Python工程师面试题集合

2016/01/17 · [实践项目](http://python.jobbole.com/category/project/) · [1 评论](http://python.jobbole.com/84153/#article-comment)· [面试](http://python.jobbole.com/tag/%e9%9d%a2%e8%af%95/)

分享到：**6**

原文出处： [刘元涛](http://blog.csdn.net/u013510614/article/details/50509387)

**Python 基础语法相关**

**Python中pass语句的作用是什么？**

pass语句什么也不做，一般作为占位符或者创建占位程序，pass语句不会执行任何操作。

**Python是如何进行类型转换的？**

Python提供了将变量或值从一种类型转换成另一种类型的内置函数。比如int函数能够将符合数学格式数字型字符串转换成整数。否则，返回错误信息。

**Python是如何进行内存管理的？**

Python引用了一个内存池(memory pool)机制，即Pymalloc机制(malloc:n.分配内存)，用于管理对小块内存的申请和释放。

**dict 的 items() 方法与 iteritems() 方法的不同？**

items方法将所有的字典以列表方式返回，其中项在返回时没有特殊的顺序；

iteritems方法有相似的作用，但是返回一个迭代器对象

**什么是lambda函数？它有什么好处?**

编程中提到的 lambda 表达式，通常是在需要一个函数，但是又不想费神去命名一个函数的场合下使用，也就是指匿名函数。

Python允许你定义一种单行的小函数。定义lambda函数的形式如下：labmda 参数：表达式lambda函数默认返回表达式的值。你也可以将其赋值给一个变量。lambda函数可以接受任意个参数，包括可选参数，但是表达式只有一个。

**说明os,sys模块不同，并列举常用的模块方法？**

官方解释：

os： This module provides a portable way of usingoperating system dependent functionality.

翻译：提供一种方便的使用操作系统函数的方法。

sys：This module provides access to some variablesused or maintained by the interpreter and to functions that interact stronglywith the interpreter.

翻译：提供访问由解释器使用或维护的变量和在与解释器交互使用到的函数。

os 常用方法



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | os.remove()删除文件  os.rename()重命名文件  os.walk()生成目录树下的所有文件名  os.chdir()改变目录  os.mkdir/makedirs创建目录/多层目录  os.rmdir/removedirs删除目录/多层目录  os.listdir()列出指定目录的文件  os.getcwd()取得当前工作目录  os.chmod()改变目录权限  os.path.basename()去掉目录路径，返回文件名  os.path.dirname()去掉文件名，返回目录路径  os.path.join()将分离的各部分组合成一个路径名  os.path.split()返回（dirname(),basename())元组  os.path.splitext()(返回filename,extension)元组  os.path.getatime\ctime\mtime分别返回最近访问、创建、修改时间  os.path.getsize()返回文件大小  os.path.exists()是否存在  os.path.isabs()是否为绝对路径  os.path.isdir()是否为目录  os.path.isfile()是否为文件 |

sys 常用方法



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | sys.argv           命令行参数List，第一个元素是程序本身路径  sys.modules.keys() 返回所有已经导入的模块列表  sys.exc\_info()     获取当前正在处理的异常类,exc\_type、exc\_value、exc\_traceback当前处理的异常详细信息  sys.exit(n)        退出程序，正常退出时exit(0)  sys.hexversion     获取Python解释程序的版本值，16进制格式如：0x020403F0  sys.version        获取Python解释程序的版本信息  sys.maxint         最大的Int值  sys.maxunicode     最大的Unicode值  sys.modules        返回系统导入的模块字段，key是模块名，value是模块  sys.path           返回模块的搜索路径，初始化时使用PYTHONPATH环境变量的值  sys.platform       返回操作系统平台名称  sys.stdout         标准输出  sys.stdin          标准输入  sys.stderr         错误输出  sys.exc\_clear()    用来清除当前线程所出现的当前的或最近的错误信息  sys.exec\_prefix    返回平台独立的python文件安装的位置  sys.byteorder      本地字节规则的指示器，big-endian平台的值是'big',little-endian平台的值是'little'  sys.copyright      记录python版权相关的东西  sys.api\_version    解释器的C的API版本  sys.version\_info |

**Python里面如何拷贝一个对象？deepcopy 和 copy的区别？**

copy 仅拷贝对象本身，而不拷贝对象中引用的其它对象。

deepcopy 除拷贝对象本身，而且拷贝对象中引用的其它对象。

**os.path和sys.path的区别？**

os.path是module，包含了各种处理长文件名(路径名)的函数。

sys.path是由目录名构成的列表，Python 从中查找扩展模块( Python 源模块, 编译模块,或者二进制扩展). 启动 Python 时,这个列表从根据内建规则,PYTHONPATH 环境变量的内容, 以及注册表( Windows 系统)等进行初始化.

