Сборка unstable-версии Wine для stable-ветки Debian
TrueЪ Debian-way

Shroom [OmskLUG]*

10 января 2013 г.

Аннотация

В статье описан способ получения набора пакетов последней версии Wine для stable-ветки Debian GNU/Linux. В качестве инструмента используется утилита dpkg-buildpackage. Также кратко описан вариант использования dh_make.

1 Где найти?

Первая мысль, которая приходит в ответ на вопрос, где взять свежий Wine, — посмотреть собственно на сайте WineHQ в разделе Wine downloads (`http://www.winehq.org/download/`). Однако, как можно обнаружить при детальном исследовании ссылок на этой странице, никаких бинарников там нет. На самом же деле всё, что нам нужно, хранится на сайте стороннего разработчика, который и публикует сборки для Debian sid/unstable, а также выкладывает исходники с патчами. То есть, если у вас стоит sid (или подключены репозитории от unstable и experimental для разрешения зависимостей), вы можете взять пакеты прямо отсюда: `http://dev.carbon-project.org/debian/wine-unstable/`. Если же вы предпочитаете стабильность, но при этом хотите использовать новейшие разработки создателей Wine, придётся качать исходники. Ссылки на них размещены на этой же странице в разделе *Source package*. Фактически для сборки понадобятся два архива: **wine-unstable_1.5.5.orig.tar.bz2** (это собственно дерево исходников) и **wine-unstable_1.5.5-0.1.debian.tar.bz2** (это параметры для системы сборки пакетов). А файл **wine-unstable_1.5.5-0.1.dsc** — это подписанное маинтейнером описание архивов с исходниками.

2 С чего начать?

1. Распаковать архив **wine-unstable_1.5.5.orig.tar.bz2**. Получится каталог `wine-1.5.5`.

2. Распаковать архив **wine-unstable_1.5.5-0.1.debian.tar.bz2 внутрь** каталога `wine-1.5.5`. Таким образом внутри `wine-1.5.5` должен появиться каталог с именем `debian`.

3. Проверить, есть ли в системе пакет **dpkg-dev** (**dpkg-buildpackage** находится именно в нём): выполнить `dpkg -l dpkg-dev | grep ii` — если в результате вернётся строка с кратким описанием пакета, значит он уже есть, если ничего не будет, значит его нет. Если такой пакет не установлен, это недоразумение нужно исправить, выполнив `apt-get install dpkg-dev`.

4. Перейти в `wine-1.5.5` и запустить `dpkg-buildpackage -b -us -uc` (значения ключей будут описаны в следующем разделе). Если после всевозможных проверок сразу же началась компиляция, значит по какой-то волшебной причине у вас уже были поставлены все пакеты, от которых зависит сборка Wine. Если же процесс ещё и завершился корректно с кучей пакетов на выходе, вас можно только поздравить. Однако, с большой степенью вероятности **dpkg-buildpackage** завершится через пару секунд, вывалив примерно такое сообщение:
   ...

   ```
   dpkg-checkbuilddeps: Неудовлетворённые сборочные зависимости:
   debhelper flex bison libx11-dev libxext-dev libxi-dev
   libxrandr-dev libxrender-dev libxt-dev libxkbfile-dev
   libxxf86dga-dev libxxf86vm-dev libxinerama-dev libxcomposite-dev
   libgl1-mesa-dev | libgl-dev libglu1-mesa-dev | libglu-dev
   freeglut3-dev | libglut-dev libxmu-dev libxcursor-dev
   libncurses5-dev libcups2-dev libjpeg-dev libpng-dev
   libfreetype6-dev libfontconfig1-dev libopenal-dev libasound2-dev
   oss4-dev libsane-dev libusb-dev libgsm1-dev libmpg123-dev
   libcapi20-dev libdbus-1-dev | dbus-1-dev libgphoto2-2-dev
   liblcms1-dev libldap2-dev libssl-dev libgnutls-dev libxml2-dev
   libxslt1-dev unixodbc-dev prelink dctrl-tools | grep-dctrl lzma
   sharutils libgstreamer-plugins-base0.10-dev gettext valgrind
   dpkg-buildpackage: предупреждение: Неудовлетворительные
   зависимости/конфликты при сборке, останов.
   ```

   ...

5. Поставить все пакеты из списка неудовлетворённых зависимостей, полученного на предыдущем шаге. **Обратите внимание** на пары имён пакетов, разделённых символом «|» («конвейер»). Это *альтернативы*, то есть одно имя соответствует реальному пакету, который найдётся в репозитории, а второе имя — это или виртуальный пакет, или устаревшее название. И ставить нужно **реальный** пакет. Так что, прежде чем копировать кучей сразу все названия в командную строку после `apt-get install`, выясните, какие из названий, разделённых «|», реально присутствуют в репозитории (например, через `apt-cache search <regexp>` или `dpkg -l <regexp>`).

