一、什么是Sass?

Sass 是一种 CSS 的预编译语言。它提供了 变量、嵌套、 混合（mixins）、 函数等功能，并且完全兼容 CSS 语法。Sass 能够帮助复杂的样式表更有条理， 并且易于在项目内部或跨项目共享设计。

文件后缀为.scss或者.sass，一般使用.scss。

类似的css预处理有less。

二、安装

1. vscode下载插件“ easy sass”
2. 在用户设置中加入一下代码：`"easysass.targetDir": "./css"`

三、基本语法

1. 变量与插值与法

a. SCSS允许使用“$”声明变量

```
$blue : #1875e7;
div {
  color : $blue;
}
```

b. 如果变量需要镶嵌在字符串之中，就必须需要写在插值语句“#{}”之中

```
$side: left;
.rounded {
    border-#{$side}-radius: 5px;
}
```

2. 计算功能：**SCSS**允许在代码中使用算式

```
body {
    margin: (14px/2);
    top: 50px + 100px;
    right: $var * 10%;
}
```

3. 嵌套：**SCSS**允许选择器嵌套

```
div {
    h1 {
        color : red;
    }
}
// 属性也可以嵌套
p{
    border:{
        color:red;
     }
     //注意 : 属性嵌套后面需要加上冒号。
}
// 在嵌套的代码块内，可以使用 & 引用父元素
a{
    &:hover{
        color:#ffb3ff;
    }
}
```

4. 注释

a. /* 多行注释 */，会保留到编译后的文件。

b. // 单行注释，只保留在SASS源文件中，编译后被省略。

c. 在/*后面加一个感叹号，表示这是"重要注释"。即使是压缩模式编译，也会保留这行注释，通常可以用于声明版权信息。

```
/*!
    重要注释!
*/
```

四、代码的重用

1. 占位符%

```
// SCSS允许通过 % 来声明一个选择器，但必须通过 @extend 指令调用
%wh{
    width:100px;
    height:100px;
}

div{
    @extend wh;
}
```

2. 继承 @extend

```
// SCSS允许一个选择器，继承另一个选择器
.class1 {
   border: 1px solid #ddd;
}
// class2要继承class1，就要使用 @extend 命令
.class2 {
   @extend .class1;
}
```

3. 混合宏 @mixin

```
// 使用 @mixin 命令，定义一个可以重用的代码块。
@mixin left{
    float: left;
    margin-left:10px;
}
// 使用 @include 命令，调用这个混合宏。
div{
    @include left;
}
// @mixin的强大之处，在于可以指定参数和默认值。
@mixin common($w,$h,$bg:pink) {
    width:$w;
    height:$h;
    background:$bg;
}
// 使用 的时候，根据需要加入参数:
div{
    @include common(100px,100px,red)
}
```

4. 插入文件

```
// @import== 命令，用来插入外部文件。
@import "reset.scss";
```

五、高级用法

1. 循环语句

```
// @for $var from through  含头含尾
```

```
@for $i from 1 through 10{
    .border-#{$i}{
        border:#{$i}px solid blue;
    }
}
// @for $var from to  含头不含尾
@for $i from 1 to 4 {
    .list li {
        &:nth-child(#{$i}) {
            font-size: #{$i}0px;
        }
    }
}
```

2. 自定义函数

```
// @function 用于声明函数 @return 用于返回值
// 先声明后使用
@function double($n) {
    @return $n * 2;
}

#sidebar {
    width: double(5px);
}
```

> 背景: 由于手机电脑设备种类繁多，尺寸大小不一，为了能够让一个网站兼容多个终端设备，出现了以下几种解决方案:

一、流式布局(百分比布局)

通过高度定死宽度百分比来适应不同的屏幕

经典的流式布局

1. 左侧固定,右侧自适应

```
* {
    box-sizing: border-box;
}

.box {
    width: 100%;
    height: 700px;
    position: relative;
}

.right {
    width: 100%;
    height: 100%;
    background: red;
    padding-left: 200px;
}

.left {
    width: 200px;
    height: 100%;
```

```
    background: green;
    position: absolute;
}
```

2. 右侧固定左侧自适应

```
* {
    box-sizing: border-box;
}

.box {
    width: 100%;
    height: 700px;
    position: relative;
}

.right {
    width: 200px;
    height: 100%;
    background: ref;
    position: absolute;
    right: 0;
    top: 0;
}

.left {
    width: 100%;
    height: 100%;
    padding-right: 200px;
    background: green;
}
```

3. 两侧固定,中间自适应(圣杯布局)

```
  参考以上两个案例，发挥自己聪明的小脑壳实现它吧！！！
```

缺点

对于大屏幕来说,用户体验并不是特别好,有些布局元素会显得很长

> 解决方案: rem布局

二、响应式布局

媒体查询是实现响应式布局的关键技术

1. 常见设备尺寸

```
/* 超小设备（手机，小于 768px） */
@media screen and (min-width:480px){ ... }

/* 小型设备（平板电脑，768px 起） */
@media screen and (min-width:768px) { ... }

/* 中型设备（台式电脑，992px 起） */
@media screen and (min-width:992px){ ... }

/* 大型设备（大台式电脑，1200px 起） */
@media screen and (min-width:1200px){ ... }
```

2. 最小宽度 min-width

```
@media screen and (min-width:480px){ ... }
```

3. 最大宽度 max-width

```
@media screen and (max-width:1200px){ ... }
```

4. 多个媒体特性的使用:并且

```
@media screen and (min-width:480px) and (max-width:992px){ ... }
```

5. not关键词的使用： 除了

```
@media not screen and (min-width:1200px){ ... }
```

6. only关键词的使用: 仅仅，一般用来排除不支持媒体查询的浏览器

```
@media only screen and (min-width:480px){ ... }
```