# 音视频播放

## audio标签

·向页面中引入外部音频文件

·默认情况下不允许用户自己控制播放停止

·ie8及以下浏览器不支持

·src

- 引入音频文件

<audio src="source/horizon.mp3"></audio>

·controls

- 添加之后可控制播放暂停和下载

- 不需要赋值

<audio src="source/audio.mp4" controls></audio>

·autoplay

- 打开页面后自动播放

- 不需要赋值

- 大部分浏览器不支持

<audio src="source/horizon.mp3" controls autoplay></audio>

·loop

- 循环播放

<audio src="source/horizon.mp3" controls loop></audio>

·source标签

- 也可以引入音频文件

- 支持的浏览器会显示播放器

- 不支持的浏览器会显示文字

- 可以同时写多个source

<audio controls>

对不起，您的浏览器不支持播放音频，请升级或更换浏览器

<source src="source/horizon.mp3">

</audio>

## embed标签

·几乎所有浏览器都支持

·可播放视频

<embed src="source/audio.mp4">

·最好为它指定width和height属性，才能正常显示

·非常的不好用

·type

- 指定文件格式

<embed src="source/horizon.mp3" type="audio/mp3">

·当ie8及以下浏览器时使用embed，其他浏览器使用audio

<audio controls>

<source src="source/horizon.mp3">

<embed src="source/horizon.mp3" type="audio/mp3">

</audio>

## video标签

·引入视频文件

·用法和audio基本一样

·ie9及以上浏览器只支持mp4格式

·ie8及以下浏览器不支持

<video controls>

<source src="source/audio.mp4">

</video>

·当ie8及以下浏览器时使用embed，其他浏览器使用video

<video controls>

<source src="source/audio.mp4">

<embed src="source/audio.mp4" type="audio/mp4">

</video>

## 引入外部网站的音视频



# 水平方向的布局

·元素在其父元素中水平方向的位置由以下几个属性共同决定

- margin-left（左外边距）

- border-left（左边框）

- padding-left（左内边距）

- width（内容区）

- padding-right（右内边距）

- border-right（右边框）

- margin-right（右外边距）

·一个元素在其父元素中，水平方向必须满足以下等式

- margin-left + border-left + padding-left + width + padding-right + border-right + margin-right = 其父元素的内容区宽度

·如果不满足且没有auto情况，则称为过度约束，浏览器会自动调整margin-left或margin-right使等式成立

·这七个值中，有三个值可以设置成auto

- margin-left

- width

- margin-right

- 如果某个值为auto，则会自动调整为auto的那个值以使auto成立

- width的值默认就是auto

·如果将width和一个或两个外边距设置为auto，则默认只有width为auto，外边距会变为0

·如果将两个外边距设置为auto，宽度固定值，则会将外边距设置为相同的值（水平居中）

·如果子元素设置的width超过父元素内容区宽度，则子元素会溢出父元素，即浏览器会自动调整margin-left或margin-right变成负数使等式成立

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.outer{

width: 800px;

height: 200px;

border: 10px red solid;

}

.inner{

/\* width: auto; width的值默认就是auto\*/

width: 200px;

height: 200px;

background-color: #bfa;

margin-right: auto;

margin-left: auto;

/\* margin-left: 100px;

margin-right: 400px \*/

/\*

0 + 0 + 0 + 200 + 0 + 0 + 0 = 800

0 + 0 + 0 + 200 + 0 + 0 + 600 = 800

100 + 0 + 0 + 200 + 0 + 0 + 400 = 800

100 + 0 + 0 + 200 + 0 + 0 + 500 = 800

0 + 0 + 0 + auto + 0 + 0 + 0 = 800 auto = 800

0 + 0 + 0 + auto + 0 + 0 + 200 = 800 auto = 600

200 + 0 + 0 + auto + 0 + 0 + 200 = 800 auto = 400

auto + 0 + 0 + 200 + 0 + 0 + 200 = 800 auto = 400

auto + 0 + 0 + 200 + 0 + 0 + auto = 800 auto = 300

}

</style>

</head>

<body>

<div class="outer">

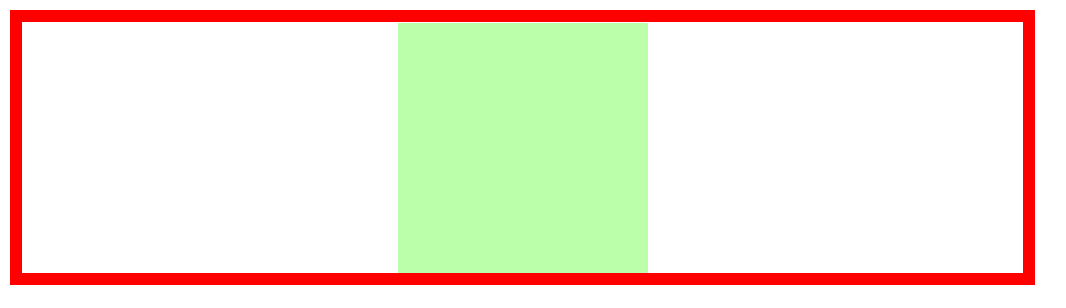
<div class="inner"></div>

</div>

</body>

</html>

·效果



# 垂直方向的布局

·默认情况下父元素的高度会被内容撑开

·将子元素设置浮动后可能会导致高度塌陷

·如果子元素设置的height超过父元素内容区高度，则子元素会溢出父元素

·可使用overflow为父元素处理溢出内容

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.outer{

background-color: #bfa;

height: 600px;

/\*

默认情况下父元素的高度被内容撑开

\*/

}

.inner{

width: 100px;

background-color: yellow;

height: 100px;

margin-bottom: 100px;

}

.box1{

width: 200px;

height: 200px;

background-color: #bfa;

/\*

子元素是在父元素的内容区中排列的，

如果子元素的大小超过了父元素，则子元素会从父元素中溢出

使用 overflow 属性来设置父元素如何处理溢出的子元素

可选值：

visible，默认值 子元素会从父元素中溢出，在父元素外部的位置显示

hidden 溢出内容将会被裁剪不会显示

scroll 生成两个滚动条，通过滚动条来查看完整的内容

auto 根据需要生成滚动条

overflow-x:

overflow-y:

\*/

overflow: auto;

}

.box2{

width: 100px;

height: 400px;

background-color: orange;

}

</style>

</head>

<body>

<!-- <div class="outer">

<div class="inner"></div>

<div class="inner"></div>

</div> -->

<div class="box1">

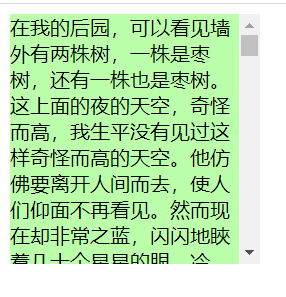
在我的后园，可以看见墙外有两株树，一株是枣树，还有一株也是枣树。 这上面的夜的天空，奇怪而高，我生平没有见过这样奇怪而高的天空。他仿佛要离开人间而去，使人们仰面不再看见。然而现在却非常之蓝，闪闪地䀹着几十个星星的眼，冷眼。他的口角上现出微笑，似乎自以为大有深意，而将繁霜洒在我的园里的野花草上。 我不知道那些花草真叫什么名字，人们叫他们什么名字。我记得有一种开过极细小的粉红花，现在还开着，但是更极细小了，她在冷的夜气中，瑟缩地做梦，梦见春的到来，梦见秋的到来，梦见瘦的诗人将眼泪擦在她最末的花瓣上，告诉她秋虽然来，冬虽然来，而此后接着还是春，蝴蝶乱飞，蜜蜂都唱起春词来了。她于是一笑，虽然颜色冻得红惨惨地，仍然瑟缩着。 枣树，他们简直落尽了叶子。先前，还有一两个孩子来打他们，别人打剩的枣子，现在是一个也不剩了，连叶子也落尽了。他知道小粉红花的梦，秋后要有春；他也知道落叶的梦，春后还是秋。他简直落尽叶子，单剩干子，然而脱了当初满树是果实和叶子时候的弧形，欠伸得很舒服。但是，有几枝还低亚着，护定他从打枣的竿梢所得的皮伤，而最直最长的几枝，却已默默地铁似的直刺着奇怪而高的天空，使天空闪闪地鬼䀹眼；直刺着天空中圆满的月亮，使月亮窘得发白。 鬼䀹眼的天空越加非常之蓝，不安了，仿佛想离去人间，避开枣树，只将月亮剩下。然而月亮也暗暗地躲到东边去了。而一无所有的干子，却仍然默默地铁似的直刺着奇怪而高的天空，一意要制他的死命，不管他各式各样地䀹着许多蛊惑的眼睛。 哇的一声，夜游的恶鸟飞过了。 我忽而听到夜半的笑声，吃吃地，似乎不愿意惊动睡着的人，然而四围的空气都应和着笑。夜半，没有别的人，我即刻听出这声音就在我嘴里，我也即刻被这笑声所驱逐，回进自己的房。灯火的带子也即刻被我旋高了。 后窗的玻璃上丁丁地响，还有许多小飞虫乱撞。不多久，几个进来了，许是从窗纸的破孔进来的。他们一进来，又在玻璃的灯罩上撞得丁丁地响。一个从上面撞进去了，他于是遇到火，而且我以为这火是真的。两三个却休息在灯的纸罩上喘气。那罩是昨晚新换的罩，雪白的纸，折出波浪纹的叠痕，一角还画出一枝猩红色的栀子。 猩红的栀子开花时，枣树又要做小粉红花的梦，青葱地弯成弧形了……我又听到夜半的笑声；我赶紧砍断我的心绪，看那老在白纸罩上的小青虫，头大尾小，向日葵子似的，只有半粒小麦那么大，遍身的颜色苍翠得可爱，可怜。 我打一个呵欠，点起一支纸烟，喷出烟来，对着灯默默地敬奠这些苍翠精致的英雄们。 一九二四年九月十五日。

</div>

</body>

</html>

·效果



# 盒子的大小

·默认情况下，盒子可见框的大小 = 内容区 + 内边距 + 边框

·box-sizing

- 用来设置盒子尺寸的计算方式（设置width和height的作用）

- content-box

-- 默认值，宽度和高度用来设置内容区的大小

- border-box

-- 宽度和高度用来设置整个盒子可见框的大小

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

width: 100px;

height: 100px;

background-color: #bfa;

padding: 10px;

border: 10px red solid;

box-sizing: border-box;

}

</style>

</head>

<body>

<div class="box1"></div>

</body>

</html>

# 轮廓阴影和圆角

## 轮廓

·outline

- 用来设置元素的轮廓线，用法和border一模一样

- 轮廓和边框不同的点

-- 轮廓不会影响到可见框的大小，不会影响页面布局

## 阴影

·box-shadow

- 用来设置元素的阴影效果，阴影不会影响页面布局

- 第一个值

-- 水平偏移量，设置阴影的水平位置，正值向右移动，负值向左移动

- 第二个值

-- 垂直偏移量，设置阴影的水平位置，正值向下移动，负值向上移动

- 第三个值

-- 阴影的模糊半径

-- 值越大，模糊效果越好

- 第四个值

-- 阴影的颜色

## 圆角

·border-radius

- 用来设置圆角

- 可以分别指定四个角的圆角

- 四个值

-- 左上 右上 右下 左下

- 三个值

-- 左上 右上/左下 右下

- 两个值

-- 左上/右下 右上/左下

·设置两个值时，当两个值大小不一样，可变成椭圆圆角

- border-radius: 20px / 40px;

·border-top-left-radius

- 左上角

·border-top-right-radius

- 右上角

·border-bottom-left-radius

- 左下角

·border-bottom-right-radius

- 右下角

·设置两个值时，当两个值大小不一样，可变成椭圆圆角

- border-top-left-radius:50px 100px

·将元素设置为一个圆形

- border-radius: 50%;

# 粘滞定位

·兼容性并不是很好

· position: sticky

- 开启粘滞定位

·粘滞定位和相对定位的特点基本一致

·不同的是粘滞定位可以在元素**到达某个位置时将其固定**

·top: 10px

- 距离顶部10px时将其固定住

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>导航条</title>

<link rel="stylesheet" href="./css/reset.css">

<style>

body{

height: 3000px;

}

/\* 设置nav的大小 \*/

.nav{

/\* 设置宽度和高度 \*/

width: 1210px;

height: 48px;

/\* 设置背景颜色 \*/

background-color: #E8E7E3;

margin:100px auto;

/\*

粘滞定位

- 当元素的position属性设置为sticky时则开启了元素的粘滞定位

- 粘滞定位和相对定位的特点基本一致，

不同的是粘滞定位可以在元素到达某个位置时将其固定

\*/

position: sticky;

top: 10px;

}

/\* 设置nav中li \*/

.nav li{

/\* 设置li向左浮动，已使菜单横向排列 \*/

float: left;

/\* 设置li的高度 \*/

/\* height: 48px; \*/

/\* 将文字在父元素中垂直居中 \*/

line-height: 48px;

