Getting things done with Evernote and Perl

Unforgettable

To reanimate temporarily frozen projects, users of "Getting Things Done" sort labeled folders by date and regularly check these "Tickler files." In contrast, Perl and Evernote automatically send reminders to users. *By Mike Schilli*

The "Getting Things Done" (GTD) productivity method by best-selling author David Allen [2] promises "mind like water," the spiritual peace and extreme flexibility of a Karate fighter in mastering everyday life. The basic rule: Do not burden yourself with tasks for which you cannot immediately handle the next step; instead, file them in an ordering system (Figure 1).

If You Like Paper …

The hanging files have labels for the days of the month and the months of the year. If a colleague announces a result for the 14th of a month, GTD disciples write a slip of paper with the key figures and throw it into the folder with the number 14. And, to start planning your summer vacation in January, because there are cheap flights, a slip of paper with the URL of the online booking system ends up in the folder with the "January" tab. If you check the folders regularly, you will notice that certain tasks exist for the next day or the next month. To the amazement of the unorganized world around them, GTD followers take care of these tasks punctually and reliably examine whether promised deadlines really were observed.

Automating with Evernote

As I already mentioned in a previous article [3], the free basic version of the Evernote [4] service is a blessing if you want to optimize daily tasks in line with GTD. The user defines an Inbox as "`00-Inbox`" so that Evernote sorts the folder right to the top. Into the tray (Figure 2) go all the inquiries for which the user then determines the next processing step. If the step takes less than two minutes, the user will tackle it immediately; otherwise, the task is dumped into a folder for a specific project.

You can set up a tickler system with Evernote that is similar to slips of paper in hanging files. The "`01-Tickler`" notebook contains individual entries with the action date in a YYYY-MM-DD format in the subject line. With the Evernote API, a cron job that runs once a day opens the Tickler notebook, wanders through all the entries, and checks to see whether a dated entry is due the next day (Figure 3). If so, the script pushes the notice into the user's Inbox, and the user is happy to see that they can now complete the next step of a miniature project.

Planning with Tickler

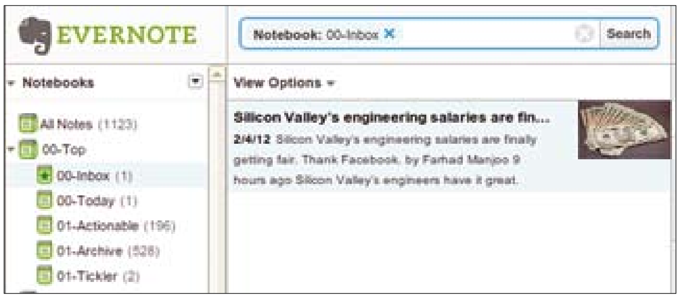
The entry "2013-01 plan summer holidays" reminds me to book a flight for my vacation in August as early as January 2013, and the cronjob pulls this entry out of the Tickler notebook on December 31, 2012, and puts it in my Inbox so that I can tackle a new "Holiday plans" project and start the action ("check out Lufthansa offers").

And, the Tickler entry, "2012-04-14: "Smith has finished the Linux version"

MIKE SCHILLI

Mike Schilli works as a software engineer with Yahoo! in Sunnyvale, California. He can be contacted at *mschilli@perlmeister.com*. Mike's homepage can be found at *http://perlmeister.com*.

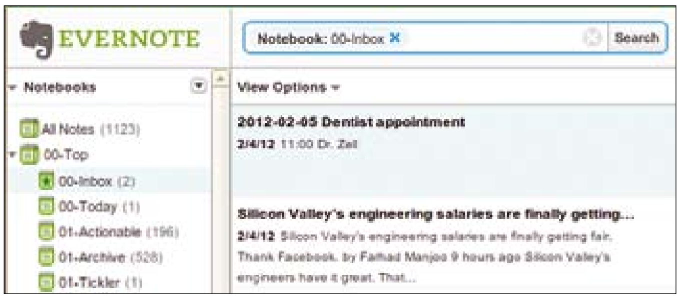
Figure 1: A tickler system for paper notes.



All Notes (1123)
00-Top
00-Inbox (1)
00-Today (1)
01-Actionable (196)
01-Archive (528)
01-Tickler (2)

Silicon Valley's engineering salaries are fin...
2/4/12 Silicon Valley's engineering salaries are finally getting fair. Thank Facebook. by Farhad Manjoo 9 hours ago Silicon Valley's engineers have it great.

Figure 2: The user's Inbox initially displays only an article captured from the web …



All Notes (1123)
00-Top
00-Inbox (2)
00-Today (1)
01-Actionable (196)
01-Archive (528)
01-Tickler (1)

2012-02-05 Dentist appointment
2/4/12 11:00 Dr. Zell

Silicon Valley's engineering salaries are finally getting...
2/4/12 Silicon Valley's engineering salaries are finally getting fair. Thank Facebook. by Farhad Manjoo 9 hours ago Silicon Valley's engineers have it great. That...

