Sobre Perl

Perl tiene unos 23 años hasta ahora. El lenguaje ha evolucionado desde una herramienta simple para la administración de sistemas tomando cosas de shell scripting y C (Perl 1) hasta convertirse en un poderoso lenguaje de propósito general (Perl 5), consistente, coherente, cambiando el paradigma de programación en general con la intensión de mantenerse por otros 25 años. (Perl 6)

Aun así, la mayoría de los programas en el mundo, escritos en Perl 5, aprovechan muy poco las características del lenguaje. Usted puede escribir programas en Perl 5 como si estuviera en Perl 4 (o Perl 3 o 2 o 1), pero los programas que aprovechan todo el increíble poder del mundo de Perl 5 que la comunidad ha inventado, y mejorado, son mas cortos, rápidos, mas poderosos, y fáciles de mantener que sus alternativas en anteriores versiones.

¿ Porque Larry creo Perl ?

Larry crea Perl a mediados de 1980, cuando intentaba producir reportes sobre la información ordenada jerárquicamente en archivos de la red de noticias USENET para un sistema de reporte de bugs, y awk no dio la talla. Entonces Larry, como buen programador perezoso que es, decide hacer una herramienta que pueda usar para resolver este problema y que pueda volver a usar para otro problema similar en otro lugar, en otro tiempo.

¿ Porque Larry simplemente no uso otro lenguaje ?

En este momento hay gran variedad de lenguajes, pero en esa época Larry no encontró un lenguaje que se adaptara a sus necesidades.

Si uno de los lenguajes de hoy, hubiese existido para la época, pues tal vez Larry lo hubiera usado para resolver el problema.

Necesitaba algo con la rapidez de codificación disponible en **Shell** o **AWK** y con algo del poder de las herramientas como **grep**, **cut**, **sort**, y **sed**, sin tener que recurrir a un lenguaje como **C**.

Perl, intenta llenar la brecha entre los lenguajes de bajo nivel como C, C++, y Assembler, y los lenguajes de alto nivel de la época como Shell Scripting. La programación a bajo nivel, usualmente es difícil y fea de escribir, pero rápida y sin limitaciones, es difícil superar la velocidad de un programa bien escrito a bajo nivel en una máquina dada. Y no hay mucho que se pueda hacer.

En el otro extremo, la programación convencional de alto nivel, tiende a ser lenta, dura, fea y limitada. Hay varias cosas que no puedes hacer con programación en shell o batch.

Perl es fácil, casi ilimitado, muy rápido y solo un poco feo.

Características principales de Perl

Perl es fácil

Cuando nos referimos que Perl es fácil, hablamos de que es fácil de usar, no es especialmente fácil de aprender. Si usted conduce un carro, que paso semanas o meses aprendiendo a conducir, entonces luego va a ser fácil de conducir. De igual manera cuando pase la misma o mas cantidad de horas aprendiendo Perl, entonces despues va a ser muy fácil de usar.

Perl es casí ilimitado

Hay muy pocas cosas que no podrás hacer con Perl. Seguramente usted no deseará escribir un controlador de algún dispositivo de interrupción a nivel de micro-kernel (A pesar de que ya alguien hizo esto). Pero la mayoria de las cosas que la gente común y no tan común necesita, la mayoria del

tiempo son buenas tareas para Perl, desde scripts muy pequeños, hasta aplicaciones de nivel industrial.

Perl es sobre todo rápido

Esto es porque no hay nadie en el desarrollo que no lo utilice, de modo que entonces todos queremos que Perl sea rápido. Si alguien quiere añadir una nueva característica que sea genial, pero que introduzca lentitud a otros aspectos, es casí seguro que Larry va a rechazar la nueva característica hasta que no se encuentre una manera de escribirla de manera eficiente.

Perl es un poco feo

Esto es cierto, el camello se ha convertido en el simbolo de Perl, desde la cubierta del legendario libro **The Camel Book**, conocido tambien como **Programming Perl by Larry Wall, Tom Christianesen, and Jon Orwant**. Los camellos tambien son un poco feos, pero trabajan duro, incluso en condiciones difíciles. Los camellos siempre están ahí siempre para hacer el trabajo a pesar de todas las dificultades, incluso cuando se ven y huelen mal e incluse le escupen. Perl es un poco así.

¿ Perl es fácil o difícil ?

