# CSS 问题答案

下面所有的代码都是个人作答，可能存在理解错误或者答案错误的地方，欢迎大家在[项目下方](https://github.com/springHyc/InterviewLibrary)或者[项目的 issues](https://github.com/springHyc/InterviewLibrary/issues)中留言批评指正。

## 1. 下面给出的代码你觉得有什么问题吗？

<div class="container">  
 <div class="item left">left: width: 100px</div>  
 <div class="item center ">中间宽度自定义</div>  
 <div class="item right ">right width: 100px</div>  
</div>  
  
<style>  
 .container {  
 height: 100px;  
 border: 1px solid #000;  
 }  
 .item {  
 height: 100%;  
 border: 1px solid red;  
 }  
 .left {  
 float: left;  
 width: 100px;  
 }  
 .center {  
 margin: 0 100px;  
 width: auto;  
 }  
 .right {  
 float: right;  
 width: 100px;  
 }  
</style>

个人答案

1. 这样实现不了三栏布局，使用 float 布局的话，中间的 div 会占一行，右侧部分会另起一行展示
2. 将中间部分和右侧部分的 html 结构换一下，就可以实现三栏布局，也就是下面这样：

<div class="container">  
 <div class="item left">left: width: 100px</div>  
 <div class="item right ">right width: 100px</div>  
 <div class="item center ">中间宽度自定义</div>  
</div>

## 2. css 选择器有哪些？

在 CSS 中，选择器是一种模式，用于选择需要添加样式的元素。 "CSS" 列指示该属性是在哪个 CSS 版本中定义的。（CSS1、CSS2 还是 CSS3。）

|  |  |  |  |
| --- | --- | --- | --- |
| 选择器 | 例子 | 例子描述 | CSS |
| [.*class*](http://www.w3school.com.cn/cssref/selector_class.asp) | .intro | 选择 class="intro" 的所有元素。 | 1 |
| [#*id*](http://www.w3school.com.cn/cssref/selector_id.asp) | #firstname | 选择 id="firstname" 的所有元素。 | 1 |
| [\*](http://www.w3school.com.cn/cssref/selector_all.asp) | \* | 选择所有元素。 | 2 |

| [*element*](http://www.w3school.com.cn/cssref/selector_element.asp) | p | 选择所有

元素。 | 1 | | [*element*,*element*](http://www.w3school.com.cn/cssref/selector_element_comma.asp) | div,p | 选择所有

元素和所有

元素。 | 1 | | [*element* *element*](http://www.w3school.com.cn/cssref/selector_element_element.asp) | div p | 选择

元素内部的所有

元素。 | 1 | | [*element*>*element*](http://www.w3school.com.cn/cssref/selector_element_gt.asp) | div>p | 选择父元素为

元素的所有

元素。 | 2 | | [*element*+*element*](http://www.w3school.com.cn/cssref/selector_element_plus.asp) | div+p | 选择紧接在

元素之后的所有

元素。 | 2 | | [[*attribute*]](http://www.w3school.com.cn/cssref/selector_attribute.asp) | [target] | 选择带有 target 属性所有元素。 | 2 | | [[*attribute*=*value*]](http://www.w3school.com.cn/cssref/selector_attribute_value.asp) | [target=\_blank] | 选择 target="\_blank" 的所有元素。 | 2 | | [[*attribute*~=*value*]](http://www.w3school.com.cn/cssref/selector_attribute_value_contain.asp) | [title~=flower] | 选择 title 属性包含单词 "flower" 的所有元素。 | 2 | | [[*attribute*=*value*]](http://www.w3school.com.cn/cssref/selector_attribute_value_start.asp) | [lang|=en] | 选择 lang 属性值以 "en" 开头的所有元素。 | 2 | | [:link](http://www.w3school.com.cn/cssref/selector_link.asp) | a:link | 选择所有未被访问的链接。 | 1 | | [:visited](http://www.w3school.com.cn/cssref/selector_visited.asp) | a:visited | 选择所有已被访问的链接。 | 1 | | [:active](http://www.w3school.com.cn/cssref/selector_active.asp) | a:active | 选择活动链接。 | 1 | | [:hover](http://www.w3school.com.cn/cssref/selector_hover.asp) | a:hover | 选择鼠标指针位于其上的链接。 | 1 | | [:focus](http://www.w3school.com.cn/cssref/selector_focus.asp) | input:focus | 选择获得焦点的 input 元素。 | 2 | | [:first-letter](http://www.w3school.com.cn/cssref/selector_first-letter.asp) | p:first-letter | 选择每个

