perlfaq1

Table des matières

1 NAME/NOM

perlfaq1 - Questions d'ordre général sur Perl

2 DESCRIPTION

Cette section de la FAQ répond aux questions d'ordre général et de haut niveau sur Perl.

2.1 Qu'est ce que Perl ?

Perl est un langage de programmation de haut niveau avec un héritage éclectique écrit par Larry Wall et un bon millier de développeurs. Il dérive de l'omniprésent langage C et, dans une moindre mesure, de Sed, Awk, du shell Unix et d'au moins une douzaine d'autres langages et outils. Son aisance à manipuler les processus, les fichiers et le texte le rend particulièrement bien adapté aux tâches faisant intervenir le prototypage rapide, les utilitaires système, les outils logiciels, les gestionnaires de tâches, l'accès aux bases de données, la programmation graphique, les réseaux, et la programmation web. Ces points forts en font un langage particulièrement populaire auprès des administrateurs système et des auteurs de scripts CGI, mais l'utilisent également des mathématiciens, des généticiens, des journalistes et même des managers. Et peut être vous aussi ?

2.2 Qui supporte Perl ? Qui le développe ? Pourquoi est-il gratuit ?

La culture d'origine d'Internet et les croyances profondément ancrées de l'auteur de Perl, Larry Wall, ont donné naissance à la politique de distribution gratuite et ouverte de perl. Perl est soutenu par ses utilisateurs. Le noyau, les bibliothèques standards Perl, les modules optionnels, et la documentation que vous êtes en train de lire ont tous été rédigés par des volontaires. Vous pouvez consulter les notes personnelles à la fin du fichier README de la distribution du code source de Perl pour plus de détails. L'historique des versions de Perl (jusqu'à la 5.005) est disponible ici : *perlhist*.

En particulier, l'équipe principale de développement (connue sous le nom 'Perl Porters') est une bande hétéroclite d'individus des plus altruistes engagés à faire un logiciel gratuit meilleur que tout ce que vous pourriez trouver dans le commerce. Vous pouvez glaner des informations sur les développements en cours en lisant les archives sur http://www.xray.mpe.mpg.de/mailing-lists/perl5-porters/ et http://archive.develooper.com/perl5-porters@perl.org/ ou via la passerelle de news nntp nntp://nntp.perl.org/perl.perl5.porters ou son interface web http://nntp.perl.org/group/perl.perl5.porters ou encore en lisant la faq sur http://simon-cozens.org/writings/p5p-faq. Vous pouvez aussi vous inscrire à la liste de diffusion en envoyant une demande d'inscription à perl5-porters-request@perl.org (un message vide sans sujet suffit).

Bien que le projet GNU inclue Perl dans ses distributions, il n'y a pas à proprement parler de "GNU Perl". Perl n'est pas produit ni entretenu par la Free Software Foundation. Les termes de la licence de Perl sont aussi moins restrictifs que ceux de la licence GNU.

Vous pouvez obtenir une assistance commerciale pour Perl si vous le souhaitez, mais pour la plupart des utilisateurs, une assistance informelle devrait être largement suffisante. Regardez la réponse à question "Où acheter une version commerciale de perl" pour de plus amples informations.

2.3 Quelle version de Perl dois-je utiliser ?

(contribution de brian d foy)

Tous les goûts sont dans la nature... il n'y a donc pas de réponse à cette question qui puisse convenir à tout le monde. En général, vous voudrez utiliser soit la dernière version stable soit celle qui la précède immédiatement. Actuellement, ce sont respectivement perl5.8.x et perl5.6.x.

Après cela, vous devez tout de même considérer plusieurs pour décider au mieux de vos intérêts.

