

4.2 CSS 基础

- CSS 概述
- CSS 基本语法
- CSS 常用属性
- CSS 布局

目录

1

CSS 概述

2

CSS 基本语法

3

CSS 常用属性

4

CSS 布局

CSS

- **CSS** (Cascading Style Sheets, 层叠样式表) 是一种用于描述 HTML 页面样式的**样式表**[スタイルシート]语言。
- CSS 指定了在浏览器上显示 HTML 元素的一些具体细节, 如大小、颜色等。
- 单独的 CSS 代码不能运行, 化妆品离开了人就没有意义了。



HTML は
顔のパーツ



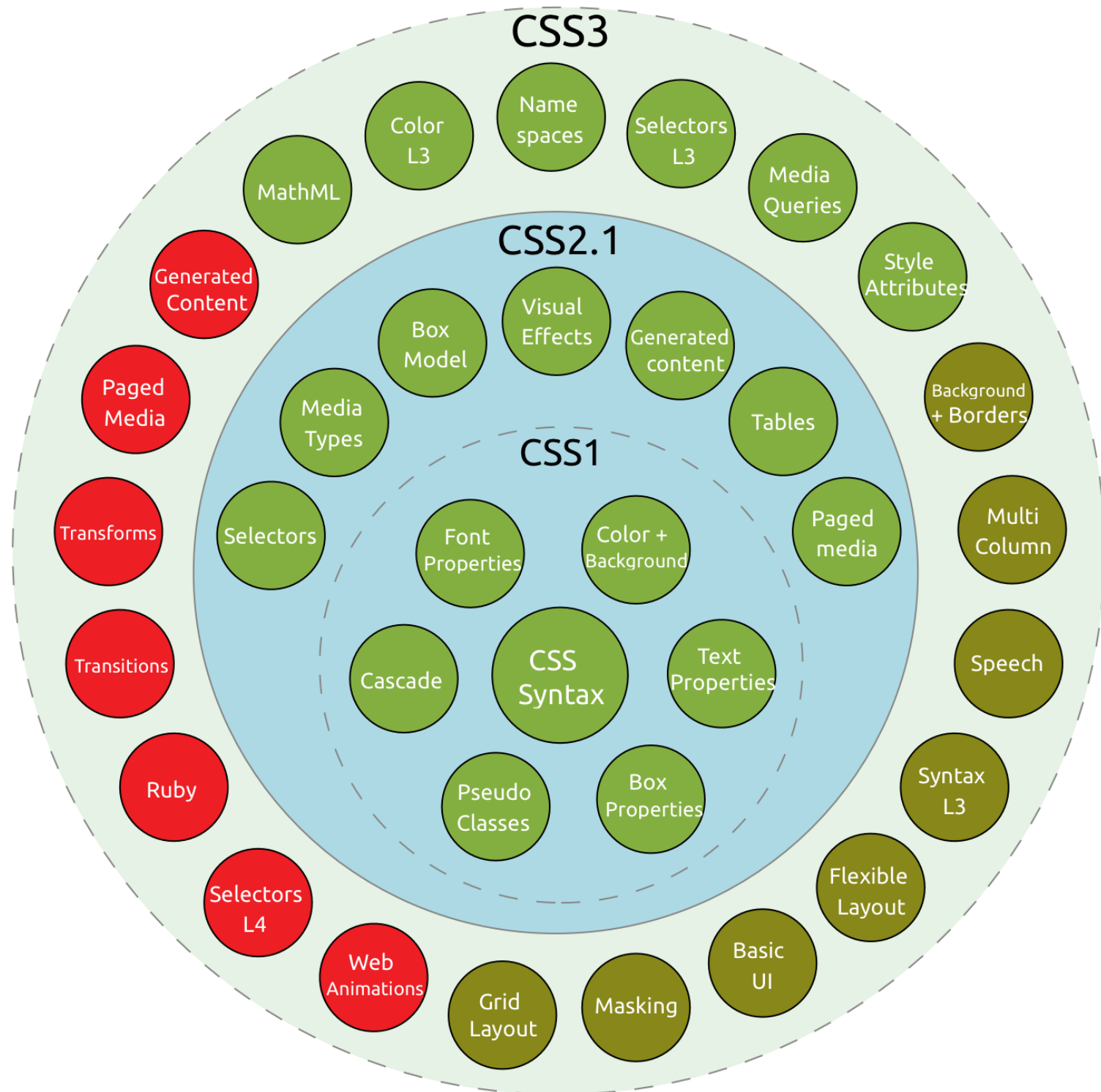
CSS は
お化粧



JavaScript は
動き



CSS 历代版本



- 本课程学习 CSS3。

CSS 的使用

- 要使用 CSS 修饰 HTML 元素，一共有三种方法：
 - 使用 **<link>** 标签导入外部 CSS 文件。
 - 使用 **<style>** 标签在 HTML 文档内部直接书写 CSS 代码。
 - 使用 **style** 属性直接修饰某个元素（**内联**[\[インライン\]](#) CSS）。

导入外部 CSS 文件

- CSS 的代码可以写在一个文件中，以 “.css” 结尾。
- 要导入外部 CSS 文件，使用 `<link>` 标签（一般放在 `<head>` 标签内）：

```
<link rel="stylesheet" href="代码文件地址">
```

- 其中，`rel` 属性的值 “stylesheet” 表示要导入一个样式表；`href` 属性的值是想导入的 **CSS** 文件的 **URL**，里面包含了 CSS 代码。

直接嵌入 CSS 代码

- 也可以在 **<style>** 标签里直接书写 CSS 代码（一般放在 **<head>** 标签内）：

```
1 <style>
2   p {
3       color:green;
4   }
5 </style>
```

内联 CSS

- 也可以使用任意元素的 **style** 属性直接修饰该元素的样式:

```
<button style="color:blue">This is a button</button>
```



CSS 的应用优先级

- 如果我们同时使用多种不同的方法修饰同一个 CSS 元素，最终哪一个 CSS 的样式会被应用？
- 多个 CSS 样式的应用流程如下：
 1. 如果该元素有内联样式，内联样式会被应用。
 2. 如果该元素没有内联样式，则所有 CSS 样式（包括相同的或不同的文件 / 代码中的样式）中**最后**被导入 / 嵌入的会被使用。



CSS 的使用方法选择

- 使用哪种方法为元素添加样式并没有严格的标准，只需要根据自己的需求选择最方便的方法即可。
- 一般来说：
 - `<link>` 标签导入的外部 CSS 可以用来指定**整个网站**上统一的样式；
 - `<style>` 标签书写的内部 CSS 可以用来指定**整个网页**上统一的样式；
 - `style` 属性书写的内联 CSS 可以用来指定**某 1 个**元素特有的样式。
- 总的来说，建议尽量多使用外部 CSS 文件以加强统一性和修改的灵活性。

