

Vamei

编程, 数学, 设计

[博客园](#)[首页](#)[订阅](#)[管理](#)[随笔-209](#) [文章-1](#) [评论-3802](#)

Python简史

作者: Vamei 出处: <http://www.cnblogs.com/vamei> 欢迎转载, 也请保留这段声明。谢谢!

Python是我喜欢的语言, 简洁, 优美, 容易使用。前两天, 我很激昂的向朋友宣传Python的好处。

听过之后, 朋友问我: 好吧, 我承认Python不错, 但它为什么叫Python呢?

我不是很确定: 呃, 似乎是一个电视剧的名字。

朋友又问: 那你说的Guido是美国人么? (Guido von Rossum, Python的作者)

我再次不是很确定: 他从google换到Dropbox工作, 但他的名字像是荷兰人的 (有一个von在中间)。

所以, 后面我花了些时间调查Python的历史。这是很好的学习。我看到了Python中许多功能的来源和Python的设计理念, 比如哪些功能是历史遗留, 哪些功能是重复, 如何增加功能..... 而且, Python也是开源(open source)运动的一个成功案例。从Python的历史中, 我们可以一窥开源开发的理念和成就。

这也可以作为我写的[Python快速教程](#)的序篇。

Python的起源

Python的作者, Guido von Rossum, 确实是荷兰人。1982年, Guido从阿姆斯特丹大学(University of Amsterdam)获得了数学和计算机硕士学位。然而, 尽管他算得上是一位数学家, 但他更加享受计算机带来的乐趣。用他的话说, 尽管拥有数学和计算机双料资质, 他总趋向于做计算机相关的工作, 并热衷于做任何和编程相关的活儿。



Guido von Rossum

在那个时候，他接触并使用过诸如Pascal、C、Fortran等语言。这些语言的基本设计原则是让机器能更快运行。在80年代，虽然IBM和苹果已经掀起了个人电脑浪潮，但这些个人电脑的配置很低（在今天看来）。比如早期的Macintosh，只有8MHz的CPU主频和128KB的RAM，一个大的数组就能占满内存。所有的编译器的核心是做优化，以便让程序能够运行。为了增进效率，语言也迫使程序员像计算机一样思考，以便能写出更符合机器口味的程序。在那个时代，程序员恨不得用手榨取计算机每一寸的能力。有人甚至认为C语言的指针是在浪费内存。至于动态类型，内存自动管理，面向对象..... 别想了，那会让你的电脑陷入瘫痪。

然而，这种思考方式让Guido感到苦恼。Guido知道如何用C语言写出一个功能，但整个编写过程需要耗费大量的时间（即使他已经准确的知道了如何实现）。他的另一

个选择是shell。Bourne Shell作为UNIX系统的解释器(interpreter)已经长期存在。UNIX的管理员们常常用shell去写一些简单的脚本,以进行一些系统维护的工作,比如定期备份、文件系统管理等等。shell可以像胶水一样,将UNIX下的许多功能连接在一起。许多C语言下上百行的程序,在shell下只用几行就可以完成。然而,shell的本质是调用命令。它并不是一个真正的语言。比如说,shell没有数值型的数据类型,加法运算都很复杂。总之,shell不能全面的调动计算机的功能。

(关于shell,你可以参考[Linux架构](#)和[Linux命令行与命令](#))

Guido希望有一种语言,这种语言能够像C语言那样,能够全面调用计算机的功能接口,又可以像shell那样,可以轻松的编程。ABC语言让Guido看到希望。ABC是由荷兰的CWI (Centrum Wiskunde & Informatica, 数学和计算机研究所)开发的。Guido在CWI工作,并参与到ABC语言的开发。ABC语言以教学为目的。与当时的大部分语言不同,ABC语言的目标是“**让用户感觉更好**”。ABC语言希望让语言变得**容易阅读,容易使用,容易记忆,容易学习**,并以此来激发人们学习编程的兴趣。比如下面是一段来自Wikipedia的ABC程序,这个程序用于统计文本中出现的词(word)的总数:



```
HOW TO RETURN words document:
  PUT {} IN collection
  FOR line IN document:
    FOR word IN split line:
      IF word not.in collection:
        INSERT word IN collection
  RETURN collection
```

