什么是层叠样式表？

给文字添加修饰，简化代码

层叠：将多个样式叠加在一块，控制html元素

样式表：文档的体现 外观

<html>

<head>

<meta charset="utf-8">

<title></title>

<style>

#navigation{

text-align: center;

}

.item{

text-decoration: none;

color: black;

}

</style>

</head>

<body>

* <div id="navigation"> ˌnævɪˈɡeɪʃn 导航

<a href="#" class="item">首页</a>

<span>|</span>

<a href="#" class="item">办公家具</a>

<span>|</span>

<a href="#" class="item">数码科技</a>

<span>|</span>

<a href="#" class="item">母婴</a>

<span>|</span>

<a href="#" class="item">团购</a>

<span>|</span>

<a href="#" class="item">秒杀活动</a>

<span>|</span>

</div>

</body>

</html>

Style是什么？

Style属性声明样式 <style> {}</style> 定义文档样式信息（当标签）

如果直接使用style属性声明样式 <h1 style="font-size:x-large;color:red;">不加{} (标签内)

什么是标签？

<>里的

长度单位：em（font-size:2em）是一个与一个字符高度大致相同的单位

　　　　　　　　     px（font-size：12px）是一个像素单位

　　　　　　　　　 pt（font-size：12pt）是一个磅的单位

　　　　　　　　　 %（font-size：80%）是一个百分比

　颜色的值是一个关键字或一个RGB格式的数字，16个关键字：aqua、black、blue、fuchsia、gray、green、lime、maroon、navy、olive、purple、red、silver、teal、white、yellow、RGB颜色可以有以下四种形式：

背景图

Background（‘ur’l） body{Background:url()}

Body{background:url(‘1.img’)}

什么是内联样式表？

**外部样式表:** 当样式需要应用于很多页面时，外部样式表将是理想的选择。在使用外部样式表的情况下，你可以通过改变一个文件来改变整个站点的外观。每个页面使用 <link> 标签链接到样式表。<link> 标签在（文档的）头部：<head> <link rel="stylesheet" type="text/css" href="mystyle.css" /> </head>浏览器会从文件 mystyle.css 中读到样式声明，并根据它来格式文档。外部样式表可以在任何文本编辑器中进行编辑。**文件不能包含任何的 html 标签。**样式表应该以 .css 扩展名进行保存。下面是一个样式表文件的例子：hr {color: sienna;} p {margin-left: 20px;} body {background-image: url("images/back40.gif");}不要在属性值与单位之间留有空格。假如你使用 “margin-left: 20 px” 而不是 “margin-left: 20px” ，它仅在 IE 6 中有效，但是在 Mozilla/Firefox 或 Netscape 中却无法正常工作。特征：  
1、有一个单独的CSS文件存在！[如：001.css]

2、首先，通过【格式】→【样式表连接】与CSS文件[001.css]建立连接！

3、在合适的地方使用 class=“样式名” 调用具体的样式效果！

**内部样式表:** 当单个文档需要特殊的样式时，就应该使用内部样式表。你可以使用 <style> 标签在文档头部定义内部样式表，就像这样:<head> <style type="text/css"> hr {color: sienna;} p {margin-left: 20px;} body {background-image: url("images/back40.gif");} </style> </head>

**.嵌入式样式（属于内部样式表）**

特征：

1、样式的属性内容以代码的形式[放在 ]写在网页代码中！

2、首先，通过【格式】→【样式】设置样式的属性内容！

3、在合适的地方使用 class=“样式名” 调用具体的样式效果！

如：

.main{ width:1002px; margin:0 auto;}

## 内联样式: 由于要将表现和内容混杂在一起，内联样式会损失掉样式表的许多优势。请慎用这种方法，例如当样式仅需要在一个元素上应用一次时。要使用内联样式，你需要在相关的标签内使用样式（style）属性。2.1.内联式样式（属于内部样式表）

特征：

1、样式的属性内容直接跟在将要修饰的文字标记里[如：

2 style=“font-size:10px;font-color:#ff0000

css选择器：HTML选择器、类选择器、ID选择器、组合选择器、关联选择器，伪类和伪元素

HTML选择器：用来改变一个指定标签的样式。p { text-indent:3em }

　　　　　　　　　　　　　　　　　　　　　　　h1{ color:red }

类选择器：在HTML<p>标签使用class属性引用<p>指定一个类名。

　　p.dark-row { background-color ：#EAEAEA }/\*定义<p>元素的一个类的背景元素\*/

　　p.light-row { background-color ：#F8F8F8}/\*\*定义<p>元素的另一个背景元素/

<p class=" one two three">为p元素定义了三个类one、two、three、

ID选择器id属性指定了某一单元素，id选择器用来对这个单一元素定义单独样式。id选择器的应用类似，把class换成id即可，id名称前要加一个#　　　　#main  < text-indent:3em>　　　　　　<p id="main">文本缩进3em</p>

