
寬度及高度	width	內容的寬度	長度、%、auto
	height	內容的高度	長度、%、auto
寬度及高度的最大/最小	max-width, max-height	內容的最大寬度及高度	長度、%、none
	min-width, min-height	內容的最小寬度及高度	長度、%

margin／padding屬性

邊界及留白是利用margin屬性及padding屬性進行指定。兩種都會根據值的數量而套用在上下左右的地方，而產生下列的變化。一邊在腦中仔細整理，一邊進行指定吧！

- 1個值：「上下左右」
- 2個值：「上下」、「左右」
- 3個值：「上」、「左右」、「下」
- 4個值：「上」、「右」、「下」、「左」



方塊模型

例如，利用「margin: 5px 10px 8px」來指定3個值，所以邊界的上緣是5 px、左右是10 px、下緣則是8 px。「padding: 1em 2em 1.5em 1.8em」指定4個值，所以留白的結果就是上1 em、右2 em、下1.5 em、左1.8 em。

元素之間的邊界相抵

元素之間的邊界有上下會產生（套用值較大的一方）相抵，但左右並不會相抵的規定（p.272）。

邊界、留白也可分割成上下左右（-top、-right、-bottom、-left）4邊，分別設定各別的值。就像「margin-top: 5px」、「padding-left: 10px」這樣的指定。

預設值	可套用的元素
0	所有的元素（表格內部元素除外）
0	所有的元素（表格內部元素除外）
參照各屬性	所有的元素
medium	所有的元素
none	所有的元素
color屬性的值	所有的元素
參照各屬性	所有的元素
0	所有的元素（表格內部元素除外。th、td元素可）
0	所有的元素（表格內部元素除外。th、td元素可）
auto	所有的元素（非置換行內元素、表格的欄元素和欄群組化元素除外）
auto	所有的元素（非置換行內元素、表格的列元素和列群組化元素除外）
none	參照width/height屬性
0	參照width/height屬性

border屬性

外框是以border屬性來指定。指定的值有「粗細」、「形狀」、以【空白】鍵區隔「顏色」（如「1px solid #ff6600」）。粗細除了有「thin」、「medium」、「thick」這種細、中、粗三種層級的值之外，也可以指定任意的粗細。形狀的種類很多，最常使用的是實線「solid」、點線「dotted」、虛線「dashed」及雙實線「double」。

也可以各別指定粗細、形狀、顏色（-width、-style、-color），將上右下左（-top、-right、-bottom、-left）分割成4邊，設定各自的值。例如，「border-bottom: 2p dashed #cc0000」的指定。

width／height屬性

內容的寬度是以width屬性指定，高度則是以height屬性指定。這終究只是內容的寬度和高度而已，並不包含邊界、外框、留白，這一點請多加注意。

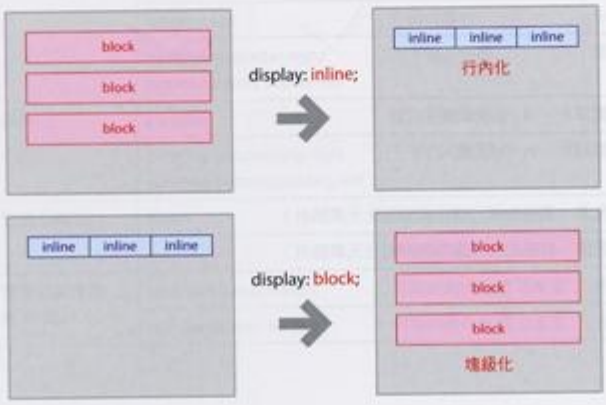
視覺格式模型的屬性

決定如何配置、顯示產生各自元素的方塊的結構就稱為「視覺格式模型（visual formatting model）」，可利用下列的屬性。

display屬性

指定產生的方塊種類，可指定「block」、「inline」、「list-item」、「none」等值。大部分應用是在塊級元素（block）產生的方塊（例如li元素）中指定「inline」以進行行內化，或是在行內元素（inline）產生的方塊（例如a元素）中指定「block」，進行塊級化。