元素的首字母。 | 1 | | [:first-line](http://www.w3school.com.cn/cssref/selector_first-line.asp) | p:first-line | 选择每个

元素的首行。 | 1 | | [:first-child](http://www.w3school.com.cn/cssref/selector_first-child.asp) | p:first-child | 选择属于父元素的第一个子元素的每个

元素。 | 2 | | [:before](http://www.w3school.com.cn/cssref/selector_before.asp) | p:before | 在每个

元素的内容之前插入内容。 | 2 | | [:after](http://www.w3school.com.cn/cssref/selector_after.asp) | p:after | 在每个

元素的内容之后插入内容。 | 2 | | [:lang(*language*)](http://www.w3school.com.cn/cssref/selector_lang.asp) | p:lang(it) | 选择带有以 "it" 开头的 lang 属性值的每个

元素。 | 2 | | [*element1*~*element2*](http://www.w3school.com.cn/cssref/selector_gen_sibling.asp) | p~ul | 选择前面有

元素的每个

元素。 | 3 | | [[*attribute*^=*value*]](http://www.w3school.com.cn/cssref/selector_attr_begin.asp) | a[src^="https"] | 选择其 src 属性值以 "https" 开头的每个 元素。 | 3 | | [[*attribute*$=*value*]](http://www.w3school.com.cn/cssref/selector_attr_end.asp) | a[src$=".pdf"] | 选择其 src 属性以 ".pdf" 结尾的所有 元素。 | 3 | | [[*attribute\*\*=*value\*]](http://www.w3school.com.cn/cssref/selector_attr_contain.asp) | a[src\*="abc"] | 选择其 src 属性中包含 "abc" 子串的每个 元素。 | 3 | | [:first-of-type](http://www.w3school.com.cn/cssref/selector_first-of-type.asp) | p:first-of-type | 选择属于其父元素的首个

元素的每个

元素。 | 3 | | [:last-of-type](http://www.w3school.com.cn/cssref/selector_last-of-type.asp) | p:last-of-type | 选择属于其父元素的最后

元素的每个

元素。 | 3 | | [:only-of-type](http://www.w3school.com.cn/cssref/selector_only-of-type.asp) | p:only-of-type | 选择属于其父元素唯一的

元素的每个

元素。 | 3 | | [:only-child](http://www.w3school.com.cn/cssref/selector_only-child.asp) | p:only-child | 选择属于其父元素的唯一子元素的每个

元素。 | 3 | | [:nth-child(*n*)](http://www.w3school.com.cn/cssref/selector_nth-child.asp) | p:nth-child(2) | 选择属于其父元素的第二个子元素的每个

元素。 | 3 | | [:nth-last-child(*n*)](http://www.w3school.com.cn/cssref/selector_nth-last-child.asp) | p:nth-last-child(2) | 同上，从最后一个子元素开始计数。 | 3 | | [:nth-of-type(*n*)](http://www.w3school.com.cn/cssref/selector_nth-of-type.asp) | p:nth-of-type(2) | 选择属于其父元素第二个

元素的每个

元素。 | 3 | | [:nth-last-of-type(*n*)](http://www.w3school.com.cn/cssref/selector_nth-last-of-type.asp) | p:nth-last-of-type(2) | 同上，但是从最后一个子元素开始计数。 | 3 | | [:last-child](http://www.w3school.com.cn/cssref/selector_last-child.asp) | p:last-child | 选择属于其父元素最后一个子元素每个

