這段程式碼會建立一個包含四個方格的漸層圖像，然後調整該圖像的大小並重複顯示。使用它不會比使用重複的線性漸層更有效率或更優雅，但它有吸引人的巧思。

重複的圓錐漸層

接著來討論如何重複顯示錐形漸層，這非常有用，很適合用來建立星爆（starburst）圖案，甚至是像棋盤圖案之類的簡單圖像。例如：

```
conic-gradient(
    #0002 0 25%, #FFF2 0 50%, #0002 0 75%, #FFF2 0 100%
    )
```

這條規則用四個顏色停駐點，和僅僅兩種顏色來設置一個棋盤圖案。我們可以使用 `repeating-conic-gradient` 來重新定義它，並使用新顏色來讓圖案更加清晰：

```
repeating-conic-gradient(
     #343 0 25%, #ABC 0 50%
     )
```

對這個簡單的重複案例而言，我們只要設定前兩個顏色停駐點就夠了，之後，這些停駐點會重複顯示，直到錐形漸層的整個 360 度都被填滿，如圖 9-46 所示。

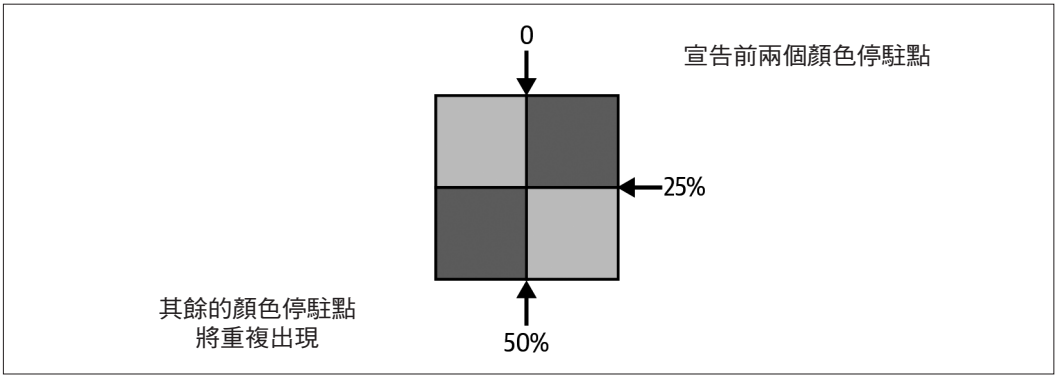

圖 9-46　重複顯示的錐形漸層

這意味著我們可以建立任何大小的楔形區域、任何顏色變化效果，並讓它們在整個錐狀圓周上重複。舉三個例子，如圖 9-47 所示：

```
repeating-conic-gradient(#117 5deg, #ABE 15deg, #117 25deg)
repeating-conic-gradient(#117 0 5deg, #ABE 0 15deg, #117 0 25deg)
repeating-conic-gradient(#117 5deg, #ABE 15deg)
```

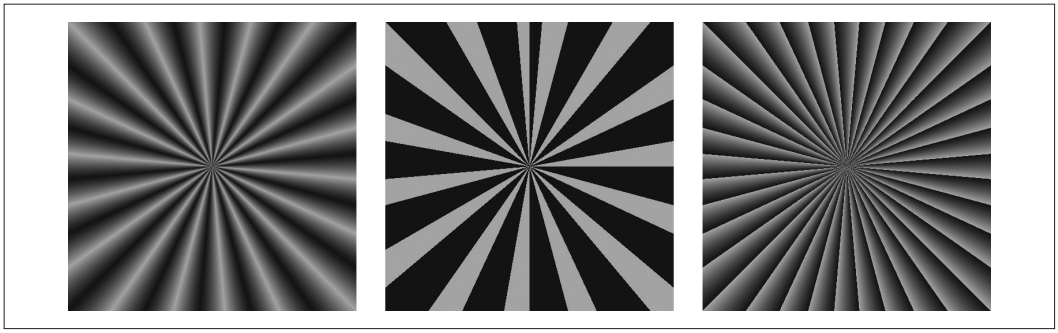

圖 9-47　重複性錐形漸層的三個版本

注意第一個（最左邊）例子中的平順變化，即使在圖像的上方也是如此：從 350 度的 #117 到 5 度的 #ABE 之間的變化與所有其他變化相同。就此而言，重複性錐形漸層很特別，因為線性和放射狀漸層都不會「繞回來」讓終點與起點相遇。這個情況也可以在圖 9-47 的第三個（最右邊）例子中看到。