<!--  在HTML的p标记指定id属性值为main   -->

关联选择器：table a { color ：red} 只有在<table>内的链接<a>改变了样式

组合选择器：只要用逗号隔开选择器就可以。h1,h2,h3,h4,h5,h6 { color：red; font-family：sans-serif} /\*使用组合器修改标题\*

伪元素选择器：指对同一个HTML元素不同状态的一种定义方式： HTML 标签：伪元素{属性：值；}

　　　　　　分优先级：关联>ID>类>HTML，从左到右逐级递减

css常见的样式属性和值:字体，文本，背景，位置，边框、列表、以及其他一些样式表属性。

ext-decoration:none  //默认，定义标准的文本，没有任何样式，正常显示

text-decoration:underline  //定义文本下的一条线

text-decoration:overline  //定义文本上的一条线

text-decoration:line-through  //定义文本中间的一条线

text-decoration:blink  //定义闪烁的文本， IE、Chrome 或 Safari 不支持 "blink" 属性值。

text-deration:inherit  //从父元素继承text-decoration的值，任何的版本的 Internet Explorer （包括 IE8）都不支持属性值 "inherit"。

什么是块元素？

一块区域的元素

每个块元素通常都会独自占据一整行或多整行，可以对其设置宽度、高度、对齐等属性，常用于网页布局和网页结构的搭建。  
常见的块元素有 h1~h6、p、div、ul、ol 、li等，其中div标签是最典型的块元素。  
块级元素的特点：  
（1）总是从新行开始  
（2）高度，行高、外边距以及内边距都可以控制。  
（3）宽度默认是容器的100%  
（4）可以容纳内联元素和其他块元素。

什么是行内元素？

一行内所有元素

行内元素：  
行内元素（内联元素）不占有独立的区域，仅仅靠自身的字体大小和图像尺寸来支撑结构，一般不可以设置宽度、高度、对齐等属性，常用于控制页面中文本的样式。  
常见的行内元素有 a，strong，b，em，i，del，s，u，span，等，其中span标签是最典型的行内元素。  
行内元素的特点：  
(1)和相邻内元素在一行上。  
(2)高、宽无效，但水平方向的padding和margin可以设置，垂直方向的无 效。  
（3）默认宽度就是它本身内容的宽度。  
（4）行内元素只能容纳文本或则其他行内元素。（a特殊）

**常见的行内块元素**

在行内元素中有几个特殊的标签——img、input/、td，可以对它们设  
置宽高和对齐属性，有些资料可能会称它们为行内块元素。  
  
行内块元素的特点：  
（1）和相邻行内元素（行内块）在一行上,但是之间会有空白缝隙。  
（2）默认宽度就是它本身内容的宽度。  
（3）高度，行高、外边距以及内边距都可以控制。

标签显示模式互换（display）  
块转行内：display:inline;  
行内转块：display:block;  
块、行内元素转换为行内块：display:inline-block;

# HTML可以将元素分类方式分为行内元素、块状元素和行内块状元素三种。

# 使用display属性能够将三者任意转换

(1)display:inline;转换为行内元素

　　(2)display:block;转换为块状元素

　　(3)display:inline-block;转换为行内块状元素

### 1.行内元素

　　行内元素最常使用的就是span，其他的只在特定功能下使用，修饰字体<b>和<i>标签，还有<sub>和<sup>这两个标签可以直接做出平方的效果，而不需要类似移动属性的帮助，很实用。

　　行内元素特征：(1)设置宽高无效

　　　　　　　　　(2)对margin仅设置左右方向有效，上下无效；padding设置上下左右都有效，即会撑大空间,行内元素尺寸                                  由内含的内容决定，盒模型中 padding, border 与块级元素并无差异，都是标准的盒模型，但是 margin                                    却只有水平方向的值，垂直方向并没有起作用。行内元素的水平方向的padding-left,padding-right,margin-                                           left,margin-right 都产生边距效果，但是竖直方向的padding-top,padding-bottom,margin-top,margin-bottom都                                        不会产生边距效果。padding设置上下左右都有效，即会撑大空间但是不会产生边距效果。

　　　　　　　　　(3)不会自动进行换行

### 2.块状元素

　　块状元素代表性的就是div，其他如p、nav、aside、header、footer、section、article、ul-li、address等等，都可以用div来实现。不过为了可以方便程序员解读代码，一般都会使用特定的语义化标签，使得代码可读性强，且便于查错。

　　块状元素特征：(1)能够识别宽高

　　　　　　　　　(2)margin和padding的上下左右均对其有效

　　　　　　　　　(3)可以自动换行

　　　　　　　　　(4)多个块状元素标签写在一起，默认排列方式为从上至下

### 3.行内块状元素

　　行内块状元素综合了行内元素和块状元素的特性，但是各有取舍。因此行内块状元素在日常的使用中，由于其特性，使用的次数也比较多。

　　行内块状元素特征：(1)不自动换行

　　　　　　　　　　　(2)能够识别宽高

　　　　　　　　　　　(3)默认排列方式为从左到右

　　在HTML5中，程序员可以自定义标签，在任意定义标签中，加入display:block;即可，当然也可以是行内或行内块状。

元素(block element)

