CSS总结

CSS总结

由于最近在准备前端方面的面试，所以对自己平常工作中用到的地方做出一些总结。有许多地方叙述的并不是十分详细，只是大致的描述一下，给自己提供一个知识轮廓。本篇中主要描述了CSS中的基础部分，以及一些CSS3的新特性。下篇将会继续介绍CSS3动画部分以及移动端部分。

CSS渲染的过程就好像在一张白纸上画画一样，你需要清楚的知道在什么位置画上那些内容(定位过程)，这些内容的大小(盒模型)是多少以及该内容的颜色(color)，背景(background属性)，文字(text属性)等属性。

盒模型

文档中的每个元素被描绘为矩形盒子。渲染引擎的目的就是判定大小，属性——比如它的颜色、背景、边框方面——及这些盒子的位置。在CSS中，这些矩形盒子用**标准盒模型**来描述。这个模型描述了一个元素所占用的空间。每一个盒子有四条边界：外边距边界margin, 边框边界border, 内边距边界padding与内容边界content。

IE盒模型和W3C盒模型在计算总宽度存在一些差异:

- 在W3C模型中: 总宽度 = margin-left + border-left + padding-left + width + padding-right + border-right + margin-right
- 在IE模型中: 总宽度 = margin-left + width(包括除外边距的所有部分) + margin-right

在CSS3中引入了box-sizing属性, 它可以允许改变默认的CSS盒模型对元素宽高的计算方式. 共包括两个选项：

- content-box：标准盒模型，CSS定义的宽高只包含content的宽高
- border-box：IE盒模型，CSS定义的宽高包括了content，padding和border

定位机制

css有三种基本定位机制: 标准文档流(Normal flow), 浮动(Floats)和绝对定位(Absolute positioning)。

- 标准文档流
    - 从左到右，从上向下，输出文档内容
    - 由块级元素(从左到右撑满页面，独占一行，触碰到页面边缘时自动换行的元素, 如div, ul, li, dl, dt, p)和行级元素组成(能在同一行内显示并且不会改变HTML文档结构，如span, input)
- 浮动
    - 设置为浮动的元素将会往左(float:left)或者往右(float:right)漂移, 直到遇到阻挡 - 其他浮动元素或者父元素的边框。浮动元素不在标准文档流中

占据空间,但会对其后的浮动元素造成影响。

- 绝对定位
  - 设置为绝对定位的元素(posistion:absolute)将从标准文档流中删除，其所占据的标准流空间也不存在。然后通过top,left,right,bottom属性对其相对父元素进行定位。

网页布局

- 流动布局
  - 需了解实现块居中常见的几种方式
- 浮动布局
  - 当设置float:left或者float:right时, 元素会左移或右移直到触碰到容器位置，仍然处于标准文档流中。当元素没有设置宽度值，而设置了浮动属性，元素的宽度随内容的变化而变化。当元素设置为浮动属性后，会对紧邻之后的元素造成影响，紧邻之后的元素会紧挨着该元素显示。当父元素包含块缩成一条时，用clear:both方法无效，它一般用于紧邻后面的元素的清除浮动，要用overflow属性。清除浮动的方法有两种:使用clear属性 - clear: both; 同时设置width:100%(或固定宽度) + overflow:hidden。
  - 浮动布局可实现横向多列布局
- 绝对定位布局
  - position：static, relative, absolute, fixed
    - static是默认值
    - 相对定位relative
      - 相对于自身原有位置进行偏移
      - 仍处于标准文档流中
      - 随即拥有偏移属性和z-index属性
    - 固定定位fixed
      - 一个固定定位（position属性的值为fixed）元

素会相对于视窗来定位,这意味着即便页面滚动，它还是会停留在相同的位置。一个固定定位元素不会保留它原本在页面应有的空隙。

- 绝对定位absolute
  - 相对于最近的已定位的祖先元素, 有已定位(指position不是static的元素)祖先元素, 以最近的祖先元素为参考标准。如果无已定位祖先元素, 以body元素为偏移参照基准, 并且它会随着页面滚动而移动。
  - 完全脱离了标准文档流。
  - 随即拥有偏移属性和z-index属性。

- 实现横向两列布局
  - 常用于一列固定宽度，另一列宽度自适应的情况
  - relative - 父元素相对定位
  - absolute - 自适应宽度元素定位