但這種特殊行為可以打破，從第二個例子（中間的）可以看到：注意從 355 度到 360 度的較窄的楔形區域。之所以如此，是因為第一個顏色停駐點明確地指定從 0 度到 5 度，因此漸層無法從 355 度變化到 5 度，導致在 360/0 度有一個硬轉變。

操作漸層圖像

就像我們之前所強調的（或許強調太多次了），漸層是圖像。這意味著你可以使用各種背景屬性來調整它們的大小、位置、重複…等，就像你對任何 PNG 或 SVG 檔案做的那樣。

其中一種可以利用的策略是重複顯示簡單的漸層（以更複雜的方式來重複是下一節的主題）。例如，你可以使用「硬停駐的放射狀漸層」在背景上產生點狀外觀，如圖 9-48 所示：

```
body {background: radial-gradient(circle at center,
                    rgba(0 0 0 / 0.1), rgba(0 0 0 / 0.1) 10px,
                    transparent 10px, transparent)
                    center / 25px 25px repeat,
                    tan;}
```

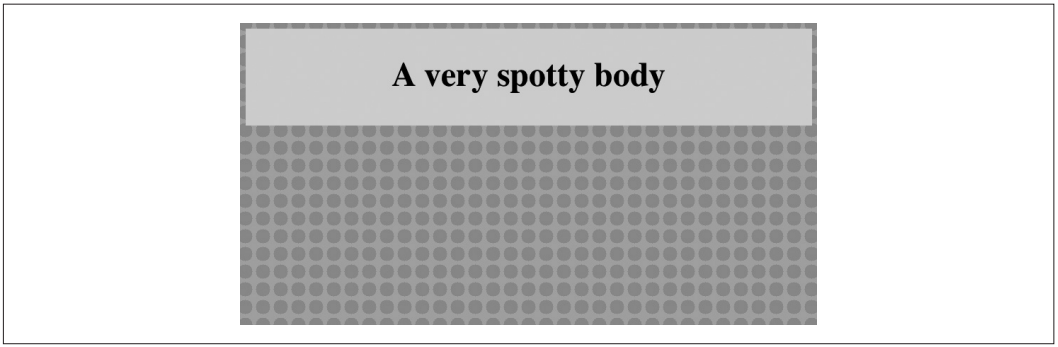

圖 9-48　平鋪放射狀漸層圖像

沒錯，在視覺上，這與平鋪一個大部分是透明的、直徑為 10 像素的深色圓圈的 PNG 幾乎相同，對這個例子而言，使用漸層有三個優點：

- CSS 的 bytes 大小幾乎一定比 PNG 的更小。
- 更重要的是，使用 PNG 需要對伺服器發出額外的請求，這既降低網頁速度，也降低伺服器的性能。CSS 漸層是樣式表的一部分，因此可以免除傳給伺服器的額外請求。
- 改變漸層要簡單得多，因此進行試驗來找出確切的大小、形狀和明暗度要容易得多。

創造特效

漸層無法完成點陣或向量圖像可以做的所有事情，因此即使有了漸層，你也不會完全放棄外部圖像。但你仍然可以利用漸層來實現一些令人印象深刻的效果。考慮圖 9-49 中的背景效果。