使用display屬性的地方



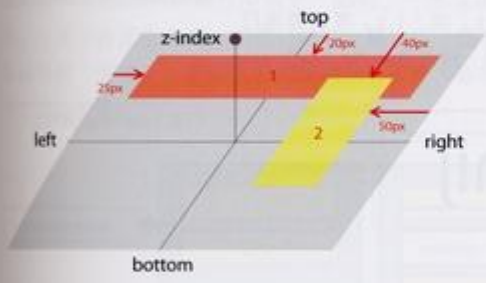
position屬性

指定方塊的配置方法。可以指定代表相對配置、絕對配置、固定配置的「relative」、「absolute」、「fixed」及歸回原位（一般配置）的「static」。可以利用top、right、bottom、left屬性決定更詳細的位置，甚至還可以利用z-index屬性指定方塊的重疊順序。

利用position屬性指定「relative」時，是以元素原本的位置作為配置的基準，「absolute」則是以父方塊的位置為基準（「fixed」則是針對滾動捲軸做出固定配置），這點請多加注意。

另外，作為「absolute」的基準位置的父方塊，必須利用position屬性指定「relative」等值，決定出配置方法才行。如果沒有決定配置方法，基準位置便會追溯至祖元素（ancestor element）。如果還是找不到可決定配置方法的元素時，最後會以body元素作為基準位置，這點請多加注意。

配置方法（position屬性）



定位及重疊順序

float屬性

使方塊左右移動的屬性稱為「浮動」。float屬性中可指定「left」、「right」等值。這是將圖像配置在左或右方，可使內文緊密貼合在周圍的使用方法。

浮動的解除（清除）可利用clear屬性，以「left」、「right」、「both」的解除方向作為指定的值。

浮動（float屬性）

在實際應用上，這些屬性經常被利用在欄框排版（multi-column layout）。就是讓主要部分浮動於左方，側欄部分浮動於右方，然後清除後續內容的浮動之使用方法。


```
#main{
 width: 70%;
 float: left;/*浮動在左邊*/
}

#side{
 width: 30%;
 float: right;/*浮動在右邊*/
}

#footer{
 clear: both;/*清除浮動*/
}
```

把浮動套用在欄框排版中

另外，方塊中只有浮動元素時，方塊的高度會自動縮放。因此，在方塊中指定背景色時，會讓人有高度略顯不足的印象，這點請多加注意（解決方法請參照p.274）。

line-height屬性

指定行高。可利用「%」、「em」、「pt」等各種單位，但「%」和「em」會有繼承（繼承來自於祖元素的計算值）的問題；而「pt」再加上繼承，與字體大小之間會有平衡（字體放大時，內文會產生重疊）問題。所以，最常使用的方法是利用「1」或「1.5」等實數進行指定（關於利用line-height屬性指定值的問題，詳細請參考p.285）。

祖元素

統稱包含某元素的元素，稱為「祖元素」。

行高（line-height屬性）

vertical-align屬性

指定行內的本文或圖像的上下位置，或者是表格儲存格內容的上下位置。可以指定「baseline」、「top」、「middle」、「bottom」等值。

vertical-align屬性的代表性值

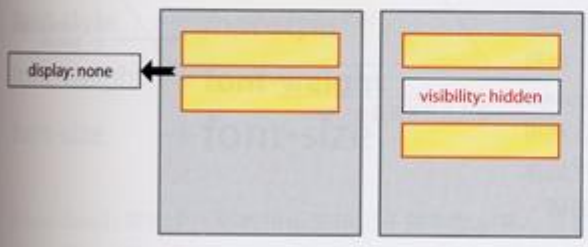
屬性	效果	可指定的值	預設值	可套用的元素
display	方塊的種類	inline、block、list-item、none等	inline	所有的元素
position	方塊的配置方法	static、relative、absolute、fixed	static	所有的元素
top right bottom left	指定方塊的上下左右的位置	長度、%、auto	auto	決定配置的元素（position屬性的值為「static」之外的元素）
z-index	方塊的重疊順序	整數、auto	auto	決定配置的元素（position屬性的值為「static」之外的元素）
float	方塊的浮動	left、right、none	none	所有的元素
clear	清除浮動	left、right、both、none	none	塊級元素
line-height	行高	normal、實數、長度、%	normal	所有的元素
vertical-align	行內及儲存格內的上下位置	baseline、top、middle、bottom等	baseline	行內元素、th、td元素