元素。 | 3 | | [:root](http://www.w3school.com.cn/cssref/selector_root.asp) | :root | 选择文档的根元素。 | 3 | | [:empty](http://www.w3school.com.cn/cssref/selector_empty.asp) | p:empty | 选择没有子元素的每个

元素（包括文本节点）。 | 3 | | [:target](http://www.w3school.com.cn/cssref/selector_target.asp) | #news:target | 选择当前活动的 #news 元素。 | 3 | | [:enabled](http://www.w3school.com.cn/cssref/selector_enabled.asp) | input:enabled | 选择每个启用的 元素。 | 3 | | [:disabled](http://www.w3school.com.cn/cssref/selector_disabled.asp) | input:disabled | 选择每个禁用的 元素 | 3 | | [:checked](http://www.w3school.com.cn/cssref/selector_checked.asp) | input:checked | 选择每个被选中的 元素。 | 3 | | [:not(*selector*)](http://www.w3school.com.cn/cssref/selector_not.asp) | :not(p) | 选择非

元素的每个元素。 | 3 | | [::selection](http://www.w3school.com.cn/cssref/selector_selection.asp) | ::selection | 选择被用户选取的元素部分。 | 3 |

## 3. 描述一下 CSS 的优先级规则

* 内联样式>id 选择器> 伪类（：nth-child()）> 属性选择器> 类选择器> 元素选择器（div）> 通用选择器（\*)
* !important 规则例外，该样式声明会覆盖 CSS 中任何其他的声明。

## 4. css 盒模型是什么？

盒模型的组成：由里向外是：content、padding、border、margin。

盒模型有两种标准：一个 w3c 标准模型，一个是 IE 模型。

标准模型 ie模型

从上面两图不难看出在标准模型中，盒模型的宽高只是内容（content）的宽高，

而在 IE 模型中盒模型的宽高是内容(content)+填充(padding)+边框(border)的总宽高。

## 5. 实现 div 的动画移动

使用 css3 的动画 animation 和@keyframes 来实现 div 的移动。

<html>  
<head>  
 <style>  
 div {  
 width: 100px;  
 height: 100px;  
 border: 1px solid red;  
 position: relative;  
 animation: div 5s infinite alternate;  
 }  
  
 @keyframes div {  
 form {left: 0; background-color:'#fff'}  
 to {left: 200px; background-color: red}  
 }  
 </style>  
</head>  
  
<body>  
 <div></div>  
</body>  
</html

## 6. css 模块化是什么？

css 模块化就是所有的类名都只有局部作用域的 css 文件。

好处 css 模块化将作用域限制于组件中，从而避免了全局作用域的问题，编译过程还能帮你完成命名。

css 模块化的解决方案 目前解决方案有两种: 第一，彻底抛弃 css,使用 js 或 json 来写样式，例如：[Radium](https://github.com/FormidableLabs/radium)，[jsxstyle](https://github.com/smyte/jsxstyle)，[react-style](https://github.com/js-next/react-style) 属于这一类。 第二, 依旧使用旧的 css,使用 js 来管理样式依赖，代表是[css-modules](https://github.com/css-modules/css-modules)

## 7. 如何实现一个三列布局，中间固定，两边自适应？

<header>  
 <style>  
 .contaniner {  
 display: flex;  
 flex-direction: row;  
 justify-content: space-between;  
 }  
  
 .item {  
 border: 1px solid red;  
 height: 100%;  
 }  
  
 .center {  
 width: 100px;  
 }  
  
 .left,  
 .right {  
 width: calc(50% - 50px);  
 }  
 </style>  
</header>  
<div class="contaniner">  
 <div class="item left">  
 left  
 </div>  
 <div class="item center">  
 中间  
 </div>  
 <div class="item right">  
 右边  
 </div>  
</div>

相对应的问题来了，如何实现一个三栏布局，中间自适应，两边固定？

只需要把中间和两边的宽度调整一下即可。

.center {  
 width: calc(100% - 200px);  
  