}

/\* 设置a的样式 \*/

.nav a{

/\* 将a转换为块元素 \*/

display: block;

/\* 去除下划线 \*/

text-decoration: none;

/\* 设置字体颜色 \*/

color: #777777;

/\* 修改字体大小 \*/

font-size: 18px;

padding: 0 39px;

}

.nav li:last-child a{

padding: 0 42px 0 41px;

}

/\* 设置鼠标移入的效果 \*/

.nav a:hover{

background-color: #3F3F3F;

color: #E8E7E3;

}

</style>

</head>

<body>

<!-- 创建导航条的结构 -->

<ul class="nav">

<li>

<a href="#">HTML/CSS</a>

</li>

<li>

<a href="#">Browser Side</a>

</li>

<li>

<a href="#">Server Side</a>

</li>

<li>

<a href="#">Programming</a>

</li>

<li>

<a href="#">XML</a>

</li>

<li>

<a href="#">Web Building</a>

</li>

<li>

<a href="#">Reference</a>

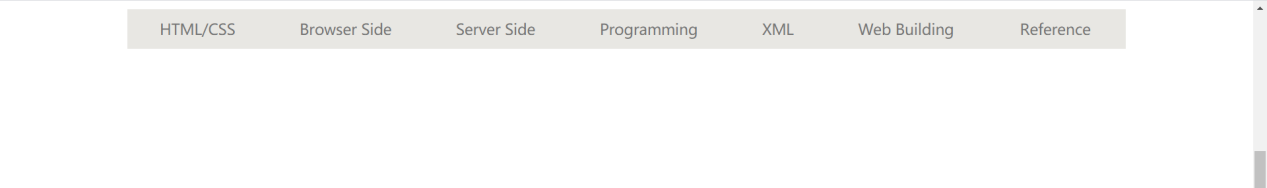
</li>

</ul>

</body>

</html>

·效果



# 绝对定位元素的位置

## 水平布局

·left + margin-left + border-left + padding-left + width + padding-right + border-right + margin-right + right = 包含块的内容区的宽度

·当我们开启了绝对定位后，水平方向的布局等式就需要添加left 和 right 两个值

·此时规则和之前一样只是多添加了两个值

- 如果9个值中没有 auto 则自动调整right值以使等式满足

- 如果有auto，则自动调整auto的值以使等式满足

·可设置auto的值

- margin

- width

- left

- right

·因为left 和 right的值默认是auto，所以如果不指定left和right，则等式不满足时，会自动调整这两个值

·如果要设置水平居中，需要把left和right设置成相同的值

·垂直方向布局的等式的也必须要满足

- top + margin-top/bottom + padding-top/bottom + border-top/bottom + height = 包含块的高度

- 可以利用这个特点，让元素垂直居中

·水平垂直居中

- 开启绝对定位，margin为auto，上下左右为0

- 这种方式只适用于大小确定的元素

- width: 100px;

- height: 100px;

- position: absolute;

- margin: auto;

- left: 0;

- right: 0;

- top: 0;

- bottom: 0;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

width: 500px;

height: 500px;

background-color: #bfa;

position: relative;

}

.box2{

width: 100px;

height: 100px;

background-color: orange;

position: absolute;

margin: auto;

left: 0;

right: 0;

top: 0;

bottom: 0;

}

</style>

</head>

<body>

<div class="box1">

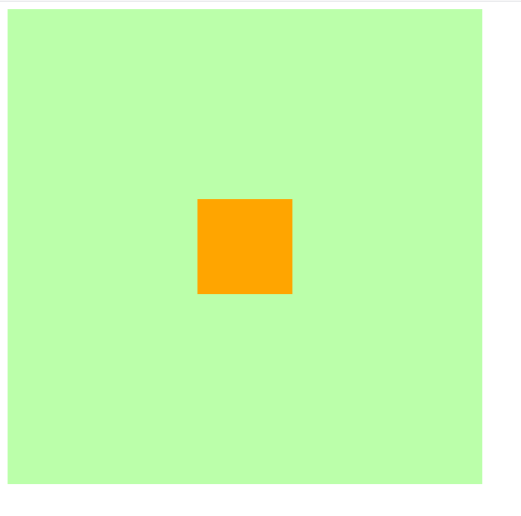
<div class="box2"></div>

</div>

</body>

</html>

·效果



# 字体样式补充

·@font-face选择器

- 可以将服务器中的字体直接提供给用户去使用

- font-family

-- 指定字体的名字

font-family:'myfont' ;

- src

-- 服务器中字体的路径和字体格式

-- format写不写都行

src: url('font/sgt.ttf') format("truetype");

·问题：

- 加载速度

- 版权

- 字体格式

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

/\*

font-face可以将服务器中的字体直接提供给用户去使用

问题：

1.加载速度

2.版权

3.字体格式

\*/

@font-face {

/\* 指定字体的名字 \*/

font-family:'myfont' ;

/\* 服务器中字体的路径 \*/

src: url('font/sgt.ttf') format("truetype");

}

p{

color: blue;

font-size: 40px;

font-family: myfont;

}

</style>

</head>

<body>

<p>

今天天气真不错，Hello Hello How are you！

</p>

</body>

</html>

·字体都属于矢量图，不会失真

# 图标字体

·在网页中经常需要使用一些图标

·可以通过图片来引入图标，但是图片大小本身比较大，并且非常的不灵活

·所以在使用图标时，我们还可以将图标直接设置为字体，然后通过font-face的形式来对字体进行引入

·使用时注意版权问题

## fontawesome

### 使用步骤

·下载 https://fontawesome.com/

·解压

·将css和webfonts移动到项目中

·将all.css引入到网页中

·使用图标字体

- 直接通过类名来使用图标字体

- class="fas fa-bell"

- class="fab fa-accessible-icon"

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" href="./fa/css/all.css">

</head>

<body>

<i class="fas fa-bell" style="font-size:80px; color:red;"></i>

<i class="fas fa-bell-slash"></i>

<i class="fab fa-accessible-icon"></i>

<i class="fas fa-otter" style="font-size: 160px; color:green;"></i>

</body>

</html>

·效果



### 其他使用方式

·通过伪元素来设置图标字体

- 找到要设置图标的元素通过before或after选中

- 在content中设置字体的编码

- 设置字体的样式

-- fab

--- font-family: 'Font Awesome 5 Brands';

-- fas

--- font-family: 'Font Awesome 5 Free';

--- font-weight: 900;

·通过实体方式使用

- &#x图标的编码;

<span class="fas">&#xf0f3;</span>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" href="./fa/css/all.css">

<style>

li{

list-style: none;

}

li::before{

/\*

通过伪元素来设置图标字体

1.找到要设置图标的元素通过before或after选中

2.在content中设置字体的编码

3.设置字体的样式

fab

font-family: 'Font Awesome 5 Brands';

fas

font-family: 'Font Awesome 5 Free';

font-weight: 900;

\*/

content: '\f1b0';

/\* font-family: 'Font Awesome 5 Brands'; \*/

font-family: 'Font Awesome 5 Free';

font-weight: 900;

color: blue;

margin-right: 10px;

}

</style>

</head>

<body>

<!-- <i class="fas fa-cat"></i> -->

<ul>

<li>锄禾日当午</li>

<li>汗滴禾下土</li>

<li>谁知盘中餐</li>

<li>粒粒皆辛苦</li>

</ul>

<!--

通过实体来使用图标字体：

&#x图标的编码;