　　\* address - 地址  
　　\* blockquote - 块引用  
　　\* center - 举中对齐块  
　　\* dir - 目录列表  
　　\* div - 常用块级容易，也是css layout的主要标签  
　　\* dl - 定义列表  
　　\* fieldset - form控制组  
　　\* form - 交互表单  
　　\* h1 - 大标题  
　　\* h2 - 副标题  
　　\* h3 - 3级标题  
　　\* h4 - 4级标题  
　　\* h5 - 5级标题  
　　\* h6 - 6级标题  
　　\* hr - 水平分隔线  
　　\* isindex - input prompt  
　　\* menu - 菜单列表  
　　\* noframes - frames可选内容，（对于不支持frame的浏览器显示此区块内容  
　　\* noscript - 可选脚本内容（对于不支持script的浏览器显示此内容）  
　　\* ol - 排序表单  
　　\* p - 段落  
　　\* pre - 格式化文本  
　　\* table - 表格  
　　\* ul - 非排序列表

内联元素(inline element)

　　\* a - 锚点  
　　\* abbr - 缩写  
　　\* acronym - 首字  
　　\* b - 粗体(不推荐)  
　　\* bdo - bidi override  
　　\* big - 大字体  
　　\* br - 换行  
　　\* cite - 引用  
　　\* code - 计算机代码(在引用源码的时候需要)  
　　\* dfn - 定义字段  
　　\* em - 强调  
　　\* font - 字体设定(不推荐)  
　　\* i - 斜体  
　　\* img - 图片  
　　\* input - 输入框  
　　\* kbd - 定义键盘文本  
　　\* label - 表格标签  
　　\* q - 短引用  
　　\* s - 中划线(不推荐)  
　　\* samp - 定义范例计算机代码  
　　\* select - 项目选择  
　　\* small - 小字体文本  
　　\* span - 常用内联容器，定义文本内区块  
　　\* strike - 中划线  
　　\* strong - 粗体强调  
　　\* sub - 下标  
　　\* sup - 上标  
　　\* textarea - 多行文本输入框  
　　\* tt - 电传文本  
　　\* u - 下划线  
　　\* var - 定义变量

可变元素

　　可变元素为根据上下文语境决定该元素为块元素或者内联元素。  
　　\* applet - java applet  
　　\* button - 按钮  
　　\* del - 删除文本  
　　\* iframe - inline frame  
　　\* ins - 插入的文本  
　　\* map - 图片区块(map)  
　　\* object - object对象  
　　\* script - 客户端脚本

## css浮动 float属性详解

一、什么是浮动？  
浮动，顾名思义，就是漂浮的意思。指的是一个元素脱离文档流，悬浮在父元素之上的现象。  
二、如何产生浮动？  
给元素本身添加float属性

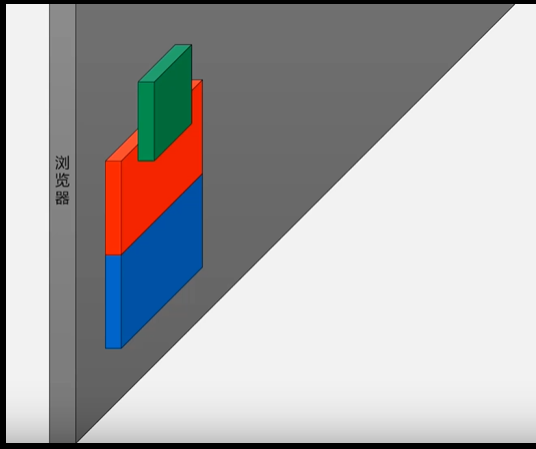
float值：

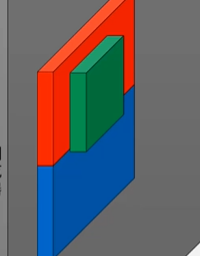
| **left** | **元素向左浮动。** |
| --- | --- |
| right | 元素向右浮动。 |
| inherit | 规定应该从父元素继承 float 属性的值。 |
| none | 默认值。元素不浮动，并会显示在其在文本中出现的位置。 |

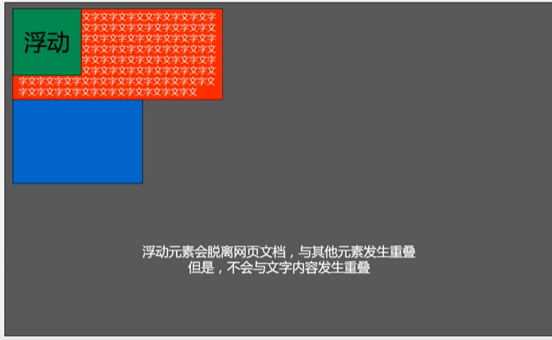
三、浮动有什么作用？  
作用：在html文档流中，分为行元素、块元素和行内块元素，

