CSS盒子模型与定位

主讲人与课程设计：耕耕

wx：suency

Html核心解读 —— 元素分类 （根据css）

可以分为三大类：**块级元素，内联元素，内联块级元素**

- 块级元素
  - 每个块级元素都从新的一行开始，并且其后的元素也另起一行。（一个块级元素**独占一行**）
  - 元素的**高度(height)，宽度(width)，行高(line-height)**以及顶和底边距(margin,padding)都**可设置**
  - 元素**宽度**在不设置的情况下，是它本身**父容器的100%**（和父元素的宽度一致）常用的块元素有：** ,** <p>,<h1>...<h6>,<ul>,<ol>,<dl>,<table>,<address>,<blockquote>,<form>设置display:block，可以将元素转换块级元素

Html核心解读 —— 元素分类 （根据css）

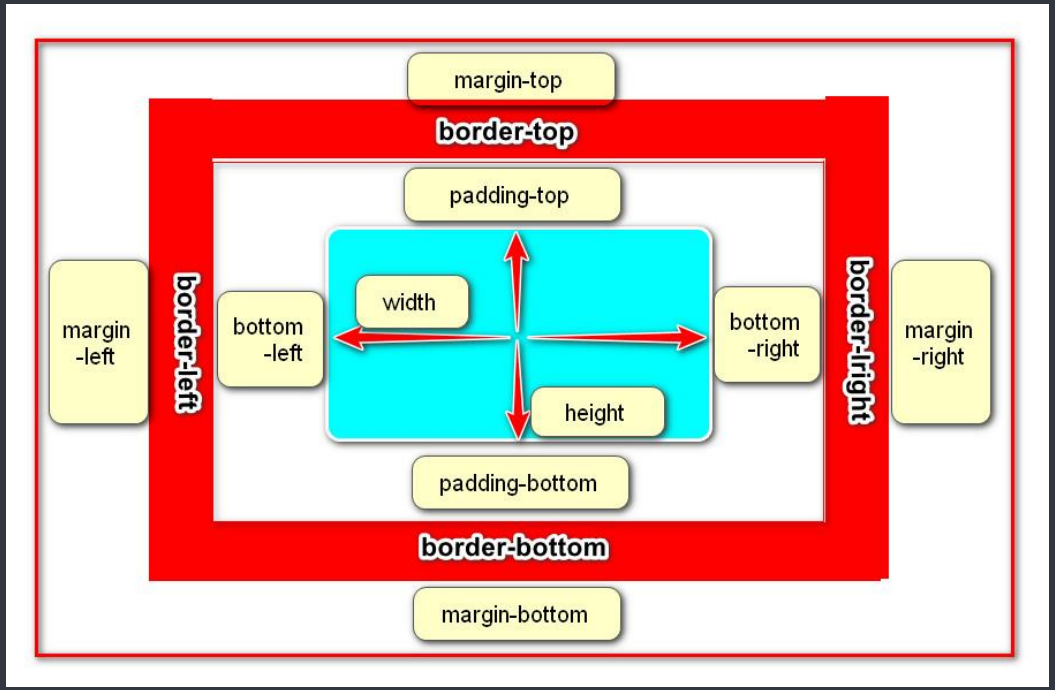
- 内联元素
  - 和其它元素都在**一行上**
  - 元素的**高度，宽度**及顶部和底部边距**不可设置**
  - 元素的宽度就是他包含的文字和图片的宽度，不可改变。
  - 常用的内联元素有：<a>, ****，
,,
  - 设置display:inline；可以将块级元素转换为内联元素

- 内联块级元素
  - 和其它元素都在**一行上**
  - 元素的**高度，宽度**及顶部和底部边距**可设置**
  - 常用的内联块状元素有：**<img>** , <input>display:inline-block;
  - float:left/right; position: absolute/fixed;可以将元素设置为内联块级元素（BFC）

CSS 盒子模型(Box Model)

所有HTML元素可以看作盒子，在CSS中，"box model"这一术语是用来设计和布局时使用。

CSS盒模型本质上是一个盒子，封装周围的HTML元素，它包括：**边距(margin)**，**边框(border)**，**填充(padding)**，和**实际内容(content)**。每个盒子有自己**大小和位置**外，还影响着其他盒子的大小和位置




```
<div>content</div>
```