-->

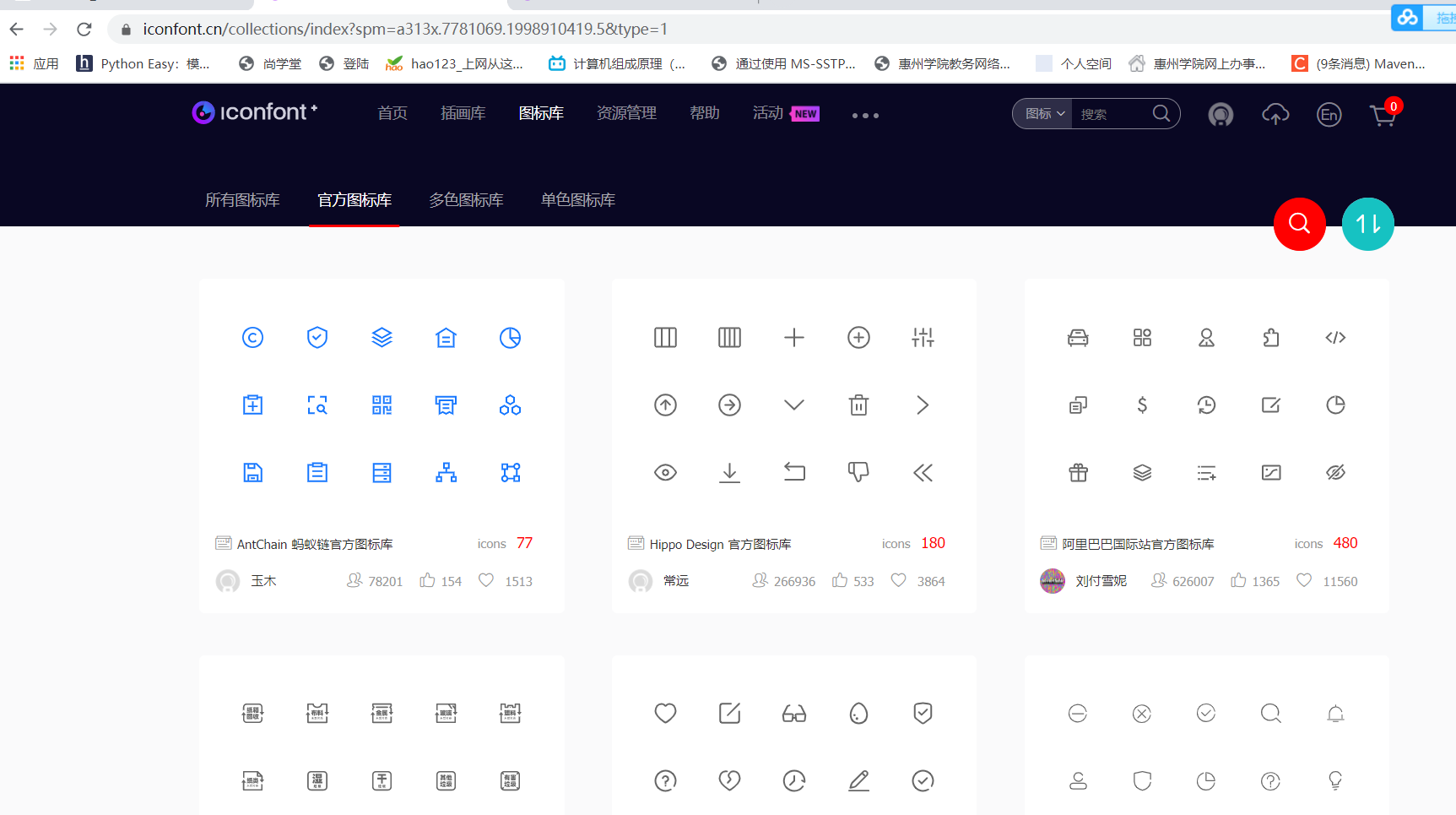
<span class="fas">&#xf0f3;</span>

</body>

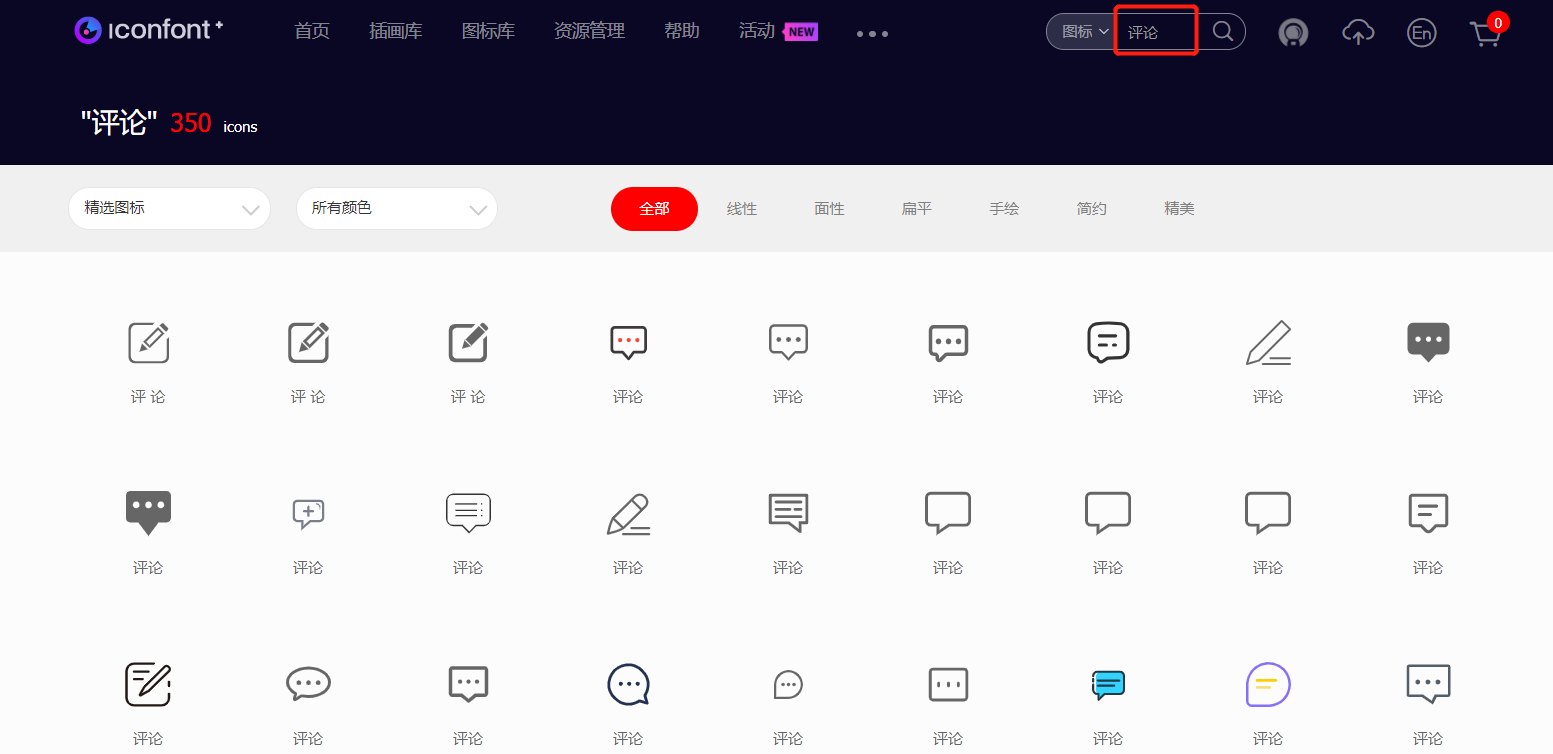
</html>

# 阿里字体库

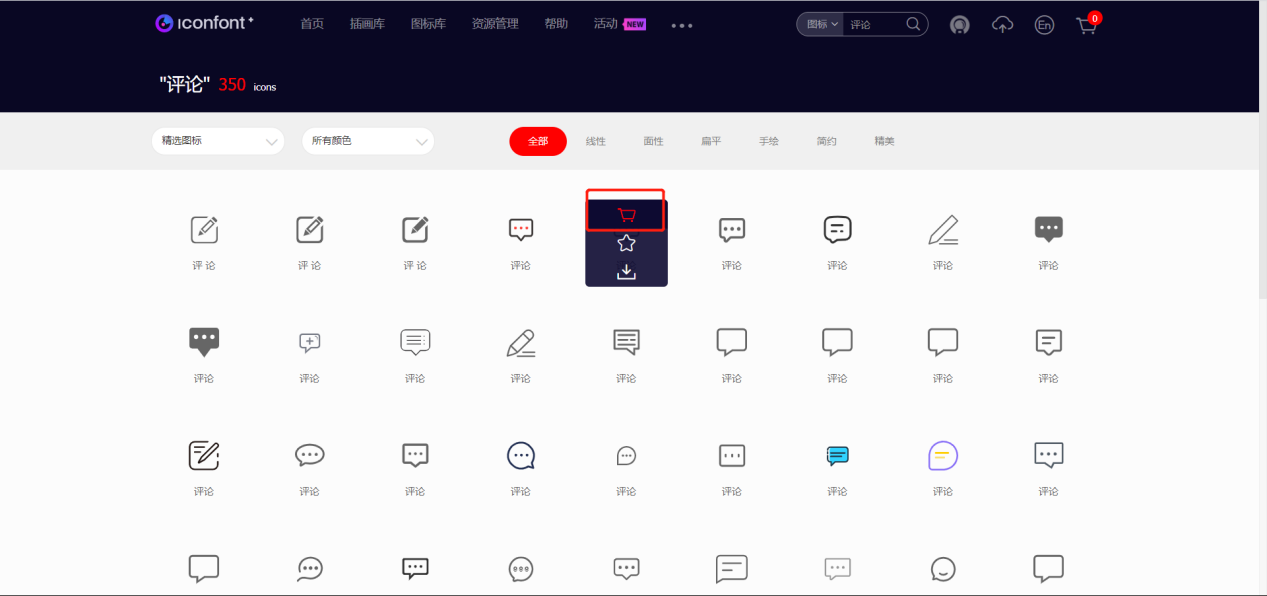
<https://www.iconfont.cn/>



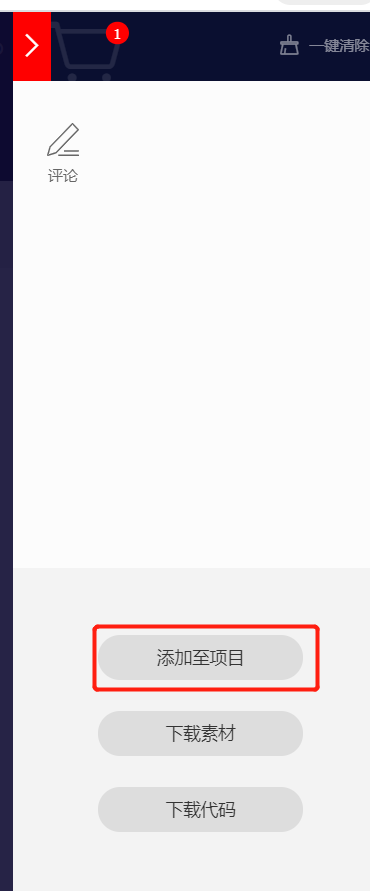
搜索需要的图标



将图标添加至购物车



添加到项目

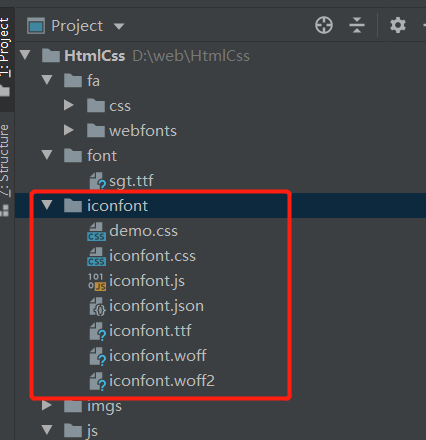




下载至本地



将里面的文件移入到项目



将CSS文件引入到页面

<link rel="stylesheet" href="./iconfont/iconfont.css">

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" href="./iconfont/iconfont.css">

<style>

i.iconfont{

font-size: 50px;

color: #0086b3;

}

p::before{

content: '\e6af;';

font-family: 'iconfont';

font-size: 10px;

}

</style>

</head>

<body>

<i class="iconfont">&#xe609;</i>

<i class="iconfont">&#xe618;</i>

<i class="iconfont">&#xe6ad;</i>

<i class="iconfont icon-mianxingshoucang"></i>

<p>Hello</p>

</body>

</html>

·效果



# 文本

·文本的水平对齐在之前文档有写，使用text-align属性

## 文本的垂直对齐

·vertical-align

- 设置元素垂直对齐的方式

- baseline

-- 默认值，基线对齐

- top

-- 顶部对齐

- bottom

-- 底部对齐

- middle

-- 居中对齐

-- 不是严格意义的居中

- 指定值

-- 正值往上，负值往下

·可用此方法取消图片的基线对齐

- vertical-align：top

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

div{

width: 800px;

border: 1px red solid;

font-size: 50px;

}

span{

font-size: 20px;

border: 1px blue solid;

vertical-align:baseline;

}

p{

border: 1px red solid;

}

img{

vertical-align: bottom;

}

</style>

</head>

<body>

<div>

今天天气 Helloyx<span>真不错 Hello</span>！

</div>

<!-- <div>

Lorem ipsum dolor, sit amet consectetur adipisicing elit. Illo nihil iure at ab atque nostrum molestiae totam porro, dolorem maiores repudiandae molestias veritatis, eligendi laudantium incidunt dolores corporis? Quibusdam, consequatur.

</div> -->

<p>

<img src="./img/an.jpg" alt="">

</p>

</body>

</html>

·效果



## 其它文本样式

·设置下划线的样式

- 部分浏览器不兼容

text-decoration: underline red dotted;

·white-space

- normal

-- 正常，默认值

- nowrap

-- 不换行

- pre

-- 保留空白

·裁剪多余的内容

overflow: hidden;

·在溢出内容的后方显示省略号

text-overflow: ellipsis;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

font-size: 50px;

font-family: 微软雅黑;

/\* text-decoration: overline; \*/

/\*text-decoration: underline red dotted;\*/

}

.box2{

width: 200px;

white-space: nowrap;

overflow: hidden;

text-overflow: ellipsis;

}

</style>

</head>

<body>

<div class="box2">

Lorem ipsum dolor sit amet consectetur adipisicing elit. Consequatur, minus fugit in perspiciatis reprehenderit consequuntur aspernatur repellat cumque quidem asperiores quaerat placeat, tenetur vel veritatis deserunt numquam. Dolores, cupiditate enim.

</div>

<div class="box1">

今天天气真不错

</div>

</body>

</html>

# 背景

·background-clip

- 设置背景颜色的范围

- border-box

-- 默认值，背景会出现在边框的下边

- padding-box

-- 背景不会出现在边框，只出现在内容区和内边距

- content-box

-- 背景只会出现在内容区

- ie8及以下浏览器不支持

·background-origin

- 背景图片的偏移量计算的原点

- padding-box

-- 默认值，background-position从内边距处开始计算

-- content-box

- content-box

-- 背景图片的偏移量从内容区处计算

- border-box

-- 背景图片的变量从边框处开始计算

·background-size

- 设置背景图片的大小

- 第一个值表示宽度

- 第二个值表示高度

- 如果只写一个，则第二个值默认是 auto

- cover

-- 图片比例不变，将元素铺满

- contain

-- 图片比例不变，将图片在元素中完整显示

·backgound

- 背景相关的简写属性

- background-size必须写在background-position的后边，并且使用/隔开

- background-origin background-clip 两个样式 ，orgin要在clip的前边

background: url('./imgs/8.jpg') #bfa center center/contain border-box content-box no-repeat ;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

width: 500px;

height: 500px;

overflow: auto;

background-color: #bfa;

background-image: url("./imgs/8.jpg");

background-repeat: no-repeat;

background-position: 0 0;

padding: 10px;

background-size: contain;

}

.box2{

width: 300px;

height: 1000px;

background-image: url('./imgs/9.jpg');

background-repeat: no-repeat;

background-position: 100px 100px;

background-attachment: fixed;

}

.box3{

border: 10px red double;

padding: 50px;

width: 500px;

height: 500px;

background: url('./imgs/8.jpg') #bfa center center/contain border-box content-box no-repeat ;

}

</style>

</head>

<body>

<div class="box3">

</div>

<!-- <div class="box1">

<div class="box2">

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Totam aut, odio iusto accusantium ipsum aliquid omnis facere sapiente, nobis vel dicta alias ducimus. Repellat similique unde eius tempore, quia quo.

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Accusantium, accusamus quibusdam. Adipisci in dolorem qui accusantium accusamus voluptatibus magnam nesciunt minus enim quaerat! Quidem, rem. Ipsum amet praesentium enim aliquid!

Lorem ipsum dolor sit amet consectetur adipisicing elit. Aperiam provident repellendus ipsum dolorum optio quo, iure eveniet beatae cupiditate rerum minus corporis illum aliquam illo ut quidem aliquid expedita deserunt.