- Si tout marche bien, la mise à jour de perl peut faire que ça ne marche plus (ou au minimum amener de nouveaux avertissements).
- Les dernières versions de perl bénéficie de plus de corrections de bugs.
- La communauté Perl fournit de l'aide plutôt pour les versions les plus récentes, il vous sera donc plus facile de vous faire aider avec ces versions.
- Les versions antérieures à perl5.004 présentent de très sérieux problèmes de sécurité comme des dépassements de tampons et font l'objet de plusieurs avertissements du CERT (par exemple, http://www.cert.org/advisories/CA-1997-17.html).
- Les toutes dernières versions sont sans aucun doute les moins déployées et testées donc, si vous n'aimez pas prendre de risques, vous pouvez attendre quelques mois après leur sortie afin de voir les problèmes rencontrés par les autres.
- L'avant-dernière version (actuellement perl5.6.x) est habituellement maintenue assez longtemps, bien qu'elle ne soit pas au même niveau que la version courante.
- Personne ne met à jour perl4.x. Il y a cinq ans, c'était déjà une simple carcasse de chameau mort. Maintenant, ce n'est même plus un squelette et ses os d'un blanc immaculé s'érodent et tombent en miettes.
- perl6.x ne verra pas le jour tout de suite. Restez attentif mais ne craignez pas d'avoir à changer de versions majeures de Perl avant un moment (NdT: pas avant fin 2006).
- Il existe deux pistes parallèles de développement de perl : une piste pour les versions de maintenance et une piste pour les versions expérimentales. Les versions de maintenance sont stables et leur numéro de version mineur est paire (par exemple perl5.8.x où 8 est le numéro de version mineur). Les versions expérimentales peuvent inclure des fonctionnalités qui ne sont pas dans les versions stables et leur numéro de version mineur est impair (par exemple perl5.9.x où 9 est le numéro de version mineur).

2.4 Qu'est-ce que veut dire perl4, perl5 ou perl6 ?

(contribution de brian d foy)

Pour faire court, perl4 est le passé, perl5 est le présent et perl6 est le futur.

Le numéro après perl (le 5 dans perl5) est le numéro de version majeur de l'interpréteur perl ainsi que la version du langage. Chaque version majeure possède des différences significatives que les versions antérieures ne peuvent pas supporter.

La version majeure courante de Perl est perl5 et date de 1994. Elle peut exécuter des scripts de la version majeure précédente (perl4 qui date de mars 1991 mais a des différences importantes. Elle introduit le concepte de références, les structures de données complexes et les modules. L'interpréteur perl5 est une réécriture complète des sources perl précédents.

Perl6 est la prochaine version majeure de Perl mais il est encore en développement tant du point de vue de sa syntaxe que du point de vue des choix de conception. Le travail a commencé en 2002 et est encore en cours. La plupart des nouvelles fonctionnalités sont apparues dans les dernières versions de perl5 et certains modules perl5 vous permettent même d'utiliser quelques éléments de la syntaxe perl6 dans vos programmes. Vous pouvez en apprendre plus sur http://dev.perl.org/perl6/ .

Voir *perlhist* pour un historique des révisions de Perl.

2.5 Qu'est-ce que Ponie ?

Lors de la convention O'Reilly Open Source Software en 2003, Artur Bergman, Fotango et la Fondation Perl ont annoncé un projet nommé Ponie dont le but est de faire tourner perl5 sur la machine virtuelle Parrot. Ponie signifie Perl On New Internal Engine (Perl sur le nouveau moteur). L'implémentation du langage Perl 5.10 sera utilisée pour Ponie et il n'y aura aucune différence de langage entre perl5 et ponie. Ponie n'est pas une réécriture complète de perl5.

2.6 Qu'est-ce que perl6 ?

Lors de la seconde convention O'Reilly Open Source Software, Larry Wall a annoncé que le développement de Perl6 allait commencé. Perl6 avait parfois été utilisé pour désigner le projet Topaz de Chip Salzenberg visant à réécrire Perl en C++. Ce projet, bien qu'ayant fourni des apports non négligeables pour la prochaine version de Perl et de son implémentation, est maintenant abandonné.

Si vous voulez en savoir plus sur Perl6 ou si vous souhaiter aider à améliorer Perl, allez sur la page des développeurs de Perl6 http://dev.perl.org/perl6/ .

Il n'y a pas encore de date prévue pour Perl6 et même après sa sortie, Perl5 restera encore longtemps maintenu. N'attendez donc pas Perl6 pour faire ce que vous devez faire.

"Sérieusement, nous réinventons tout ce qui doit être réinventé." – Larry Wall

2.7 Est-ce que Perl est stable ?

Les nouvelles versions produites, qui incluent des corrections de bugs et de nouvelles fonctionnalités, sont amplement testées avant d'être distribuées. Depuis la version 5.000, on compte en moyenne une version par an.

Larry et l'équipe de développement Perl font occasionnellement des changements dans le noyau interne du langage, mais tous les efforts possibles sont déployés pour assurer la compatibilité descendante. Bien que quelques scripts perl4 ne tournent pas impeccablement sous perl5, une mise à jour de perl ne devrait presque jamais invalider un programme écrit pour une version plus ancienne. (exception faite des corrections de bugs accidentels et des rares nouveaux mots réservés).