視覺格式模型的屬性

視覺效果的屬性

visibility屬性

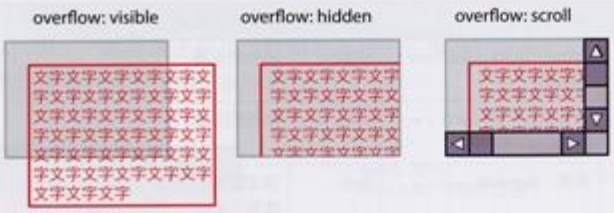
指定方塊的顯示、隱藏狀態。「visible」是顯示、「hidden」則是隱藏的值。若在display屬性中指定「none」，所呈現的結果是「不顯示」，並不會產生方塊。若在visibility屬性中，指定「hidden」時，其結果是「隱藏」，會產生方塊範圍但看不見內容的狀態。請注意兩者之間的差異。



「display: none」和「visibility: hidden」的差異

overflow屬性

指定方塊的裁切方法（內容超出方塊時的顯示方法）。可以指定「visible」、「hidden」、「scroll」等值。若由瀏覽器判斷則指定值為「auto」。



方塊的裁切方法（overflow屬性）

屬性	效果	可指定的值	預設值	可套用的元素
visibility	方塊的顯示、隱藏	visible、hidden、collapse	visible	所有的元素
overflow	方塊的裁切方法	visible、hidden、scroll、auto	visible	塊級元素、th、td元素等

視覺效果的屬性

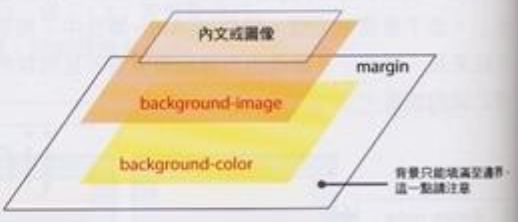
文字顏色、背景的屬性

color屬性

以RGB值或顏色名稱指定文字顏色。如果文字顏色沒有與稍後提到的「背景」呈現對比，內文將會變得不容易閱讀，這點請多加注意（p.279）。

背景屬性

指定背景的屬性有很多種。指定背景色是background-color屬性，背景圖像則是background-image屬性。甚至還有可以指定背景圖像重複排列方法的background-repeat屬性、指定圖像開始位置的background-position屬性、針對滾動捲軸移動、固定圖像的background-attachment屬性。另外，也可以利用縮寫的方式，一次指定這些屬性值在background上，並以【空白】鍵區隔、順序不同的方式指定值。



背景色和背景圖像的關係

```
body {
  background-color: #fff;
  background-image: url(bg.gif);
  background-repeat: repeat-x;
  background-position: 0px -10px;
}
```

```
body {
  background: #fff url(bg.gif) repeat-x 0px -10px;
}
```

縮寫的範例

在此需注意，背景可以填滿至外框領域，但無法填滿至邊界領域。另外，也請先了解背景色在下、背景圖像在上的結構。在沒有顯示背景圖像的環境（瀏覽器關閉讀取圖像功能等）和背景圖像中有透明部分時，皆會顯示出底下的背景色。

屬性	效果	可指定的值	預設值	可套用的元素
color	文字顏色	#rrggbb、#rgb、顏色名稱等	視瀏覽器而定	所有的元素
background-color	背景色	#rrggbb、#rgb、顏色名稱、transparent等	transparent	所有的元素
background-image	背景圖像	url(…)、none	none	所有的元素
background-repeat	背景圖像的重複排列方法	repeat、repeat-X、repeat-y、no-repeat	repeat	所有的元素
background-attachment	背景圖像的移動、固定	scroll、fixed	scroll	所有的元素
background-position	背景圖像的開始位置	left、right、center、top、bottom、%、長度	0% 0%	所有的元素
background	背景各屬性的一次指定	以【空白】鍵區隔各屬性的值	參照各屬性	所有的元素

