PYTHON 的设计哲学是“优雅”、“明确”、“简单”。因此，Perl 语言中“总是有多种方法来做同一件事”的理念在 PYTHON 开发者中通常是难以忍受的。PYTHON 开发者的哲学是“用一种方法，最好是只有一种方法来做一件事”。在设计 PYTHON 语言时，如果面临多种选择，PYTHON 开发者一般会拒绝花俏的语法，而选择明确的没有或者很少有歧义的语法。由于这种设计观念的差异，PYTHON 源代码通常被认为比 Perl 具备更好的可读性，并且能够支撑大规模的软件开发。这些准则被称为 PYTHON 格言。在 PYTHON 解释器内运行 importthis 可以获得完整的列表。

PYTHON 开发人员尽量避开不成熟或者不重要的优化。一些针对非重要部位的加快运行速度的补丁通常不会被合并到 PYTHON 内。所以很多人认为 PYTHON 很慢。不过，根据二八定律，大多数程序对速度要求不高。在某些对运行速度要求很高的情况，PYTHON 设计师倾向于使用 JIT 技术，或者用使用 C/C++语言改写这部分程序。可用的 JIT 技术是 PyPy。

PYTHON 是完全面向对象的语言。函数、模块、数字、字符串都是对象。并且完全支持继承、重载、派生、多继承，有益于增强源代码的复用性。PYTHON 支持重载运算符和动态类型。相对于 Lisp 这种传统的函数式编程语言，PYTHON 对函数式设计只提供了有限的支持。有两个标准库(functools, itertools)提供了 Haskell 和 StandardML 中久经考验的函数式程序设计工具。

虽然 PYTHON 可能被粗略地分类为“脚本语言”（scriptlanguage），但实际上一些大规模软件开发计划例如 Zope、Mnet 及 BitTorrent，Google 也广泛地使用它。PYTHON 的支持者较喜欢称它为一种高级动态编程语言，原因是“脚本语言”泛指仅作简单程序设计任务的语言，如 shellscript、VBScript 等只能处理简单任务的编程语言，并不能与 PYTHON 相提并论。

PYTHON 本身被设计为可扩充的。并非所有的特性和功能都集成到语言核心。PYTHON 提供了丰富的 API 和工具，以便程序员能够轻松地使用 C 语言、C++、Cython 来编写扩充模块。PYTHON 编译器本身也可以被集成到其它需要脚本语言的程序内。因此，很多人还把 PYTHON 作为一种“胶水语言”(glue language)使用。使用 PYTHON 将其他语言编写的程序进行集成和封装。在 Google 内部的很多项目，例如 Google Engine 使用 C++编写性能要求极高的部分，然后用 PYTHON 或 Java/Go 调用相应的模块。《PYTHON 技术手册》的作者马特利（AlexMartelli）说：“这很难讲，不过，

2004 年，PYTHON 已在 Google 内部使用，Google 召募许多 PYTHON 高手，但在这之前就已决定使用 PYTHON，他们的目的是 PYTHON where we can, C++ where we must，在操控硬件的场合使用 C++，在快速开发时候使用 PYTHON。”