CSS書寫順序



例子：



Visual Studio Code

## 插件

Open in Browser 右击选择在浏览器中打开HTML文件

JS-CSS-HTML Formatter 每次保存，都会自动格式化js css和HTML代码

Auto Rename Tag 自动重命名配对的HTML/XML标签

CSS Peek 追踪样式

## Emmet語法

Emmet語法的前身是Zen coding，它使用縮寫，來提高html/css的編寫速度。Vscode內部已集成該語法。

### （1）Emmet快速生成HTML結構語法

零 生成HTML框架：輸入!，按tab鍵

① 生成標籤：直接輸入標籤名，按tab鍵

② 生成多個相同標籤：輸入div\*3，按tab鍵

③ 父子級關係：輸入ul>li，按tab鍵

④ 兄弟關係：輸入div+p，按tab鍵

⑤ 生成帶有類名或者id名字的div：輸入.demo或者#demo，按tab鍵

⑥ 如果類名或者id名帶有序號：輸入.demo$\*5，按tab鍵，可以得到：

<div class="demo1"></div>

<div class="demo2"></div>

<div class="demo3"></div>

<div class="demo4"></div>

<div class="demo5"></div>

⑦標籤裡默認顯示文字：

輸入div{天朗氣清}\*5，按tab鍵

<div>天朗气清</div>

<div>天朗气清</div>

<div>天朗气清</div>

<div>天朗气清</div>

<div>天朗气清</div>

輸入div{$}\*5，按tab鍵

<div>1</div>

<div>2</div>

<div>3</div>

<div>4</div>

<div>5</div>

### （2）Emmet快速生成CSS結構語法

①寫單詞開頭字母。比如text-align：輸入ta，然後按tab鍵或者回車。

②寫寬：輸入w100，然後按tab鍵或者回車。寫高同理。

③寫縮進：輸入ti2e，然後按tab鍵或者回車。

### （3）格式化代碼

Visual Studio Code中，在代碼文件中單擊右鍵，選擇Format Document，可以自動格式化代碼。

在保存代碼文件時，自動格式化代碼的設置：

打開File→Preferences→Settings

搜索Emmet.include

必須選擇User下的項目

點擊 Edit in settings.json



添加：

“editor.formatOnType”: true,

“editor.formatOnSave”: true

HTML基礎知識

## HTML概念

HTML指的是超文本標記語言（Hyper Text Markup Language），它是用來描述網頁的一種語言。

HTML是標記語言，而非編程語言。

超文本，有2層含義：

1. 它可以加入圖片、聲音、動畫、多媒體等內容（超越了文本限制）。

2. 它還可以從一個文件跳轉到另一個文件，與世界各地主機的文件連接（超級鏈接文本）。

## 常用瀏覽器以及內核

常用瀏覽器：谷歌，IE，Edge，火狐，Safari，Opera

瀏覽器內核（渲染引擎）：負責讀取網頁內容，整理訊息，計算網頁的顯示方式並顯示頁面。

|  |  |  |
| --- | --- | --- |
| 瀏覽器 | 內核 | 備註 |
| IE | Trident | IE，獵豹安全，360急速瀏覽器，百度瀏覽器 |
| Firefox | Gecko | 火狐瀏覽器內核 |
| Safari | Webkit | 蘋果瀏覽器內核 |
| Chrome，Opera | Blink | Chrome/ Opera瀏覽器內核，Blink其實是WebKit的分支 |

國產瀏覽器一般採用Webkit/ Blink內核，如360，UC，QQ，搜狗等。

## Web標準

Web標準是由W3C組織和其它標準化組織制定的一系列標準的集合。

W3C（萬維網聯盟）是國際著名的標準化組織。

Web標準的必要性：

瀏覽器不同，同樣的代碼，在不同的瀏覽器上有不同的顯示。遵循Web標準，可以讓開發代碼更規範，這樣在不同的瀏覽器裡，用戶看到的效果基本一致。

Web標準構成：

① 結構 structure 對網頁元素進行整理和分類 HTML

② 表現 presentation 設置網頁元素的板式、顏色、大小等外觀樣式 CSS

③ 行為 behaviour 網頁模型的定義及交互的編寫 JavaScript

## VSCode工具生成骨架標籤新增代碼

1

<!DOCTYPE> 文檔類型聲明，告訴瀏覽器使用哪種HTML版本來顯示網頁。

<!DOCTYPE html> 這段代碼意思：當前頁面採用的是HTML5版本來顯示網頁。

<!DOCTYPE>位置：位於文檔中最前面的位置，處於<html>標籤之前。

<!DOCTYPE>不是HTML標籤，是文檔類型聲明標籤。

2

<html lang=”en”>

定義文檔語言，英文是en，中文是”zh-CN”，法語只”fr”。其實不論定義那種語言，文檔中都能寫中英雙語。

這個屬性對瀏覽器和搜索引擎有作用。比如Chrome讀到這裡，會提示，是否需要翻譯。

3

<meta charset=”UTF-8”>必須寫，否則亂碼。

字符集（Character set）是多個字符的集合，以便計算機能夠識別和存儲各種文字。

在<head>標籤內，通過<meta>標籤的charset屬性來規定HTML文檔應該使用哪種字符編碼。

charset常用的值有：GB2312、BIG5、GBK和UTF-8，其中UTF-8被稱為萬國碼，基本包含了全世界所有國家需要用到的字符。

## 加粗，傾斜，刪除，下劃線

|  |  |  |
| --- | --- | --- |
| 語義 | 標籤 | 推薦使用下列標籤，因為語義更強烈 |
| 加粗 | <strong></strong>或者<b></b> | strong |
| 傾斜 | <em></em>或者<i></i> | em |
| 刪除線 | <del></del>或者<s></s> | del |
| 下劃線 | <ins></ins>或者<u></u> | ins |

## img標籤

屬性：title 屬性值：文本 說明：提示文本，鼠標放到圖像上，顯示的文字。

## 合併單元格

跨行：最上側單元格為目標單元格，寫合併代碼

跨列：最左側單元格為目標單元格，寫合併代碼

合併單元格3步：

1. 確定跨行還是跨列合併

2. 找到目標單元格，寫上合併方式=合併的單元格數量。比如：<td colspan= “2”></td>

3. 刪除多餘的單元格

## 自定義列表

商務網站首頁底部的佈局，一般用自定義列表

基本語法如下：

<dl>

<dt>名詞1</dt>

<dd>名詞1解釋1</dd>

<dd>名詞1解釋2</dd>

<dd>名詞1解釋3</dd>

</dl>

## 表單標籤



表單域=form標籤

表單控件=表單元素

## <label>標籤

<label>標籤為input元素定義標註（標籤）

<label>標籤用於綁定一個表單元素，當點擊<label>標籤內的文本時，瀏覽器會自動將焦點（光標）轉到或者選擇對應的表單元素上，用來增加用戶體驗。

語法：

<label for=”gender”>女</label>

<input type=”radio” name=”gender” id=”gender” />

其中lable的for屬性和input的id屬性值要一樣

## HTML元素顯示模式

HTML元素顯示模式：塊元素，行內元素，行內塊元素。

### （1）塊元素

<h1>到<h6>，<p>，<div>，<ul>，<ol>，<li>等

特點：

① 自己佔一行。

② 可改變高、寬、外邊距、內邊距。

③ 默認寬度是容器（父級寬度）的100%。

④ 是一個容器/盒子，裡面可以放塊元素、行內元素或者行內塊元素。

注意：

文字類的元素（<h1>到<h6>，<p>）內不能使用塊元素。

### （2）行內元素

行內元素=內聯元素

<a>，<span>，<strong>，<b>，<em>，<i>，<del>，<s>，<ins>，<u>

特點：

① 相鄰行內元素在一行上，一行可以顯示多個。

② 無法直接設置寬、高。

③ 默認寬度是內容的寬度。

④ 行內元素只能容納文本或其它行內元素。

注意：

① 鏈接裡面不能再放鏈接。

② 鏈接<a>裡面可以放塊級元素，但是最好把<a>轉換成塊元素。

#### em製作小豎線

<style>

      em {

        font-style: normal;

        color: #ebe4e0;

        margin: 0 6px 0 15px; /\* 竖线和左右盒子的距离 \*/

      }

    </style>

  </head>

  <body>

    <em>|</em>

  </body>

### （3）行內塊元素

<img />，<input />，<td>同時具有塊元素和行內元素的特點。

特點：

① 一行可以顯示多個行內塊元素，但元素之間有空隙。

② 默認寬度是本身內容的寬度。

③ 可以設置高、寬、行高、外邊距、內邊距。

### （4）元素顯示模式轉換

比如要增加<a>的觸發範圍。把<a>從行內元素變成塊元素。

變成塊元素方式：

display: block;

變成行內元素方式：

display: inline;

變成行內塊元素方式：

display: inline-block;

HTML5

HTML5增加了新標籤，新表單，新表單屬性。

這些新特性都有兼容性問題，基本是IT9+以上的版本的瀏覽器才支持，如果不考慮兼容性問題，可以大量使用這些新特性。

## HTML5新增語義化標籤

<header>：頭部標籤

<nav>：導航標籤

<article>：內容標籤

<section>：定義文檔某個區域（大號的div）

<aside>：側邊欄標籤

<footer>：尾部標籤

語義化標籤針對搜索引擎

這些新標籤可以在頁面中多次使用

在IE9中，需要把這些元素轉換為塊級元素，因為在IE9裡面，它們默認狀態下，不是塊級元素。

移動端更常用這些標籤

## HTML5新增多媒體標籤

### （1）視頻：<video>

使用它們可以很方便地在網頁中嵌入音頻和視頻，而不再去使用flash和其它瀏覽器插件。

當前<video>元素支持三種視頻格式，盡量使用mp4格式。



語法

<video src=”文件地址” controls=”controls”></video>

為了兼容性更好，可以這樣寫（了解即可，現在瀏覽器兼容性都比較好了）：

<video width=”320” height=”240” controls>

<source src=”movie.mp4” type=”video/mp4”>

<source src=”movie.ogg” type=”video/ogg”>

您的瀏覽器不支持video標籤

</video>

設置video標籤的值時，最好寫雙標籤，比如autoplay=”autoplay” muted=”muted” loop=”loop”



### （2）音頻：<audio>

當前<audio>元素支持三種音頻格式

所有主流瀏覽器都支持MP3

語法：

<audio src=”文件地址” controls=”controls”></audio>

如果一定要考慮兼容性的問題：

< audio controls="controls" >

<source src="happy.mp3" type="audio/mpeg" >

<source src="happy.ogg" type="audio/ogg" >

您的浏览器暂不支持 <audio> 标签。

</ audio>



谷歌瀏覽器把音頻和視頻自動播放都禁止了，用JavaScript解決禁止音頻自動播放問題，用muted解決禁止視頻自動播放問題。

## HTML5新增input表單



重點記住：number，tel，search這三個



上面的表單樣式由下列代碼寫成

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

  </head>

  <body>

    <!-- 驗證的時候必須添加form表單域 -->

    <form action="">

      <ul>

        <li>郵箱：<input type="email" /></li>

        <li>網址：<input type="url" /></li>

        <li>日期：<input type="date" /></li>

        <li>時間：<input type="time" /></li>

        <li>數量：<input type="number" /></li>

        <li>手機號碼：<input type="tel" /></li>

        <li>搜索：<input type="search" /></li>

        <li>顏色：<input type="color" /></li>

        <!-- 點擊提交按鈕可以驗證表單 -->

        <li><input type="submit" value="提交" /></li>

      </ul>

    </form>

  </body>

</html>

## HTML5新增表單屬性

|  |  |  |
| --- | --- | --- |
| **屬性** | **值** | **說明** |
| required | required | 表單擁有該屬性表示其內容不能為空，必填 |
| placeholder | 提示文本 | 表單的提示信息，存在默認值將不顯示 |
| autofocus | autofocus | 自動聚焦屬性，頁面加載完成自動聚焦到指定表單 |
| autocomplete | off / on | 當用戶在字段開始鍵入時，瀏覽器基於之前鍵入過的值，應該顯示出在字段中填寫的選項。  默認已經打開，如autocomplete=“on”。成功提交過才會有autocomplete的內容。關閉是autocomplete=“off”，一般情況下，為了安全，會關閉。需要放在表單內，同時加上name屬性。 |
| multiple | multiple | 可以多選文件提交 |

可以通過以下設置方式修改placeholder裡面的字體顏色：

input::placeholder {

color: pink;

}

CSS

## CSS代碼風格

1. 樣式格式書寫

使用展開式的格式：

h3 {

color: pink;

font-size: 20px;

}

2. 樣式大小寫

寫樣式時，用小寫字母。

大寫字母也能執行，但是不規範。

3. 空格

① 屬性值前面和冒號的後面，寫1個空格。比如上面的pink和20前面都有1個空格。

② 選擇器（標籤）和大括號中間保留空格。比如上面h3和大括號之間有空格。

## CSS選擇器

CSS選擇器分為基礎選擇器和復合選擇器兩大類。

### 1. CSS基礎選擇器

基礎選擇器由單個選擇器組成

基礎選擇器包括：標籤選擇器，類選擇器，id選擇器，通配符選擇器

#### （1）標籤選擇器（=元素選擇器）

指用HTML標籤名稱作為選擇器，為頁面中某一類標籤指定統一的CSS樣式。

比如把所有的<div>字體變成紅色，把所有的<span>字體變成藍色。

例子：

h3 {

color: pink;

font-size: 20px;

}

#### （2）類選擇器

單獨選擇一個或者某幾個標籤，可以使用類選擇器。

語法

.類名 {

屬性1: 屬性值1;

…

}

類選擇器口訣：樣式點定義，結構類調用；一個或多個，開發最常用。

例子：

定義

.tea {

color: red;

}

調用

<div class=’tea’>變紅色</div>

不要用標籤名字作為類名。

一個標籤可以調用多個類。兩個類名之間用空格分開。

例子<div class=’red size’>變紅色</div>

#### （3）ID選擇器

語法

#id名{

屬性1: 屬性值1;

…

}

例子

定義

#tea {

color: red;

}

調用

<div id=”tea”>茶葉的顏色</div>

id選擇器只能被調用1次；類選擇器可以被多次調用。

類選擇器在修改樣式中用得最多，id選擇器一般用於頁面唯一性的元素上，經常和JavaScript搭配使用。

#### （4）通配符選擇器

通配符選擇器用“\*”定義，它表示選取頁面中的所有元素（標籤）。定義後系統會自動調用。

語法

\* {

屬性1: 屬性值1;

…

}

### 2. CSS復合選擇器

把基礎選擇器組合在一起，就形成了復合選擇器。

常用的復合選擇器包括：後代選擇器，子選擇器，並集選擇器，偽類選擇器等。

#### （1）後代選擇器（常用）

後代選擇器=包含選擇器

例子

    <style>

      ol li {

        font-weight: bold;

      }

    </style>

  </head>

  <body>

    <ol>

      <li>天朗气清</li>

      <li>天朗气清</li>

      <li>天朗气清</li>

    </ol>

    <ul>

      <li>惠风和畅</li>

      <li>惠风和畅</li>

      <li>惠风和畅</li>

    </ul>

元素1和元素2中間用空格隔開。

元素1 是父級，元素2是後代，最終選擇的是元素2。

元素2可以是子級，也可以是孫級，只要是元素1 的後代即可。

元素1和元素2可以是任意基礎選擇器。

#### （2）子選擇器

子元素選擇器（子選擇器）只能選擇作為某元素的最近一級子元素=子級元素

語法：

元素1>元素2（樣式聲明）

例子：

    <style>

      .nav > a {

        font-weight: bold;

      }

    </style>

  </head>

  <body>

    <div class="nav">

      <a href="">天朗气清</a>

      <p><a href="">惠风和畅</a></p>

    </div>

  </body>

#### （3）並集選擇器（常用）

並集選擇器可以選擇多組標籤，同時為它們定義相同的樣式。通常用於集體聲明。

各個選擇器通過英文逗號連接而成，任何形式的基礎選擇器都可以作為並集選擇器的一部分。

語法：

元素1, 元素2 {樣式聲明}

例如：

ul, div {

font-size: 20px;

}

語法規範：並集選擇器豎著寫。最後一個選擇器後面不用加逗號。

<style>

      div,

      p,

      .tea>li {

        font-size: 20px;

      }

    </style>

#### （4）偽類選擇器

偽類選擇器作用：①給鏈接添加特殊效果，②選擇第1個、第n個元素。

書寫特點：用冒號表示。例子 :hover :first-child。

##### （4-①）連接偽類選擇器（常用）

語法：

a:link 選擇所有未被訪問的連接

a:visited 選擇所有已被訪問的連接

a:hover 選擇鼠標指針位於其上的連接 （最常用）

a:active 選擇活動連接（鼠標按下未彈起的連接）

為了確保生效，按照LVHA的順序聲明：:link :visited :hover :active

例子：

a {

color: gray;

}

a:hover {

color: red;

}

##### （4-②）focus偽類選擇器

:focus 用於選取獲得焦點的表單元素，主要針對<input>

例子：

input:focus {

background-color:yellow;

}

## CSS的font

### （1）font-family

如果字體名字由多個單詞構成，要給字體名字加引號。單引號和雙引號都可以。

如果寫了多個字體，每個字體之間用逗號+空格隔開。瀏覽器會根據列出的字體順序逐個去檢查，優先使用已經安裝了的、排在前面的字體。如果所有字體都沒安裝，就使用瀏覽器默認的字體。Chrome默認的是微軟雅黑

例子：

p {

font-family: ‘Times New Roman’, Times, serif, Arial;

}

字體名字可以用中文，用引號包裹，但是用英文會更規範。

h2 {

font-family: ‘微軟雅黑’;

}

微軟雅黑的英文：

h2 {

font-family: ‘Microsoft yahei’;

}

在開發裡，把字體定義進body標籤

### （2）font-size

有單位px，一定要寫上。

Chrome默認字體大小為16px

不同瀏覽器默認顯示的字號大小不一致，盡量給一個明確大小值，不要默認。

設置了

body {

font-size: 16px;

}

標題的字號不會跟著變，需要另外設置標題的字號大小，比如

h2 {

font-size: 16px;

}

### （3）font-weight

font-weight: 700; （沒有單位）等於 font-weight: bold

標題文字默認是加粗的，如果想讓標題文字變細：

h2 {

font-weight: 400;

}

或者

h2 {

font-weight: normal;

}

在實際開發中，數字用得更多。

### （4）font-style

font-style的值有2個：normal和italic

平時很少給文字加斜體，一般是把斜體標籤（em，i）改為不傾斜字體。

例子：

em {

font-style: normal;

}

### （5）字體復合屬性

可以把上述的font屬性寫在一行。

語法：

body {

font: font-style font-weight font-size/ line-height font-family;

}

要按照上面的順序來寫，不能隨意顛倒順序。

各個屬性間用空格隔開。

不需要設置的屬性可以省略，但必須保留font-size和font-family屬性。否則font不起作用。

例子1：

div {

font: italic 700 16px ‘Microsoft yahei’;

}

例子2：

body {

font: 16px/28px ‘Microsoft yahei’

}

上述代碼意思：所有字體設為16px，行高28px，使用微軟雅黑字體。

## CSS修飾文本

文本屬性定義文本的外觀，比如文本的顏色、對齊、裝飾（刪除線、下劃線）、文本縮進、行間距等。

### （1）color

color的屬性值，可以用顏色的英文單詞（red，green等）、16進制值（#FF0000，#29D795等）、RGB代碼（ rgb(255,0,0)或者rgb(100%,0%,0%) ）。用得最多的是16進制值。

16進制的值，如果兩兩相同，可以只寫3個值

比如#666666可以寫為#666；#ff00ff可以寫為#f0f

### （2）text-align

設置元素內文本內容的水平對齊方式

例子：讓h1盒子裡的文字水平居中對齊（h1自己占1行的）

h1 {

text-align: center;

