perl常问问题集第1篇 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/167/2021_2022_perl_E5_B8_B8_E9_97_AE_c103_167244.htm Perl是什么？ Perl是一个高阶程式语言，由 Larry Wall和其他许多人所写，融合了许多语言的特性。它主要是由无所不在的 C语言，其次由 sed、awk，UNIX shell 和至少十数种其他的工具和语言所演化而来。Perl对 process、档案，和文字有很强的处理、变换能力，因此举凡有关快速原型设计、系统工具、软体工具、系统管理、资料库连结、图像程式设计、网路连结，和 WWW程式设计等之类的任务，都特别 适合用 Perl来做。这些特长不但使 Perl成为系统维护管理者和 CGI作者的宠儿，就连数学家、遗传学家、新闻从业者，甚至企业管理者也都用 Perl，所以或许您也该用。 谁对 perl提供支援？由谁负责发展？它为什麼是免费的？ Perl自由开放的发行方式要归功於发烧前的 Internet的传统文化及其作者 Larry Wall。Perl是由使用者提供支援。现在 Perl的核心、标准程式库、选择性安装的模组，以及您现在正在阅读的使用说明都出自於义务者之手。详情请见 perl原始码发行版中所附的 README档案底部的私人注记。 值得一提的是，核心发展小组（称为 Perl Porters）的成员是一群高度热情奉献的人仕，全心投入发展出比您所能想像、用钱能买得到还要更好的免费软体。您可经由 news://genetics.upenn.edu/perl.porters-gw/ 和 http://www.frii.com/~gnat/perl/porters/summary.html取得关於新近发展计画 的情报。 尽管 GNU计画将 Perl囊括在它的发行中，但是没有叫「GNU Perl」这样的东西。 Perl既非自由软体基金

会所创，亦非由其负责维护。Perl的发行条款同时也较 GNU 软体更来得开放。 如果您愿意，您可以购买商业性的 Perl支援。但对大多数使用者来说，非正式性的支援通常已相当足够。详情请见「到哪里可买到商业性的 Perl支援」一问的回答。 我该用哪一个版本的 Perl？ 您绝对该用第五版。第四版不但老旧、功能较局限，而且已经不再维护了。它最後 一次的修正 (4.036)是在 1992年。Perl最新的量产发行版本是5.004。等到您 读这篇文章时，我们可能已经又发行了几个正式的除错版本，同时大概又会有些替 下一版 路的实验版出来。本文由此开始凡提及 Perl语言，皆以目前的量产发行 为准，除非另外特别注明。 perl4和 perl5各代表什麼？ perl4 和 perl5 是对 Perl程式语言的两个不同版本的非正式称呼，因为说「perl5」要比说「第 5(.004)版的 Perl」要来得简单。但是有些人误将其会意为：perl5是一个单独的语言；这是不正确的。perl5只不过是对第五个主要发行版本（1994年 10 月）常用的称呼罢了。就像 perl4是指第四个主要发行（1991年 3 月），还有 perl1（1988年 1月）、perl2（1988 年 6 月），以及 perl3（1989年 10 月）。 5.0的发行基本上是从零开始，所有程式码完全重新写过的版本。它已经被模组化、物件导向化、微调、精简化，及效率化，以致程式码几乎已变得和原来的不相同了。尽管如此，使用介面大致上仍然相同，而且和先前的版本之间保持了很高的 一致性。 为了避免「perl5是什麼语言？」这类的混淆，有些人索性完全避免「perl5」， 而单用「perl」来指称最近的 perl版本。其实用不着这麼累就是了。 Perl的发展已稳定了吗？ 融合了除错和新功能的量产发行在推出前皆经过广泛的测试。自 5.000发行以来， 我们平均一