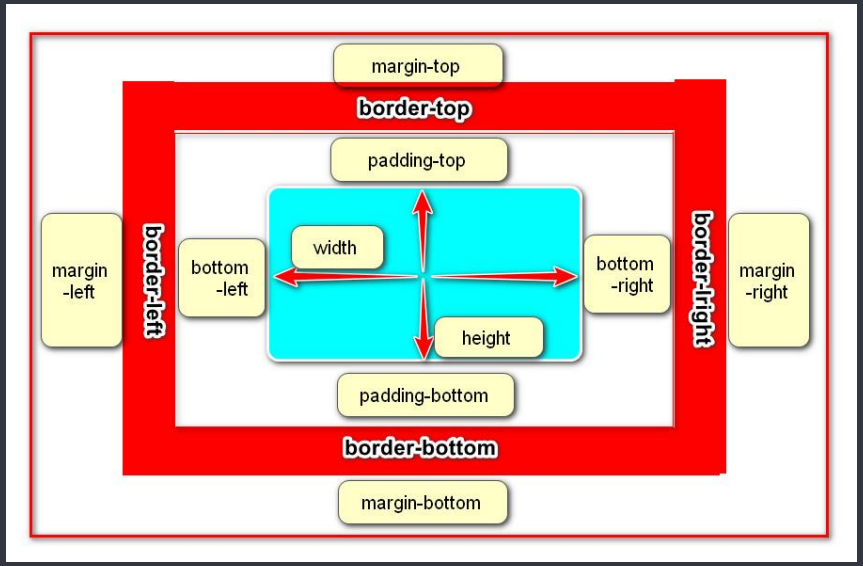
盒子边框（border）

border 属性来定义盒子的边框，该属性包含3个子属性：border-style(边框样式)，border-color(边框颜色)，border-width(边框宽度)。

语法：

border : border-width || border-style || border-color

如：`border: 50px solid blue;`



https://www.w3school.com.cn/css/css_border.asp

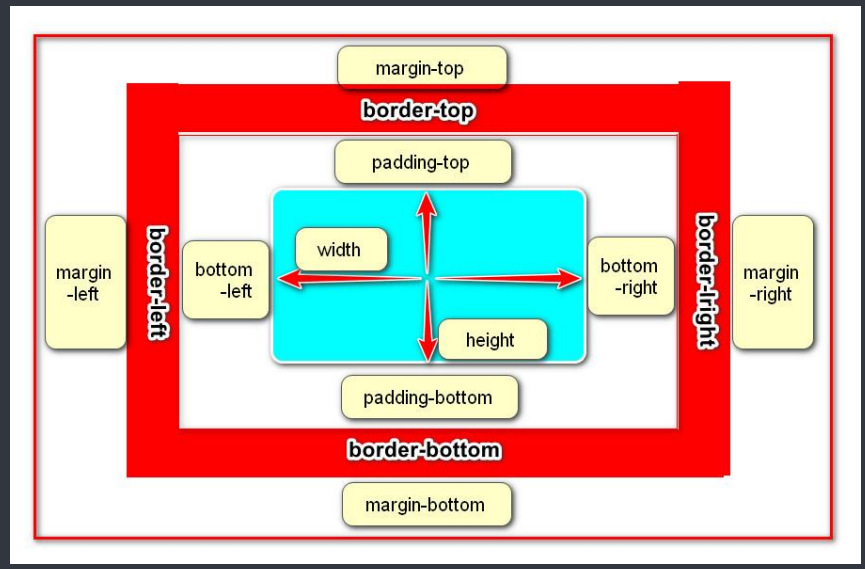
内边距（padding）

CSS padding 属性定义元素的内边距。

padding-top: 上内边距
padding-right: 右内边距
padding-bottom:下内边距
padding-left: 左内边距

```
          上    右    下    左
padding: 10px 20px 30px 40px;
```

```
          上下   左右
padding: 10px 20px;
```



https://www.w3school.com.cn/css/css_padding.asp

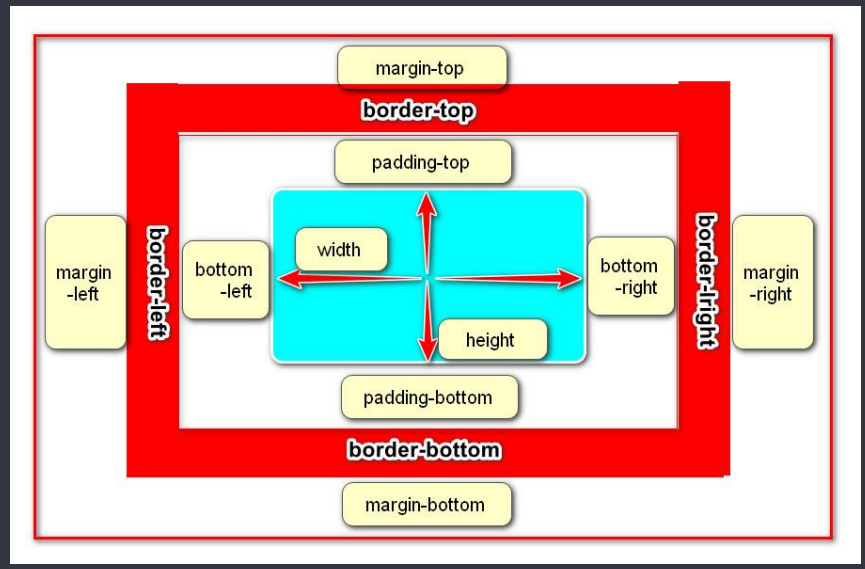
外边距（margin）

设置外边距会在元素之间创建“空白”，定义了元素与其他相邻元素的距离， 这段空白通常不能放置其他内容。

margin-top:上外边距
margin-right:右外边距
margin-bottom:下外边距
margin-left:上外边距

```
          上    右    下    左
margin: 10px 20px 30px 40px;

          上下   左右
margin: 10px 20px;
```



https://www.w3school.com.cn/css/css_margin.asp

两种盒子模型类型 (重要)