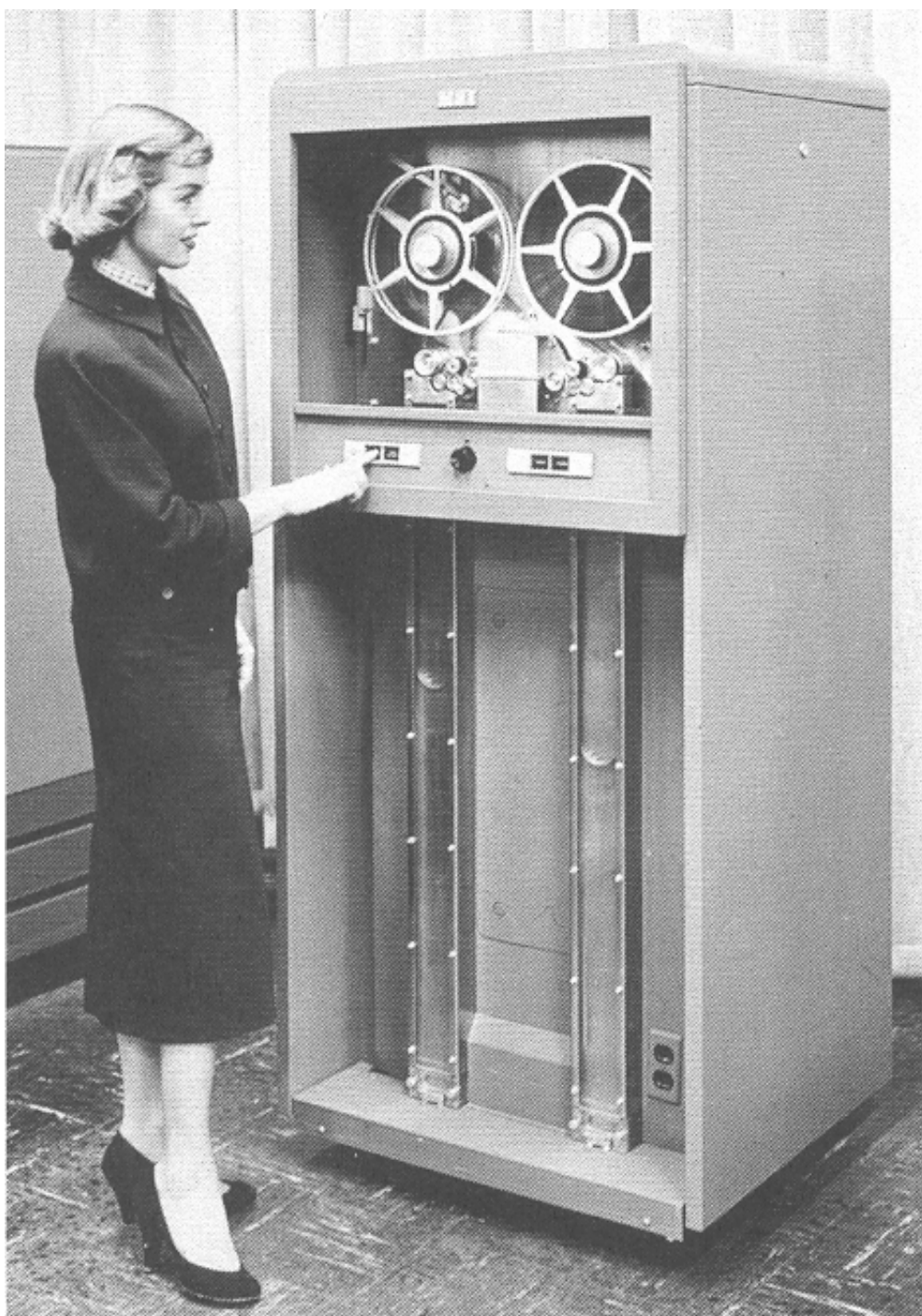


HOW TO用于定义一个函数。一个Python程序员应该很容易理解这段程序。ABC语言使用冒号(:)和缩进来表示程序块(C语言使用{}来表示程序块)。行尾没有分号。for和if结构中也没有括号()。如果将HOW TO改为def,将PUT行改为collection = [],将INSERT行改为collection.append(word),这就几乎是一个标准的Python函数。上面的函数读起来就像一段自然的文字。

尽管已经具备了良好的可读性和易用性,ABC语言最终没有流行起来。在当时,ABC语

言编译器需要比较高配置的电脑才能运行。而这些电脑的使用者通常精通计算机，他们更多考虑程序的效率，而非它的学习难度。除了硬件上的困难外，ABC语言的设计也存在一些致命的问题：