}

text-align 的值：left（默認左對齊），right，center

### （3）text-decoration

添加下劃線

div {

text-decoration: underline;

}

刪除下劃線

a {

text-decoration: none;

}

### （4）text-indent

em是個相對單位，相當於當前元素（font-size）1個文字的大小。

比如當前字體大小為16px，那麼1em=16px

如果當前元素沒有設置大小，則會按照父元素的1個文字大小。

如果中文文章段首要縮進2個漢字，就可以不用px而用em這個單位。

比如：

p {

text-indent: 2em;

}

### （5）line-height

完整的行間距由3部分構成：上間距+文本高度+下間距



比如給font-size: 16px的div設定line-height: 16px; 那麼上間距和下間距都是0。

如果給font-size: 16px的div設定line-height: 26px; 那麼上間距和下間距都是5。

在PS切圖測量行高時，從第一行的下沿測量到第二行的下沿，就是行高了。

### （6）單行文字垂直居中

設置方式：文字的行高=盒子的高度

原理







## CSS 的clientHeight，offsetHeight，offsetTop，scrollHeight、scrollTop

網頁可見區域高：document.body.clientHeight

網頁可見區域高（包括邊線的高）：document.body.offsetHeight

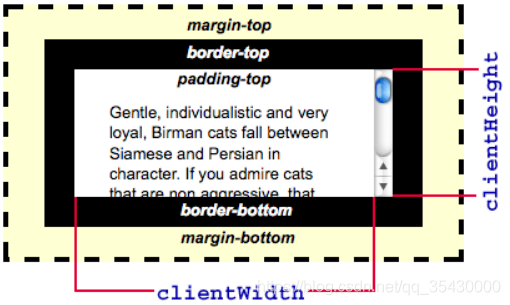
網頁正文全文高：document.body.scrollHeight

網頁被卷去的高：document.body.scrollTop

屏幕分辨率高：window.screen.height

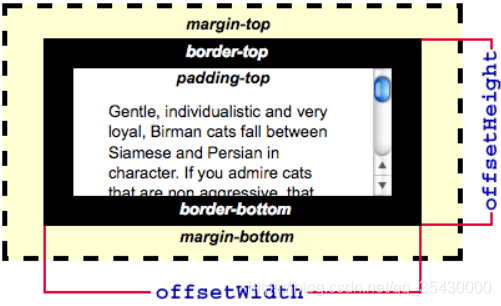
### （1）clientHeight

* clientHeight：包括padding但不包括border、水平滾動條、margin的元素的高度。對於inline的元素這個屬性一直是0，單位px，只讀元素。



### （2）offsetHeight

* offsetHeight：包括padding、border、水平滾動條，但不包括margin的元素的高度。對於inline的元素這個屬性一直是0，單位px，只讀元素。



### （3）offsetTop

offsetTop: 當前元素border外到最近的有定位的父元素（position: relative;或者absolute或者fixed）border內且都不包括border的距離，如果父元素有padding，则此padding也算在内。和有沒有滾動條沒有關系。即便有滾動，父子元素的offsetTop都不變。單位px，只讀元素。



### （4）scrollHeight

scrollHeight，單位px，只讀元素。

當本元素的子元素比本元素高且overflow=scroll或者auto時，本元素不想被子元素撐的一樣高就顯示出了滾動條，在滾動的過程中子元素有部分被隱藏了。

可見部分的高度其實就是clientHeight

scrollHeight>=clientHeight恒成立。

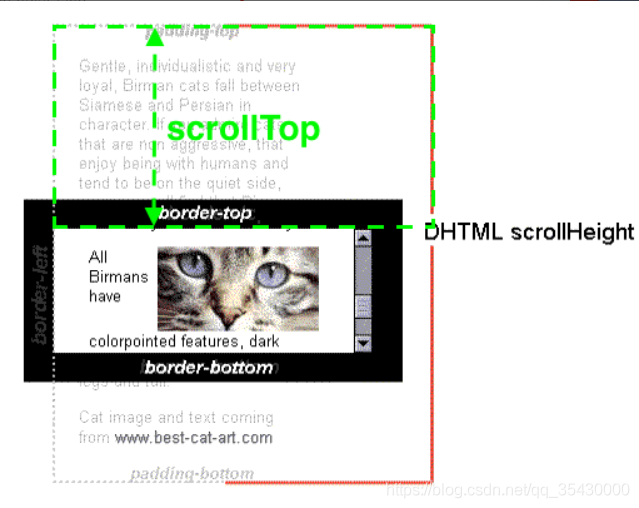
scrollHeight=子元素的height+padding+border+margin（上下）+本元素的padding（上下）



### （5）scrollTop

scrollTop: 代表在有滾動條時，滾動條向下滾動的距離也就是子元素元素頂部被遮住部分的高度。在沒有滾動條時scrollTop==0恒成立。單位px，可讀可設置。

* 注意，scrollTop是相對於本盒子的（比如下面這個黑框）。子盒子（下面這個字和畫的頁面）的scrollTop永遠是0。
* 要scrollTop生效，必須給本盒子設置height。或者給本盒子設置position: fixed; 後，設置top和bottom的值。



scrollTop最大的時候，是滾動條到底的時候。這時scrollTop=scrollHeight – clientHeight

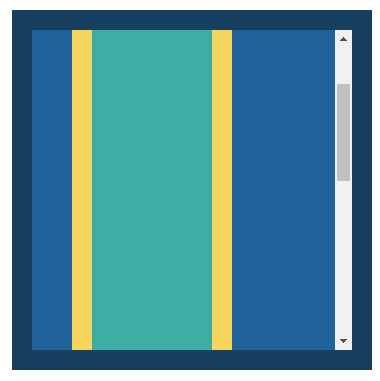
### （6）例子

寫這樣2個盒子：

父盒子.div1 藍背景深藍邊框，高300px

子盒子.div2 綠背景黃色邊框，高800px

重點注意一下scrollTop的值。scrollTop是相對於父盒子.div1的。子盒子.div2的scrollTop永遠是0。



    <style>

      .box1 {

        margin: 100px auto;

        width: 300px;

        height: 300px;

        padding: 10px;

        background-color: #20639b;

        overflow-y: auto;

        position: relative;

        border: 20px solid #173f5f;

      }

      .box2 {

        width: 100px;

        height: 800px;

        background-color: #3caea3;

        padding: 10px;

        border: 20px solid #f6d55c;

        margin: 30px;

      }

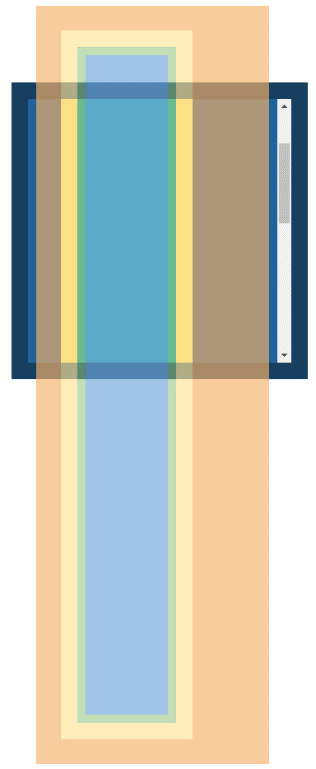
    </style>

    <div class="box1">

      <div class="box2"></div>

    </div>

如果完全顯示div2，是這樣的：



scrollTop最大的時候，是滾動條到底的時候。這時scrollTop=scrollHeight – clientHeight

## CSS引入方式

### （1）行內樣式表（行內式）

在元素標籤內部的style屬性中設定CSS樣式。

用得非常少，適用於簡單修改樣式。

書寫方便，權重高。

例子：

<div style=”color: red; font-size: 12px;”>碧螺春</div>

style是標籤的屬性。

要用雙引號。

### （2）內部樣式表（嵌入式）

寫在HTML頁面內部，把所有的CSS代碼抽取出來，單獨放到<style>標籤中。

理論上CSS代碼可以放在HTML頁面的任何位置，但是一般放在<head>部分<title>的後面。

初級練習的時候使用。

### （3）外部樣式表（鏈接式）

把樣式寫在CSS文件中，然後在HTML文件中引入。

在開發中用得最多的。

在HTML頁面中引入CSS文件的語法：

<link rel=”stylesheet” href=”css文件路徑”>

放在<head>部分<title>的後面

## CSS背景

### （1）背景顏色

默認值：

background-color: transparent;

取值可以是顏色單詞，16進制值，RGB代碼。

### （2）背景圖片

背景圖片比插入圖片更容易控制位置。

頁面元素可以同時添加背景圖片和背景顏色。背景圖片在背景顏色上層。

運用場景：logo，裝飾性小圖片，超大背景圖片，精靈圖

background-image: none | url(url)

none為默認值

例子：

background-image: url(images/logo.png);

### （3）背景平鋪

background-repeat: repeat | no-repeat | repeat-x | repeat-y

上述值的意思：平鋪，不平鋪，沿x軸平鋪，沿y軸平鋪。

默認值是repeat

### （4）背景圖片位置（重要）

background-position: x y;

參數代表的意思是：x坐標和y坐標。可以使用方位名詞，也可以使用精確單位。

#### ①參數是方位名詞：

top，center，bottom，left，center，right

①-a）如果指定的2個值都是方位名詞，兩個值前後順序無關。

比如這2個是一樣的：

background-position: left top;

background-position: top left;

比如這2個也是一樣的（都是水平靠右，垂直居中）：

background-position: right center;

background-position: center right;

①-b）如果只指定了一個方位名詞，另一個值省略，則第二個值默認center。

#### ②參數是精確單位：px，%

第1個參數必定是x坐標，第2個參數必定是y坐標。

例子：

background-position: 50px 20px;

如果只寫1個參數，那個參數必定是x軸的。

比如：

background-position: 50px; 這是x軸上的50px，另外一個值默認是垂直方向的center了。

#### ③參數混合使用

第1個值一定是x坐標，第2個值一定是y坐標。

例子：

background-position: 20px center; x軸是20px，y軸是center。

background-position: center 20px; x軸是center ，y軸是20px。

### （5）固定背景圖片

background-attachment屬性設置背景圖像是否固定，或者隨著頁面的其餘部分滾動。

background-attachment應用場景：製作視差滾動的效果。

語法：

background-attachment: scroll | fixed

scroll是默認值

### （6）背景屬性復合寫法（常用）

沒有特定的書寫順序，一般習慣約定順序為：

background: 背景顏色 背景圖片地址 背景圖片是否平鋪 背景圖片是否滾動 背景圖片位置;

各個值用空格隔開。

### （7）背景顏色半透明

background: rgba(0, 0, 0, 0.3);

a是alpha的簡稱，取值範圍在0-1之間。

3個0是黑色。上面這個例子是黑色的半透明。

上面0.3裡面的0可以省略，寫成：background: rgba(0, 0, 0, .3);

這只是讓盒子的背景色半透明，盒子的內容不受影響。

這是CSS3新增屬性，是IE9+版本瀏覽器才支持的。但是現在實際開發，不太關注兼容性寫法了，可以放心使用。

## CSS三大特性

层叠性，继承性，优先级

### （1）层叠性

相同選擇器給設置相同的樣式，一個樣式就會覆蓋（層疊）另一個衝突的樣式。

①樣式衝突，遵循的原則是就近原則：哪個樣式離結構近，就執行哪個樣式。

比如這個例子裡，最終執行pink顏色

<style>

      div {

        color: blue;

      }

      div {

        color: pink;

      }

    </style>

  </head>

  <body>

    <div></div>

  </body>

② 樣式不衝突，就不會覆蓋

下面的例子中，只有pink會覆蓋blue，height不會被覆蓋

<style>

      div {

        color: blue;

        height: 30px;

      }

      div {

        color: pink;

      }

    </style>

  </head>

  <body>

    <div></div>

  </body>

### （2）繼承性

子標籤會繼承父標籤的某些樣式。

繼承的樣式一般是：text-，font-，line-，color，行高。

高度、內外邊距不會繼承。

body {

font: 12px/1.5 Microsoft YaHei;

}

行高可以寫單位，也可以不寫。上面的例子就沒有寫單位，直接寫了1.5。這指的是行高是當前文字大小（font-size）的1.5倍。

比如下面的例子，是要body裡面所有子元素的行高是font-size的1.5倍。div的font-size是14px，那麼line-height就是21px。

<style>

      body {

        font: 12px/1.5 Microsoft YaHei;

      }

      div {

        font-size: 14px;

      }

    </style>

  </head>

  <body>

    <div></div>

  </body>

這樣寫的優勢：讓子元素根據自己的文字大小來調整行高。

### （3）優先級

選擇器相同，執行層疊性。

選擇器不同，根據選擇器權重執行。

|  |  |
| --- | --- |
| 選擇器 | 選擇器權重（依次提高）（權重會疊加，但是不會進位） |
| 繼承 或者 \*（通配符選擇器） | 0，0，0，0 |
| 元素選擇器 | 0，0，0，1 |
| 類選擇器，偽類選擇器 | 0，0，1，0 |
| ID選擇器 | 0，1，0，0 |
| 行內樣式 style=“” | 1，0，0，0 |
| !important 重要的 | ∞無窮大 |

標籤a有默認樣式：顏色藍色，有下劃線。這個時候，如果其父級元素雖然設置了顏色，和text-decoration，但由於是繼承來的，權重小於a本身的元素選擇器權重，所以顯示的a是藍色有下劃線。

#### 優先級——權重疊加

div ul li → 0，0，0，3

.nav ul li → 0，0，1，2

a:hover → 0，0，1，1

.nav a → 0，0，1，1

## CSS盒子模型

### （1）網頁佈局過程

① 準備好相關的網頁元素，網頁元素基本都是盒子Box

② 利用CSS設置盒子樣式，然後擺放到相應位置

③ 往盒子裡裝內容

### （2）盒子模型（Box Model）組成

外邊距margin，邊框border，內邊距padding，內容content。

#### ①外邊距

#### a）margin的簡寫方式和padding完全一致

#### b）外邊距典型運用：讓塊級盒子水平居中

條件：

① 盒子指定了寬度（width）；

② 盒子左右外邊距設置為auto，常見寫法如下3種：

margin-left: auto; margin-right: auto;

margin: auto;

margin: 0 auto;

注意：標準流、相對定位的盒子才可以這樣寫。絕對定位和固定定位的盒子，這樣寫無效。

#### c）讓行內元素或者行內塊元素水平居中

讓行內元素或者行內塊元素水平居中：給父元素添加text-align: center;

#### d）相鄰塊元素垂直外邊距的合併

當兄弟關係的上下2個塊元素相遇時，如果上面的元素有下外邊距margin-bottom，下面的元素有上外邊距margin-top，他們之間的垂直間距不是margin-bottom和margin-top之和，而是兩個值之中的加大者，這種現象被稱為相鄰塊元素垂直外邊距的合併。



解決方案：盡量只給一個盒子添加margin值

#### e）嵌套塊元素垂直外邊距的塌陷

對於2個嵌套關係（父子關係）的塊元素，父元素有上外邊距，同時子元素也有上外邊距，此時上外邊距會變成較大的外邊距值。



解決方案：

* 為父元素定義上邊框。
* 為父元素定義上內邊距padding-top。
* 為父元素添加overflow: hidden。
* 還有其他方法，比如浮動、固定、絕對定位的盒子不會有塌陷問題。

#### f）行內元素的外邊距

為了兼容性更好，盡量只設置行內元素左右的內外邊距，不要設置上下的內外邊距。

行內元素轉為塊級元素和行內塊元素就可以四邊都設置內外邊距了。

#### ②邊框

border: border-width（粗細，單位px） | border-style | border-color

border-style的值：none（無邊框），solid（實線邊框），dotted（虛線邊框），dashed（虛線邊框）

邊框值的復合寫法，任意順序：

border: 1px solid red;

只設定某一條邊框：

border-top: 1px solid red; （這是上邊框，其餘同理）

邊框會影響盒子寬高。寫盒子寬高時，要減去border值。

#### ③內邊距

padding簡寫

padding: 5px; 上下左右都有5個像素內邊距；

padding: 5px 10px; 上下5px，左右10px；

padding: 5px 10px 20px; 上5px，左右10px，下20px；

padding: 5px 10px 20px 30px; 上右下左。

內邊距會影響盒子寬高。寫盒子寬高時，要減去padding值。

只要不指定盒子寬高，padding就不會撐開盒子。

清除內外邊距

\* {

padding: 0;

margin: 0;

}

## CSS浮動

網頁佈局的本質：用CSS來擺放盒子

傳統網頁佈局的三種方式：

* 普通流（標準流）（標籤按照默認的方式排列）
* 浮動
* 定位

### （1）浮動定義

float屬性用於創建浮動框，將其移動到一邊。直到左邊緣或者右邊緣觸及包含塊或另一個浮動框的邊緣。

語法：

選擇器 {float: 屬性值;}

屬性值：

* none 元素不浮動（默認值）
* left 元素向左浮動
* right 元素向右浮動

### （2）浮動特性

#### （2-1）浮動元素會脫離標準流（脫標）

① 脫標：脫離標準普通流的控制，移動（浮動）到指定位置

② 浮動的盒子不再保留原先的位置

例子：兩個div，第1個加了左浮動，第2個標準流。那麼視覺上，第1個div會壓住第2個div。

③ 浮動只會影響後面的標準流，不會影響前面的。

#### （2-2）浮動元素會一行內顯示並且元素頂部對齊

#### （2-3）浮動元素會具有行內塊元素的相似特性

任何元素都可以浮動。不管原先是什麼模式的元素，添加浮動之後具有行內塊元素相似的特性。

給行內元素設置了浮動，則不需要轉換為塊元素/行內塊元素，就可以直接給寬度和高度。

如果塊元素沒有設置寬度，默認寬度和父級元素一樣寬；添加浮動後，如果沒有設置寬度，其大小根據內容來決定。

### （3）浮動元素經常搭配標準流的父元素

先用標準流的父元素排列上下位置，之後內部子元素採取浮動排列左右位置。



### （4）常見網頁佈局





上面3個圖依次更複雜。

其中top和footer不需要設置寬度，默認和瀏覽器一樣寬。

## CSS清除浮動

### （1）為什麼要清除浮動

不方便給父盒子高度的場景：

① 產品很多

② 每篇新聞字數不一樣

這時候，讓子盒子撐開父盒子，不給父盒子高度。這時候，出現一個問題：當所有子盒子浮動，父盒子高度會變成0；父盒子所在標準流下面的盒子，會佔據本應屬於父盒子的空間。

### （2）清除浮動本質

父盒子根據浮動的子盒子自動檢測高度。父盒子有了高度，就不會影響下面的標準流了。

### （3） 清除浮動語法

選擇器{clear: 屬性值;}

|  |  |
| --- | --- |
| **屬性值** | **描述** |
| left | 不允許左側有浮動元素（清除左側浮動的影響） |
| right | 不允許右側有浮動元素（清除右側浮動的影響） |
| both | 同時清除左右兩側浮動的影響 |

實際工作中，最常用clear: both;

### （4） 清除浮動方法① 額外標籤法=隔墻法

在浮動元素末尾添加一個空標籤（必須是塊級元素），並寫上清除浮動的語法。

比如<div style=”clear: both”></div>

優點：通俗易懂，書寫方便。

缺點：添加許多無意義的標籤，影響結構優化。

雖然W3C推薦這麼寫，但講師不推薦。

### （5）清除浮動方法② 父級添加overflow屬性

給父盒子添加overflow屬性，將值設為hidden（常用）、auto或者scroll。

優點：代碼簡潔

缺點：無法顯示父盒子溢出的部分

原理：在添加overflow屬性後，浮動元素又回到了容器層，把容器高度撐起，達到了清理浮動的效果。

### （6）清除浮動方法③ 父級添加after偽元素

.clearfix:after {

content: “”, （偽元素必須寫的屬性）

display: block; （插入的元素必須是塊級）

height: 0; （不要看見這個元素）

clear: both; （核心代碼清除浮動）

visibility: hidden; （不要看見這個元素）

}

IE6、7專用：

.clearfix {

\*zoom: 1;