年才出版一次量产发行。 Larry 和 Perl发展小组有时候会修改语言的核心部分，但总是尽一切力量让新版 和旧版保持一致。因此，尽管不是所有的 perl4 scripts都能在 perl5 之下跑得天衣无缝，因升级而导致按照先前版本的 perl所写的程式无法使用的情形几乎不曾发生（除非该程式倚赖已经被去除的 bugs，或使用了极少数新加入的指令来 命名）。 Perl难学吗？ Perl不但容易上手，也容易继续学下去。它看起来和大多数您可能已接触过的语言一样。所以如果您只写过 C 程式、或 awk script、shell script，或甚至只是 Excel的 macro（巨集），您已经在半路了。 大多数的任务只需要 Perl语言的一小部分即可完成。发展 Perl程式的座右铭即 是「不只一种方法可以达到」（TMTOWTDI. Theres More Than One Way To Do It, 有时读作「堤姆投迪」）。因此，Perl的学习曲线是既平（易学）且长的 （如果您要的话，有一大堆够您学的）。 最後，Perl（通常）算是解译式的语言。也就是说您写了程式後不需经由一道中 间的编码过程即可测试；这让您可以很快、很容易地测试及除错。这个方便试验的 特性又让学习曲线变得更加平坦。 有助於修习 Perl 的一些事：UNIX经验、对几乎是任何一种程式语言的经验、了解 regular expressions（正规表示法），以及看得懂旁人写的程式的能力。如果您有什麼想用 Perl来做的事，那麼可能已经有前人做过了，而且实例通常可免费取得。还有别忘了新的 Perl模组。模组在这份 FAQ 的第叁部分有详细的讨论，还有【别忘了您的好朋友】 CPAN，这会在第二部分谈到。 Perl和其他的程式语言比起来如何？例如 Java, Python, REXX, Scheme,或 Tcl？ Perl在某些地方比较好，某些地方较差。精确地说到底哪些方面好或坏通常视个

人偏好而定，所以在新闻讨论群中问这种问题很可能会掀起一场毫无建设性的圣战。 要比较各语言的异同最好的方法是试着用不同的语言写功能相同的程式。各程式语言都各有属於它们各自的新闻讨论群，您可从中学习（但希望您不是去和人辨论吵 架的）。 我可以用 Perl来做【某种差事】吗？ Perl有足够的弹性和扩充性，从只需要写短短一行的档案处理工作到复杂的系统，几乎没有什麼做不到的。对有些人来说，Perl的是拿来做写 shell程式的理想替代品。其他人则用高阶的 Perl来替代处理许多原先需要用 C或 C 一类的低阶语言来达到的程式。哪些差事决定要用 Perl来处理，这一切都得看您（或许还有您的经理...）。 如果您有一个提供 API的程式库的话，您可用 C或 C 来写一个 Perl 延伸，然後便可透过它将程式库中的任何一部分动态载入您的 Perl主程式中。您也可以 反过来，用 C或 C 来写主程式，然後以即时动态载入的方式插入一些Perl程式码，产生一个威力强大的应用程式。 话虽如此，对解决某些特定的问题，使用小型、专精，专为特殊用途设计的语言总 是比较方便的。 Perl的设计是尽力地满足各种不同人的需要，因而不特别偏颇任何人。至於特殊功能语言的例子，随便举两个，譬如 prolog 和 matlab 便是。 哪些场合下不适合用 Perl？ 当您的主管禁止的时候 -- 不过请务必考虑把他们换掉 。 说真的，如果您已经有用另一个语言写成的应用程式（而且写得很好）的时候，或 者是已经有替某些特定的工作设计的语言（例如：prolog, make），这个时候就不 需要用 Perl。 由於种种因素，Perl大概不太适合拿来做即时内嵌式系统、属於低层级的作业系统发展工作，例如周边设备的 drivers或环境转换码、复杂的多线共用记忆体应

