(2) 前端基础 (2) CSS0

—

1 CSS盒模型：
1.1 CSS中的盒子模型包括IE盒子模型 和 标准的 W3C盒子模型。
box-sizing有3个值： border-box【IE】、padding-box【火狐私有模型（没人用）,padding计算入宽度内】、content-box【W3C】

1.2 W3C标准盒子模型的宽度：左右border+左右的padding+width。
IE盒子模型的宽度：width。

2 画一条 0.5px的线
2.1 采用 meta viewport 的方式【为啥可以生效？？2个都是 1.0比例？？】：

```
<meta name="viewport" content="initial-scale=1.0, maximum-scale=1.0, user-scalable=no" />
```

2.2 采用 transform: scale() 的方式，进行比例的调节。

2.3 采用 border-image的方式。

3 link标签 和 @import 引入CSS的区别
【link是标签形式、无兼容性问题；
优先级、权重高；
可同时被加载、无 FOUC问题。】
3.1 link属于标签、无兼容性问题，@import是CSS提供的【最低 IE5】。

3.2 link引入的CSS权重高于 @import的 【为啥？？】

3.3 页面被加载时，link会被同时的加载【可并行？？】，而@import引用的CSS会等页面加载结束后加载、从而可能产生FOUC【flash of unstyled content。白屏、样式“再次挂载”】

3.4 link是标签、无兼容性问题；而@import是IE5以上才能识别！！！【可以通过@import这一特点对一些老版本的浏览器进行特定CSS样式的隐藏。】

总之：优先使用 link 去引入CSS，而不是 @import 的方式。

4 transition【需要触发事件、只有2帧，from...to...】 和 animation【无需触发事件、多帧。】的区别。

Animation和transition大部分属性是相同的，他们都是随时间改变元素的属性值，他们的主要区别是transition需要触发一个事件才能改变属性，而animation不需要触发任何事件的情况下才会随时间改变属性值，并且transition为2帧，从from .... to，而animation可以一帧一帧的。

5 flex布局【抽空可以看看、多实践，因为 RN 应用多半是用 flex布局的！！】

flex-direction：决定主轴的方向（即子item的排列方法）

```
.box {
flex-direction: row | row-reverse | column | column-reverse;
}
```

flex-wrap：决定换行规则

```
.box{
flex-wrap: nowrap | wrap | wrap-reverse;
}
```

flex-flow：

```
.box {
flex-flow: <flex-direction> || <flex-wrap>;
}
```

justify-content：对其方式，水平主轴对齐方式

align-items：对齐方式，竖直轴线方向

项目的属性（元素的属性）：

order属性：定义项目的排列顺序，顺序越小，排列越靠前，默认为0

flex-grow属性：定义项目的放大比例，即使存在空间，也不会放大

flex-shrink属性：定义了项目的缩小比例，当空间不足的情况下会等比例的缩小，如果定义个item的flow-shrink为0，则为不缩小

flex-basis属性：定义了在分配多余的空间，项目占据的空间。

flex：是flex-grow和flex-shrink、flex-basis的简写，默认值为0 1 auto。

align-self：允许单个项目与其他项目不一样的对齐方式，可以覆盖align-items，默认属性为auto，表示继承父元素的align-items

6 BFC【块格式化上下文，用于清除浮动 、 防止margin重叠等】
6.1 是一个独立的渲染区域，并且有一定的布局规则。

6.2 BFC区域是不会与 float box重叠。

6.3 BFC是页面的一个独立容器，子元素不会影响到外面，反之、外面也不会影响到它【外面、里面互不影响！！】

6.4 计算BFC宽度时，浮动元素也会参与其中【为啥？？】！！！

6.5 会生成BFC的元素
根元素 html？？？
float 不为 none 的元素
position 不为 fixed、absolute 的元素【原因？？】
display为 inline-block、table-cell、table-caption、flex、inline-flex的元素。
overflow 不为visible 【hidden，还有啥值？？？】的元素。

7 垂直、水平居中的方法。【分别至少有 4-5种 方案？？！ 统一整理】

8 JS动画 和 CSS3动画的差异性。

渲染线程分为main thread和compositor thread，如果css动画只改变transform和opacity，这时整个CSS动画得以在compositor trhead完成（而js动画则会在main thread执行，然后出发compositor thread进行下一步操作），特别注意的是如果改变transform和opacity是不会layout或者paint的。
区别：