}

把上面的2組代碼（即class=”clearfix”）運用到父盒子上。意味著在父盒子的內部、所有子盒子的最後面添加了這個:after的內容。注意：after是添加在父盒子內部最後面，而不是父盒子本身的後面。

優點：沒有增加標籤，結構更簡單。

缺點：要照顧低版本瀏覽器。

代表網站：百度、淘寶、網易。

我在網上查到的一個版本：

.clearfix:after {

    content:"\200B";

    display:block;

    height:0;

    clear:both;

}

.clearfix {\*zoom:1;}/\*IE/7/6\*/

解釋:content:"\200B";這個參數，Unicode字符裏有一個「零寬度空格」，即 U+200B，代替原來的「.」，可以縮減代碼量。而且不再使用visibility:hidden。

### （7）清除浮動方法④ 父級添加雙偽元素

.clearfix:before, .clearfix:after {

content: “”; （偽元素必須寫的屬性）

display: table; ~~（變成塊級元素並且一行顯示。如果用display: block;會導致before和after兩個元素上下顯示。而display: table;可以讓before和after兩個元素在一行顯示。）~~ 對黑馬的這個解釋存疑。display:table的作用見“清除浮動的一些總結”之（3）的內容

}

.clearfix:after {

clear: both; （核心代碼清除浮動）

}

.clearfix {

\*zoom:1;

}

把上面的3組代碼（即class=”clearfix”）運用到父盒子上。

優點：代碼更簡潔。

缺點：照顧低版本瀏覽器。

代表網站：小米，騰訊等。

講師自己喜歡這種方式。

## 清除浮動的一些總結

### （1）::after和:after（偽元素和偽類的區別）

① 單冒號和雙冒號的區別

CSS3之後才有雙冒號:: 表示偽元素。

偽類用單冒號。

② 偽元素

偽元素是選中了某個元素的符合邏輯的某個實際卻不存在的部分，本質上是創建了一個有內容的虛擬容器；

用法 selector:pseudo-element {property:value;}

first-letter設置首字母的樣式，first-line設置首行的樣式

before在元素之前添加內容

after在元素之後插入內容

偽類則是像真正的類一樣發揮著類的作用

用法 selector : pseudo-class {property: value}

③ 偽類

常見偽類

:link 未被訪問的鏈接

:visited 已被訪問的鏈接

:hover 鼠標懸停

:active 被激活的元素

:focus 有鍵盤輸入焦點

css定義中，:hover必須位於:link和:visited之後才有效，

:active 必須被置於:hover 之後，才有效

:lang——:lang(no)指定lang屬性為no或no-的元素

:lang 的處理與 |= 選擇器相同。

:first-child 父元素的第一個子元素（在IE中必須聲明DOCTYPE）

CSS3新增了許多偽類

第n個特定類型元素（xxxtype）

:first-of-type，:last-of-type，:nth-of-type(n)，:only-of-type，:nth-last-of-type(n)

特定類型元素的首個/尾個/第幾個/唯一一個/倒數第幾個

類似first-child的選擇第n個元素的偽類

:last-child，:nth-child(n)，:only-child，:nth-last-child(n)

:root根元素（HTML中根元素始終是HTML元素）

:empty沒有子元素的元素（包括文本節點）

:target當前活動的目標元素

url後帶有錨名稱#指向文檔內某元素，該被指向元素為目標元素

:enable, :valid, :read-only,:checked,:required

......

註意

可以同時使用多個偽類，而只能同時使用一個偽元素；

### （2）BFC小結

#### ① 常見定位方案

在講 BFC 之前，我們先來了解一下常見的定位方案，定位方案是控製元素的布局，有三種常見方案:

* 普通流 (normal flow)

在普通流中，元素按照其在 HTML 中的先後位置至上而下布局，在這個過程中，行內元素水平排列，直到當行被占滿然後換行，塊級元素則會被渲染為完整的一個新行，除非另外指定，否則所有元素默認都是普通流定位，也可以說，普通流中元素的位置由該元素在 HTML 文檔中的位置決定。

* 浮動 (float)

在浮動布局中，元素首先按照普通流的位置出現，然後根據浮動的方向盡可能的向左邊或右邊偏移，其效果與印刷排版中的文本環繞相似。

* 絕對定位 (absolute positioning)

在絕對定位布局中，元素會整體脫離普通流，因此絕對定位元素不會對其兄弟元素造成影響，而元素具體的位置由絕對定位的坐標決定。

#### ② BFC 概念

Formatting context(格式化上下文) 是 W3C CSS2.1 規範中的一個概念。它是頁面中的一塊渲染區域，並且有一套渲染規則，它決定了其子元素將如何定位，以及和其他元素的關系和相互作用。

那麽 BFC 是什麽呢？

BFC 即 Block Formatting Contexts (塊級格式化上下文)，它屬於上述定位方案的普通流。

**具有 BFC 特性的元素可以看作是隔離了的獨立容器，容器裏面的元素不會在布局上影響到外面的元素，並且 BFC 具有普通容器所沒有的一些特性。**

**通俗一點來講，可以把 BFC 理解為一個封閉的大箱子，箱子內部的元素無論如何翻江倒海，都不會影響到外部。**

#### ③ 觸發 BFC

只要元素滿足下面任一條件即可觸發 BFC 特性：

* body 根元素
* 浮動元素：float 除 none 以外的值
* 絕對定位元素：position (absolute、fixed)
* display 為 inline-block、table-cells、flex
* overflow 除了 visible 以外的值 (hidden、auto、scroll)

#### ④ BFC 特性及應用

#### ④-1 同一個 BFC 下外邊距會發生折疊

<head>

div{

width: 100px;

height: 100px;

background: lightblue;

margin: 100px;

}

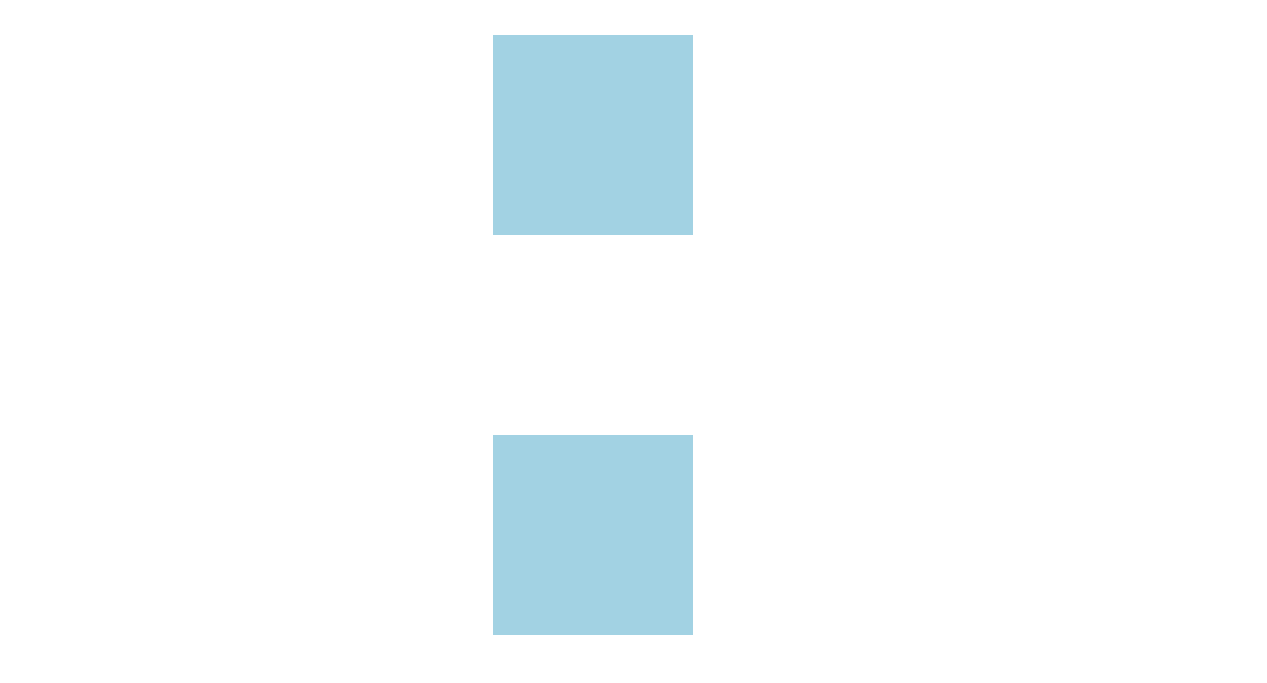
</head>

<body>

<div></div>

<div></div>

</body>



從效果上看，因為兩個 div 元素都處於同一個 BFC 容器下 (這裏指 body 元素) 所以第一個 div 的下邊距和第二個 div 的上邊距發生了重疊，所以兩個盒子之間距離只有 100px，而不是 200px。

這不是 CSS 的 bug，我們可以理解為一種規範，如果想要避免外邊距的重疊，可以將其放在不同的 BFC 容器中。

<div class**=**"container">

<p></p>

</div>

<div class**=**"container">

<p></p>

</div>

.**container** {

**overflow**: **hidden**;

}

p {

**width**: 100**px**;

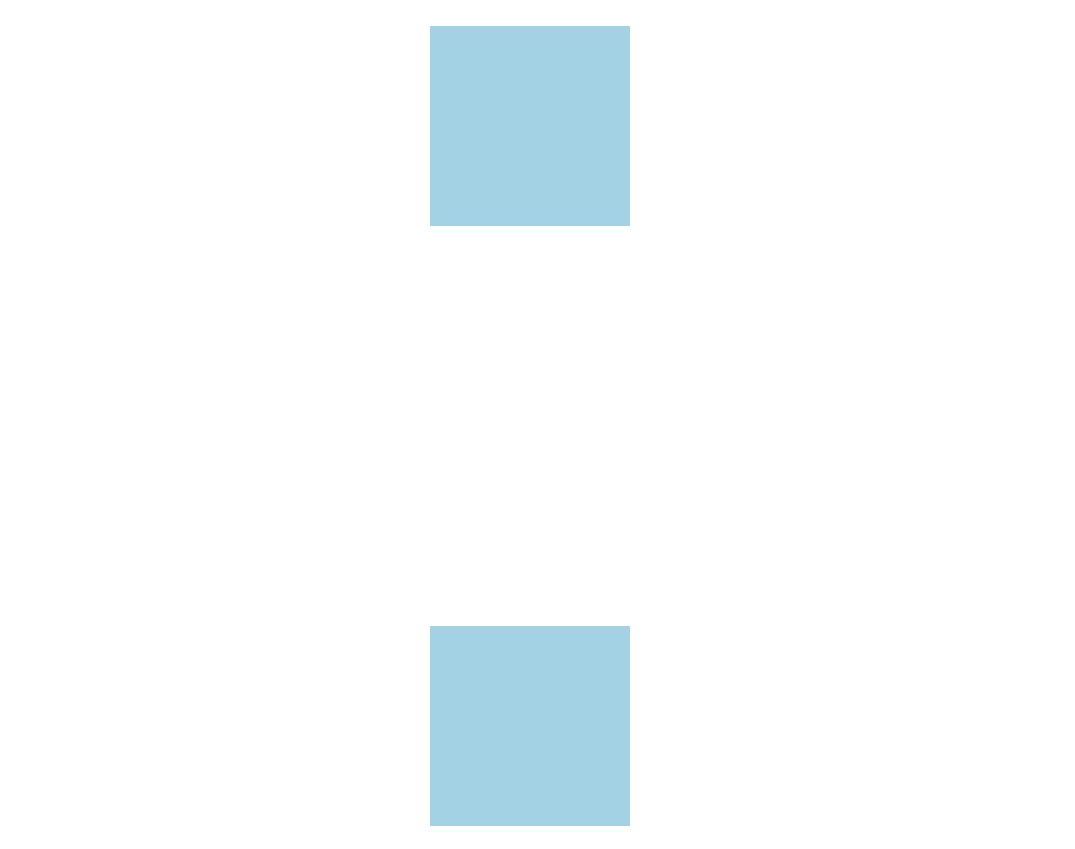
**height**: 100**px**;

**background**: **lightblue**;

**margin**: 100**px**;

}

這時候，兩個盒子邊距就變成了 200px



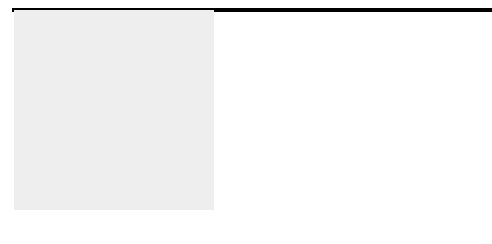
#### ④-2 BFC 可以包含浮動的元素（清除浮動）

我們都知道，浮動的元素會脫離普通文檔流，來看下下面一個例子

<div style**=**"border: 1px solid #000;">

<div style**=**"width: 100px;height: 100px;background: #eee;float: left;"></div>

</div>



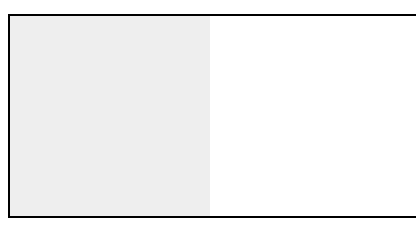
由於容器內元素浮動，脫離了文檔流，所以容器只剩下 2px 的邊距高度。如果使觸發容器的 BFC，那麽容器將會包裹著浮動元素。

<div style**=**"border: 1px solid #000;overflow: hidden">

<div style**=**"width: 100px;height: 100px;background: #eee;float: left;"></div>

</div>

效果如圖：



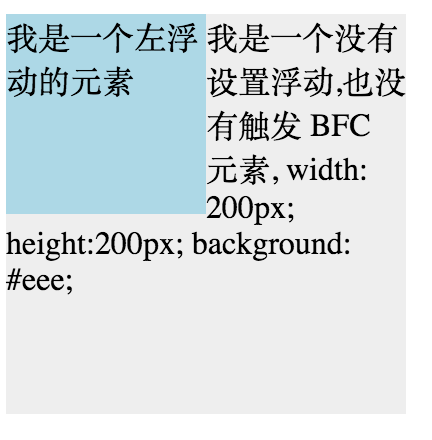
#### ④-3 BFC 可以阻止元素被浮動元素覆蓋

先來看一個文字環繞效果：

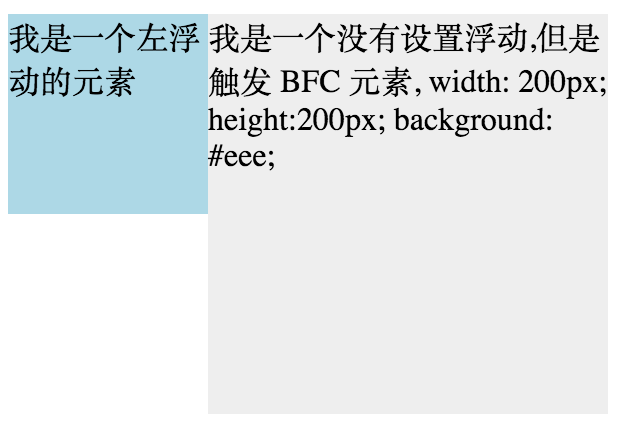
<div style**=**"height: 100px;width: 100px;float: left;background: lightblue">我是一个左浮动的元素</div>

<div style**=**"width: 200px; height: 200px;background: #eee">我是一个没有设置浮动,

也没有触发 BFC 元素, width: 200px; height:200px; background: #eee;</div>



這時候其實第二個元素有部分被浮動元素所覆蓋，(但是文本信息不會被浮動元素所覆蓋) 如果想避免元素被覆蓋，可觸第二個元素的 BFC 特性，在第二個元素中加入 overflow: hidden，就會變成：



這個方法可以用來實現兩列自適應布局，效果不錯，這時候左邊的寬度固定，右邊的內容自適應寬度(去掉上面右邊內容的寬度)。

### （3）清除浮動的display:table

.clearfix:before, .clearfix:after {

content: “”;

display: table;

}

.clearfix:after {

clear: both;

}

.clearfix {

\*zoom:1;

}

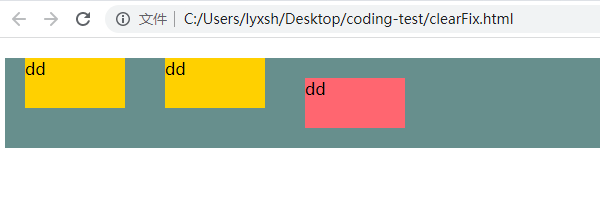
上面的代碼中，為什麼用display:table，而不用display:block？

經過試驗，display:table會觸發:before和:after偽元素的BFC。

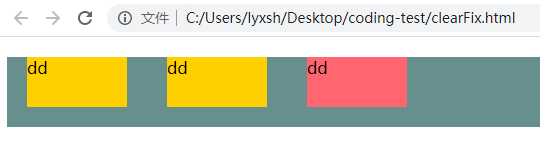
如果父盒子內的子盒子全部浮動，display:table和display:block無區別。

如果父盒子內部分盒子浮動，用display:table的話，可以防止不浮動的子盒子和父盒子垂直外邊距的塌陷。

比如下面的例子，如果用display:table，會這樣：



如果用display:block，會這樣



原因。display:table會觸發:before和:after偽元素的BFC。紅色的Box3的上面和下面就分別有了一個BFC區域，能夠阻止父子盒子邊距合併。

相關代碼

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      .box-wrapper {

        background-color: #678f8d;

        margin: 20px 0;

      }

      .box {

        float: left;

        width: 100px;

        height: 50px;

        background-color: #ffd000;

        margin: 0 20px;

      }

      .box3 {

        width: 100px;

        height: 50px;

        background-color: #ff6670;

        margin: 20px 0 20px 300px;

      }

      .clearfix:before,

      .clearfix:after {

        content: "";

        display: block;

      }

      .clearfix:after {

        clear: both;

      }

      .clearfix {

        \*zoom: 1;

      }

    </style>

  </head>

  <body>

    <div class="box-wrapper clearfix">

      <div class="box">dd</div>

      <div class="box">dd</div>

      <div class="box3">dd</div>

    </div>

  </body>

</html>

另發現一個新屬性display: flow-root，不支持Safari。它也可以創建BFC。清除浮動時，display: flow-root和display: table作用一樣。

## 定位

### （1）定位組成

定位=定位模式+邊偏移

定位模式通過position設置，值有4個：

static 靜態定位

relative 相對定位

absolute 絕對定位

fixed 固定定位

邊偏移有top，bottom，left和right四個屬性。

|  |  |  |
| --- | --- | --- |
| 邊偏移屬性 | 示例 | 描述 |
| top | top: 80px; | 頂端偏移量，定義元素相對於其父元素上邊線的距離。 |
| bottom | bottom: 80px; | 底部偏移量，定義元素相對於其父元素下邊線的距離。 |
| left | left: 80px; | 左側偏移量，定義元素相對於其父元素左邊線的距離。 |
| right | right: 80px; | 右側偏移量，定義元素相對於其父元素右邊線的距離。 |