}  
  
 .left,  
 .right {  
 width: 100px;  
 }

同样可以完成中间自适应，两边固定的布局：

<body class="container">  
 <div class="item left">100px宽</div>  
 <div class="item center">自适应</div>  
 <div class="item right">100px宽</div>  
</body>  
  
<style>  
 .container {  
 border: 1px solid black;  
 width: 100vw;  
 height: 100vh;  
  
 display: flex;  
 flex-direction: row;  
 justify-content: space-between;  
 }  
  
 .item {  
 border: 1px solid red;  
 height: 100%;  
 }  
  
 .center {  
 flex: 1;  
 }  
  
 .left,  
 .right {  
 width: 100px;  
 }  
</style>

在上一段代码的基础上更改一下即可完成中间固定，两边自适应

.center {  
 width: 100px;  
}  
  
.left,  
.right {  
 flex: 1;  
}

## 8. 如何完成下面的样式？

<html>  
 <head>  
 <meta charset="utf-8" />  
 <title>test.html</title>  
 <style>  
 .container {  
 width: 100vw;  
 height: 100vh;  
  
 display: flex;  
 flex-direction: row;  
 justify-content: center;  
 align-items: center;  
 }  
  
 .item {  
 width: 200px;  
 height: 200px;  
 border-radius: 100px;  
 box-shadow: 0 0 0 50px rgb(124, 122, 122);  
 background-color: rgb(70, 64, 54);  
 }  
  
 .item2 {  
 width: 50px;  
 height: 50px;  
 border-radius: 25px;  
 box-shadow: 0 0 0 25px rgb(31, 25, 25);  
 position: relative;  
 top: 75px;  
 left: 75px;  
 background-color: black;  
 }  
 </style>  
 </head>  
  
 <body class="container">  
 <div class="item">  
 <div class="item2"></div>  
 </div>  
 </body>  
</html>

## 8. 通过 html img 标签设置图片和通过 div 背景图设置图片，两种设置图片的方式有什么优劣？

* 占位符 <img> 标签定义 HTML 页面中的图像。从技术上讲，图片并不会插入 HTML 页面中，而是链接到 HTML 页面上。img 标签的作用是为被引用的图像创建占位符。

background-image 作为背景，在图片没有加载的时候或者加载失败的时候，不会有图片的占位标记，不会出现红叉。

* 加载时间 img是写在 HTML 里的是以 HTML 插入 img 标签的形式存在，CSS 图片背景是等结构加载完成后再去加载的。
* 是否为内容 非内容的图片写在 css 里，内容的图片就写在 HTML 里。

## 8. 实现一下一个 div 的居中。如果不适用 flex 布局怎么做？

以前整理的文件中有

## 9. 实现一下一个 div 的居中。如果不适用 flex 布局怎么做？

<header>  
 <style>  
 .contaniner {  
 margin: 0 auto; // 左右居中  
 border: 1px solid red;  
 width: 100px;  
 height: 100px;  
 top: calc(50% - 50px);  
 position: relative;  
 }  
 </style>  
</header>  
<div class="contaniner" />

必须有 width,margin: 0 auto才有效果； top: calc(50% - 50px); position: relative;是可以上下居中

## 10. 普通的实现一下两个<div>横着排列？

<header>  
 <style>  
 .contaniner {  
 height: 100%;  
 width: 100%;  
 }  
  
 .item {  
 border: 1px solid red;  
 width: 100px;  
 height: 100px;  
 float: left;  
 }  
 </style>  
</header>  
  
<div class="contaniner">  
 <div class="item"></div>  
 <div class="item"></div>  
</div>