Figure 3: … but after running the Tickler cron job, tomorrow's appointment with the dentist ends up in the Inbox.

automatically falls into my Inbox on the evening of April 13, which allows me to remind the very surprised colleague about the release deadline they had promised to keep weeks earlier.

Thanks to the Evernote Web API, implementing the Tickler is pretty straightforward. A previous installment of this Perl column [3] details how the Thrift protocol works with Perl and how application programmers can pick up an application key from the Evernote website. Equipped with the credentials, developers can start running their scripts against the Evernote sandbox, and once all bugs have been ironed out, apply for access to the production server.

In Listing 1, the `BEGIN` block initially switches to the `$Bin` directory, where the script resides, to ensure that it will be able to find the automatically generated thrift modules in the `gen-perl` subdirectory when it starts the cron job. The CPAN local::lib module ensures that it finds the CPAN modules installed in the user's home directory. Line 27 initializes `Log4perl`, which writes debug instructions to a logfile to tell us what the script is doing while it's running. This step is particularly useful if you use a cron job to launch a script; logging helps you find the causes of errors and eradicate bugs because the process is not connected to any terminal. Besides the `$DEBUG` logging level, it also limits output to the `"main"` category to limit logging to the main program and suppress output of CPAN modules with built-in `Log4perl` support. Figure 4 shows the log data of a successful script run.

Open Heart Surgery

Line 68 authenticates the user against the Evernote web server. If the password and the consumer key are correct, the script is given unrestricted read and write access. Because you are handling sensitive data that you don't want to lose, some caution is advisable during programming. At the same time, you should ensure that the script only runs on a secure system behind a firewall to avoid exposing the credentials on more easily cracked systems, like Internet-facing web servers.

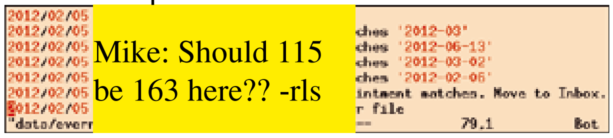
To find the entries from the 01-Tickler notebook, the script now needs the notebook's GUID. Line 105 iterates across all notebooks for this account and checks if the current notebook has this name. The same applies for 00-Inbox. The `evernote-tickler` script dumps the GUIDs for each into the `$tickler_guid` and `$inbox_guid` variables, respectively, and in the logfile if it finds them. If not, lines 126 and 131 terminate the program with an error: It wouldn't make sense to process an account lacking those folders.

The Evernote API doesn't offer a directory function to search notes in a specific notebook; instead, it insists that you use a `findNotes()` method that searches all notebooks for notes. That said, a filter of the `EDAMNoteStore::NoteFilter` type, armed with the `notebookGuid` parameter, will restrict the search to one notebook with the specified GUID.

The second parameter for `findNotes()` specifies an offset, with which you can set up paging for notes you find. In the example, the script obtains the complete list of events and uses a third parameter to restrict the list to 50 entries, which should be fine, even for longer Tickler lists. Because Evernote restricts the maximum number of notebooks returned per call to 50, you will need an offset and multiple calls if you want more.

Line 151 uses the CPAN `DateTime` module to compute tomorrow's date by adding a single day to today's date (`today()`). The `ymd()` method converts the resulting `DateTime` object into a string of the format YYYY-MM-DD. The regular expression in line 115 cuts the date out of the note's subject line (`title()`) and deposits it in the `$date_in_title` variable.

The `if` condition in line 169 checks if the title date fully or partially matches tomorrow's date. Specifying a month (YYYY-MM) or a specific date (YYYY-



```
2012/02/05                         ches '2012-03"
2012/02/05                         ches '2012-06-13'
2012/02/05                         ches '2012-03-02'
2012/02/05                         ches '2012-02-06'
2012/02/05                         intment matches. Move to Inbox.
2012/02/05                         r file
"data/everr                        --          79,1          Bot
```

Figure 4: The cron job has found a Tickler entry for the next day and pushes it into the user's inbox.