### （2）靜態定位

默認定位方式，無定位。

靜態定位按照標準流特性擺放位置，沒有邊偏移。

很少用到靜態定位。

### （3）相對定位（重要）

① 參照它自己原來的位置來移動。

② 相對定位前的位置以標準流的形式繼續保留。

### （4）絕對定位（重要）

① 元素移動位置時，相對於祖先元素。如果沒有祖先元素或者祖先元素沒有定位，則以瀏覽器為準定位（Document文檔）。

② 如果祖元素有定位（相對，絕對，固定定位），則以**最近一級**的**有定位**祖元素為參考點移動位置。

③ 絕對定位不再佔有原先的位置，脫離標準流了。

### （5）子絕父相

① 子級絕對定位，不會佔有位置，可以放到父盒子裡面的任何一個地方，不會影響其它的兄弟盒子。

② 給父盒子加定位，以限制子盒子，讓子盒子只在父盒子內移動。

不能給父盒子加絕對定位，因為加了絕對定位的父盒子會脫離標準流，進而影響整個頁面佈局。所以只能加相對定位。

### （6）固定定位（重要）

① 主要使用場景：瀏覽器頁面滾動時，元素的位置不變。

② 已瀏覽器**可視窗口**為參照點移動元素。根據瀏覽器界面放大縮小靈活變化。跟父元素沒關係。不隨滾動條滾動。

③ 固定定位不佔有原先的位置。

固定定位是脫標的，可以把固定定位看做是一種特殊的絕對定位。

### （7）固定定位小技巧：固定在版心右側位置

第①步：讓固定定位的盒子left: 50%，走到瀏覽器可視區（也是版心）的一半位置。

第②步：讓這個盒子margin-left：版心寬度的一半。這樣再往右走版心寬度的一半位置。就可以讓這個盒子貼著版心右邊對齊了。

比如下圖的示例



### （8）黏性定位（了解）

相對定位和固定定位的混合

語法： 選擇器{ position: sticky; top: 10px; }

黏性定位特點：

① 以瀏覽器可視窗口為參照點移動元素（固定定位特點）

② 黏性定位佔有原先的位置（相對定位特點）

③ 必須添加top、left、right、bottom其中一個才有效。

一般跟頁面滾動搭配使用。但因為兼容性差（IE不支持），所以不常用。

### （9）定位總結

|  |  |  |  |
| --- | --- | --- | --- |
| 定位模式 | 是否脫標 | 移動位置 | 是否常用 |
| static 靜態定位 | 否 | 不能使用邊偏移 | 很少 |
| relative 相對定位 | 否（佔有位置） | 相對於自身位置移動 | 常用 |
| absolute 絕對定位 | 是（不佔位置） | 帶有定位的父級 | 常用 |
| fixed 固定定位 | 是（不佔位置） | 瀏覽器可視區 | 常用 |
| sticky 黏性定位 | 否（佔有位置） | 瀏覽器可視區 | 當前階段少 |

① 一定記住 相對定位、固定定位、絕對定位兩個大的特點：a 是否佔位置（脫標了嗎）b 以誰為基準點移動位置。

② 重點學會子絕父相。

### （10）定位疊放次序 z-index

使用定位佈局時，可能會出現盒子重疊的情況。可以使用z-index來控制盒子的次序（z軸）

語法： 選擇器{z-index: 1;}

① 值可以是正整數，負整數，或者0。默認是auto，數值越大，盒子越靠上。

② 如果屬性值相同，則按照代碼順序，後來居上。

③ 數字後面不能加單位。

④ 只有定位的盒子才有z-index屬性。標準流和浮動沒有z-index屬性。

### （11）絕對定位的盒子居中方法

第①步： 往右走父容器寬度的50% left: 50%;

第②步： 往左走自己寬度的50% margin-left: -100px;

### （12）定位特點

絕對定位、固定定位和浮動類似。

① 行內元素添加絕對或者固定定位，可以直接設置高度和寬度。

② 塊級元素添加絕對或者固定定位，如果不給寬度或者高度，默認大小是內容的大小。

③ 脫標的盒子不會觸發外邊距塌陷。

浮動元素、絕對定位的元素、固定定位的元素都不會觸發外邊距合併的問題，因為它們都脫離了標準流。

④ 是否壓住文字和圖片

a 浮動元素只會壓住下面標準流的盒子，但不會壓住這些標準流盒子裡的文字和圖片。

浮動之所以不會壓住文字，是因為浮動產生的最初目的是為了做文字環繞效果的，文字會環繞浮動元素。

b 絕對定位和固定定位會壓住下面標準流盒子裡的所有內容。

## 元素的顯示和隱藏

### （1）display屬性

display屬性用於設置一個元素如何顯示

display: none; 隱藏元素。元素隱藏後，不再佔有原來的位置。

display: block; 除了轉換為塊級元素外，還有顯示元素的意思。

### （2）visibility可見性

visibility屬性用於指定一個元素應該可見還是隱藏。

visibility: visible; 元素可見

visibility: hidden; 元素隱藏。元素隱藏時，繼續佔有原來的位置。

### （3）overflow 溢出顯示隱藏

overflow: visible; 默認值。一個盒子大小固定，文字內容太多，超過盒子大小，默認狀態會溢出盒子全部顯示出來。

overflow: hidden; 把盒子裡溢出的部分隱藏起來。

overflow: scroll; 給盒子加滾動條，把所有的文字內容都顯示在盒子內。（不論文字是否超出盒子，都添加滾動條。）

overflow: auto; 在需要的時候（文字溢出盒子的時候），才給盒子添加滾動條。

### （4）應用案例：當鼠標經過父盒子，改變子盒子屬性

<div class=”tudou”>

<div class=”mask”></div>

<img src=”images/tudou.jpg” alt=”tudou”>

</div>

class=”mask”這個div本來是display: none;的。鼠標經過class=”tudou”這個div時，顯示class=”mask”這個div

.tudou:hover .mask {

display: block;

}

CSS進階

## 精靈圖

### （1）為什麼需要精靈圖

一個網頁中往往會應用很多小的背景圖作為修飾，當網頁中的圖像過多時，服務器就會頻繁地接收和發送請求圖片，造成服務器請求壓力過大，這將大大降低頁面的加載速度。

為了有效地減少服務器接收和發送請求的次數，提高頁面的加載速度，出現了CSS精靈技術（也稱CSS Sprites，CSS雪碧）。

核心原理：將網頁中的一些小背景圖像整合到一張大圖中，服務器只需要一次請求就可以了。

### （2）使用精靈圖

用於背景圖片。把多個小背景圖片整合到一張大圖片中。

移動背景圖片位置，用background-position

移動時沿著X和y軸移動，一般情況下是往上往左移動，其數值都是負數。

用PS來測量需要移動多少位置。在測量之前，先把圖片的圖層鎖定（在眼睛圖標後面，有個小鎖的圖標）

在PS中，選擇切片工具，選中相關圖片，在PS左下角，可以看到寬高和XY軸的數值。

## 字體圖標

### （1）為什麼需要字體圖標

字體圖標使用場景：顯示網頁中通用、常用的小圖標

精靈圖缺點：

① 圖片文件比較大

② 圖片本身放大和縮小會失真

③ 更換圖片複雜

字體圖標iconfont可以解決上述問題

字體圖標是一種高效方便的圖標使用方式，展示的是圖標，本質屬於字體。

字體圖標的優點：

① 輕量級：圖標字體比圖像小。一旦字體加載了，圖標會馬上渲染出來，減少服務器請求

② 靈活性：本質是文字，所以可以隨意改變顏色、產生陰影、透明效果、旋轉等。

③ 兼容性：幾乎支持所有瀏覽器。

注意：字體圖標不能代替精靈技術，只是對工作中圖標部分技術的提升和優化。

① 結構樣式簡單的小圖標，用字體圖標。

② 結構樣式複雜的圖片，用精靈圖。

### （2）如何使用字體圖標

① 下載字體圖標

推薦下載網站：

icomoon <https://icomoon.io/>

IcoMoon成立於2011年。推出了第一個自定義圖標字體生成器，允許用戶選擇所需要的圖標。種類繁多，全面。外國服務器，打開速度慢。

進入網站後，點擊右上角的IconMoon App紫色圖標，進入圖標挑選頁面。

在該頁面選擇需要的圖標。

如果沒找到需要的，可以在網頁最下面的Add Icons From Library…中找。

網頁最上方的工具欄裡，有編輯按鈕，可以編輯被選中的圖標。

選好所有圖標後，點擊頁面右下角Generate Font。

圖標生成好後，會進入新的頁面，在該頁面右下角Font Download。

阿里iconfont字庫

這是阿里媽媽M2UX的一個iconfont字體圖標字庫，包含了淘寶圖標和阿里媽媽圖標庫，可以使用AI製作圖標上傳生成。

② 引入字體圖標（引入到HTML頁面中）

下載後的字體是個壓縮包，將其解壓。

把fonts文件夾放到項目頁面的根目錄下。（fonts文件夾裡有4個字體文件，為了兼容不同瀏覽器）

在CSS樣式中全局聲明字體（把這些字體文件通過CSS引入頁面）

在被解壓的文件夾中，有個style.css文件，把裡面的代碼複製一下：

@font-face {

font-family: 'icomoon';

src: url('fonts/icomoon.eot?1gc3g9');

src: url('fonts/icomoon.eot?1gc3g9#iefix') format('embedded-opentype'),

url('fonts/icomoon.ttf?1gc3g9') format('truetype'),

url('fonts/icomoon.woff?1gc3g9') format('woff'),

url('fonts/icomoon.svg?1gc3g9#icomoon') format('svg');

font-weight: normal;

font-style: normal;

font-display: block;

}

把上面的代碼（叫做字體聲明）粘貼到HTML head 部分的<style></style>之間

在被解壓的文件夾裡，找到demo.html。在這裡可以找到所有下載下來的圖標的對應字符。複製一下小方塊，然後粘貼到HTML代碼相關的盒子中。



比如這個英鎊符號的小方塊放在了span裡面

然後在head 部分的<style></style>之間寫上：

span{

font-family: 'icomoon';

（還可以指定各種字體樣式，比如）

font-size: 10px;

color: pink;

}

③ 追加字體圖標

如果在工作中，原來的字體圖標不夠用了，需要添加新的字體圖標到原來的字體文件中。

打開icomoon.io主頁，點擊右上角的IcoMoon App紫色圖標。

在新頁面點擊左上角Import Icons紫色圖標。把壓縮包裡的selection.json上傳，然後選中自己想要的新圖標，重新下載壓縮包，並替換原來的文件即可。

## CSS寫三角形

寫個div，不給寬高，只給某一個邊框顏色，其它邊框顏色透明，同時設置所有邊框粗細和實線。通過有顏色邊框的方位、粗細、顏色來控制三角形的樣式。

語法如下：

div {

width: 0;

height: 0;

line-height: 0; （為了兼容低版本瀏覽器）

font-size: 0; （為了兼容低版本瀏覽器）

border: 50px solid transparent;

border-left-color: pink;

}

## 用戶界面樣式

### （1）鼠標樣式cursor

li {cursor: pointer;}

### （2）取消表單輪廓

① 去掉默認的邊框border: 0;

② 把光標放在input和textarea中時，會出現默認藍色邊框，要去掉這個邊框，可以：

outline:0; 或者outline: none;

例子：

input, textarea {outline:none;}

### （3）防止拖拽文本域 resize

textarea{resize:none;}

### （4）防止文本域裡開頭出現空白區域

把textarea開頭結尾2個標識寫在一行：<textarea></textarea>。而不是分2行來寫。

## vertical-align 圖片或者以文本域為代表的表單（行內塊元素）和文字垂直對齊

用於設置一個元素的垂直對齊方式，只針對行內元素或者行內塊元素。



問題：圖片在不給寬高時，自己撐開父盒子，圖片和父盒子底端之间有個縫隙。因為盒子屬於行內塊元素，默認的vertical-align是基線對齊。

解決方法有2個：

① 給圖片添加vertical-align: middle | top | bottom等 （導師建議用這種）

② 把圖片轉換為塊級元素：img{display: block;} （塊級元素獨佔一行，可能會影響其它元素的佈局）

## 文字溢出省略號

#### ① 單行文本溢出顯示省略號

a 強制一行內顯示文本

white-space: nowrap; （默認normal，自動換行）

b 超出的部分隱藏

overflow: hidden;

c 文字用省略號代替超出的部分

text-overflow: ellipsis;

#### ② 多行文本溢出顯示省略號

多行文本溢出顯示省略號，有較大兼容性問題，適合於webKit瀏覽器或移動端（移動端大部分是Webkit內核）

使用下列代碼時，要先確定裝內容的盒子寬高。

overflow: hidden;

text-overflow: ellipsis;

/\* 彈性伸縮盒子模型顯示 \*/

display: -webkit-box;

/\* 限制在一個塊元素顯示的文本的行數 \*/

-webkit-line-clamp: 2;

/\* 設置或檢索伸縮和對象的子元素的排列方式 \*/

-webkit-box-orient: vertical;

更推薦讓後台人員來做這個效果，因為後台人員可以設置顯示多少個字，操作更簡單。

## 佈局技巧

### （1）margin負值

運用場景：在ul裡，給每個左浮動的li設置邊框1px，相鄰的邊框靠在一起，視覺上這裡的邊框就變成了2px。要避免這種情況，可以設置li{margin-left: -1px}，讓相鄰的邊框上下擺放，這樣就都是1px的邊框了。

上述場景延伸：鼠標經過某個li的時候，要顯示這個li的整個邊框。不做後續操作的時候，只能顯示上、左、下邊框，右側邊框被右邊的盒子壓住了。要解決這個問題，可以在hover裡面給當前的盒子裡：若無定位，加相對定位（保留位置）；若有定位，加z-index。

### （2）文字圍繞浮動元素巧妙運用

寫個父盒子，裡面直接放入文字，不用設置寬高，標準流。

然後放入img，給圖片添加左浮動。

文字就能自動環繞圖片。

比如做個這樣的效果圖：



代碼如下：

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      \* {

        margin: 0;

        padding: 0;

      }

      .box {

        width: 310px;

        height: 70px;

        background-color: pink;

        margin: 0 auto;

        padding: 5px;

      }

      .pic {

        float: left;

        width: 120px;

        height: 60px;

        margin-right: 5px;

      }

      .pic img {

        width: 100%;

      }

    </style>

  </head>

  <body>

    <div class="box">

      <!-- 拿盒子包住圖片是因為在後面圖片可能會被經常換掉。 -->

      <div class="pic"><img src="./th.jpg" alt="" /></div>

      <p>隐隐飞桥隔野烟，石矶西畔问渔船。 桃花尽日随流水，洞在清溪何处边。</p>

    </div>

  </body>

</html>

### （3）行內塊巧妙運用

需求：頁碼在頁面中間顯示



實現方法：把這些鏈接盒子轉換為行內塊元素，然後給父盒子指定text-align:center;

代碼：

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      \* {

        margin: 0;

        padding: 0;

      }

      .box {

        text-align: center;

      }

      .box a {

        display: inline-block;

        width: 36px;

        height: 36px;

        background-color: #f7f7f7;

        border: 1px solid #ccc;

        text-align: center;

        line-height: 36px;

        text-decoration: none;

        color: #333;

        font-size: 14px;

      }

      .box .prev,

      .box .next {

        width: 85px;

      }

      .box .current,

      .box .elp {

        background-color: #fff;

        border: none;

        /\* 或者border: 0; \*/

      }

      .box input {

        height: 36px;

        width: 45px;

        border: 1px solid #ccc;

        outline: none;

      }

      .box button {

        width: 60px;

        height: 36px;

        background-color: #f7f7f7;

        border: 1px solid #ccc;

      }

    </style>

  </head>

  <body>

    <div class="box">

      <a href="#" class="prev">&lt;&lt;上一頁</a>

      <a href="#" class="current">2</a>

      <a href="#">3</a>

      <a href="#">4</a>

      <a href="#">5</a>

      <a href="#">6</a>

      <a href="#" class="elp">...</a>

      <a href="#" class="next">&gt;&gt;下一頁</a>

      到第

      <input type="text" />

      頁

      <button>確定</button>

    </div>

  </body>

</html>

上述代碼寫出來後，做出來的頁碼在頁面居中顯示：



### （4）三角形強化



用代碼寫出如下圖像：



<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      .box1 {

        width: 0;

        height: 0;

        /\* 只保留右边的边框有颜色 \*/

        border-color: transparent red transparent transparent;

        /\* 边框式样都是solid \*/

        border-style: solid;

        /\* 上边框宽度大，右边框宽度稍小，其余边框为0 \*/

        border-width: 100px 50px 0 0;

      }

      .price {

        width: 160px;

        height: 24px;

        line-height: 24px;

        border: 1px solid red;

        margin: 0 auto;

      }

      .current-price {

        position: relative;

        float: left;

        width: 90px;

        height: 100%;

        background-color: red;

        text-align: center;

        color: #fff;

        font-weight: 700;

        margin-right: 8px;

      }

      /\* 用来做三角形 \*/

      .current-price i {

        position: absolute;

        right: 0;

        top: 0;

        width: 0;

        height: 0;

        border-color: transparent #fff transparent transparent;

        border-style: solid;

        border-width: 24px 10px 0 0;

      }

      .original-price {

        font-size: 12px;

        color: gray;

        text-decoration: line-through;

      }

    </style>

  </head>

  <body>

    <div class="box1"></div>

    <div class="price">

      <span class="current-price">

        $1650

        <i></i>

      </span>

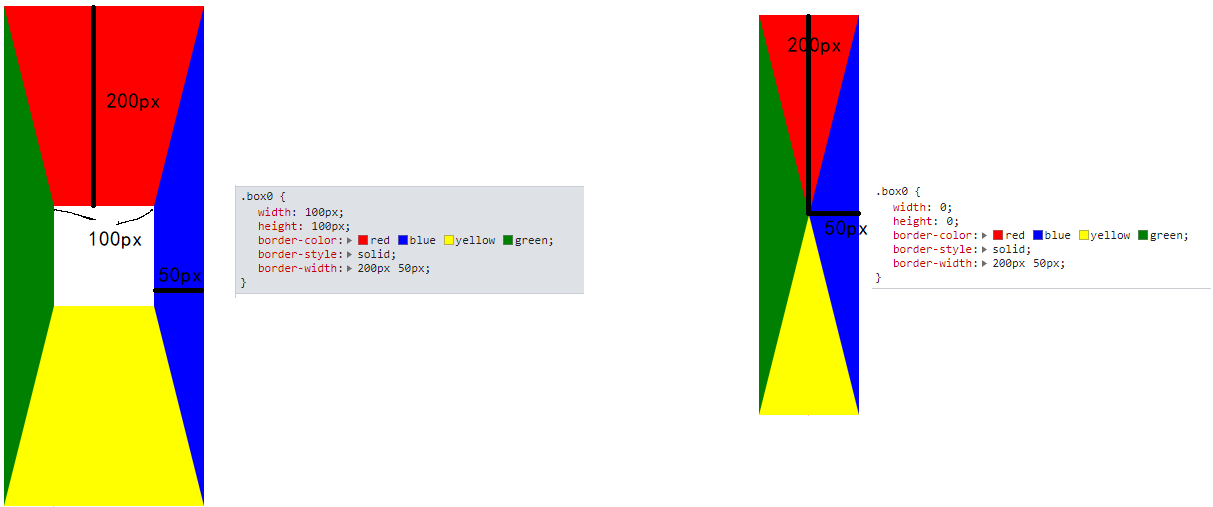
      <span class="original-price">$5650</span>

    </div>

  </body>

</html>

下面的代碼和示例幫助理解邊框畫三角形的方法



### （5）CSS初始化

不同瀏覽器對有些標籤的默認值是不同的，為了消除不同瀏覽器對HTML文本呈現的差異，照顧瀏覽器的兼容，要對CSS初始化。

每個網頁都必須進行CSS初始化。

以京東CSS初始化代碼為例。

img{

border: 0; （照顧低版本瀏覽器，如果圖片外面包含了連接，會有邊框的問題。）

vertical-aligh: middle; （解決圖片底部有空白縫隙的問題）

}

body{

-webkit-font-smoothing: antialiased; （文字放大的時候，有時有鋸齒形。這個CSS3特性可以抗鋸齒形，讓文字顯示得更加清晰）

}

/\* 把我们所有标签的内外边距清零 \*/

\* {

  margin: 0;

  padding: 0;