行元素和行内块元素都是以横向排列的，而块元素则是以流的形式从上往下排列的，当我们想使块元素横向排列时，就用到了我们的浮动(float)。

容器内的元素无论大小如何，都是独占一行，排列 由上而下



写在后面的

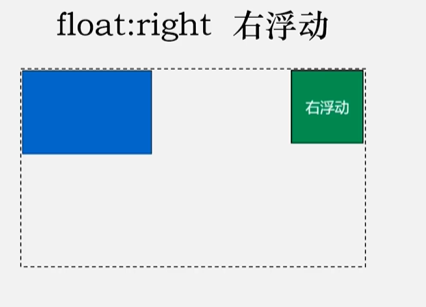
* 

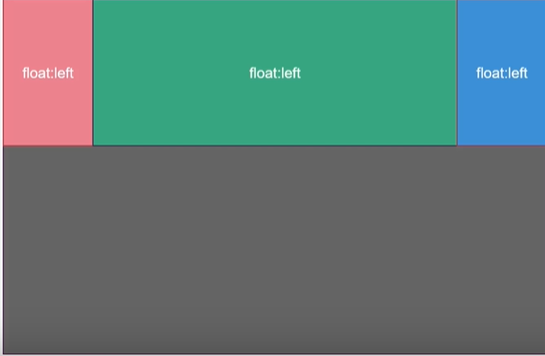
如果做三个容器

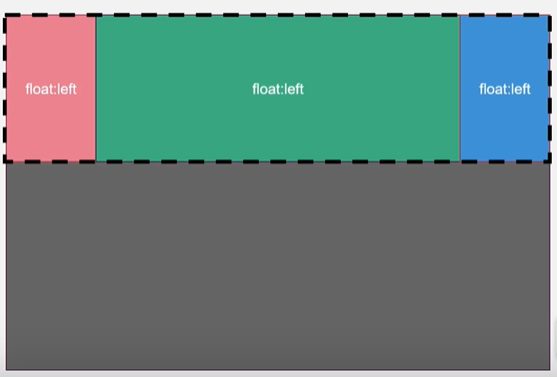
如何让他们3个容器横向排列在一行

没浮动 写3个div





浮动元素最大特点：脱离文档流，与其他元素产生重叠现象如何让后面的元素不被遮挡？

添加父元素

父元素和后面不产生关系



对于普通元素来讲，子元素可以将父元素撑开

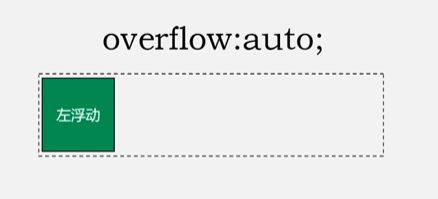


父元素高度为0 3元素浮动 不占空间 没办法撑开父元素

清楚浮动元素的影响

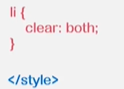
为了让父元素高度保持正常

给父元素增加 overflow：auto （溢出）（绿色超出边界）

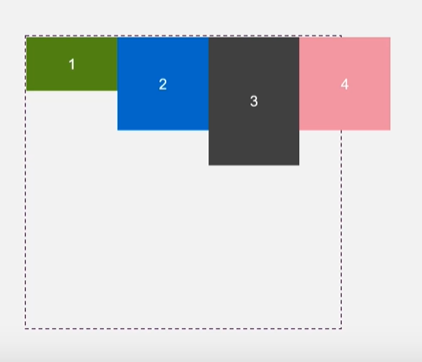
