- CSS
  - 面试方式
  - display有哪些属性？ *
  - inline & inline-block的区别能说说么？ *
  - 行内元素和块级元素有什么区别呢？ *
  - 有哪些行内和块级元素？ *
  - 块级元素和内联元素有哪几种转换方式？ **
- 选择器 & 优先级
  - 1. 选择器的优先级是什么样的？ 选择器如何做样式判断？ 这段样式能不能生效？ *
  - 2. 特殊场景的优先级如何判断？ *
  - 3. 可继承的样式有哪些？ **
- 隐藏和显示相关
  - 1. 有哪些可以隐藏过一个元素的方法？有什么区别？ **
  - 2. display vs visibility有什么区别？ **
- 盒模型及其特性
  - 1. 简单说说标准盒模型、IE盒模型分别是什么？怎么转换？*
  - 2. 伪元素和伪类是什么？如何使用？区别是什么？ *
- 图片格式以及CSS-sprites
  - 1. 图片格式有哪些？怎么应用？如何选择？*
  - 2. CSS-sprites 精灵图、雪碧图怎么处理？ *
- 像素密度与图片应用
  - 1. 像素密度有了解吗？ *
  - 2. 如何在图片的加载上应用动态密度？ *
- css工程化与预处理
  - 1. css类库 与工程化的理解？ ***
- 单行多行文本超出
  - 1. 手写一个单行 & 多行的文本超出省略 *
- px em rem
  - 1. 多种单位的差别 *
  - 2. 如何利用rem实现响应式？ 项目如何实现响应式的?
- 布局
  - 1. 定位浮动 - 简单聊聊看浮动的影响还有原理？ *
  - 2. 浮动停留的条件？浮动元素移动遵循的空间？*
  - 3. 高度塌陷原因，如何解决高度塌陷？*
  - 4. 简单说说如何创建BFC，以及如何解决相应一些问题？ **
  - 5. BFC正作用有哪些？**

# CSS

**面试方式**

1. 分级 初级 => 中级 * 概念 + 原理 + 场景

中级 ** 实际应用 + 复合场景

高级 *** 技巧 + 工程化

2. 面试方式 特点：相对零散，从点到面

> 面试题：

**display有哪些属性？ ***

- none - 不展示
- block - 块类型
- inline - 行内
- inline-block - 默认行内块状
- list-item | table
- inherit - 继承

**inline & inline-block的区别能说说么？ ***

- inline: 共享一行，行内概念
- block: 独占一行
- inline-block：共享一行，内容作为block对象呈现

**行内元素和块级元素有什么区别呢？ ***

- 行内元素：
  - 无法设置宽高

    - 水平方向可设置margin + padding，垂直方向则无法设置
    - 不会自动换行

- 块级元素：

    - 可以设置宽高
    - 水平垂直方向可设置margin + padding
    - 可以自动换行
    - 多个块状是默认从上往下换行排列

**有哪些行内和块级元素？ ***

- 块级元素

> div form h1 pre table ul……

- 行内

> a p br code em img i input strong textarea……

**块级元素和内联元素有哪几种转换方式？ **

1. display
2. float

float: left/right => display: block ** => 去除行内元素之间的空白问题 ***

1. position position: absolute / fixed => 块级

# 选择器 & 优先级

**1. 选择器的优先级是什么样的？ 选择器如何做样式判断？ 这段样式能不能生效？ ***

- 内联样式 1000
- id选择器 #id 100
- 类选择器 .class 10
- 属性选择器 a[ref="link"] 10
- 标签选择器 div 1

- 伪类选择器 li:last-child 10
- 伪元素选择器 li:before 1
- 兄弟选择器 div+p 0
- 子选择器 ul>li 0
- 后代选择器 li a 0
- 通配符 * 0

**2. 特殊场景的优先级如何判断？ ***

- !important 优先级最高
- 如果优先级相同，则后者高于前者
- 继承得到的样式，优先级最低

**3. 可继承的样式有哪些？ **

- 字体
  - font-family,font-weight,font-size,font-style
- 文本
  - text-indent,text-align,line-height,word-spacing,letter-spacing,color
- 元素
  - visibility
- 列表布局
  - list-style
- 光标
  - cursor

# 隐藏和显示相关

**1. 有哪些可以隐藏过一个元素的方法？有什么区别？ **

- display: none; 不占位
- visibility: hidden; 占位
- opacity: 0; 占位

- position: absolute; 不占位
- z-index: 负值; 不占位
- clip 占位
- transform: scale(0, 0) 占位