## 11. 请写出你所知道的清除浮动的方法（代码）

* clear :both
* overflow:auto

## 12. 如何实现一个两栏 布局，左边是 100px 宽，右边自定义？（flex 和非 flex 各实现一个）

### flex 布局

<head>  
 <style>  
 .container {  
 border: 1px solid black;  
 width: 100vw;  
 height: 100vh;  
  
 display: flex;  
 flex-direction: row;  
 justify-content: space-between;  
 }  
  
 .item {  
 border: 1px solid red;  
 height: 100%;  
 }  
  
 .left {  
 width: 100px;  
 }  
  
 .right {  
 flex: 1;  
 }  
 </style>  
</head>  
  
<body class="container">  
 <div class="item left">100px宽</div>  
 <div class="item right">自适应</div>  
</body>

非 flex 布局方式

<html>  
 <head>  
 <meta charset="utf-8" />  
 <title>test.html</title>  
 <style>  
 .container {  
 border: 1px solid black;  
 width: 100vw;  
 height: 100vh;  
 }  
  
 .item {  
 border: 1px solid red;  
 height: 100%;  
 }  
  
 .left {  
 float: left;  
 width: 100px;  
 }  
  
 .right {  
 margin: 0 0 0 100px;  
 }  
 </style>  
 </head>  
  
 <body class="container">  
 <div class="item left">100px宽</div>  
 <div class="item right">自适应</div>  
 </body>  
</html>

## 13. 有一个日历组件，如何实现各个屏幕的自适应？

其实这道题目考察的也就是屏幕自适应如何实现。在以前我只知道 flex 布局以及@media 媒体查询，其实还有很多种不同的实现方式。

* 流式布局
* 固定宽度做法
* 响应式做法
* 设置 viewport 进行缩放
* rem 能等比例适配所有屏幕

### 详细介绍

* 流式布局

在页面中的布局是：宽度适用百分比，高度使用 px 来固定。

缺点：在大屏幕的手机下显示效果会变成有些页面元素宽度被拉的很长，但是高度还是和原来一样，实际显示非常的不协调，往往只有几个尺寸的手机下看到的效果是令人满意的。

* 固定宽度做法

还有一种是固定页面宽度的做法，早期有些网站把页面设置成 320 的宽度，超出部分留白，这样做视觉，前端都挺开心，视觉在也不用被流式布局限制自己的设计灵感了，前端也不用在搞坑爹的流式布局。

缺点：在大屏幕手机下两边是留白的，还有一个就是大屏幕手机下看起来页面会特别小，操作的按钮也很小，手机淘宝首页起初是这么做的。

* 响应式做法

分别为不同的屏幕分辨率定义布局，同时，在每个布局中，应用流式布局的理念，即页面元素宽度随着窗口调整而自动适配。

媒体查询+流式布局。通常使用 @media 媒体查询 和网格系统 (Grid System) 配合相对布局单位进行布局，实际上就是综合响应式、流动等上述技术通过 CSS 给单一网页不同设备返回不同样式的技术统称。

缺点：1. 媒体查询是有限的，也就是可以枚举出来的，只能适应主流的宽高;2. 要匹配足够多的屏幕大小，工作量不小，设计也需要多个版本。

* 设置 viewport 进行缩放

以 320 宽度为基准，进行缩放，最大缩放为 320\*1.3 = 416。

缺点：使用过程中反应缩放会导致有些页面元素会糊的情况。

* rem 能等比例适配所有屏幕

上面说过 rem 是通过根元素进行适配的，网页中的根元素指的是 html 我们通过设置 html 的字体大小就可以控制 rem 的大小。

根元素的大小为 1rem。

## 14. css3 有代表性的新特性？

* css3 边框（Borders）

用 CSS3，你可以创建圆角边框，添加阴影框

* css3 背景

CSS3 中包含几个新的背景属性，提供更大背景元素控制，可以实现多背景。

* css3 渐变
* 线性渐变
* 径向渐变
* css3 文本效果
* text-justify（ 规定当 text-align 设置为 "justify" 时所使用的对齐方法。）
* text-outline（规定文本的轮廓。）
* text-overflow（规定当文本溢出包含元素时发生的事情。）
* text-shadow（向文本添加阴影。）
* text-wrap（规定文本的换行规则。）
* CSS3 字体

