1、什么是sass

动态的css    （可以定义变量，可以定义嵌套、混合、写一些编程语句 if   for...）

2、sass使用

安装：  首先安装ruby

在ruby平台下 安装sass      ：  gem  install  sass   （如果不能正常安装  需要直接安装 sass文件 ）   gem  install d:/sass-3.4.22.gem

1、sass使用

创建目录，在该目录下创建一个扩展名为scss文件   ：

从c盘转到d盘 ：  命令--  d:

从d盘进入到某个目录下 ：  命令--cd  目录名   cd..  回退一级目录

设置scss文件的输出目录 ：   sass style.scss style.css

自动监听scss 文件 ： sass --watch style.scss:style.css

SASS提供四个编译风格的选项：

\* nested：嵌套缩进的css代码，它是默认值。

\* expanded：没有缩进的、扩展的css代码。

\* compact：简洁格式的css代码。

\* compressed：压缩后的css代码。

例如：　sass --style compressed style.scss style.css

sass编译中文出错 ，找到engine.rb文件  ， 在该文件中 记一下 中文编码格式配置：

                                                  Encoding.default\_external = Encoding.find('utf-8')

sass语法：

注释 ：  //    /\*\*/

变量定义  ：   $变量

嵌套 ：

传统css:

  .nav {

     ...

     }

   .nav  ul{

          ...

     }

    .nav ul li{

    ...

     }

嵌套方式：

     .nav{

          width:960px;

          height : 40px;

          ul{

               margin-left : 20px;

               li{

                    float: left;

               }

          }

     }

符合属性嵌套：

border:{

     color: red;

     style:solid;

     width:1px;

}

&表示继承父级标签

a{

     &:hover{ color:red }

}

代码重用(函数   混合)：

width:960px;

margin:0 auto;

.public{  //无参数   使用 @extend  .public;

     width:960px;

}

@mixin  public{  //可以定义参数   使用 :  @include 导入

     ....

}

@mixin public($height:200px){//定义一个默认值

    width: 960px;

    height: $height;

    margin: 0 auto;

}

#header{

    @include public(400px);给具体的参数值 默认值无效，如果不传递参数 就按照默认值执行

}

导入：  @import "xxx.scss"   将多个scss文件合并成一个css文件

3、git

svn 使用 ： 检出---修改内容--提交--更新

svn -- 服务器创建一个svn项目仓库--客户端检出项目仓库  ---  客户端添加文件 ---  提交  ---- 更新update

svn ： 集中式版本管理工具 ， 一个服务器对应多个客户端      svn工作时不能断网

git ：  分布式版本管理工具  ， 一台电脑 既可以是客户端  也可以是服务器端   ， 工作过程中  可以断开网络

git使用：

1、创建一个项目目录

2、检出git的项目仓库    命令  ----   git init

git status  查看当前被git管理的文件

3、本地项目文件夹新增的文件 首先要进行 添加 操作，添加到 本地的项目仓库下 ，  命令  ---   git  add  文件全称

   将全部文件添加到git项目中 ：  git  add .    或   git  add --all

   如果添加的过程中想忽略某个或某些文件不被git管理 ， 需要在项目中创建一个   .gitignore 文件 ，把忽略的 文件全称 写入到 该文件中即可。

     touch   .gitignore  创建一个  .gitignore文件

4、添加好后的文件，需要提交到git的服务器上

  git  commit  -m"日志"   日志必填

5、添加好的文件 如果被修改  ，先  做add操作  ，再 commit  操作

6、git log   查看日志信息

github   git的服务提供商  通过GitHub可以上传项目  项目可以通过github来管理

git  clone  xxxx   克隆github上的项目到本地文件夹

本地操作项目修改后，将修改的新项目重新推送到远端github ： git push -v origin master

如果在github的线上做项目的修改，线下同步：  1--重新克隆      2 ---   在当前的项目目录下，使用命令：  git  pull origin master

1、gulp

1、安装

先安装node.js

通过 node -v   查看node安装版本

安装gulp：

    1、先全局安装  ：   npm  install gulp -g

    2、创建gulp目录项目 ， 通过 cd  目录名  进入到该目录

    3、在gulp 项目目录下局部安装gulp  ：    npm install gulp --save-dev

    4、配置package.json文件 ：   npm  init （插件安装）

2、什么是gulp？

     gulp --- 前端自动化工具

     作用（插件功能）：

          自动压缩js文件、css文件

          自动压缩图片文件

          自动合并文件

          ......

