comic-get - your source of internet comics

Tommi Saviranta

wnd@iki.fi

1. Introduction

comic-get is a script in spirit of all-famous apt-get, the ultimate packet manager for Debian. comic-get allows downloading comic strips from internet with minimal fuzz. While you can write your own rules to get comics, comic-get also supports "repositories" (or rule databases) just like apt-get. This means you can also use rule sets created by others!

2. Requirements

comic-get should be pretty much happy with anything that runs Perl and has IO::Socket module available.

3. Features

TODO

4. Installation

Just copy the files in your favourite place and make sure Perl can find the directory called `ComicGet`. The easiest way to make sure of this is to maintain the original directory structure and call `comic-get.pl` in directory which also has `ComicGet`.

5. Configuration

comic-get stores its configuration and rulefiles in `$HOME/.comic-get`.

Configuration file is called `config` and for time being it can contain exactly two keywords, **source** and **target**. All keywords and their values follow the format "key = value". Whitespaces are optional and ignored.

5.1. source

source defines URI of repository to use. At the moment this can only be HTTP URL. More than one **source** can be defined.

Example:

```
source = http://wnd.iki.fi/comic-get
source = http://comic-get.org/
```

Notice that `http://comic-get.org/` does *not* exist.

5.2. target

This is the place where comic strips are downloaded to. It should be an absolute path on filesystem.

Example:

```
target = /home/wnd/comics
```

5.3. repository-pool

Defines URI of *master repository pool* to be used with command **masterpool**. Master repository pool lists known (read: reported) comic-get repositories. In normal operation user should not need to set this.

6. Usage

6.1. Searching for comics

6.1.1. Searching

comic-get allows user to search comics by their name, description, and any other given information such as name of the author if this information is filled in. Syntax for searching is **comic-get search keyword**.

Example:

```
% comic-get search IT
user-friendly (en_GB) - User Friendly
%
```

6.1.2. Showing package details

Once you've found one or more hit with your search details with **comic-get search**, you may want to look into package details. This is done with mantra **comic-get show package**.

Example:

```
% comic-get show user-friendly
Package: user-friendly
Version: 2
Compatibility: 2
Section: office
Maintainer: Tommi Saviranta <wnd@iki.fi>
Language: en_GB
Author: JD Illiad Frazer
Source: http://www.userfriendly.org/
Description: User Friendly
 IT-professionals at work
%
```

6.2. Installing comics

To install a package, command **comic-get install package** can be issued. If the comic is available in more than one language, language can be specified by appending **=language** after the package name such as **dilbert=en_GB**. Languages are listed with **search** and **show**.

If package is available from multiple repositories or in multiple languages, but language was not specified, list of possible candidates is printed, and user is asked to choose one.

Example:

```
Example:
% comic-get install dilbert
% ./comic-get.pl install dilbert
More than one candidate for dilbert:
  0: en_GB using wnd.iki.fi/comic-get
  1: en_US using wnd.iki.fi/comic-get
Which one should I install? Press enter to cancel.
 : 0
Installing dilbert (en_GB)...
Get:1 http://www.unitedmedia.com/ Dilbert [32.4 KiB]
Get:1 http://www.unitedmedia.com/ Dilbert [15.3 KiB]
Get:1 http://www.unitedmedia.com/ Dilbert [15.3 KiB] dilbert.png
Fetched 49.1 KiB in 8 seconds (6.1 KiB/s, 3 files)
%
```

Sometimes packages write comics in files that have conflicting filenames. When this happens, comic-get prompts you to enter another filename or cancel installation.

Example:

```
% comic-get install dilbert=en_US
Installing dilbert (en_US)...
File dilbert.png already installed by dilbert (en_GB)
If you would like to install the package anyway enter new filename here.
If you want to cancel, just press enter.
 : foo.png
Get:1 http://members.comics.com/ Dilbert [20.0 KiB]
Get:1 http://members.comics.com/ Dilbert [15.3 KiB]
Get:1 http://members.comics.com/ Dilbert [15.3 KiB] foo.png
Fetched 36.6 KiB in 5 seconds (7.3 KiB/s, 3 files)
%
```

6.3. Uninstalling comics

If you no longer want **upgrade** or **refresh** some of the comics, you can uninstall package with **comic-get remove package**. Like with **install**, language can be specified by using **=lang** after the package name. Also, if more than one package matched given package name, a list will be presented.