使用 CSS3，网页设计师可以使用他/她喜欢的任何字体。

@font-face

* CSS3 转换和变形
* 2D 新转换属性
* 3D 转换属性
* CSS3 过渡

CSS3 中，我们为了添加某种效果可以从一种样式转变到另一个的时候，无需使用 Flash 动画或 JavaScript。

* CSS3 动画

要创建 CSS3 动画，你需要了解@keyframes 规则。@keyframes 规则是创建动画。 @keyframes 规则内指定一个 CSS 样式和动画将逐步从目前的样式更改为新的样式。

* CSS3 伸缩布局盒模型(弹性盒) flexbox
* CSS3 多媒体查询@media

## 15. 如何实现动画效果？

## 16. 怎么确定一个块的位置，涉及到 css 的盒模型，以及 DOM api

## 17. 伪元素和伪类的区别？说一个常用的伪元素？说一个常用的伪类？

### 简介

伪元素为 DOM 树没有定义的虚拟元素。不同于其他选择器，它不以元素为最小选择单元，它选择的是元素指定内容。比如::before 表示选择元素内容的之前内容，也就是""；::selection 表示选择元素被选中的内容。 伪元素的效果可以通过添加实际的元素来实现。

伪类用于选择 DOM 树之外的信息，或是不能用简单选择器进行表示的信息。前者包含那些匹配指定状态的元素，比如:visited，:active；后者包含那些满足一定逻辑条件的 DOM 树中的元素，比如:first-child，:first-of-type，：target。

伪类的效果可以通过添加实际的类来实现。

### 伪元素和伪类的区别？

* 伪元素修改为以::开头；伪类以:开头
* 伪类与伪元素的本质区别就是是否抽象创造了新元素
* 伪元素在一个选择器中只能出现一次，并且只能出现在末尾

### 相同

* 伪类与伪元素都是用于向选择器加特殊效果
* 伪类与伪元素优先级分别与类、标签优先级相同

### 详细介绍

* 什么是伪类

<p>  
 <em>This</em>  
 <em>is a text</em>  
</p>

如果我们想要第一个 em 标签字体颜色变红怎么做呢?

使用我们熟悉的伪类很简单

em:first-child {  
 color: red;  
}

表示选择的元素既要是 em 标签，同时要是其父元素的第一个子元素，不要错误认为是表示 em 元素的第一个子元素

但是如果不存在伪类我们怎么做呢

这是我们就需要为第一个 em 标签添加类

<p>  
 <em class="first-child">This</em>  
 <em>is a text</em>  
</p>  
  
em.first-child { color: red; }

* 什么是伪元素

现在我想让这个段落的第一个字母变红 怎么做呢 这回我们需要使用伪元素

p::first-letter {  
 color: red;  
}

同样假设伪元素不存在的情况 这时我们只能嵌套 span 标签来实现

<p>  
 <em><span>T</span>his</em>  
 <em>is a text</em>  
</p>

p span {  
 color: red;  
}

### 常用

伪元素

|  |  |  |
| --- | --- | --- |
| Selector | Meaning | CSS |
| ::first-letter | 选择指定元素的第一个单词 | 1 |
| ::first-line | 选择指定元素的第一行 | 1 |
| ::after | 在指定元素的内容前面插入内容 | 2 |
| ::before | 在指定元素的内容后面插入内容 | 2 |
| ::selection | 选择指定元素中被用户选中的内容 | 3 |