功能涵盖面，js比css大

实现/重构难度不一，CSS3比js更加简单，性能跳优方向固定

对帧速表现不好的低版本浏览器，css3可以做到自然降级

css动画有天然事件支持

css3有兼容性问题

9 块元素 和 行元素【加上 行内块元素呢？？？】
块元素：独占一行，自动填满父元素；可以设置 width、hieght、margin、padding ！！

行元素：不会独占一行，width和height会失效；并且 垂直方向的 padding 和 margin 会失效。【但是可以设置 水平方向的 padding和margin 吧？？！！ 反正行内块元素可以设置 水平向的 margin、padding，但 垂直向不行。】

10 多行文本的文本省略号【有待实验、验证！！！】

```
display: -webkit-box
-webkit-box-orient:vertical
-webkit-line-clamp:3
overflow:hidden
```

11 将某元素隐藏起来的方法：
最直观的、常见的
visibility:hidden,
display:none,

几何学（）方面的
width、height:0,
transition: scale(0),

空间上的
translate: translate(-9999999px),【移动得远远的】,
z-index: -9999999,

颜色上的
opacity:0,
color: transparent。

12 双边距重叠问题（外边距重叠）

12.1 折叠的情况：

多个相邻（兄弟 | 父子关系）普通流的块元素垂直方向margin会重叠。

12.2 折叠的结果：

都是正数，取2者最大值【不用管方向？？？】

都是负数，取2者绝对值的最大者【不用管方向？？？】

一正一负，折叠结果为2者的相加的和。

13 position 各个取值的比较【默认的static、fixed、relative、absolute、sticky、inherit（继承父元素的position值）】

固定定位fixed：

元素的位置相对于浏览器窗口是固定位置，即使窗口是滚动的它也不会移动。Fixed定位使元素的位置与文档流无关，因此不占据空间。 Fixed定位的元素和其他元素重叠。

相对定位relative：

如果对一个元素进行相对定位，它将出现在它所在的位置上。然后，可以通过设置垂直或水平位置，让这个元素“相对于”它的起点进行移动。 在使用相对定位时，无论是否进行移动，元素仍然占据原来的空间。因此，移动元素会导致它覆盖其它框。

绝对定位absolute：

绝对定位的元素的位置相对于最近的已定位父元素，如果元素没有已定位的父元素，那么它的位置相对于<html>。 absolute 定位使元素的位置与文档流无关，因此不占据空间。 absolute 定位的元素和其他元素重叠。

粘性定位sticky：

元素先按照普通文档流定位，然后相对于该元素在流中的flow root（BFC）和 containing block（最近的块级祖先元素）定位。而后，元素定位表现为在跨越特定阈值前为相对定位，之后为固定定位。

默认定位Static：

默认值。没有定位，元素出现在正常的流中（忽略top, bottom, left, right 或者 z-index 声明）。

inherit:

规定应该从父元素继承position 属性的值。

14 清除浮动【只有 1、5 比较懂。其他应该需要多多实践！！！ BFC好像也可以、对应 方法2？？】

方法一：使用带clear属性的空元素

在浮动元素后使用一个空元素如 ，并在CSS中赋予.clear{clear:both;}属性即可清理浮动。亦可使用<br class="clear" />或<hr class="clear" />来进行清理。

方法二：使用CSS的overflow属性

给浮动元素的容器添加overflow:hidden;或overflow:auto;可以清除浮动，另外在 IE6 中还需要触发 hasLayout ，例如为父元素设置容器宽高或设置 zoom:1。

在添加overflow属性后，浮动元素又回到了容器层，把容器高度撑起，达到了清理浮动的效果。

方法三：给浮动的元素的容器添加浮动

给浮动元素的容器也添加上浮动属性即可清除内部浮动，但是这样会使其整体浮动，影响布局，不推荐使用。

方法四：使用邻接元素处理

什么都不做，给浮动元素后面的元素添加clear属性。

方法五：使用CSS的:after伪元素

结合:after 伪元素（注意这不是伪类，而是伪元素，代表一个元素之后最近的元素）和 IEhack ，可以完美兼容当前主流的各大浏览器，这里的 IEhack 指的是触发 hasLayout。

