

1989年发明

 这个是python的特点

如果不做合适的缩进 就会语法错误 ---省去了{} 这样 代码量大大减少

 python编程 很多时候 就像搭积木一样 灵活调用python类库 使用的代码量非常小

 ----- 

很轻松把其他语言的功能整合到一个框架下



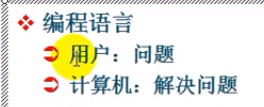
事实上 要想知道python --- 要了解编程语言

编程语言 --- 为什么要有编程语言

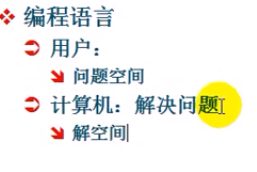
要完成的任务 一定是有些问题 人本身不容易解决 所以 作为使用者是问题

能够帮忙解决问题的是 计算机

---



计算机和我们识别的语言 不大一样



必须有一种方式 把问题空间映射到解空间 ----

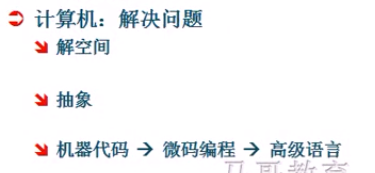
要么人迁就机器 要么机器迁就人 ----- 机器语言太痛苦

讲计算机解决问题的能力 进一步抽象 更接近于人

这样 计算机编程语言就诞生了

最早期的就是机器代码 过于底层

机器代码在芯片级别的封装 就是微码编程



最后出现了高级语言

需要一种工具 需要把人类理解的语言 转换成计算机要理解的语言 ----- 这个中间需要一个编译器或者解释器

到底是编译器 还是 解释器 ---

动态检查一段代码 然后执行代码 这个就是**解释器**

事先代码必须转换成二进制格式 --- 二进制格式 机器直接理解 ---然后机器从头至尾执行即可 ---- 事先的转换就是编译器来完成的 --- 不是边转换 边执行

解释器 类似于同声传译 ---- 执行流程要慢【在线】

编译 就是代码写完之后 把整个代码转换成机器可以理解 ---- 整个编译的过程如果某一部分出现问题 就要重新编译一个库文件 而非整个程序代码 【离线】

编程语言就是把计算机的计算能力 抽象成人类能够理解的一种方式 这种抽象成都越高 离人类越近 就是高级语言 ---- 高级 低级 仅仅用来描述离人类的思维方式的远近程度

这样来讲 python是一种非常高级的语言 非常接近人类的思维方式 比起来众多的语言 python都算得上较为高级的语言

无论怎么转化 很难做到人说一句话 计算机就执行 这是人工智能 ---背后依然是很多代码来支撑

无论机器如何语言符号 但是 无法理解人类的情绪

很显然 无论哪一种语言 用户在借助各种语言 将自己的问题 描述成解决方案

Python就是众多语言

语言可以分成动态 或者静态 还可以分成脚本和编译型的语言

某种角度 python是一种完成自动化管理的脚本语言

---- Python比你想象的强大的多的脚本语言

Openstack 就是Python开发的

Python一样适合开发大型程序 python是一种完备的语言

所以 不要使用脚本编程语言来描述python python一样适合编写大型程序

Python适合三种

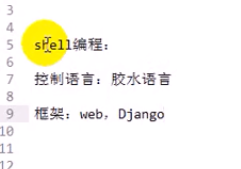
\*\*shell编程工具 此前要完成Linux某些自动化 使用bash shell脚本 要比使用python脚本更合适 ---因为python调用系统的某些命令 这时候 bash shell更合适 ---学完python 仅仅做自动化脚本 大材小用

Python处理文本化的东西 是bash shell 无法比拟的

因此 抓取网页内容分析 固然可以使用shell的awk grep等工具 --- 但是没有Python没有内置类库强大 ---无论如何 python无论如何 不太适合平时的shell脚本 ---- 这时候请使用bash shell

\*\*python是完整的编程语言--- 控制语言 ---- 用于实现链接 ---- 把其他语言编写的库直接调用 --- 从这个角度 python更像一种控制语言 或者 胶水语言

\*\*python很多外部框架 --- 基于框架可以完成很多应用的开发 比如web开发



Python有个缺点 ---- **不能编译 或者 即使能编译 也是解释器现场完成** ---类似于php的opcode 编译成**类似于Java的字节码** ---- 字节码 **以求重复执行** 第一次一定有一个编译过程

