详细参考： [编译型语言、解释型语言、脚本语言等解释](http://blog.csdn.net/zhangliangzi/article/details/50413727)

## 一、编译型语言：

定义：指用专用的编译器，针对特定的操作平台（操作系统）将某种高级语言源代码一次性翻译成可被硬件平台直接运行的二进制机器码（具有操作数，指令、及相应的格式），这个过程叫做编译；编译好的可执行性文件（.exe），可在相对应的平台上运行（移植性差，但运行效率高）。

有些程序编译后，还需要把其他编译好的，可能需要组装两个以上的目标代码生成最终的可执行性文件，称为链接（可实现对低层次代码的复用）。

典型的编译型语言有，我们非常熟悉的C语言、C++等。

## 二、解释型语言：

定义：指用专门解释器对源程序逐行解释成特定平台的机器码并立即执行的语言；相当于把编译型语言的编译链接过程混到一起同时完成的。

解释型语言执行效率较低，且不能脱离解释器运行，但它的跨平台型比较容易，只需提供特定解释器即可。

常见的解释型语言有，笔者正在学习的Python（同时是脚本语言）与Ruby等。

## 三、Java语言：

Java语言是一门很特殊的语言，Java程序需要进行编译步骤，但并不会生成特定平台的二进制机器码，它编译后生成的是一种与平台无关的字节码文件（\*.class）（移植性好的原因），这种字节码自然不能被平台直接执行，运行时需要由解释器解释成相应平台的二进制机器码文件。

大多数人认为Java是一种编译型语言，但我们说Java即是编译型语言，也是解释型语言也并没有错。

## 三、脚本语言：

定义：为了缩短传统的编写-编译-链接-运行（edit-compile-link-run）过程而创建的计算机编程语言。

它的特点是：程序代码即是最终的执行文件，只是这个过程需要解释器的参与，所以说脚本语言与解释型语言有很大的联系。脚本语言通常是被解释执行的，而且程序是文本文件。

典型的脚本语言有，JavaScript，Python等。

## 四、系统语言：

系统语言是与脚本语言相对应的，他们的区别是：

1、抽象的级别：这是最重要也是最明显示的不同。脚本语言对程序员提供了更高级的抽象。这一点明显表现在：在这种语言自身中，存在有高级的数据结构，如列表和字典结构，和对这种结构简单方便的嵌套和操作。这样可以创建非常成功的程序。

2、类型定义：系统语言通常是强类型和静态类型定义。这就意味着所有变量的类型要在程序中指定，在编译时检查。相反地，脚本语言是最松散的类型定义，完全没有类型声明，并且在运行时进行动态类型检查。

3、执行：系统语言的特点是编译的。程序被编译成可执行的二进制。另一方面，脚本语言的特点是解释，也就是，指令被立即执行，不存在一个编译的中间状态。这就意味着脚本语言是交互式的(你可以在提示符下敲入命令，并且看到结果)，这是另一个巨大的胜利。这样完全将编译过程从编辑-编译-运行循环中去掉了。

4、速度：以上三点是脚本语言一方面在速度与效率， 一方面在易用性与表示式的强大性之间进行折衷的典型例子。这就使得脚本语言的执行速度比系统语言慢一个数量级。这就是对脚本语言诽谤最多的方面。性能的降低不是真正的问题，因为思想是用脚本语言来组合组件，这些组件是用像C这样的快速系统语言来编写的。所以所有需要运行快速的东西将因为是用快速的语言实现的而运行得快速。脚本语言只用于将东西绑在一起，并且这些通常不是性能的瓶颈(或如果是，你需要重新检查你的设计)。