**2. display vs visibility有什么区别？ **

- 他们俩都是让元素隐藏和展示。
- 浏览器渲染时，display不占据空间，渲染树中会不存在
- visibility，占据一根树枝
- 继承属性来说，display不会被继承，visibility会被继承
- 性能影响上，display造成文档的**重排**，但是修改visibility只会导致文本的**重绘**

# 盒模型及其特性

**1. 简单说说标准盒模型、IE盒模型分别是什么？怎么转换? ***

> 盒模型特点：

- content + padding + border + margin
- 区别：
  - 标准盒模型 - width和height只包含content部分
  - IE盒模型 - width和height包含了content + padding + border部分
- 转换：
  - box-sizing: content-box / border-box;

**2. 伪元素和伪类是什么？如何使用？区别是什么？ ***

伪元素：只出现在css样式表中，不存在于doc中。

```
p::before {
    content: 'zhaowa'
}
p::first-line {
```

```
        background: red;
    }
```

伪类：已有的元素上加上特殊类别，不产生新的元素。

```
    div:hover {
        color: red;
    }
```

# 图片格式以及CSS-sprites

**1. 图片格式有哪些？怎么应用？如何选择？ ***

- BMP，无损、没有压缩。通常体积较大。
- GIF，无损、采用了LZW压缩算法。仅支持8bit索引色，支持动图。
- JPEG，有损、直接色存储，适合还原度要求较高的照片。
- PNG-8，无损、使用索引色。体积更优秀，并且支持透明度调节。
- PNG-24，无损、使用直接色，压缩。
- SVG，无损、svg放大不会失真，所以适合logo、icon。
- webP，有损+无损、直接色、支持透明度、压缩。chrome、opera支持

**2. CSS-sprites 精灵图、雪碧图怎么处理？ ***

> 所有涉及到的图片，放到一张大图中去 background-image, background-repeat, background-position

# 像素密度与图片应用

**1. 像素密度有了解吗？ ***

经典设备宽高 414px * 896px

物理像素 1242px * 2688px => 1242 / 414 = 3

=> 逻辑像素：物理像素 = 1：3 => 像素密度3 => 3倍屏

**2. 如何在图片的加载上应用动态密度？ ***

设计师提供 @2x，@3x，@4x 200 * 300 => 3倍屏幕 => 600 * 900

```
    image {
        background: ('1x.png');
    }
    // 利用媒体查询
    @media only screen and (min-deivce-pixel-ratio: 3) {
        image {
            background: ('3x.png');
        }
    }
```

```
@media screen and (max-width: 992px) {
  body {
    background-color: blue;
  }
}

/* 在 600 像素或更小的屏幕上，将背景色设置为橄榄色 */
@media screen and (max-width: 600px) {
  body {
    background-color: olive;
  }
```

# css工程化与预处理

### 1. css类库 与工程化的理解？ ***

- 预处理器：less scss stylus => 利用编译库提供能力，提供层级、mixin、变量、循环、函数
- 后处理器：postCss => 利用后处理编译，属性增加前缀，实现跨浏览器兼容

# 单行多行文本超出

### 1. 手写一个单行 & 多行的文本超出省略 *

```
    // 单行超出
    overflow: hidden;
    text-overflow: ellipsis; // 超出省略号
```

```
    white-space: nowrap;    // 不换行

    // 多行超出
    overflow: hidden;
    text-overflow: ellipsis; // 超出省略号
    display: -webkit-box;  // 弹性伸缩盒子模型
    -webkit-box-orient: vertical; // 从上往下垂直排列
    -webkit-line-clamp: 3;  // 显示的行数

    // 兼容性方案
    p {
        position: relative;
        line-height: 18px;
        height: 40px;
        overflow: hidden;
    }
    p::after {
        content: '...';
        position: absolute;
        bottom: 0;
        right: 0;
    }
    // 方案有什么不足之处 => 固定行高 => js处理 后处理器
```

# px em rem

**1. 多种单位的差别 ***

- 百分比：子元素的百分比相对于直接父元素的对应属性
- em: 相对于父元素的字体大小倍数
- rem: 相对于**根元素字体大小**的倍数
- vw: 视窗宽度，满视窗宽度为 100vw
- vh: 视窗高度，满视窗高度为 100vh
- vmin: vw和vh中较小值
- vmax: vw和vh中较大值

