居中方案

垂直/水平居中问题，是css世界最经典的问题。方案有很多，我们只梳理常用的。

思路一：绝对定位方案

以这个结构为例，实现 #center 在 #container 中垂直居中。

```
<div id="container">
    <div id="center"></div>
</div>
```

1. 常规操作：margin 置为负值

这条路的前提是，#center 盒子的宽高确定。

贴上最终的CSS代码：

```
<style>
#container {
  position: relative;
}
#center {
  width: 200px;
  height: 200px;
  background-color: red;
  position: absolute;
  left: 50%;
  top: 50%;
  margin-left: -100px;
  margin-top: -100px;
}
</style>
```

这里 `margin-left: -100px` 是关键。因为 left: 50% 是盒子左上角相对父容器的距离，若要居中，还要左移盒子宽度的一半。

`margin-top` 同理。

2. 流体特性：神奇的 margin: auto

通常想要水平居中某个元素时，我们会这样做：

```
margin: 0 auto;
```

那么 auto 是如何实现水平居中呢?

auto 的取值有两种可能：

1. 父元素剩余空间的宽度
2. 0

当元素在文档流内（未设置float与定位）且宽高确定时，取 1 的值；否则取 2 的值。

> 注意，以上 auto 的取值均指水平方向，垂直方向无效。

那么问题来了：如何利用 auto 实现垂直居中?

答案是利用元素的 **流体特性。**

流体特征：**绝对定位的元素，对立方向的值一致**（即left=right，top=bottom），就会发生流体特征。

流体特性的妙处，在于 **元素可自动填充父级元素的可用尺寸。**

再配合 margin: auto; 就会自动平分父元素的剩余空间。

```
#center {
  background-color: red;
  width: 200px;
  height: 200px;
  position: absolute;
  top: 10px;
  bottom: 10px;
  left: 20px;
  right: 20px;
  margin: auto;
}
```

3. 动画属性来帮忙

以上两个方案有个条件，就是子元素的宽高**必须指定。**

如果宽高不确定该怎么办呢?

别急，救星就是 —— **transform**

这个方案在 方案1 的基础上做了升级。利用 translate 平移直接移动自身宽高的 50%。

```
#center {
  position: absolute;
  left: 50%;
  top: 50%;
  transform: translate(-50%, -50%);
}
```

利用 calc() 计算高度也可以实现。

思路二：Flex 布局

这个方式就简单多了，它是现代浏览器最常用的方式，也是未来的布局方式。

微信小程序 和 flutter 就是用的这种布局思路。

flex 布局的关键是理解主轴与副轴，参考 阮一峰老师讲flex

```
#container {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

思路三：table 布局

table 布局历史悠久，但是兼容性好呀！

不过有个前提：父容器 #container 必须指定宽高。

```
#container {
  display: table-cell;
  text-align: center;
  vertical-align: middle;
  width: 200px;
  height: 80px;
}
#center {
  background-color: red;
  display: inline-block;
}
```