Wikiprint Book

Title: Development

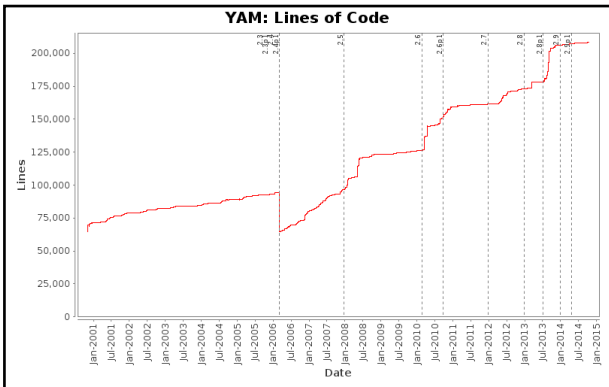
Subject: YAM - Yet Another Mailer - Development

Version: 17

Date: 25.10.2014 11:35:51

Table of Contents

Development	3
Source code access	3
Browsing our repository	3
Anonymous SVN access	3
Developer SVN access	3
SSL certificate warnings	4
Submit your change	4



Development

The following pages are meant to provide an insight view on the development of YAM. So if you are YAM developer, and interested developer considering joining our team or if you just want to compile YAM yourself. You may be able to find valueable information here to do so.

Source code access

As an open-source development effort, YAM is using a source code versioning system. We use the subversion (SVN) versioning tool and the SVN respository facilities provided by the trac engine this web site is running on. All latest, but also source code of older versions (back to version 2.2) are stored in this repository and are available to all interested developers.

Our SVN repository can be accessed either as an anonymous user, or as a registered YAM developer. You require a subversion (SVN) client, if you don't already have one. SVN is available for AmigaOS compatible systems from AmiNET (?http://aminet.net/package/dev/misc/subversion-1.1.4) and should also be available for most Unix operating systems, easily available on *BSD systems. In addition, there are also clients available for Win32 and MacOS.

The following instructions on how to access our repository should be the bare-bones needed to get you started with YAM and SVN; this is not a SVN tutorial. However, there exists a very good online tutorial at ?http://svnbook.red-bean.com/ and there are many HOWTOs available on the web as well.

Browsing our repository

Aside from the possibility to directly download/access the sources of YAM via a SVN client, you may browse the sources directly via the internal Browse Source facility provided by this website. By using the following link you may have a direct look at the current sources of YAM:

?http://trac.yam.ch/browser

By using this link you will find 4 top-level directories, whereas the 'trunk' directory corresponds to the very latest source code of YAM. This directory is also used to build the nightly builds you may already know of.

Anonymous SVN access

You can obtain the very latest source code of YAM (from 'trunk') with the following command sequence:

```
svn checkout https://svn.yam.ch/svn/yam/trunk yam
```

This will checkout the sources from our 'trunk' to a local directory 'yam' on your hard disk.

WARNING:
As usual, it is **not** guaranteed in any way, that the 'trunk' version will work for you properly. The 'trunk' of a SVN repository is always meant to carry highly experimental code which is not meant to be used on normal productive systems. It may break your current installation and may destroy data stored on your harddisks. On the other hand, downgrading may cause the same effect. And, maybe you will have to update your configuration files and/or images files aswell, which are probably not directly available through SVN. Please be always aware of that when you want to compile YAM yourself.

Developer SVN access

JavaScript required!

If you are an official registered developer of the YAM project, you should also have write permissions to the SVN repository. Thus, you should be able to commit changes to it. For this, the above mentioned procedure is more or less the same, as you are also going to checkout the latest 'trunk' with the very same command:

```
svn checkout https://svn.yam.ch/svn/yam/trunk yam
```

However, for writing changes back to the repository, you have to use the 'commit' command of SVN.

```
cd <your 'yam' directory>
svn commit
```

Similar to the above anonymous SVN description, the update of an already checked out version of the YAM sources can be update with:

```
cd <your 'yam' directory>
svn update
```

SSL certificate warnings

As we don't have an explicit SSL certificate for the yam.ch domain the SVN server is running the same SSL certificate like the supplier of the server we are running these webpages on. So in case you try to checkout the YAM sources from our SVN repository via https you will receive a certificate warning similar to:

```
Error validating server certificate for 'https://svn.yam.ch:443':
- The certificate is not issued by a trusted authority. Use the
  fingerprint to validate the certificate manually!
- The certificate hostname does not match.
Certificate information:
- Hostname: *.light-speed.de
- Valid: from Thu, 25 Jun 2009 06:43:57 GMT until Tue, 24 Jun 2014 06:43:57 GMT
- Issuer: MainServer (Karlsruhe), LightSpeed Communications GbR, Dresden, Sachsen, DE
- Fingerprint: a2:6a:6c:ce:a6:1c:0b:a8:4c:e6:e6:63:19:74:0b:54:39:fa:8b:99
(R)eject, accept (t)emporarily or accept (p)ermanently?
```

However, if the *Fingerprint* in your warning is exactly the same like the above one you are safe to permanently accept that warning as this is a valid certificate which still allows to encrypt the data connection but is simply issued for a different domain.

If your SVN version doesn't show such a certificate warning please make sure you have the latest version available of it. In case you have AmigaOS3 or AmigaOS4 you can try the latest version on AmiNET (?http://aminet.net/package/dev/misc/subversion-1.1.4).

Submit your change

There are several ways you can contribute to the YAM development. As the first steps you should:

1. Get the code. Install a nightly build and do an anonymous SVN checkout.

2. Find a bug to solve or a feature to implement. If you decide to implement a new feature, it might be worth discussing this in the developer ?mailing list mailing list first, to test water, get feedback from lead developers and see if anybody is already working on it, or if it fits at all into the general direction of the project.

3. Submit a patch. There are generally two ways to do a patch. You could make a copy of file you edit beforehand, and create a patch with:

```
diff -u file.c.orig file.c > file.c.patch
```

or do it with SVN:

```
svn diff -u file.c > file.c.patch
```

Submit your patch to the project's development mailing list. Please include your patch as a 'text/plain' attachment. Be prepared to be met with a critical analysis of your patch. In case your patch is refused, ask why and either correct your patch or constructively argue your point. In case you receive no feedback, try bumping the issue once after a day or three. If you still do not receive an answer, maybe your patch is not worthwhile. Keep it in case

someone raises the issue in the list sometime in the future, but otherwise sleep on it.

And some general points to remember:

- Respect the project coding standards, otherwise your patches will not be committed. You can find the coding standards in the 'trunk' as STYLEGUIDE.
- Work within the current framework, that is use the available classes and mechanisms.
- Monitor the development mailing lists, the ?discussion forum, and those of the apps you work on.

Lastly, remember that, although you work on the project on your own goodwill, this does not grant you any specific privileges. The lead developers make the final call. Sometimes, they may make decisions that you may not agree with. Obviously, you are entitled to voice your opinion and argue your point, but stay civil, do not drag it out and respect their decisions.

That is pretty much it. One last thing: do not do it for an ego boost, do it for the love of coding. The unfortunate truth is that contributing to an open source project will most likely never get you the kind of fame Linus Torvalds and friends enjoy, nonetheless, as with any Open Source project, your contribution will be greatly appreciated by the community, especially in the still shrinking Amiga community.