- 可拓展性差。ABC语言不是模块化语言。如果想在ABC语言中增加功能，比如对图形化的支持，就必须改动很多地方。
- 不能直接进行IO。ABC语言不能直接操作文件系统。尽管你可以通过诸如文本流的方式导入数据，但ABC无法直接读写文件。输入输出的困难对于计算机语言来说是致命的。你能想像一个打不开车门的跑车么？
- 过度革新。ABC用自然语言的方式来表达程序的意义，比如上面程序中的HOW TO (如何)。然而对于程序员来说，他们更习惯用function或者define来定义一个函数。同样，程序员也习惯了用等号(=)来分配变量。这尽管让ABC语言显得特别，但实际上增加了程序员的学习难度（程序员大都掌握不止一种语言）。
- 传播困难。ABC编译器很大，必须被保存在磁带(tape)上。当时Guido在访问的时候，就必须有一个大磁带来给别人安装ABC编译器。这样，ABC语言就很难快速传播。



IBM tape drive: 读写磁带

1989年，为了打发圣诞节假期，Guido开始写Python语言的编译/解释器。Python来自Guido所挚爱的电视剧Monty Python's Flying Circus (BBC1960-1970年代播放的室内情景幽默剧，以当时的英国生活为素材)。他希望这个新的叫做Python的语言，能实现他的理念(一种C和shell之间，功能全面，易学易用，可拓展的语言)。Guido作为一个语言设计爱好者，已经有过设计语言的(不很成功)的尝试。这一次，也不过是一次纯粹的hacking行为。

Python的诞生

1991年，第一个Python编译器(同时也是解释器)诞生。它是用C语言实现的，并能够调用C库(.so文件)。从一出生，Python已经具有了：类(class)，函数(function)，异常处理(exception)，包括表(list)和词典(dictionary)在内的核心数据类型，以及模块(module)为基础的拓展系统。



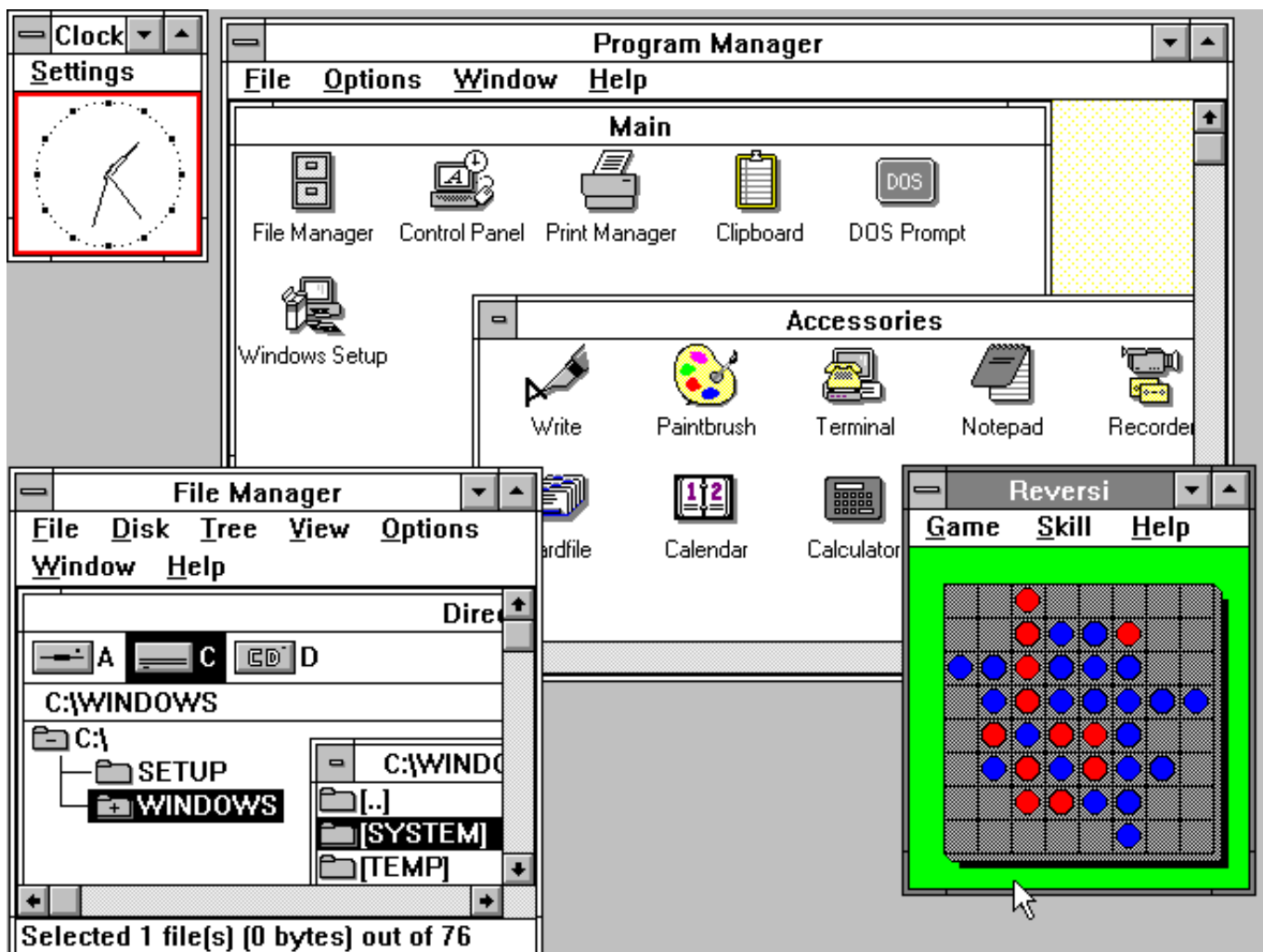
最初的Python logo：由Guido的兄弟Just von Rossum设计