所有HTML元素可以看作盒子，在CSS中，"box model"这一术语是用来设计和布局时使用。

1. box-sizing: content-box (默认)

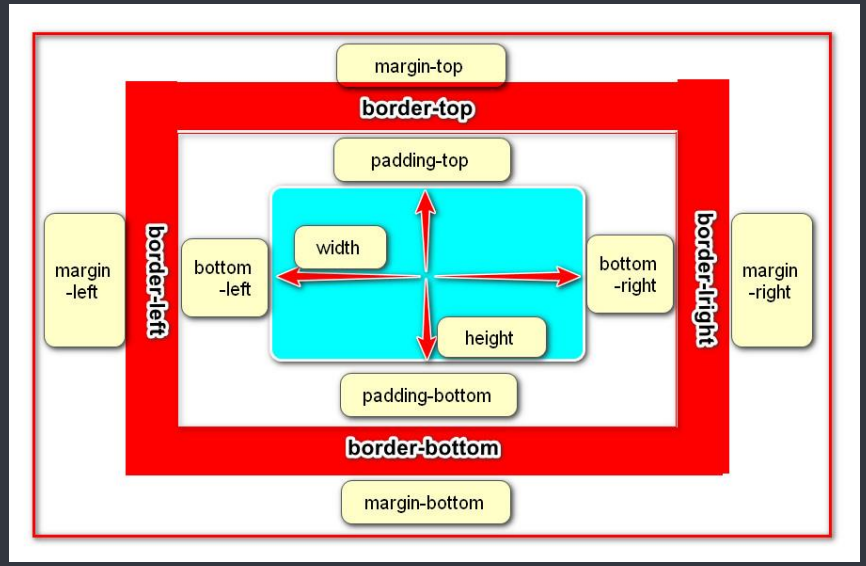
此时，**元素的width=content的宽度**

自己会膨胀

2. box-sizing: border-box

此时，**元素的width= content的宽度+ padding + border**

自己不会膨胀



注意：这一切跟margin没有关系

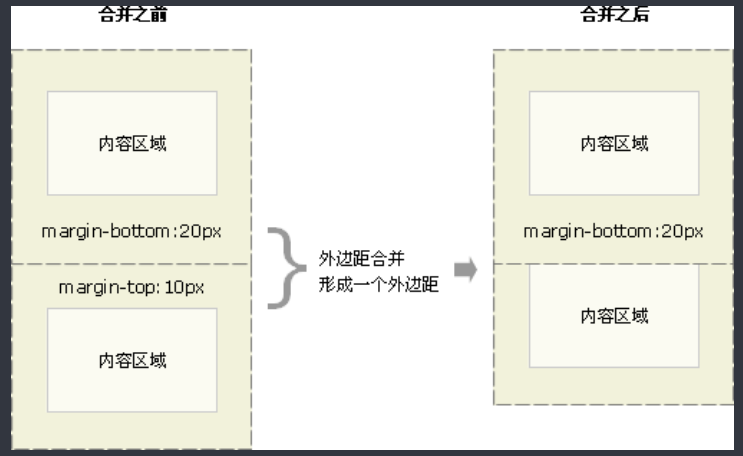
补充

1. 对于**行级元素**， **margin-top**和**margin-bottom**无效

2. 对于**行级元素**， **padding-top**和**padding-bottom**显示上有效果， 但是对周围元素没有影响， 你也可以理解为无效

3. 外边距合并

当上下相邻的两个块元素相遇时，如果上面的元素有下外边距margin-bottom，下面的元素有上外边距margin-top，则他们之间的垂直间距不是margin-bottom与margin-top之和，而是两者中的较大者。这种现象被称为相邻块元素垂直外边距的合并（也称外边距塌陷）