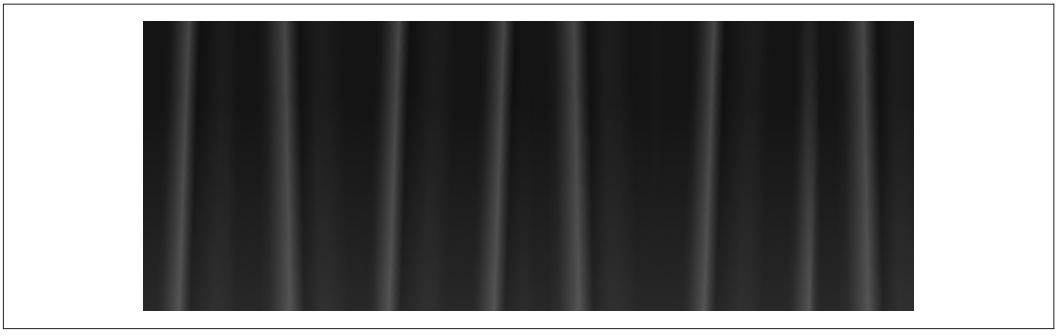

圖 9-49　該下音樂了…

要做出這個布幕效果，你只需要使用兩個線性漸層，讓它們在不同的間隔處重複，再加上第三個漸層，在背景底部產生一個「發光」效果。以下是做出這個效果的程式碼：

```
background-image:
    linear-gradient(0deg, rgba(255 128 128 / 0.25), transparent 75%),
    linear-gradient(89deg,
        transparent 30%,
        #510A0E 35% 40%, #61100F 43%, #B93F3A 50%,
        #4B0408 55%, #6A0F18 60%, #651015 65%,
        #510A0E 70% 75%, rgba(255 128 128 / 0) 80%, transparent),
    linear-gradient(92deg,
        #510A0E 20%, #61100F 25%, #B93F3A 40%, #4B0408 50%,
        #6A0F18 70%, #651015 80%, #510A0E 90%);
background-size: auto, 300px 100%, 109px 100%;
background-repeat: repeat-x;
```

第一個（因此是最上面的）漸層從 75% 透明度的淺紅色變化成漸層線的 75% 處的完全透明。然後我們建立兩個「摺疊」圖像。圖 9-50 分別展示它們。

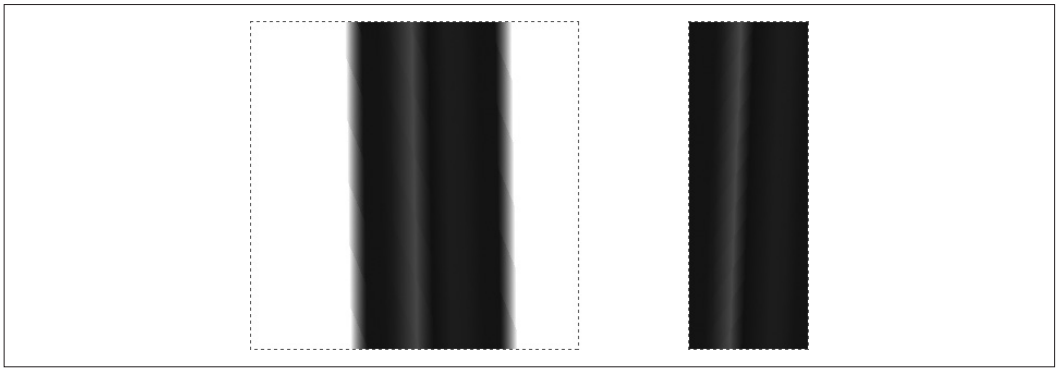

圖 9-50　兩個「摺疊」漸層

定義了這些圖像後，我們沿著 x 軸重複它們，並指定不同的大小。第一個圖像是「發光」效果，它被設為 auto 大小，讓它覆蓋整個元素背景。第二個被設為寬度 300px，高度 100%；因此，它將與元素背景一樣高，寬 300 像素。這意味著它將在 x 軸上每 300 像素平鋪一次。第三個圖像也是如此，不同之處在於它每 109 像素平鋪一次。最終結果看起來像一個不規則的舞臺布幕。