文字顏色、背景的屬性

字型、段落的屬性

文字的屬性

字型的屬性有指定字型種類的font-family、指定斜體或正常字體等的font-style、指定粗細的font-weight、指定字體大小的font-size等。

font-family → font-family
font-style → *font-style*
font-weight → **font-weight**
font-size → font-size

字型的屬性

在font-family屬性中，可利用逗點指定多個字型名稱，並建議在最後指定統稱字體（generic-family）的「serif」、「sans-serif」、「cursive」、「fantasy」、「monospace」的任一字型。

一般來說，指定宋體時是在最後指定「serif」，指定黑體時則是在最後指定「sans-serif」。

統稱字體

即代表概略的意思，CSS規格中有定義serif（襯線體，如宋體）、san-serif（非襯線體，如黑體）、cursive（如手稿體，草書體）、fantasy（此為裝飾字型）、monospace（此為等寬字型）五種統稱字體。

屬性	效果	可指定的值	預設值	可套用的元素
font-family	字型的種類	以逗號區隔，指定多個字型名稱	視瀏覽器而定	所有的元素
font-style	字型的樣式	normal、italic、oblique	normal	所有的元素
font-weight	字型的粗細	normal、bold等	normal	所有的元素
font-size	字型的大小	長度、%、medium等關鍵字	medium	所有的元素
font	字型屬性的一次指定	以【空白】鍵區隔各屬性的值	參照各屬性	所有的元素
text-indent	段落開頭的縮排寬度	長度、%	0	塊級元素、th、td元素等
text-align	段落的對齊	left、right、center、justify	視文字標記方向而定	塊級元素、th、td元素等
text-decoration	段落的裝飾	underline、line-through、none等	none	所有的元素
text-transform	段落的大寫、小寫的轉換	capitalize、uppercase、lowercase、none	none	所有的元素
letter-spacing	段落的文字/字母間隔	normal、長度	normal	所有的元素
word-spacing	段落的單字間隔	normal、長度	normal	所有的元素

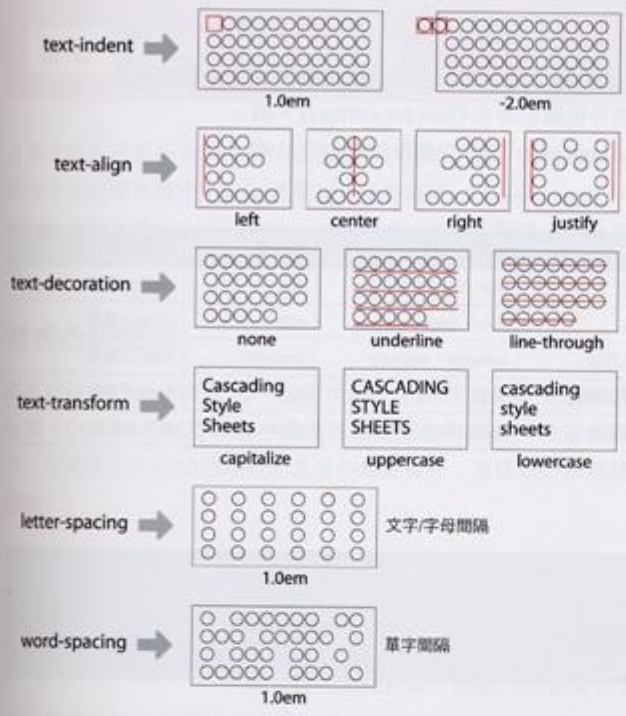
字型、段落的屬性

在font-size屬性中，除了「1.2 em」、「120 %」等長度或百分比之外，也可以指定「xx-small」、「x-small」、「small」、「medium」、「large」、「x-large」及「xx-large」等七個層級的關鍵字。