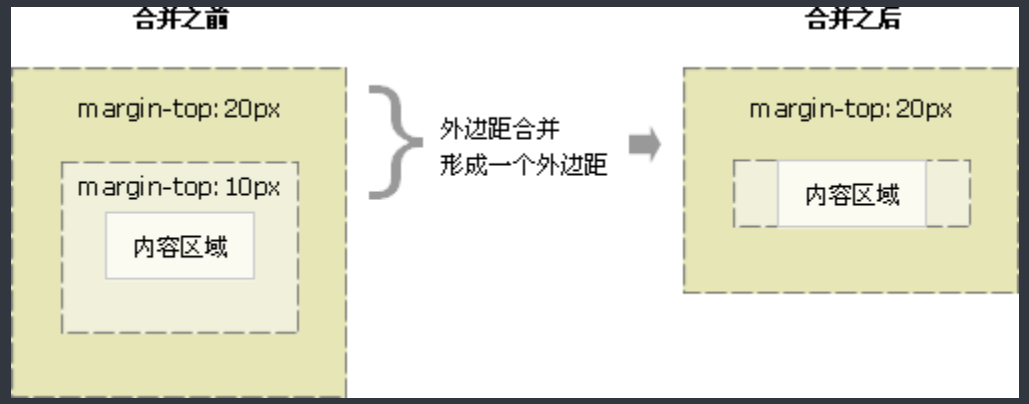


解决方案：涉及到BFC的内容，之后再讲解

补充 —— 练习

4. 嵌套块元素垂直外边距的合并

对于两个嵌套关系的块元素，如果父元素没有上内边距及边框，则父元素的上外边距会与子元素的上外边距发生合并，合并后的外边距为两者中的较大者，即使父元素的上外边距为0，也会发生合并

