1. Introducción

Una de las grandes ventajas de Perl es que a lo largo del tiempo se han ido desarrollando una gran cantidad de módulos y bibliotecas, de manera que hay mucho código disponible para ser utilizado en este momento y podemos crear aplicaciones que realicen tareas muy potentes con muy pocas líneas de código. A continuación, vamos a ver tres ejemplos muy distintos.

2. Ejemplos

2.1. Contador de visitas

En el primer ejemplo, simplemente vamos a acceder a un archivo y devolver un dato. Pero lo vamos a hacer de tal manera que esto se puede incluir dentro de un HTML, es por eso que utilizamos la biblioteca CGI.

CGI significa *Common Gateway Interface*. La idea de los programas CGI es que son scripts de perl que general código HTML, el código que generan puede ser uno muy complejo o uno muy sencillo. En el ejemplo que vamos a ver, vamos a tener una salida muy simple, pero la biblioteca CGI nos permite generar HTMLs mucho más complejos, con muy poco código.

CGI se utiliza para generar páginas web dinámicas, páginas que muestren un contenido distinto según determinados patrones. El objetivo es similar al de lenguajes de programación como PHP o ASP, pero en el caso de CGI, no hay código HTML mezclado con código Perl, sino que todo es código Perl.

Dicho esto, en este ejemplo, simplemente usamos la función **print header** que envía un encabezado imprescindible para cualquier página web.

El resto de la función lo que hace es abrir un archivo, leer la primera línea, incrementar ese valor, guardarlo y devolverlo.

Es interesante notar la forma en que se abren los archivos para lectura o escritura en Perl. Si el archivo se abre para lectura, se debe colocar un `<` antes del nombre. Si se abre para escritura, se debe colocar un `>`.

Una vez abierto, una sencilla asignación, utilizando la forma `<ARCHIVO>` lee la primera línea y la guarda en la variable. Si se volviera a realizar esta acción, el valor almacenado en la variable sería el de la segunda línea.

Y de la misma manera, para escribir en un archivo se utiliza simplemente el comando **print** con el nombre de archivo como primer parámetro y la cadena a escribir como segundo parámetro.

2.2. Web Server

En este ejemplo usamos un módulo que permite devolver páginas web que nos piden. El servidor permanece ejecutándose eternamente, entregando las páginas que pidan los usuarios (siempre que sean páginas válidas).

Si una página no se encuentra, devuelve el error 404 (no encontrada), y si una página no puede ser leída por el webserver, devuelve error 403 (no permitida).

Algunas mejoras que se le podrían hacer a este servidor: que pueda entregar más de una página a la vez, haciendo un *fork* en el momento en que recibe una conexión; que almacene la información de las páginas que entregó o no entregó en un archivo de texto, en lugar de mostrarlo por pantalla, o agregarle soporte de cgi, asi podemos poner en línea nuestros programitas.

2.3. Navegador Web

En este ejemplo utilizamos el módulo Perl-GTK.

GTK significa Gimp ToolKit, es una biblioteca creada por los programadores del programa para manipulación de gráficos The Gimp, que luego se extendió a la mayoría de los programas en GNU/Linux y también en Windows. La biblioteca GTK original fue hecha en lenguaje C. La flexibilidad de Perl permite adaptar todo este código (que está disponible por ser Software Libre) a Perl, de una manera muy sencilla, ya que no hay que reescribir todo, sino simplemente crear un *conector* que una la biblioteca GTK con el código Perl.

GTK es una biblioteca creada para hacer más sencillo el desarrollo de aplicaciones gráficas. Está *orientada a eventos*, es decir que el usuario genera una serie de eventos (click, enter, mover el mouse, etc), y el programa lleva a cabo distintas acciones según los eventos que se van sucediendo.

En este ejemplo, utilizamos la biblioteca Perl-GTK en conjunto con el módulo GTKXmHTML, que permite mostrar por pantalla una página web, de una forma muy sencilla.

Además, en este ejemplo se utilizan las llamadas de los eventos para poder avanzar a la siguiente página cuando el usuario hace click en un link. En este caso, se trata del evento **activate**.

El ejemplo en sí es muy simple, sirve para navegar, seguir links. Algunos agregados interesantes podrian ser agregarle soporte de imagenes, revisar el soporte de forms.