另外，雖然可以利用font的縮寫方式，一次指定字型的相關屬性和line-height屬性，但是，必要值和任意值的區別、指定順序會變得複雜、省略的值也會變為預設值，哪個值是哪個屬性的值也無法一眼就判斷出來，如此一來，閱讀性及使用性都會變得比較差。因此，不使用font屬性，而逐一指定個別的屬性是比較普遍的做法（參考 p.284）。

段落的屬性

段落的屬性有指定段落開頭（第1行的前面）的縮排寬度的text-indent、指定對齊的text-align、指定底線裝飾的text-decoration、指定大寫文字和小寫文字變換的text-transform、指定字母或文字間隔的letter-spacing，以及指定單字間隔的word-spacing等。



段落的屬性

字型及段落

套用樣式時，對於判斷屬性是否為「字型」相關或「段落」相關，應該會產生疑惑吧。字型所代表的是單一的文字（字體），段落指的則是整個區塊的多數文字，這就是不同的地方。只要以「字」和「文」的差異去思考就行了。

清單的屬性

清單的屬性有指定顯示在清單前面的符號種類list-style-type、指定符號圖像的list-style-image、指定符號位置的list-style-position。另外，也可以利用統一指定這些屬性的縮寫性質list-style。

屬性	效果	可指定的值	預設值	可套用的元素
list-style-type	清單符號的種類	disc、circle、square、decimal等	disc	在display屬性中被指定為「list-item」的元素（li元素）
list-style-image	清單符號的圖像	url(…)、none	none	在display屬性中被指定為「list-item」的元素（li元素）
list-style-position	清單符號的位置	inside、outside	outside	在display屬性中被指定為「list-item」的元素（li元素）
list-style	關於清單符號的一次指定	以【空白】鍵區隔各屬性的值	參照各屬性	在display屬性中被指定為「list-item」的元素（li元素）

清單的屬性

表格的屬性

表格的屬性有指定表格（table）標題位置的caption-side、指定表格排版方法的table-layout、指定表格外框顯示方法的border-collapse、指定表格外框內外緣間隔的border-spacing、指定空儲存格顯示方法的empty-cells。

屬性	效果	可指定的值	預設值	可套用的元素
caption-side	表格的標題位置	top、bottom	top	caption元素
table-layout	表格的排版方法	auto、fixed	auto	table元素等
border-collapse	表格外框的顯示方法	collapse、separate	separate	table元素等
border-spacing	表格外框內外緣的間隔	長度	0	table元素等
empty-cells	空儲存格的顯示方法	show、hide	show	th、td元素

表格的屬性

其他的屬性

使用者介面的屬性

使用者介面的屬性cursor是指定滑鼠游標的種類，outline則是指定鎖定時的輪廓樣式。

content屬性

指定產生內容的屬性。和:before/:after虛擬元素一起使用（參照p.058）。

屬性	效果	可指定的值	預設值	可套用的元素
cursor	滑鼠游標	auto、pointer、wait等	auto	所有的元素
outline-width	輪廓的粗細	長、thin、medium、thick	medium	所有的元素
outline-style	輪廓的形狀	solid、dotted、dashed、none等	none	所有的元素
outline-color	輪廓的顏色	invert、#rrggbb、#rgb、顏色名稱等	invert	所有的元素
outline	輪廓的統一指定	粗細、形狀、以【空白】鍵區隔顏色	參照各屬性	所有的元素
content	產生內容	none、normal、文字列、url(…)等	normal	:before/:after虛擬元素

其他的屬性

2-6 樣式的優先順序

當樣式的套用對象有衝突時，可依照「詳細度」及「讀取順序」兩種規則來決定優先順序。此結構稱為「層疊處理」（cascading），這也是CSS這個名稱的由來。另外，也可體會CSS中的特徵－「繼承」結構。

詳細度

也稱為「特殊性」、「特異性」、「個別性」。英文為「specificity」。

樣式的詳細度

詳細度就是「個別樣式比一般樣式優先」的規則。萬用選擇器不受影響，CLASS選擇器和屬性選擇器則比類型選擇器（元素名稱）優先，而更進一步的ID選擇器則比這些都更優先（選擇器的種類請參照 2-4 ）。