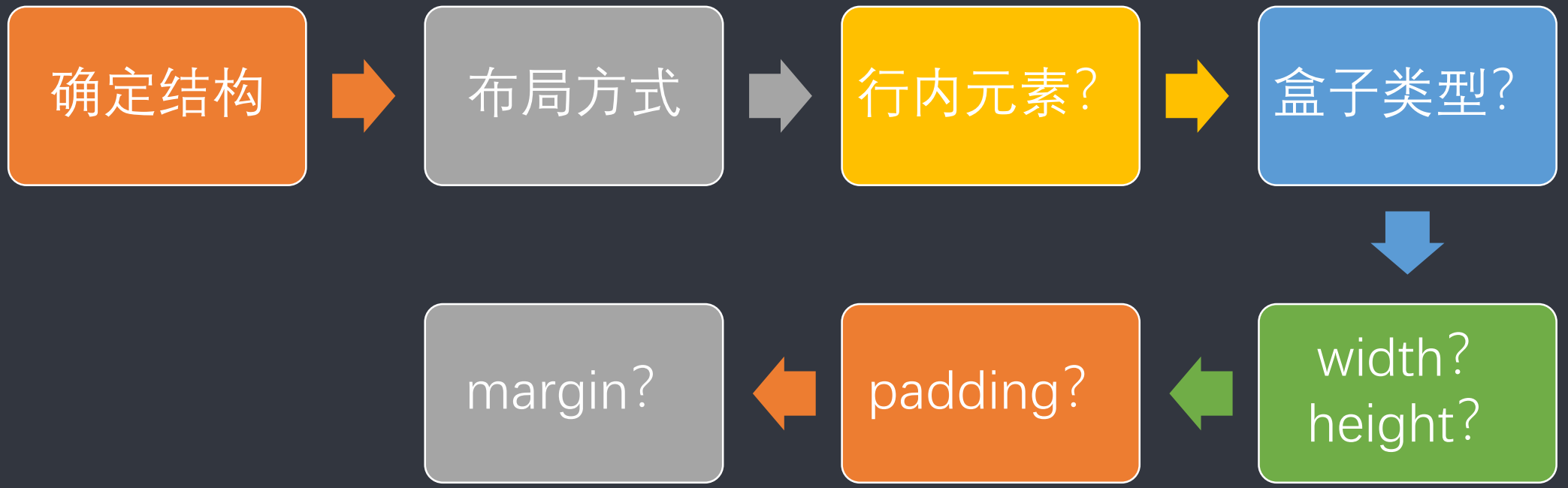


解决方案: 给父元素设置边框或者内边距

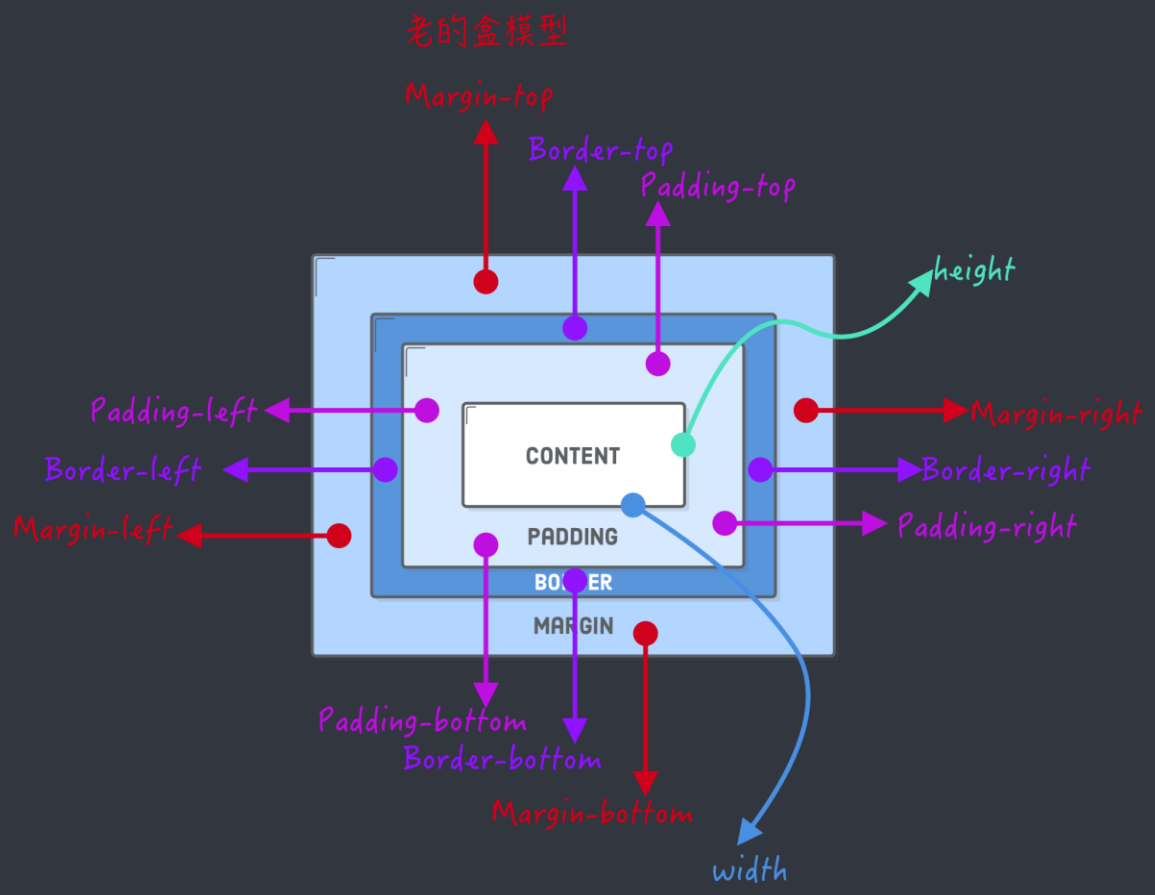
盒子模型布局稳定性

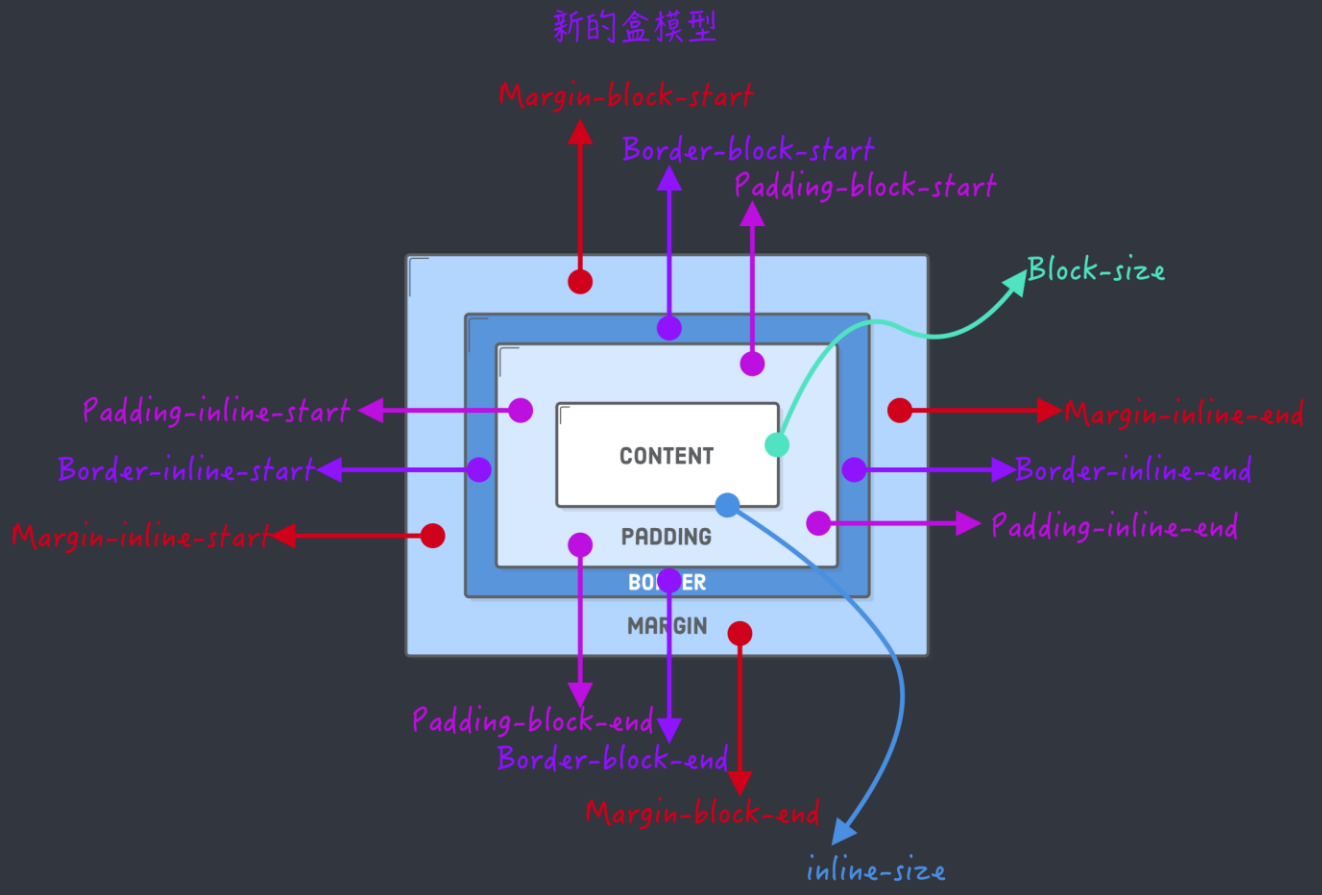
建议优先级如下 **width > padding > margin**

顺序



新旧对比 —— 了解


老的盒模型
Margin-top
Border-top
Padding-top
height
Padding-left
Border-left
Margin-left
CONTENT
PADDING
BORDER
MARGIN
Margin-right
Border-right
Padding-right
Padding-bottom
Border-bottom
Margin-bottom
width


新的盒模型
Margin-block-start
Border-block-start
Padding-block-start
Block-size
Padding-inline-start
Border-inline-start
Margin-inline-start
CONTENT
PADDING
BORDER
MARGIN