這樣做的好處在於，你只要編輯樣式表就可以調整平鋪間隔了。改變顏色停駐點位置或顏色比較複雜一點，但如果你知道你想要的效果是什麼，這也不會太難。而且加入第三組重複的摺疊很簡單，只要加入另一個漸層即可。

使用 flex-direction 屬性

如果你想讓布局從上到下、從左到右、從右到左，甚至從下到上，你可以使用 `flex-direction` 來控制 flex 項目將沿著哪個主軸排列。

flex-direction	
值	`row` | `row-reverse` | `column` | `column-reverse`
初始值	`row`
適用於	flex 容器
計算值	按指定
可否繼承	否
可否動畫化	否

`flex-direction` 屬性指定如何在 flex 容器中放置 flex 項目。它定義了 flex 容器的主軸，這個軸是排列 flex 項目的主要軸（詳情見第 494 頁的「瞭解軸」）。

假設有以下的基本標記結構：

```
<ol>
   <li>1</li>
   <li>2</li>
   <li>3</li>
   <li>4</li>
   <li>5</li>
</ol>
```

圖 11-5 展示如何使用 `flex-direction` 的四個值來排列這個簡單的列表，假設語言是從左到右。

乍看之下，預設值 `row` 和許多行內或浮動元素沒有太大差異。這其實是假象，理由很快就會展示，但注意其他的 `flex-direction` 值如何影響列表項目的排列。

例如，你可以使用 `flex-direction: row-reverse` 來將這個項目的布局反過來。設定 `flex-direction: column` 時，flex 項目是從上到下排列的，設定 `flex-direction: column-reverse` 時，則從下到上，如圖 11-5 所示。

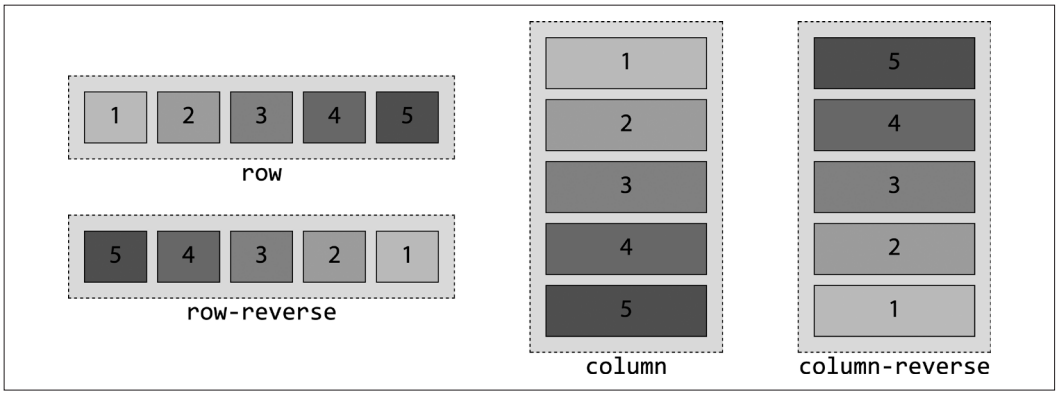

圖 11-5　flex-direction 屬性的四個值 ▶

我們指定了從左到右的語言，因為 `row` 的主軸方向（flex 項目的排列方向）是當下的書寫模式的方向。稍後將討論書寫模式如何影響 flex 方向和布局。

不要使用 `flex-direction` 來改變從右到左的語言的布局，請改用 HTML 的 `dir` 屬性，或使用第 818 頁的「設定書寫模式」中介紹的 CSS 屬性 `writing-mode` 來指定語言方向。若要瞭解關於語言方向和 flexbox 的更多資訊，請參考第 489 頁的「配合其他書寫方向」。

`column` 值將 flex 容器的主軸設成與當下書寫模式的區塊軸相同的方向。在像英語這樣的水平書寫模式中，這是垂直軸；而在像傳統日語這樣的垂直書寫模式中，這是水平軸。