}

/\* em 和 i 斜体的文字不倾斜 \*/

em,

i {

  font-style: normal;

}

/\* 去掉li 的小圆点 \*/

li {

  list-style: none;

}

img {

  /\* border 0 照顾低版本浏览器 如果 图片外面包含了链接会有边框的问题 \*/

  border: 0;

  /\* 取消图片底侧有空白缝隙的问题 \*/

  vertical-align: middle;

}

button {

  /\* 当我们鼠标经过button 按钮的时候，鼠标变成小手 \*/

  cursor: pointer;

}

a {

  color: #666;

  text-decoration: none;

}

a:hover {

  color: #c81623;

}

button,

input {

  /\* "\5B8B\4F53" 就是宋体的意思 这样浏览器兼容性比较好。如果写汉字，在很多浏览器里会是乱码显示。 \*/

  /\* 黑体 \9ED1\4F53

     宋体 \5B8B\4F53

     微软雅黑 \5FAE\8F6F\96C5\9ED1      \*/

  font-family: Microsoft YaHei, Heiti SC, tahoma, arial, Hiragino Sans GB,

    "\5B8B\4F53", sans-serif;

}

body {

  /\* CSS3 抗锯齿形 让文字显示的更加清晰 \*/

  -webkit-font-smoothing: antialiased;

  background-color: #fff;

  font: 12px/1.5 Microsoft YaHei, Heiti SC, tahoma, arial, Hiragino Sans GB,

    "\5B8B\4F53", sans-serif;

  color: #666;

}

.hide,

.none {

  display: none;

}

/\* 清除浮动 \*/

.clearfix:after {

  visibility: hidden;

  clear: both;

  display: block;

  content: ".";

  height: 0;

}

.clearfix {

  \*zoom: 1;

}

CSS3

## 圓角邊框

語法：

border-radius: length;

原理：length是一個圓的半徑長度，用於和邊框相切。切好了時，邊框就沿著圓弧走，不沿著邊框走了。如下圖，其它3個角同理。



length可以是數值，單位px；也可以是百分比。

把正方形設置為圓，把length修改為邊框一半，或者寫50%。

為長方形設置圓角矩形，把length改為高度的一半。

border-radius:後面可以寫4個值，分別代表左上角、右上角、右下角、左下角的圓半徑。

分開寫：border-top-left-radius，border-top-right-radius，border-bottom-left-radius，border-bottom-right-radius

兼容性：ie9+瀏覽器支持。不影響頁面佈局，可以放心使用。

## 盒子陰影

語法：

box-shadow: h-shadow v-shadow blur spread color inset;

|  |  |
| --- | --- |
| **值** | **描述** |
| h-shadow | 必須。水平陰影的位置。允許負值。 |
| v-shadow | 必須。垂直陰影的位置。允許負值。 |
| blur | 可選。模糊距離。（陰影是虛的還是實的） |
| spread | 可選。陰影的尺寸。（陰影大小） |
| color | 可選。陰影的顏色。可以寫半透明的陰影rgba(0, 0, 0, 0.3) |
| Inset | 可選。將外部陰影（outset）改為內部陰影。 |

注意：

默認的是外陰影（outset），但是不可以寫這個單詞，否則造成陰影無效。

盒子陰影不佔空間，不會影響其它盒子排列。

例子： div { box-shadow: 10px 10px 5px #888888; }

## CSS3的2D轉換及相關案例實現

轉換（transform）是CSS3中具有顛覆性的特征之一，可以實現元素的位移、旋轉、縮放等效果。

### （1）translate

translate 改變元素在頁面中的位置，類似定位。

語法：

transform: translate(x, y); 例如transform: translate(100px, 100px);

或者分開寫：

transform: translateX(n);

transform: translateY(n);

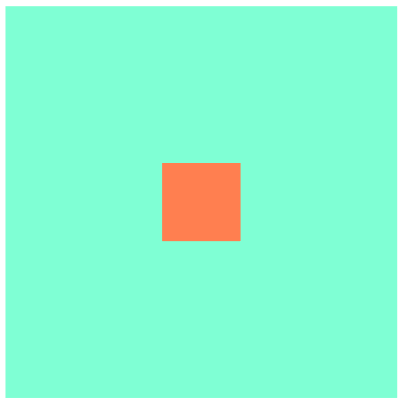
移動盒子的位置有3種方法：①定位 ②margin ③2d轉換移動

* translate不會影響到其他元素的位置，這是定位和外邊距不能實現的效果。
* translate中的百分比單位是相對於自身元素的，比如translate:(50%,50%);
* translate對行內標籤沒有效果

#### translate應用：盒子水平居中垂直居中

先“子絕父相”，然後子盒子top:50%; left:50%;，然後子盒子transform: translate(-50%, -50%);

例子：



<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>盒子垂直水平居中</title>

    <style>

      .big {

        position: relative;

        width: 500px;

        height: 500px;

        background-color: aquamarine;

      }

      .small {

        position: absolute;

        top: 50%;

        left: 50%;

        width: 100px;

        height: 100px;

        background-color: coral;

        transform: translate(-50%, -50%);

      }

    </style>

  </head>

  <body>

    <div class="big">

      <div class="small"></div>

    </div>

  </body>

</html>

### （2）2D轉換之旋轉rotate

讓元素在2維平面內順時針旋轉或者逆時針旋轉。

transform: rotate(45deg); （順時針旋轉45度）

transform: rotate(-45deg); （逆時針旋轉45度）

應用：用於製作箭頭 < >

方法：給before/after 偽元素寫2個邊框，然後rotate旋轉。

下面的例子，鼠標hover span時，向下的箭頭變成向上的箭頭



<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      span {

        position: relative;

        display: inline-block;

        width: 100px;

        height: 30px;

        background-color: coral;

      }

      span::after {

        position: absolute;

        right: 5px;

        top: 6px;

        content: "";

        width: 10px;

        height: 10px;

        border-right: 3px solid yellow;

        border-bottom: 3px solid yellow;

        transform: rotate(45deg);

      }

      span:hover::after {

        transform: rotate(225deg);

      }

    </style>

  </head>

  <body>

    <span></span>

  </body>

</html>

### （3）transform-origin設置旋轉中心點

語法：

transform-origin: x y;

注意：

① 參數x和y用空格隔開

② x y默認轉換的中心點是元素的中心點（50% 50%）

③ 可以給x和y設置%，xp，方位名詞（top bottom left right center）

案例：

鼠標放在div上，隱藏的div旋轉上來







代碼如下：

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      div {

        width: 100px;

        height: 100px;

        margin: 50px auto;

        background-color: coral;

        /\* overflow: hidden; \*/

      }

      div::after {

        content: "coding";

        display: block;

        width: 100%;

        height: 100%;

        background-color: yellow;

        color: #fff;

        transform-origin: left bottom;

        transform: rotate(180deg);

        transition: all 0.2s;

      }

      div:hover::after {

        transform: rotate(0);

      }

    </style>

  </head>

  <body>

    <div></div>

  </body>

</html>

### （4）scale縮放

transform: scale(x, y);

x和y是數字，不跟單位，表示的是倍數。1是原來的1倍，即沒有變化；2是原來的2倍；以此類推。

不會影響其它盒子，而且可以設置縮放的中心點（transform-origin）。

示例：

① 寬是原來的2倍，高不變

transform: scale(2, 1);

② 寬高都是原來的2倍，有2種寫法：

transform: scale(2, 2);

transform: scale(2);

③ 寬高都是原來的一半，有2種寫法：

transform: scale(0.5, 0.5);

transform: scale(0.5);

④ 寬高都是原來的1.2倍，有2種寫法：

transform: scale(1.2, 1.2);

transform: scale(1.2);

案例，img hover，img緩慢變為原來的1.2倍。代碼：

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      div {

        width: 300px;

        height: 168px;

        overflow: hidden;

      }

      img {

        width: 100%;

        height: 100%;

        transition: all 0.5s;

      }

      img:hover {

        transform: scale(1.2);

      }

    </style>

  </head>

  <body>

    <div><img src="./london.jpg" alt="" /></div>

  </body>

</html>

### （5）2D轉換綜合寫法

① 同時使用多個轉換，其格式為：transform: translate() rotate() scale() 等，用空格隔開。

② 順序會影響轉換的效果（比如，先旋轉會改變坐標軸方向）。

③ 同時有位移和其它屬性的時候，要將位移放到最前。

## CSS3的動畫animation

動畫可以實現更多變化、更多控制、連續自動播放等效果。

### （1）動畫的基本使用

製作動畫分為兩步：

① 先定義動畫

用keyframes定義動畫（類似定義類選擇器）

@keyframes 動畫名稱{

0%{

width: 100px;

}

100%{

width: 200px;

}

}

動畫序列

* 0%是動畫的開始，100%是動畫完成。等價於“from”和“to”。
* 期間可以設定任意多的變化，用百分比來規定發生的時間。比如25%，50%，75%。按照這些百分比來劃分動畫的運行時間animation-duration。

例子：

@keyframes move{

0%{

transform: translate(0, 0);

}

25%{

transform: translate(1000px, 0);

}

50%{

transform: translate(1000px, 500px);

}

75%{

transform: translate(0, 500px);

}

100%{

transform: translate(0, 0);

}

}

② 再使用（調用）動畫

比如某個div要使用

div{

width: 200px;

height: 200px;

background-color: aqua;

margin: 100px auto;

/\* 調用動畫 \*/

animation-name: 動畫名稱;

/\* 持續時間\*/

animation-duration: 持續時間（單位是s）;

}

### （2）常用的動畫屬性

|  |  |
| --- | --- |
| 屬性 | 描述 |
| @keyframes | 規定動畫 |
| animation | 所有動畫屬性的簡寫屬性，除了animation-play-state屬性 |
| animation-name | 規定@keyframes動畫的名稱。（必須） |
| animation-duration | 規定動畫完成一個週期所花費的秒或毫秒，默認值是0秒.（必須） |
| animation-timing function | 規定動畫的速度曲線，默認值是“ease”。 |
| animation-delay | 規定動畫何時開始，是否延時開始，默認值是0。 |
| animation-iteration-count | 規定動畫播放的次數，默認是值1，還有infinite。 |
| animation-direction | 規定動畫是否在下一週期逆向播放，默認值是“normal”。alternate逆向播放。 |
| animation-play-state | 規定動畫是否正在運行或暫停。默認值是“running”。還有值“paused”。 |
| animation-fill-mode | 規定動畫結束後狀態。停在終點值forwards。回到起始值backwards。 |

### （3）animation簡寫（重點）

語法：

animation: 動畫名稱 持續時間 運動曲線 何時開始（動畫開始延遲時間） 播放次數 是否反方向 保持在結束狀態或者回到起始點

* 其中：動畫名稱 持續時間 一定要寫
* 簡寫屬性裡面不包含animation-play-state
* 暫停動畫animation-play-state: paused; 經常和鼠標經過等配合使用

#### 案例應用：大數據地圖big data map 地點發送光暈

### （4）animation-timing function

|  |  |
| --- | --- |
| 值 | 描述 |
| linear | 動畫從頭到尾的速度是相同的。勻速。 |
| ease | 默認。動畫以低速開始，然後加快，在結束前變慢。 |
| ease-in | 動畫以低速開始。 |
| ease-out | 動畫以低速結束。 |
| ease-in-out | 動畫以低速開始和結束。 |
| steps() | 指定了時間函數中的間隔數量（步長）  下面的例子是div分成5步，从0宽度变成200px的宽度  div{  width: 0;  height: 30px;  background-color: pink;  animation: wider 4s steps(5) forwards;  }  @keyframes wider{  0% {  width: 0;  }  100%{  width: 200px;  }  } |

#### ① animation-timing function應用1：用steps做打字機的效果



<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      div {

        overflow: hidden;

        width: 0;

        height: 60px;

        background-color: lightseagreen;

        font-size: 40px;

        color: yellow;

        /\* 強制文字一行內顯示 \*/

        white-space: nowrap;

        animation: wider 4s steps(5) forwards;

      }

      @keyframes *wider* {

        0% {

          width: 0;

        }

        100% {

          width: 200px;

        }

      }

    </style>

  </head>

  <body>

    <div>掬水月在手</div>

  </body>

</html>

#### ② animation-timing function應用2：running bear奔跑的小熊

## CSS3的3D轉換

### （1）認識3D轉換

三維坐標系。

X軸，水平，右邊正值，左邊負值。→

Y軸，垂直，下面正直，上面負值。↓

Z軸，垂直屏幕，屏幕外面正值，屏幕裡面負值。

### （2）3D移動translate3d

3D移動在2D移動的基礎上多加了一個可以移動的方向，就是Z軸方向。

transform: translateX(100px); 僅在X軸上移動。

transform: translateY(100px); 僅在Y軸上移動。

transform: translateZ(100px); 僅在Z軸上移動。（注意：translateZ一般用px單位）

transform: translate3d(x, y, z); 其中x，y，z分別指要移動的軸的方向的距離。

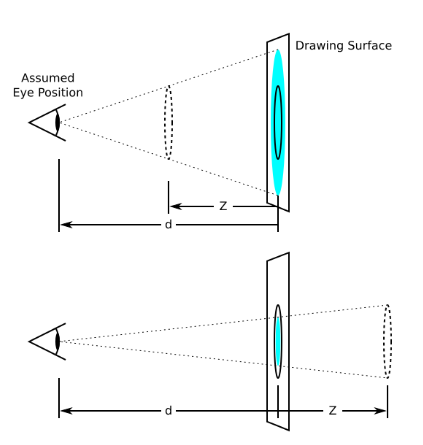
### （3）透視perspective

單位px

透視寫在被觀察元素的父盒子上，寫在祖盒子上也可以。下圖中

d：視距，人的眼睛到屏幕的距離。視距越小，眼睛距離屏幕越近，看到的物體就越大。

z：z軸，物體距離屏幕的距離，z軸越大（正直），我們看到的物體就越大。



例子： perspective: 300px;

### （4）3D旋轉

3D旋轉指可以讓元素在三維平面內沿著x軸，y軸，z軸或者自定義軸進行旋轉。

語法例子：

transform: rotateX(45deg) 沿著x軸正方向旋轉45度。

transform: rotateY(45deg) 沿著y軸正方向旋轉45度。

transform: rotateZ(45deg) 沿著z軸正方向旋轉45度。

transform: rotate3d(x, y, z, deg) 沿著自定義軸旋轉deg為角度（了解）。

transform: rotate3d(1, 0, 0, 45deg)，沿着x軸旋轉45deg

transform: rotate3d(1, 1, 0, 45deg)，沿著對角線旋轉45deg

左手準則

* 左手的手拇指指向 x/ y/ z轴的正方向
* 其余手指的弯曲方向就是该元素沿着x/ y/ z轴旋转的方向（正值）

或者

讓x/ y/ z軸箭頭對著自己，正值沿著順時針方向，負值沿著逆時針方向。

### （5）3D呈現transform-style（重要）

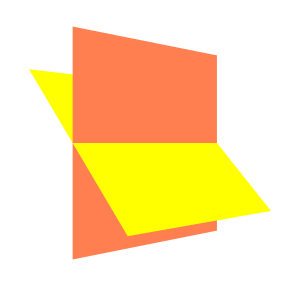
控制子元素是否開啟三維立體環境。

transform-style: flat; 子元素不開啟3d立體空間 默認

transform-style: preserve-3d; 子元素開啟立體空間

代碼寫給父級，影響的是子盒子。

例子，寫出這樣的樣式



代碼

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      body {

        perspective: 500px;

      }

      .box {

        position: relative;

        width: 200px;

        height: 200px;

        margin: 200px auto;

        transform: rotateY(45deg);

        transform-style: preserve-3d;

      }

      .box div {

        position: absolute;

        width: 100%;

        height: 100%;

        background-color: coral;

      }

      .box div:last-child {

        background-color: yellow;

        transform: rotateX(45deg);

      }

    </style>

  </head>

  <body>

    <div class="box">

      <div></div>

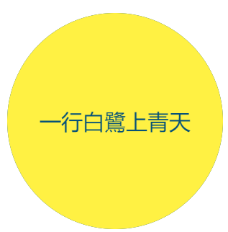
      <div></div>

    </div>

  </body>

</html>

### （6）案例：兩面翻轉盒子

正面：hover後翻到反面

hover盒子時把bottom翻到front



代碼

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>3D box burning over</title>

    <style>

      \* {

        margin: 0;

        padding: 0;

      }

      .turning-over-box {

        position: relative;

        width: 300px;

        height: 300px;

        margin: 100px auto;

        transition: all 0.6s;

        transform-style: preserve-3d;

      }

      .turning-over-box:hover {

        cursor: pointer;

        transform: rotateY(180deg);

      }

      .turning-over-box:hover .back {

        z-index: 2;

      }

      .front,

      .back {

        position: absolute;

        top: 0;

        left: 0;

        width: 100%;

        height: 100%;

        font-size: 30px;

        line-height: 300px;

        text-align: center;

        border-radius: 50%;

      }

      .front {

        background-color: #065279;

        color: #fff143;

        z-index: 1;

      }

      .back {

        background-color: #fff143;

        color: #065279;

        transform: rotateY(180deg);

      }

      .turning-over-boxes {

        margin: 100px;

      }

      .turning-over-boxes ul li {

        float: left;

        width: 130px;

        height: 36px;

        margin-right: 20px;

        list-style: none;

        perspective: 1000px;

      }

      .turning-over-box-item {

        position: relative;

        transform-style: preserve-3d;

        transition: all 0.4s;

      }

      .turning-over-box-item:hover {

        cursor: pointer;

        transform: rotateX(90deg);

      }

      .front-side,

      .bottom-side {

        position: absolute;

        top: 0;

        left: 0;

        width: 120px;

        height: 26px;

        padding: 5px;

        line-height: 26px;

        text-align: center;

      }

      .front-side {

        background-color: #065279;

        color: #fff143;

        z-index: 1;

        /\* 然後再把front盒子迎面移動一下，這樣可以保證旋轉的軸一直不變 \*/

        transform: translateZ(18px);

      }

      .bottom-side {

        background-color: #fff143;

        color: #065279;

        /\* 重要：必須先寫移動後寫旋轉 \*/

        /\* 先把bottom盒子移動下去 \*/

        transform: translateY(18px) rotateX(-90deg);

      }

    </style>

  </head>

  <body>

    <div class="turning-over-box">

      <div class="front">兩個黃鸝鳴翠柳</div>

      <div class="back">一行白鷺上青天</div>

    </div>

    <div class="turning-over-boxes">

      <ul>

        <li>

          <div class="turning-over-box-item">

            <div class="front-side">水晶簾動微風起</div>

            <div class="bottom-side">滿架薔薇一院香</div>

          </div>

        </li>

        <li>

          <div class="turning-over-box-item">

            <div class="front-side">稻花香裡說豐年</div>

            <div class="bottom-side">聽取蛙聲一片</div>

          </div>

        </li>

        <li>

          <div class="turning-over-box-item">

            <div class="front-side">綠樹蔭濃夏日長</div>

            <div class="bottom-side">樓台倒影入池塘</div>

          </div>

        </li>

        <li>

          <div class="turning-over-box-item">

            <div class="front-side">梅子留酸軟齒牙</div>

            <div class="bottom-side">芭蕉分綠與窗紗</div>

          </div>

        </li>

        <li>

          <div class="turning-over-box-item">

            <div class="front-side">晴日暖風生麥氣</div>

            <div class="bottom-side">綠蔭幽草勝花時</div>

          </div>

        </li>

      </ul>

    </div>

  </body>

</html>

## CSS3 animation和3D轉換案例：旋轉木馬 carousel



## 瀏覽器私有前綴

瀏覽器私有前綴是為了兼容老版本的寫法，比較新版本的瀏覽器無需添加。

私有前綴

-moz- Firefox瀏覽器私有屬性

-ms- ie瀏覽器私有屬性

-webkit- safari、Chrome私有屬性

-o- Opera私有屬性

例如border-radius有兼容性問題，提倡這樣寫：

-moz-border-radius: 10px;

-webkit-border-radius: 10px;

-o-border-radius: 10px;

border-radius: 10px;

ie瀏覽器的可以不用寫，因為微軟已逐漸不再用這款瀏覽器了。

CSS3新特性

新增的CSS3特性有兼容性問題，ie9+才支持

移動端支持優於PC端

不斷改進中

應用相對廣泛

現階段主要學習：新增選擇器，盒子模型，其它特性。

## CSS3新增選擇器

### （1）屬性選擇器

屬性原則器根據元素特定屬性來選擇元素，這樣可以不用藉助於類或者id選擇器。

屬性選擇器，類選擇器，偽類選擇器，權重為10.