Q & A

Question and answer

目录

1

CSS 概述

2

CSS 基本语法

3

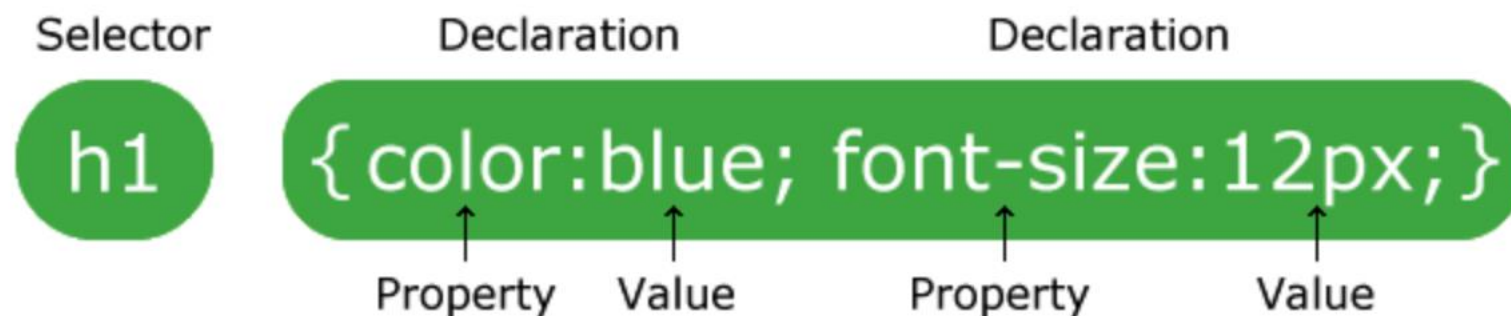
CSS 常用属性

4

CSS 布局

CSS 基本语法

- CSS 代码由多个**规则集**[規則セット]组成：



- 规则集的开头是一个**选择器**[セレクター]，它指定了**哪种元素**会应用大括号“`{}`”内声明的样式。这里是所有的 `<h1>` 元素。
- 大括号“`{}`”括起来的部分是一个**声明块**[宣言ブロック]。一个声明块内可以包含多个**声明**[宣言]。
- 每个声明又包含一个**属性**[プロパティ]（**Property**）和它对应的**值**，用冒号“`:`”隔开。每个声明以分号“`;`”结尾。

选择器

- **选择器**`[セレクトター]`用于指定哪些元素会使用规则集里的样式。
- 我们可以将选择器大体分为 5 类：
 - **基本的选择器**：根据元素的**名称**、**id** 或**类**选择元素；
 - **属性选择器**：根据 HTML **属性**选择元素；
 - **组合器选择器**：根据元素和其他元素的关系选择元素；
 - **伪类选择器**：选择元素的某个特殊状态；
 - **伪元素选择器**：选择元素的某个特定部分；

元素选择器

- 基本的选择器包括**元素选择器**、**ID 选择器**和**类选择器**。
- **元素选择器**[要素型セレクター]根据元素名称（即 HTML 标签的名称）选择特定元素。
- 要选择某一种元素，直接书写该元素名即可。
- 下面的代码会让页面上的所有 `<p>` 元素的文字变为红色：

```
1 p {  
2   color: red;  
3 }
```

id 选择器

- **ID 选择器**[ID セレクター]根据元素的 **id 属性**选择特定元素。
- 理论上说，每一个元素的 **id 属性**在页面中应该设为**唯一的**。因此，ID 选择器用于选择 **1** 个特定的元素。
- 要选择具有特定 ID 的元素，写一个井号“**#**”，后面紧跟该元素的 **ID**。
- 下面的代码会让页面上 ID 为 text-1 的元素的文字变为红色：

```
1 #text-1 {  
2     color: red;  
3 }
```

类选择器

- **类选择器**[クラスセレクター]根据元素的 **class 属性**选择特定元素。
- 和 id 不同，多个元素的 **class 属性**可以设为同样的值。因此，类选择器选择的是同一个类的所有元素。
- 要选择具有特定类的元素，写一个点“.”，后紧跟类名。
- 下面的代码会让页面上 class-1 类的所有元素的文字变为红色：

```
1 .class-1 {  
2     color: red;  
3 }
```

Tips

和 Java 的变量名不同，HTML 元素的 ID 和类名可以包含连字符“-”。

元素名和类选择器

- 元素名后直接跟类选择器可以特定属于某个类的某种元素。
- 下面的代码会让页面上属于 `class-1` 类的所有 `<p>` 元素的文字变为红色：

```
1 p.class-1 {  
2     color: red;  
3 }
```

属于多个类的元素

- 同一个元素可以属于多个类。
- 在设置 class 属性时，可以将多个类名用空格隔开。
- 下面这个 <p> 元素将同时属于 class-1 和 class-2 类，并因此同时受到两个类的样式的影响：

```
<p class="class-1 class-2">Two Classes</p>
```

通用选择器

- **通用选择器**[全称セレクトター]用于选择页面上的**所有元素**。
- 使用星号 “*” 表示通用选择器。
- 下面的代码会让页面上所有元素的文字变为红色：

```
1 * {  
2   color: red;  
3 }
```

Try 01011
11010
01011

selector.html

Note



不同选择器的优先应用顺序：

1. id 选择器。
2. 类选择器。
3. 元素名选择器。
4. 通用选择器。

选择器列表

- 有时，多个选择器会定义相同的（或包含共通的）样式。
- 我们可以用**选择器列表**[セレクトーリスト]表示多个选择器。
- 要使用选择器列表，用逗号将多个选择器隔开。
- 下面的代码会让页面上所有 `<h1>` 元素、`<h2>` 元素以及 ID 为 `text-1` 的元素的文字变为红色：

```
1 h1, h2, #text-1 {  
2     color: red;  
3 }
```

组合器选择器

- 组合器组合多个选择器，基于和其他元素的关系选择某个元素。
- CSS中有四种不同的组合器：
 - **后代选择器**（空格）：选择某种元素内的元素（后代元素）；
 - **子选择器**（>）：选择**直接**包含在某种元素内的元素（子元素）；
 - **相邻兄弟选择器**（+）：选择直接连在某种元素之后的元素；
 - **一般同级选择器**（~）：选择和某种元素在同一个元素里的元素（兄弟元素）。



伪类选择器

- **伪类选择器**[擬似クラスセレクター]指定某个元素在特定状态下的样式。
- 比如一个超链接元素 `<a>` 可能处于**普通** (link)、**已访问** (visited)、**鼠标悬停时** (hover)、**鼠标点按时** (active) 等状态。
- 要使用伪类选择器，使用冒号 “:” 链接普通的选择器和状态名。



注释

- CSS 的注释类似 Java 的多行注释，以 “*/**” 开头， “**/*” 结尾：

```
1  /*  
2   * This is a CSS comment.  
3   */  
4  p { color: red; }
```

Q & A

Question and answer

目录

1

CSS 概述

2

CSS 基本语法

3

CSS 常用属性

4

CSS 布局

文字颜色

- 就像之前很多例子中我们看到的一样, **color** 属性可以设置元素中文字的颜色:

```
1 <h1 style="color:red">This Is A Heading</h1>
2 <p style="color:green">This is a paragraph.</p>
3 <button style="color:blue">This is a button</button>
```

This Is A Heading

This is a paragraph.