**re模块中match和search方法的不同？**

match() 函数只检查 RE 是否在字符串开始处匹配，而search() 则是扫描整个字符串。

**解释生成器(generator)与函数的不同，并实现和使用简单generator？**

生成器和函数的主要区别在于函数 return avalue，生成器 yield a value同时标记或记忆point of the yield 以便于在下次调用时从标记点恢复执行。 yield 使函数转换成生成器，而生成器反过来又返回迭代器。

**Python Web相关**

**解释一下 WSGI 和 FastCGI 的关系？**

**CGI**全称是“公共网关接口”(CommonGateway Interface)，HTTP服务器与你的或其它机器上的程序进行“交谈”的一种工具，其程序须运行在网络服务器上。　CGI可以用任何一种语言编写，只要这种语言具有标准输入、输出和环境变量。如php,perl,tcl等。

**FastCGI**像是一个常驻(long-live)型的CGI，它可以一直执行着，只要激活后，不会每次都要花费时间去fork一次(这是CGI最为人诟病的fork-and-execute模式)。它还支持分布式的运算, 即 FastCGI 程序可以在网站服务器以外的主机上执行并且接受来自其它网站服务器来的请求。

**FastCGI**是语言无关的、可伸缩架构的CGI开放扩展，其主要行为是将CGI解释器进程保持在内存中并因此获得较高的性能。众所周知，CGI解释器的反复加载是CGI性能低下的主要原因，如果CGI解释器保持在内存中并接受FastCGI进程管理器调度，则可以提供良好的性能、伸缩性、Fail- Over特性等等。

**WSGI**的全称为： PythonWeb Server Gateway Interface v1.0 （Python Web 服务器网关接口），

它是 Python 应用程序和 WEB 服务器之间的一种接口。

它的作用，类似于FCGI 或 FASTCGI 之类的协议的作用。

WSGI 的目标，是要建立一个简单的普遍适用的服务器与 WEB 框架之间的接口。

**Flup**就是使用 Python 语言对 WSGI 的一种实现，是可以用于 Python 的应用开发中的一种工具或者说是一种库。

Spawn-fcgi是一个小程序，这个程序的作用是管理fast-cgi进程，那么管理wsgi进程也是没有问题的，功能和php-fpm类似。

故，简单地说，WSGI和FastCGI都是一种CGI，用于连接WEB服务器与应用程序，而WSGI专指Python应用程序。而flup是WSGI的一种实现，Spawn-fcgi是用于管理flup进程的一个工具，可以启动多个wsgi进程，并管理它们。

**解释一下 Django 和 Tornado 的关系、差别**

**Django**源自一个在线新闻 Web站点，于 2005 年以开源的形式被释放出来。

Django 框架的核心组件有：

用于创建模型的对象关系映射为最终用户设计的完美管理界面一流的 URL 设计设计者友好的模板语言缓存系统等等

它鼓励快速开发,并遵循MVC设计。Django遵守 BSD版权，最新发行版本是Django

1.4，于2012年03月23日发布.Django的主要目的是简便、快速的开发数据库驱动的网站。它强调代码复用,多个组件可以很方便的以“插件”形式服务于整个框架，Django有许多功能强大的第三方插件，你甚至可以很方便的开发出自己的工具包。这使得Django具有很强的可扩展性。它还强调快速开发和DRY(Do Not RepeatYourself)原则。

**Tornado**是 FriendFeed使用的可扩展的非阻塞式 web 服务器及其相关工具的开源版本。这个 Web 框架看起来有些像 web.py 或者 Google 的 webapp，不过为了能有效利用非阻塞式服务器环境，这个 Web 框架还包含了一些相关的有用工具和优化。

Tornado 和现在的主流 Web 服务器框架（包括大多数Python 的框架）有着明显的区别：它是非阻塞式服务器，而且速度相当快。得利于其 非阻塞的方式和对epoll的运用，Tornado 每秒可以处理数以千计的连接，这意味着对于实时 Web服务来说，Tornado 是一个理想的 Web 框架。我们开发这个 Web 服务器的主要目的就是为了处理 FriendFeed 的实时功能 ——在 FriendFeed 的应用里每一个活动用户都会保持着一个服务器连接。（关于如何扩容 服务器，以处理数以千计的客户端的连接的问题。

**解释下django-debug-toolbar的使用**

使用django开发站点时，可以使用django-debug-toolbar来进行调试。在settings.py中添加’debug\_toolbar.middleware.DebugToolbarMiddleware’到项目的MIDDLEWARE\_CLASSES 内。

**解释下Django使用redis缓存服务器**

为了能在Django中使用redis，还需要安装redis for Django的插件。然后在Django的settings中配置了。现在连接和配置都已经完成了，接下来是一个简单的例子：