伪类

|  |  |  |
| --- | --- | --- |
| Selector | Meaning | CSS |
| :active | 选择正在被激活的元素 | 1 |
| :hover | 选择被鼠标悬浮着元素 | 1 |
| :link | 选择未被访问的元素 | 1 |
| :visited | 选择已被访问的元素 | 1 |
| :first-child | 选择满足是其父元素的第一个子元素的元素 | 2 |
| :lang | 选择带有指定 lang 属性的元素 | 2 |
| :focus | 选择拥有键盘输入焦点的元素 | 2 |
| :enable | 选择每个已启动的元素 | 3 |
| :disable | 选择每个已禁止的元素 | 3 |
| :checked | 选择每个被选中的元素 | 3 |
| :target | 选择当前的锚点元素 | 3 |
| :first-of-type | 选择满足是其父元素的第一个某类型子元素的元素 | 3 |
| :last-of-type | 选择满足是其父元素的最后一个某类型子元素的元素 | 3 |
| :only-of-type | 选择满足是其父元素的唯一一个某类型子元素的元素 | 3 |
| :nth-of-type(n) | 选择满足是其父元素的第 n 个某类型子元素的元素 | 3 |
| :nth-last-of-type(n) | 选择满足是其父元素的倒数第 n 个某类型的元素 | 3 |
| :only-child | 选择满足是其父元素的唯一一个子元素的元素 | 3 |
| :last-child | 选择满足是其父元素的最后一个元素的元素 | 3 |
| :nth-child(n) | 选择满足是其父元素的第 n 个子元素的元素 | 3 |
| :nth-last-child(n) | 选择满足是其父元素的倒数第 n 个子元素的元素 | 3 |
| :empty | 选择满足没有子元素的元素 | 3 |
| :in-range | 选择满足值在指定范围内的元素 | 3 |
| :out-of-range | 选择值不在指定范围内的元素 | 3 |
| :invalid | 选择满足值为无效值的元素 | 3 |
| :valid | 选择满足值为有效值的元素 | 3 |
| :not(selector) | 选择不满足 selector 的元素 | 3 |
| :optional | 选择为可选项的表单元素，即没有“required”属性 | 3 |
| :read-only | 选择有"readonly"的表单元素 | 3 |
| :read-write | 选择没有"readonly"的表单元素 | 3 |
| :root | 选择根元素 | 3 |

## 18. :before和:after的区别？都用来做什么？

::before 和::fater 都是伪元素。

是在元素前后插入生成内容。如果是插入一张图片，插入一段文字，或者做一个小三角等等

## 19. css3 的滤镜（filter）听过吗？

|  |  |
| --- | --- |
| Filter | 描述 |
| none | 没有效果 |
| blur(px) | 给图像设置高斯模糊。 |
| brightness(%) | 给图片应用一种线性乘法，使其看起来更亮或更暗。如果值是 0%，图像会全黑。值是 100%，则图像无变化。其他的值对应线性乘数效果。值超过 100%也是可以的，图像会比原来更亮。如果没有设定值，默认是 1。 |
| contrast(%) | 调整图像的对比度。值是 0%的话，图像会全黑。值是 100%，图像不变。值可以超过 100%，意味着会运用更低的对比。若没有设置值，默认是 1。 |
| drop-shadow(h-shadow v-shadow blur spread color) | 给图像设置一个阴影效果。阴影是合成在图像下面，可以有模糊度的，可以以特定颜色画出的遮罩图的偏移版本。 |
| grayscale(%) | 将图像转换为灰度图像。值定义转换的比例。值为 100%则完全转为灰度图像，值为 0%图像无变化。值在 0%到 100%之间，则是效果的线性乘子。若未设置，值默认是 0； |
| opacity(%) | 转化图像的透明程度。 |

## 20. 有个场景：国家发生灾难后，需要将该页面做成灰白色，如何实现？

## 21. css 中@import 和<link rel="stylesheet" href="xxx.css">有什么区别？

### 共同点

* link 和@import 都是可以用来引入 css 样式的方法。

### 不同点

* 应用方式不同
* link(外部引用):
* @import(导入式):@import url(xxx.css);
* 放置的位置不同
* link 一般放在 head 标签中
* @import 必须放在<style type="text/css">标签中
* <style type="text/css">  
   @import "jisuan.css";  
  </style>
* 加载方式不同
* ink 会和 dom 结构一同加载渲染
* @import 只能能 dom 结构加载完成以后才能渲染页面
* 当使用 Javascript 控制 DOM 去改变样式的时候，只能使用 link 方式，因为 @import 只有 CSS ，不是 DOM 可以控制的
* 兼容性不同
* link 是在 xhtml 的标签，兼容 IE 各个版本
* @import 是 css2.1 时提出来的，只能在 IE6 以上进行解析。