3 Как собрать?

Совершенно тривиально:
`dpkg-buildpackage -b -us -uc`

После тучи ошибок первого запуска **dpkg-buildpackage** наверняка предложил во второй раз запустить себя с ключом **-d**. Этот ключ означает «не определять зависимости». Если вы уверены, что ничего не упустили, можно, конечно, добавить и его... Но ведь заранее никогда не знаешь... Поэтому перезапускаем как есть, с теми же тремя ключами:

-b Ключ, предписывающий собирать только бинарные пакеты, без упаковки исходников. **Без** этого ключа **dpkg-buildpackage** попытается создать *src-deb* и будет искать архив с исходниками (то есть **wine-unstable_1.5.5.orig.tar.bz2**) рядом с каталогом сборки (то есть с `wine-1.5.5`). Если всё будет на месте, дополнительно получим ещё один пакет, который, впрочем, никому особо не нужен. И чтобы не множить сущности, ставим этот ключ.

-us Не пытаться подписать *.dsc*-файл.

-uc Не пытаться подписать файл *.changes*.

С момента запуска процесса сборки до получения набора пакетов может пройти (в зависимости от скорости работы компьютера) от 20 минут до полутора часов. В это время можно ~~курить бамбук~~ пить чай или кофе, поскольку возможности каким-либо образом повлиять на процесс до его окончания уже не представится.

Если всё прошло удачно, то в результате вы получите примерно такой список пакетов:

- libwine-alsa-unstable_1.5.5-0.1_i386.deb
- libwine-bin-unstable_1.5.5-0.1_i386.deb

- libwine-capi-unstable_1.5.5-0.1_i386.deb
- libwine-cms-unstable_1.5.5-0.1_i386.deb
- libwine-dbg-unstable_1.5.5-0.1_i386.deb
- libwine-dev-unstable_1.5.5-0.1_i386.deb
- libwine-gl-unstable_1.5.5-0.1_i386.deb
- libwine-gphoto2-unstable_1.5.5-0.1_i386.deb
- libwine-ldap-unstable_1.5.5-0.1_i386.deb
- libwine-openal-unstable_1.5.5-0.1_i386.deb
- libwine-oss-unstable_1.5.5-0.1_i386.deb
- libwine-print-unstable_1.5.5-0.1_i386.deb
- libwine-sane-unstable_1.5.5-0.1_i386.deb
- libwine-unstable_1.5.5-0.1_i386.deb
- wine-bin-unstable_1.5.5-0.1_i386.deb
- wine-unstable_1.5.5-0.1_i386.deb

Сразу замечу, что некоторые из пакетов можно не ставить:

libwine-cms-unstable_1.5.5-0.1_i386.deb если вам не нужна система управления цветом (Color Management System) под вайном (то есть, если вы не собираетесь калибровать ваши монитор, принтер и сканер из-под Wine);

libwine-dbg-unstable_1.5.5-0.1_i386.deb поскольку это отладочные символы для бинарников;

libwine-dev-unstable_1.5.5-0.1_i386.deb потому что этот пакет нужен для разработки под Wine;

libwine-gphoto2-unstable_1.5.5-0.1_i386.deb если вы не собираетесь смотреть фотки со своего цифрового фотоаппарата из виндовых приложений, запущенных под Wine'ом;

libwine-ldap-unstable_1.5.5-0.1_i386.deb если вы не собираетесь авторизоваться через Wine в домене;

libwine-oss-unstable_1.5.5-0.1_i386.deb поскольку технология OSS в Линуксе считается атавизмом (в отличие, например, от BSD-систем);

libwine-sane-unstable_1.5.5-0.1_i386.deb если вы не собираетесь давать виндовым приложениям доступ к вашему сканеру.

Если же вожделенных пакетов вы так и не обнаружили... Что ж, переходите к чтению следующего раздела.

4 Что делать, если что-то пошло не так?

Во-первых, ещё раз проверить зависимости. Если какой-то пакет не ставится — скорее всего, вы пытаетесь поставить альтернативный вариант, который стоит рядом с «|». В этом случае нужно ставить тот пакет, который записан с другой стороны от значка конвейера.