Example:

```
% comic-get remove dilbert
More than one package matching dilbert installed:
  0: en_GB through wnd.iki.fi/comic-get
  1: en_US through wnd.iki.fi/comic-get
Which one should I remove? Empty line cancels.
 : 1
dilbert (en_US) removed.
%
```

6.4. Refreshing comics

Installing a package will only download ruleset to download comics and fetch the latest comic strip. In order to keep your comics fresh and download the latest, **comic-get upgrade** can be used. This command will try to download the latest comics of installed packages.

Example:

```
% comic-get upgrade
Get:1 http://www.hs.fi/ Viivi ja Vagner [56.9 KiB]
Get:1 http://www.hs.fi/ Viivi ja Vagner [49.6 KiB]
Get:1 http://www.hs.fi/ Viivi ja Vagner [49.6 KiB] viivi_ja_wagner.png
Get:2 http://story.news.yahoo.com/ Garfield [25.6 KiB]
Get:2 http://story.news.yahoo.com/ Garfield [26.6 KiB]
Get:2 http://story.news.yahoo.com/ Garfield [26.6 KiB] garfield.png
Get:3 http://www.unitedmedia.com/ Dilbert [32.4 KiB]
Get:3 http://www.unitedmedia.com/ Dilbert [15.3 KiB]
Get:3 http://www.unitedmedia.com/ Dilbert [15.3 KiB] dilbert.png
Fetched 209.0 KiB in 8 seconds (26.1 KiB/s, 8 files)
%
```

6.5. Retrieving repository list

comic-get can retrieve a list of known repositories from official comic-get website by issuing **comic-get masterpool**. These servers can (probably) be used as repositories.

7. Setting up your own repository

Basically all you need for your very own comic-get repository is a place to put your files online. Technically comic-get repository consists of four files plus rules.

7.1. Basic files

7.1.1. `identifier`

`identifier` contains globally unique identifier for your repository. Identifier is used to distinguish your repository from other repositories possibly in use. If you are using registered domain name to host the files, it is probably a good idea to use repository URI as identifier. Author prefers to omit protocol descriptor but this is up to you.

7.1.2. `Packages`

`Packages` contains the actual package information. This information is usually extracted from rulefiles, but can also be manually entered. It is recommended to use update.sh to create this file.

7.1.3. `md5sums`

`md5sums` contains md5sums of `Packages` and `rules.md5sums`. When comic-get updates the repository information, in most cases only this file is retrieved. If md5sums at remote host do not match local files, appropriate files are updated. This file is usually created by update.sh.

7.1.4. `rules.md5sums`

`rules.md5sums` is like `md5sums`. It contains md5sums of rulefiles. When rulefile md5sums do not match local copies, rulefiles are updated. This file is also created by update.sh.

7.2. Rulefiles

Rulefile consists of two parts: first there are the actual rules to fetch the target, and then there are metadata for the package such as language.

The basic format of rules goes as follows:

```
title Mutts
```

```
uri http://seattlepi.nwsource.com/fun/mutts.asp
fetch
image gif /content/Mutts\?date=\d+
fetch gif
save mutts.png convert:gif
```

This particular ruleset defined a URI to http://seattlepi.nwsource.com/fun/mutts.asp (**uri**) and fetched the content (**fetch**). **image** is then used to find a image with filename matching regular expression **/content/Mutts\?date=\d+**. This image is tagged with **gif**. Then the first image with tag **gif** is fetched with **fetch**. Finally a file named `mutts.png` is written using fetched data, and target with tag **gif** is converted using convert.

7.2.1. Rulefile commands

7.2.1.1. ***title***

title defined a title for a set of rules. Currently this information is only used in progress indicator.

7.2.1.2. ***uri***

Push new URI to *URI list*.

7.2.1.3. ***fetch***

Fetch target. If no extra parameters are given, the first URI in the URI list is used. If one or more parameters are given, the first URI with given tag(s) is fetched. Parameters (tags) are given in descending order of preference.

