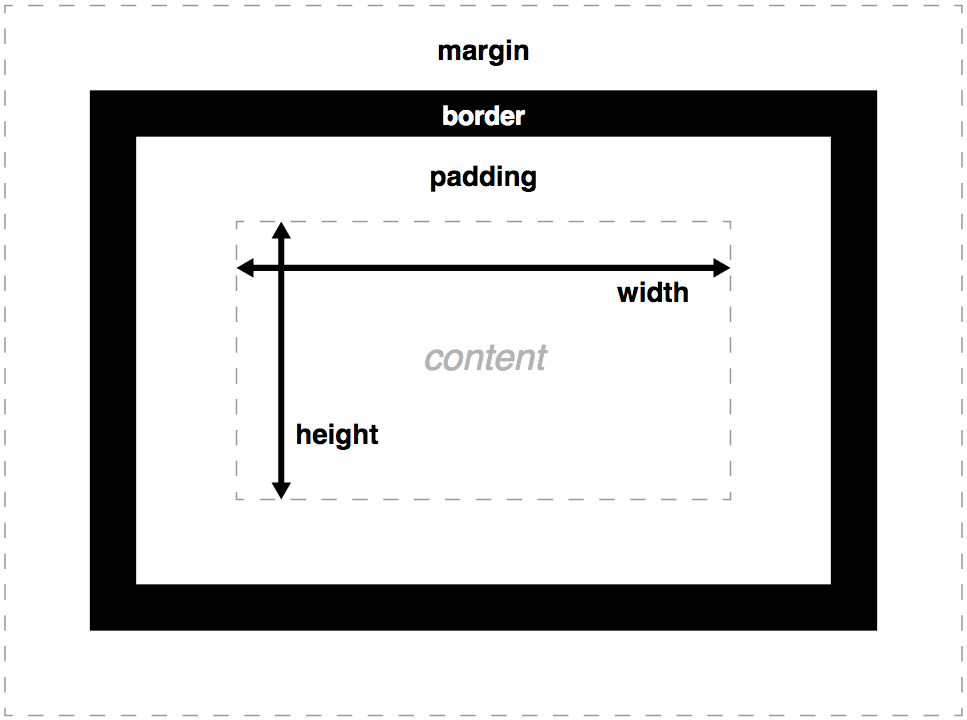
# 盒属性



* [width](https://developer.mozilla.org/zh-CN/docs/Web/CSS/width) 和[height](https://developer.mozilla.org/zh-CN/docs/Web/CSS/height) 设置内容框的宽/高。

注意：

盒子的高度不适用百分比长度

边界（border）也会忽略百分比宽度设置。

外边距（margin）有一个特殊的行为，称为 [外边距塌陷](https://developer.mozilla.org/zh-CN/docs/Web/CSS/CSS_Box_Model/Mastering_margin_collapsing)： 当两个盒子挨在一起时，二者之间的距离为两个挨着的外边距中最大的那个值，而不是二者的和。

约束宽高

通过属性 [min-width](https://developer.mozilla.org/zh-CN/docs/Web/CSS/min-width)、[max-width](https://developer.mozilla.org/zh-CN/docs/Web/CSS/max-width)、[min-height](https://developer.mozilla.org/zh-CN/docs/Web/CSS/min-height) 和 [max-height](https://developer.mozilla.org/zh-CN/docs/Web/CSS/max-height) 实现

width: 70%;

max-width: 1280px;

min-width: 480px;

margin: 0 auto;

auto（在这种情况下）共享父容器左右边距之间的可用空间使它居中：

父容器在最小和最大宽度限制内时，它将填满整个视口宽度；当父容器超过1280px宽度时，布局将保持在1280px宽，并居中。 当宽度低于480px时，视口将小于容器，必须滚动才能看得到完全的内容

而当容器变得比图像更窄时，图像开始溢流容器（因为它是一个固定的宽度）

display: block;