Margin-inline-end
Border-inline-end
Padding-inline-end
Padding-block-end
Border-block-end
Margin-block-end
inline-size

CSS定位

一个属性，5个值

Position（定位）

参数	描述
absolute	绝对定位；脱离文档流的布局，遗留下来的空间由后面的元素填充。定位的起始位置为最近的父元素(postion不为static)，否则为Body文档本身。
relative	相对定位；不脱离文档流的布局，只改变自身的位置，在文档流原先的位置遗留空白区域。定位的起始位置为此元素原先在文档流的位置。
fixed	固定定位；类似于absolute，但不随着滚动条的移动而改变位置。
static	默认值；默认布局。 忽略 top, bottom, left, right和z-index
sticky	类似relative和fixed的结合体

共同配合5个属性 （static除外）

top, bottom, left, right和z-index

static 属性值

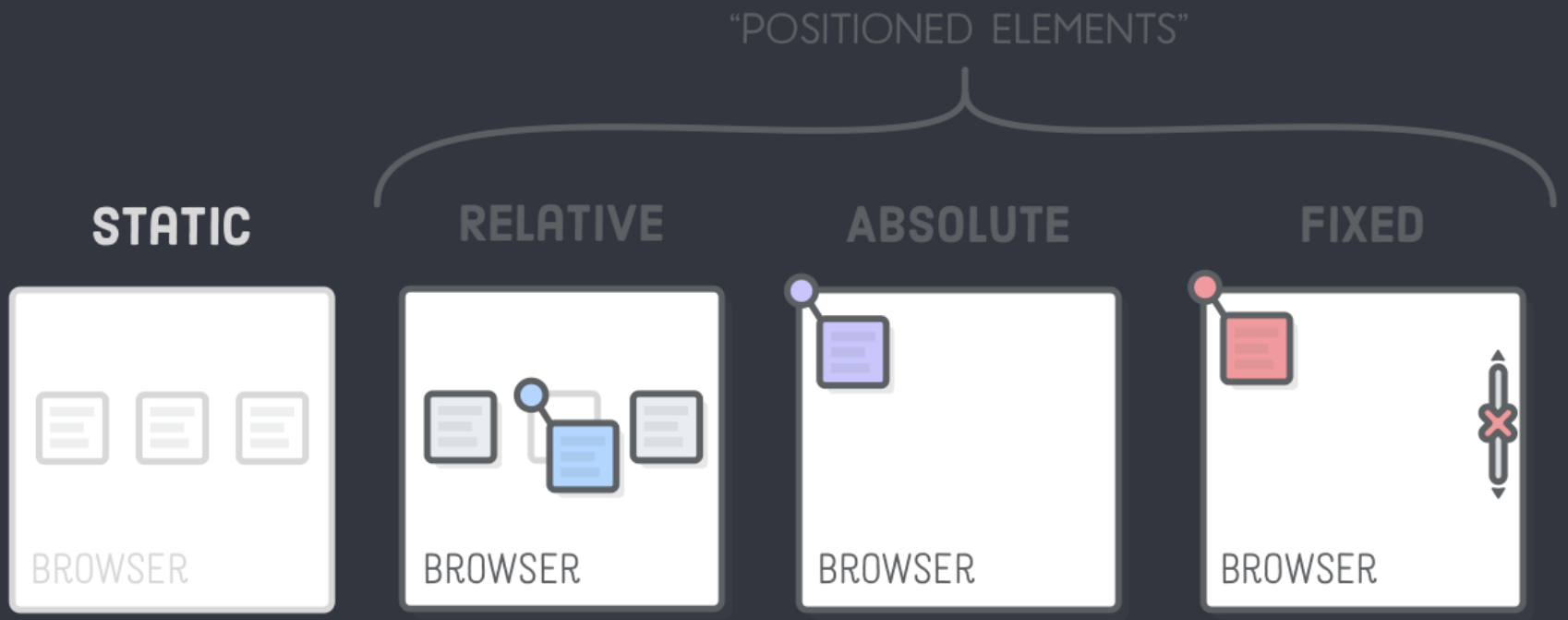
static是position属性的默认值。如果省略position属性，浏览器就认为该元素是static定位。