父元素做出自我调整 不影响后面布局

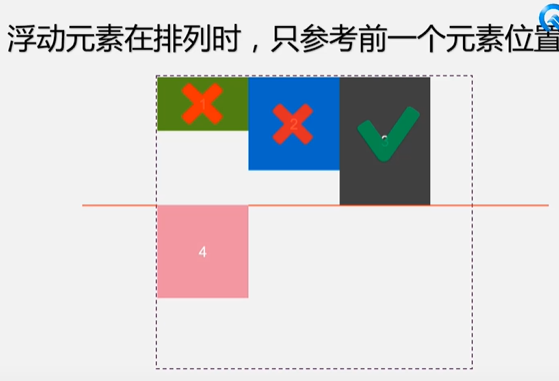
当前元素不受前面浮动影响 

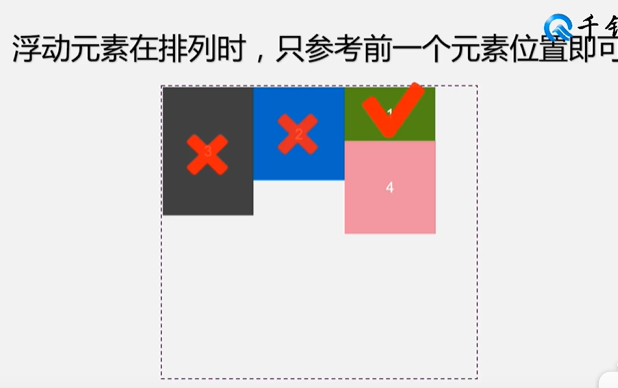


Clear ：both 

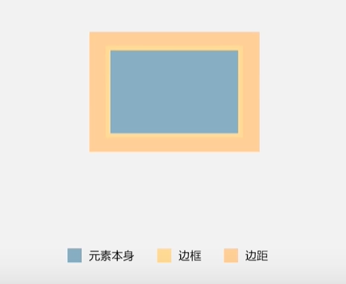
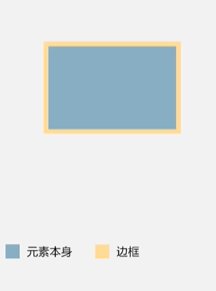
浮动元素在排列时，之参考钱一个元素位置即可

第4个换行出现在哪？

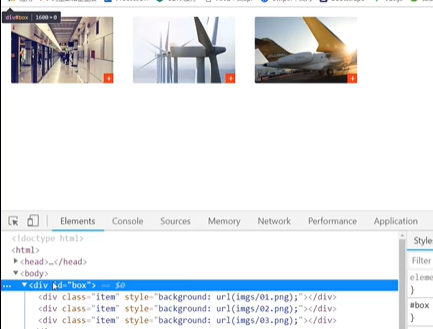


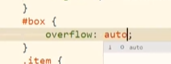
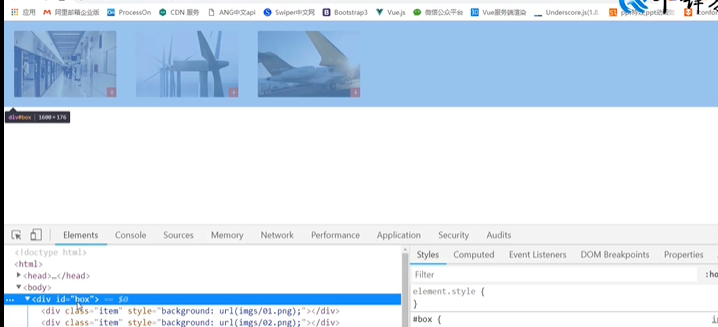




元素间距离靠边距

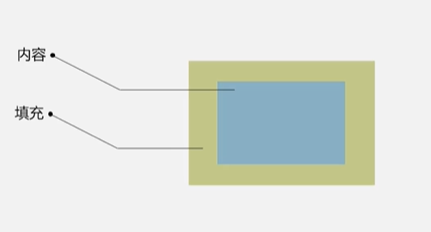
 有些元素自带边距

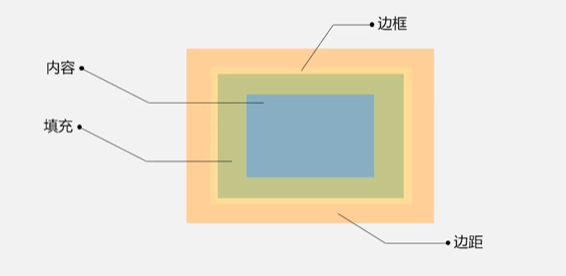
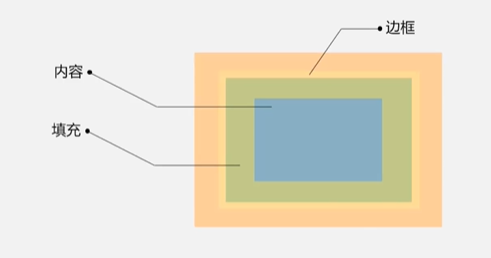
 子元素进行浮动的时候，没办法撑开父元素的大小 （有宽度没高度）