This is a button

CSS 背景颜色

- 元素的背景颜色可以使用 **background-color** 属性指定：

```
1 <h1 style="background-color:red">This Is A Heading</h1>
2 <p style="background-color:green">This is a paragraph.</p>
3 <button style="background-color:blue">This is a button</button>
```

This Is A Heading

This is a paragraph.

This is a button

网页颜色

- 在 CSS 中可以使用很多方法表示颜色：

- 颜色的名字：tomato

- RGB 表示：rgb(255, 99, 71);

- 十六进制表示：#ff6371;

- HSL 表示：hsl(9, 100%, 64%) ;

- RGBA 表示：rgba(255, 99, 71, 0.5) ;

-

rgb(255,99,71)


#ff6371

hsl(9,100%,64%)

rgba(255,99,71,0.5)

hsla(9,100%,64%,0.5)

CSS 颜色名

- CSS 3 支持 147 种颜色的名称。
- 可以在这个网页上查看所有名称的列表：
 <https://www.w3.org/wiki/CSS/Properties/color/keywords>



文字相关属性

- 下表列出了一些常用的文字相关属性：

属性	定义内容
color	文字的颜色
text-align	文字的水平对齐方式
vertical-align	元素内文字的垂直对齐方式
text-decoration	文字的装饰（下划线、删除线等）
text-overflow	控制元素内显示不下文字时的表现
font	字体

水平对齐

- **text-align** 属性可以设置文本的水平对齐方式。使其靠左、靠右或居中对齐：

```
1 <p style="text-align:left">This is left-aligned.</p>  
2 <p style="text-align:center">This is center-aligned.</p>  
3 <p style="text-align:right">This is right-aligned.</p>
```

This is left-aligned.

This is center-aligned.

This is right-aligned.

字体

- **font**[フォント]属性定义了字体的种类，大小和一些其他样式。常见的字体相关属性如下：

属性	定义内容
font-family	字体系列（种类）
font-size	文字大小
font-style	文字的特殊风格（粗体、斜体等）
font-weight	文字笔画的粗细

字体系列

- **font-family** 属性定义了文字属于什么字体系列。
- **字体系列**[フォントファミリー] 可以理解为字体的种类。但是一个系列可能包含多种字体，比如不同粗细的版本。
- 还有一些特别的字体系列，被称为**通用系列**[ジェネリックファミリー]。其他字体都属于某些种通用系列。常见的通用系列包括 Serif、Sans-serif、monospace 等。

Serif和Sans-serif字体之间的区别



接次页 ➞

字体系列

- **font-family** 属性后可以写多个字体系列名，用逗号“,” 隔开：

```
1 p {  
2   font-family: Arial, Helvetica, sans-serif;  
3 }
```

- 浏览器会先检查第 1 个字体是否可以使用；如果不能，则检查第 2 个字体，以此类推。因此，建议将最后一种字体设置成通用字体中的一种，保证浏览器能够找到。
- 如果字体系列的长度超过一个单词，则它必须用引号引起来，例如："Times New Roman"。

使用更多字体

- 浏览器直接能支持的字体有限，我们可以使用一些 **Web 字体** 来装饰页面。
- 有很多网站向我们提供了字体，这里以使用起来最快捷的 **Google 字体 API** 为例：
 1. 在  <https://fonts.google.com/> 上搜索需要的字体。
 2. 复制网页右侧的 `<link>` 标签放入 HTML 文档。
 3. 使用该字体名设置对应元素的 `font-family`。



Use on the web

To embed a font, copy the code into the `<head>` of your html

☒ `<link>` ☐ `@import`

```
<link rel="preconnect" href="https://
fonts.googleapis.com">
<link rel="preconnect" href="https://
fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.
com/css2?family=VT323&display=swap" r
el="stylesheet">
```

CSS rules to specify families

```
font-family: 'VT323', monospace;
```

图标

- Google 字体还为我们提供了一个简单的添加图标[アイコン]的功能：
 1. 在  <https://fonts.google.com/icons?icon.set=Material+Icons> 上找到想用的图标的名称。

2. 在 <head> 标签中链接以下样式表：

```
<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
```

3. 在想插入图标的位置插入以下标签：

```
<span class="material-icons">图标名称</span>
```

4. 你还可以使用 CSS 设置图标的颜色等样式。



文字大小

- **font-size** 属性设置文字大小。
- 有几种可用的单位, 包括 px (像素)、em (与父元素文字大小的比例)、rem (与根元素文字大小的比例) 等:

font-size:16px

font-size:32px

font-size:1.5rem

Note !

不要使用文字大小属性区分标题和文本。始终使用 `<h1>` ~ `<h6>` 标签设置标题。

使用 `` 标签设置文字属性

- 如果你将 CSS 文字属性应用在 `<h1>` 或 `<p>` 等标签上，其影响范围将会是所有 `<h1>` 或 `<p>` 标签内的文字。
- 如果你想要单独设置一部分文字的属性，可以使用 `` 标签将这部分括起，再使用 CSS 设置 `` 标签的属性。
- 比如，下面的代码中将只有“红色”部分文字被设为红色：

```
<p>这句话中只有部分文字是<span style="color:red">红色</span>的。</p>
```


高度和宽度

- **height** 和 **width** 属性分别可以用于设置元素的高度和宽度。
- height 和 width 的值也可以以 px（像素）、em（相对于文字大小）、%（百分比）等为单位。

总结：CSS 中的长度单位

Sum Up

单位	含义
px	像素
cm、mm、inch 等	厘米、毫米、英尺等物理上的长度单位
em	文字大小
rem	根元素的文字大小
%	父元素宽度 / 高度的百分之几
(不写单位)	父元素宽度 / 高度的几倍
vw / vh	视口宽度 / 高度

