Perl para apresurados

Juan Julián Merelo Guervós

Historial de revisiones

Revisión 1.0 Jul 2006
Preparando la primera versión para el curso de Extremadura

1. ¿Quién eres tú?

Eso mismo te estarás preguntando, que quién diablos eres, que a qué dedicas el tiempo libre, todo eso. Así que te vamos a echar una mano. Supongo que ya sabes programar, que el concepto de *variable* no va para ti asociado a la nubosidad ni el de *bucle* a la cabeza de Nellie Olleson. Puede que conozcas el C, sólo para precavidos, o hables con lengua de serpiente (pitón (http://www.python.org)), o incluso que el símbolo mayor y menor van para tí asociados de forma indisoluble a un acrónimo capicúa (http://www.php.org).

Vamos, que puede extrañarte las formas ignotas en las que un nuevo lenguaje de programación repite cachos de código o mete valores en variables o representa listas de datos, pero los conceptos en sí no son nada nuevo para tí. A tí, pues, va dirigido este mini-tutorial.

Supongo también que tienes prisa. Si no, no estarías leyendo este tutorial para *apresurados*. Estarías leyendo uno titulado, por ejemplo, *Perl para los que tienen todo el tiempo del mundo*. Es decir, que es necesariamente breve, con la idea de poder ser impartido (y espero que asimilado) en unas dos horas. Igual no te da tiempo a teclear todos los ejemplos de código, pero este ordenador que estás mirando tiene una cosa maravillosa llamada "corta y pega" con la que sin duda estás familiarizado, y que podrás usar para tu provecho y el de la Humanidad.

Y quizás todavía no lo sabes, pero *necesitas* saber Perl. Para vigilar ese fichero de registro y crear alertas que te avisen de lo inesperado. Para ese CGI terriblemente complicado. Para convertir una página web demasiado compleja en algo que también es complejo, pero que puedes leer con tu lector de cosas complejas favorito. Para hacer lo que siempre quisiste hacer: escribir poesía () en tu lenguaje de programación favorito. En fin, donde quiera que haga falta convertir cosas en otras cosas, ahí hace falta saber Perl.

> **Sugerencia:** Y con ello damos entrada a la primera *flamewar* de este tutorial, que es donde tú, que estás entre el público, dices aquello de *Pues yo hago todo eso, y más, en (Fortran|Postscript|Haskell)*. Que vale, que si. Los lenguajes de programación son universales. Se puede hacer de todo con ellos. Y siempre es más fácil hacer algo en el lenguaje que uno conoce mejor. Pero al menos tendrás más donde elegir, ¿no?

Sobre todo, que no cunda el pánico. Y no te olvides de la toalla (http://es.wikipedia.org/wiki/Dia_de_la_Toalla).

2. Todo listo para despegar

> Si ya has usado algún lenguaje de scripting, lo más probable es que te aburras como un bivalvo en esta sección. Así que ahórrate un bostezo y pasa directamente a la siguiente. O si no descárgate los fuentes (http://www.cpan.org/src/latest.tar.gz) y echas un ratillo compilándolo en silencio, para no desmoralizarme a la parroquia.

Lo primero que necesitas en tu lista de comprobación son las cualidades de todo programador en Perl: la pereza, el orgullo y la curiosidad. No te preocupes si no tienes ninguna de ellas, las irás adquiriendo con el tiempo. Sobre todo la pereza. Y una cierta habilidad de entender lenguas muertas como el caldeo y el dálmata.

Segundo, necesitas amar a los camélidos. El Perl no es como esos otros lenguajes que incitan a la avaricia a través de la adoración de las piedras preciosas (http://www.ruby-lang.org), o a la hiperactividad por ingestión de bebidas excitantes (http://www.java.com). Los camellos son buenos. Los camellos son útiles. Llevan cosas encima. Tienen joroba. Amemos a los camélidos (las llamas también son camélidos (http://es.wikipedia.org/wiki/llama)).