Python语法很多来自C，但又受到ABC语言的强烈影响。来自ABC语言的一些规定直到今天还富有争议，比如强制缩进。但这些语法规则让Python**容易读**。另一方面，Python聪明的选择服从一些惯例(特别是C语言的惯例)。比如使用等号赋值，使用def来定义函数。Guido认为，如果“常识”上确立的东西，没有必要过度纠结。

Python从一开始就特别在意**可拓展性**(extensibility)。Python可以在多个层次上拓展。从高层上，你可以引入.py文件。在底层，你可以引用C语言的库。Python程序员可以快速的使用Python写.py文件作为拓展模块。但当性能是考虑的重要因素时，Python程序员可以深入底层，写C程序，编译为.so文件引入到Python中使用。Python就好像是使用钢构建房一样，先规定好大的框架。而程序员可以在此框架下相当自由的拓展或更改。

最初的Python完全由Guido本人开发。Python得到Guido同事的欢迎。他们迅速的反馈使用意见，并参与到Python的改进。Guido和一些同事构成Python的核心团队。他们将自己大部分的业余时间用于hack Python(也包括工作时间，因为他们将Python用于工作)。随后，Python拓展到CWI之外。Python将许多机器层面的细节隐藏，交给编译器处理，并凸显出逻辑层面的编程思考。Python程序员可以花更多的时间用于思考程序的逻辑，而不是具体的实现细节(Guido有一件T恤，写着：人生苦短，我用Python)。这一特征吸引了广大的程序员。Python开始流行。

我们不得不暂停我们的Python时间，转而看一看这时的计算机概况。1990年代初，个人计算机开始进入普通家庭。Intel发布了486处理器，windows发布window 3.0开始的一系列视窗系统。计算机的性能大大提高。程序员开始关注计算机的易用性(比如图形化界面)。



Windows 3.0

由于计算机性能的提高，软件的世界也开始随之改变。硬件足以满足许多个人电脑的需要。硬件厂商甚至渴望高需求软件的出现，以带动硬件的更新换代。C++和Java相继流行。C++和Java提供了面向对象的编程范式，以及丰富的对象库。在牺牲了一定的性能的代价下，C++和Java大大提高了程序的产量。语言的易用性被提到一个新的高度。我们还记得，ABC失败的一个重要原因是硬件的性能限制。从这方面说，Python要比ABC幸运许多。

另一个悄然发生的改变是Internet。1990年代还是个人电脑的时代，windows和Intel挟PC以令天下，盛极一时。尽管Internet为主体的信息革命尚未到来，但许多程序员以及资深计算机用户已经在频繁使用Internet进行交流（包括email和newsgroup）。Internet让信息交流成本大大下降。一种新的软件开发模式开始流行：开源（open source）。程序员利用业余时间进行软件开发，并开放源代码。1991年，Linus在comp.os.minix新闻组上发布了Linux内核源代码，吸引大批hacker的加入。Linux和GNU相互合作，最终构成了一个充满活力的开源平台。