</div>

</div> -->

</body>

</html>

## 普通渐变

·渐变是图片，需要通过background-image来设置

·linear-gradient()

- 线性渐变，颜色沿着一条直线发生变化

- linear-gradient(red,yellow) 红色向黄色过渡渐变

- 线性渐变的开头，我们可以指定一个渐变的方向

-- to left

-- to right

-- to bottom

-- to top

-- 以上四个可以组合使用

-- xxxdeg 表示度数

-- turn 表示圈

·渐变可以同时指定多个颜色，多个颜色默认情况下平均分布，也可以手动指定渐变的分布情况

·repeating-linear-gradient()

- 可以平铺的线性渐变

- 用法和linear-gradient()一样

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

width: 200px;

height: 200px;

/\* background-color: #bfa; \*/

/\* background-image: linear-gradient(red,yellow,#bfa,orange); \*/

/\* background-image: linear-gradient(red 50px,yellow 100px, green 120px, orange 200px); \*/

background-image: repeating-linear-gradient(to right ,red, yellow 50px);

}

</style>

</head>

<body>

<div class="box1"></div>

</body>

</html>

## 径向渐变

·径向渐变(放射性的效果)

·radial-gradient()

·默认情况下径向渐变的形状根据元素的形状来计算的

- 正方形 --> 圆形

- 长方形 --> 椭圆形

·手动指定径向渐变的大小

- 传入两个值

background-image: radial-gradient(100px 100px, red , #bfa);

- circle

-- 表示正圆

background-image: radial-gradient(circle, red , #bfa);

- ellipse

-- 表示椭圆

background-image: radial-gradient(circle, red , #bfa);

·也可以指定渐变的位置

- radial-gradient(大小 at 位置, 颜色 位置 ,颜色 位置 ,颜色 位置)

- 大小：

-- circle 圆形

-- ellipse 椭圆

-- closest-side 近边

-- closest-corner 近角

-- farthest-side 远边

-- farthest-corner 远角

- 位置：

-- top right left center bottom

background-image: radial-gradient(farthest-corner at 100px 100px, red , #bfa);

# 表格

·默认情况下元素在td中是垂直居中的，可以通过 vertical-align 来修改

·display: table-cell

- 将元素设置为单元格 td

·vertical-align: middle

- 使元素垂直居中

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

table{

width: 50%;

border: 1px solid black;

margin: 0 auto;

/\* border-spacing: 指定边框之间的距离 \*/

/\* border-spacing: 0px; \*/

/\* border-collapse: collapse; 设置边框的合并 \*/

border-collapse: collapse;

}

td{

border: 1px solid black;

height: 100px;

/\* 默认情况下元素在td中是垂直居中的 可以通过 vertical-align 来修改\*/

vertical-align:middle;

text-align: center;

}

/\*

如果表格中没有使用tbody而是直接使用tr，

那么浏览器会自动创建一个tbody，并且将tr全都放到tbody中

tr不是table的子元素

\*/

tbody > tr:nth-child(odd){

background-color: #bfa;

}

.box1{

width: 300px;

height: 300px;

background-color: orange;

/\* 将元素设置为单元格 td \*/

display: table-cell;

vertical-align: middle;

}

.box2{

width: 100px;

height: 100px;

background-color: yellow;

margin: 0 auto;

}

</style>

</head>

<body>

<div class="box1">

<div class="box2"></div>

</div>

<table>

<tr>

<td>学号</td>

<td>姓名</td>

<td>性别</td>

<td>年龄</td>

<td>地址</td>

</tr>

<tr>

<td>1</td>

<td>孙悟空</td>

<td>男</td>

<td>18</td>

<td>花果山</td>

</tr>

<tr>

<td>2</td>

<td>猪八戒</td>

<td>男</td>

<td>28</td>

<td>高老庄</td>

</tr>

<tr>

<td>3</td>

<td>沙和尚</td>

<td>男</td>

<td>38</td>

<td>流沙河</td>

</tr>

<tr>

<td>4</td>

<td>唐僧</td>

<td>男</td>

<td>16</td>

<td>女儿国</td>

</tr>

<tr>

<td>1</td>

<td>孙悟空</td>

<td>男</td>

<td>18</td>

<td>花果山</td>

</tr>

<tr>

<td>2</td>

<td>猪八戒</td>

<td>男</td>

<td>28</td>

<td>高老庄</td>

</tr>

<tr>

<td>3</td>

<td>沙和尚</td>

<td>男</td>

<td>38</td>

<td>流沙河</td>

</tr>

<tr>

<td>4</td>

<td>唐僧</td>

<td>男</td>

<td>16</td>

<td>女儿国</td>

</tr>

<tr>

<td>1</td>

<td>孙悟空</td>

<td>男</td>

<td>18</td>

<td>花果山</td>

</tr>

<tr>

<td>2</td>

<td>猪八戒</td>

<td>男</td>

<td>28</td>

<td>高老庄</td>

</tr>

<tr>

<td>3</td>

<td>沙和尚</td>

<td>男</td>

<td>38</td>

<td>流沙河</td>

</tr>

<tr>

<td>4</td>

<td>唐僧</td>

<td>男</td>

<td>16</td>

<td>女儿国</td>

</tr>

</table>

</body>

</html>

# 表单

·邮箱检验

<input type="email">

·颜色选择器

<input type="color">

·autocomplete="off"

-- 关闭自动补全

<input type="text" name="username" autocomplete="off">

·readonly

- 将表单项设置为只读，数据会提交

<input type="text" name="username" value="hello" readonly>

·disabled

- 将表单项设置为禁用，数据不会提交

<input type="text" name="username" disabled>

·autofocus

- 设置表单项自动获取焦点

<input type="text" name="username" autofocus>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

</head>

<body>

<form action="target.html">

<input type="text" name="username" autocomplete="off">

<input type="text" name="username" value="hello" readonly>

<br><br>

<input type="text" name="username" autofocus>

<br><br>

<input type="text" name="b">

<br><br>

<!-- <input type="color">-->

<br><br>

<!-- <input type="email"> -->

<br><br>

<input type="submit">

<!-- 重置按钮 -->

<input type="reset">

<!-- 普通的按钮 -->

<input type="button" value="按钮">

<br><br>

<button type="submit">提交</button>

<button type="reset">重置</button>

<button type="button">按钮</button>

</form>

</body>

</html>

# 过渡

·通过过渡可以指定一个属性发生变化时的切换方式

·通过过渡可以创建一些非常好的效果，提升用户的体验

·transition

- 当高度发生变化的时候，有2秒过渡效果

transition: height 2s;

- 当全部发生变化的时候，有2秒过渡效果

transition: all 2s;

- 如果所有属性都需要过渡，则使用all关键字

transition-property: all;

- 大部分属性都支持过渡效果，注意过渡时必须是从一个有效数值向另外一个有效数值进行过渡

·transition-property

- 指定要执行过渡的属性

transition-property: width;

- 多个属性间使用,隔开

transition-property: width,height;

·transition-duration

- 指定过渡效果的持续时间

transition-duration: 2s;

- 时间单位

--可使用 s和ms 1s = 1000ms

- 可以分别指定不同属性的动画执行时间

/\*指定要执行过渡的属性\*/

transition-property: height,width,background-color;

/\*指定过渡效果的持续时间\*/

transition-duration: 2s,1s,3s;

·transition-timing-function

- 过渡的时序函数

- 指定过渡的执行的方式

- ease

-- 默认值，慢速开始，先加速，再减速

- linear

-- 匀速运动

- ease-in

-- 加速运动

- ease-out

-- 减速运动

- ease-in-out

-- 先加速 后减速

transition-timing-function: ease;

- cubic-bezier()

-- 贝塞尔曲线函数

-- <https://cubic-bezier.com>

transition-timing-function: cubic-bezier(.24,.95,.82,-0.88);

- steps()

-- 分步执行过渡效果

-- 可以设置一个第二个值

--- end ， 在时间结束时执行过渡(默认值)

--- start ， 在时间开始时执行过渡

transition-timing-function: steps(2, start);

·transition-delay

-- 过渡效果的延迟，等待一段时间后在执行过渡

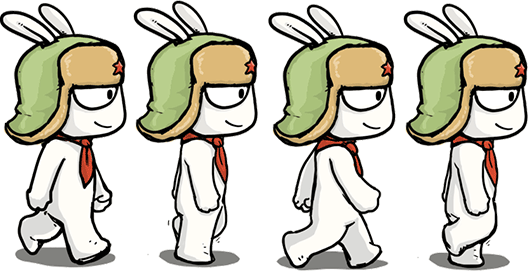
transition-delay: 2s;

·transition

- 可以同时设置过渡相关的所有属性，只有一个要求，如果要写延迟，则两个时间中第一个是持续时间，第二个是延迟

transition:2s margin-left 1s cubic-bezier(.24,.95,.82,-0.88);

## 米兔练习



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style type="text/css">

.box1{

height: 271px;

width: 132px;

margin: 0 auto;

background-image: url("imgs/bigtap-mitu-queue-big.png") ;

background-position: 0 0;

/\*动画执行两秒，分3个步骤\*/

transition: 2s steps(3);

}

.box1:hover{

background-position: -13px 0;

}

</style>

</head>

<body>

<div class="box1">

</div>

</body>

</html>

# 动画

·动画和过渡类似，都是可以实现一些动态的效果

·不同的是过渡需要在某个属性发生变化时才会触发

·动画可以自动触发动态效果

·设置动画效果，必须先要设置一个关键帧，关键帧设置了动画执行每一个步骤

·@keyframes 关键帧名{

from{}

to{}

}

- 创建关键帧

- test是关键帧的名字

- from表示动画的开始位置，也可以使用 0%

- to动画的结束位置，也可以使用100%

@keyframes test {

from{

margin-left: 0;

background-color: orange;

}

to{

margin-left: 700px;

background-color: red;

}

}

·animation-name: 关键帧名

- 为某元素设置动画

animation-name: test;

·animation-duration: 时长

- 设置动画执行时长

animation-duration: 2s;

·animation-delay: 时长

- 设置动画延时

animation-delay: 2s;

·animation-timing-function: ease-in-out

- 设置动画的时序函数

- 指定动画的执行的方式

- ease

-- 默认值，慢速开始，先加速，再减速

- linear

-- 匀速运动

- ease-in

-- 加速运动

- ease-out

-- 减速运动

- ease-in-out

-- 先加速 后减速

animation-timing-function: ease-in-out;

·animation-iteration-count：次数

- 动画执行的次数

- infinite

-- 重复循环执行

animation-iteration-count: 3;

·animation-direction

- 指定动画运行的方向

- normal

-- 默认值，从 from 向 to运行 每次都是这样

- reverse

-- 从 to 向 from 运行 每次都是这样

- alternate

-- 从 from 向 to往返执行

- alternate-reverse

-- 从 to 向 from往返执行

animation-direction: alternate;

·animation-play-state

- 设置动画的执行状态

- running

-- 默认值 动画执行

- paused

--动画暂停

animation-play-state: paused;

· animation-fill-mode

- 动画的填充模式

- none

-- 默认值 动画执行完毕元素回到原来位置

- forwards

-- 动画执行完毕元素会停止在动画结束的位置

- backwards

-- 动画延时等待时，元素就会处于开始位置

- both

-- 结合了forwards 和 backwards

animation-fill-mode: both;

·animation

- 可以同时设置动画相关的所有属性

animation: test 2s 2 1s alternate;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

}

.box1{

width: 800px;

height: 800px;

background-color: silver;

overflow: hidden;

}

.box1 div{

width: 100px;

height: 100px;

margin-bottom: 100px;

margin-left: 10px;

}

.box2{

background-color: #bfa;

/\*设置box2的动画\*/

animation-name: test;

/\*动画的执行时间\*/

animation-duration: 2s;

/\*延时几秒执行\*/

animation-delay: 2s;

/\*设置动画过渡的时序函数\*/

animation-timing-function: ease-in-out;

/\*动画执行的次数\*/

animation-iteration-count: 3;

/\*从from到to往返执行\*/

animation-direction: alternate;

/\*控制动画暂停\*/

/\*animation-play-state: paused;\*/

animation-fill-mode: both;

/\*animation: test 2s 2 1s alternate;\*/

}

.box1:hover div{

animation-play-state: paused;

}

/\*@keyframes创建关键帧，test是关键帧的名字\*/

@keyframes test {

/\* from表示动画的开始位置 也可以使用 0% \*/

from{

margin-left: 0;

background-color: orange;

}

/\* to动画的结束位置 也可以使用100%\*/

to{

margin-left: 700px;

background-color: red;

}

}

</style>

</head>

<body>

<div class="box1">

<div class="box2"></div>

<!-- <div class="box3"></div> -->

</div>

</body>

</html>

## 奔跑的少年

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.box1{

width: 256px;

height: 256px;

margin: 0 auto;

background-image: url("imgs/bg2.png");

animation: run 0.5s steps(6) infinite;

}

@keyframes run {

from{

background-position: 0 0;

}

to{

background-position: -1536px 0;

}

}

</style>

</head>

<body>

<div class="box1"></div>

</body>

</html>

## 关键帧

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

.outer{

height: 500px;

border-bottom: 10px black solid;

margin: 50px auto;

overflow: hidden;

}

.outer div{

float: left;

width: 100px;

height: 100px;

border-radius: 50%;

background-color: #bfa;

animation: ball .5s forwards linear infinite alternate;

}

div.box2{

background-color: orange;

animation-delay: .1s;

}

div.box3{

background-color: yellow;

animation-delay: .2s;

}

div.box4{

background-color: yellowgreen;

animation-delay: .3s;

}

div.box5{

background-color: blue;

animation-delay: .4s;

}

div.box6{

background-color: pink;

animation-delay: .5s;

}

div.box7{

background-color: tomato;

animation-delay: .6s;

}

div.box8{

background-color: skyblue;

animation-delay: .7s;

}

div.box9{

background-color: chocolate;

animation-delay: .8s;

}

/\* 创建小球下落的动画 \*/

@keyframes ball {

from{

margin-top: 0;

}

to{

margin-top: 400px;

}

/\* 2 to{

margin-top: 400px;

animation-timing-function: ease-in;

}

40%{

margin-top: 100px;

}

80%{

margin-top: 200px;