用程式，或非常大的应用程式。您会发现 Perl 本身便不是以 Perl写成的。 刚出炉的 Perl纯码编译器或许可帮忙去除一些上述的限制，但您要了解：Perl在本质上仍是一活性变数语言 (dynamically typed language)，而非固性变数 (statically typed)。只要您不将核电厂或脑科手术监视器所用的程式放心地用 Perl来写，您自然就不会闯祸遭殃。这样 Larry晚上也可以睡得安稳些 --股市分析程式不在此限 。 「perl」和「Perl」有什麼不同？ 二者差一个位元。喔，您不是说在 ASCII上的差别啊？ Larry 现在用「Perl」来代表语言本身，而以「perl」来表示该语言的体现，即目前的解译器。因此，作者有句幽默小语说：「只有 perl可以解译 Perl」。要不要遵照这个用法是您的自由。举一反叁的话，我们可依样画葫芦地说「awk 和 perl」还有「Python 和 Perl」，但却不可将「awk 和 Perl」或是「Python 和 perl」摆在一起。 Perl程式应算是 program还是 script？ 都无所谓。按标准术语来讲，program指已经由编译程序编译好、转为机器码，可多次执 行的程式；而 script则是每次执行时都必须透过一个解译程式来解译。然而，Perl程式严格说来，既非编译 (compiled)，亦非解译式 (interpreted)；因 Perl程式可转译成位元码形式存在（可说是某种 Perl虚拟机器 [virtual machine]），或转译为完全不同的语言，如 C或组合语言。所以光看原始码很难 说它到底是替纯解译器、或是 parse-tree解译器、位元码解译器，还是纯码编译器而写；因此这题很难给它一个确切的答案。 JAPH是什麼？ 这是过去一些在讨论群中自称 ``just another perl hacker 的人的签名档，约有一百个比较早期的，可在 http://www.perl.com/CPAN/misc/japh 取得。 到哪儿可拿到

Larry Wall的智慧讽语 (witticisms)？ 一百多条 Larry的讽语，源自他【在讨论群】的 posts或原始码，可在http://www.perl.com/CPAN/misc/lwall-quotes 取得。 我要如何取信、说服我的系统管理者／上司／属下使用第 5/5.004版的 Perl，而不去用其他的语言？ 如果您的管理阶层或属下对没有支援的软体，或是未正式包含在所购买的作业系统中的软体存有戒心的话，您可以试着从有助他们自身利益这方面下手。因为如果程式设计师能由善加利用 Perl的结构、功能性、简单性，和威力而获得更大的生产力的话，那麼典型的管理者／上司／员工或许便可因而加以说服。此外，使用 Perl，总的来讲，和其他语言相较，或许也有助於减少交件的时间。强调这个论 点或许对说服他们会有帮助。 如果您的专题碰到瓶颈，特别是有关转译或测试方面的问题，那麼 Perl 可以说绝 对会是一个既可行且快的解决之道。您在当说客的时候，千万别忘了要提：Perl已 被世界上许多大型的软硬体公司广泛、大量地使用，极为可靠、有效。事实上，现 Perl 已成为许多 Unix业者所售的作业系统的标准配备了。而且如果您无法在 详尽的使用说明，包括这份 FAQ之中为您的问题找到解答的话，送封 post 到新闻讨论群即可。 如果您面对反对 perl升级的声音，那麼告诉他们 Perl发展小组已经完全不再维护或支援第四版的 perl了。perl5的另一个大卖点是它有大量的模组和延伸，可大大减少计画的发展时间。还有，告诉他们第四和第五版 Perl之间的差异就如 awk 和 C 的差别一样（嗯，或许没有差得那麼明显，但您知道我的意思就好）。如果您想得到支援而且想确保您现在所发展的软体在未来能继续工作的话，那麼您得跑有支援的版本。这大概也就是说

要跑 5.004 版的，尽管 5.003 版仍算是不错（它只落後一年、一版）。不过因为有些严重的 bugs曾在 5.000 和 5.002版之间被消除，所以您至少应升级到比这几个版本高才是。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com