## 22. 实现一个 img 图片旋转的方法？

* transform: rotate(30deg);
* CSS3 动画 @keyframes

百度-17 年上半年

## 23.固定区域内一个图片实现上下垂直居中，图片高度不固定，但是图片高度小于固定区域的高度。

* flex 布局解决方法

<div class="container">  
 <img src="http://img4.imgtn.bdimg.com/it/u=2075750630,4216747848&fm=23&gp=0.jpg" />  
</div>

.container {  
 width: 300px;  
 height: 300px;  
 border: 1px solid rebeccapurple;  
 margin: 10px;  
 display: flex;  
 flex-direction: row;  
 align-items: center;  
 justify-content: center;  
}



flex.png

这个方法的不好处就在于，使用了flex，CSS3 中的新增属性，各浏览器的支持程度不一样。

flex支持程度

flex支持程度

* 设置为单元格的方式

<div class="container">  
 <img src="http://img4.imgtn.bdimg.com/it/u=2075750630,4216747848&fm=23&gp=0.jpg" />  
</div>

.container {  
 width: 300px;  
 height: 300px;  
 border: 1px solid rebeccapurple;  
 display: table-cell;  
 text-align: center;  
 vertical-align: middle;  
}

cell.png

cell.png

这种方式除了 IE67 之外，其他主流浏览器中基本上实现了图片的垂直居中对齐。

* 利用定位来实现垂直居中 - 兼容 IE5.50 firefox chrome 在支持 display 为 table-cell 的浏览器中依然用 vertical-align:middle 来实现，在 IE5.567 中利用定位来实现垂直居中。

<div class="container">  
 <p>  
 <img src="http://img4.imgtn.bdimg.com/it/u=2075750630,4216747848&fm=23&gp=0.jpg" />  
 </p>  
</div>

.container {  
 border: 1px dashed #ccc;  
 height: 300px;  
 width: 300px;  
 overflow: hidden;  
 display: table-cell;  
 vertical-align: middle;  
 \*position: relative;  
}  
  
p {  
 \*position: absolute;  
 \*top: 50%;  
 width: 100%;  
 text-align: center;  
}  
img {  
 \*position: relative;  
 \*top: -50%;  
}

\*： 一般\*的属性作用于:IE6、IE7

* transform 的方式

.container {  
 border: 1px solid;  
 width: 400px;  
 height: 400px;  
 position: absolute;  
}  
  
img {  
 left: 50%;  
 top: 50%;  
 position: absolute;  
 transform: translate(-50%, -50%);  
}

### 错误做法：

* 1. div : position: relative; img : position: absolute;这种方式只能实现左右居中，不能上下居中。

.container {  
 position: relative;  
}  
  
// 只能实现左右居中，上下不行  
img {  
 position: absolute;  
 margin: auto;  
 left: 0;  
 right: 0;  
 top: 0;  
 bottom: 0;  
}

## 24. 如何实现下图的布局？

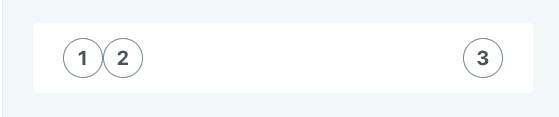


image.png

* 采用 flex 布局

<div>  
 <item>1</item>  
 <item>2</item>  
 <div>  
 <item>3</item>  
 </div>  
</div>

如果是这面这种 html 结构，该怎办？

<div>  
 <item>1</item>  
 <item>2</item>  
 <item>3</item>  
</div>

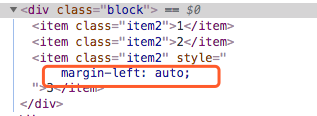


image.png

知识点： 