

Scripts Perl que envían mensajes de log a Twitter

# TWITTER PARA GEEKS



El servicio Twitter puede ser una plataforma para mensajes sin interés, o podemos automatizar el acceso con una API y usarlo de maneras jamás imaginadas por sus creadores.

**POR MICHAEL SCHILLI**

¿Por qué demonios alguien querría usar un teléfono móvil para comunicar al mundo que está comiendo pizza con unos amigos, ha bajado al gimnasio o acaba de llegar a algún sitio en avión?

El microblogging, como el que ofrece Twitter o la variante libre identi.ca, tiene algunos efectos secundarios realmente sorprendentes. Si queremos averiguar cuáles son los temas que despiertan más interés para la humanidad en este mismo momento, ninguna búsqueda de Google ni de la Wikipedia nos será de utilidad. Eche un vistazo a la página principal del servicio Twitter (véase la Figura 1) y verá a la gente twitteando acerca de los escándalos políticos, fenómenos naturales, la última película o eventos deportivos a menudo mucho antes de que la editorial del periódico o los canales de televisión siquiera se den cuenta de lo que está pasando.

## Un Gigante Frágil

El hecho de que Twitter funcione es algo muy cer-

cano a un milagro. El servicio ofrece sólo una mínima funcionalidad, y su infraestructura es tan frágil que los cortes del servicio suelen ser algo habitual. Aún así, 6 millones de usuarios alimentan incansablemente a la frágil bestia con noticias, infundiendo continuamente savia nueva en forma de información fresca, y manteniéndolo vivo desde 2006. El equipo de Twitter es reactivo a añadir nuevas funcionalidades. De hecho, esto generalmente sólo ocurre cuando los fieles seguidores de Twitter descubren soluciones creativas para añadir con dificultad nuevas funciones al margen del gigante.

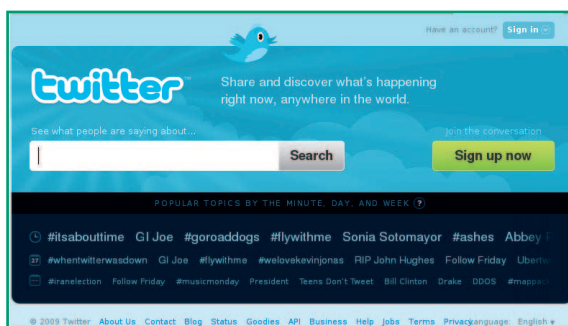


Figura 1: ¿Por qué demonios necesitamos a Twitter? La página principal del popular servicio de Internet muestra de lo que está hablando la gente.

En realidad, Twitter no ofrece mucho: los usuarios con una cuenta pueden enviar mensajes de texto de hasta 140 caracteres al servicio, el cual los publica tanto en la Public Time Line como bajo la cuenta del usuario. Si combinamos esto con un teléfono móvil con capacidad de texto o de Web, ya tenemos un estilo de comunicación algo superficial pero oportuno.

Si estamos interesados en lo que dice otro usuario, podemos registrarnos como seguidor de ese usuario. Esto es como hacernos amigos de alguien, ya que se encaminan automáticamente sus mensajes hacia nosotros. Generalmente, un usuario sigue a muchos otros usuarios; e inversamente, otros usuarios le seguirán. Los desequilibrios son indicadores de algún problema. Por ejemplo, los spammers suelen seguir a muchos usuarios, pero nadie los sigue ni tiene interés en leer sus mensajes.

A pesar de que Twitter es gratuito por ahora, es muy probable que aplique publicidad en algún momento, simplemente para ganar algo de dinero. Su clon libre identi.ca parece ligeramente diferente, pero nos ofrece exactamente la misma funcionalidad, incluyendo detalles técni-

**Listado 1: unfollow-all**

```

01 #!/usr/bin/perl -w
02 use strict;
03 use Net::Twitter;
04
05 my $nt = Net::Twitter->new(
06     traits =>
07     [qw/API::REST/],
08     ssl => 1,
09     username =>
10     "perlsnapshot",
11     password =>
12     "*****",
13 );
14
15 my $friends =
16 $nt->following();
17
18 for my $friend (@$friends) {
19     print
20     "$friend->{screen_name}\n";
21     $nt->destroy_friend(
22     $friend );
23 }

```

cos como el diseño URL (véase la Figura 2). Esto significa que podemos apuntar el módulo Perl Net::Twitter, por ejemplo, no a Twitter, sino a identi.ca, mediante un único parámetro que reemplace `http://twitter.com` por `http://identi.ca`.

**Lo Básico**

El poco interés de Twitter por introducir nuevas funcionalidades lleva a la gente a hacer un uso abusivo de los 140 caracteres disponibles de cada mensaje como una especie de lenguaje de programación para probar nuevas funcionalida-

des. Los “Retweets” (mensajes que merecen la pena ser mencionados y que los seguidores de un usuario reenvían a otros seguidores) son un ejemplo de esto. Como el Twitter original no tenía esta funcionalidad, algunos usuarios inteligentes añadían repetidamente la cadena `RT @nombreusuario` a sus mensajes, hasta que Twitter finalmente cedió y reconoció el “estándar”.