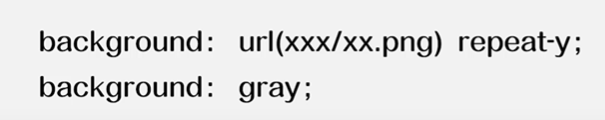




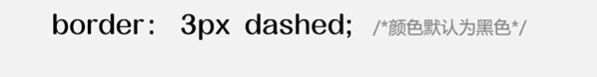
填充实际大小变大



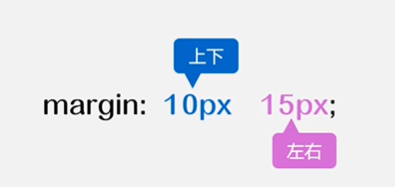
可省略

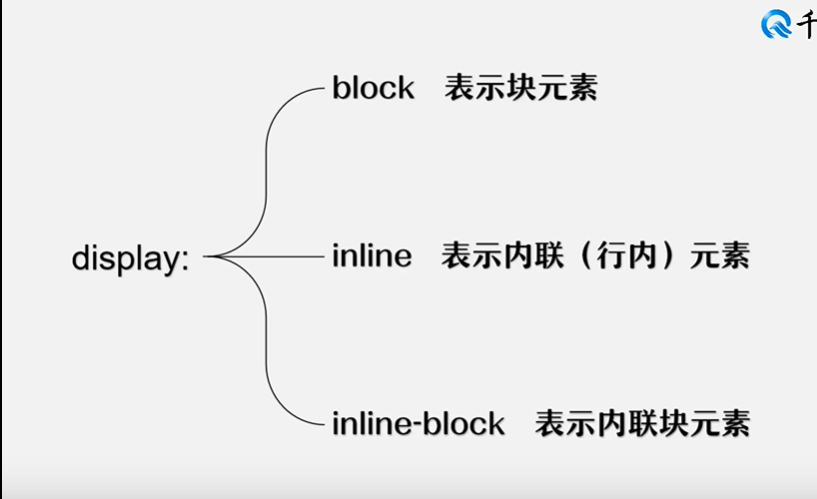


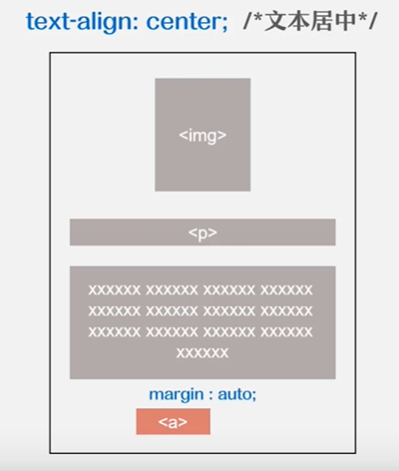
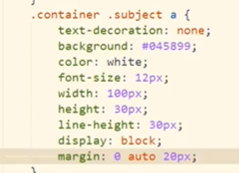