7.2.1.4. ***image***

Find image from fetched data (usually HTML). **image** can take one or more parameters. If **image** is given two parameters, first parameter is tag for the image and the second one is regular expressions. If only one parameter is given, tag is assumed empty. Tag is later used with **fetch** and **save**. If more than two parameters are given, the first parameter is used as tag (as explained before), the second one as regular expression, and the remaining parameters are used to limit matches to HTML-elements that include given string(s). For example,

```
image jpeg p\d+.png class="latest"
```

will match

```
<img src="p123.png" class="latest" />
```

but not

```
<img src="p122.png" />
```

7.2.1.5. ***tag***

Find a tag from fetched data (usually HTML). **tag** takes three or more parameters. The first parameter is resource tag (e.g. **jpeg**), the second is the tag we're looking for (e.g. **meta**) and the third actual search/match pattern. Additional parameters are free-formed match pattern to further filter matches.

```
tag jpeg meta http://[^\s]+\.jpg property="og:image"
```

7.2.1.6. ***link***

Find link from fetched data (usually HTML). **link** takes one or more parameters. First parameter is always regexp to find link while remaining parameters can be used limit matches to HTML-elements that include given string. For example how to use extra parameters to limit matches, see **image**.

7.2.1.7. ***save***

Save data. **save** takes one or more arguments. First argument is the filename to use when saving the data. Following arguments are optional, and currently only **convert** is recognised. By default if options need arguments, arguments are separated from option name with a colon, and with comma from each other.

convert can be used to use convert retrieved data. This is particulary useful if more than one image is defined with **image** and they are of different format. For example if the comic strip is distributed in png format, but occasionally in jpeg, that part of the ruleset could be as follows:

```
image png /stuff/\d+\.png
image jpeg /stuff/j\d+\.jpeg
fetch png jpeg
save foo.png convert:jpeg
```

This would search data for images and tag them with png and jpeg. Then comic-get would try to fetch png. If downloading png would fail, jpeg would be downloaded. Finally data would be written to `foo.png`, and if the data would be from image with tag jpeg, it would be converted into png before writing.

7.2.1.8. ***select***

Select one of matches given by **image** or such. **select** takes exactly one argument which indicates index number of selected URI. Index numbers *start from one (1), not zero (0)*, and negative index numbers are considered to refer to reverse list, i.e. -2 matches the second last match.

7.2.2. Package metadata

Package metadata is separated from rules with line nothing but **--info--** on it. Generally speaking metadata fields begin with field name, followed by colon and whitespace.

7.2.2.1. ***Package***

Package defined package name. This must match the base filename of the rulefile. Package name should not contain underscores, and it is recommended to use only lowercase characters (a-z), numers (0-9) and dash (-).

7.2.2.2. ***Version***

Defines version of the ruleset. Currently not used anywhere.

7.2.2.3. ***Compatibility***

Defines ruleset compatibility. Current version of comic-get can only parse level-2 files. Currently this field is not used, but expect this to change in future releases.

7.2.2.4. ***Maintainer***

Define maintainer for the package. Can contain anything, but "Forename Surname <email@host.net> is recommended. (TODO: which standard?) Not used anywhere.

7.2.2.5. ***Language***

Defines comic strip language. Must be xx_XX (TODO: which standard?) and must match language part of the rulefile.

7.2.2.6. ***Author***

Defines (original) author of the comic. Used mostly for searching packages.

7.2.2.7. ***Source***

Defines website from which the strip is fetched. Not used.

7.2.2.8. ***Description***

Field different from others. **Description** should be followed with short description of the package (usually just the name). No other fields can follow **Description**. Lines after **Description** are treated as long (multiline) description for the strip. Each line must begin with a single space, and empty lines must have " **.**" (space-dot). Long description should be used to give out extra information about the comic strip which may help user to decide whether or not to try it.

8. Improvements and errors

If you find this documentation insufficient, erroneous, or have improvement suggestions, please let me know.

Feedback, bugfixes, and such for the scripts are also warmyl welcomed.

9. Future plans

comic-get should have a master repository that would list the other repositories. This feature alone is trivial, but maintaining that list is non-trivial should comic-get become popular. (Hah!)

10. Contact

Tommi Saviranta

<wnd@iki.fi>

For up to date information see http://wnd.katei.fi/contact.html.