3、gulp中的方法

     task()   布置任务

     三个参数：

     第一个参数：  任务名称   默认任务  default

     第二个参数：  该任务依赖的其它任务   是一个数组 （可选）

     第三个参数：  任务回调函数 （任务执行）

执行其它任务的方式有两种：

     第一种：通过default任务依赖执行

     第二种： gulp  任务名称

src()  源文件路径    （参数可以是一个数组）

dest() 目标文件路径   dest参数中的文件夹名称可以自动创建

pipe() 管道方法  表示输送  也就是 “下一步”

watch() 监听

两个参数：

     第一个参数： 监听文件的路径

     第二个参数： 监听的任务名称   数组

4、gulp 插件

gulp-sass    将sass文件转成css文件

gulp-cssmin  压缩css文件

gulp-uglify   压缩js文件

gulp-concat   合并文件

gulp-rename   重命名

gulp-imagemin  压缩图片文件

.....

插件安装：  npm install   --save-dev   插件名称

批量安装  ：  将某个已安装好的package.json文件复制到gulp项目目录下   ： 运动  npm install --save-dev

2、什么是模块化

模块化开发 ，可以一次性引入多个模块文件 ，同时也没有引入的顺序

require.js加载的模块，采用AMD规范。也就是说，模块必须按照AMD的规定来写。

具体来说，就是模块必须采用特定的define()函数来定义。如果一个模块不依赖其他模块，那么可以直接定义在define()数之中。函

<script src="jquery.js">

<script src="ajax.js">

<script src="public.js">

3、如何实现模块化开发（requirejs）

第一步：

     在页面中引入：<script src="require.js" data-main="main"></script>

                 data-main的值是一个入口文件   main 实际上就是 main.js

第二步： 入口函数实现：

     通过config方法为模块指定别名

requirejs.config({  //为引入的模块指定别名

     paths:{

           jquery:"jquery-1.11.1.min"   //为jquery库文件指定一个别名，方便后期调用

     }

})

第三步： 通过requirejs方法将写好的模块引入,并根据他们编写子代码

4、requirejs模块化开发常用的方法

1.requirejs.config为模块指定别名,方便模块的引用

2.requirejs将写好的模块引入,并根据他们编写子代码

有两个参数

第一个参数是一个数组：引入的每个模块名称，

第二个参数是一个回调函数，函数中就是页面要实现某个功能的子代码

3.define用它编写模块（功能），在相应的地方进行引入

有两个参数

第一个参数是该模块要依赖的其它模块，是一个数组  （可以省略）

第二个参数是一个回调函数  函数中实现该模块的功能

4、关于requirejs（模块化开发）好处

1、整个项目中一般不会出现全局变量（防止全局变量的污染）

2、整个项目中所有模块的项目加载顺序不固定（顺序任意）

3、整个项目中模块之间的执行是异步的

4、整个项目实现代码均是面向对象的书写方式，便于模块的管理和维护

关于AMD和CMD

AMD：  requirejs所倡导的就是AMD开发方式

     是"Asynchronous Module Definition"的缩写，意思就是"异步模块定义"。它采用异步方式加载模块，模块的加载不影响它后面语句的运行。所有依赖这个模块的语句，都定义在一个回调函数中，等到加载完成之后，这个回调函数才会运行。

AMD推崇的是依赖前置

CMD： Seajs 所倡导的都是CMD模块定义规范。 所有的JavaScript都遵循这个规范。

     CMD（Common Module Definition）   该规范明确了模块的基本书写格式和基本交互规则。

CMD推崇的是依赖就近

-----------------------------------------------

seajs注意：

注意jquery文件的内容要被下面的文件包起来

define(function(){

     此处是jquery的封装代码

     return $.noConflict();

})

seaJs中 ：  通过  use方法引入入口文件

--------------------------------

和require的写法区别

<script src="sea.js"></script>

<script type="text/javascript">

     seajs.use(["main2"])  入口函数的导入

</script>

--------------------------------------

其它模块的引入

define(function(require){

     $ = require("jquery");

     $("body").css("background","red")

})

-----------------------------------

功能定义：

define(function(require,exports,module){

     var obj = {

          "isEqual":function(a,b){

              return a==b;

          }

     }

     module.exports = obj;   模块的暴露

})