樣式的詳細度

```
*             { color: black; }     /*詳細度「0, 0, 0, 0」*/
P             { color: blue; }      /*詳細度「0, 0, 0, 1」*/
p em          { color: red; }       /*詳細度「0, 0, 0, 2」*/
p.note        { color: orange; }    /*詳細度「0, 0, 1, 1」*/
div#aside p   { color: gray; }      /*詳細度「0, 1, 0, 2」*/
```

萬用選擇器所代表的是「所有元素」，沒有特定的套用對象。類型選擇器是XHTML中可以利用的元素名稱，比萬用選擇器更具有個別性。CLASS選擇器和屬性選擇器則是以製作者自行指定的屬性及屬性值作為套用樣式的基礎，因此，比單純的元素名稱更具有個別性。再更進一步，ID選擇器只能在頁面內指定單一id屬性值，個別性的存在相當明顯，因此，比CLASS選擇器和屬性選擇器都還要優先。大致上來說，「越詳細的選擇器就會越優先套用」，務必記住這一點。

樣式詳細度的印象

你或許會覺得複雜，但是當「指定的樣式無法順利套用」的問題發生時，大部分的原因都是來自於「詳細度」。所以發生問題時，請先朝「詳細度」方面思考（試想是不是已經有其他樣式的詳細度比剛才寫的樣式還來得仔細）是很重要的。

另外，在XHTML中指定「style屬性」也是能獲得認同的。不從CSS的程式碼中套用樣式時，也可以直接指定作為屬性值的樣式。例如，「<body style=" color: black; background-color: #ffffff; " >」。

style屬性的詳細度是「1, 0, 0, 0」，會比ID（0,1, 0, 0）、類別和屬性（0, 0,1,0）、元素名稱（0,0,0,1）更優先套用。但是，正如 2-3 中所說的，style屬性一旦使用太多，便無法統一管理整個網站的CSS，所以還是請抱持著「不使用為原則」的觀念。

樣式的讀取順序

讀取順序的規則是「當詳細度相同時，會以之後讀取到的樣式為優先」。例如，「p { color: blue; }」和「p { color: green; }」之類的樣式，寫在後面的樣式會優先套用。

這個規則也會套用在利用link元素和@import讀取CSS的時候。在程式碼中，寫在後面的（之後讀取）檔案會優先。只要把「覆寫」這樣的觀念套用在這種規則上就行了。

CSS的程式碼中

```
p { color: blue; }
...
p { color: green; } /* 優先 */
```

利用link元素讀取CSS檔

```
<link rel="stylesheet" type="text/css" href="css/aaa.css" /> <!-- 優先順序1 -->
<link rel="stylesheet" type="text/css" href="css/bbb.css" /> <!-- 優先順序2 -->
<link rel="stylesheet" type="text/css" href="css/ccc.css" /> <!-- 優先順序1 -->
```

利用@import讀取CSS檔

```
@import "css/aaa.css"; /* 優先順序3 */
@import "css/bbb.css"; /* 優先順序2 */
@import "css/ccc.css"; /* 優先順序1 */
```

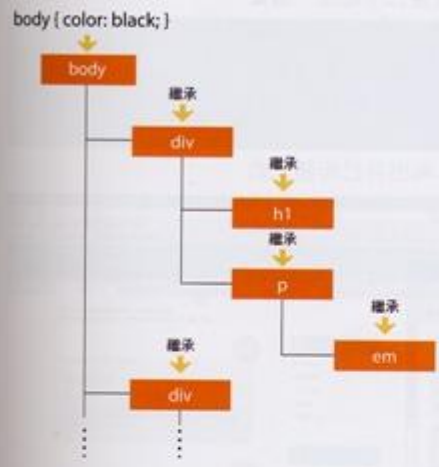
樣式讀取的順序

樣式的繼承

CSS有個特徵就是，套用於某元素的樣式會自動被後代元素所承襲。這種結構稱為「繼承」，大部分的樣式都會自動承襲。

「明示性」的繼承

把屬性的值指定為「inherit」，就可以明確地指出要把不做繼承的屬性繼承下來（在大部分的屬性中都可以指定）。



樣式的繼承