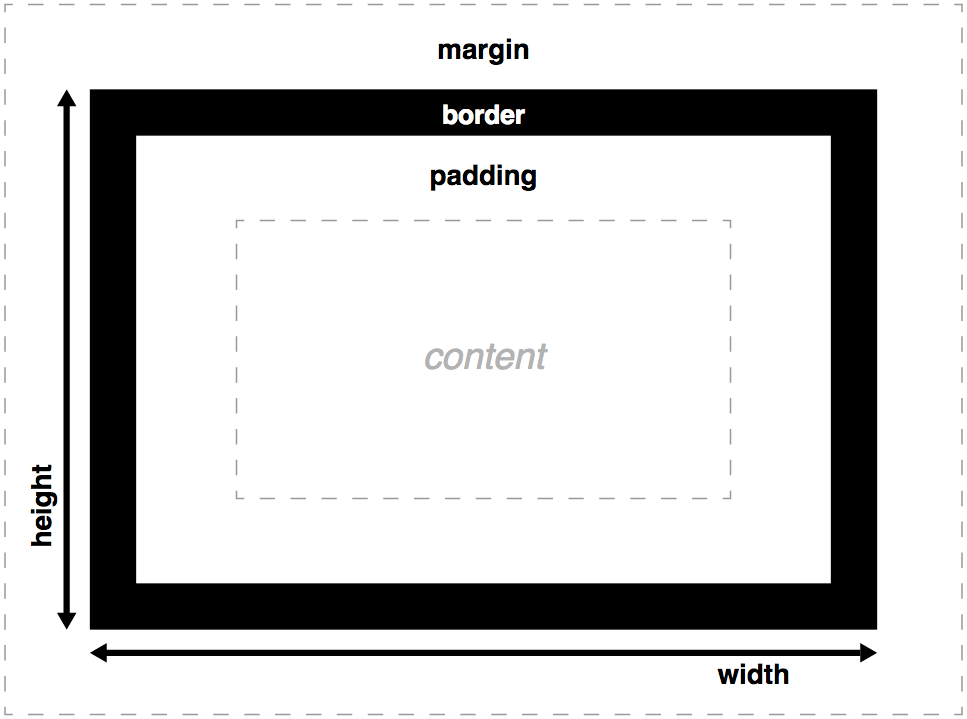
margin: 0 auto;

max-width: 100%;

前两条使图片展示行为像块元素并居中，第三条规则使图像的最大宽度与父容器的宽度相等，父容器宽度缩小到小于图像的宽度时，图像会一起缩小

改变盒模型

调整盒模型，用**box-sizing:border-box**：



Width=content+2\*padding+2\*border

盒子显示类型

**display**最常见的三个值：

**block**盒子之前或之后的内容出现在不同行，可设置宽高

**inline**跟随其他行内元素出现在同一行，且内容会像段落的文本行一样换行

**inline-block**不会前后自动换行，但可设置宽高，行内盒会自动换行

块级元素默认设置为 display: block; ，行内元素默认设置为 display: inline;

# CSS样式边框

**border-style：**

border-style: solid;

border-style: dashed;

border-style: dotted;

边界圆角半径

[border-radius](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-radius)实现圆角

border-radius: 20px;

可以有多个数值表达不同角

可以与椭圆形角混合使用

**椭圆形角**

（x半径与y半径不同）。两个不同的半径用正斜杠（/）分隔

border-radius: 10px / 20px;

边界图像

background-clip: padding-box;

需要指定边界，否则边界图像没有展示空间

[border-image](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image)

[border-image-source](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image-source)

border-image-source: url(border-image.png);

[border-image-slice](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image-slice)来设置所需大小的切片：

* 两个值:上和下，左和右。
* 三个值:上、左和右、下。
* 四个值:上、右、下、左。

[border-image-repeat](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image-repeat)

* stretch：默认;侧面的图像被拉伸来填满边界。这通常看起来很糟糕和像素化，所以不推荐。
* repeat：边图像被重复，直到边界被填满。根据具体情况，这可能看起来不错，但您可能会看到一些难看的图像片段。
* round： 边的图像被重复，直到边界被填满，它们都被稍微拉伸，这样就不会出现碎片。
* space：边图像被重复，直到边界被填满，每个拷贝之间添加了少量的间隔，这样就不会出现任何片段。
* border: 20px solid black;
* background-clip: padding-box;
* border-image-source: url(https://mdn.mozillademos.org/files/13060/border-image.png);
* border-image-slice: 40;
* border-image-repeat: round;

