AMuseWiki

Alexander Krotov
Hacking on AmuseWiki

amusewiki.org

Hacking on AmuseWiki

Alexander Krotov

Contents

Cleaning the workdir

Even though you want to keep your modifications, you may want to remove modifications made automatically, such as creation of the SQLite database. Such automatically created files are already listed in `.gitignore`. You can remove them with

```
$ git clean -dffX
```

Don't confuse `-X` switch with `-x`! `-X` removes *only* the files from .gitignore, while `-x` also removes untracked files created by you. The safest way is to

```
git stash
```

or commit your modifications before cleanup.

Where to go from here

To learn more about Vagrant, start by reading the Getting Started guide.

AmuseWiki built with Catalyst framework and Bootstrap. Depending on the area you want to modify, you may want to learn more about these technologies.

Happy hacking.

This guide describes how to setup a development environment for AmuseWiki using Vagrant. Vagrant is a program that automates setting up virtual machines with development environments. Virtual machine is a good alternative to actually installing AmuseWiki and its dependencies on your computer.

Installing Vagrant

First, you need to install Vagrant and VirtualBox. Vagrant will manage virtual machine in VirtualBox.

On Debian-based operating systems you can install Vagrant and VirtualBox by running (as a superuser)

```
# apt-get install vagrant virtualbox
```

You may need to reboot the system before running VirtualBox for the first time.

For other operating systems refer to Vagrant documentation.

Obtaining the source code

AmuseWiki source code is managed with Git. On Debian-based systems you can install it with

```
# apt-get install git
```

To get the copy of source code, run (as normal user)

```
$ git clone https://github.com/melmothx/amusewiki
```

This will create `amusewiki` directory. It contains the `Vagrantfile`, which specifies how to setup a development environment for AmuseWiki. You don't need to read it, Vagrant does it for you.

To create and start the virtual machine, run

```
$ cd amusewiki
$ vagrant up
```

On the first run, Vagrant will download the operating system image, called box in Vagrant terms. This may take some time.

When Vagrant is done setting up your development environment, point your web browser to `http://localhost:8080`. You will be present with AmuseWiki web interface, which asks you for login and password.

Resetting AmuseWiki password

AmuseWiki development environment comes with a preconfigured website with one user, `amusewiki`. Its password is generated randomly on setup, so you need to reset it in order to login.

To reset AmuseWiki password, run

```
$ vagrant ssh
```

It will connect to the virtual machine and present you with a shell prompt. To reset

```
vagrant@debian-9:~$ cd /vagrant
vagrant@debian-9:/vagrant$ script/amusewiki-reset-password amusew
```

On the first line we change the working directory to `/vagrant`. `/vagrant` inside the virtual machine is the same directory as the directory with the source code on the host machine. It is automatically synchronized.

On the second line we reset the password for the `amusewiki` user. It will output something like

```
vagrant@debian-9:/vagrant$ script/amusewiki-reset-password amusew
Password for amusewiki is now 'correct horse battery staple'
```

Then, you can login as `amusewiki` with the password `correct horse battery staple` in your web browser.

Changing the source code

As the repository is mounted into `/vagrant` inside the virtual machine, any modifications to the code will immediately affect the running web application. Good place to start is to try modifying web page templates, which are stored in `amusewiki/root/src/`.

Running test suite

To run the test suite, use

```
vagrant@debian-9:/vagrant$ make test
```

If some tests are failing, they will be reported in the end:

```
...
Test Summary Report
-------------------
t/full-blog-mode.t              (Wstat: 256 Tests: 228 Failed: 1)
  Failed test:  166
  Non-zero exit status: 1
t/xapian.t                      (Wstat: 1024 Tests: 43 Failed: 4)
  Failed tests:  26-29
  Non-zero exit status: 4
...
```

To rerun a particular test suite, for example `t/full-blog-mode.t`, use

```
vagrant@debian-9:/vagrant$ perl -Mblib t/full-blog-mode.t
```

Stopping the development environment

When done, you can suspend the virtual machine by running

```
$ vagrant suspend
```

or shutdown it completely with

```
$ vagrant halt
```

To resume the development environment, simply run

```
$ vagrant up
```

again.

If you want to reinstall your development environment from scratch, you can delete the virtual machine by running

```
$ vagrant destroy
```

All your code modifications will be left intact, as they are stored in the shared folder. You can then commit them and submit a pull request.