Python的字节码叫byte code

某种意义讲 java字节码 在类文件的描述上 接口统一 这样二者可以互换执行

如果编译出来的Python代码编译出来的字节码遵循java的类文件的格式 ---- 这样jvm可以执行

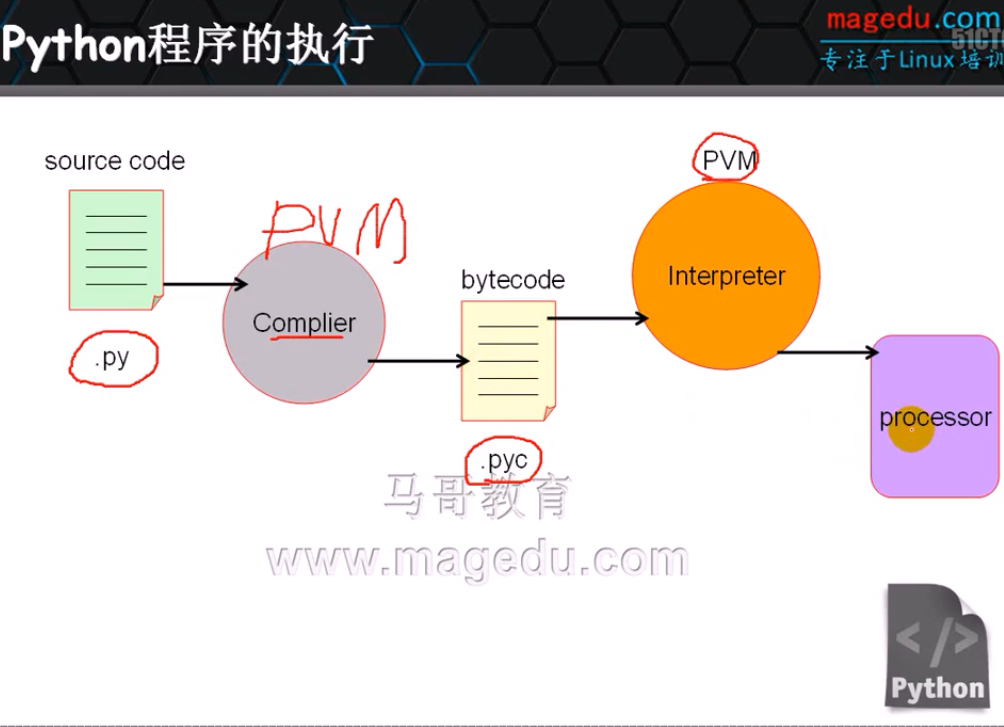
事实上 python有一种实现方式 叫jthon

因此 python和C C++ 性能要差一点

因为有丰富的类库 python开发非常快

这样 开发周期缩短带来的效益 比起执行速度的某些损失 可以忍受

现在计算机的性能非常好 --- 简单的性能损失可以做到忽略不计 更何况 python性能损失换来了 一次编译 处处运行 这种好处 和java一样 ----- 有python pvm 那么这个代码就能运行起来



第一次执行 python的源程序.py compiler在第一次执行的时候 转化成字节码 .pyc ----- 字节码在解释器PVM中运行 ---- 解释器叫PVM 事实上 编译过程也在PVM中进行