背景

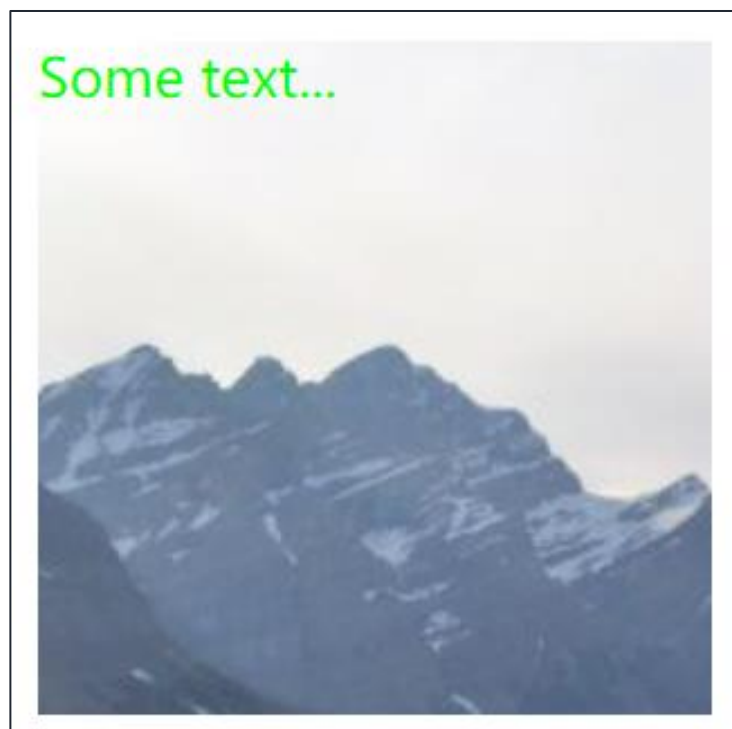
- 常见的和背景相关的属性如下：

属性	定义内容
background-color	背景颜色
background-image	背景图片
background-size	背景图片的大小
background-position	背景的位置
background-repeat	背景的重复方式
background-attachment	背景图片是否会随页面滚动而移动

背景图片

- **background-image** 属性指定图片作为元素的背景:

```
1 #a {  
2   background-image: url(img/mountains.jpg);  
3 }
```



背景图片大小

- **background-size** 属性可以告诉浏览器调整图片大小的方式，可以设为以下几种值：

值	效果
contain	尽可能显示全图片（不会改变图像比例）
cover	尽可能铺满元素（不会改变图像比例）
长度值 长度值 如：200px 300px	将图片缩放成指定宽度和高度。

Tips

如果不想让缩放后的图片重复，可以设置
repeat: no-repeat。

背景简写

- 可以把刚刚说到的多个属性一并用 **background** 属性定义：

```
1 #first {  
2   background-image: url("img/mountains.jpg");  
3   background-repeat: no-repeat;  
4   background-attachment: fixed;  
5 }
```



```
1 #first {  
2   background: no-repeat fixed url("img/mountains.jpg");  
3 }
```

- 这被称为属性的**简写**[一括指定] (Shorthand)。除了 background, CSS 还有很多这样的简写属性。

Try 

background.html

其他常用的属性

- opacity: 通用属性, 设置元素的透明度。
- animation: 通用属性, 设置动画效果。
- inherit: 通用属性值, 使该值继承父元素。
- linear-gradient、radial-gradient: 用于设置颜色值的函数, 可以设置渐变色。
- CSS 的官方文档不便查阅, 可以查阅一些其他公司提供的文档, 比如 Mozilla:
 <https://developer.mozilla.org/en-US/docs/Web/CSS>