Twitter sólo nos ofrece los bloques constructivos en bruto, como se refleja en la ligera API. Sorprendentemente, O’Reilly publicó un tomo de 400 páginas sobre esta materia [2] describiendo la interfaz de programación, incluyendo un corto tutorial sobre programación Web, e introduciendo todo un conjunto de exitosas aplicaciones de terceros para Twitter.

**Aniquilados**

Aunque podríamos usar la API para programar otro cliente con interfaz gráfica o interfaz basada en texto, ésta es también útil para limpiar algunos errores de Twitter. Cuando creé la cuenta de pruebas `perlsnapshot`, debí presionar el botón equivocado, porque de pronto aparecieron por lo menos 20 amigos, de los que curiosamente jamás había oído hablar. Borrarlos uno a uno con mi navegador habría sido una pesadilla, pero los aniquilé al instante con el script mostrado en el Listado 1, que programé en menos de cinco minutos, y al ejecutarlo contra la lista de falsos amigos los saqué de mi vida para siempre.



Figura 2: El clon libre compatible con Twitter, identi.ca.

**¡Mi Contraseña se Fue!**

Si le preocupa su seguridad, probablemente se haya dado cuenta de que el código incluye la contraseña necesaria para que el script se loguee en el servicio de Twitter. La opción `ssl` evita tener que enviar la contraseña en texto claro. En su lugar, el módulo Net::Twitter de Perl se comunica con las URLs HTTPS de Twitter. Pero con aplicaciones que aparecen de la nada como setas, la pregunta es: ¿cómo puedo evitar tener que facilitar mi contraseña a terceros? Imaginemos que un eslabón débil de la cadena pudiese revelar nuestra contraseña: sería muy preocupante.

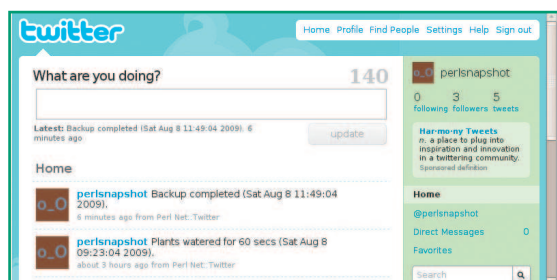
Un reciente añadido de Twitter, OAuth, resuelve este problema. Este protocolo abierto proporciona a diferentes aplicaciones los permisos para una cuenta de usuario. En lugar de reenviar nuestro nombre de usuario y contraseña a los diferentes proveedores, esperando que nadie haga mal uso de nuestra generosidad, OAuth asigna a cada aplicación un token único. El token puede revocarse en

**Listado 2: TwitSend.pm**

```

01 package TwitSend;                               $opts->{message};                               "$home/.twitsend" );
02 use strict;                                     18     }                                           33
03 use Net::Twitter;                               19     }                                           34     my $nt = Net::Twitter->new(
04 use YAML qw(LoadFile);                         20     if(! defined $message) {                   35         traits =>
05 use base qw(Exporter);                         21         die "No message given!";               36         [qw/API::REST/],
06 our @EXPORT_OK = qw(twit);                     22     }                                           37         ssl => 1,
07                                                 23     }                                           38         # identica => 1,
08 #####                                           24     if(length $message > 140) {                 39         username =>
09 sub twit {                                       25         die "Message needs to be <           "perlsnapshot",
10 #####                                           26         140 chars";                             39         password =>
11     my($message) = @_;                          27     }                                           40         $yaml->{password},
12                                                 28     }                                           41     );
13     my $opts = {};                              29     my $home = $opts->{home};                     42     $nt->update( $message );
14                                                 30     ($home) = glob "~" unless                   43 }
15     if(ref($message) eq "HASH")                 31     defined $home;                               44
16     {                                           32     my $yaml = LoadFile(                       45 1;
17         $opts    = $message;
18         $message =

```



**Figura 3: Usamos Twitter como archivo de log para monitorizar un servidor.**

cualquier momento sin afectar a otros proveedores, que están usando tokens diferentes.

El módulo `Net::Twitter` de Perl soporta OAuth. Sin embargo, Twitter insiste en que nos registremos en la aplicación [3] antes de darnos la Token Consumer Key y Consumer Secret. Esta es la teoría, todo ello, en cualquier caso, si funciona el servicio, lo que no sucedía en el momento de escribir este artículo.

## Loguearse con Twitter.

El Listado 2 muestra otra aplicación para la API de Twitter: un módulo Perl que podemos incrustar en otros procesos Perl para enviar mensajes a Twitter: la función `twit()`. De la manera habitual en Perl, la función acepta tanto un mensaje en forma de cadena como una referencia a un hash con el campo `message`. Un segundo parámetro, `home`, apunta al directorio de usuario, por si acaso fuese el usuario root el que ejecuta el programa (por ejemplo, un script de backup). `Twitsend.pm` aguarda encontrar un archivo de configuración YAML en el directorio del usuario denominado `.twitsend` que contiene la contraseña de la cuenta de Twitter. De esta manera, la contraseña sólo está en un lugar y no en múltiples scripts.