其他不可省略

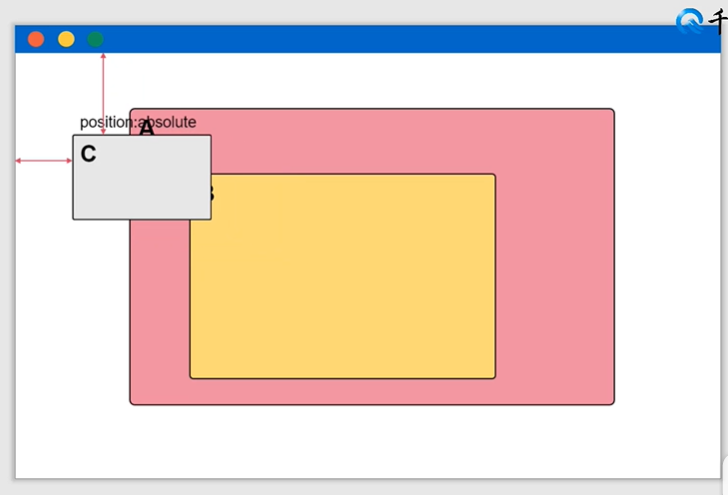


 padding一样



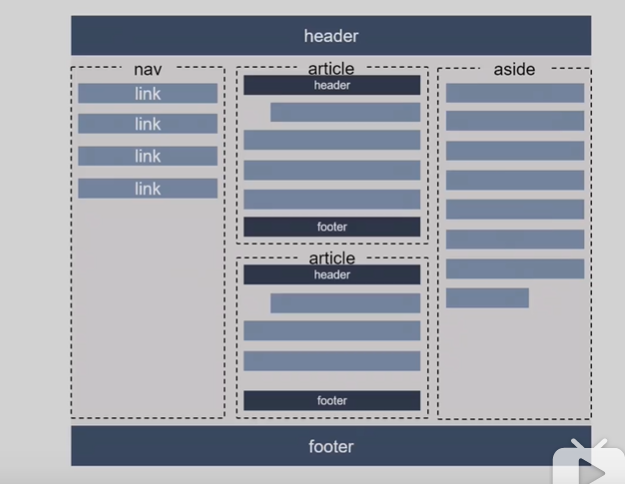
块元素

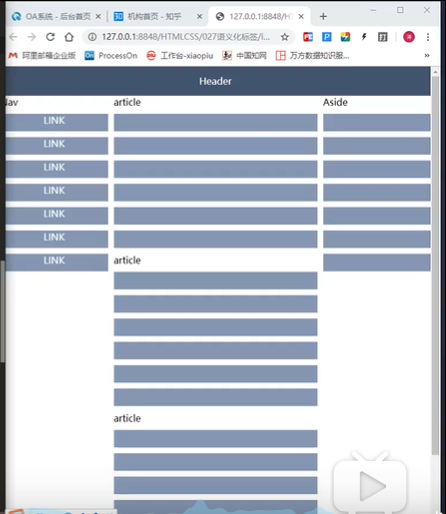
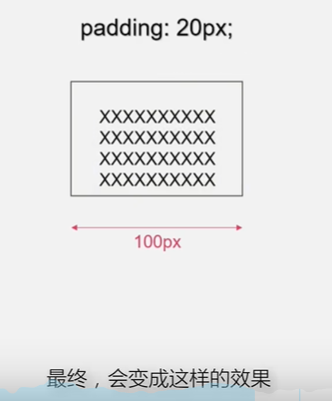
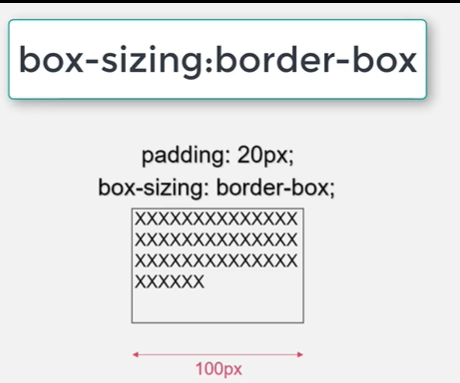


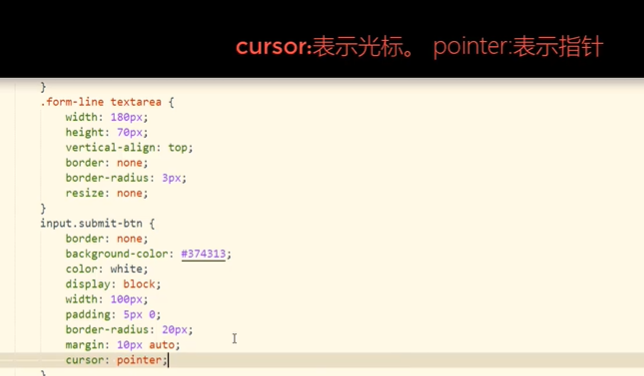
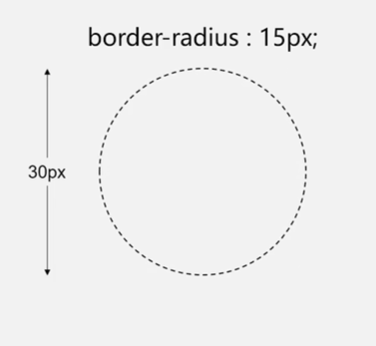
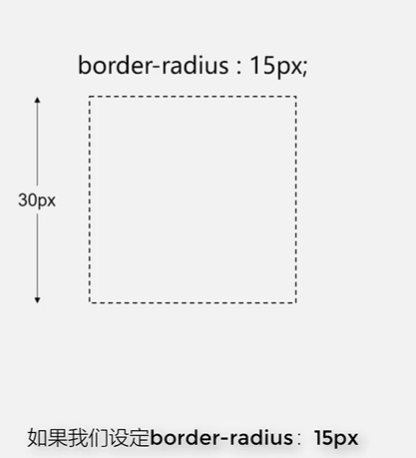
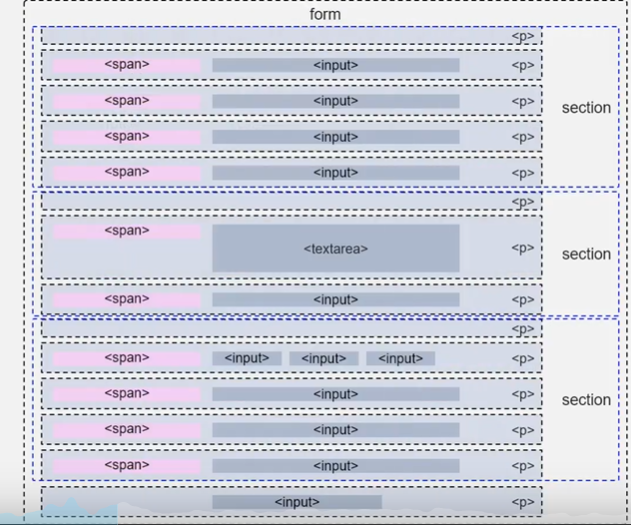
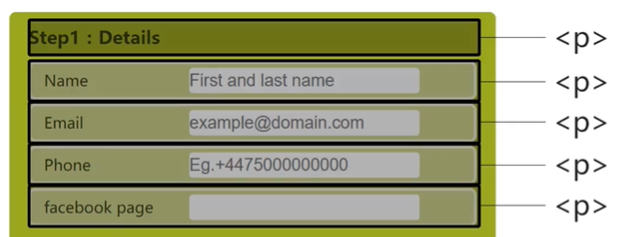
如果都没有relative 则参考整个浏览器窗口