硬件性能不是瓶颈，Python又容易使用，所以许多人开始转向Python。Guido维护

了一个maillist, Python用户就通过邮件进行交流。Python用户来自许多领域, 有不同的背景, 对Python也有不同的需求。Python相当的开放, 又容易拓展, 所以当用户不满足于现有功能, 很容易对Python进行拓展或改造。随后, 这些用户将改动发给Guido, 并由Guido决定是否将新的特征加入到Python或者标准库中。如果代码能被纳入Python自身或者标准库, 这将极大的荣誉。Python自身也因此变得更好。

(Guido不得不作出许多决定, 这也是他被称为Benevolent Dictator For Life的原因)

Python被称为“Battery Included”, 是说它以及其标准库的功能强大。这些是整个社区的贡献。Python的开发者来自不同领域, 他们将不同领域的优点带给Python。比如Python标准库中的正则表达(regular expression)是参考Perl, 而lambda, map, filter, reduce函数参考Lisp。Python本身的一些功能以及大部分的标准库来自于社区。Python的社区不断扩大, 进而拥有了自己的newsgroup, 网站(python.org), 以及基金 (Python Software Foundation)。从Python 2.0开始, Python也从maillist的开发方式, 转为完全开源的开发方式。社区气氛已经形成, 工作被整个社区分担, Python也获得了更加高速的发展。

(由于Guido享有绝对的仲裁权, 所以在Python早期maillist的开发时代, 不少爱好者相当担心Guido的生命。他们甚至作出假设: 如果Guido挂了的话, Python会怎样。见If Guido was hit by a bus)

到今天, Python的框架已经确立。Python语言以对象为核心组织代码 (Everything is object), 支持多种编程范式 (multi-paradigm), 采用动态类型 (dynamic typing), 自动进行内存回收 (garbage collection)。Python支持解释运行 (interpret), 并能调用C库进行拓展。Python有强大的标准库 (battery included)。由于标准库的体系已经稳定, 所以Python的生态系统开始拓展到第三方包。这些包, 如Django, web.py, wxpython, numpy, matplotlib, PIL, 将Python升级成了物种丰富的热带雨林。

今天Python已经进入到3.0的时代。由于Python 3.0向后不兼容, 所以从2.0到3.0的过渡并不容易。另一方面, Python的性能依然值得改进, Python的运算性能低于C++和Java (见Google的讨论)。Python依然是一个在发展中的语言。我期待看到Python的未来。

Python启示录

Python崇尚优美、清晰、简单，是一个优秀并广泛使用的语言（TIOBE语言排行第八，Google的第三大开发语言，Dropbox的基础语言，豆瓣的服务器语言）。这个世界并不缺乏优秀的语言，但Python的发展史作为一个代表，带给我许多启示。

在Python的开发过程中，社区起到了重要的作用。Guido自认为自己不是全能型的程序员，所以他只负责制订框架。如果问题太复杂，他会选择绕过去，也就是`cut the corner`。这些问题最终由社区中的其他人解决。社区中的人才是异常丰富的，就连创建网站，筹集基金这样与开发稍远的事情，也有人乐意于处理。如今的项目开发越来越复杂，越来越庞大，合作以及开放的心态成为项目最终成功的关键。

Python从其他语言中学到了很多，无论是已经进入历史的ABC，还是依然在使用的C和Perl，以及许多没有列出的其他语言。可以说，Python的成功代表了它所有借鉴的語言的成功。同样，Ruby借鉴了Python，它的成功也代表了Python某些方面的成功。每个语言都是**混合体**，都有它优秀的地方，但也有各种各样的缺陷。同时，一个语言“好与不好”的评判，往往受制于平台、硬件、时代等等外部原因。程序员经历过许多语言之争。我想，为什么不以开放的心态和客观的分析，去区分一下每个语言的具体优点缺点，去区分内部和外部的因素。说不定哪一天发现，我不喜欢的某个语言中，正包含了我所需要的东西。

无论Python未来的命运如何，Python的历史已经是本很有趣的小说。

如果你因为本文对Python产生了兴趣，欢迎阅读我的Python快速教程。

本文主要参考：

Guido在Dropbox所做演讲

http://v.youku.com/v_show/id_XNTExOTc1NTU2.html

[python.org](#)的文档

[Wikipedia](#)