因此，當你在英語（或具有相同書寫方向的語言）中宣告 `column` 方向時，flex 項目會按照原始文件中的宣告順序來顯示，但是是從上到下，而不是從左到右，所以 flex 項目是一個疊在另一個上面，而不是並排。考慮以下範例：

```
nav {
  display: flex;
  flex-direction: column;
  border-right: 1px solid #ccc;
}
```

因此，只要寫幾個 CSS 屬性，我們就可以為之前的連結列表建立一個漂亮的側邊欄風格導覽列，並將它們做成水平的標籤列。在新布局中，我們將 `flex-direction` 從預設值 `row` 改為 `column`，並將邊框從底部移至右側。圖 11-6 是顯示出來的結果。

圖 11-6　改變 flex 方向可以完全改變布局 ▶

`column-reverse` 值與 `column` 相似，只是主軸是反過來的，因此，`main-start` 會被放在主軸的結尾，`main-end` 會被放在主軸的起點。在從上到下的書寫模式中，這意味著 flex 項目是向上排列的，如圖 11-5 所示。`-reverse` 值僅改變外觀。鍵盤瀏覽 tab 順序與底層標記保持一致。

目前展示的功能非常強大，讓許多版面設計工作變得非常簡單。把這個導覽列放入完整的文件裡可以發現宣告幾個 flexbox 屬性來設計版面有多麼簡單。

我們來稍微擴展前面的 HTML 範例，將導覽列當成一個組件放入首頁：

```
<body>
  <header>
    <h1>My Page's title!</h1>
  </header>
  <nav>
    <a href="/">Home</a>
    <a href="/about">About</a>
    <a href="/blog">Blog</a>
    <a href="/jobs">Careers</a>
    <a href="/contact">Contact Us</a>
  </nav>
  <main>
     <article>
       <img alt="" src="img1.jpg">
       <p>This is some awesome content that is on the page.</p>
       <button>Go Somewhere</button>
     </article>
     <article>
       <img alt="" src="img2.jpg">
       <p>This is more content than the previous box, but less than
       the next.</p>
       <button>Click Me</button>
     </article>
```

```
      <article>
        <img alt="" src="img3.jpg">
        <p>We have lots of content here to show that content can grow, and
        everything can be the same size if you use flexbox.</p>
        <button>Do Something</button>
      </article>
    </main>
    <footer>Copyright &#169; 2023</footer>
  </body>
```

只要增加幾行 CSS 就可以產生一個排列整齊的首頁（圖 11-7）：

```
* {
  outline: 1px #ccc solid;
  margin: 10px;
  padding: 10px;
}
body, nav, main, article {
  display: flex;
}
body, article {
  flex-direction: column;
}
```

沒錯，元素可以同時是 flex 項目和 flex 容器，就像這個例子裡的 navigation、main 和 article 元素一樣。我們將 `<body>` 和 `<article>` 元素的 flex 方向設為 `column`，並讓 `<nav>` 和 `<main>` 使用預設的 `row`，做這些事情只需要兩行 CSS！

澄清一下，圖 11-7 還有其他樣式參與其中。我們為所有元素設定了邊框、邊距和內距，讓你在學習時，可以在視覺上區分 flex 項目（我們不想讓這個不怎麼吸引人的網站上線！）。除此之外，我們的工作只是將 body、navigation、main 和 articles 宣告為 flex 容器，使得 navigation 連結、main、article、圖片、段落和按鈕成為 flex 項目。