- 能够实现横向多列布局

常见布局实现

常见的布局有以下几种: 单列水平居中布局, 一列定宽一列自适应布局, 两列定宽一列自适应布局, 两侧定宽中间自适应三列布局。

重点介绍一下常见的三列布局之: 圣杯布局和双飞翼布局。

圣杯布局和双飞翼布局

两者都属于三列布局，是一种很常见的页面布局方式，三列一般分别是子列sub、主列main和附加列extra，其中子列一般是居左的导航，且宽度固定；主列是居中的主要内容，宽度自适应；附加列一般是广告等额外信息，居右且宽度固定。

圣杯布局

```
<div class="container">
  <div class="main"></div>
  <div class="sub"></div>
  <div class="extra"></div>
```

```
</div>
body {
    min-width: 600px; /* 2*sub + extra */
}
.container {
    padding-left: 210px;
    padding-right: 190px;
}
.main {
    float: left;
    width: 100%;
    height: 300px;
    background-color: rgba(255, 0, 0, .5);
}
.sub {
    position: relative;
    left: -210px;
    float: left;
    width: 200px;
    height: 300px;
    margin-left: -100%;
    background-color: rgba(0, 255, 0, .5);
}
.extra {
    position: relative;
    right: -190px;
    float: left;
    width: 180px;
    height: 300px;
    margin-left: -180px;
    background-color: rgba(0, 0, 255, .5);
}
```

双飞翼布局(淘宝使用的布局方式)

```
<div class="main-wrapper">
    <div class="main"></div>
</div>
<div class="sub"></div>
<div class="extra"></div>
.main-wrapper {
    float: left;
    width: 100%;
}
.main {
    height: 300px;
    margin-left: 210px;
    margin-right: 190px;
    background-color: rgba(255, 0, 0, .5);
}
.sub {
    float: left;
```

```
    width: 200px;
    height: 300px;
    margin-left: -100%;
    background-color: rgba(0, 255, 0, .5);
}
.extra {
    float: left;
    width: 180px;
    height: 300px;
    margin-left: -180px;
    background-color: rgba(0, 0, 255, .5);
}
```

总结:

- 俩种布局方式都是把主列放在文档流最前面，使主列优先加载。
- 两种布局方式在实现上也有相同之处，都是让三列浮动，然后通过负外边距形成三列布局。
- 两种布局方式的不同之处在于如何处理中间主列的位置：圣杯布局是利用父容器的左、右内边距定位；双飞翼布局是把主列嵌套在div后利用主列的左、右外边距定位。

TODO:

- margin为负值时的使用

Flex布局

Flexbox又叫弹性盒模型。它可以简单使用一个元素居中（包括水平垂直居中），可以让扩大和收缩元素来填充容器的可利用空间，可以改变源码顺序独立布局，以及还有其他的一些功能。Flex布局是我最喜欢的布局，合理使用它能够大大减少布局方面的工作, 例如以上列举的三列布局也可以使用flex轻松实现。此外在移动端使用flex也比较常见。

使用请参考:

- CSS flex完全指南
- flex历险记

响应式布局(Responsive Web Design)

响应式布局是指，网页可以自动识别设备屏幕宽度，根据不同的宽度采用不同的CSS的样式，从而达到兼容各种设备的效果。响应式布局使用媒体查询(CSS3 Media Queries), 根据不同屏幕分辨率采用不同CSS规则, 使用方式如下:

```
@media screen and (max-width:1024px) {
 /* 视窗宽度小于1024px时 */
  ....
}
```

Bootstrap grid系统的实现

Bootstrap是很受欢迎的HTML, CSS和JS框架, 用于开发响应式布局和移动设备优先的Web项目。它提供了一套响应式，移动优先的流式栅格系统(grid system)，将屏幕分成12列来实现响应式的。它的实现原理非常简单，Media Query加上float布局，如果想了解实现细节，请参考我另外一篇博客Boostrap网格系统。

CSS reset
CSS reset的目的是为了将不同浏览器自带样式重置，达到保持浏览器一致性的目的;reset.css通常是样式设计开始之前第一个引用的CSS文件。
TODO:

- Normalize.css

CSS hacks
不同的浏览器对CSS的解析结果是不同的，因此会导致相同的CSS输出的页面效果不同，这就需要CSS Hack来解决浏览器局部的兼容性问题。而这个针对不同的浏览器写不同的CSS 代码的过程，就叫CSS Hack。CSS Hack常见的有三种形式：CSS属性Hack、CSS选择符Hack以及IE条件注释Hack， Hack主要针对IE浏览器。
TODO:

- 常用css hacks技巧

CSS sprite
把网页中的一些零星背景图片整合到一张图片当中，再利用CSS背景图片定位属性定位到要显示的位置，因此也叫图片拼合技术。
好处：减少文件体积和服务器请求次数，从而提高开发效率。一般情况下保存为PNG-24，可以设计出丰富多彩的图标。
难处：你需要预先确定每个图标的大小。注意小图标与小图标之间的距离。
实现方式: background-image + background-position。
TODO:

- 如何制作CSS sprite图

iconfont字体
iconfont是指使用字体文件取代图片文件，来展示图标和一些特殊字体等元素。它使用CSS3中的@font-face属性，它允许加载自定义字体样式，而且它还能够加载服务器端的字体文件，让客户端显示客户端所没有安装的字体。它有很多优势: 首先它的体积要比图片小得多;特定的属性(颜色,大小,透明)等修改起来就像是操作字体一样简单;iconfont具有矢量性, 放大缩小不会失真;使用步骤:

- 设计师设计出Icon矢量图,需要保存为多种格式(可以使用一些online webfont工具完成)。
    - EOT(Embedded OpenType Fonts) IE专用格式
    - WOFF(The Web Open Font Format) Web字体最佳格式, 它是一个开放的TrueType/OpenType的压缩版本。09年被开发, 如今是W3C阻止的推荐标准。
    - TTF(TrueType Fonts) MacOS和WIN操作系统中最常见格式。
    - SVG(SVG Fonts) 用于SVG字体渲染的一种格式, W3C制定的开放标准图形格式。

- 制作完成之后，进行字体声明。由于各个浏览器支持的字体文件不同, 所以要声明多种字体达到浏览器兼容的目的。声明格式如下:

```
@font-face {
font-family: <font-family-name>;
src: [<url> [format(<string> #)]?| <font-face-name>] #;
font-weight: <weight>;
font-style: <style>;
}
```

- 在网页中使用字体。主要有两种方式: 一种是直接在网页中输入相应的Icon所代表的字体;另一种是使用CSS after伪类, 这样可以通过CSS直接控制Icon类别，只是IE6不兼容。

第一种方式:

```
<li>\01FD</li>
```

第二种方式:

```
.icon:after{
   content: '\01FD';
}
<li class="icon"></li>
```

推荐:

- IconMoon图标工具
- Font Awesome

CSS3常见新特性

新的元素选择器

E:nth-last-child(n), E:nth-of-type(n), E:nth-last-of-type(n), E:last-child, E:first-of-type, E:only-child, E:only-of-type, E:empty, E:checked, E:enabled, E:disabled, E::selection, E:not(s)

@font-face

见上文中iconfont部分。

border-radius

又称圆角属性，通常使用该属性将图片圆角化，如头像。

```
border-radius: 50%;
```

border-radius另外一个常用的手段是CSS动画。

word-wrap & text-overflow

word-wrap属性用来指出浏览器在单词内进行断句，防止字符串太长而找不到它的自然断句点时产生的溢出。

```
word-wrap: break-word;
```

text-overflow用于文本溢出:

单行缩略的实现如下:

```
.oneline {
   white-space: nowrap; //强制文本在一行内输出
   overflow: hidden; //隐藏溢出部分
   text-overflow: ellipsis; //对溢出部分加上...
}
```

多行缩略实现如下(主要针对webkit内核):

```
.multiline {
   display: -webkit-box !important;
   overflow: hidden;

   text-overflow: ellipsis;
   word-break: break-all;

   -webkit-box-orient: vertical;
   -webkit-line-clamp: 2;
}
```

background

主要是以下三个属性:

- background-clip 规定背景的绘制区域, 取值为border-box | padding-box | content-box | no-clip
- background-origin 规定背景的定位区域, 取值为border | padding | content
- background-size 规定背景图片的尺寸, 取值为[<length> | <percentage> | auto]{1,2} | cover | contain

Reference

- CSS-TRICKS
- Bootstrap中文