} \*/

}

</style>

</head>

<body>

<div class="outer">

<div class="box1"></div>

<div class="box2"></div>

<div class="box3"></div>

<div class="box4"></div>

<div class="box5"></div>

<div class="box6"></div>

<div class="box7"></div>

<div class="box8"></div>

<div class="box9"></div>

</div>

</body>

</html>

# 变形平移

·变形就是指通过CSS来改变元素的形状或位置

·变形不会影响到页面的布局

·transform

- 用来设置元素的变形效果

- translateX()

-- 沿着x轴方向平移

- translateY()

-- 沿着y轴方向平移

- translateZ()

-- 沿着z轴方向平移

- 平移元素，百分比是相对于自身计算的

transform: translateX(100%);

·使用变形平移将不确定大小的元素水平居中

/\*使用变形平移将元素居中\*/

left: 50%;

top: 50%;

transform: translateX(-50%) translateY(-50%);

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

body{

background-color: rgb(236, 236, 236);

}

.box1{

width: 200px;

height: 200px;

background-color: #bfa;

margin: 0 auto;

margin-top: 200px;

transform: translateX(100%);

}

.box2{

width: 200px;

height: 200px;

background-color: #0086b3;

margin: 0 auto;

margin-top: 200px;

}

/\*.box3{\*/

/\* background-color: orange;\*/

/\* position: absolute;\*/

/\* !\*使用变形平移将元素居中\*!\*/

/\* left: 50%;\*/

/\* top: 50%;\*/

/\* transform: translateX(-50%) translateY(-50%);\*/

/\*}\*/

.box4,.box5{

width: 220px;

height: 300px;

background-color: #019e8b;

float: left;

margin: 0 10px;

transition:all .3s;

}

.box4:hover,.box5:hover{

transform: translateY(-10px);

box-shadow: 0 0 10px rgba(0,0,0,.3);

}

</style>

</head>

<body>

<div class="box1"></div>

<div class="box2"></div>

<!--<div class="box3">aaa</div>-->

<div class="box4">

</div>

<div class="box5">

</div>

</body>

</html>

## Z轴平移

·调整元素在z轴的位置，正常情况就是调整元素和人眼之间的距离

·距离越大，元素离人越近

·z轴平移属于立体效果（近大远小），默认情况下网页是不支持透视，如果需要看见效果，必须要设置网页的视距

·perspective: 800px;

- 设置当前网页的视距为800px，人眼距离网页的距离

perspective: 800px;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

html{

/\* 设置当前网页的视距为800px，人眼距离网页的距离 \*/

perspective: 800px;

}

body{

border: 1px red solid;

background-color: rgb(241, 241, 241);

}

.box1{

width: 200px;

height: 200px;

background-color: #bfa;

margin: 200px auto;

transition:2s;

}

body:hover .box1{

transform: translateZ(800px);

}

</style>

</head>

<body>

<div class="box1"></div>

</body>

</html>

## 旋转

·通过旋转可以使元素沿着x y 或 z旋转指定的角度

·transform

- rotateX()

- rotateY()

- rotateZ()

/\*顺时针旋转45度\*/

transform: rotateZ(45deg);

·backface-visibility

- 是否显示元素的背面

- visible

-- 显示，默认值

- hidden

-- 不显示

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

html{

/\* 设置当前网页的视距为800px，人眼距离网页的距离 \*/

perspective: 800px;

}

body{

border: 1px red solid;

background-color: rgb(241, 241, 241);

}

.box1{

width: 200px;

height: 200px;

background-color: #bfa;

margin: 200px auto;

transition:2s;

}

body:hover .box1{

/\*顺时针旋转45度\*/

/\*transform: rotateZ(180deg);\*/

transform: rotateY(180deg);

/\*transform: rotateX(45deg);\*/

/\* transform: rotateZ(.25turn); \*/

/\* transform: rotateY(180deg) translateZ(400px);\*/

/\* transform: translateZ(400px) rotateY(180deg) ; \*/

/\* 是否显示元素的背面 \*/

backface-visibility: hidden;

}

</style>

</head>

<body>

<div class="box1"></div>

</body>

</html>

## 钟表

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

/\*清除默认样式\*/

\* {

margin: 0;

padding: 0;

}

/\* 设置表的样式 \*/

.clock{

width: 500px;

height: 500px;

/\*background-color: #0086b3;\*/

/\*水平居中\*/

margin: 100px auto;

/\*圆形\*/

border-radius: 50%;

/\*边框\*/

/\*border: 10px solid black;\*/

position: relative;

background-image: url("imgs/bg3.jpg");

background-size: cover;

}

.clock > div{

/\*使子元素在父元素中水平垂直居中\*/

position: absolute;

margin: auto;

left: 0;

right: 0;

top: 0;

bottom: 0;

}

/\*时针包裹器样式\*/

.hour-wrapper{

height: 70%;

width: 70%;

/\*background-color: #0086b3;\*/

/\*时针分步旋转\*/

animation: run 43200s steps(12) infinite;

/\*animation: run 720s steps(12) infinite;\*/

}

/\*时针样式\*/

.hour{

height: 50%;

width: 6px;

background-color: black;

/\*水平居中\*/

margin: 0 auto;

}

/\*分针包裹器样式\*/

.min-wrapper{

height: 80%;

width: 80%;

/\*background-color: #0086b3;\*/

/\*分针分步旋转\*/

animation: run 3600s steps(60) infinite;

/\*测试\*/

/\*animation: run 60s steps(60) infinite;\*/

}

/\*分针样式\*/

.min{

height: 50%;

width: 4px;

background-color: black;

/\*水平居中\*/

margin: 0 auto;

}

/\*秒针包裹器样式\*/

.sec-wrapper{

height: 95%;

width: 95%;

/\*background-color: #0086b3;\*/

/\*使秒针包裹器旋转\*/

animation: run 60s linear infinite;

/\*测试\*/

/\*animation: run 1s linear infinite;\*/

/\*分步旋转\*/

/\*animation: run 60s steps(60) infinite;\*/

}

/\*秒针样式\*/

.sec{

height: 50%;

width: 2px;

background-color: red;

/\*水平居中\*/

margin: 0 auto;

}

/\*秒针\*/

/\*!\*原理\*!\*/

/\*.sec-wrapper{\*/

/\* width: 500px;\*/

/\* height: 500px;\*/

/\* background-color: #0086b3;\*/

/\* !\*让秒针包裹容器旋转\*!\*/

/\* animation: run 60s linear infinite;\*/

/\*}\*/

/\*.sec{\*/

/\* height: 250px;\*/

/\* width: 4px;\*/

/\* background-color: red;\*/

/\* !\*水平居中\*!\*/

/\* margin: 0 auto;\*/

/\*}\*/

/\*创建旋转关键帧动画\*/

@keyframes run {

from{

/\*旋转起始位置\*/

transform: rotateZ(0);

}

to{

/\*旋转终止位置\*/

transform: rotateZ(360deg);

}

}

</style>

</head>

<body>

<!-- 创建表的容器 -->

<div class="clock">

<!-- 创建时针 -->

<div class="hour-wrapper">

<div class="hour"></div>

</div>

<!-- 创建分针 -->

<div class="min-wrapper">

<div class="min"></div>

</div>

<!-- 创建秒针 -->

<div class="sec-wrapper">

<div class="sec"></div>

</div>

</div>

</body>

</html>

## 立方体科幻图

·transform-style: preserve-3d

- 开启3D效果

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

html {

/\*设置当前网页的视距为800px\*/

perspective: 800px

}

body{

background-color: black;

}

.cube{

width: 200px;

height: 200px;

/\*background-color: #019e8b;\*/

margin: 100px auto;

/\*开启3D效果\*/

transform-style: preserve-3d;

/\*旋转立方体\*/

animation: rotate 20s infinite linear;

/\*transform: rotateX(45deg) rotateZ(45deg);\*/

}

.cube > div{

width: 200px;

height: 200px;

/\*设置透明度\*/

opacity: 0.7;

/\*开启绝对定位,让每张图片重叠\*/

position: absolute;

}

img{

/\*取消图片的基线对齐\*/

vertical-align: top;

}

/\*左右两张图\*/

.box1{

/\*y轴向右旋转90度,向右平移100px\*/

transform: rotateY(90deg) translateZ(100px);

}

.box2{

/\*y轴向左旋转90度,向左平移100px\*/

transform: rotateY(-90deg) translateZ(100px);

}

/\*上下两张图\*/

.box3{

/\*x轴向上旋转90度,向上平移100px\*/

transform: rotateX(90deg) translateZ(100px);

}

.box4{

/\*X轴向下旋转90度,向下平移100px\*/

transform: rotateX(-90deg) translateZ(100px);

}

/\*前后两张图\*/

.box5{

/\*Y轴向右旋转180度,向前平移100px\*/

transform: rotateY(180deg) translateZ(100px);

}

.box6{

/\*Y轴向右旋转0度,向下平移100px\*/

transform: rotateY(0deg) translateZ(100px);

}

@keyframes rotate {

from{

transform: rotateX(0) rotateZ(0);

}

to{

transform: rotateX(360deg) rotateZ(360deg);

}

}

</style>

</head>

<body>

<!-- 创建一个外部的容器 -->

<div class="cube">

<!-- 引入图片 -->

<div class="box1">

<img src="./imgs/14/1.jpg">

</div>

<div class="box2">

<img src="./imgs/14/2.jpg">

</div>

<div class="box3">

<img src="./imgs/14/3.jpg">

</div>

<div class="box4">

<img src="./imgs/14/4.jpg">

</div>

<div class="box5">

<img src="./imgs/14/5.jpg">

</div>

<div class="box6">

<img src="./imgs/14/6.jpg">

</div>

</div>

</body>

</html>

# 缩放

· transform:scaleX(倍数)

- 水平方向缩放

· transform:scaleY(倍数)

- 垂直方向缩放

· transform:scale(倍数)

- 双方向缩放

·transform-origin:20px 20px;

- 设置变形的原点

- 默认值 center

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

html{

/\*设置当前网页的视距为800px\*/

perspective:800px;

}

.box1{

width: 100px;

height: 100px;

background-color: #bfa;

/\*2s过渡时长\*/

transition:2s;

margin: 100px auto;

/\* 变形的原点 默认值 center\*/

/\* transform-origin:20px 20px; \*/

}

.box1:hover{

transform:scale(2);

}

.img-wrapper{

width: 200px;

height: 200px;

border: 1px red solid;

/\*裁剪溢出\*/

overflow: hidden;

}

img{

/\*0.2s过渡时长\*/

transition: .2s;

}

.img-wrapper:hover img{

transform:scale(1.2);

}

</style>

</head>

<body>

<div class="box1"></div>

<div class="img-wrapper">

<img src="imgs/an.jpg" width="100%">

</div>

</body>

</html>

# less

·css的预处理语言

·less是一个css的增强版，通过less可以编写更少的代码实现更强大的样式

·在less中添加了许多的新特性：像对变量的支持、对mixin的支持

·less的语法大体上和css语法一致

·但是less中增添了许多对css的扩展，所以浏览器无法直接执行less代码

·要执行必须向将less转换为css，然后再由浏览器执行

## CSS原生变量

·设置变量

html{

/\*设置变量\*/

--color:#ff0;

--lenth:100px;

}

·使用变量

.box1{

/\*width: 100px;\*/

/\*height: 100px;\*/

width: var(--lenth);

height: var(--lenth);

/\*使用变量设置背景颜色\*/

background-color: var(--color);

}

·calc()

- 计算函数

width: calc(200px/2);