这时，浏览器会按照源码的顺序，决定每个元素的位置，这称为"**正常的页面流**"（normal flow）。每个块级元素占据自己的区块（block），元素与元素之间不产生重叠，这个位置就是元素的默认位置。

注意，static定位所导致的元素位置，是浏览器自主决定的，所以这时**top、bottom、left、right、z-index**这五个属性无效。

relative， absolute， fixed属性值

都是相对于某个**基点**的定位

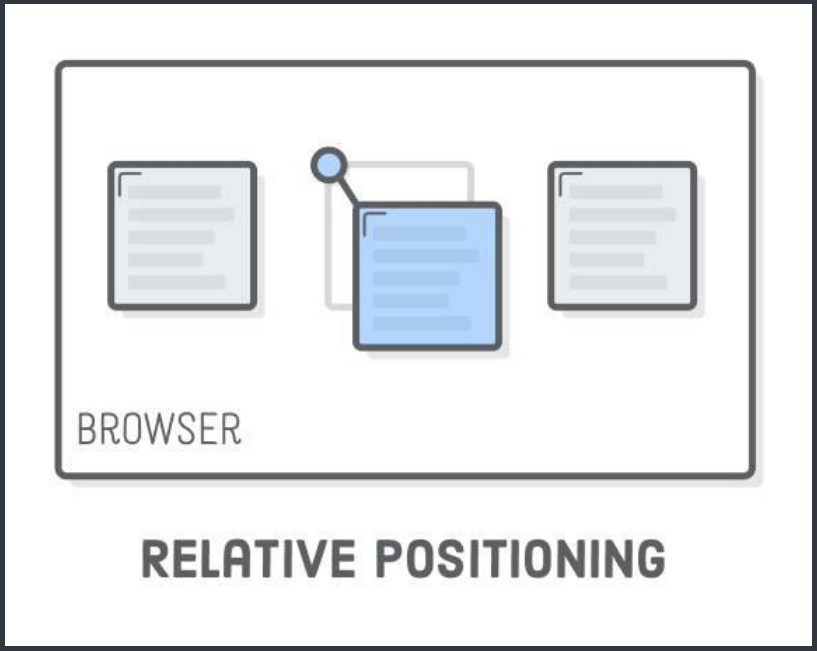


相对定位(relative)

不脱离文档流的布局，只改变自身的位置，在文档流原先的位置遗留空白区域

定位的起始位置为此**元素原先在文档流的位置**

只是可以通过4个属性值改变自己的位置，**top, bottom, left, right**



绝对定位(absolute)

脱离文档流的布局，遗留下来的空间由后面的元素填充

定位的**起始位置**为最近的**父元素**(postion不为static)，否则为**Body**文档本身。

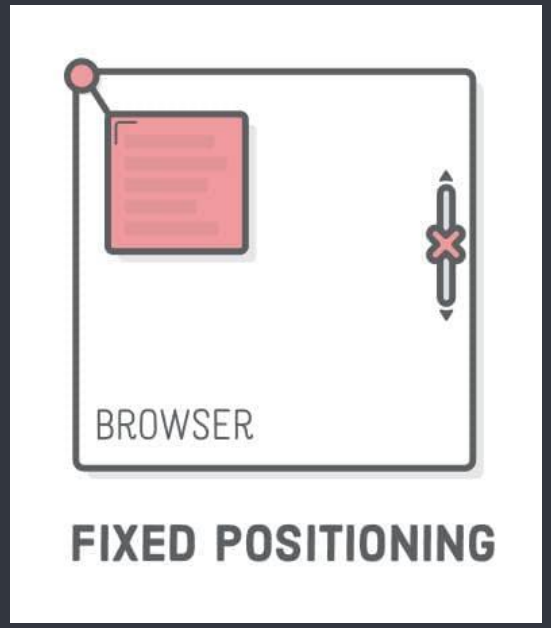
总结：起始位置看爸爸的位置，如果爸爸定位是static，直接看到老祖宗body，然后通过**top, bottom, left, right**，改变自己的整体位置