Q & A

Question and answer

目录

1

CSS 概述

2

CSS 基本语法

3

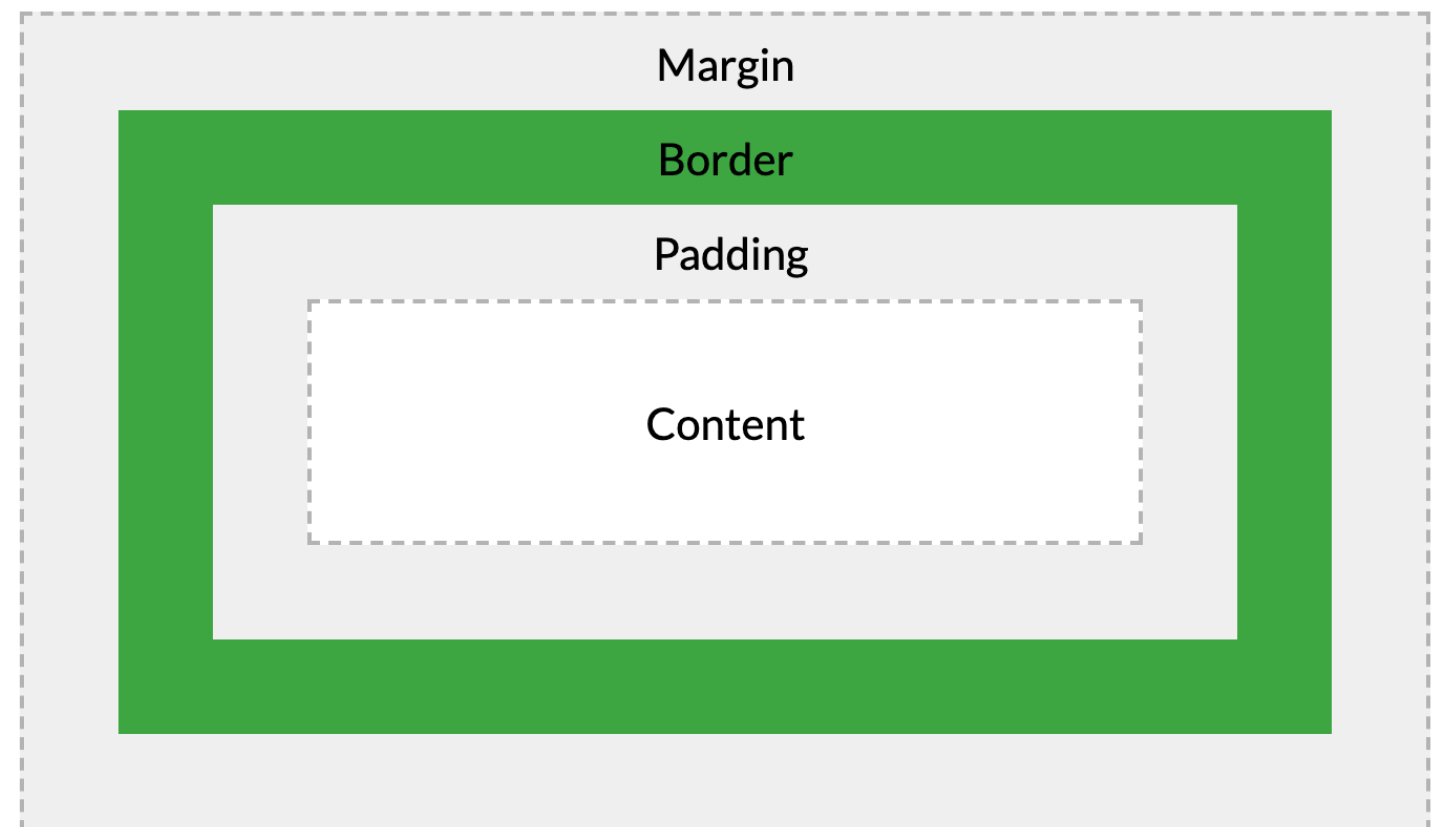
CSS 常用属性

4

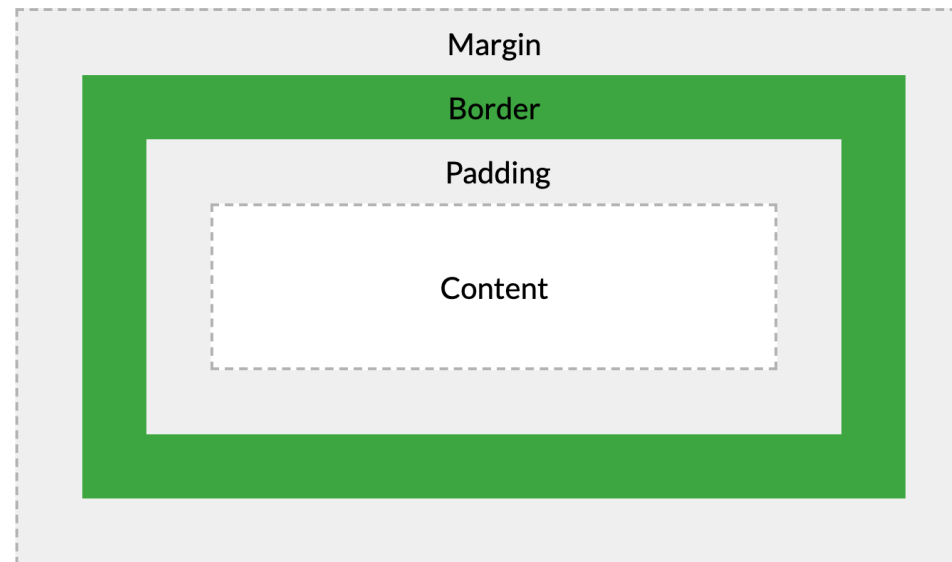
CSS 布局

盒模型

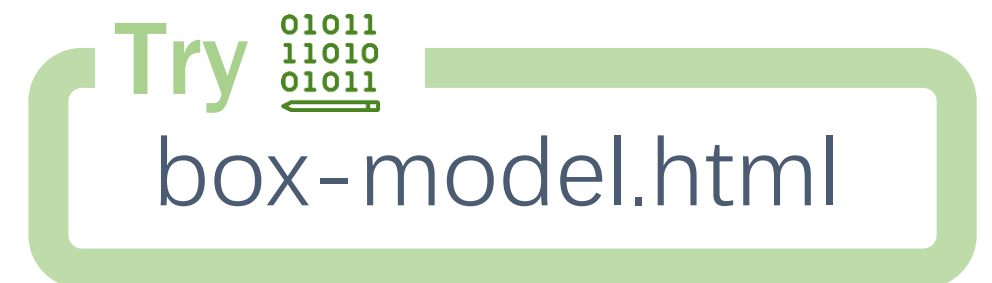
- 在讨论 CSS 中的设计和布局时会用到术语“**盒模型**[ボックスモデル]”。
- 盒模型把每一个 HTML 元素的视图分解成四个部分：
 1. 内容 (Content)、
 2. 内边距 (Padding)、
 3. 边框 (Border)、
 4. 外边距 (Margin)。



盒模型综述



- **内容**：文本、图像等实际显示的区域。
- **内边距**：内容和边框之间的区域。
- **边框**：边框所占区域。
- **外边距**：边框外的透明区域，用于与其他元素分隔。



边框

- 常见的和元素边框相关的属性如下：

属性	定义内容
border-style	边框类型
border-width	边框宽度
border-color	边框颜色
border-radius	边框圆角的半径

边框类型

- **border-style** 属性指定要显示的边框类型。比如，将属性值设为 **solid** 将定义一个实线边框，设为 **dashed** 将定义一个虚线边框：

border-style: solid

border-style: dashed

border-style: dotted

border-style: double

border-style: groove

border-style: ridge

border-style: inset

border-style: outset

边框宽度

- **border-width** 属性指定边框的宽度。
- 可以使用 px、em 等长度单位，也可以使用 **thin**、**medium** 或 **thick** 分别指定较窄、中等或较宽的边框：

```
border-width: 5px
```

```
border-width: thin
```

```
border-width: medium
```

```
border-width: thick
```

边框颜色

- **border-color** 属性指定边框的颜色：

border-color: red

border-color: yellow

border-color: green

border-color: blue

设置每条边的属性

- 一个边框包括 4 条边。如果刚刚说到的属性只包含了 1 个值，它将被应用于全部 4 条边：

```
border-color: red
```

- 如果包含了 2 个值，它们将分别被应用于上下边和左右边：

```
border-color: red lime
```

- 如果包含了 3 个值，它们将分别被应用于上边、左右边和下边：

```
border-style: solid double dotted
```

- 如果包含了 4 个值，它们将分别被应用于上、右、下、左边：

```
border-width: 2px 3px 4px 5px
```


单独设置某一边的属性

- 也可以使用 **border-top-**、**border-right-**、**border-bottom-** 或 **border-left-** 开头的属性单独设置某一边的属性：

```
border-style: solid;  
border-top-color: red;  
border-right-style: dotted;  
border-left-width: 5px;  
border-left-color: blue;  
border-bottom-style: none;
```

边框简写

- **border** 属性是刚刚介绍的属性的简写属性。你可以用它统一设定边框的类型、宽度和颜色：

```
border: solid 2px red
```

```
border: dashed 5px blue
```

- 你也可以用 border-top、border-right、border-bottom 或 border-left 统一指定单边的各种属性：

```
border-right: double 3px green;
```

```
border-bottom: solid 2px red;
```