No menos importante es tener un ordenador con sistema operativo. Incluso sin él (http://www.windows.com). Ejecuta lo siguiente para saber si lo tienes: **`perl -v`** a lo que el ordenador debidamente contestará algo así:

Figura 1. Contestación de un ordenador educado a `perl -v`

```
This is perl, v5.8.7 built for i486-linux-gnu-thread-multi
(with 1 registered patch, see perl -V for more detail)

Copyright 1987-2005, Larry Wall

Perl may be copied only under the terms of either the Artistic License or the
GNU General Public License, which may be found in the Perl 5 source kit.

Complete documentation for Perl, including FAQ lists, should be found on
this system using ‘man perl’ or ‘perldoc perl’.  If you have access to the
Internet, point your browser at http://www.perl.org/, the Perl Home Page.
```

si es que está instalado. Si no lo está, es poco probable que conteste eso. Incluso imposible. Dirá algo así como `bash: perl: command not found` e incluso pitará. El muy desagradable.

No hay que dejarse descorazonar por tal eventualidad. Encomendándonos al *Gran Camélido*, y sin necesidad de ver una vez más Ishtar, diremos en voz alta "Abracadabra" mientras que escribimos **`sudo`**

`yum install perl` o bien **`sudo apt-get install perl`** Si es que están en un linux no-debianita (en el primer caso) o en uno debianita (en el segundo). Habrá gente que incluso lo haga sin necesidad de bajarse del ratón. Pero los apresurados no usan el ratón salvo que sea estrictamente necesario. Que no es el caso. En otros sistemas operativos, lo mejor es ir a Perl.com (si es que no has ido todavía) (http://www.perl.com) y te bajes la versión compilada.

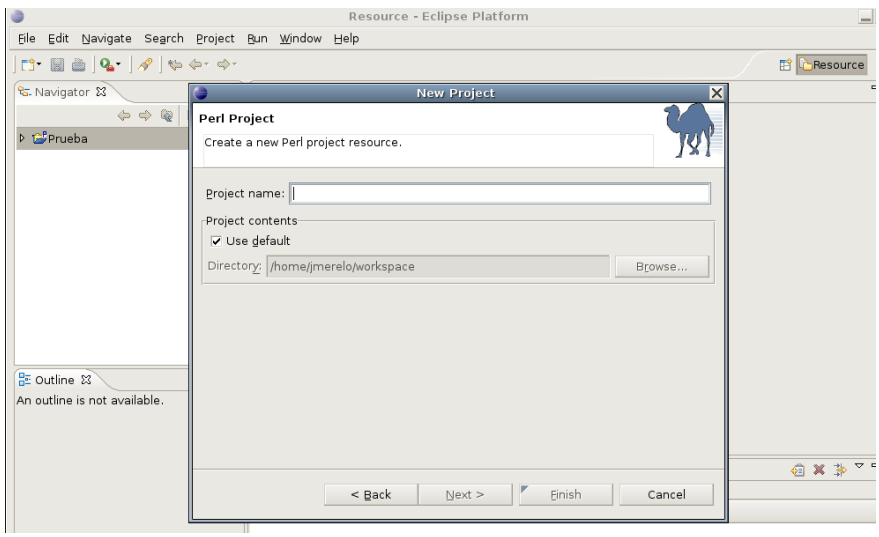
También puedes compilarlo tú. Pero no creo que lo hagas, porque eres un apresurado, y la compilación no está hecha para los apresurados (si eres usuario de Gentoo (http://www.gentoo.org), es el momento de abandonar este tutorial).

Lo que tienes o has instalado es un intérprete de Perl. Perl es generalmente un lenguaje interpretado, con lo que no hace falta ningún encantamiento intermedio para pasar de un programa escrito en Perl a su ejecución. Si te hará falta un editor. No *un* editor. *El* editor.

> **Sugerencia:** Los que apoyen al ínclito (x)emacs de este lado del *flamewar*, los que se queden con el sólido pero escuálido vi(m), de este otro lado. Los que estén con kate, jot, o incluso el kwrite, que elijan armas y padrinos y que pidan hora.

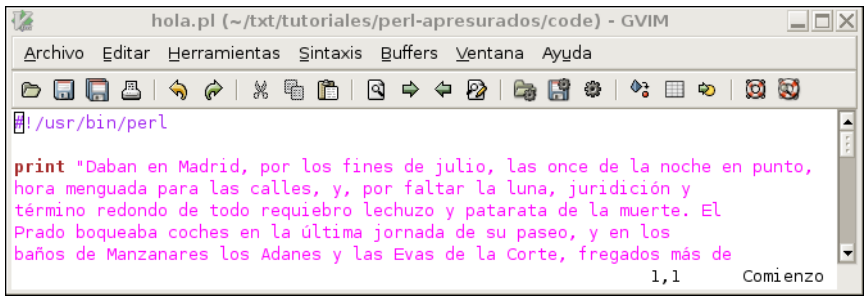
Vuelvo contigo entre el fragor de la batalla hablarte de otras opciones. No es que haya muchas, pero hay alguna. Por ejemplo, puedes usar el conocido entorno Eclipse () con el plugin EPIC (http://e-p-i-c.sourceforge.net) para desarrollar proyectos, como se muestra en la figura siguiente.