注意：不设置top, bottom, left, right，起始位置走**正常文档流**

固定定位(fixed)

fixed表示，相对于视口（viewport，浏览器窗口）进行偏移，即定位基点是**浏览器窗口**。这会导致元素的位置不随页面滚动而变化，好像**固定**在网页上一样。

总结：起始位置**不看任何人**，只看浏览器窗口，然后通过**top, bottom, left, right**，改变自己的整体位置



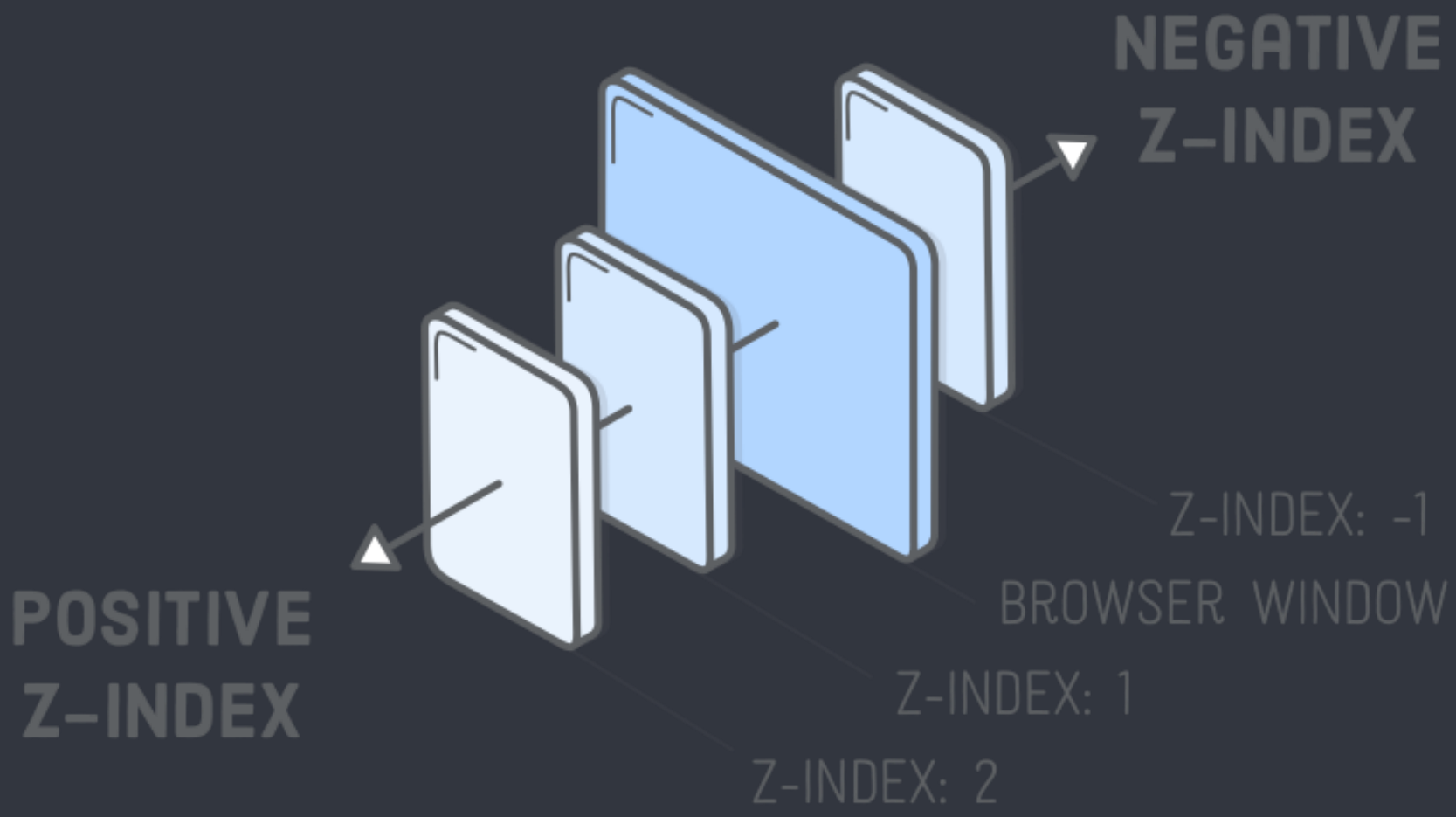
注意：不设置top, bottom, left, right，起始位置走**正常文档流**

sticky 属性值

sticky跟前面四个属性值都不一样，它会产生动态效果，很像relative和fixed的结合：一些时候是relative定位（定位基点是自身默认位置），另一些时候自动变成fixed定位（定位基点是视口）。

总结：不设置**top, bottom, left, right**，跟relative一样。它的具体规则是，当页面滚动，父元素开始脱离视口时（即部分不可见），就跟fixed效果一样。其他情况跟relative一样。

Z-index


NEGATIVE
Z-INDEX
POSITIVE
Z-INDEX
Z-INDEX: -1
BROWSER WINDOW
Z-INDEX: 1
Z-INDEX: 2