## En mi Balcón

Mi sistema de riego controlado por Perl [4] ha proporcionado un incalculable y valioso servicio a las plantas de mi balcón desde hace más de dos años. Sin embargo, me gusta echarle un vistazo a lo que está haciendo cuando estoy fuera, así que ¿por qué no dejar que Twitter muestre su actividad de manera que pueda verificarlo desde cualquier Internet Café o mediante mi teléfono móvil?

Tras sólo en cinco minutos, ya era capaz de comunicar el módulo `TwitSend.pm` con el script `water` del sistema

de riego que presenté en un artículo anterior [4]. A continuación llamé a la función `twit()` con una cadena de mensaje y la ruta al directorio que contenía el archivo YAML con la contraseña de la cuenta `perlsnapshot` de Twitter.

La implementación se muestra en el script de ejemplo `twitsend` (Listado 3), que

publica los resultados en Twitter, como se muestra en la Figura 3. La duración del riego para las plantas en un día caluroso de agosto lleva unos 60 segundos, y la copia de seguridad de mi sistema de desarrollo se completó a las 11:49am. Para las últimas actualizaciones, no dude en visitar [twitter.com/perlsnapshot](http://twitter.com/perlsnapshot).

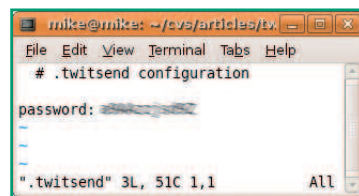
Este método es similar al usado recientemente por mi compañero en la revista, Charly Kühnast, con el cliente `ttytter` [5]. El script visto aquí sólo necesita un archivo YAML `.twitsend` en el directorio de usuario (por ejemplo `/home/mike`), con los permisos a 0500, para asegurar que el archivo sólo se puede leer y escribir por el propietario de la cuenta. Contiene una única línea tal que `password: ****`, que define la contraseña de Twitter (véase la Figura 4).

Si prefiere usar el servicio libre `identi.ca` en su lugar, descomente la directiva `identica` en la línea 37 del módulo `TwitSend.pm`. Esta directiva dirige los scripts al clon `identi.ca` en lugar de a Twitter.

Tenga en cuenta también que Twitter ignorará de forma silenciosa mensajes

### Listado 3: twitsend

```
01 #!/usr/bin/perl -w
02 use strict;
03 use MyTwitSend qw(twit);
04
05 twit({
06   message => "This is yet
07   another test message.",
08   home    => "/home/mike",
09 });
```



**Figura 4: El archivo de configuración de Twitsend.pm, .twitsend, que configura el usuario y contraseña para la cuenta de Twitter.**

idénticos. Una tarea cron que envíe un mensaje como “Copia realizada” todos los días no funcionará, pero si añadimos la fecha sí lo hará.

Si otros procesos usan twitter con cuentas diferentes, podemos usar nuestra cuenta personal para seguir todos o algunos de ellos. Cuando volvamos de vacaciones, sólo tenemos que hacer `unfollow` en la página Web de Twitter para deshabilitar el log del flujo de mensajes proporcionado por nuestros amigos.

La API, que es accesible vía `Net::Twitter`, también nos permite borrar tweets, realizar funciones administrativas como `follow()`, `unfollow()` o `block()`, enviar mensajes privados, interceptar el stream de mensajes públicos y hacer muchas otras cosas. Twitter restringe el acceso a 100 intentos por hora y monitoriza tanto las IPs como las cuentas.

## Instalación

Desafortunadamente, `Net::Twitter` y la larga lista de módulos dependientes de CPAN no están disponibles como paquetes de Ubuntu. Sin embargo, una llamada mediante la shell de CPAN (`perl -MCPAN -eshell`) nos permitirá instalar los módulos dependientes necesarios con sólo teclear `install Net::Twitter`. Debido a que `Net::Twitter` también carga Moose y soporta OAuth, puede ser aconsejable tomarse un descanso mientras se descarga.

## RECURSOS

- Listados de este artículo: <http://www.linux-magazine.es/Magazine/Downloads/60>
- Makice, Kevin. Twitter API: Up and Running: Learning How to Build Applications with the Twitter API. O'Reilly, 2009
- Aplicación de Registro de Twitter OAuth: [http://twitter.com/oauth\\_clients](http://twitter.com/oauth_clients)
- “Bricomanía” por Michael Schilli, Linux Magazine nº 30, [http://www.linux-magazine.es/issue/30/046-049\\_PerlLM30.crop.pdf](http://www.linux-magazine.es/issue/30/046-049_PerlLM30.crop.pdf)
- “Twittering Machine” por Charly Kühnast, Linux Pro Magazine, Agosto 2009, pág. 59