当编译结束了 这个执行过程 PVM读取bytecode 然后在处理器processor上面执行

这是python的执行流程

谁在使用python语言？

事实上 python确实流行 ----- 比方google在网络搜索 很多地方使用了python --- python的作者一段时间就职于google

另外 google收购的youtube 服务端是python研发的

Intel syscode hupu 都是使用python做工具 对硬件做测试

还有openstack就是NASA使用Python进行研发

很多智能家居 扫地的机器人 能够无死角除尘 有一家 公司 使用python做研发的

Python是一种非常优秀的语言

【咖啡的杯子 有一条蛇 有意思 咖啡杯是Java的代表 蛇是python 所以 Jython就是Java实现的Python解析器】

需要解释器 bash shell也需要bash shell解释器

Python要能运行起来 也要解释器

主流的Python解释器 有三种

cPython ---- 最原始 最流行的

第二种 Jython ---- java集成实现的JVM

IronPython --- 于.NET --- .net语言实现的python

Cpython运行最快

Jython可以让python运行的时候调用Java的库

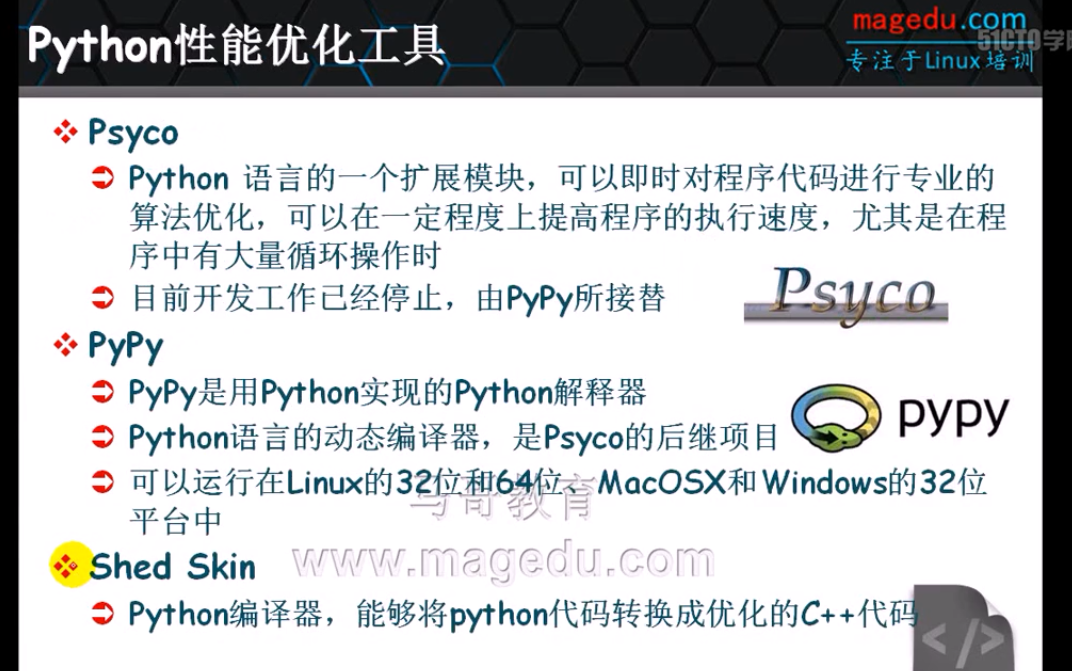
--- Java程序员是非常好的环境

同理是IronPython

Cpython可以调用C某些外部的库 但是不能调用外部的java库

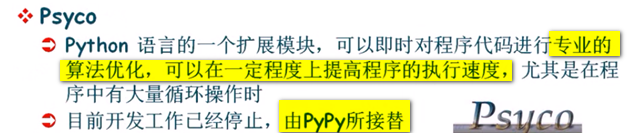
但是 Jython和IronPython也可以调用C的库

===python性能优化工具

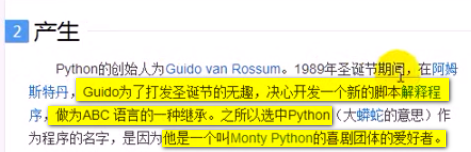


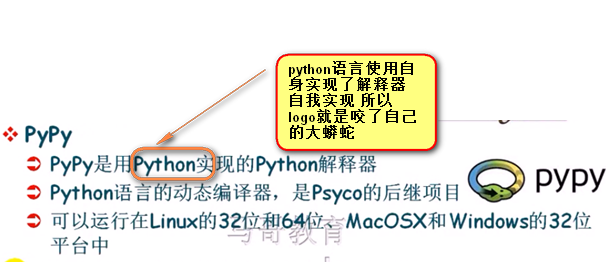
如何在性能上对python进行性能优化？

Psyco --- python的插件



Pypy的logo 

 Python的名字的来历



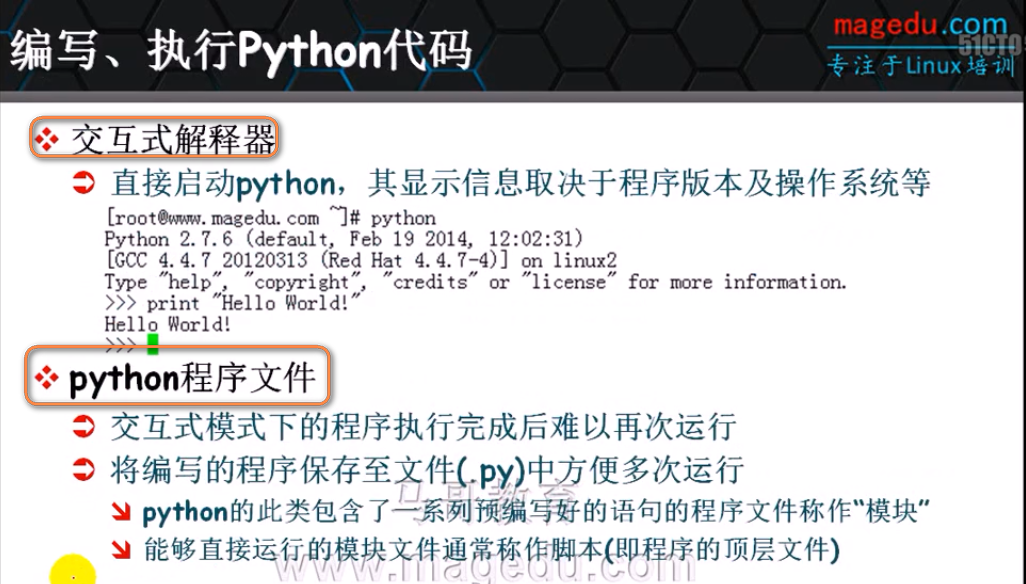
PyPy能够对python进行即使的优化

 ---- 然后使用C++编译器进行编译

这样 性能更高

但是 facebook很多都是php ----但是性能很好 原因就是 php被编译成C++代码 ----

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



Python有很多IDE

Python在windows和linux中的实现 有一个自带的IDLE

IDLE本身就是python实现的应用程序 ---- 自身撞在了很多类库 才能运行

还有很多商业工具 pydev 等等

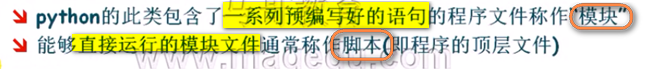
Python手动开发文件



大段代码 反复多次执行 尽可能把代码保存在程序文件中

Python程序文件叫.py ----并非严格要求

Python编程环境中 包含一些列Python语句的文件 统统被称为 “模块”



这个是Python脚本和模块的区别