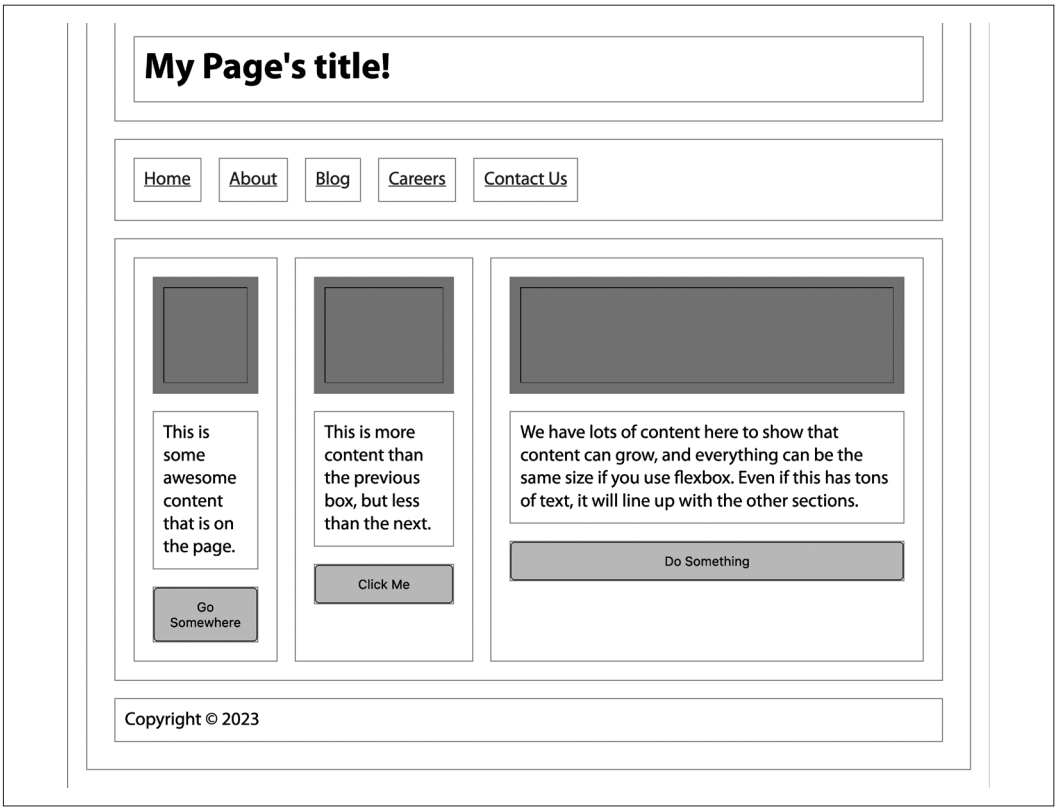

圖 11-7　使用 `flex-direction: row` 和 `column` 來製作的首頁版面 ▶

配合其他書寫方向

如果你正在使用英文或其他從左至右（LTR）的語言來建立網站，你可能希望 flex 項目從左到右、從上到下排列。使用預設值 `row` 可以做到這一點。如果你使用阿拉伯語或其他從右至左（RTL）的語言，你可能希望 flex 項目從右到左、從上到下排列。預設值 `row` 也可以做到。

使用 `flex-direction: row` 會將 flex 項目排成與文字方向（也稱為書寫模式）相同的方向，無論該語言是 RTL 還是 LTR。雖然大多數網站都使用從左至右的語言，但有些網站使用從右至左的語言，也有一些使用從上至下的。當你改變書寫模式時，flexbox 會自動為你改變 flex 方向。

書寫模式是用 writing-mode、direction 和 text-orientation 屬性來設定的，或是用 HTML 的 dir 屬性來設定（第 15 章會介紹它們）。當書寫模式是從右至左時，主軸的方向（因此也是 flex 容器內的 flex 項目的方向）在 flex-direction 為 row 時會從右至左排列。圖 11-8 解釋這個情況。