2.8 Est-il difficile d'apprendre Perl ?

Non, il est facile de débuter et même de continuer l'apprentissage de Perl. Il ressemble à la plupart des langages de programmation que vous avez probablement rencontrés ; aussi, si vous avez déjà écrit un programme en C, un script awk, un script shell ou même un programme en BASIC, vous avez déjà fait une partie du chemin.

La plupart des tâches ne requiert qu'un petit sous-ensemble du langage Perl. Une des idées phares en matière de développement Perl est : "Il y a plus d'une façon de procéder". La courbe d'apprentissage de Perl est étroite (facile à apprendre) et longue (il y a beaucoup de choses faisables si vous le voulez réellement).

Enfin, puisque Perl est souvent (mais pas toujours, et certainement pas par définition) un langage interprété, vous pouvez écrire vos programmes et les tester sans phase intermédiaire de compilation, vous permettant ainsi d'expérimenter et de déboguer/tester rapidement et facilement. Cette facilité d'expérimentation aplatit encore plus sa courbe d'apprentissage.

Les choses qui facilitent l'apprentissage de Perl sont : l'expérience d'Unix, quasiment n'importe quelle expérience de la programmation, une compréhension des expressions rationnelles, et la capacité de comprendre le code des autres. S'il y a quelque chose que vous avez besoin de faire, elle a probablement été déjà faite, et un exemple fonctionnant est généralement disponible gratuitement. N'oubliez pas non plus les modules perl. Ils sont abordés dans la partie 3 de cette FAQ, et avec le CPAN dans la partie 2.

2.9 Est-ce que Perl tient la comparaison avec d'autres langages comme Java, Python, REXX, Scheme ou Tcl ?

Oui pour certains domaines, moins pour d'autres. Définir précisément dans quels domaines Perl est bon ou mauvais est souvent une question de choix personnel, et poser cette question sur Usenet risque fortement de déclencher un débat houleux et stérile.

La meilleure chose à faire est certainement d'essayer d'écrire le code équivalent dans plusieurs langages pour accomplir un ensemble de tâches. Ces langages ont leurs propres newgroups dans lesquels vous pouvez en apprendre plus (et non, espérons le, vous disputer) sur eux.

Des comparatifs se trouvent sur http://www.perl.com/do©FMTEYEWTK/versus/ si vous ne pouvez vraiment pas vous en passer.

2.10 Que puis-je faire avec Perl ?

Perl est suffisamment souple et extensible pour que vous puissiez l'employer virtuellement pour tout type de tâche, du traitement de fichier ligne par ligne à des grands systèmes élaborés. Pour de nombreuses personnes, Perl remplace avantageusement les outils existants pour les scripts shell. Pour d'autres, c'est un substitut efficace et de haut niveau pour tout ce qu'ils programmeraient en langage de bas niveau comme C ou C++. En définitive c'est à vous (et certainement à votre direction...) de voir pour quelles tâches vous allez utiliser Perl ou non.

Si vous avez une bibliothèque fournissant une API, vous pouvez en rendre n'importe quel composant disponible comme une fonction ou une variable Perl supplémentaire en utilisant une extension Perl écrite en C ou C++, et dynamiquement liée dans votre interprêteur perl principal. À l'opposé, vous pouvez également écrire votre programme principal en C/C++, et ensuite lier du code Perl au vol pour créer une puissante application. Voir *perlembed*.

Ceci dit, il y aura toujours des langages dédiés à une classe de problèmes qui sont tout simplement plus pratiques. Perl essaye d'être tout à la fois pour tout le monde, mais rien de précis pour personne. Les exemples de langages spécialisés qui viennent à l'esprit comptent prolog et matlab.

2.11 Quand ne devrais-je pas programmer en Perl ?

Quand votre patron vous l'interdit - mais envisagez de lui trouver un remplaçant :-)

En fait, une bonne raison pour ne pas utiliser Perl est d'avoir une application déjà existante écrite dans un autre langage qui est toute faite (et bien faite), ou si vous avez une application conçue pour une tâche spécifique dans un langage particulier (i.e. prolog, make)

Pour des raisons variées, Perl n'est probablement pas bien adapté pour des systèmes embarqués temps-réel, des développements systèmes de bas niveau, des travaux comme des pilotes de périphérique ou du code à commutation de contexte, des applications parallèles complexes en mémoire partagée, ou des applications très grandes. Vous remarquerez que perl n'est pas lui-même écrit en Perl.

