# 课程介绍

* 1. 动态网页和静态网页;（了解）
* 2. WSGI接口规范;（了解）
* 3. 实现动态web服务器;（了解）

1. 动态网页和静态网页
   1. 静态网页

静态网页是指存放在服务器文件系统中实实在在的HTML文件。当用户在浏览器中输入页面的URL，然后回车，浏览器就会将对应的html文件下载、渲染并呈现在窗口中。早期的网站通常都是由静态页面制作的。

静态网页，相对比较简单，但内容固定，交互性差，无法充分支持客户自定义需求（外观选择，cookie）

* 1. 动态网页

动态网页是相对于静态网页而言的。当浏览器请求服务器的某个页面时，服务器根据当前时间、环境参数、数据库操作等动态的生成HTML页面，然后在发送给浏览器（后面的处理就跟静态网页一样了）。很明显，动态网页中的“动态”是指服务器端页面的动态生成，相反，“静态”则指页面是实实在在的、独立的文件。

动态网页一般以数据库技术为基础

采用动态网页技术的网站可以实现更多的功能，如用户注册、用户登录、在线调查、用户管理、订单管理等等；

动态网页实际上并不是独立存在于服务器上的网页文件，只有当用户请求时服务器才返回一个完整的网页；

* 1. 误区

会动的就是动态页面，比如通过gif, flash技术这些。

1. WSGI接口
   1. 为什么需要WSGI

一个动态网站在实际部署时，往往采用的方案是：

* 部署一个Web服务器专门用来处理HTTP协议层面相关的事情，
* 部署一个用各种语言编写（Java, PHP, Python, Ruby等）的应用程序，这个应用程序会从Web服务器上接收客户端的请求，处理完成后，再返回响应给Web服务器，最后由Web服务器返回给客户端。

要采用这种方案，Web服务器和应用程序之间就要知道如何进行交互。为了定义Web服务器和应用程序之间的交互过程，就形成了很多不同的规范。这种规范里最早的一个是CGI，1993年开发的。

后来又出现了很多这样的规范。比如改进CGI性能的FasgCGI，Java专用的Servlet规范，还有Python专用的**WSGI规范**等。提出这些规范的目的就是为了定义统一的标准，提升程序的可移植性。

在WSGI规范的最开始的PEP-333中一开始就描述了为什么需要WSGI规范。

* 1. 什么是WSGI

WSGI（Web Server Gateway Interface）**Web服务器网关接口**，是为Python语言定义的Web服务器和Web应用程序或框架之间的一种简单而通用的接口。

WSGI不是服务器，不是API，不是Python模块，更不是什么框架，而是一种服务器和动态应用程序交互的接口规范。这个规范的目的是：

1、让Web服务器知道如何调用Python应用程序，并且把用户的请求告诉应用程序。

2、让Python应用程序知道用户的具体请求是什么后，进行业务处理后如何返回结果给Web服务器。

* 1. WSGI的规范

为了更好说明WSGI规范，定义下面的名词：



web服务器端称之为Server或者Gateway；

动态生成数据处理的业务模块称之为application或framework，比如后面要学习的django框架。

WSGI规定每个application都要有一个接口，这个接口要求：

2个输入参数，1个返回值的接口（此接口可以是函数，也可以是具备\_\_call\_\_接口的类对象）

def simple\_app(environ, start\_response):

pass

* + 1. 两个输入参数

第一个参数必须是字典类型，目的是存放web服务器收到的用户请求信息。

第二个参数是一个回调函数，目的是让application能够反向更新web服务器的响应头字段。

start\_response也接收2个必选参数，一个可选参数：

第一个是字符串，代表这个请求的响应状态，比如'200 OK'。

第二个是一个列表，里面存放元组类型(header\_name, header\_value)，代表响应字段。

* + 1. 一个返回值

返回一个可迭代对象，web服务器通过迭代这个对象，填充body段。

1. 实现动态web服务器
   1. 项目目的

* 熟悉HTTP的请求头和响应头的python解析方式，复习python基础数据结构的使用。
* 掌握文本流和字节流的使用，实际编程设计中，如何避免使用混乱。
* 复习HTTP响应字段Content-Length和Content-Type的使用及作用。
* 了解动态网站WSGI标准，及接口设计方式。
  1. 项目要求

设定特定的URL地址来访问静态文件。

若本机浏览器支持pdf，Word，Excel文件的预览功能，实现这类文件的预览功能。

可以访问当前系统时间。

可以随机生成双色球或点名系统。

1. 课程总结
   1. 重点
      * 1. 字节流和字符流的区别和使用；
        2. HTTP请求协议头的解析；
        3. HTTP响应协议头的构造；
   2. 难点
      * 1. WSGI规范
        2. 回调函数的使用场景
   3. 如何掌握？
      * 1. 利用chrome的网络监视功能分析；
        2. 画清楚流程图；
2. 课后练习
   * + 1. 临摹web服务器代码
3. 面试题
4. 扩展知识或课外阅读推荐（可选）