Python脚本如何保证游戏正常开发

Python脚本是一种广泛应用于玩游戏开发的通信语言，在实际应用的过程中还是有不少的问题困扰着开发人员，下面是Python脚本在实际应用中的具体问题解决方案。希望读完下面的文章对其有所帮助。

在早期的一些游戏中，大部分的游戏逻辑都直接写入游戏代码，例如计算公式、游戏流程等。但随着游戏产业的不断发展，游戏开发本身也变得越来越复杂，游戏企划需要更多的时间来对游戏进行调整，如果游戏逻辑还写在代码中，则企划对游戏的每次修改都要通过程序进行，而且还需要重新编译以及重启程序，这样工作效率就大大降低了.

* [深度剖析Python脚本语言功能](http://developer.51cto.com/art/201002/184959.htm)
* [关于Python脚本语言进行学习介绍](http://developer.51cto.com/art/201002/185287.htm)
* [使用Python脚本语言进行程序编辑](http://developer.51cto.com/art/201002/185333.htm)
* [Python脚本程序与Per进行比较](http://developer.51cto.com/art/201003/185573.htm)
* [调用Python脚本基本应用方式简述](http://developer.51cto.com/art/201003/186239.htm)

在游戏中使用Python脚本，就正好能解决上述问题，小到一些计算公式，大到游戏的控制流程都可以通过Python脚本完成，而且目前的脚本系统大多是解释执行的，因此都可以支持运行时动态修改，这样一来可以立即看到修改结果，非常方便。

**如何使用脚本**

脚本主要有两种方式。一种方式是主程序使用一门高级语言，比如C++进行编写，然后对其嵌入一个脚本解释器，在运行时动态执行一些脚本函数；另一种方式则是整个程序全部使用脚本编写，例如一些泥巴游戏就是直接用LPC脚本编写的。

本文主要研究嵌入式Python脚本使用方法，因为目前大部分的脚本都不能提供如VC++那样方便的调试环境，如果程序全部用脚本编写，当脚本多达几万行甚至十几万行的时候，调试会成为一件非常痛苦的事。而且嵌入式使用时可以将一些非常耗时的代码用C++编写，以保持较好的运行效率。

程序从C++的main( )函数开始启动，然后进入主循环，在一些C++函数中会直接调用脚本函数，在脚本函数的运行过程中，又可能调用C++的扩展函数。C++扩展函数的主要功能有两个：一是用来增加Python脚本无法直接编写的功能，二是用来替换脚本中运行速度过慢的函数。

上述过程的关键点就在于C++和脚本如何相互调用函数，以及如何传递参数与结果。一般的解决方法是在程序启动时利用脚本的API向脚本注册C++的扩展函数，将函数指针传递给脚本系统以便将来调用，调用脚本函数则使用脚本系统的API将调用参数压进栈，取得运行结果也要通过API进行。

**Python 脚本简介**

目前有许多第三方脚本语言可供直接使用，例如Tcl、Lua等等，本文要介绍的是Python脚本。Python已经有超过十年的历史，是一种解释性的、面向对象的脚本语言。Python的解释器在大部分的操作系统上都可以运行，如Windows、Linux、Solaris、Mac 等。

**1. 安装与配置**

安装完成后在开始菜单中会有Python的图形化编辑器(IDLE，但目前版本不支持中文字符)，Python的命令行解释器以及用户手册

为了在C++程序中调用Python的API函数，需要将头文件与lib路径添加到VC++的搜索目录中，头文件路径是本地Python安装目录下的include目录，lib路径是本地Python安装目录下的libs目录。这里需要注意的是安装包只提供了release版本的lib与dll，如果需要调试运行，则必须自己下载Python的源代码以编译debug版本的lib与dll.

**2. 语法简介**

详细的语法说明请参考Python安装包自带的文档，这里我只介绍一些常用的关键字与注意事项。

Python脚本没有C++中的 { 和 } ，它使用缩进来代替。变量不需要单独声明，但不能引用未经赋值的变量。

Python脚本中引入了模块的概念，类似C++中Lipary的概念。模块可以包含函数、变量、类。一个脚本文件就是一个模块，模块在使用前需要导入。

Python中没有switch，使用if判断代替：

1. if ( num==1 ):
2. print "1"
3. elif ( num==2 ):
4. print "2"
5. else:
6. print "unknown

"while 是Python的一个循环语句。在while循环内可以使用continue跳到下个循环，使用peak可以跳出整个循环

1. cnt = 5
2. while ( cnt **>** 0 ):
3. print cnt
4. cnt -= 1
5. for 循环：
6. list = ["test1", "test2", "test3"]
7. for str in list:
8. print str

词典是Python脚本的一种映射数据类型，它能从一个键值(key)映射到实际内容(value)：

1. accounts = {'tom':'123456', 'mike':'654321'}
2. print accounts['tom']
3. print accounts['mike']

**3．API 介绍**

Python脚本提供了大量的C API，C++与Python的交互都是通过这些API进行。下面介绍几个比较重要的API函数：void Py\_Initialize( )在使用Python系统前，必须使用Py\_Initialize对其进行初始化。它会载入Python的内建模块并添加系统路径到模块搜索路径中。

这个函数没有返回值，检查系统是否初始化成功需要使用Py\_IsInitialized。int PyRun\_SimpleString(char \*command)把输入的字符串作为Python代码直接运行，返回0表示成功，-1表示有。大多时候错误都是因为字符串中有语法错误。PyObject\* Py\_BuildValue(char \*format, ...)把C++的变量转换成一个Python对象。

当需要从C++传递变量到Python时，就会使用这个函数。此函数有点类似C的printf，但格式不同。常用的格式有s表示字符串，i表示整型变量，f表示浮点数，O表示一个Python对象。PyObject\* PyObject\_CallObject(PyObject\*callable\_object, PyObject \*args)调用一个callable\_object指向的Python函数，args为调用参数

在使用此函数前可以用PyCallable\_Check来检测callable\_object是否为一个可被调用的Python对象。PyObject\* PyImport\_Import(PyObject \*name)载入一个n a m e 指定的模块。可以先使用PyString\_FromString将模块名转换为Python对象，再使用PyImport\_Import载入。void Py\_Finalize()关闭Python系统，一般在程序退出时调用此函数。