CSS深入理解：float、inline-block、vertical-align、line-height

CSS基本常识：

**1.元素的分类：**

**block元素**：宽度自适应填充父容器

**inline元素**:无法设置width、height。通过padding,margin来影响水平的宽度，height是由内容撑开的。在垂直方向上 margin-top 及 margin-bottom 无效

**可替换性元素**：直接设置宽高含有width及height属性。如button、img 、iframe、 input、 textarea

box-sizing:border-box 从border开始计算宽度，主要应用在**流体布局**中（以百分比作为单位）。

**自适应布局：**容器的宽度自适应容器剩余的宽度

**响应式布局：**media查询

1. **table布局的弊端：**

* tr、td与css耦合的太紧密，嵌套太多层语义不清晰
* 表格里面的内容需要等待页面上的资源(img,js,样式表)加载完毕后才显示

table是具备包裹性的 block元素；**同行等高，同列等宽**，常用于多列布局。

父元素display:table ,子元素 display:table-cell ，孩子产生父节点display:row的匿名节点

margin：百分比是以父元素的width属性作为计算的基准；

定位元素的百分比是以closet() 第一个已定位的父元素作为基准来作为计算基准

padding：无法取负值， padding:50% 绘制正方形

1. **负margin 可以影响**
2. **DOM在文档流中的位置（位移）**

自适应两栏布局:

**右边自适应-->**左边float右边BFC化，使用padding-left控制间距(margin-left会出现”鞭长莫及”的情况)

**左边自适应-->** 左边float width：100% 使用padding-right/margin-right预留空间，右边float，并且width 及**负margin 值相同，那么可以把这个float块拉上去**

**td元素的特性：无论宽度多大都不会超出父容器的宽度，所以经常设置 width:9999px 来做自适应布局。**

把float:left的兄弟节点设置为display:table-cell 或者 overflow:auto(这是利用了BFC特性)可以做到两栏自适应布局

1. **影响未设定width的水平block元素的宽度 （改变元素的DOM尺寸可视区的大小）**
2. **垂直方向上的 负margin，无法把包括文字的block元素，拉出父容器**，（**位移+改变元素DOM占据的空间**：文字,img等内联元素无法在容器之外，并且默认与baseline对齐，及X字母的下边缘）

做垂直方向的方向的布局，需要同时使用margin-top && margin-bottom 利用margin的折叠特性。

**四.margin:auto:就是用来处理剩余空间。它作用的情况如下**：

前提条件：

1. 水平块状元素，如果不指定width那么它的宽度是自适应父容器

所以 定宽 && margin:0 auto; 可以水平居中,而 **定height && margin:auto 0;无法垂直居中。（因为高度不指定无法）**

**定宽的block元素： 等价于 width:300px && margin-right:auto**

1. absolute定位元素 ,top:0 right:0 bottom:0 left:0 （定位元素的拉伸盒子属性） 如果不指定width 及 height那么它会width 及 height会同时自适应父容器。**所以 margin:auto; 定位的元素可以水平垂直居中。**
2. 定宽的block水平元素，**设置 margin-left:auto,margin-right:20px，**是的容器按照margin-right来排列

margin失效：

* 在display:table-cell元素上
* 在内联元素的垂直方向
* 在absolute元素的相反方向上
* 在float的相反方向上

**五.CSS中的高度模型：（line-height vertical-align）**

1. 盒子模型形成高度

由height属性来指定

1. line-boxs形成高度

一个div中如果没有指定高度，那么它的高度是由，里面的一个个的inline元素的“行高”决定，在div上使用line-height属性来影响。一系列的 inline元素的排列方式，默认是以base-line作为对齐基准，使用vertical-align属性来影响各个inline元素的排列方式。

**六、float就是带方向性质的inline-block主要的特性：**

1. 破坏高度，宽度属性还存在（并没有脱离文档流）

例子：兄弟节点中的div中的文字，

1. 元素的包裹性
2. 使元素block化（float元素的display的值取为block）

脱离文档流：

float元素：**破坏了height，width属性还存在**，所以能文字环绕，配合左侧的BFC，margin-\*值还是可以影响相邻的元素。

absolute：完全**不占据文档流中 width及height，所以margin值不会把周边的元素推开**

**七、absolute元素：**

不指定width、height使用top、right、bottom、left来拉伸盒子

跟随性（无依赖的特性） 配合margin使用，带来位置的微调

使得overflow失效：1.hidden裁剪失效；2.使得滚动条失效（效果类似于fixed效果）

含有overflow属性的元素 在 **定位元素**及**包含块（[定位属性]的父元素如果没有则是body）**之间，裁剪会失效.

解决方案：使 含有overflow属性的元素，成为**绝对定位**

css：attr() 获取HTML上的属性值

不指定width/height 的absolute元素通过left/top/right/bottom拉伸后，其子元素的width/height可以使用百分比作为

1. **line-height与 vertical-align**

vertical-align 的百分比值 是以line-height作为计算的基准。

font-size:0 解决inline-block水平方向“空隙” 以及垂直方向上与 baseline对齐的”空隙”。兼容性问题（Chrome最小字体大小 font-size：4px 目前60版本没有这个问题）

letter-spacing:取负值 兼容性好

**幽灵空白节点**