Perl es fácil de usar, pero a veces difícil de aprender. Esto por su puesto es una gereralización. En el diseño de Perl, Larry tuvo que hacer muchas consideraciones. Cuando ha tenido la oportunidad de hacer algo más fácil para el programador a costa de ser más difícil para el estudiante, ha decidido casi todo el tiempo a favor del programador. Esto es así, porque usted va a aprender Perl una sola vez, pero lo vas a usar una y otra vez. Perl posee cualquier cantidad de comodidades que le permiten al programador ahorrar tiempo. Por ejemplo, la mayoria de las funciones tienen un valor predeterminado, con frecuencia, el valor por defecto es justamente la manera en la que usted desea usar la función. De esta manera, podrás encontra lineas de código en Perl como las siguientes:

```
while (<>) {
    chomp;
    print join("\t", (split /:/)[0, 2, 1, 5] ), "\n";
}
```

En su totalidad, sin usar los atajos y los valores por defecto de Perl, este trozo de código seria aproximadamente 10 o 12 veces mas largo, entonces tomaria mas tiempo para escribirlo y leerlo. Va a ser mas duro de mantener y depurar cuando tenga mas variables.

¿ Como se hizo Perl tan popular ?

Despues de jugar un rato con Perl, agregando cosas aquí y allá, Larry se lanzó a la comunidad de lectores de la Usenet, comunmente conocidad como **La Red**. Los usuarios de esta flota fugitiva que trabajaban en sistemas en todo el mundo (decenas de miles de ellos), le dieron retroalimentación, pidiendo a Larry, maneras de hacer esto, aquello o lo otro, muchos de los cuales Larry nunca había imaginado en su hasta el momento poco manejo de Perl.

Pero como resultado, Perl creció, y creció y creció. Creció en características. Creció en portabilidad. Lo que antes era un lenguaje poco disponible en solo un par de sistema tipo Unix, ahora tiene miles de páginas de documentación en linea gratis, docenas de libros, y miles de listas de correo en cientos de idiomas con un número incontable de lectores, e implementaciones en varios sistemas de uso cotidiano hoy en día, y adicionalmente un curso como este.

¿ Que esta pasando con Perl ahora ?

Durante estos días, Larry Wall no escribe código, pero el sigue guiando el desarrollo y toma las desiciones importantes. Perl es mayormente mantenido por un grupo de resistentes y valientes personas llamadas **The Perl 5 Porters**. Puede suscribirse a la lista de correo en **perl5-porters@perl.org** para seguir el trabajo de esta gente y sus debates. Al momento de escribir esta documentación, muchas cosas estan pasando con Perl. Durante los últimos años, muchas personas han estado trabajando en la próxima versión de Perl: Perl 6.

No tire a un lado Perl 5, este sigue siendo la versión actual y estable. No se espera una version estable de Perl 6 durante un rato largo. Perl 5 hace todo lo de siempre, y siempre sera así. Perl 5 no va a desaparecer cuando Perl 6 salga. Y la gente puede seguir usando Perl 5 durante muchos años mas.

¿ Que es CPAN ?

CPAN es **Comprehensive Perl Archive Network**, su primera parada para Perl. Posee el código fuente de Perl, listo para instalar y los **ports** para todos los sistemas no Unix, ejemplos, documentación, y extensiones del lenguaje.

CPAN es replicado en cientos de máquinas al rededor del mundo, comenzando en **http://search.cpan.org** o **http://kobesearch.cpan.org** para navegar o buscar un paquete. Si no tienes acceso a la red, todo CPAN puede caber en 8 o 10 GB. Es importante mencionar que debido a los cambios diarios en CPAN, un mirror con dos años sin actualizares es definitivamente una antiguedad.

¿ Como hacer un programa en Perl ?

Ya es tiempo de hacer esta pregunta (aunque usted no la halla hecho aun), Los programas de Perl son archivos de texto plano, usted puede crearlos y editarlos en su editor de textos preferido, yo personalmente prefiero **VIM**. (Usted no necesita entornos de desarrollo especiales, aunque hay disponibles editores comerciales de varios proveedores. Nunca hemos utilizado uno de estos lo suficiente como para recomendarlo). Por lo general debería usar un editor de texto pensado para programadores, en lugar de un editor de textos convencional. ¿ Cual es la diferencia ? Bueno, un editor pensado para programadores le va a permitir hacer cosas que necesitan los programadores, como sangrado automático o subrayado de bloques de código, o indicar automáticamente la llave de cierre de un bloque de código. En sistemas operativos tipo Unix, los editores de texto mas populares son **Emacs** y **Vim**. BBEdit y TextMate son buenos para Mac OS X, mucha gente ha hablado muy bien sobre UltraEdit y PFE en Windows. La página del manual perlfaq2 tiene una lista de otros editores que también pueden usarse.

Un programa simple

Según la regla mas antigua de los libros de computación, cualquier libro sobre algún lenguaje de programación que tenga raices Unix, va a comenzar con **Hola, Mundo**, por ejempo en Perl seria así:

```
#!/usr/bin/perl
print "Hello, world!\n";
```

Sobre los ejercicios y sus respuestas

Los ejercicios se encuentran al final de cada capítulo, una vez que se desarrolle cada capítulo, se tomaran 45 a 50 minutos para resolver los ejercicios correspondientes a cada capítulo.