**2. 如何利用rem实现响应式？ 项目如何实现响应式的?**

- 根据当前设备的视窗宽度与设计稿的宽度得到一个比例
- 根据比例设置根节点的font-size
- 所有长度单位都用rem

# 布局

**1. 定位浮动 - 简单聊聊看浮动的影响还有原理？ ***

- 浮动工作原理：

> 浮动元素脱离文档流，不占据空间 => 不受原有文档流的影响，同时无法影响原有父类 => 高度塌陷

**2. 浮动停留的条件？浮动元素移动遵循的空间？ ***

> 浮动元素碰到包含他的边框或者其他浮动元素的时候会停留 => 浮动元素可以左右移动 => 浮动元素高度独立，不会再影响撑开原有父类的高度

**3. 高度塌陷原因，如何解决高度塌陷？ ***

- 给父级定义height
- 浮动元素之后，给一个div，clear: both;
- 父级标签增加overflow:hidden;
- 用伪元素模拟div

**4. 简单说说如何创建BFC，以及如何解决相应一些问题？ **

- 创建BFC的条件：
    - 根元素body
    - 元素设置浮动：float除了none之外
    - position 脱离文本流的操作
    - display的inlin-block table-cell table-caption flex
    - overflow的 hidden auto scroll
- BFC的特点：
    - 垂直方向上，自上而下排列的，和文档流的排列方式一致
    - BFC中上下相邻的两个容器margin会重叠
    - 计算BFC高度时要计算浮动元素
    - BFC不会影响外部元素

**5. BFC正作用有哪些？ **

解决margin重叠问题、解决高度塌陷、创建自适应布局

**6. 有几种办法能实现两列布局？实现一个左边宽度固定，右侧宽度自适应的两列布局？***

```
// 1. 浮动 + 生成BFC不重叠
.left {
    width: 100px;
    height: 200px;
    float: left;
}
.right {
    height: 200px;
    overflow: hidden;
}

// 2. 浮动 + width auto
.container {
    height: 200px;
}
.left {
    width: 200px;
    height: 200px;
    float: left;
}
.right {
    margin-left: 200px;
    width: auto;
}

// 3. flex大法好
.container {
    height: 200px;
    display: flex;
}
.left {
    width: 200px;
}
.right {
    flex: 1;
}
```

**7. 两列布局可以，那再加一列呢？左右两栏宽度固定，中间自适应？***

```
// 1. 绝对布局法
.container {
    position: relative;
    height: 200px;
}
.left {
```

```
    position: absolute;
    width: 100px;
    height: 200px;
}
.right {
    position: absolute;
    width: 200px;
    height: 200px;
    top: 0;
    right: 0;
}
.center {
    margin-left: 100px;
    margin-right: 200px;
    height: 200px;
}
// 2. flex大法好
.container {
    display: flex;
    height: 200px;
}
.left {
    width: 100px;
}
.right {
    width: 200px;
}
.center {
    flex: 1;
}
// 3. 圣杯布局
.container {
    height: 200px;
    padding-left: 100px;
    padding-right: 200px;
}
.center {
    float: left;
    width: 100%;
    height: 200px;
}
.left {
    position: relative;
    left: -100px;

    float: left;
    margin-left: -100%;
    width: 100px;
    height: 200px;
}
.right {
    position: relative;
    left: 200px;

    float: right;
    margin-left: -200px;
    width: 200px;
```

```
    height: 200px;
}
// 4. 双飞翼
.container {
    height: 200px;
}
.left {
    float: left;
    margin-left: -100%;
    width: 100px;
    height:200px;
}
.right {
    float: left;
    margin-left: -200px;
    width: 200px;
    height: 200px;
}
.wrapper {
    float: left;
    width: 100%;
    height: 200px;
}
.center {
    margin-left: 100px;
    margin-right: 200px;
    height: 200px;
}
```

### 8. 水平垂直居中问题？ **

```
// 1. 绝对定位
div {
    position: absolute;
    left: 50%;
    top: 50%;
    margin-top: -height/2;
    margin-left: -width/2;
}
// 2. 自我拉扯
div {
    margin: auto;
    position: absolute;
    top: 0;
    left: 0;
    right: 0;
    bottom: 0;
}
// 3. flex大法好
.parent {
    display: flex;
    justify-content: center;
```

```
        align-items: center;
    }
```

### 奇技淫巧

- 三角形 => 梯形 => 扇形 => 基础元素 + 技巧

```
    width: 0;
    height: 0;
    border-top: 100px solid black;
    border-left: 100px solid transparent;
    border-right: 100px solid transparent;
    transform: rotate(90deg)
```

1px、12px font => 变形进行处理 动画 => 幻灯片 / 电影

```
/* 动画代码 */
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}

/* 向此元素应用动画效果 */
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  0%   {background-color: red;}
  25%  {background-color: yellow;}
  50%  {background-color: blue;}
  100% {background-color: green;}
}
```