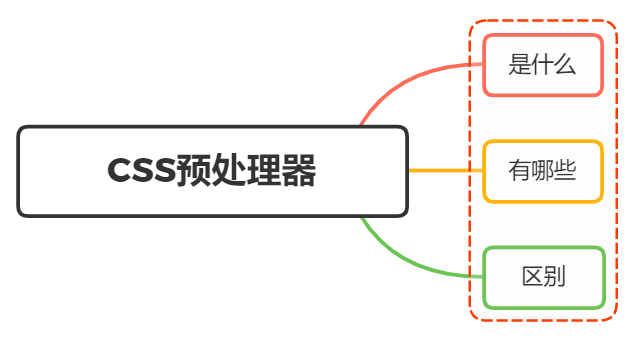
# 面试官：说说对Css预编语言的理解？有哪些区别?



## 一、是什么

Css 作为一门标记性语言，语法相对简单，对使用者的要求较低，但同时也带来一些问题

需要书写大量看似没有逻辑的代码，不方便维护及扩展，不利于复用，尤其对于非前端开发工程师来讲，往往会因为缺少 Css 编写经验而很难写出组织良好且易于维护的 Css 代码

Css预处理器便是针对上述问题的解决方案

#### 预处理语言

扩充了 Css 语言，增加了诸如变量、混合（mixin）、函数等功能，让 Css 更易维护、方便

本质上，预处理是Css的超集

包含一套自定义的语法及一个解析器，根据这些语法定义自己的样式规则，这些规则最终会通过解析器，编译生成对应的 Css 文件

## 二、有哪些

Css预编译语言在前端里面有三大优秀的预编处理器，分别是：

* sass
* less
* stylus

### sass

2007 年诞生，最早也是最成熟的 Css预处理器，拥有 Ruby 社区的支持和 Compass 这一最强大的 Css框架，目前受 LESS 影响，已经进化到了全面兼容 Css 的 Scss

文件后缀名为.sass与scss，可以严格按照 sass 的缩进方式省去大括号和分号

### less

2009年出现，受SASS的影响较大，但又使用 Css 的语法，让大部分开发者和设计师更容易上手，在 Ruby社区之外支持者远超过 SASS

其缺点是比起 SASS来，可编程功能不够，不过优点是简单和兼容 Css，反过来也影响了 SASS演变到了Scss 的时代

### stylus

Stylus是一个Css的预处理框架，2010 年产生，来自 Node.js社区，主要用来给 Node 项目进行 Css 预处理支持

所以Stylus 是一种新型语言，可以创建健壮的、动态的、富有表现力的Css。比较年轻，其本质上做的事情与SASS/LESS等类似

## 三、区别

虽然各种预处理器功能强大，但使用最多的，还是以下特性：

* 变量（variables）
* 作用域（scope）
* 代码混合（ mixins）
* 嵌套（nested rules）
* 代码模块化（Modules）

因此，下面就展开这些方面的区别

### 基本使用

less和scss

.box {  
 display: block;  
}

sass

.box  
 display: block

stylus

.box  
 display: block

### 嵌套

三者的嵌套语法都是一致的，甚至连引用父级选择器的标记 & 也相同

区别只是 Sass 和 Stylus 可以用没有大括号的方式书写

less

.a {  
 &.b {  
 color: red;  
 }  
}

### 变量

变量无疑为 Css 增加了一种有效的复用方式，减少了原来在 Css 中无法避免的重复「硬编码」

less声明的变量必须以@开头，后面紧跟变量名和变量值，而且变量名和变量值需要使用冒号:分隔开

@red: #c00;  
  
strong {  
 color: @red;  
}

sass声明的变量跟less十分的相似，只是变量名前面使用@开头

$red: #c00;  
  
strong {  
 color: $red;  
}

stylus声明的变量没有任何的限定，可以使用$开头，结尾的分号;可有可无，但变量与变量值之间需要使用=

在stylus中我们不建议使用@符号开头声明变量

red = #c00  
  
strong  
 color: red

### 作用域

Css 预编译器把变量赋予作用域，也就是存在生命周期。就像 js一样，它会先从局部作用域查找变量，依次向上级作用域查找

sass中不存在全局变量

$color: black;  
.scoped {  
 $bg: blue;  
 $color: white;  
 color: $color;  
 background-color:$bg;  
}  
.unscoped {  
 color:$color;  
}

编译后

.scoped {  
 color:white;/\*是白色\*/  
 background-color:blue;  
}  
.unscoped {  
 color:white;/\*白色（无全局变量概念）\*/  
}

所以，在sass中最好不要定义相同的变量名

less与stylus的作用域跟javascript十分的相似，首先会查找局部定义的变量，如果没有找到，会像冒泡一样，一级一级往下查找，直到根为止

@color: black;  
.scoped {  
 @bg: blue;  
 @color: white;  
 color: @color;  
 background-color:@bg;  
}  
.unscoped {  
 color:@color;  
}

编译后：

.scoped {  
 color:white;/\*白色（调用了局部变量）\*/  
 background-color:blue;  
}  
.unscoped {  
 color:black;/\*黑色（调用了全局变量）\*/  
}

### 混入

混入（mixin）应该说是预处理器最精髓的功能之一了，简单点来说，Mixins可以将一部分样式抽出，作为单独定义的模块，被很多选择器重复使用

可以在Mixins中定义变量或者默认参数

在less中，混合的用法是指将定义好的ClassA中引入另一个已经定义的Class，也能使用够传递参数，参数变量为@声明

.alert {  
 font-weight: 700;  
}  
  
.highlight(@color: red) {  
 font-size: 1.2em;  
 color: @color;  
}  
  
.heads-up {  
 .alert;  
 .highlight(red);  
}

编译后

.alert {  
 font-weight: 700;  
}  
.heads-up {  
 font-weight: 700;  
 font-size: 1.2em;  
 color: red;  
}

Sass声明mixins时需要使用@mixinn，后面紧跟mixin的名，也可以设置参数，参数名为变量$声明的形式

@mixin large-text {  
 font: {  
 family: Arial;  
 size: 20px;  
 weight: bold;  
 }  
 color: #ff0000;  
}  
  
.page-title {  
 @include large-text;  
 padding: 4px;  
 margin-top: 10px;  
}

stylus中的混合和前两款Css预处理器语言的混合略有不同，他可以不使用任何符号，就是直接声明Mixins名，然后在定义参数和默认值之间用等号（=）来连接

error(borderWidth= 2px) {  
 border: borderWidth solid #F00;  
 color: #F00;  
}  
.generic-error {  
 padding: 20px;  
 margin: 4px;  
 error(); /\* 调用error mixins \*/  
}  
.login-error {  
 left: 12px;  
 position: absolute;  
 top: 20px;  
 error(5px); /\* 调用error mixins，并将参数$borderWidth的值指定为5px \*/  
}

### 代码模块化

模块化就是将Css代码分成一个个模块

scss、less、stylus三者的使用方法都如下所示

@import './common';  
@import './github-markdown';  
@import './mixin';  
@import './variables';

## 参考文献

* https://jelly.jd.com/article/5dcb9c73641a030153732a89
* https://zhuanlan.zhihu.com/p/23382462
* https://baike.baidu.com/item/Less/17570158