LISTING 1: evernote-tickler

```
#!/usr/local/bin/perl -w
#############################
# evernote-tickler
# Mike Schilli, 2012
# (m@perlmeister.com)
#############################
use strict;

BEGIN {
 use FindBin qw($Bin);
 chdir $Bin;
}

use local::lib;
use Thrift;
use Thrift::HttpClient;
use Thrift::BinaryProtocol;

use lib 'gen-perl';
use EDAMUserStore::Constants;
use EDAMUserStore::UserStore;
use EDAMNoteStore::NoteStore;
use EDAMNoteStore::Types;
use EDAMErrors::Types;
use EDAMTypes::Types;
use DateTime;
use Log::Log4perl qw(:easy);

my ($home) = glob "~";

Log::Log4perl->easy_init(
 {
  level    => $DEBUG,
  category => "main",
  file => ">>$home/data/" .
    "evernote-tickler.log"
 }
);

my $username = "my-user";
my $password = "my-passwd";
my $consumer_key =
  "perlsnapshot";
my $consumer_secret =
  "my-consumer-secret";

my $evernote_host =
  "evernote.com";
my $user_store_uri =
  "https://$evernote_host"
  . "/edam/user";
my $note_store_uri_base =
  "https://$evernote_host"
  . "/edam/note/";

my $http_client =
  Thrift::HttpClient->new(
 $user_store_uri);
my $protocol =
  Thrift::BinaryProtocol
  ->new($http_client);

my $client =
  EDAMUserStore::UserStoreClient
  ->new($protocol);

my $result =
  $client->authenticate(
 $username,
 $password,
 $consumer_key,
 $consumer_secret
  );

my $user = $result->user();

my $note_store_uri =
    $note_store_uri_base
  . $user->shardId();

my $note_store_client =
  Thrift::HttpClient->new(
 $note_store_uri);

my $note_store_protocol =
  Thrift::BinaryProtocol
  ->new($note_store_client);

my $note_store =
  EDAMNoteStore::NoteStoreClient
  ->new(
 $note_store_protocol);

my $notebooks =
  $note_store->listNotebooks(
 $result
   ->authenticationToken(
   )
  );

my $tickler_guid;
my $inbox_guid;

for
  my $notebook (@$notebooks)
{
 if ($notebook->name() eq
  "01-Tickler")
 {
  $tickler_guid =
    $notebook->guid();
  DEBUG
    "Found Tickler notebook";
 }
 if ($notebook->name() eq
  "00-Inbox")
 {
  $inbox_guid =
    $notebook->guid();
  DEBUG
    "Found Inbox notebook";
 }
}

if (!defined $tickler_guid) {
 die
"No Tickler notebook found";
}

if (!defined $inbox_guid) {
 die
   "No Inbox notebook found";
}

my $filter =
  EDAMNoteStore::NoteFilter
  ->new();
$filter->notebookGuid(
 $tickler_guid);

my $note_list =
  $note_store->findNotes(
 $result
   ->authenticationToken(
   ),
 $filter, 0,
 50
  );

my $tomorrow =
  DateTime->today(
 time_zone => "local")
  ->add(days => 1);
my $tomorrow_date_match =
  $tomorrow->ymd();

for my $note (
 @{ $note_list->{notes} })
{
 my $title = $note->title();

 my ($date_in_title) =
   ($title =~ /^(\S+)/);

 DEBUG "Check if ",
 "$tomorrow_date_match ",
 "matches '$date_in_title'";

 if ($tomorrow_date_match =~
  /^$date_in_title/)
 {

  DEBUG "$title matches. ",
        "Move to Inbox.";
```

LISTING 1: evernote-tickler (continued)

```
175
176   my $worked =
177     $note_store->copyNote(
178    $result
179      ->authenticationToken(
180      ),
181    $note->guid(),
182    $inbox_guid
183     );
184
185   die "copy note failed ($!)"
186     if !defined $worked;
187
188   DEBUG "Deleting note in ",
189         "Tickler file";
190
191   $note_store->deleteNote(
192    $result
193      ->authenticationToken(
194      ),
195    $note->guid()
196   );
197  }
198 }
```

MM-DD) will thus both return matches. Evernote's web API does not provide a move command, which explains why line 177 copies the Tickler entry to the user's Inbox if the date matches. The copy that remains in the Tickler notebook is subsequently deleted by the `deleteNote()` method in line 191.

Reliable Cron

An entry in your crontab of the following format

```
00 16 * * * /path/evernote-tickler
```

will ensure that Tickler is launched every day at four o'clock in the afternoon and that you will find a collection of tasks for the next day in your Inbox. If this isn't a good time to resume any one of the projects received yet, you can correct the date and put the note back into the Tickler notebook.

For time-critical events such as meetings, users can schedule appointments in a calendar application. And, if users can handle the next step toward completing the task and the step takes less than two minutes, they will do so immediately, according to GTD.

Otherwise, the user will create a new note in a notebook that lists all of their hot new projects and their next steps. Then, they can select an item from this list for immediate processing according to mood, energy level, context, and other considerations. ■■■

INFO

[1] Listings for this article: *http://www.linux-magazine.com/Resources/Article-Code*

[2] Allen, David. *Getting Things Done: The Art of Stress-Free Productivity*. Penguin, 2002: *http://www.amazon.com/dp/0142000280*

[3] "Perl: Evernote" by Mike Schilli, *Linux Magazine*, February 2012, pg. 58

[4] Evernote: *http://www.evernote.com*

**1/2 Ad
with bleed**