边界设置为20px宽，而图像的宽度为40——在这种情况下，浏览器只调整大小为20px宽

* [border-image-width](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image-width)：调整边界图像，如果图像宽小于[border-width](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-width)，它会贴着边界外侧，而不是填满边界。如果是它更大，那么它就会越过边界内侧，并开始重叠在内边距/内容上。
* [border-image-outset](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-image-outset)：边界内部和内边距之间的额外空间的大小——有点像“边界填充”。可以将边界图像移出一点

# 样式化表格

间距和布局

table {

table-layout: fixed;

width: 100%;

border-collapse: collapse;

border: 3px solid purple;

}

thead th:nth-child(1) {

width: 30%;

}

thead th:nth-child(2) {

width: 20%;

}

th, td {

padding: 20px;

}

**table-layout**:

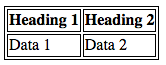
**auto**默认，表格大小随内容变化

**fixed**表格和列的宽度通过表格的宽度来设置，某一列的宽度仅由该列首行的单元格决定

其后的单元格内容并不会自适应当前列宽。任何一个包含溢出内容的单元格可以使用 [overflow](https://developer.mozilla.org/zh-CN/docs/Web/CSS/overflow)  属性控制是否允许内容溢出

[border-collapse](https://developer.mozilla.org/zh-CN/docs/Web/CSS/border-collapse)

默认表格



设置**border-collapse: collapse;**后，边框合为一条

https://mdn.mozillademos.org/files/13066/border-collapse.png

在[<th>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/th" \o "HTML <th> 元素 scope and headers 属性" \t "_blank)和[<td>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/td" \o "The Table cell HTML element (<td>) defines a cell of a table that contains data. It participates in the table model." \t "_blank)元素上设置了一些[padding](https://developer.mozilla.org/zh-CN/docs/Web/CSS/padding)

图形和颜色

thead, tfoot {

background: url(leopardskin.jpg);

color: white;

text-shadow: 1px 1px 1px black;

}

thead th, tfoot th, tfoot td {

background: linear-gradient(to bottom, rgba(0,0,0,0.1), rgba(0,0,0,0.5));

border: 3px solid purple;

}

**条纹图案**

tbody tr:nth-child(odd) {

background-color: #ff33cc;

}

tbody tr:nth-child(even) {

background-color: #e495e4;

}

样式化标题

caption {

font-family: 'Rock Salt', cursive;

padding: 20px;

font-style: italic;

caption-side: bottom;

color: #666;

text-align: right;

letter-spacing: 1px;

}

**caption-side**属性规定标题位置：

**top bottom** right left

# 高级区块效果

盒子阴影

与text-shadow等价属性**box-shadow**

box-shadow: 5px 5px 5px rgba(0,0,0,0.7);

* 1. 水平偏移（正为右，负为左）
  2. 垂直偏移（正向下，负向上）
  3. 模糊半径
  4. 基本颜色

[box-shadow](https://developer.mozilla.org/zh-CN/docs/Web/CSS/box-shadow)有一个inset关键字可用——把它放在一个影子声明的开始，使它变成一个内部阴影，而不是一个外部阴影。

box-shadow: inset 2px 2px 1px black,



过滤器

作用于盒内内容的确切形状，而不是盒子本身这一大块

.filter {

-webkit-filter: drop-shadow(5px 5px 1px rgba(0,0,0,0.7));

filter: drop-shadow(5px 5px 1px rgba(0,0,0,0.7));

}



# blend modes（多模型）

* [background-blend-mode](https://developer.mozilla.org/zh-CN/docs/Web/CSS/background-blend-mode), 用来将单个元素的多重背景图片和背景颜色设置混合在一起。
* [mix-blend-mode](https://developer.mozilla.org/zh-CN/docs/Web/CSS/mix-blend-mode), 用来将一个元素与它覆盖的那些元素各自所设置的背景（background）和内容(content)混合（重叠部分）在一起，颜色叠加

[background-blend-mode](https://developer.mozilla.org/zh-CN/docs/Web/CSS/background-blend-mode)

background-blend-mode: multiply;

[mix-blend-mode](https://developer.mozilla.org/zh-CN/docs/Web/CSS/mix-blend-mode)

mix-blend-mode: multiply;

## [-webkit-background-clip: text](https://developer.mozilla.org/zh-CN/docs/Learn/CSS/Styling_boxes/Advanced_box_effects#-webkit-background-clip_text)

将背景图像剪切到元素文本的形状

-webkit-background-clip: text;

-webkit-text-fill-color: transparent;