给浮动元素的容器添加一个clearfix的class，然后给这个class添加一个:after伪元素实现元素末尾添加一个看不见的块元素（Block element）清理浮动。

15 如何实现图片在某个容器中的居中【？？！ 之前一直直接是 水平、垂直 居中】

父元素固定宽高，利用定位及设置子元素margin值为自身的一半。

父元素固定宽高，子元素设置position: absolute，margin：auto平均分配margin

css3属性transform。子元素设置position: absolute; left: 50%; top: 50%;transform: translate(-50%,-50%);即可。

将父元素设置成display: table, 子元素设置为单元格 display: table-cell。

弹性布局display: flex。设置align-items: center; justify-content: center

16 float元素的 display 是 block！！！

17 三栏布局，尽可能的多写，浮动布局时，3个div的生成顺序有没有影响？？？

2列定宽 1列自适应【需要多多实践、总结？！！】：

1、使用float+margin：

给div设置float：left，left的div添加属性margin-right：left和center的间隔px,right的div添加属性margin-left：left和center的宽度之和加上间隔

2、使用float+overflow：

给div设置float：left，再给right的div设置overflow:hidden。这样子两个盒子浮动，另一个盒子触发bfc达到自适应

3、使用position：

父级div设置position：relative，三个子级div设置position：absolute，这个要计算好盒子的宽度和间隔去设置位置，兼容性比较好，

4、使用table实现：

父级div设置display：table，设置border-spacing：10px//设置间距，取值随意,子级div设置display:table-cell，这种方法兼容性好，适用于高度宽度未知的情况，但是margin失效，设计间隔比较麻烦，

5、flex实现：

parent的div设置display：flex；left和center的div设置margin-right；然后right 的div设置flex：1；这样子right自适应，但是flex的兼容性不好

6、grid实现：

parent的div设置display：grid，设置grid-template-columns属性，固定第一列第二列宽度，第三列auto，

对于两侧定宽中间自适应的布局，对于这种布局需要把center放在前面，可以采用双飞翼布局：圣杯布局，来实现，也可以使用上述方法中的grid，table，flex，position实现

18 CSS中的calc属性【任何长度值均可使用， 运算符前后都需要保留一个空格】：

calc动态计算长度值，任何长度值都可以使用calc()函数去计算。
注意：运算符前后需要有 一个空格 隔开。如 —— width: calc(50% - 2px).

19 有一个width 300, hieght 300的元素，怎么实现在屏幕上垂直、水平居中？？

对于行内块级元素，

1、父级元素设置text-alig：center，然后设置line-height和vertical-align使其垂直居中，最后设置font-size：0消除近似居中的bug

2、父级元素设置display：table-cell，vertical-align：middle达到水平垂直居中

3、采用绝对定位，原理是子绝父相，父元素设置position：relative，子元素设置position：absolute，然后通过transform或margin组合使用达到垂直居中效果，设置top：50%，left：50%，transform：translate（-50%，-50%）

4、绝对居中，原理是当top,bottom为0时，margin-top&bottom设置auto的话会无限延伸沾满空间并平分，当left，right为0时,margin-left&right设置auto会无限延伸占满空间并平分，

5、采用flex，父元素设置display：flex，子元素设置margin：auto

6、视窗居中，vh为视口单位，50vh即是视口高度的50/100，设置margin：50vh auto 0，transform：translate(-50%)

20 display：table 和 本身的table有什么区别？ 【看不懂题目呀？？！】

Display:table和本身table是相对应的，区别在于，display：table的css声明能够让一个html元素和它的子节点像table元素一样，使用基于表格的css布局，是我们能够轻松定义一个单元格的边界，背景等样式，而不会产生因为使用了table那样的制表标签导致的语义化问题。

之所以现在逐渐淘汰了table系表格元素，是因为用div+css编写出来的文件比用table边写出来的文件小，而且table必须在页面完全加载后才显示，div则是逐行显示，table的嵌套性太多，没有div简洁