Figura 2. Iniciando un proyecto en Perl con EPIC/Eclipse



Otros entornos de desarrollo, como PerlIDE o Komodo, o bien no siempre funcionan o bien son de pago. Si consigues que te lo compren, suertudo de ti. Si no, apoya proyectos de software libre. Suficientes personas han estado desarrollando sobre esos entornos durante el suficiente tiempo como para que presenten la sana apariencia que se muestra en la figura de abajo.

Figura 3. Editando un programilla con gvim



Un editor decente tiene que tener colorines. Y también cerrar paréntesis. Ninguno va a evitar que cometas errores, pero va a hacértelo lo más complicado posible.

> **Ejercicios.** ¿Tienes un intérprete de Perl instalado en tu sistema? ¿Tienes un editor (chulo, si es posible) para editar programas en Perl? Si la respuesta a alguno de ellos es *no*, ¿a qué esperas para tenerlos? Venga, te espero.

3. Comenzando una nueva carrera

Si has llegado hasta aquí, supongo que se te llevarán todos los diablos, porque con la hora que es, las camas están sin hacer y lo que s dice picar código, todavía no has picado nada. Y eso está bien: hay que convertir esa rabia en energía creativa, y aprovechando que uno de los diablos que se te llevan es cojuelo, escribir el siguiente fragmento de literatura:

```
#!/usr/bin/perl                                                    ❶
print "Daban en Madrid, por los fines de julio, las once de la noche en punto..."; ❷
print "\n";                                                        ❸
```

❶ Tratándose de diablos, lo mejor es usar los conjuros lo antes posible. En esta línea, clásica de los lenguajes interpretados y denominada *shebang* se escribe el camino completo donde se halla el intérprete del lenguaje en cuestión. Si está en otro sitio, pues habrá que poner otro camino. Por ejemplo, podría ser **#!/usr/local/bin/perl** o bien **#!/usr/bin/perl6.0.por.fin**. O **#!perl** y que se busque la vida. Si se trabaja (es un decir) en Windows, esa línea no es necesaria, pero es conveniente

para que el programa sea compatible con otros sistemas operativos. Cuando un Unix/GNU/Linux decente y trabajador encuentra esa línea, carga ese programa y le pasa el resto del fichero para que lo interprete.

❷ Aquí se imprime, con el *nihil obstat* obtenido previamente. Obsérvense las comillas y el punto y coma. Las órdenes en Perl acaban con un punto y coma, para que quede bien claro dónde acaban y se puedan meter varias sentencias en una sola línea, con el objetivo de crear programas innecesariamente ofuscados. Lo que no se puede hacer en *esos otros lenguajes*. El `print` es herencia de aquellos primeros tiempos de los ordenadores, cuando el único periférico de salida era un convento de monjes trapenses dedicados a la sana tarea de copiar textos (y que se quedaron sin trabajo cuando el señor Hewlett se unió al señor Packard y crearon la impresora). En aquella época, la salida de un programa venía encuadernada en piel de cabrito y con todas las primeras letras de párrafo bellamente miniadas. Ah, tiempos aquellos, de los que sólo nos queda el nombre de una orden.

❸ Y aquí pasamos a la línea siguiente. Borrón y cuenta nueva. Se acabó lo que se daba. Si ya conoces algún lenguaje de programación, que se supone que lo conoces, pillín, porque te lo he preguntado en la primera sección, no hace falta que te explique que `\n` es un *retorno de carro*, ¿verdad?[1]

4. Viéndole las tripas al producto

Notas

1. Lo que es recuerdo también de aquellos mismos tiempos en que STDOUT era un convento, en el que para seguir en la página siguiente tenían que esperar que retornara el carro que les traía las pieles de becerro curtidas en las que escribían lo que el programador les ordenaba