Во-вторых, проверить, не поломалась ли структура «дебианизации» дерева исходников — содержимое каталога `debian`. Если есть сомнения, его нужно прибить и распаковать заново из скачанного архива. В данном случае из-за отсутствия некоторых специфических настроек в `debian/rules` после сборки могут запуститься тесты, которые на подавляющем большинстве систем будут давать сбои (как заявляют сами разработчики Wine'а в одном из списков рассылки). А потеряться эти настройки могут, если по ошибке каталог `debian` будет перезаписан или удалён и создан заново по шаблону с помощью команды **dh_make**.

В-третьих, прочитать man по **dpkg-buildpackage** и понять, какие параметры реально нужны в данной ситуации. Например, если его запускать без ключа **-b**, дополнительно будет собран *src*-пакет. Вернее, он будет собран, если рядом с каталогом `wine-1.5.5` будет обнаружен архив **wine-unstable_1.5.5.orig.tar.bz2**. В противном случае вам предъявят следующее:

...

```
dpkg-source: ошибка: невозможно собрать с форматом
исходника «3.0 (quilt)»: файл orig.tar не найден
dpkg-buildpackage: ошибка:
dpkg-source -b wine-1.5.5 возвратил код ошибки 255
```

...

И если после этого захотеть исправить ситуацию с помощью **dh_make**, как об этом пишут на некоторых форумах, мы получим сброшенные настройки правил сборки пакетов и вернёмся к пункту «во-вторых».

И последнее. Все операции, кроме установки пакетов, осуществляются **с правами обычного пользователя**. Если в процессе сборки от вас потребовали права администратора, скорее всего, у вас не установлен пакет **fakeroot** (или подобный).

Примечание: если вы чувствуете растерянность от обилия в тексте непонятных команд и утомлены описаниями вещей, которые кажутся вам бессмысленными, настоятельно рекомендую сделать паузу в чтении, отдохнуть и **прочитать**

документацию ко всем упомянутым командам в формате *man* или *info*.

5 Что же такое dh_make?

Как сказано в мануале, это утилита для подготовки оригинальных сырцов к сборке из них дебиановского пакета. Вопрос «для чего это нужно?» мы сейчас рассматривать не будем, поскольку он является, скорее, философским и вполне может стать поводом для новой войны. В данном случае нас больше интересует, *как* это сделать.

Итак, если у вас есть просто голые исходники чего-либо (слитые, например, из какого-то репозитория на github или SourceForge), и вам позарез нужно сделать из них *.deb*-пакет, вам нужно будет совершить несколько нехитрых действий.

Самое первое — конечно же, распаковать архив с исходниками. Название директории после распаковки может быть, вообще говоря, любым, но для использования с **dh_make** нужно привести его к виду `<названиепакета>-<версия>`, причём `<названиепакета>` обязано содержать только маленькие буквы латинского алфавита, возможно, с цифрами и знаками «-» (обычный минус).

Следующим шагом будет собственно зайти в переименованную директорию и запустить там `dh_make`. И тут, как обычно, начинаются разные «но» и «если». Если не указан ключ `--native`, **dh_make** будет искать рядом с каталогом исходников архив вида `<названиепакета>-<версия>.orig.tar.gz` (впрочем, он может быть не только `.gz`, но также и `.bz2`, и `.lzma`). Если там такого файла нет, но указан ключ `-f <имяфайла>`, то **dh_make** скопирует `<имяфайла>` и использует его в качестве оригинального архива. Если же ключ `-f` не указан, но есть ключ `--createorig`, то **dh_make** создаст архив из текущего дерева исходников. Архив, кстати, нужен будет другим утилитам для генерации различий между оригинальными файлами и обновлёнными версиями (соответственно, для подготовки патчсета) и для построения дебиановской структуры *src*-пакета.

После того, как разобрались с архивом, нужно будет ответить на дополнительный вопрос: что мы хотим получить в результате — единственный пакет (*single binary*), архитектурно-независимые файлы (*arch-independent*), несколько пакетов (*multiple binary*), библиотеку (*library*), модуль ядра (*kernel module*) или же патч для ядра (*kernel patch*). От сделанного выбора будут зависеть параметры, которые **dh_make** запишет в структуру внутри каталога `debian`.

В сущности, это всё. После того, как **dh_make** отработал (будем надеяться, без ошибок), можно переходить к процедуре сборки пакета с помощью **dpkg-buildpackage**.

На самом же деле, если вы не собираетесь становиться маинтэйнером, скорее всего, эта утилита вам никогда не понадобится.

6 Благодарности

Автор выражает признательность члену OmskLUG с ником Lumpy за осуществление на практике и тестирование всего процесса, описанного в этом руководстве.

Спасибо за внимание!