·ie浏览器大多数不支持

## less环境配置

下载node.js

<https://nodejs.org/zh-cn/>

安装

<https://segmentfault.com/a/1190000023390756>

·查看是否安装成功

- node -v 查看 node 版本

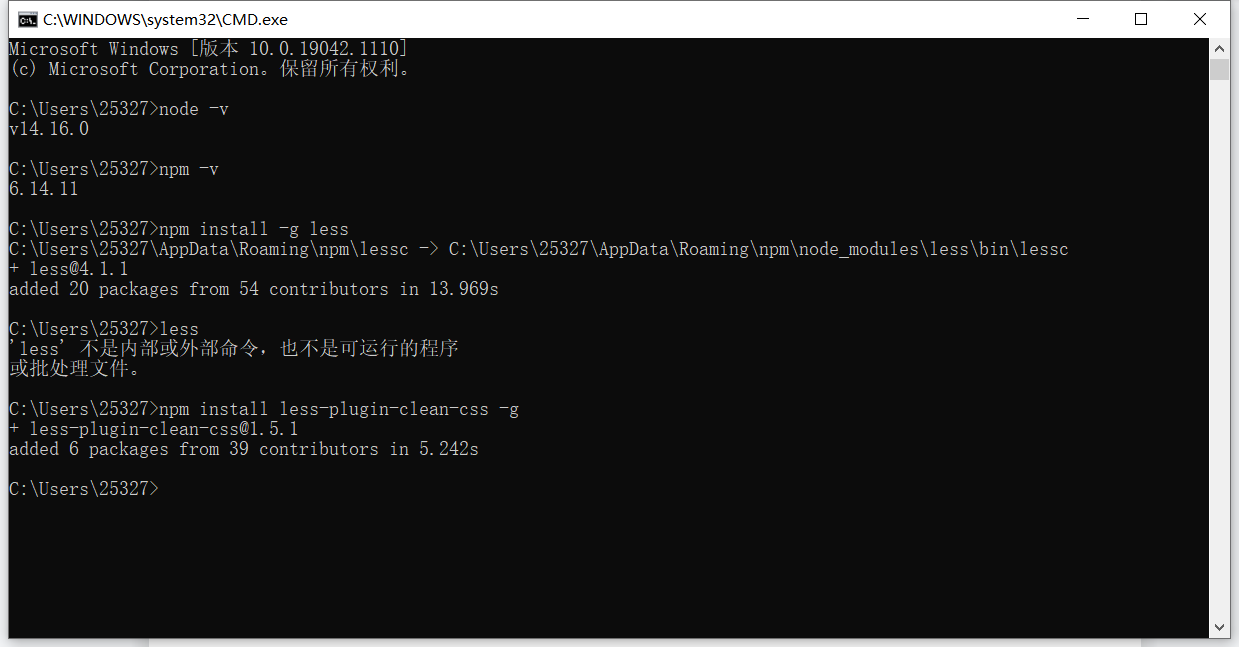
- npm -v 查看 npm 版本

·安装less

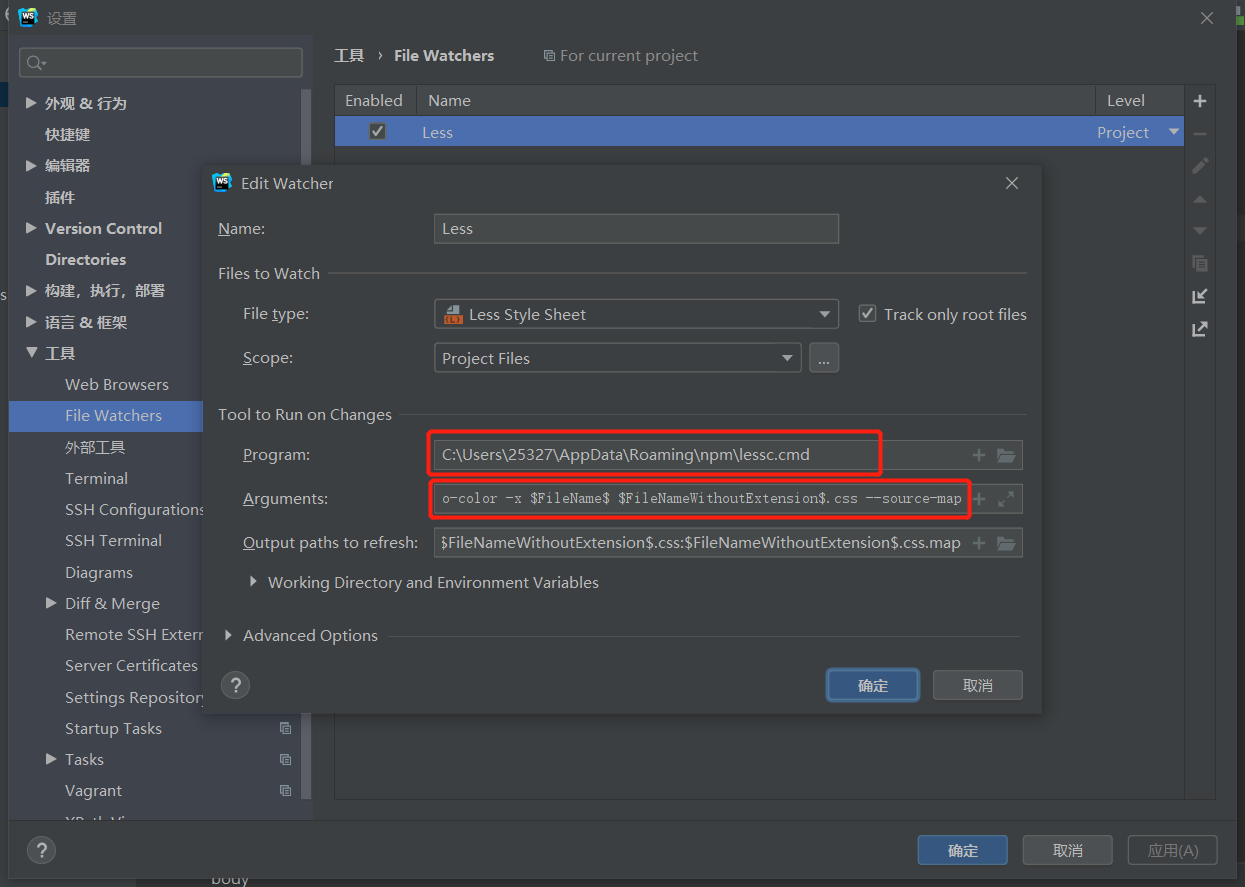
- npm install -g less

·安装less-plugin-clean-css插件（less的插件，用于压缩代码）

- npm install less-plugin-clean-css -g



·在webstorm中配置



·C:\Users\25327\AppData\Roaming\npm\lessc.cmd

·--clean-css --no-color -x $FileName$ $FileNameWithoutExtension$.css --source-map

## Less语法

·//

- less中的单行注释，注释中的内容不会被解析到css中

// less中的单行注释，注释中的内容不会被解析到css中

·/\* \*/

- css中的注释，内容会被解析到css文件中

/\*

css中的注释，内容会被解析到css文件中

\*/

·选择器的嵌套使用

//为box1设置样式

.box1{

width: 200px;

height: 200px;

background-color: #0077aa;

//为box1的后代元素box2设置样式

.box2{

width: 100px;

height: 100px;

background-color: #9a6e3a;

//为box2的后代元素box4设置样式

.box4{

width: 50px;

height: 50px;

background-color: #795da3;

}

}

//为box1的后代元素box3设置样式

.box3{

width: 50px;

height: 50px;

background-color: #019e8b;

}

}

## 变量

·@变量名:变量值

- 定义变量

@a:100px;

·变量的使用

- 直接使用则以 @变量名 的形式使用即可

width: @a;

- 作为类名，或者一部分值使用时必须以 @{变量名} 的形式使用

.@{c}{

width: @a;

height: @a;

background-image: url("../@{d}/an.jpg");

background-size: cover;

}

·变量发生重名时，会优先使用比较近的变量

·可以在变量声明前就使用变量

div{

// 变量发生重名时，会优先使用比较近的变量

@d:115px;

width: @d;

height: @e;

}

//可以在变量声明前就使用变量

@e:335px;

·直接引用上面width的值

div{

width: 300px;

// 新版的语法，直接引用上面width的值

height: $width;

}

// less中的单行注释，注释中的内容不会被解析到css中

/\*

css中的注释，内容会被解析到css文件中

\*/

//为box1设置样式

.box1{

width: 200px;

height: 200px;

background-color: #0077aa;

//为box1的后代元素box2设置样式

.box2{

width: 100px;

height: 100px;

background-color: #9a6e3a;

//为box2的后代元素box4设置样式

.box4{

width: 50px;

height: 50px;

background-color: #795da3;

}

}

//为box1的后代元素box3设置样式

.box3{

width: 50px;

height: 50px;

background-color: #019e8b;

}

}

//变量，在变量中可以存储一个任意的值

// 并且我们可以在需要时，任意的修改变量中的值

// 变量的语法： @变量名

@a:100px;

@b:#bfa;

@c:box6;

@d:imgs;

.box5{

//使用变量时，如果是直接使用则以 @变量名 的形式使用即可

width: @a;

height: @a;

background-color: @b;

}

//作为类名，或者一部分值使用时必须以 @{变量名} 的形式使用

.@{c}{

width: @a;

height: @a;

background-image: url("../@{d}/an.jpg");

background-size: cover;

}

@d:200px;

@d:300px;

div{

// 变量发生重名时，会优先使用比较近的变量

@d:115px;

width: @d;

height: @e;

}

//可以在变量声明前就使用变量

@e:335px;

div{

width: 300px;

// 新版的语法，直接引用上面width的值

height: $width;

}

## 父元素和扩展

//为box1设置样式

.box1{

//为box1的后代元素box2设置样式

.box2{

background-color: #990055;

//为box2的子元素box3设置样式

> .box3{

background-color: #795da3;

//为box3设置一个hover

&:hover{

color: blue;

}

}

}

//为box1设置一个hover

//& 就表示外层的父元素

&:hover{

color: orange;

}

div &{

width: 100px;

}

}

.p1{

width: 100px;

height: 200px;

}

//:extend() 对当前选择器继承指定选择器的样式（选择器分组）

//.p2继承.p1的样式（选择器分组）

.p2:extend(.p1){

color: red;

}

.p3{

//直接对指定的样式进行引用，这里就相当于将p1的样式在这里进行了复制

//mixin 混合

.p1();

}

// 使用类选择器时可以在选择器后边添加一个括号，这时我们实际上就创建了一个mixins

//创建.p4选择器方法

.p4(){

width: 100px;

height: 100px;

}

.p5{

//使用p4选择器方法

.p4;

}

## 混合函数

//混合函数 在混合函数中可以直接设置变量，并为变量设置默认值

.test(@w:100px,@h:200px,@bg-color:red){

width: @w;

height: @h;

border: 1px solid @bg-color;

}

div{

//调用混合函数，按顺序传递参数

// .test(200px,300px,#bfa);

.test(300px);

// .test(@bg-color:red, @h:100px, @w:300px);

}

span{

//average对颜色取平均数

color: average(red,blue);

}

html{

width: 100%;

height: 100%;

}

body {

width: 100%;

height: 100%;

background-color: #bfa;

}

body:hover{

//darken对颜色进行加深

background-color: darken(#bfa,50%);

}

## less补充

//import用来将其他的less引入到当前的less

//可以通过import来将其他的less引入到当前的less中

@import "syntax2.less";

.box1{

// 在less中所有的数值都可以直接进行运算

// + - \* /

width: 100px + 100px;

height: 100px/2;

background-color: #bfa;

}

# 弹性盒

·是CSS中的又一种布局手段，它主要用来代替浮动来完成页面的布局

·flex可以使元素具有弹性，让元素可以跟随页面的大小的改变而改变

·弹性容器

- 要使用弹性盒，必须先将一个元素设置为弹性容器

- 我们通过 display 来设置弹性容器

-- display:flex 设置为块级弹性容器

-- display:inline-flex 设置为行内的弹性容器

- 其子元素会自动变成弹性元素

·弹性元素

- 弹性容器的子元素是弹性元素（弹性项）

·弹性元素可以同时是弹性容器

·flex-direction

- 指定容器中弹性元素的排列方式

- row

-- 默认值，弹性元素在容器中水平排列（左向右）

-- 主轴 自左向右

- row-reverse

-- 弹性元素在容器中反向水平排列（右向左）

-- 主轴 自右向左

- column

-- 弹性元素纵向排列（自上向下）

- column-reverse

-- 弹性元素方向纵向排列（自下向上）

flex-direction: row;

·主轴

- 弹性元素的排列方向称为主轴

·侧轴

- 与主轴垂直方向的称为侧轴

·flex-grow

- 设置子元素的大小权重值

- 指定弹性元素的伸展的系数

- 当父元素有多余空间的时，子元素如何伸展

- 父元素的剩余空间，会按照比例进行分配

- 默认值为0，不伸展

·flex-shrink

- 指定弹性元素的收缩系数

- 当父元素中的空间不足以容纳所有的子元素时，对子元素进行收缩

- 默认值是1，等比例收缩

- 0是不收缩

- 值越大，收缩的越多

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

/\*清除列表样式\*/

list-style: none;

}

ul{

width: 800px;

border: 10px red solid;

/\*将ul设置为弹性容器，可使li水平排列，并且不会高度塌陷\*/

display: flex;

/\*也可以用此方法设置弹性容器，不会独占一行\*/

/\*display: inline-flex;\*/

/\*弹性元素在容器中水平排列（左向右）\*/

flex-direction: row;

}

li{

width: 200px;

height: 100px;

background-color: #bfa;

font-size: 50px;

text-align: center;

line-height: 100px;

/\*容易导致高度塌陷\*/

/\*float: left;\*/

/\*弹性元素可以同时是弹性容器\*/

/\*display: flex;\*/

/\*设置子元素大小权重，都为1，如果宽度相同，则相同大小\*/

/\*flex-grow: 1;\*/

/\*flex-shrink: 0;\*/

}

li:nth-child(1){

/\*flex-grow: 0;\*/

/\*flex-shrink: 1;\*/

}

li:nth-child(2){

background-color: pink;

/\*flex-grow: 2;\*/

/\*flex-shrink: 2;\*/

}

li:nth-child(3){

background-color: orange;