圆角边框

- **border-radius** 属性可以设置圆角边框。属性值为一个长度，长度越长，边框越圆：

`border-radius: 2px`

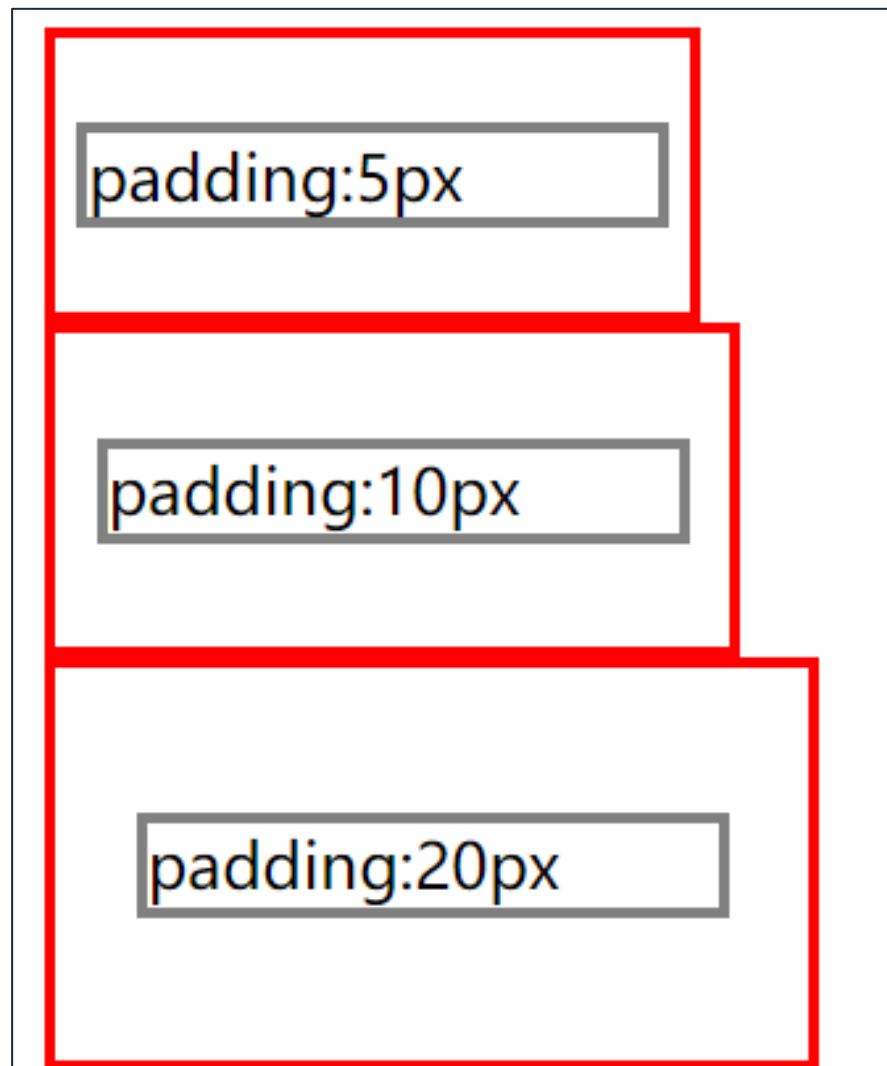
`border-radius: 5px`

`border-radius: 20px`



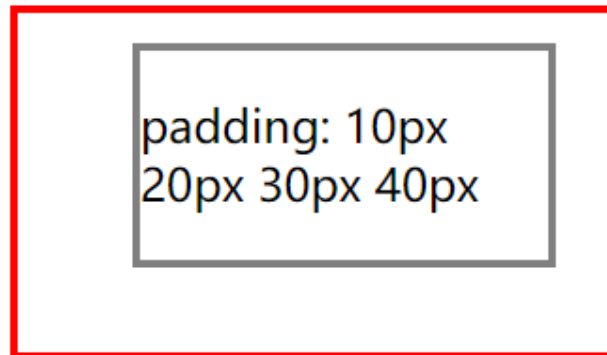
内边距

- **padding** 属性用于设定元素 *内边距* 的大小:

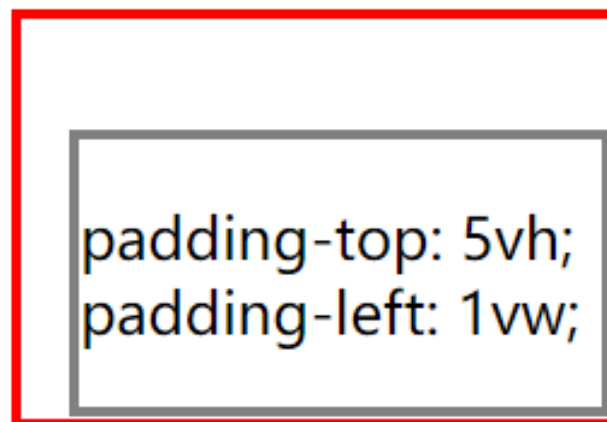


每个方向的内边距

- 和 border 属性类似，你也可以用多个值设置不同方向的内边距：

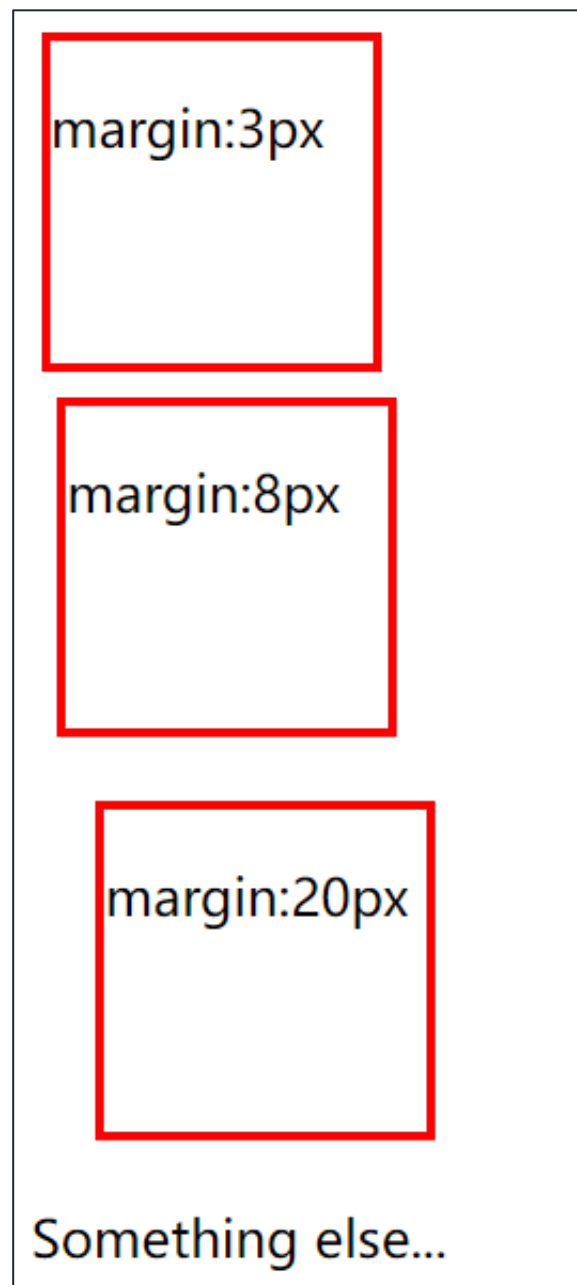


- 也可以使用 **padding-top** 等属性设置单独某个方向的内边距：



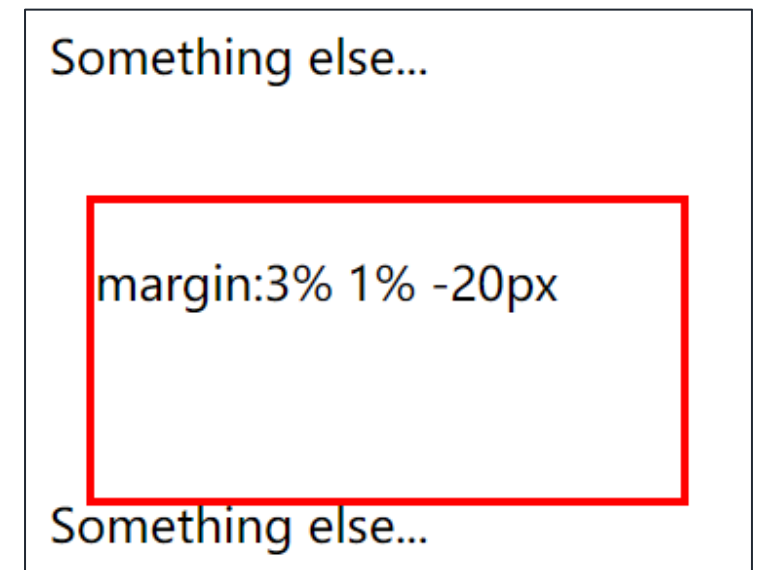
外边距

- **margin** 属性用于设定元素外边距的大小:

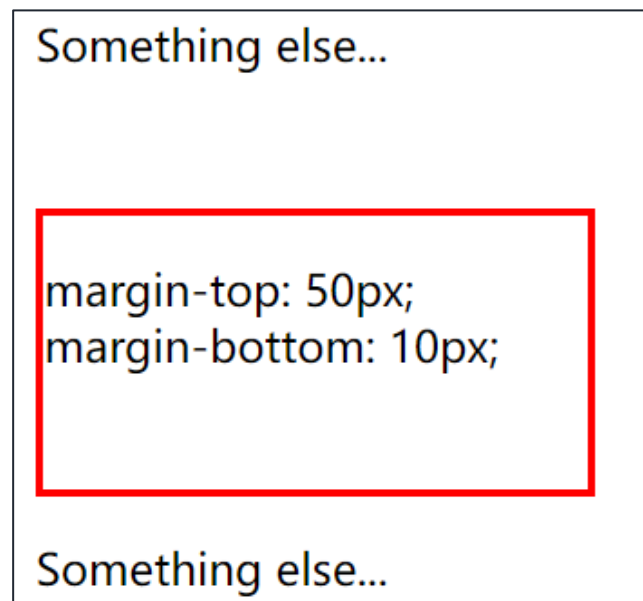


每个方向的外边距

- 也可以用多个值设置不同方向的外边距:

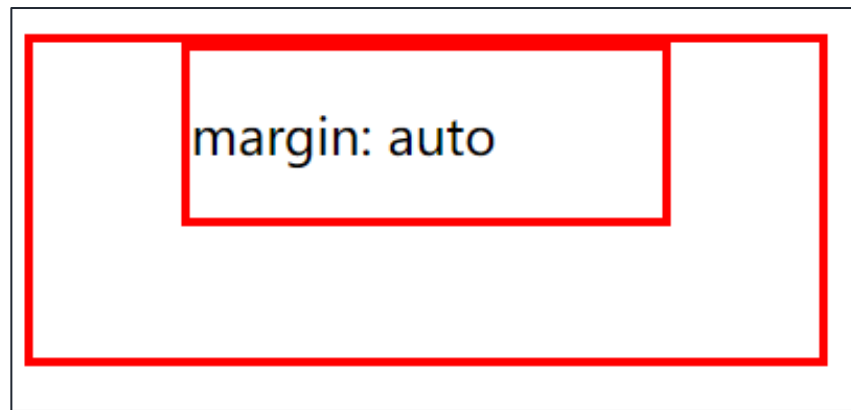


- 也可以使用 **margin-top** 等属性设置单独某个方向的外边距:

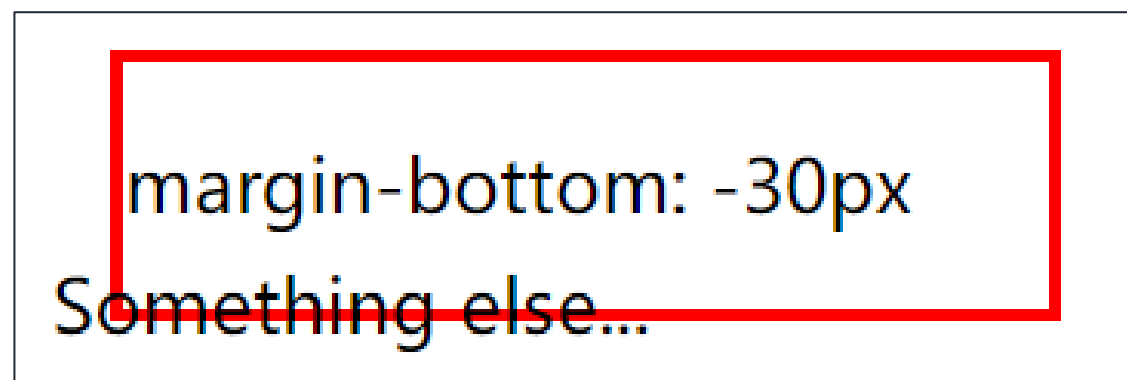


边距的数值

- 可以将 margin 属性设置为 **auto**，使元素水平居中显示：



- 可以将 margin 属性设为负值，使元素和别的元素“重叠”：



盒模型的高度和宽度


- 注意：**height** 和 **width** 属性只能用于设置元素 **Contents** 部分的高度和宽度。
- height 和 width 的值默认**不包括** Padding、Border 和 Margin 部分。要计算元素的完整大小，还必须添加 Padding、Border 和 Margin 部分的长度。



高度和宽度的计算

- 比如这个元素：

```
1 #box {  
2   width: 320px;  
3   padding: 10px;  
4   border: 5px solid red;  
5   margin: 0;  
6 }
```