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>CSS3新增屬性選擇器</title>

    <style>

      /\* 1. 必須是input，同時具有value這個屬性 \*/

      input[value] {

        color: pink;

      }

    </style>

  </head>

  <body>

    <!-- 1. 利用屬性選擇器可以不用藉助於類或者id選擇器 -->

    <input type="text" value="請輸入用戶名" />

    <input type="text" />

  </body>

</html>

下面這個是重點

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>CSS3新增屬性選擇器</title>

    <style>

      /\* 2. 必須是input，同時value屬性的值為text \*/

      input[type="text"] {

        color: pink;

      }

    </style>

  </head>

  <body>

    <!-- 2. 屬性選擇器還可以選擇屬性=值的某些元素 -->

    <input type="text" /><input type="password" />

  </body>

</html>

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>CSS3新增屬性選擇器</title>

    <style>

      /\* 把class='icon'開頭的全部選出來 \*/

      /\* 首先是div，然後具有class屬性，並且屬性值必須是icon開頭的這些元素 \*/

      div[class^="icon"] {

        color: pink;

      }

    </style>

  </head>

  <body>

    <!-- 3. 屬性選擇器可以選擇屬性值開頭的某些元素 -->

    <div class="icon1">小圖標1</div>

    <div class="icon2">小圖標2</div>

    <div class="icon3">小圖標3</div>

    <div class="icon4">小圖標4</div>

    <div>醬油div</div>

  </body>

</html>

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>CSS3新增屬性選擇器</title>

<style>

/\*4. 屬性選擇器可以選擇屬性值結尾的某些元素 \*/

      section[class$="data"] {

        color: blue;

      }

      /\*5.  只要class這個屬性裡，有ico這個值，就都選出來 \*/

      section[class\*="ico"] {

        font-weight: 700;

      }

    </style>

  </head>

  <body>

    <section class="icon1-data">清風不識字</section>

    <section class="icon2-data">何故亂翻書</section>

    <section class="icon3-ico">故國有明</section>

  </body>

</html>

### （2）結構偽類選擇器

結構偽類選擇器屬於偽類選擇器。

屬性選擇器，類選擇器，偽類選擇器，權重為10。

常用於根據父級選擇子元素

|  |  |
| --- | --- |
| 選擇符 | 簡介 |
| E:first-child | 匹配父元素中的第一個子元素 |
| E:last-child | 匹配父元素中最後一個子元素 |
| E:nth-child(n) | 匹配父元素中的第n個子元素   * n可以是數字，從1開始計數。 * n可以是even（偶數），odd（奇數） * n可以是公式：常見的公式如下（如果n是公式，則從0開始計數。但是第0個元素或者超出了元素的個數會被忽略。）   + 比如 ol li:nth-child(n) {background-color: pink;}把所有的子元素都選出來了。   + 2n 偶數   + 2n+1 奇數   + 5n 5 10 15……   + n+5 從第5個開始（包含第5個）到最後   + -n+5 前5個（包含第5個） |
| E:first-of-type | 指定類型元素的第1個 |
| E:last-of-type | 指定類型元素的最後一個 |
| E:nth-of-type(n) | 指定類型元素的第n個 |

例子：

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      /\* 1. 選擇ul裡面的第1個是li的孩子 \*/

      ul li:first-child {

        background-color: pink;

      }

      /\* 2. 選擇ul裡面第3個是li的孩子 \*/

      ul li:nth-child(3) {

        background-color: skyblue;

      }

    </style>

  </head>

  <body>

    <ul>

      <li>第1個li</li>

      <li>第2個li</li>

      <li>第3個li</li>

      <li>第4個li</li>

      <li>第5個li</li>

      <li>第6個li</li>

      <li>第7個li</li>

      <li>第8個li</li>

    </ul>

  </body>

</html>

#### nth-child和nth-of-type的區別

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      /\* nth-child會把所有的子元素都排列序號 \*/

      section div:nth-child(1) {

        /\* 會先匹配:nth-child(1)，看後再匹配div。發現沒有這樣的div，所以無從修改樣式。 \*/

        background-color: pink;

      }

      /\* nth-of-type會把指定元素的盒子排列序號 \*/

      /\* 這個例子中，指定的元素盒子是div。這裡會先找section中的所有div，然後把第1個div選出來。 \*/

      section div:nth-of-type(1) {

        background-color: skyblue;

      }

    </style>

  </head>

  <body>

    <section>

      <p>醉眠海棠</p>

      <div>天朗氣清</div>

      <div>惠風和暢</div>

    </section>

  </body>

</html>

### （3）偽元素選擇器

偽元素選擇器：利用CSS創建新標籤元素，而不需要HTML標籤，從而簡化HTML結構。

::before 在元素內部的前面插入內容

::after 在元素內部的後面插入內容

before和after創建一個元素，屬於行內元素。

新創建的元素在文檔樹中不存在，所以稱為偽元素。

語法： element::before{}

before和after必須有content屬性

偽元素選擇器和標籤選擇器一樣，權重為1.

給class=”tudou”這個div，用::before做了個遮罩層。當鼠標經過class=”tudou”這個div時，讓裡面的遮罩層顯示出來，語法如下（字符之間不要有空格）：

.tudou:hover::before{

display: block;

}

## CSS3盒子模型

CSS3中可以通過box-sizing來制定盒模型，有2個值：content-box，border-box。

默認情況 box-sizing: content-box （盒子大小為width+padding+border）

box-sizing: border-box 盒子大小就是with和height（哪怕後來padding和border變化）

## CSS3其它特性（了解）

### （1）圖片變模糊

filter: 函數();

例如：filter:blur(5px); blur是個函數，小括號裡數值越大，圖片越模糊，注意數值要加px單位。

例如

img{

filter: blur(5px);

}

### （2）計算盒子寬度width: calc函數

例如 width:calc(100%\*80px); 加減乘除運算符號分別是+-\*/

應用，讓子盒子永遠比父盒子小30px

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      .father {

        width: 300px;

        height: 200px;

        background-color: pink;

      }

      .son {

        /\* 子盒子永遠比父盒子小30px \*/

        width: calc(100% - 30px);

        height: 30px;

        background-color: skyblue;

      }

    </style>

  </head>

  <body>

    <div class="father">

      <div class="son"></div>

    </div>

  </body>

</html>

## CSS3過度（重點）

過度（transition）是CSS3中具有顛覆性的特征之一，我們可以在不使用Flash動畫或JavaScript的情況下，當元素從一種樣式變換為另一種樣式時為元素添加效果。

低版本瀏覽器（ie9以下版本）不支持，但是不會影響頁面佈局。

經常和:hover搭配使用。

語法

transition: 要過度的屬性 花費時間 運動曲線 何時開始;

解釋：

* 要過度的屬性：想要變化的css屬性，寬度、高度、背景顏色、內外邊距都可以。如果想要所有的屬性都變化過度，寫一個all就可以。
* 花費時間：單位是秒（必須寫單位），比如0.5s
* 運動曲線：默認是ease（可以省略）
* 何時開始：單位是秒（必須寫單位），可以設置延遲觸發時間，默認是0s（可以省略）

例子，在hover時，想同時給個div改寬和高。要給哪個元素做過度效果，就把transition加給那個元素。

div{

width: 200px;

height: 100px;

background-color: pink;

transition: width .5s, height .5s; （如果同時修改多個屬性，用逗號隔開）

}

div:hover{

width: 400px;

height: 200px;

}

### CSS過度練習：進度條

要點：①如何佈局進度條；②使用transition效果



代碼如下

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Document</title>

    <style>

      .bar {

        width: 150px;

        height: 15px;

        border: 1px solid red;

        border-radius: 7px;

        padding: 1px;

      }

      .bar\_in {

        width: 50%;

        height: 100%;

        background-color: red;

        transition: all 0.7s;

      }

      /\* 鼠標經過父盒子時，讓子盒子的寬度和父盒子一樣 \*/

      .bar:hover .bar\_in {

        width: 100%;

      }

    </style>

  </head>

  <body>

    <div class="bar">

      <div class="bar\_in"></div>

    </div>

  </body>

</html>

## 廣義H5（了解）

廣義的HTML5是HTML5本身+CSS3+JavaScript。

雖然HTML5的一些特性不被某些瀏覽器支持，但是它是一種發展趨勢。

切圖

## 常見圖片格式

* jpg（=JPEG）對色彩的信息保留較好，高清，顏色較多，產品類的圖片經常用jpg格式。
* gif 最多只能存儲256色，所以通常用來顯示簡單圖形及字體。可以保存透明背景和動畫效果，實際經常用於一些圖片小動畫效果。
* png 新興格式，結合了jpg和gif的優點，能夠保存透明背景。如果要切成背景透明的圖片，用png格式。
* psd 是Photoshop專用格式

PS常見貼圖方式：圖層切圖，切片切圖，PS插件切圖。

## 圖層切圖

右擊圖層 → 快速導出為PNG

同時導出2個圖層：

按住shift鍵，把2個圖層都選中→右擊圖層 → 快速導出為PNG

## 切片切圖

選擇切片工具選中圖片→文件菜單→導出→存儲為web設備所用格式→選擇圖片格式→存儲→選擇“選中的切片”來保存

如果要切透明背景的圖，先把PS圖層中的“背景”前面的小眼睛去掉（圖層不要包含背景），然後再開始上面的過程。

## Cutterman插件切圖

安裝cutterman時，PS必須是完整版，而非綠色版。查看PS是否為完整版：看窗口→拓展功能是否正常

打開cutterman：窗口→擴展功能→cutterman 切圖神器

可以導出單個圖層，也可以導出多個圖層，還可以導出選中的切片

工具

## PS

打開標尺：Ctrl+R或者 選擇 視圖→標尺。

右擊標尺，把標尺單位改為像素。

Ctrl+加號（+）放大視圖，Ctrl+減號（-）縮小視圖。

在右邊工具欄選擇矩形選框工具，可以測量大小。

Ctrl+D可以取消選區，或者在旁邊空白處點擊一下，也可以取消選區。

吸管工具可以取色。

## Snipaste

① F1截圖，測量大小，在圖上做標記，書寫文字。

② F3在桌面置頂顯示。

③ 點擊圖片，alt可以取色（按下shift可以切換取色模式）。

④ 按下esc取消圖片顯示。

項目——學成網

## 佈局

1. 確定頁面版心（可視區）。

2. 確定行模塊，以及每個行模塊中的列模塊。

列模塊經常浮動佈局。先確定每個列的大小，之後確定列的位置。

3. 製作HTML結構。

4. 運用盒子模型原理，CSS美化頁面。

## header要點



寫3（首頁、課程、職業規劃）的時候，在實際開發中，不會直接用a，而是用li包含a的做法，原因有二：

① li+a語義更清晰，一看就是有條理的列表型內容。

② 如果直接用a，容易被搜索引擎判定為堆砌關鍵字（搜索引擎對鏈接很敏感），進而被搜索引擎降低被搜索到的權重，進而影響網站排名。

## banner要點

做banner左邊的課程菜單（subnav）時遇到一個疑問，答案在於【嵌套塊元素垂直外邊距的塌陷】和【相鄰塊元素垂直外邊距的合併】



.banner .subnav li {

  margin: 30px 0 0 22px;

  line-height: 14px;

  font-size: 14px;

}

.header {

  height: 42px;

  margin: 30px auto; /\*此地层叠w里面的margin\*/

}

疑問：去掉subnav的左浮動時，第1個li的上邊距直接頂在了header中學成在線logo的下面。

原因：

①【嵌套塊元素垂直外邊距的塌陷】3次：li的上邊距30px變成了ul的，然後變成了w的，然後變成了banner的。

②【相鄰塊元素垂直外邊距的合併】：header下邊距是30px，banner上邊距是30px，合併成了30px

## 精品推薦要點

這裡10個圖是10個li。每個li會有個右邊距，除了最右邊的那些li右邊距為0。寫樣式的時候，給每個li加個margin-right: 15px。不必把最右邊的margin-right設為0。可以把ul寫寬些，寫得超出版心的寬度。ul的寬度=（li的寬度+margin-right的值）X每一行li的個數。

即：（228px+15px）\*5=1215px。



項目——品優購

## 品優購項目規劃

### （1）網站製作流程



其中預付定金是總金額的30%左右。

後期維護：半年到一年的時間。

### （2）品優購項目整體介紹

品優購是個電商網站，要完成PC端首頁、列表頁、註冊頁面的製作。

品優購項目學習目的：電商類網站比較綜合，裡面需要大量的佈局技術，包括佈局方式、常見效果以及周邊技術。寫完品優購項目，能對實際開發中製作PC端頁面流程有一個整體的感知。

開發工具：VSCode，PS，Chrome

技術棧：利用HTML5+CSS3手動佈局，可以大量使用H5新增標籤和樣式。

結構與樣式分離，模塊化開發。

## 品優購項目搭建

以前初始化過一個京東的css項目，可以直接拷貝過來放到項目裡。

有些網站初始化不太提倡\* { margin: 0; padding: 0; }

比如新浪用的是：

html,body,ul,li,ol,dl,dd,dt,p,h1,h2,h3,h4,h5,h6,form,fieldset,legend,img{margin:0;padding:0}

直接用星號，會把所有的標籤都渲染一遍，對於大型網站來說，這樣太耗費資源。

## 模塊化開發樣式

將一個項目按照功能劃分，一個功能一個模塊，互不影響。

模塊化開發優點：重複使用，更換方便。

類比，手機由不同部件組成。生產手機時工人各自負責各自的組件，同時進行。維修時只需要更換壞掉的部件。

有些樣式和結構在很多頁面都會出現，比如頁面頭部和底部，大部分頁面都有。可以把這些結構和樣式單獨作為一個模塊，然後重複使用。

典型應用：common.css公共樣式，包含版心寬度，清除浮動，頁面文字顏色等公共樣式。

## 網站TDK三大標籤SEO優化

SEO（Search Engine Optimization）搜索引擎優化。利用搜索引擎的規則提高網站在有關搜索引擎內自然排名。

頁面必須有3個標籤來符合SEO優化。這3個標籤時：title，description，keyword。

### （1）title

title是搜索引擎了解網頁的入口，搜索引擎通過title判斷網頁類別。

建議：網站名（產品名）-網站的介紹（盡量精煉，不要超過30個漢字）

例子：

京東（JD.COM）-綜合網購首選-正品低價、品質保障、配送及時、輕鬆購物！

小米商城-小米5s、紅米Note4、小米MIX、小米筆記本官方網站

### （2）description

簡要說明網站主要是做什麼的

description作為網站的總體業務和主題概括，多採用“我們是……”，“我們提供……”，“XXX網作為……”，“電話：010……”之類的語句。

比如京東的：

<meta name="description" content="京东JD.COM-专业的综合网上购物商城,销售家电、数码通讯、电脑、家居百货、服装服饰、母婴、图书、食品等数万个品牌优质商品.便捷、诚信的服务，为您提供愉悦的网上购物体验!">

上述文字是由負責SEO的員工去寫，前端準備好標籤，然後把相關語句裝進標籤即可。

### （3）keywords

keywords最好限制為6-8個關鍵詞，關鍵詞之間用英文逗號隔開。

京東的例子：

<meta name="Keywords" content="网上购物,网上商城,手机,笔记本,电脑,MP3,CD,VCD,DV,相机,数码,配件,手表,存储卡,京东">

## 網站favicon圖標

favicon.ico一般用於作為縮略的網站標誌，它顯示在瀏覽器的地址欄或者標籤上。

主要瀏覽器都支持favicon.ico圖標。

步驟：

① 製作favicon圖標

a. 把優品購圖標切成png圖片

b. 把png圖片轉換為ico圖標，藉助第三方網站，例如<http://www.bitbug.net/>

② 把favicon圖標放到網站根目錄下

③ HTML頁面引入favicon圖標

在HTML頁面的<head></head>元素之間、title下面引入代碼：

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon"/>

## 常用模塊類名命名



## Header-LOGO SEO優化

① logo裡面首先放一個h1標籤，告訴搜索引擎，這個地方很重要。

② h1裡面再放一個連接，可以返回首頁，連接的背景用logo圖片即可。

③ 為了搜索引擎收錄我們，鏈接裡要放文字（網站名稱），但是文字不要顯示出來。

* 方法a：text-indent移到盒子外面（text-indent:-9999px），然後overflow:hidden，淘寶的做法。
* 方法b：直接給font-size:0; 就看不到文字了，京東的做法。

④ 最後給連接一個title屬性，這樣鼠標放到logo上就可以看到提示文字了。

## Header-購物車計數 設置技巧

* count統計部分用絕對定位做。
* count統計部分不要給寬度，因為可能買的件數比較多，個位數、十位數寬度不一樣。讓件數自己撐開寬度。如果覺得左右太窄，就給個左右的padding值。只給一個高度。
* 一定注意左下角不是圓角，其餘三個是圓角。寫法：border-radius: 7px 7px 7px 0;

## 圖片水平居中，垂直居中

在Homepage-main-likes（猜你喜歡模塊），讓圖片垂直居中和水平居中，用的是給圖片加絕對定位。

HTML代碼

<div class="likes-bd-img">

   <img src="products/main-likes1.png" alt="main-likes1" />

</div>

CSS代碼

.likes-bd-img {

  position: relative;

  width: 100%;

  height: 145px;

}

.likes-bd-img img {

  position: absolute;

  top: 50%;

  left: 50%;

  transform: translate(-50%, -50%);

}

因為不知道圖片自身的寬高（到時候從數據庫裡隨機拿取不同產品圖片），所以用在用top50%和left50%把圖片左上角定在父盒子中間時，用transform進行調整。

## 主頁——電梯區，樓層區

### （1）樓層區（類名floors）不要給高度，讓內容自行撐開。

### （2）Tab欄原理

準備2個盒子：一個類名叫tab\_list，放選項卡；一個類名叫tab\_content，放待顯示的內容。選項卡個數和內容盒子個數要一致。

比如，點擊選項卡2時，內容盒子2顯示，其它內容盒子隱藏。



## 註冊頁

register.html

注意：註冊頁面比較隱私，為了保護用戶信息，不需要對當前頁面做SEO優化。

所以在header部分，不需要title，description，keywords。寫個標題即可。

## 詳情頁

## 讓盒子裡的文字兩端對齊

比如讓span裡的文字兩端對齊

.price span {

  display: inline-block;（先變成行內塊，好設置寬度）

  width: 48px;

  text-align: justify;

  text-align-last: justify;

}

寫出來的樣子：价 格

服務器

（1）什麼是Web服務器

服務器（也稱主機）是提供計算服務的設備，本質是一台計算機。在網絡環境下，根據服務器提供的服務類型不同，服務器又分為文件服務器，數據庫服務器，應用程序服務器，Web服務器等。