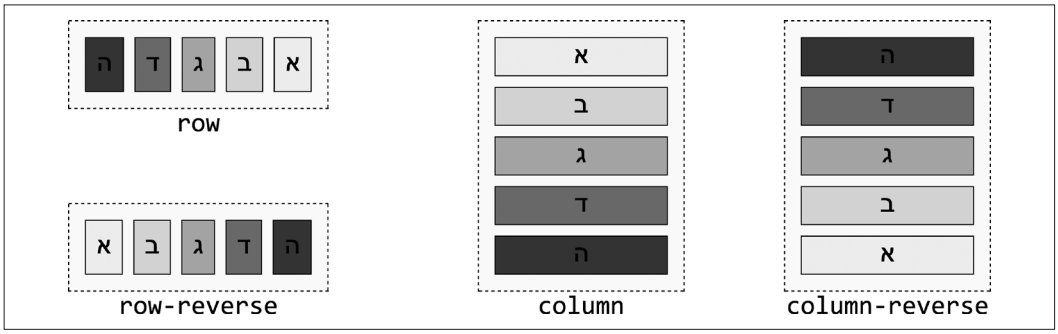

圖 11-8　當書寫方向是從右至左時，flex-direction 的四個值 ▶

如果 CSS 的 direction 值與元素的 dir 屬性值不同，CSS 屬性值優先於 HTML 屬性。規範強烈建議你使用 HTML 屬性而非 CSS 屬性。

垂直書寫的語言包括 Bopomofo（注音符號）、埃及象形文字、平假名、片假名、漢字、韓文、麥羅埃文草書體和象形文字、蒙古文、歐甘文、古突厥文、八思巴文、彝文，有時還有日文。指定垂直書寫模式時，這些語言才會垂直顯示，否則，這些語言都會被當成水平的語言。

對於由上至下的語言，writing-mode: horizontal-tb 是有效的，這意味著主軸會從預設的從左至右旋轉 90 度。因此，flex-direction: row 是從上到下，而 flex-direction: column 是從右到左。圖 11-9 展示了不同的 flex-direction 值如何顯示以下的標記：

```
<ol lang="jp">
    <li> 一 </li>
    <li> 二 </li>
    <li> 三 </li>
    <li> 四 </li>
    <li> 五 </li>
</ol>
```

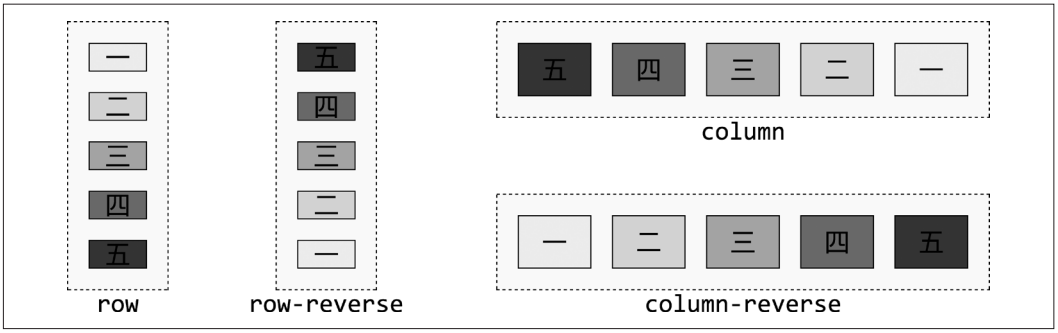

圖 11-9　當書寫模式是 horizontal-tb 時，`flex-direction` 的四個值 ▶

沒錯，row（列）是垂直的，column（行或欄）是水平的。不僅如此，基本的 `column` 方向是從右至左，而 `column-reverse` 則是從左至右。我們在此看到的，就是將這些值用於由上至下、由右至左的語言的結果。

好的，你已經看了 flex 方向和書寫模式如何互相影響。但是到目前為止，所有的例子都只展示了一列或一行的 flex 項目。如果 flex 項目的主維度（對於 `row`，是指它們行內尺寸總和，對於 `column`，是指區塊尺寸總和）無法放入 flex 容器該怎麼辦？我們可以讓它們溢出容器，或讓它們換行到額外的 flex 行。稍後，我們還會談論如何讓 flex 項目收縮（或擴大），以配合容器。

讓 Flex 行換行

如果你無法將所有的 flex 項目放入 flex 容器的主軸，那麼 flex 項目在預設情況下不會換行，也不一定會調整大小。你可以藉著使用 flex 項目的 `flex` 屬性來允許 flex 項目收縮（見第 532 頁的「增長因子與 flex 屬性」），否則，flex 項目會溢出容器外圍框。