二

1 想要改变一个DOM元素的字体颜色，但又不能在它身上操作，该如何？？
可以更改 父元素的 color【CSS的color是继承属性！！】。

2 CSS的新属性
flex布局（灵活，但是兼容性不强）、first-of-type和nth-child、box-sizing、动画animation、2d和3d变换、颜色加了透明和rgba、text和box跟踪阴影shadow、媒体查询meta（已完成自适应布局？？！）等等。

CSS3的新特性中，在布局方面新增了flex布局，在选择器方面新增了例如first-of-type,nth-child等选择器，在盒模型方面添加了box-sizing来改变盒模型，在动画方面增加了animation，2d变换，3d变换等，在颜色方面添加透明，rbga等，在字体方面允许嵌入字体和设置字体阴影，最后还有媒体查讯等

3 line-height 和 height 的区别？？
line-height一般是指布局里面一段文字上下行的高度，是针对字体来设置的，height一般是指容器的整体高度。

4 背景色会填充元素的哪些区域？？
content、padding、border。【margin肯定不会呀！！ 所以 margin区域 永远是 无色、透明的】

5 inline-block、inline 和 block的区别？？？
为啥 img是 inline【不是行内块？？】 还可以设置宽高？？

块级元素，前后均有换行符，能设置宽、高； margin、padding 水平垂直方向都有效。

行内元素，设置 宽高无效；margin、padding 都只能设置、水平方向的，前后均无换行符！！

行内块元素，能设置 宽高； margin、padding 在垂直水平方向 均可设置！！！ 前后无换行符！！

6 重排重绘？？ 怎么减少它们？？ 让文档脱离文档流的方法有哪些？？

DOM的变化影响到了预算内宿的几何属性比如宽高，浏览器重新计算元素的几何属性，其他元素的几何属性也会受到影响，浏览器需要重新构造渲染书，这个过程称之为重排，浏览器将受到影响的部分重新绘制在屏幕上 的过程称为重绘，引起重排重绘的原因有：

添加或者删除可见的DOM元素，

元素尺寸位置的改变

浏览器页面初始化，

浏览器窗口大小发生改变，重排一定导致重绘，重绘不一定导致重排，

减少重绘重排的方法有：

不在布局信息改变时做DOM查询，

使用csstext,className一次性改变属性

使用fragment

对于多次重排的元素，比如说动画。使用绝对定位脱离文档流，使其不影响其他元素

7 overflow的原理【有点看不懂呀？？！！】

要讲清楚这个解决方案的原理，首先需要了解块格式化上下文，A block formatting context is a part of a visual CSS rendering of a Web page. It is the region in which the layout of block boxes occurs and in which floats interact with each other.翻译过来就是块格式化上下文是CSS可视化渲染的一部分，它是一块区域，规定了内部块盒 的渲染方式，以及浮动相互之间的影响关系

当元素设置了overflow样式且值部位visible时，该元素就构建了一个BFC，BFC在计算高度时，内部浮动元素的高度也要计算在内，也就是说技术BFC区域内只有一个浮动元素，BFC的高度也不会发生塌缩，所以达到了清除浮动的目的，

8 display: none【重排，对应 v-if】 和 visibility: hidden【重绘，仍占据空间、对应 v-show。】 的区别

display: none，真的删除了元素，再恢复会触发重排、较耗性能。
对应 v-if 可通过 其将某元素 进行离线化 ---> 性能优化的点。

visibility: hidden，不删除元素、只是将其隐藏起来、且不会触发该元素已绑定的事件；恢复时应该只会触发 重绘而已吧？？
对应 v-show。

1. visibility：hidden，该元素隐藏起来了，但不会改变页面布局，但是不会触发该元素已经绑定的事件

2. display：none，把元素隐藏起来，并且会改变页面布局，可以理解成在页面中把该元素删除掉。

9 垂直居中【注意： transform的 translate的百分比分别对应的是当前元素标签的宽高比例。】。

父元素固定宽高，利用定位及设置子元素margin值为自身的一半。

父元素固定宽高，子元素设置position: absolute，margin：auto平均分配margin

css3属性transform。子元素设置position: absolute; left: 50%; top: 50%;transform: translate(-50%,-50%);即可。

将父元素设置成display: table, 子元素设置为单元格 display: table-cell。

弹性布局display: flex。设置align-items: center; justify-content: center;

10 CSS预处理器。

less、sass等

完