Le nouveau compilateur Perl de code natif peut éventuellement réduire ces limitations jusqu'à un certain degré, mais comprenez que Perl reste fondamentalement un langage à typage dynamique, et non à typage statique. On ne vous en voudra pas si vous ne lui faites pas confiance pour un code de centrale nucléaire ou de contrôle de chirurgie cérébrale. Larry en dormira d'autant mieux - en ne tenant pas compte des programmes de Wall Street. :-)

2.12 Quelle est la différence entre "perl" et "Perl" ?

Un bit. Ah, mais vous ne parliez pas de codes ASCII ? :-) Larry emploie le terme "Perl" pour désigner le langage en lui même, tandis que "perl" désigne son implémentation, c'est-à-dire l'interpréteur actuel. D'où la boutade de Tom : "Rien d'autre que perl ne peut analyser Perl". Vous pouvez ou non choisir de suivre cet usage. Le parallèlisme implique par exemple que "awk et perl" et "Python et Perl" ont l'air correct, tandis que "awk et Perl" ou "Python et perl" ne le sont pas. Mais n'écrivez jamais "PERL" parce que perl n'est pas un acronyme, si l'on ne tient pas compte du folklore apocryphe ni des expansions a posteriori.

2.13 Parle-t-on de programme Perl ou de script Perl ?

Larry ne s'en soucie pas vraiment. Il dit (à moitié pour rire) qu'un "script est ce que l'on donne aux acteurs. Un programme est ce que vous donnez à l'audience."

Originellement, un script était une séquence en boîte de commandes interactives normales, c'est-à-dire un script de chat. Une chose telle qu'un script de chat UUCP ou PPP ou un script expect est plutôt conforme à cette description, tout comme les scripts de configuration exécutés par un programme lors de son lancement, comme *.cshrc* ou *.ircrc*, par exemple. Les scripts de chat étaient juste des pilotes pour des programmes existants, et non pas des programmes indépendants.

Un scientifique de l'informatique expliquera convenablement que tous les programmes sont interprétés, et que la seule question qui se pose est à quel niveau. Mais si vous posez cette question à quelqu'un d'autre, il pourra vous dire qu'un *programme* a été compilé une seule fois pour devenir du code machine physique, et peut alors être exécuté plusieurs fois, tandis qu'un *script* doit être traduit par un programme à chaque fois qu'il est utilisé.

Les programmes Perl ne sont (habituellement) ni strictement compilés, ni strictement interprétés. Ils peuvent être compilés sous la forme d'un pseudo-code (pour une sorte de machine virtuelle Perl) ou dans un tout autre langage, comme du C ou de l'assembleur. Vous ne pouvez pas dire juste en le regardant si la source est destinée à un interpreteur pur, un interpréteur d'arbre syntaxique, un interpréteur de pseudo-code, ou un compilateur générant du code machine, donc il est délicat ici de donner une réponse définitive.

Maintenant que "script" et "scripting" sont des termes ayant été abusés par des marketeux sans scrupules ou ignorants pour leur propre bénéfice infâme, ils ont commencé à avoir des significations étranges et souvent péjoratives, comme "pas sérieux", ou "pas de la vraie programmation". Par conséquent, certains programmeurs Perl préfèrent totalement les éviter.

2.14 Qu'est ce qu'un JAPH ?

Ce sont les signatures "just another perl hacker" que certains utilisent pour signer leurs messages. Randal Schwartz les a rendues célèbres. Environ une centaine parmi les plus anciennes sont disponibles sur http://www.cpan.com/CPAN/misc/japh .

2.15 Où peut on trouver une liste des mots d'esprit de Larry Wall ?

Plus d'une centaine de boutades de Larry, issues de ses messages ou du code source, peuvent être trouvées à l'adresse http://www.cpan.org/mis©lwall-quotes.txt.gz .

2.16 Comment convaincre mon administrateur système/chef de projet/employés d'utiliser Perl/Perl5 plutôt qu'un autre langage ?

Si votre directeur ou vos employés se méfient des logiciels non entretenus, ou non officiellement fournis avec votre système d'exploitation, vous pouvez essayer d'en appeler à leur intérêt personnel. Si les programmeurs peuvent être plus productifs en utilisant les constructions, la fonctionnalité, la simplicité et la puissance de Perl, alors le directeur/chef de projet/employé type devrait être convaincu. Quant à l'usage de Perl en général, il est parfois aussi utile pour démontrer que les temps de livraison peuvent être réduits en l'utilisant plutôt qu'un autre langage.

Si vous avez un projet avec des impératifs, notamment en termes de portabilité ou de tests, Perl devrait certainement apporter une solution rapide et viable. En plus de tout effort de persuasion, vous ne devriez pas manquer de montrer que Perl est utilisé, assez intensivement et avec des résultats solides et fiables, dans de nombreuses grandes entreprises informatiques de logiciel et de matériel à travers le monde. En fait, de nombreuses distributions Unix livrent maintenant Perl en standard. Les forum de discussion usenet apportent l'assistance nécessaire si vous ne trouvez pas la réponse dans la documentation *complète*, y compris cette FAQ.