（2）遠程服務器

本地服務器主要在局域網中訪問。

遠程服務器通常是別的公司為我們提供的一台電腦（主機），只要把網站目錄上傳到這台電腦上，任何人都可以利用域名訪問我們的網站。

（3）把自己的網站上傳到遠程服務器

步驟：

①註冊賬號，記下主機名，用戶名，密碼，域名

免費的遠程服務器（免費空間）<http://free.3v.do/>

②利用cutftp軟件，上傳網站到遠程服務器。

移動Web開發

## 移動端基礎

### （1）瀏覽器現狀

PC端常見瀏覽器：Chrome，Firefox，QQ，360，百度，搜狗，IE

移動端常見瀏覽器：Chrome，UC，QQ，歐朋，搜狗，百度，360，獵豹

國內的主流手機瀏覽器都是根據Webkit修改過來的內核，國內尚無自主研發的內核。就像國內的手機操作系統都是基於Android修改開發的一樣。為國內客戶開發移動端網站，兼容移動端主流瀏覽器，處理webkit內核瀏覽器即可。

### （2）手機屏幕現狀

移動端設備屏幕尺寸非常多。開發者無需關注這些分辨率，因為我們常用的尺寸單位是px。

### （3）移動端調試方法

ChromeDevTools 模擬手機測試

搭建本地web服務器，手機和服務器一個局域網內，通過手機訪問服務器

使用外網服務器，直接IP或域名訪問

## 視口

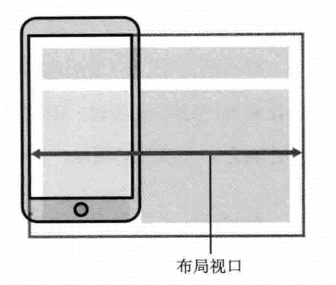
視口（viewport）是瀏覽器顯示頁面內容的屏幕區域。

視口可以分為佈局視口、視覺視口和理想視口。

### （1）佈局視口layout viewport

一般移動設備的瀏覽器都默認設置了一個佈局視口，用於解決早期的PC端頁面在手機上顯示的問題。

IOS，Android基本都將這個視口分辨率設置為980px，所以PC上的網頁大多都能在手機上呈現，只不過元素看上去很小，一般默認可以通過手動縮放網頁。



### （2）視覺視口 visual viewport

用戶正在看到的網站的區域。

用戶可以通過縮放去操作視覺視口，同時不會影響佈局視口，佈局視口仍然保持原來的寬度。

### （3）理想視口 ideal viewport

為了使網站在移動端有最理想的瀏覽和閱讀寬度而設定。

理想視口，對設備來講，是最理想的視口尺寸。

需要手動填寫meta視口標籤通知瀏覽器操作。

meta視口標籤的主要目的：佈局視口的寬度應該與理想視口的寬度一致，即設備有多寬，佈局視口就多寬。

### （4）meta視口標籤

<meta name=”viewport” content=”width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0”>

|  |  |
| --- | --- |
| 屬性 | 解釋說明 |
| width | 寬度。設置的是viewport寬度。可以設置device-width特殊值。 |
| initial-scale | 初始縮放比，大於0的數字 |
| maximum-scale | 最大縮放比，大於0的數字 |
| minimum-scale | 最小縮放比，大於0的數字 |
| user-scalable | 用戶是否可以縮放，yes或no（1或0） |

## 物理像素與物理像素比

物理像素點指的是屏幕顯示的最小顆粒，是設備出廠時就設置好了。比如iPhone11是1792x828的分辨率，指的是屏幕一列可以放1792個像素點，一行可以放828個像素點。

1px不一定等於1個物理像素點。PC端頁面，1px=1個物理像素。移動端數據各式各樣。比如iPhone8裡，1px開發像素=2個物理像素。

早期的手機屏幕1px=1物理像素。後來出現Retina（視網膜屏幕）（摩托羅拉開發，Apple發揚光大），這是一種顯示技術，可以把更多的物理像素壓縮到一塊屏幕裡，從而得到更高的分辨率，提高屏幕顯示的細膩程度。



1px能顯示的物理像素點的個數，稱為物理像素比或屏幕像素比。

## 圖片的縮放

### （1）二倍圖

問題例子：如果直接把50\*50的css px像素的圖片放到iPhone8裡會放大2倍，圖片會變得模糊。

解決方案：使用100\*100css px像素的圖片，寫css代碼的時候，把這個圖片寫成50\*50大小。

為移動端準備的圖片大小，是實際需要圖片的2倍，這種方式成為二倍圖。

### （2）多倍圖

有時會使用三倍圖、四倍圖，實際開發時看公司要求。

### （3）背景的縮放

background-size屬性規定背景圖像的尺寸

語法：

background-size: 背景圖片寬度 背景圖片高度;

例子：

background-size: 500px 300px;

只寫1個參數，那是寬度，系統默認省略高度，會等比例縮放。

background-size: 長度| 百分比| cover| contain;

長度單位px。

或者寫%。相對於父盒子，比如寬度30%，指的是寬度是父盒子寬度的30%。

cover: 把背景圖片**等比例**擴展至足夠大，以使背景圖片完全覆蓋背景區域。（很多情況下，背景圖片會溢出。所以只能看到部分圖片。）



contain: 把背景圖片等比例擴展至最大尺寸，以使其寬度或者高度完全適應內容區域。（只要有寬或者高擴展到最大尺寸了，就不再繼續縮放，所以很多情況下，背景圖片不會完全覆蓋盒子。）



### （4）用PS插件cutterman切出多倍圖

選ios裡面的@3X，@2X，@1X

## 移動端開發選擇

### （1）移動端主流方案

#### ① 單獨製作移動端頁面（主流）

通常情況下，網址域名前面加m（mobile）可以打開移動端。通過判斷設備，如果是移動設備打開，則調到移動端頁面。

例子：

京東商城手機版 m.jd.com

淘寶觸屏版 m.taobao.com

蘇寧易購手機版 m.suning.com

#### ② 響應式頁面兼容移動端（其次）

三星手機官網 [www.samsung.com/cn/](http://www.samsung.com/cn/) 通過判斷屏幕寬度來改變樣式，可以適應不同終端，這是響應式網頁。

缺點：製作麻煩，需要花很大精力去不斷調整，以解決兼容性問題。

### （2）移動端技術解決方案

#### ① 兼容性

移動端瀏覽器（Safari，UC，QQ，百度）基本以webkit內核為主，因此考慮webkit兼容性問題即可。

瀏覽器私有前綴只需要添加webkit即可。

可以放心使用H5標籤和CSS3樣式。

#### ② 移動端初始化

移動端CSS初始化推薦使用normalize.css

優點：

保護了有價值的默認值

修復了瀏覽器的bug

模塊化

文檔詳細

官網地址：<http://necolas.github.io/normalize.css/>

### （3）CSS3盒子模型box-sizing

傳統模式寬度計算：盒子的寬度=CSS中設置的width + border + padding。

CSS3盒子模型：盒子的寬度=CSS中設置的寬度width（裡面包含了border和padding）。

CSS3中的盒子模型，padding和border不會撐大盒子了。

/\*CSS3盒子模型\*/

box-sizing: border-box; （這句話可以讓盒子變成CSS3盒子模型）

/\*傳統盒子模型\*/

box-sizing: content-box;

移動端可以全部用CSS3盒子模型。

PC端如果需要完全兼容，就用傳統盒子模型；如果不考慮兼容性，用CSS3盒子模型。

### （4）移動端特殊樣式

/\*CSS3盒子模型\*/

box-sizing: border-box;

-wbkit-box-sizing: border-box;

/\*清除點擊（比如連接的）高亮，設置為transparent 透明\*/

-webkit-tap-highlight-color: transparent;

/\*去掉ios端默認的屬性，然後才能自定義按鈕和輸入框樣式\*/

-webkit-appearance: none;

/\*禁用長按頁面時的彈出菜單\*/

img, a { -webkit-touch-callout: none; }

## 移動端常見佈局

① 單獨製作移動端頁面（主流）

* 流式佈局（百分比佈局）（京東）
* flex彈性佈局（攜程網）（推薦）
* less+rem+媒體查詢佈局（蘇寧）（推薦）
* 混合佈局

② 響應式頁面兼容移動端（其次）

* 媒體查詢
* bootstrap

### （1）流式佈局（百分比佈局）

流式佈局，即百分比佈局，也稱非固定像素佈局。

把盒子的寬度設置成百分比，根據屏幕的寬度進行伸縮，不受固定像素的限制，內容向兩側填充。

max-width 最大寬度

max-height 最大高度

min-width 最小寬度（一般最小寬度定為320px）

min-height 最小高度

#### 案例：京東

方案：單獨製作移動頁面

技術：流式佈局

##### 二倍精靈圖做法

a. 在Fireworks/ Photoshop裡把精靈圖等比例縮放為原來的一半

b. 測量目標圖片的坐標

c. background-size要寫精靈圖原寬度高度的一半

##### 圖片格式

DPG圖片壓縮技術

京東研發DPG圖片壓縮技術，可節省用戶近50%的瀏覽流量，極大地提升網頁打開速度，能夠兼容jpeg，實現全平台、全瀏覽器兼容，壓縮後的圖片和webp的清晰度無差距。

webp圖片格式

谷歌開發，用於加快圖片加載速度，圖片壓縮體積大約是JPEG的2/3，能節省大量的服務器寬帶資源和數據空間。

### （2）flex佈局（推薦）

傳統佈局

* 兼容性好
* 佈局繁瑣
* 局限性，不能在移動端很好地佈局

flex彈性佈局

* 操作方便，佈局極為簡單，在移動端廣泛應用
* PC段瀏覽器支持情況差：比如IE11或更低版本，不支持或僅部分支持

導師建議：

* 如果是PC端頁面佈局，用傳統佈局。
* 如果是移動端或者不考慮兼容性問題的PC端頁面佈局，用flex彈性佈局。

#### ① flex佈局原理

flex佈局原理：通過給父盒子添加flex屬性，來控制子盒子的位置和排列方式。

flex是flexible box的縮寫，意為“彈性佈局”。任何一個容器都可以被指定為flex佈局。

把父盒子設為flex佈局後，子元素的float，clear和vertical-align屬性都失效。

flex佈局別稱：伸縮佈局，彈性佈局，伸縮盒佈局，彈性盒佈局。

採用flex佈局的元素，成為flex容器（flex container），簡稱“容器”。它的所有子元素自動成為容器成員，稱為flex項目（flex item），簡稱“項目”。項目可以橫向排列，也可以縱向排列。

#### ② flex佈局父項常見屬性

* flex-direction：設置主軸方向

默認主軸方向是x軸方向，水平向右；默認側軸方向是y軸方向，垂直向下。

flex-direction: row 默認值，從左到右；

flex-direction: row-reverse 從右到左；

flex-direction: column 從上到下；

flex-direction: column-reverse 從下到上。

* justify-content：設置主軸上的子元素排列方式

justify-content: flex-start 默認值從頭部開始，如果主軸是x軸，從左到右；

justify-content: flex-end 從尾部開始排列；

justify-content: center 在主軸居中對齊；如果主軸是x軸，則水平居中；

justify-content: space-around 平分剩餘空間；

justify-content: space-between 先兩邊貼邊，再平分剩餘空間（重要）

* flex-wrap：設置子元素是否換行。flex佈局中，默認子元素不換行，如果裝不下，會縮小子元素寬度。

flex-wrap: nowrap 默認值，不換行；

flex-wrap: wrap 換行。

* align-items：設置側軸上的子元素的排列方式（單行）

align-items: flex-start 默認值 從上到下

align-items: flex-end 從下到上

align-items: center 擠在一起垂直居中

align-items: stretch 把每個子盒子都拉伸得和父盒子一樣高（子盒子不要給高度，否則沒效果。）

* align-content：設置側軸上的子元素的排列方式（多行，單行情況沒效果）

align-content: flex-start 默認值在側軸的頭部開始排列

align-content: flex-end 在側軸的尾部開始排列

align-content: center 在側軸中間顯示

align-content: space-around 子項在側軸平分剩餘空間

align-content: space-between 子項在側軸先分佈在兩頭，再平分剩餘空間

align-content: stretch 設置子項元素高度平分父元素高度

* flex-flow：復合屬性，相當於同時設置了flex-direction和flex-wrap

例子：flex-flow: column wrap; 把主軸變成y軸並且換行

#### ③ flex佈局子項常見屬性

* flex子項目佔的份數（最重要）

flex屬性定義子項目分配剩餘空間，用flex來表示占多少份數

常見的是flex:1;

還可以寫百分比flex:20%; （這個20%指的是子盒子佔據父盒子寬度的20%）

比如：

ul裡面10個li，分2行排列。因為不方便給li設置具體的寬度，所以即便給ul設置了flex-wrap:wrap，這些li也不會自動換行還是會一行排列。這個時候，要給li設置flex:20%;，表示5個li一共佔用100%（即一行）的寬度。

* align-self控制某一個子項目自己在側軸的排列方式，可以覆蓋align-items屬性。默認值為auto，表示繼承父元素的align-items屬性，如果沒有父元素，則等同於stretch。

例子：

span:nth-child(2) {

/\*設置自己在側軸上的排列方式\*/

align-self: flex-end;

}

* order屬性定義子項目的排列順序（前後順序）。數值越小，排列越靠前，默認為0。

例子，把第2個子盒子放在第1個子盒子之前：

div span:nth-child(2){

order: -1;

}

#### ④ 背景線性漸變

語法：

background: -webkit-linear-linear-gradient (起始方向，顏色1，顏色2, …);

可以寫很多個顏色

例子：

background: -webkit-linear-gradient(left, red , blue);  
background: -webkit-linear-gradient(left top, red , blue);

背景漸變必須添加瀏覽器私有前綴，在移動端只要寫個-webkit-linear-就夠用了。

起始方向可以是：方位名詞 或者 度數；如果省略，默認是top

#### ⑤ flex: grow shrink basis

flex: grow shrink basis可以單獨設置flex-grow（默認值0），flex-shrink（默認值1）和flex-basis（默認值auto）

* flex-grow決定子盒子如何瓜分父盒子的主軸的剩餘空間。

假設父盒子.container的width=500px，主軸是默認的水平從左向右。剩餘空間200px=.container.width（500px）-B1.width（100px） -B2.width（100px） -B3.width（100px）

flex-grow默認值為0，不瓜分剩餘空間。如果設置B1的flex-grow: 1，B1會分走所有的剩餘空間200px。如果設置B1的flex-grow: 1，並且設置B2的flex-grow: 2，此時剩餘空間200px會被分成3份，1份給B1，2份給B2。

* flex-basis是width或者height的代替品（取決於主軸的方向）。

如果子容器設置了flex-basis或者width/height，在分配空間之前，他們會跟父容器預約相應的空間，剩下的歸入剩餘空間。

如果同時設置flex-basis和width/height，width/height屬性會被覆蓋，即flex-basis優先級高於width/height。

如果flex-basis和width/height其中一個是auto，非auto的屬性優先級更高。

如果flex-basis和width/height都是auto，空間大小由內容決定。

* flex-shrink決定子盒子如何縮減體積

假設父盒子.container.width:500px，B1.width（300px），B2.width（120px），B3.width（160px）。子盒子寬度比父盒子寬80px。

由於flex默認不分行/列，所有盒子會在1行/列，就註定了子盒子會縮小體積，以適應父盒子的大小。

默認的縮小比例假設為x，x通過這個方程式求得500=300x+120x+160x。

如果這時設置B1的flex-shrink為2，那麼方程式就是這樣的500=300\*2x+120x+160x，B1縮小的比例會更多。

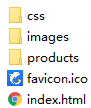
#### flex佈局案例：攜程網移動端佈局

第①步 技術選型

方案： 單獨製作移動頁面

技術： 採用flex佈局為主，其它技術為輔

第②步 構建關鍵文件和文件夾



第③步 引入視口標籤以及初始化樣式，修改title，引入description和keywords

<meta

      name="viewport"

      content="width=device-width, user-scalable-no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0"

    />

    <!-- 引入初始化样式文件 -->

    <link rel="stylesheet" href="css/normalize.css" />

    <!-- 引入首页样式文件 -->

    <link rel="stylesheet" href="css/index.css" />

第④步 常用初始化樣式

下面代碼寫在index.css裡面

body {

  max-width: 540px;

  min-width: 320px;

  margin: 0 auto;

  font: normal 14px/1.5 Tahoma, "Lucida Grande", Verdana, "Microsoft Yahei",

    STXihei, hei;

  color: #000;

  background: #f2f2f2;

  overflow-x: hidden;

  -webkit-tap-highlight-color: transparent;

}

攜程移動端各個模塊的高度有確定的值，寬度可以自由變化。

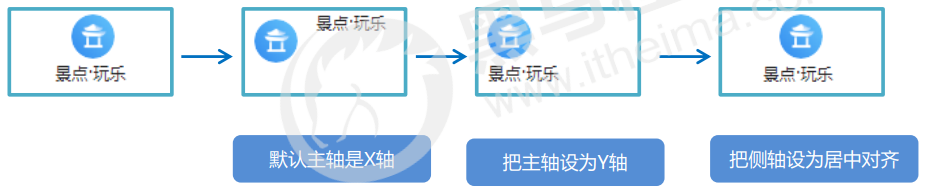
##### 關於position: fixed; 盒子

不寫下面高亮的3行也行，盒子也能居中對齊，京東就沒寫這3行。而下面的代碼是攜程網的寫法。



##### 常見的小盒子佈局：上面一張圖片，下面一行文字





HTML代碼（圖標放在span裡，用2倍精靈圖做span的背景圖片）

<ul class="nav-first">

      <li>

        <a href="#">

          <span class="nav-first-icon"></span>

          <span>景点·玩乐</span>

        </a>

      </li>

    </ul>

CSS代碼

.nav-first {

  display: flex;

  height: 64px;

  background-color: #fff;

  margin: 3px 4px;

  border-radius: 8px;

}

.nav-first li {

  flex: 1;

}

.nav-first a {

  display: flex;

  flex-direction: column;

  align-items: center;

  font-size: 12px;

}

.nav-first-icon {

  width: 32px;

  height: 32px;

  margin-top: 8px;

  background: url(*../images/localnav\_bg.png*) no-repeat 0 0;

  background-size: 32px auto;

}

##### 文字陰影

無文字陰影的效果

有文字陰影的效果

CSS代碼：

.nav-item a {

  flex: 1;

  line-height: 44px;

  color: #fff;

  font-size: 14px;

  /\* 文字阴影 \*/

  text-shadow: 1px 1px rgba(0, 0, 0, 0.2);

}

上面的0.2可以直接寫成 .2

### （3）rem適配佈局（推薦）

#### ① rem單位

rem (root em)是一個相對單位，類似於em。

em是相對於父元素的字體大小。

比如給父元素div設置font-size: 12px; 給子元素p設置width:10em; height:10em。換算成px的話，子元素的width: 100px; height:100px;

rem的基準是相對於根元素HTML元素的字體大小。

比如根元素HTML設置foot-size=12px; 非根元素設置width: 2rem，換算成px就是width: 24px;

rem的優點：通過修改HTML裡面的文字大小來改變頁面中元素的大小，可以整體控制頁面。

#### ② 媒體查詢

媒體查詢（Media Query）是CSS3新語法。

@media可以針對不同的屏幕尺寸設置不同的樣式。

語法

@media mediatype and|not|only (media feature) {

css-code;

}

* 用@media開頭 注意@符號
* mediatype 媒體類型

|  |  |
| --- | --- |
| 值 | 解釋說明 |
| all | 用於所有設備 |
| print | 用於打印機和打印預覽 |
| screen | 用於電腦屏幕、平板電腦、智能手機等 |