/\*flex-grow: 3;\*/

/\*flex-shrink: 3;\*/

}

</style>

</head>

<body>

<ul>

<li>1</li>

<li>2</li>

<li>3</li>

</ul>

</body>

</html>

## 弹性导航条

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

/\*清除列表样式\*/

list-style: none;

}

.nav{

width: 1210px;

height: 48px;

/\*字体垂直居中\*/

line-height: 48px;

margin: 50px auto;

background-color: #e8e7e3;

/\*将导航栏设置为弹性容器\*/

display: flex;

}

.nav li{

/\*设置元素权重\*/

flex-grow: 1;

}

.nav a{

/\*将超链接设置为块元素\*/

display: block;

color: #808080;

/\*去除超链接下划线\*/

text-decoration: none;

font-size: 16px;

/\*文字居中\*/

text-align: center;

}

.nav a:hover{

background-color: #636363;

color: white;

}

</style>

</head>

<body>

<ul class="nav">

<li><a href="#">HTML/CSS</a></li>

<li><a href="#">Browser Side</a></li>

<li><a href="#">Server Side</a></li>

<li><a href="#">Programming</a></li>

<li><a href="#">XML</a></li>

<li><a href="#">Web Building</a></li>

<li><a href="#">Reference</a></li>

</ul>

</body>

</html>

## 弹性容器的样式

·flex-wrap

- 设置弹性元素是否在弹性容器中自动换行

- nowrap

-- 默认值，元素不会自动换行

- wrap

-- 元素沿着侧轴方向自动换行

- wrap-reverse

-- 元素沿着侧轴反方向换行

flex-wrap: wrap;

·flex-flow

- 可同时设置wrap 和 direction 的属性

flex-flow: row wrap;

·justify-content

- 如何分配主轴上的空白空间（主轴上的元素如何排列）

- flex-start

-- 元素沿着主轴起边排列

- flex-end

-- 元素沿着主轴终边排列

- center

-- 元素居中排列

- space-around

-- 空白分布到元素两侧

- space-between

-- 空白均匀分布到元素间

- space-evenly

-- 空白分布到元素的单侧

justify-content: center

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

/\*清除列表样式\*/

list-style: none;

}

ul{

width: 800px;

border: 10px red solid;

/\*设置ul为弹性容器\*/

display: flex;

/\*溢出元素自动换行\*/

flex-wrap: wrap;

/\*自左向右排列并自动换行\*/

/\*flex-flow: row wrap;\*/

/\*元素在主轴中居中排列\*/

/\*justify-content: center\*/

}

li{

width: 200px;

height: 100px;

background-color: #bfa;

font-size: 50px;

text-align: center;

line-height: 100px;

/\*不收缩，会溢出\*/

flex-shrink: 0;

}

li:nth-child(1){

}

li:nth-child(2){

background-color: pink;

}

li:nth-child(3){

background-color: orange;

}

</style>

</head>

<body>

<ul>

<li>1</li>

<li>2</li>

<li>3</li>

</ul>

</body>

</html>

·align-items

- 元素在辅轴上如何对齐

- 元素间的关系

- stretch

-- 默认值，将元素的长度设置为相同的值

- flex-start

-- 元素不会拉伸，沿着辅轴起边对齐

- flex-end

-- 沿着辅轴的终边对齐

- center

-- 居中对齐

- baseline

-- 基线对齐

align-items: center;

·垂直水平居中对齐

justify-content: center;

align-items: center;

·align-content

- 侧轴空白空间的分布

- flex-start

-- 元素沿着侧轴起边排列

- flex-end

-- 元素沿着侧轴终边排列

- center

-- 元素居中排列

- space-around

-- 空白分布到元素两侧

- space-between

-- 空白均匀分布到元素间

- space-evenly

-- 空白分布到元素的单侧

align-content: space-between;

·align-self

- 为某个子元素单独设置自身的align-items

align-self: stretch;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

/\*清除列表样式\*/

list-style: none;

}

ul{

width: 800px;

height: 800px;

border: 10px red solid;

/\*设置ul为弹性容器\*/

display: flex;

/\*溢出元素自动换行\*/

flex-wrap: wrap;

/\*自左向右排列并自动换行\*/

/\*flex-flow: row wrap;\*/

/\*元素在主轴中居中排列\*/

/\*justify-content: center\*/

/\*justify-content: center;\*/

align-items: center;

align-content: space-between;

}

li{

width: 200px;

background-color: #bfa;

font-size: 50px;

text-align: center;

line-height: 100px;

/\*不收缩，会溢出\*/

flex-shrink: 0;

}

li:nth-child(1){

/\* align-self: 为某个子元素单独设置自身的align-items \*/

align-self: stretch;

}

li:nth-child(2){

background-color: pink;

}

li:nth-child(3){

background-color: orange;

}

li:nth-child(4){

background-color: yellow;

}

li:nth-child(5){

background-color: chocolate;

}

</style>

</head>

<body>

<ul>

<li>1</li>

<li>

2

<div>2</div>

</li>

<li>

3

<div>3</div>

<div>3</div>

</li>

<li>1</li>

<li>

2

<div>2</div>

</li>

</ul>

</body>

</html>

## 弹性元素的样式

·flex-grow

- 弹性的增长系数

- 详情跳转到67页

·flex-shrink

- 弹性元素的缩减系数

- 缩减系数的计算方式比较复杂

- 缩减多少是根据 缩减系数 和 元素大小来计算

- 详情跳转到67页

·flex-basis

- 指定的是元素在主轴上的基础长度

- 如果主轴是 横向的 则 该值指定的就是元素的宽度

- 如果主轴是 纵向的 则 该值指定的是就是元素的高度

- 默认值是 auto，表示参考元素自身的高度或宽度

- 如果传递了一个具体的数值，则以该值为准

flex-basis: 100px;

·flex

- 可以设置弹性元素所有的三个样式

- flex 增长 缩减 基础长度;

flex: 0 1 200px;

- initial

-- 相当于flex: 0 1 auto

- auto

-- 相当于flex: 1 1 auto

- none

-- 相当于0 0 auto

-- 弹性元素没有弹性

flex: initial;

·order

- 决定弹性元素的排列顺序

order: 1;

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

list-style: none;

}

ul{

width: 900px;

border: 10px red solid;

/\* 设置弹性盒 \*/

display: flex;

}

li{

width: 200px;

height: 100px;

background-color: #bfa;

font-size: 50px;

text-align: center;

line-height: 100px;

/\*flex: initial;\*/

/\*设置元素基础长度\*/

flex-basis: 100px;

}

li:nth-child(1){

/\* order 决定弹性元素的排列顺序 \*/

order: 2;

/\*设置增长、缩减、基础长度\*/

/\*flex: 0 1 200px;\*/

}

li:nth-child(2){

background-color: pink;

/\* flex-grow: 2; \*/

order: 3;

/\*flex: initial;\*/

}

li:nth-child(3){

background-color: orange;

/\* flex-grow: 3; \*/

order: 1;

}

</style>

</head>

<body>

<ul>

<li>1</li>

<li>2</li>

<li>3</li>

</ul>

</body>

</html>

## 淘宝移动端导航栏

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\* {

margin: 0;

padding: 0;

}

/\* 设置外层的容器 \*/

.nav{

width: 100%;

}

/\*设置每一行容器\*/

.nav-inner{

/\*设置为弹性容器\*/

display: flex;

/\*空白分布到元素两侧\*/

justify-content: space-around;

}

.item{

/\*background-color: #0086b3;\*/

/\*图片的宽度随窗口大小自动改变\*/

/\*flex: auto;\*/

/\*设置图片的宽度\*/

width: 18%;

text-align: center;

}

.item img{

/\*设置元素的宽度和父元素宽度一样\*/

width: 100%;

}

.item a{

color: #333;

text-decoration: none;

font-size: 16px;

}

</style>

</head>

<body>

<!-- 创建一个外层的容器 -->

<nav class="nav">

<div class="nav-inner">

<div class="item">

<a href="#">

<img src="imgs/16/1.png">

<span>天猫</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/2.png">

<span>聚划算</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/3.png">

<span>天猫国际</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/4.png">

<span>外卖</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/5.png">

<span>天猫超市</span>

</a>

</div>

</div>

<div class="nav-inner">

<div class="item">

<a href="#">

<img src="imgs/16/6.png">

<span>充值中心</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/7.png">

<span>飞猪旅行</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/8.png">

<span>领金币</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/9.png">

<span>拍卖</span>

</a>

</div>

<div class="item">

<a href="#">

<img src="imgs/16/10.png">

<span>分类</span>

</a>

</div>

</div>

</nav>

</body>

</html>

# 像素

·像素：

- 屏幕是由一个一个发光的小点构成，这一个个的小点就是像素

- 分辨率：1920 x 1080 说的就是屏幕中小点的数量

- 在前端开发中像素要分成两种情况讨论：CSS像素 和 物理像素

- 物理像素，上述所说的小点点就属于物理像素

- CSS像素，编写网页时，我们所用像素都是CSS像素

- 浏览器在显示网页时，需要将CSS像素转换为物理像素然后再呈现

- 一个css像素最终由几个物理像素显示，由浏览器决定：

-- 默认情况下在pc端，一个css像素 = 一个物理像素

·视口（viewport）

- 视口就是屏幕中用来显示网页的区域

- 可以通过查看视口的大小，来观察CSS像素和物理像素的比值

- 默认情况下：

-- 视口宽度 1920px（CSS像素）

-- 1920px（物理像素）

-- 此时，css像素和物理像素的比是 1:1

-- 放大两倍的情况：

--- 视口宽度 960px（CSS像素）

--- 1920px（物理像素）

--- 此时，css像素和物理像素的比是1:2

--- 我们可以通过改变视口的大小，来改变CSS像素和物理像素的比值

## 手机像素

·在不同的屏幕，单位像素的大小是不同的，像素越小屏幕会越清晰

·智能手机的像素点 远远小于 计算机的像素点

·一个宽度为900px的网页在iphone6中要如何显示呢？

- 默认情况下，移动端的网页都会将视口设置为980像素（css像素）

- 以确保pc端网页可以在移动端正常访问

- 但是如果网页的宽度超过了980，移动端的浏览器会自动对网页缩放以完整显示网页

·所以基本大部分的pc端网站都可以在移动端中正常浏览，但是往往都不会有一个好的体验，为了解决这个问题，大部分网站都会专门为移动端设计网页

# 完美视口

·移动端默认的视口大小是980px(css像素)

·默认情况下，移动端的像素比就是 980/移动端宽度 （980/750）

·如果我们直接在网页中编写移动端代码，这样在980的视口下，像素比是非常不好，导致网页中的内容非常非常的小

·编写移动页面时，必须要确保有一个比较合理的像素比，比如：

- 1css像素 对应 2个物理像素

- 1css像素 对应 3个物理像素

·可以通过meta标签来设置视口大小

·每一款移动设备设计时，都会有一个最佳的像素比

·一般我们只需要将像素比设置为该值即可得到一个最佳效果

·将像素比设置为最佳像素比的视口大小我们称其为完美视口

·将网页的视口设置为完美视口

- <meta name="viewport" content="width=device-width, initial-scale=1.0">

- device-width表示设备的宽度（完美视口）

- 结论：以后再写移动端的页面，就把上边这个玩意先写上

<meta name="viewport" content="width=device-width, initial-scale=1.0">

# vw单位

·不同的设备完美视口的大小是不一样的

- iphone6 -- 375

- iphone6plus -- 414

·由于不同设备视口和像素比不同，所以同样的375个像素在不同的设备下意义是不一样，比如在iphone6中 375就是全屏，而到了plus中375就会缺一块

·所以在移动端开发时，就不能再使用px来进行布局了

·vw 表示的是视口的宽度（viewport width）

- 100vw = 一个视口的宽度

- 1vw = 1%视口宽度

·vw这个单位永远相当于视口宽度进行计算

设计图的宽度

750px 1125px

设计图

750px

使用vw作为单位

100vw

创建一个 48px x 35px 大小的元素

100vw = 750px(设计图的像素) 0.1333333333333333vw = 1px

6.4vw = 48px(设计图像素)

4.667vw = 35px

# vw适配

·网页中字体大小最小是12px，不能设置一个比12像素还小的字体

·如果我们设置了一个小于12px的字体，则字体自动设置为12

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<style>

\*{

margin: 0;

padding: 0;

}

html{

/\*

网页中字体大小最小是12px，不能设置一个比12像素还小的字体

如果我们设置了一个小于12px的字体，则字体自动设置为12

0.1333333vw = 1px

5.3333vw = 40px

\*/

font-size: 5.3333vw;

}

.box1{

/\*

rem

- 1 rem = 1 html的字体大小

- 1 rem = 40 px(设计图)

\*/

width: 18.75rem;

height: 0.875rem;

background-color: #bfa;