你可以影響這種行為。`flex-wrap` 屬性設定 flex 容器究竟是僅限於單行，還是在需要時可變為多行。

flex-wrap

值	`nowrap` | `wrap` | `wrap-reverse`
初始值	`nowrap`
適用於	flex 容器
計算值	按指定

可否繼承	否
可否動畫化	否

當你將 `flex-wrap` 屬性設為 `wrap` 或 `wrap-reverse` 來允許 flex 項目分為多行時，它會決定額外的 flex 項目行要出現在原始的 flex 項目行之前還是之後。

圖 11-10 展示當 `flex-direction` 值為 `row`（語言為 LTR）時，`flex-wrap` 屬性的三個值的效果。這些範例展示兩個 flex 行，第二個和後續的 flex 行會沿著交叉軸（在這個情況下，是垂直軸）的方向加入。

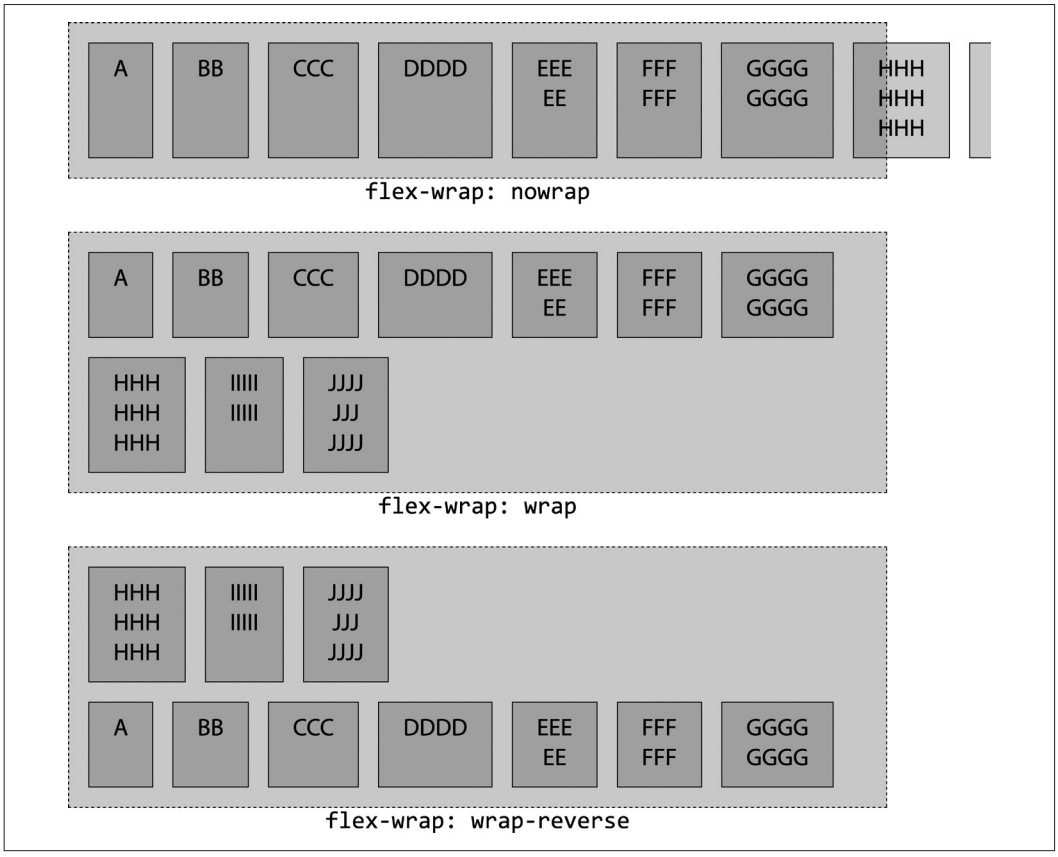

圖 11-10　在 row 方向的排列中，`flex-wrap` 屬性的三個值 ▶