* 關鍵字

|  |  |
| --- | --- |
| 值 | 解釋說明 |
| and | 可以將多個媒體特性連接到一起，相當於“且”。 |
| not | 排除某個媒體類型，相當於“非”，可省略。 |
| only | 指定某個特定的媒體類型，可省略。 |

* media feature媒體特性 必須有小括號包含

|  |  |
| --- | --- |
| 值 | 解釋說明 |
| width | 定義輸出設備中頁面可見區域的寬度。 |
| min-width | 定義輸出設備中頁面最小可見區域寬度。 |
| max-width | 定義輸出設備中頁面最大可見區域寬度。 |

min-width和max-width都是包含等於號的，即≥和≤當前設置的值。

例子：

@media screen and (max-width: 800px) {

body {

background-color: pink

}

}

上面的意思是：當屏幕的寬度≤800px時，把body的背景色改為粉紅色。

and表示既設置mediatype為screen，又設置括號裡的media feature值。

##### 例子：媒體查詢改變背景色

代碼：

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

  <style>

    /\* 媒體查詢一般按照從大到小或者從小到大的順序來，最好是從小到大的順序 \*/

    /\* 頁面寬度小於等於539px的頁面背景色為藍色 \*/

    @media screen and (max-width: 539px) {

      body {

        background-color: blue;

      }

    }

    /\* 頁面寬度在540px-970px之間時，頁面變為綠色 \*/

    /\* @media screen and (min-width: 540px) and (max-width: 970px) {

      body {

        background-color: green;

      }

    } \*/

    /\* 上面的代碼有點繁瑣，可以改成下列代碼 \*/

    @media screen and (min-width: 540px) {

      body {

        background-color: green;

      }

    }

    /\* 下面的這幾行代碼會層疊掉上面的幾行代碼 \*/

    /\* 頁面寬度大於等於971px時，頁面變為紅色 \*/

    @media screen and (min-width: 971px) {

      body{

        background-color:red;

      }

    }

  </style>

</head>

<body>

</body>

</html>

##### 根據媒體查詢，引入不同的css資源

語法：

<link rel=”stylesheet” media=”mediatype and|not|only (media feature)” href=”mystylesheet.css”>

#### ③ less基礎

##### css的弊端

css是一門非程式語言，沒有變量、函數、SCOPE（作用域）等概念。

冗餘度高。

css不方便維護及拓展，不利於複用。

css沒有很好的計算能力。

##### Less簡介

Less是Leaner Style Sheets的縮寫，是一門css拓展語言，也稱為css預處理器。

作為css的一種形式的拓展，它並沒有減少css的功能，而是在現有css語法基礎上，為css加入程序語言的特性：變量，Mixin（混入），運算，函數等功能。這大大簡化了css的編寫，並降低了css的維護成本，這樣程序員可以用更少的代碼做更多的事情。

less中文網址：http://lesscss.cn/

常見的css預處理器，除了Less外，還有Sass和Stylus。

總之，Less是一門CSS預處理語言，它擴展了CSS的動態特性。

##### 安裝Less

* 在安裝Less之前，先安裝node。
* 檢查node是否安裝成功，使用cmd命令(win10是window+r打開運行輸入cmd)——輸入“node -v”查看版本即可。
* 基於node.js在線安裝Less，使用cmd命令“npm install -g less”即可。
* 檢查是否安裝成功，使用cmd命令“lessc -v”查看版本。

##### 使用Less

首先新建一個後綴名為less的文件，在這個less文件裡面書寫less語句。

* Less變量

沒有固定的值，可以改變。例如用於顏色。

語法：

@變量名：值；

變量名：

* + 必須有@為前綴
  + 不能包含特殊字符
  + 不能以數字開頭，但是把數字寫在後面可以，比如@font14
  + 大小寫敏感。比如@color和@Color是兩個不同的變量。

例子：

在less文件中，這樣寫

@color: pink;

body {

background-color: @color;

}

* Less編譯

需要把less文件編譯成為css文件，這樣HTML頁面才能使用。

Visual Studio Code中有個插件Easy Less可以用來編譯less。安裝好後，重啟Visual Studio Code，那麼以後每次保存less文件，都會自動生成對應的css文件。

* Less嵌套

例子：父元素div裡有個子元素a，less文件中可以這樣寫：

div{

width: 200px;

height: 200px;

background-color: #fff;

a{

color: #ccc;

}

}

如果寫交集/ 偽類/ 偽元素選擇器，內層選擇器前面寫個&符號。

例子：

a{

color: #ccc;

&:hover{

color:#fff;

}

&::before{

content:”;

}

}

* Less運算

在less中，數字、顏色、變量都可以參與運算，因為less提供了加減乘除運算。其中除法运算要用括号包裹起来。

例子：

@border: 5px +5;

div{

width: 200px – 50;

height: 200px \* 2;

border: @border solid #fff;

background-color: #666- #222;

}

img {

width: (82 / 50rem);

height: (82 / 50rem);

}

例子：

width: (@width + 5) \*2;

注意乘號（\*）和除號（/）的寫法

運算符左右各一個空格隔開

兩個數參與運算，如果只有1個數有單位，結果以這個單位為準。

兩個數參與運算，如果2個數都有不同的單位，結果以第1個數的單位為準。

#### ④ rem適配方案

按照設計稿和設備的寬度，設置不同的HTML根標籤的font-size大小（媒體查詢）。

css中，設計稿元素的寬、高、相對位置等取值，換算為以rem單位的值。

下面兩種方案現在都存在。

方案2更簡單，現階段無需了解裡面的js代碼。

技術方案1：

* less
* 媒體查詢
* rem

技術方案2（推薦）：

* flexible.js
* rem

常見設計稿尺寸寬度

|  |  |
| --- | --- |
| 設備 | 常見寬度 |
| iPhone 4/5 | 640px |
| iPhone 6/7/8 | 750px |
| Android | 常見320px，360px，375px，384px，400px，414px，500px，720px。大部分4.7-5寸的安卓設備為720px； |

一般情況下，以一套或兩套效果圖適應大部分屏幕，放棄極端屏幕對其降級，犧牲一些效果。現在基本以750px為準。

HTML根元素的字體大小=設備寬度/把設備寬度分成的份數（比如15份或者20份）。

比如750px寬的設備分成15份，HTML根元素字體大小是50px。

頁面元素的rem值= 頁面元素值（px） / HTML 字體大小（px）

比如在上面的HTML頁面中，一個寬度100px的div，其rem值是2rem。

#### ⑤ 蘇寧首頁案例製作

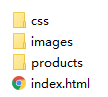
##### 第1步：技術選型

方案：採取單獨製作移動頁面方案

技術：佈局採取rem適配佈局（less+rem+媒體查詢）

設計圖：750px寬

##### 第2步：搭建相關文件夾結構



##### 第3步：設置index.html的head部分

設置視口標籤

設置網站favicon圖標

網站TDK三大標籤SEO優化：title，description，keyword。

引入初始化樣式:

    <link rel="stylesheet" href="css/normalize.css">

##### 第4步：設置common.less文件

a. 建立common.less文件，設置好最常見的屏幕尺寸，利用媒體查詢設置不同的HTML字體大小。

b. 我們關心的尺寸有320px，360px，375px，384px，400px，414px，424px，480px，540px，720px，750px。

c. 把屏幕寬度劃分為15等份。

d. pc端也可以打開蘇寧移動端首頁，默認HTML字體大小為50px，注意這句話寫到最上面（因为代码从上往下执行，如果写在最下面，不会执行到这行代码）。

##### 第5步：設置index.less文件

a. 建立index.less，寫首頁樣式。

b. 把common.less引入index.less裡，語法如下：

@import “common”

common後面可以加後綴名.less，也可以不加。

@import是less把一個less裡的樣式導入到另一個less文件裡，編譯後能夠把一個css裡的樣式導入到另一個css裡。

link是把css文件引入到HTML頁面裡。

c. 編譯後生成index.css，將其link到index.html中。

##### 第6步：設置body樣式

body {

  min-width: 320px;

  width: 15rem;

  margin: 0 auto;

  line-height: 1.5;

  font-family: Arial, Helvetica;

  background: #f2f2f2;

}

固定定位（position:fixed）的盒子必須有寬度。定位的盒子不能用margin auto居中對齊，必須用left:50%; 和translate: transform(-50%); 來調整它在x軸的位置。

#### ⑥ rem適配方案flexible.js

手機淘寶團隊出的簡潔高效移動適配庫。

有了它，不需要再寫不同屏幕的媒體查詢，因為裡面js做了處理。

它把當前設備屏幕寬度劃分為10等份。

前端程序員只要確定當前設備的HTML文字大小。

比如當前設計稿寬度是750px，則把HTML文字大小設置為75px。

其頁面元素rem值：頁面元素的px值 / 75。

其餘的，讓flexible.js去算。

GitHub地址：<https://github.com/amfe/lib-flexible>

修改flexible默認HTML字體大小

如果屏幕超過750px，不要讓頁面超過750px。js的權重比下面的媒體查詢要高，所以用!important提升其權重。

@media screen and (mid-width: 750px){

html{

font-size:75px!important;

}

}

#### ⑦ 使用適配方案2製作蘇寧移動端首頁

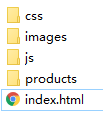
##### 第1步：技術選型

方案：單獨製作移動頁面

技術：rem適配佈局2（flexible.js + rem）

設計圖：750px寬

##### 第2步：搭建相關文件夾結構



##### 第3步：設置index.html的head部分

設置視口標籤

設置網站favicon圖標

網站TDK三大標籤SEO優化：title，description，keyword。

引入初始化樣式:

    <link rel="stylesheet" href="css/normalize.css">

在index.html中引入flexible.js文件

<script src=”js/flexible.js”></script>

##### 第4步：設置body樣式

body {

  min-width: 320px;

  width: 15rem;

  margin: 0 auto;

  line-height: 1.5;

  font-family: Arial, Helvetica;

  background: #f2f2f2;

}

##### VSCode px轉換rem 插件 cssrem

這個插件默認的HTML字體大小是16px

修改默認的HTML字體大小：settings→User→Extensions→cssrem configuration→Foot Font Size

### （4）響應式開發

響應式開發不需要單獨寫移動端頁面。

響應式開發一般不關心高度，因為隨著頁面寬度不同，每一行所包含的內容也會發生變化。

#### ① 響應式開發原理

使用媒體查詢針對不同寬度的設備進行佈局和樣式設置，從而適配不同設備。

|  |  |
| --- | --- |
| 設備劃分 | 尺寸區間 |
| 超小屏幕（手機） | < 768px |
| 小屏設備（平板） | 768px – 992px |
| 中等屏幕（桌面顯示器） | 992px – 1200px |
| 寬屏設備（大桌面顯示器） | > 1200px |

#### ② 響應式佈局容器

響應式需要一個父級作為佈局容器，來配合子級元素實現變化效果。

原理是在不同屏幕下，通過媒體查詢來改變佈局容器的大小，再改變裡面子元素的排列方式和大小，從而實現不同屏幕下，看到的佈局樣式變化。

響應式尺寸劃分

超小屏幕（手機，小於768px）：設置寬度為100%；

小屏幕（平板，大於等於768px）：設置寬度為750px；

中等屏幕（桌面顯示器，大於等於992px）：設置寬度為970px；

大屏幕（大桌面顯示器，大於等於1200px）：設置寬度為1170px。

從小屏幕開始，設置的寬度比屏幕寬度稍微小一些，讓頁面兩側有空白，頁面居中顯示，這樣更好看。

上面是最常用的劃分方法，也可以根據公司的實際需求來劃分。

#### ③ Bootstrap

##### Bootstrap簡介

Bootstrap來自Twitter，是目前最受歡迎的前端框架。Bootstrap基於HTML，CSS和JavaScript，簡潔靈活，讓Web開發更加快捷。

中文官網：<http://www.bootcss.com/>

官網：<http://getbootstrap.com/>

推薦使用：<http://bootstrap.css88.com/>

版本：

2.x.x：兼容性好，代碼不夠簡潔，功能不夠完善，停止維護了。

3.x.x：目前（2019年）使用最多，最穩定，但是放棄了IE6-IE7。支持IE8，但界面效果不好，偏向用於開發響應式佈局、移動設備優先的web項目。

4.x.x：最新版，目前還不是很流行。

##### Bootstrap使用

a. 創建文件夾結構

在文件夾結構裡，建立一個bootstrap文件夾，專門存放bootstrap相關文件（css，fonts，js）

在bootstrap的css文件夾裡，現階段重點看bootstrap.css和bootstrap.min.css這兩個文件。

b. 創建html骨架結構

c. 引入先關文件

d. 書寫內容

Bootstrap的基礎模板HTML文件解釋：

要求當前網頁使用IE瀏覽器最高版本的內核來渲染

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries --> <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>

ie9以下瀏覽器不識別HTML5新增標籤，並導致css不起作用，以下js解決這個問題。

<script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>

以下js解決ie9以下瀏覽器不識別css3 Media Query的問題。

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script> <![endif]-->

修改Bootstrap的默認樣式，可以自己寫個類進行修改，但要注意權重問題。

##### Bootstrap佈局容器

Bootstrap需要為頁面內容和柵格系統包裹一個.container容器，Bootstrap預先定義好了這個類，叫.container和container-fluid。

在一個頁面中，container和container-fluid可以反復使用。

通欄的盒子，常規佈局，不需要container。

container類

響應式佈局的容器寬度：

* 大屏（≥1200px）寬度為1170px；
* 中屏（≥992px）寬度為970px；
* 小屏（≥768px）寬度為750px；
* 超小屏（100%）。

只要寫了這個.container類，就不用自己寫媒體查詢了。

container-fluid類

* 流式佈局容器百分百寬度。
* 佔據全部視口（viewport）的容器。
* 適用於單獨做移動端開發。

##### Bootstrap柵格系統

柵格系統英文為“grid systems”，也譯為“網絡系統”，指將頁面佈局劃分為等寬的列，然後定義列數來模塊化頁面佈局。

Bootstrap提供了一套響應式、移動設備優先的流式柵格系統，隨著屏幕或視口（viewport）尺寸的增加，系統會自動分為最多12列。即把container劃分為最多12等份，其寬度會隨著頁面的變化而變化。

柵格系統通過一系列的行（row）與列（column）的組合來創建頁面佈局。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 超小屏幕（手機）<768px | 小屏設備（平板）≥768px | 中等屏幕（桌面顯示器）≥992px | 寬屏設備（大桌面顯示器）≥1200px |
| .container最大寬度 | 自動（100%） | 750px | 970px | 1170px |
| 類前綴 | .col-xs- | .col-sm- | .col-md- | .col-lg- |
| 列（column）數 | 12 | | | |

* 行（row）必須放到container佈局容器裡。
* 給列添加類前綴，即可平均劃分列。
* 可以同時為一列指定多個設備的類名，以便劃分不同份數。例如class=”col-md-4 col-sm-6”。
* 默認container和每一列（column）有左右15像素的padding。如果不想要，可以自定義樣式去掉。
* 如果row裡面子元素的份數小於12，則鋪不滿整個屏幕
* 如果row裡面子元素的份數大於12，多於的列（column）將被作為一個整體另起一行排列。

例子：

在大屏幕裡，第1行4個div均分屏幕寬度，每個div占3份（4\*3=12）；第2行4個div，div1站6分，剩下3個div個站2分（6+3\*2=12）

    <div class="container">

      <div class="row">

        <div class="col-lg-3">1</div>

        <div class="col-lg-3">2</div>

        <div class="col-lg-3">3</div>

        <div class="col-lg-3">4</div>

      </div>

      <div class="row">

        <div class="col-lg-6">5</div>

        <div class="col-lg-2">6</div>

        <div class="col-lg-2">7</div>

        <div class="col-lg-2">8</div>

      </div>

</div>

注意，這些column div不能用margin擠開。因為Bootstrap源代碼用的是float和%來平分寬度。如果用margin擠開，盒子一行就放不下了，肯定會掉下來。可以在每個column div裡面再套一個小盒子。因為column自帶左右padding，所以就能擠開盒子了。

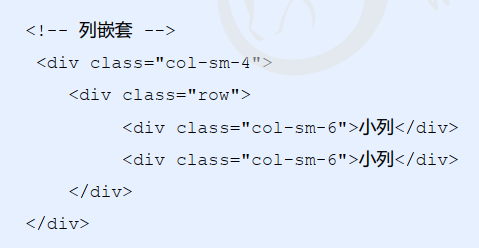


##### Bootstrap列嵌套

把一個列分成若干小列。通過添加一個.row和一系列column的類來實現。

在column裡面加row，在加column，可以取消父元素column的padding值，而且讓子column和父column的高度一樣。





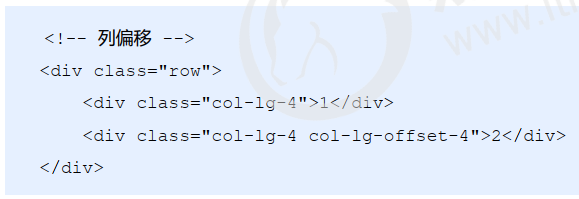
##### Bootstrap列偏移

使用.col-md-offset- 類可以將列向右側偏移。這些類實際是為當前元素增加了左側的邊距（margin）。

例子：



代碼：



偏移的分數=12-兩個盒子所佔分數

##### Bootstrap列排序

用.col-md-push-（把左側盒子push到右邊） 和.col-md-pull-（把右側盒子pull到左邊） 可以改變column的順序。



<div class="container">

      <div class="row paint">

        <div class="col-lg-3 col-lg-push-9">1

        </div>

        <div class="col-lg-9 col-lg-pull-3">2

        </div>

      </div>

    </div>

##### Bootstrap響應式工具

針對不同設備展示或隱藏頁面內容

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 類名 | 超小屏 | 小屏 | 中屏 | 大屏 |
| .hidden-xs | 隱藏 | 可見 | 可見 | 可見 |
| .hidden-sm | 可見 | 隱藏 | 可見 | 可見 |
| .hidden-md | 可見 | 可見 | 隱藏 | 可見 |
| .hidden-lg | 可見 | 可見 | 可見 | 隱藏 |

.visible-xs .visible-sm .visible-md .visible-lg只在指定的屏幕寬度下顯示。

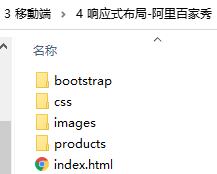
#### ④ 項目：阿里百家秀響應式頁面

方案：響應式

技術：Bootstrap框架

設計圖：1280px設計尺寸

第1步 創建文件夾結構



第2步 創建HTML骨架結構

第3步 引入相關樣式文件

無需引入normalize.css，因為Bootstrap已經整合了。

第4步 書寫內容

**修改container最大寬度**

本效果圖採取1280px的寬度，而Bootstrap裡面container寬度最大為1170px，因此要修改container最大寬度。

在index.css中寫上如下代碼：

@media screen and (min-width: 1280px) {

  .container {

    width: 1280px;

  }

}

**logo製作**

把header的padding-left變成0。

把logo裡面的img寬度修改為100%，讓其自適應寬度自由縮放。

**字體圖標**

ttps://bootstrap.css88.com/的組件中有字體圖標。

複製圖標下面的類名，比如：glyphicon glyphicon-camera

哪個盒子需要這個圖標，就把這個類名給這個盒子。比如：

<li><a href="#" class="glyphicon glyphicon-camera">生活馆</a></li>

原理：給盒子添加一個before偽元素。

要讓圖標裡和文字對齊，順著圖標的類去修改，比如這裡：

li a::before {

vertical-align: middle;

}

Bootstrap裡面的清除浮動類名是clearfix

**logo的響應式製作**

在xs狀態下隱藏logo，在xs狀態下顯示<span>阿里百秀</span>