目录

[导航栏下拉至一定高度后固定在顶部 1](#_Toc512627843)

[隐藏滚动条 3](#_Toc512627844)

[垂直水平居中 3](#_Toc512627845)

[已知外层,内层宽高（absolute） 3](#_Toc512627846)

[已知外层,不管内层宽高（relative） 4](#_Toc512627847)

[不知外层宽高,知道内层宽高(通过padding) 4](#_Toc512627848)

[第三方基准 5](#_Toc512627849)

[文字居中 6](#_Toc512627850)

[垂直line-height: 200px; 6](#_Toc512627851)

[水平text-align: center; 6](#_Toc512627852)

[图片居中 6](#_Toc512627853)

[1垂直line-height: 300px; vertical-align: middle; 6](#_Toc512627854)

[2垂直display: table; 7](#_Toc512627855)

[水平text-align:center 7](#_Toc512627856)

[水平排列(浮动或者inline) 8](#_Toc512627857)

[inline 8](#_Toc512627858)

[float 9](#_Toc512627859)

[1像素 9](#_Toc512627860)

[图片 9](#_Toc512627861)

[居中拉伸 9](#_Toc512627862)

[使用背景图片的方式 9](#_Toc512627863)

# 导航栏下拉至一定高度后固定在顶部

固定高度

<script type="text/javascript">

$(function(){

var nav=$(".nav"); //得到导航对象

var win=$(window); //得到窗口对象

var sc=$(document);//得到document文档对象。

win.scroll(function(){

if(sc.scrollTop()>=100){

nav.addClass("fixednav");

$(".navTmp").fadeIn();

}else{

nav.removeClass("fixednav");

$(".navTmp").fadeOut();

}

})

})

</script>

将这个js放到要实现效果的页面里面去，然后我们要修改的是第三行的.nav，“nav“改成你自己页面导航栏的class；第七行的”100“为下拉到100个像素的时候触发特效，可以自行修改至合适的高度。

然后在css文件里面增加这个属性：

CSS

.fixednav {

position: fixed;

top: 0px;

left: 0px;

width: 100%;

z-index: 1;

}

不固定高度

var top=$(".top").offset().top;

$(window).scroll(function () {

  if ($(window).scrollTop() >= top) {

    $(".top").attr("style", "position:fixed;top:0;left:0;z-index:10;");

  } else {

     $(".top").attr("style", "");

  }

}).trigger("scroll");

底部判断

react写法

let clientHeight = this.refs.bodyBox.clientHeight; //可视区域高度

let scrollTop = this.refs.bodyBox.scrollTop; //滚动条滚动高度

let scrollHeight = this.refs.bodyBox.scrollHeight; //滚动内容高度

if((clientHeight+scrollTop)==(scrollHeight)){ //如果滚动到底部 }

}

# 隐藏滚动条

外层套一个div，然后overflow：hidden，里面的div比外面的稍微宽一点，然后overflow-y： auto   滚动条就隐藏了(大概宽16px,其实就是让内层的导航条被挡住)

# 垂直水平居中

## 已知外层,内层宽高（absolute）

要设置top…

<style>

.Center-Container {

position: relative;

}

.Absolute-Center {

background-color:yellow;

width: 50%;

height:20%;

overflow: auto;

**margin: auto;**

position: absolute;

**top: 0; left: 0; bottom: 0; right: 0;**

}

</style>

</head>

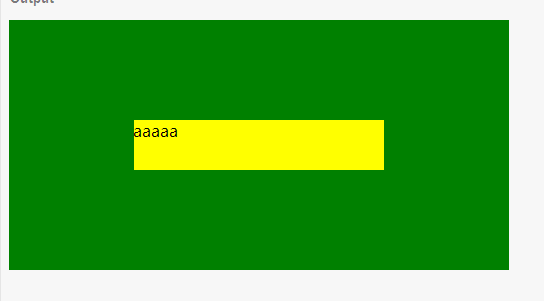
<body>

<div class="Center-Container" style="width:500px;height:250px;background-color:green;">

<div class="Absolute-Center">aaaaa

</div>

</div>



## 已知外层,不管内层宽高（relative）

减掉偏移量即可。

top: 50%; /\*偏移\*/

margin-top: -150px;

或者使用

设置div的transform: translateY(-50%)，意思是使得div向上平移（translate）自身高度的一半(50%)。这样就不用知道内层高度