}

</style>

</head>

<body>

<!--

48 x 35

-->

<div class="box1"></div>

</body>

</html>

# 移动端页面

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>移动端页面</title>

<!--引入字体图标库-->

<link rel="stylesheet" href="fa/css/all.min.css" type="text/css">

<link rel="stylesheet" href="css/phonepage.css" type="text/css">

</head>

<body>

<!--创建头部容器-->

<header class="top-bar">

<!--菜单按钮-->

<div class="menu-btn">

<a href="#">

<!--菜单字体图标-->

<i class="fas fa-stream"></i>

</a>

</div>

<!--logo超链接-->

<h1 class="logo">

<a href="#">

I Learn

</a>

</h1>

<!--搜索按钮-->

<div class="search-btn">

<a href="#">

<!--搜索字体图标-->

<i class="fas fa-search"></i>

</a>

</div>

</header>

<!--轮播图-->

<div class="banner">

<a href="#">

<img src="img/17/banner.png">

</a>

</div>

<!--菜单-->

<nav class="menu">

<!--四个子菜单-->

<a href="#" class="course">

<i class="fas fa-book"></i>

My Courses

</a>

<a href="#" class="star">

<i class="fas fa-star"></i>

Star Teacher

</a>

<a href="#" class="sub">

<i class="fas fa-subscript"></i>

Subscription

</a>

<a href="#" class="download">

<i class="fas fa-download"></i>

My Download

</a>

</nav>

<!--课程列表-->

<div class="course-list">

<!--标题栏-->

<div class="title">

<h2>Latest course</h2>

<a class="more" href="#">More+</a>

</div>

<!--列表容器-->

<div class="item-list">

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

</div>

</div>

<!--课程列表-->

<div class="course-list">

<!--标题栏-->

<div class="title">

<h2>Latest course</h2>

<a class="more" href="#">More+</a>

</div>

<!--列表容器-->

<div class="item-list">

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

</div>

</div>

<!--课程列表-->

<div class="course-list">

<!--标题栏-->

<div class="title">

<h2>Latest course</h2>

<a class="more" href="#">More+</a>

</div>

<!--列表容器-->

<div class="item-list">

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

<div class="item">

<!--图片-->

<div class="cover">

<img src="img/17/cover.png">

</div>

<!--课程名-->

<h3 class="course-title">Photography course</h3>

<!--用户信息-->

<div class="user-info">

<!--头像-->

<div class="avatar">

<img src="img/17/avatar.png">

</div>

<!--昵称-->

<div class="nickname">

Annabelle

</div>

</div>

</div>

</div>

</div>

</body>

</html>

\*{

margin: 0;

padding: 0;

}

@total-width:750;

.w{

//设置宽度

width: (693/40rem);

//水平居中

margin: 0 auto;

}

html{

//vw适配,设置rem的比值

font-size: (100vw/@total-width \* 40);

background-color: #eff0f4;

}

a{

//去除超链接下划线

text-decoration: none;

}

//头部header样式

.top-bar:extend(.w){

//设置弹性容器，使子元素水平排列

display: flex;

//设置头部高度

height: (175/40rem);

//设置行高使h1垂直居中

line-height: (175/40rem);

//background-color: aqua;

//空白均匀分布到子元素之间

justify-content: space-between;

//子元素垂直居中

align-items: center;

a{

color: #24253d;

font-size: (50/40rem);

i{

color: #999;

font-size: (40/40rem);

}

}

}

//轮播图样式

.banner:extend(.w){

img{

width: 100%;

}

}

//中间菜单栏

.menu:extend(.w){

height: (329/40rem);

//background-color: aqua;

//将菜单栏设置为弹性容器

display: flex;

//设置换行

flex-flow: row wrap;

//设置子元素主轴对齐方式

justify-content: space-between;

//设置子元素辅轴对齐方式,空白分布到元素的单侧

align-content: space-evenly;

//设置子菜单链接大小

a{

font-size: (30/40rem);

width: (327/40rem);

height: (104/40rem);

color: white;

//圆角

border-radius: (10/40rem);

//字体居中对齐

//text-align: center;

//字体垂直居中对齐

line-height: (104/40rem);

//为字体图标设置样式

i{

margin: 0 (20/40rem) 0 (32/40rem);

}

}

.course{

background-color: #f97053;

}

.star{

background-color: #cd6efe;

}

.sub{

background-color: #ff3971;

}

.download{

background-color: #1bc4fb;

}

}

//课程列表样式

.course-list:extend(.w){

//background-color: #1bc4fb;

height: (394/40rem);

//设置为弹性盒子

display: flex;

//设置自上向下垂直排列

flex-flow: column;

//空白沿主轴分布到元素的两侧

justify-content: space-between;

//设置底部外边距

margin-bottom: (46/40rem);

//标题栏样式

.title{

//开启弹性盒子

display: flex;

//子元素往主轴两边靠

justify-content: space-between;

//子元素辅轴居中

align-items: center;

//标题栏中h2的样式

h2{

font-size: (33/40rem);

color: #24253d;

//标题左边的竖线

border-left: 2px solid #3a84ff;

padding-left: 10px;

}

//more+的样式

a{

font-size: (28/40rem);

color: #656565;

}

}

}

//设置课程的列表项

.item-list{

//设置为弹性元素

display: flex;

//设置溢出内容的滚动条

overflow: auto;

//设置宽度

width: (720/40rem);

}

//每个课程

.item{

//溢出内容不希望缩减

flex: none;

//宽度和高度用来设置可见框的大小

box-sizing: border-box;

width: (320/40rem);

height: (324/40rem);

padding: 0 (22/40rem);

background-color: white;

//设置阴影

box-shadow: 0 0 10px rgba(0,0,0,.3);

//圆角

border-radius: 5px;

//设置为弹性盒子

display: flex;

//设置自上向下垂直排列

flex-flow: column;

//空白分布到元素的单侧

justify-content: space-evenly;

//每个item直接隔开一段距离

margin-right: (24/20rem);

//设置课程图片样式

img{

width: 100%;

}

//课程名

.course-title{

font-size: (26/40rem);

color: #24253d;

}

//设置用户信息的容器

.user-info{

//弹性盒子

display: flex;

//辅轴居中

align-items: center;

}

//头像

.avatar{

width: (42/40rem);

height: (42/40rem);

//img{

// width: 100%;

//}

}

//昵称

.nickname{

margin: 6px;

font-size: (22/40rem);

color: #969393;

}

}

# 响应式布局

·网页可以根据不通的设备或窗口大小呈现出不同的效果

·使用响应式布局，可以使一个网页适用于所有设备

·响应布局的关键就是 媒体查询

·通过媒体查询，可以为不同的设备，或设备不同状态来分别设置样式

## 媒体查询

·语法

- @media 查询规则{}

·媒体类型

- all 所有设备

- print 打印设备

- screen 带屏幕的设备

- speech 屏幕阅读器

/\*所有设备的样式\*/

@media all {

body{

background-color: #0086b3;

}

}

/\*打印机打印时的样式\*/

@media print {

body{

background-color: white;

}

}

/\*带屏幕的设备\*/

@media screen {

}

/\*屏幕阅读器\*/

@media speech {

}

·可以使用,连接多个媒体类型，这样它们之间就是一个或的关系

@media print,screen {

body {

background-color: #bfa;

}

}

·可以在媒体类型前添加一个only，表示只有。only的使用主要是为了兼容一些老版本浏览器

@media only screen {

body{

background-color: #bfa;

}

}

·媒体特性

- width 视口的宽度

- height 视口的高度

/\*在浏览器视口宽为500px时生效\*/

@media (width: 500px) {

body{

background-color: #0086b3;

}

}

- min-width 视口的最小宽度（视口大于指定宽度时生效）

/\*在浏览器视口宽大于500px时生效\*/

@media (min-width: 500px) {

body{

background-color: #0086b3;

}

}

- max-width 视口的最大宽度（视口小于指定宽度时生效）

/\*在浏览器视口宽小于500px时生效\*/

@media (max-width: 500px) {

body{

background-color: #0086b3;

}

}

·样式切换的分界点，我们称其为断点，也就是网页的样式会在这个点时发生变化一般比较常用的断点

- 小于768 超小屏幕 max-width=768px

- 大于768 小屏幕 min-width=768px

- 大于992 中型屏幕 min-width=992px

- 大于1200 大屏幕 min-width=1200px

·可以多个条件连接

- ,表示或

/\*视口大于500或者小于300时生效\*/

@media (min-width: 500px),(max-width:300px){

body{

background-color: #bfa;

}

}

- and 表示与

/\*只有是屏幕并且视口宽度大于500小于700时生效\*/

@media only screen and (min-width: 500px) and (max-width:700px){

body{

background-color: #bfa;

}

}

## 美图手机网站响应式布局

·响应式设计的网页

- 移动端优先

- 渐进增强

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<link rel="stylesheet" href="./css/reset.css">

<link rel="stylesheet" href="./fa/css/all.min.css">

<link rel="stylesheet" href="./css/meitu.css">

</head>

<body>

<div class="top-bar-wrapper">

<!--顶部工具条-->

<div class="top-bar">

<!--左侧菜单按钮-->

<div class="left-menu">

<!--菜单图标和内容-->

<ul class="menu-icon">

<!--导航图标按钮的三条线-->

<li></li>

<li></li>

<li></li>

</ul>

<!--创建菜单内容-->

<ul class="nav">

<li><a href="#">手机</a> </li>

<li><a href="#">美容仪器</a> </li>

<li><a href="#">配件</a> </li>

<li><a href="#">服务支持</a> </li>

<li><a href="#">企业网站</a> </li>

<li>

<a href="#">

<i class="fas fa-search"></i>

</a>

<span>搜索Meitu.com</span>

</li>

</ul>

</div>

<!--中间logo-->

<h1 class="logo">

<a href="#">meitu美图</a>

</h1>

<!--用户信息图标-->

<div class="user-info">

<a href="#">

<i class="fas fa-user"></i>

</a>

</div>

</div>

</div>

</body>

</html>

//统一设置超链接样式

a{

//取消下划线

text-decoration: none;

color: white;

&:hover{

color: rgb(197,196,196);

}

}

.top-bar-wrapper{

background-color: black;

}

//顶部工具条

.top-bar{

//指定最大宽度

max-width: 1200px;

//居中显示

margin: 0 auto;

height: 48px;

background-color: #000;

padding: 0 14px;

//设为弹性盒

display: flex;

//辅轴居中

align-items: center;

//空白分配到元素之间

justify-content: space-between;

}

//左侧菜单按钮

.left-menu{

//菜单图标

.menu-icon{

width: 18px;

height: 48px;

//background-color: cadetblue;

position: relative;

//设置导航图标按钮的三条线

li{

width: 18px;

height: 1px;

background-color: #fff;

//设为绝对定位

position: absolute;

//修改变形圆点

transform-origin: left center;

//过渡时长

transition: 0.3s;

}

//第一个li

li:nth-child(1){

top: 18px;

}

//第二个li

li:nth-child(2){

top: 24px;

}

//第三个li

li:nth-child(3){

top: 30px;

}

//为父元素设置hover，正常项目需要通过js实现

&:hover{

//第一个li

li:nth-child(1){

//向下旋转

transform: rotateZ(41deg);

}

//第二个li

li:nth-child(2){

//隐藏

opacity: 0;

}

//第三个li

li:nth-child(3){

//向上旋转

transform: rotateZ(-41deg);

}

}

}

&:hover{

.nav{

//将隐藏的菜单内容显示

display: block;

}

}

//为父元素开启相对定位，不设置时，子元素的绝对定位相对于html

//position: relative;

//菜单内容

.nav{

//将菜单内容隐藏

display: none;

//开启绝对定位

position: absolute;

//放到顶部工具栏下边

top: 48px;

//设置背景颜色

background-color: #000;

//width自动变为auto，撑满整个hrml

left: 0;

bottom: 0;

right: 0;

padding-top: 60px;

li{

width: 80%;

margin: 0 auto;

border-bottom: 1px solid #4e4e4e;

a{

line-height: 44px;

font-size: 14px;

}

span{

color: #fff;

font-size: 14px;

}

}

}

}

.logo{

a{

//将文字隐藏，搜索引擎能看到，我们看不到

text-indent: -9999px;

display: block;

//为了适应手机，给的是二倍图，在这里我们要/2

width: (244/2px);

height: (64/2px);

background-image: url("../img/18/dff63979.sprites-index@2x.png");

//图片大小也要/2

background-size: 400px 400px;

}

}

//设置媒体查询

@media only screen {

//当视图宽度大于768时

@media(min-width: 768px){

//修改弹性元素的顺序

.left-menu{

order: 2;

//设置为自动增长

flex: auto;

//隐藏菜单图标

.menu-icon{

display: none;

}

//显示菜单栏

.nav{

//将菜单栏设置成弹性盒

display: flex;

//关掉定位

position: static;

//清除内边距

padding: 0;

justify-content: space-around;

//重新设置每个菜单的样式

li{

//清除宽度

width: auto;

//去掉下边框

border-bottom: none;

//清除外边距

margin: 0;

a{

line-height: 48px;

}

//将span隐藏

span{

display: none;

}

}

}

}

.logo{

order: 1;

}

.user-info{

order: 3;

}

}

}