标签: [Python](#)

好文要顶

关注我

收藏该文



Vamei

关注 - 26

粉丝 - 4985

荣誉: [推荐博客](#)

[+加关注](#)

32

0

(请您对文章做出评价)

« [上一篇: 绘图: matplotlib核心剖析](#)

» [下一篇: Python深入05 装饰器](#)

posted @ 2013-02-06 15:23 Vamei 阅读(28346) 评论(25) 编辑 收藏

评论列表

#1楼 2013-02-06 15:39 竹风抚荷塘

这些历史还是相当有趣的^_^

支持(1) 反对(0)

#2楼[楼主] 2013-02-06 15:41 Vamei

@ 竹风抚荷塘

也很有代表性。

支持(0) 反对(0)

#3楼 2013-02-06 19:29 dudu

原来是在圣诞节假期开发出来的，说明程序员在放松的状态下更能产生好的创意

支持(0) 反对(0)

#4楼 2013-02-06 19:37 ZhouTai

清晰 流畅

支持(0) 反对(0)

#5楼[楼主] 2013-02-06 21:05 Vamei

@ ZhouTai

是指python还是这篇文章？

支持(0) 反对(0)

#6楼[楼主] 2013-02-06 21:06 Vamei

@ dudu

Depends. 我放假就彻底歇息了。

支持(0) 反对(0)

#7楼 2013-02-07 10:00 JeffWong

好玩

支持(0) 反对(0)

#8楼 2013-02-07 10:11 gowk

终于又有Python的文章了

支持(0) 反对(0)

#9楼[楼主] 2013-02-07 10:14 Vamei

@ pythonic

:-)，还是Python的花边文章。

支持(0) 反对(0)

#10楼[楼主] 2013-02-07 12:38 Vamei

@ pandaren

已经发的文章还可以重复发吗？

支持(0) 反对(0)

#11楼 2013-02-07 17:37 搏击的小船

Python很优雅，简明！

支持(0) 反对(0)

#12楼[楼主] 2013-02-07 17:48 Vamei

@ 搏击的小船

好吧，我以为是在说这个文章。

支持(0) 反对(0)

#13楼 2013-02-18 08:31 硬盘很大

@ dudu

说明我们程序员都比较操蛋。

过节没节目。。。

支持(0) 反对(0)

#14楼 2013-02-19 11:16 永远的阿哲

顶！

支持(0) 反对(0)

#15楼 2013-06-19 17:30 superchao

nice

支持(0) 反对(0)

#16楼 2013-06-30 18:56 iab

不喜欢 Python，缩进难看，速度慢。

支持(0) 反对(1)

#17楼 2013-10-02 09:46 Jason Luo

楼主的介绍十分翔实啊！

支持(0) 反对(0)

#18楼 2014-02-20 11:15 特务小强

怎么加入maillist?

支持(0) 反对(0)

#19楼[楼主] 2014-02-20 13:32 Vamei

@ 特务小强

你可以参考这一篇:

<http://www.python.org/community/lists/>

支持(0) 反对(0)

#20楼 2014-03-28 08:45 唐大侠

看了这个才知道它的历史有这么久但是现在的学校大多不会提到这个

支持(0) 反对(0)

#21楼 2014-09-19 13:43 Eason525

我是从Java快速教程开始看您的文章的，现在在自学python，觉得您写的学术方面的东西条理清晰简洁易懂，不做过多的解释，而这种趣闻类的又生动翔实，真的很佩服您！另外求微信公众号联系方式！

支持(0) 反对(0)

#22楼 2015-02-11 07:40 suifengtec

不错

支持(0) 反对(0)

#23楼 2015-02-12 14:06 wangsitan

哈哈，If Guido was hit by a bus.

那T恤我去年也买了一件（因为喜欢python且看到了Guido穿那衣服的照片），在taobao上一个专卖程序员风格衣服的店。

当时一下买了三件衣服，一个“人生苦短，我用Python”，一个openSUSE，一个“Coding. Don't fucking talk to me”

支持(0) 反对(0)

#24楼 2015-08-12 20:24 万里沙来手一挥

图片上IBM的美女挺漂亮

支持(0) 反对(0)

#25楼 2016-04-13 13:48 wqh2016

美女程序员

[支持\(0\)](#) [反对\(0\)](#)[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#)网站首页。