更好用,还可以结合float浮动（在浮动内，使用absolute方式失效）

## 不知外层宽高,知道内层宽高(通过padding)

**其实不算居中。**

#box {

width: 300px;

background: #ddd;

padding: 100px 0;

}

#child {

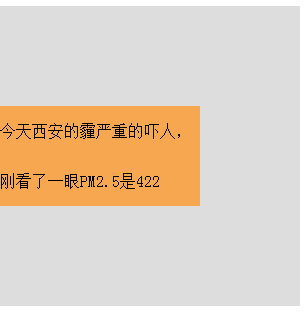
width: 200px;

height: 100px;

background: #F7A750;

line-height: 50px;

}



## 第三方基准

通过填充一个一半高度的div

#box {

width: 300px;

height: 300px;

background: #ddd;

}

#base {

height: 50%;

background: #AF9BD3;

}

#child {

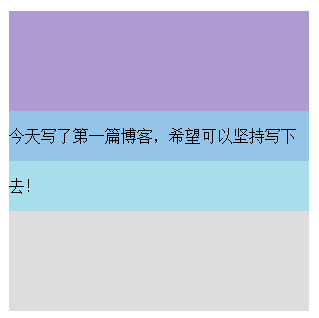
height: 100px;

background: rgba(131, 224, 245, 0.6);

line-height: 50px;

margin-top: -50px;

}



## 文字居中

### 垂直line-height: 200px;

一行就可以

### 水平text-align: center;

一样

## 图片居中

### 1垂直line-height: 300px; vertical-align: middle;

#box{

width: 300px;

height: 300px;

background: #ddd;

line-height: 300px;

}

#box img {

vertical-align: middle;

}

这种方法也适用于文字

### 2垂直display: table;

#box {

width: 300px;

height: 300px;

background: #ddd;

display: table;

}

#child {

display: table-cell;

vertical-align: middle;

}

vertical-align属性只对拥有valign特性的html元素起作用，例如表格元素中的<td><th>等等，而像<div><span>这样的元素是不行的。所以要配合display table或者line-height

也适用于文字

## 水平text-align:center

1. <!DOCTYPE >
2. <html xmlns="http://www.w3.org/1999/xhtml">
3. <head>
4. <meta charset="utf-8" />
5. <title>图片横向居中</title>
6. <style>
7. .divcss5{text-align:center}
8. </style>
9. </head>
10. <body>
11. <div class="divcss5"><img src="http://www.divcss5.com/img201305/divcss5-logo-201305.gif" /></div>
12. </body>
13. </html>



text-align必须放在div下面,但可以不用设置宽高

# 水平排列(浮动或者inline)

无论是float浮动还是display实现并排显示，要想并排显示首先总宽度要小于或等于对象上级宽度，这样才能并排显示实现横向排列排版布局。

## inline



这个方法有一个弊端：li转为内联元素后，width、height就无效了，只能用padding来撑开宽高。

## float

使用浮动时不能使用绝对定位，否则浮动会失去效果。

# 1像素

# 图片

rn中Image组件默认是居中的。

## 居中拉伸

### 使用背景图片的方式

优点：不用预先知道图片的宽高比例

**缺点: 当图片动态改变时，会重新加载导致组件闪一下**

{{

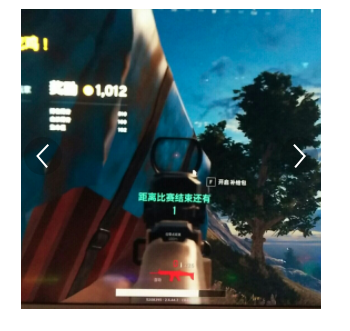
backgroundImage: `url(${selectImg})`,

backgroundPosition: 'center center',

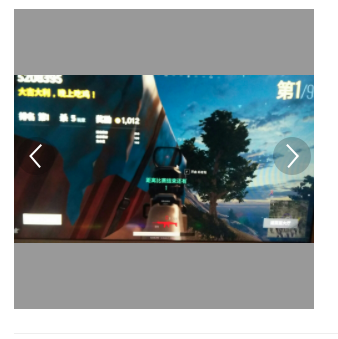
backgroundSize: 'cover',

backgroundRepeat: 'no-repeat'

}}



backgroundSize: 'contain',



# 超出文字省略号

## 单行

{

overflow: hidden;

text-overflow:ellipsis;

white-space: nowrap;

width: 247px;

}

注意： 对于input,未选中时正常，选中后失效

## 多行

-webkit-line-clamp:2; 兼容性差。

手动写一个...,通过js判断内容的长度，控制是否显示...