- 它的实际宽度是多少？
 - 320px （宽度） + $2 \times 10\text{px}$ （左右内边距） + $2 \times 5\text{px}$ （左右边框）
= 350px
- 实际使用的过程中，有些相关元素的边距还会被抵消掉，我们这里不做详细介绍，可以参考：
 https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Mastering_margin_collapsing

Q & A

Question and answer

元素的定位

- 使用 CSS 中对元素定位需要两步：
 1. 使用 **position** 属性确定元素的定位方式（基于什么移动）；
 2. 使用 **top**、**left**、**bottom** 及 **right** 属性确定具体的偏移数值（移动多少距离）。

position 属性

- **position** 属性指定用于指定元素的定位方式。
- 有 5 种不同的定位方式：
 1. static (默认)：浏览器自动定位，你无法调整位置。
 2. relative：相对于自动定位的位置确定位置。
 3. fixed：相对于窗口（视口）确定位置。
 4. absolute：相对于父元素确定位置。
 5. sticky：当滚动到它后，相对于窗口确定位置。

偏移属性

- **top**、**right**、**bottom** 及 **left** 属性指定元素的偏移量。
- top 的值代表元素离指定容器（比如 fixed 是窗口，absolute 是父元素）的上边界距离多远。其余以此类推。

Note



如要把 position 设为 **absolute**，必须保证父元素的 position **不为 static**!

Try 

layout.html

display 属性

- **display** 属性设定元素的显示类型。常见的用法包括设置为 **inline-block** 使元素可以在一行内显示多个（内联）；或者设置为 **block** 使元素在一行内只显示一个（块级）：

```
display: inline-block display: inline-block display: inline-block
```

```
display: block
```

```
display: block
```

```
display: block
```

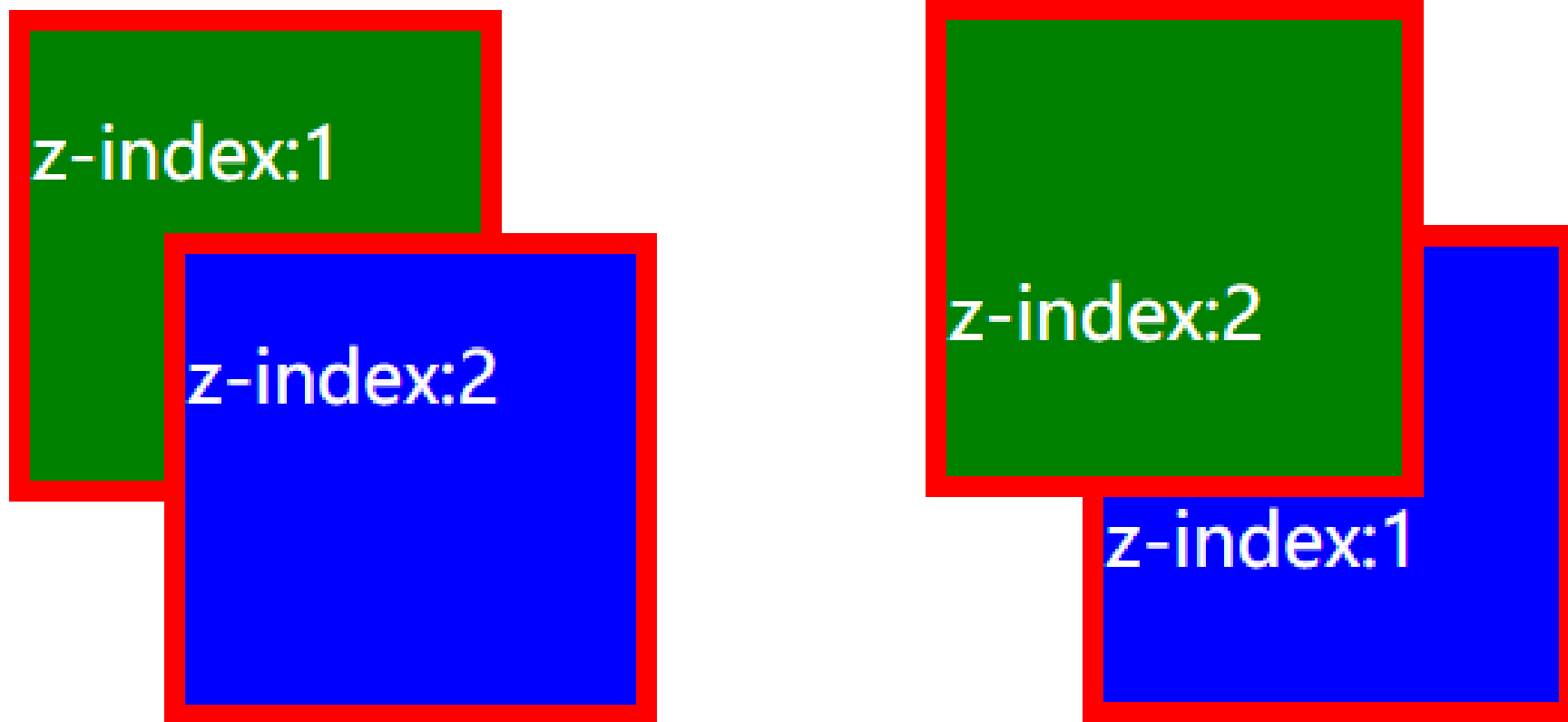
transform 属性

- **transform** 属性设定元素的各种变形，包括移动、扩缩、旋转、切变等。
- 一个常见的用法是结合 top、left 属性设定水平和 / 或垂直居中：

```
position: relative;  
top: 50%;  
left: 50%;  
transform: translate(-50%, -50%);
```


Z-index

- **z-index** 属性指定元素 Z 轴的坐标。Z 轴坐标越大，元素就越“靠上层”（比如会覆盖其他元素）：



响应式布局

- CSS 提供了 **@media** 这一特殊的语法，以应对响应式布局的需求。你可以设置某些 CSS 样式只在屏幕大小满足特定条件时才有效，从而动态应对不同大小的设备。
- 比如，下面对 `<p>` 的样式声明只在屏幕大小大于 576 像素时生效：

```
1 @media (min-width: 576px) {  
2     p {  
3         color: red;  
4     }  
5 }
```

- 在实际开发中，常常结合多个 @media 规则以实现对各种不同大小设备的对应。

Try 
responsive.html

Q & A

Question and answer

总结

Sum Up

1. 导入 CSS 的 **3** 种方法;
2. CSS 的基本语法: **选择器**和**属性**;
3. CSS 的常用**属性**:
 - ① 与文字相关的**属性**、
 - ② 与背景相关的**属性**、
 - ③ 盒模型及相关**属性**、
 - ④ 与布局位置相关的**属性**。

THANK YOU!