

Хитрости для PKGBUILD и не только.

Рассуждения с примерами.

Допустим у нас есть некий исходный код, который надо превратить одновременно в несколько разных пакетов одним PKGBUILD-ом.

Сначала задаём один единственный `pkgbase`. Затем массивом `pkgname` все необходимые наименования пакетов, включая одно идентичное название `pkgbase`.

Теперь для каждого заданного пакета создаём свою `package_pkgname()` функцию. Раньше у нас всегда была одна единственная `package()`, а теперь их несколько - согласно массиву `pkgname`.

Например: `pkgbase=nvidia-340xx-lts`

`pkgname=(nvidia-340xx-lts nvidia-340xx-lts-dkms)`

`package_nvidia-340xx-lts()` и `package_nvidia-340xx-lts-dkms()`.

В каждой такой функции можно переопределить некоторые массивы - например `depends`, `pkgdesc` и другие. Но это не самое страшное.

Самое страшное - это то, что для каждого пакета надо устанавливать свою версию. Поможет вам в этом функция `pkgver()`, но только в случае если наши исходные файлы с `git` или `svn`. Если файлы прикреплены локально, то функция не поможет. Более подробно о данном методе смотрите на ArchWiki.

Большинство всех блогов на вопрос создания pkgbuild из deb или rpm - пишут про функцию tar -xzf или tar -xjf в функции package().

Однако как мы все знаем, каждый раз менять ключ в команде не очень то и удобно. Лучше всегда пользуйтесь для любых архивов другой командой, т.к. она одна единственная с одинаковыми ключами для чего угодно: bsdtar -xf "архив-для распаковки.расширение" -C "Путь извлечения".

Например так: bsdtar -xf "html-icons.tar.gz" -C "\$pkgdir/usr/share/icons/"

Однако стоит напомнить и ещё одну переменную noextract в PKGBUILD-е. В предыдущем примере в source подключен архив с иконками. Чтобы не делать лишнюю работу по копированию и распаковке, лучше задать переменную noextract=('html-icons.tar.gz') в виде массива файлов, которые не надо извлекать, т.к. далее это будет произведено вручную меньшими затратами.

bsdtar вполне можно использовать и в Makefile-ах, что прилично увеличит их производительность.

Существует и ещё одна команда. Вот на ней остановимся более подробно: install -Dm644 "/Источник/исходный-файл" "/Назначение/конечный-файл".

Команду можно использовать как в PKGBUILD-ах, так и в Makefile-ах.

Она копирует исходный файл или папку с предоставленными правами доступа в указанный каталог с указанными именем.

Например: install -Dm644 "\${_cplugin_pkgname}.desktop" "\$pkgdir/usr/share/applications/\${_cplugin_pkgname}.desktop"

Ключ -D - копировать из источника в назначение. -m - режим, как chmod, а не rwxr-xr-x. -g - группа. -o - владелец. -t - каталог.

Надеюсь все знают о преобразованиях rwx-rwx-rwx из двоичного в десятичный прав доступа? Указывать необходимо именно в цифрами.

Другой пример копирования целой папки: `install -Dt "${pkgdir}${_extradir}" -m644 "${srcdir}/${_pkg}/kernel"/{nvidia,uvm/nvidia-uvm}.ko`

Многие из вас, как и я раньше - создавали ярлыки desktop entries вручную. Но иногда это гораздо удобнее и быстрее осуществлять утилитой gendesk.

Сразу скажу о ключе -n, т.к. вам врядли захочется видеть в качестве иконки голубого пони... Неважно. Название иконки вставьте уже сами в другом ключе.

выглядеть это будет примерно так: `gendesk -n --pkgname="Название-desktop-файла" --pkgdesc="Описание, никуда не крепится" --name="Строка Name" --comment="Комментарий" --icon="Название-иконки-без-расширения" --terminal=false --categories="Здесь категории через точку с запятой; в конце не ставится" --custom="URL=http://www.example.com/" --startupnotify=false -f`

Поле custom позволяет вставить нужную строку вручную.

А далее, если вам необходимо можно лишние строки просто вырезать с помощью sed-a.

Например так:

```
sed -i '/Type=/s/Application/Link/' ${_cplugin_pkgname}.desktop
```

```
sed -i '/Exec=/d' ${_cplugin_pkgname}.desktop
```

```
sed -i '/Version=/d' ${_cplugin_pkgname}.desktop
```

Название иконки лучше вставлять без расширения. А преобразовать картинку в кучу иконок с разными размерами можно простым Makefile.

Например так:

```
ICONDIR=./
```

```
ICON_NAME=icon-name
```

```
ICONFL=${(ICONDIR)}/${(ICON_NAME)}.png
```

```
sizes:=16 24 32 64 96 128
```

```
icon_sizes:=$(foreach sz,$(sizes),$(sz)x$(sz))
```

```
.PHONY: all icon
```

```
all: icon
```

```
icon:
```

```
for i in $(icon_sizes) ; do \
```

```
mkdir -p $(ICONDIR)/hicolor/$$i/apps/ ; \
```

```
convert $(ICON_NAME).png -resize $$i $(ICONDIR)/hicolor/$$i/apps/$(ICON_NAME).png ; \
```

```
done
```

В итоге у вас будет папка с иконками разных размеров: hicolor. Упакуйте её в архив и используйте как в примерах выше.

Запомните, что файлы иконок после установки должны располагаться по следующему пути:

`/usr/share/icons/hicolor/папки-с-размерами...`

Ну и напоследок самое интересное.

Если вы хотите что-то делать во время установки или удаления пакета, то обратите внимание на строку `install=script.install` в `PKGBUILD`-е.

просто создайте в той же папке скрипт с любым названием (например `script.install`) и подключите его только в одной строке `install=`.

В этом скрипте создайте нужную вам функцию. Никаких `#!/bin/bash` не должно быть - голый файл с набором функций.

```
pre_install(){ }, post_install(){ }, pre_upgrade(){ }, post_upgrade() {  
, pre_remove() { }, post_remove() { }.
```

Можете вставить только одну любую единственную функцию.

В качестве примера скрипт установки ключей:

```
post_upgrade() {  
  
if usr/bin/pacman-key -l >/dev/null 2>&1; then  
  
usr/bin/pacman-key --populate elseworld  
  
fi  
  
}
```

```
post_install() {  
  
if [ -x usr/bin/pacman-key ]; then  
  
post_upgrade  
  
fi  
  
}
```

Чтобы поменьше мусорить в системе - как только отладили ваш PKGBUILD - собирайте его с ключами -sCc - установка зависимостей и удаление srcdir и pkgdir после сборки пакета(ов). Ну и если вдруг что-то не заладилось с контрольными суммами - ставьте SKIP и собирайте со следующими ключами: --skipchecksums --skipgpcheck --skipinteg.