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | from django.conf import settings  from django.core.cache import cache  #read cache user id  def read\_from\_cache(self, user\_name):      key = 'user\_id\_of\_'+user\_name      value = cache.get(key)      if value == None:          data = None      else:          data = json.loads(value)      return data  #write cache user id  def write\_to\_cache(self, user\_name):      key = 'user\_id\_of\_'+user\_name      cache.set(key, json.dumps(user\_name), settings.NEVER\_REDIS\_TIMEOUT) |

**如何进行Django单元测试**

**Django的单元测试使用python的unittest模块，这个模块使用基于类的方法来定义测试。类名为django.test.TestCase,继承于python的unittest.TestCase。**



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | from django.test import TestCase  from myapp.models import Animal    class AnimalTestCase(TestCase):      def setUp(self):          Animal.objects.create(name="lion", sound="roar")          Animal.objects.create(name="cat", sound="meow")        def test\_animals\_can\_speak(self):          """Animals that can speak are correctly identified"""          lion = Animal.objects.get(name="lion")          cat = Animal.objects.get(name="cat")          self.assertEqual(lion.speak(), 'The lion says "roar"')          self.assertEqual(cat.speak(), 'The cat says "meow"') |

执行目录下所有的测试(所有的test\*.py文件)：运行测试的时候，测试程序会在所有以test开头的文件中查找所有的test cases(inittest.TestCase的子类),自动建立测试集然后运行测试。



|  |  |
| --- | --- |
| 1 | $ python manage.py test |

执行animals项目下tests包里的测试：



|  |  |
| --- | --- |
| 1 | $ python manage.py testanimals.tests |

执行animals项目里的test测试：



|  |  |
| --- | --- |
| 1 | $ python manage.py testanimals |

单独执行某个test case：



|  |  |
| --- | --- |
| 1 | $ python manage.py testanimals.tests.AnimalTestCase |

单独执行某个测试方法：



|  |  |
| --- | --- |
| 1 | $ python manage.py testanimals.tests.AnimalTestCase.test\_animals\_can\_speak |

为测试文件提供路径：



|  |  |
| --- | --- |
| 1 | $ python manage.py testanimals/ |

通配测试文件名：



|  |  |
| --- | --- |
| 1 | $ python manage.py test--pattern="tests\_\*.py" |

启用warnings提醒：



|  |  |
| --- | --- |
| 1 | $ python -Wall manage.py test |

**解释下Http协议**

HTTP是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。

HTTP协议的主要特点可概括如下：

1.支持客户/服务器模式。

2.简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。

3.灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。

4.无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。

5.无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

**解释下Http请求头和常见响应状态码**

Accept:指浏览器或其他客户可以接爱的MIME文件格式。可以根据它判断并返回适当的文件格式。

Accept-Charset：指出浏览器可以接受的字符编码。英文浏览器的默认值是ISO-8859-1.

Accept-Language：指出浏览器可以接受的语言种类，如en或en-us，指英语。

Accept-Encoding：指出浏览器可以接受的编码方式。编码方式不同于文件格式，它是为了压缩文件并加速文件传递速度。浏览器在接收到Web响应之后先解码，然后再检查文件格式。

Cache-Control：设置关于请求被代理服务器存储的相关选项。一般用不到。

Connection：用来告诉服务器是否可以维持固定的HTTP连接。HTTP/1.1使用Keep-Alive为默认值，这样，当浏览器需要多个文件时(比如一个HTML文件和相关的图形文件)，不需要每次都建立连接。

Content-Type：用来表名request的内容类型。可以用HttpServletRequest的getContentType()方法取得。

Cookie：浏览器用这个属性向服务器发送Cookie。Cookie是在浏览器中寄存的小型数据体，它可以记载和服务器相关的用户信息，也可以用来实现会话功能。

状态代码有三位数字组成，第一个数字定义了响应的类别，且有五种可能取值：

1xx：指示信息–表示请求已接收，继续处理

2xx：成功–表示请求已被成功接收、理解、接受

3xx：重定向–要完成请求必须进行更进一步的操作

4xx：客户端错误–请求有语法错误或请求无法实现

5xx：服务器端错误–服务器未能实现合法的请求

常见状态代码、状态描述、说明：

200 OK     //客户端请求成功

400 Bad Request  //客户端请求有语法错误，不能被服务器所理解

401 Unauthorized //请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用

403 Forbidden  //服务器收到请求，但是拒绝提供服务

404 Not Found  //请求资源不存在，eg：输入了错误的URL

500 Internal Server Error //服务器发生不可预期的错误

503 Server Unavailable  //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

eg：HTTP/1.1 200 OK （CRLF）