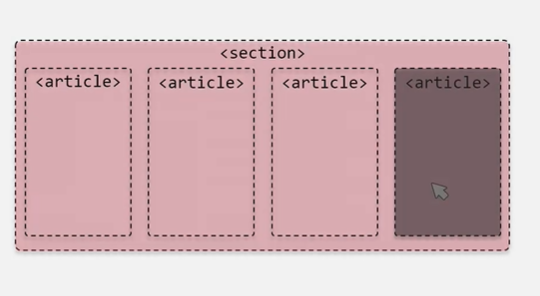


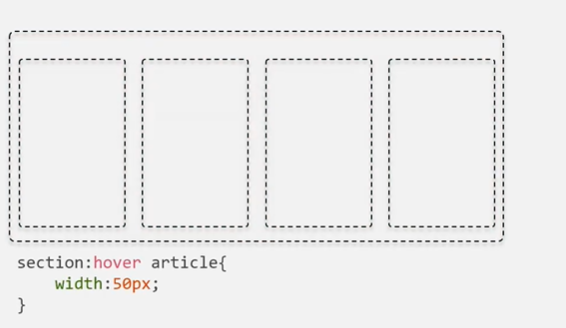
精灵图

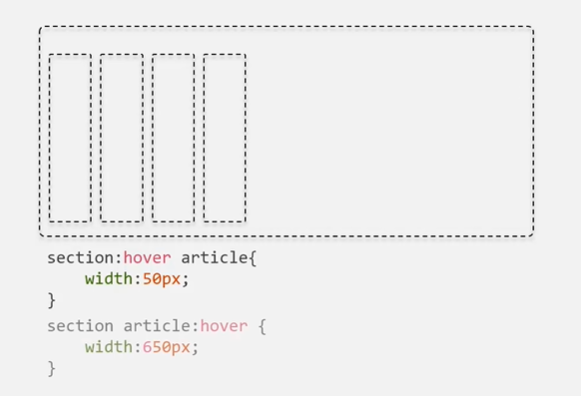
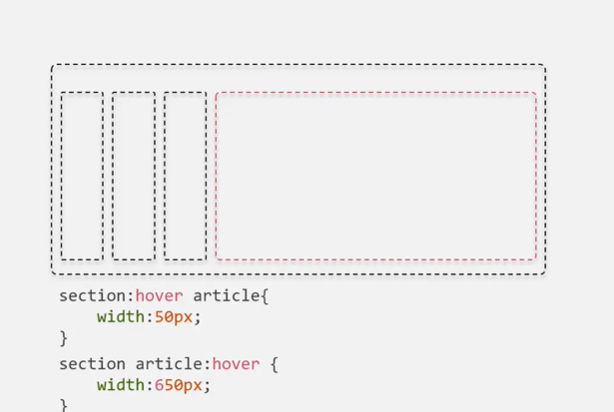
坐标控制背景图在容器中的位置 容器是固定大小 通过调整容器位置显示不同区域替代div



鼠标变小手

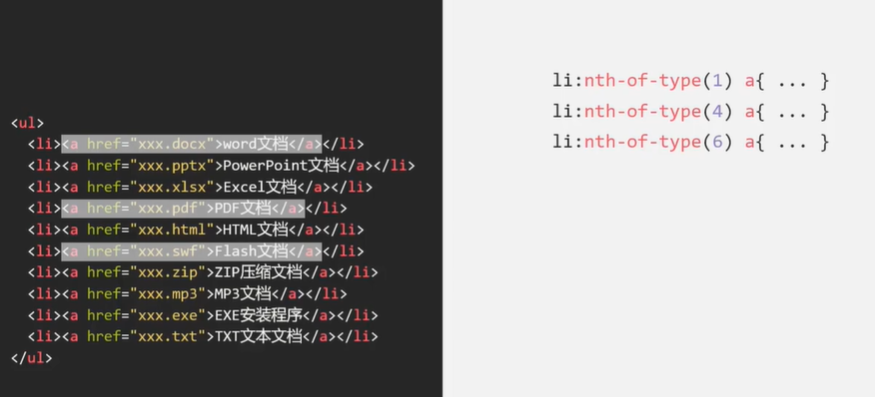
hover一个子元素时，父元素也被hover



父元素hover时写子元素的hover



去掉平铺



一个bfc区域和另一个bfc区域是相互对立的不受影响

内部的子元素出现了上下边距

边距不会跑到外面去影响别人

而是是内部拉开距离

这使得父元素变成了一个独立的bfc区域

如果一个元素被触发bfc规则

则高度会把浮动元素考虑进去

无论多少个元素都不会害怕父元素没用高度了

触发bfc（一个浏览器渲染元素的规则 块格式上下文

float

positi

display

overflow 不等于默认值就行

如何消除浮动的影响？

overflow：hidden

BFC具体表现

父子元素的上下边距不再合并

父元素的高度会将浮动元素计算在内

响应式布局：pc网页放在手机网页上正常显示

媒介查询 采取{}