Voir http://www.perl.org/advocacy/ pour plus d'informations.

Si vous êtes confronté à des réticences pour mettre à jour une version antérieure de perl, insistez sur le fait que la version 4 n'est plus entretenue ni suivie par l'équipe de développement Perl. Un autre argument de taille en faveur de Perl5 est le grand nombre de modules et d'extensions qui réduisent considérablement le temps de développement pour tout type de tâche. Mentionnez également que la différence entre la version 4 et la version 5 de Perl est similaire à celle entre awk et C++. (Bon d'accord, ce n'est peut-être pas si différent, mais l'idée est là.) Si vous voulez de l'assistance et des garanties raisonnables que ce que vous développez fonctionnera encore dans le futur, alors vous devriez utiliser une version maintenue. En décembre 2003, cela signifie utiliser soit la 5.8.2 (sortie en novembre 2003) ou l'une des versions plus anciennes telles que la 5.6.2 (sortie aussi en novembre 2003 pour assurer la compilation de perl 5.6 sur les machines récentes puisque la 5.6.1 datait d'avril 2001) ou la 5.005_03 (qui date de mars 1999). À la rigueur la 5.004_05 si vous

avez **absolument** besoin d'une aussi vieille version (avril 1999) pour des raisons de stabilité. Aucune version antérieure ne devrait être utilisée.

À noter en particulier la chasse de grande envergure aux erreurs de dépassement de tampon survenues dans la version 5.004. Toute version antérieure à celle-là, y compris perl4, est considérée comme non sûre et devrait être mise à jour aussitôt que possible.

En août 2000 dans toutes les distributions Linux, il a été trouvé un nouveau problème de sécurité dans la commande optionnelle 'suidperl' (qui n'est pas installée par défaut) fournie avec les branches Perl 5.6, 5.005 et 5.004 (voir http://www.cpan.org/src/5.0/sperl-2000-08-05/). Les versions de maintenance 5.6.1 et 5.8.0 comblent ce trou de sécurité. La plupart, mais pas toutes, des distributions Linux proposent des correctifs pour cette vulnérabilité (voir http://www.linuxsecurity.com/advisories/) mais la bonne méthode consiste tout de même à passer au moins à Perl 5.6.1.

3 AUTEUR ET COPYRIGHT

4 TRADUCTION

4.1 Version

Cette traduction française correspond à la version anglaise distribuée avec perl 5.8.8. Pour en savoir plus concernant ces traductions, consultez http://perl.enstimac.fr/.

4.2 Traducteur

Traduction initiale : Emmanuel BOURG <smanux@dream.club-internet.fr>. Mise à jour : Roland Trique <roland.trique@uhb.fr>, Paul Gaborit <paul.gaborit at enstimac.fr>.

4.3 Relecture

Gérard Delafond.

5 À propos de ce document

Ce document est la traduction française du document orginal distribué avec perl. Vous pouvez retrouver l'ensemble de la documentation française Perl (éventuellement mise à jour) en consultant l'URL <http://perl.enstimac.fr/>.

Ce document PDF a été produit Paul Gaborit. Si vous utilisez la version PDF de cette documentation (ou une version papier issue de la version PDF) pour tout autre usage qu'un usage personnel, je vous serai reconnaissant de m'en informer par un petit message <mailto:Paul.Gaborit@enstimac.fr>.

Si vous avez des remarques concernant ce document, en premier lieu, contactez la traducteur (vous devriez trouver son adresse électronique dans la rubrique TRADUCTION) et expliquez-lui gentiment vos remarques ou critiques. Il devrait normalement vous répondre et prendre en compte votre avis. En l'absence de réponse, vous pouvez éventuellement me contacter.

Vous pouvez aussi participer à l'effort de traduction de la documentation Perl. Toutes les bonnes volontés sont les bienvenues. Vous devriez trouver tous les renseignements nécessaires en consultant l'URL ci-dessus.

Ce document PDF est distribué selon les termes de la license Artistique de Perl. Toute autre distribution de ce fichier ou de ses dérivés impose qu'un arrangement soit fait avec le(s) propriétaire(s) des droits. Ces droits appartiennent aux auteurs du document original (lorsqu'ils sont identifiés dans la rubrique AUTEUR), aux traducteurs et relecteurs pour la version française et à moi-même pour la version PDF.