【推荐】50万行VC++源码：大型组态工控、电力仿真CAD与GIS源码库

【推荐】融云即时通讯云—豆果美食、Faceu等亿级APP都在用



GrapeCity
ActiveReports
企业级报表服务平台
单独部署、集成应用、报表制作、数据整合
权限管理、移动办公、二次集成开发
立即了解




JPUSH 极光推送 消息推送领导品牌全面升级 JIGUANG 极光 详情点击

公告

你好，这里是Vamei，一名编程爱好者。我在博客里写了**Python/Linux/网络协议/算法/Java/数据科学**系列文章，从这里开始阅读。非常期待和你的交流。



我的微博

Vamei ：我科研发出#功能性机器人#佳佳，据说以本校5位校花为原型。双...
4月15日 22:33 | 微博
weibo.com/vamei

下列教程已经做成电子出版物，内容经过修订，也方便离线阅读：
[协议森林](#)

欢迎阅读我写的其他书籍：

[现代小城的考古学家](#)

[天气与历史的相爱相杀](#)

随手拍光影

昵称: **Vamei**

园龄: **4年1个月**

荣誉: 推荐博客

粉丝: **4985**

关注: **26**

+加关注

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

Python(61)

Java(42)

大数据(22)

Linux(17)

网络(16)

算法(15)

文青(14)

技普(9)

系列索引(6)

开发工具(4)

更多

系列文章

Java快速教程

Linux的概念与体系

Python快速教程

数据科学

协议森林

纸上谈兵：算法与数据结构

积分与排名

积分 - 659668

排名 - 122

最新评论

1. Re:Java基础11 对象引用

受教!

--MissLost

2. Re:Python快速教程

看评论区一片喝彩！看来我得在此扎营了！

--测试小蚂蚁

3. Re:Python进阶06 循环对象

好好地列表解析变成了表推导

--ashic

4. Re:"不给力啊，老湿！”：RSA加密与破解

感谢楼主精彩分享

--worldball

5. Re:概率论04 随机变量

你写的这一系列太棒了，刚加入博客园就在你这里学到了，我要转载过去学习一下

--yixius

6. Re:Python基础03 序列

挺好的教程、、、

--王小拽的号

7. Re:Python进阶07 函数对象

```
def func(x,y): print x**ydef test(f,a,b): print 'test' print f(a,b)test (func,3,2)
```

输出的内容:tes.....

--M-edea

8. Re:Python进阶02 文本文件的输入输出

@coderXT换行符: \n...

--行者之印

9. Re:数据科学

博主啊，这里是一枚即将大二的计算机新人，大一学了python，java，还有一些算法，数据结构，图论了，感觉我对数学又一些反感，但是听说离散数学对计算机专业的很重要，不知道怎么去学比较好呢，我想像您写.....

--Acokil

10. Re:为什么要写技术博

楼主是用自己自定义的模板吗？在博客园里找不到这种风格的blog模板？

--行者之印

11. Re:来玩Play框架01 简介

挖煤哥,我补充了一下Windows下的搭建play框架,希望有点帮助,谢谢!

--Sungeek

12. Re:来玩Play框架07 静态文件

```
@helper.form(action = routes.Application.upload, 'enctype ->
    "multipart/form-data") {--action = rout.....
```

--quxiaozha

13. Re:来玩Play框架07 静态文件

该记录将/assets/下的URL，对应到项目的/public文件夹内的文件。比如在项目的/public/images/test.jpg，就可以通过/assets/images/test.jpg这一.....

--quxiaozha

14. Re:来玩Play框架06 用户验证

支持挖煤哥~~~

--quxiaozha

15. Re:“不给力啊，老湿！”：RSA加密与破解

@maanshancss请你仔细阅读了这个文章再来评价。...

--Vamei

推荐排行榜

1. “不给力啊，老湿！”：RSA加密与破解(218)
2. Python快速教程(140)
3. 野蛮生长又五年(91)
4. Java快速教程(88)
5. 协议森林01 邮差与邮局 (网络协议概观)(79)
6. 为什么要写技术博(71)
7. 编程异闻录(54)
8. 博客一年：心理之旅(49)
9. 协议森林08 不放弃 (TCP协议与流通信)(45)
10. Python快速教程 尾声(43)
11. 协议森林(42)
12. Java基础01 从HelloWorld到面向对象(42)
13. Python基础08 面向对象的基本概念(40)

14. 一天能学会的计算机技术(34)

15. 博客第二年，杂谈(33)

Copyright ©2016 Vamei

05370355