

Документация Calmira GNU/Linux

Автор: [Михаил Краснов](#)

В данном разделе приведена документация дистрибутива Calmira GNU/Linux (текущая версия создавалась для Calmira LX4 1.1 GNU/Linux). Отсюда вы узнаете о распаковке дистрибутива из squashfs снимка, копировании на предварительно заготовленный раздел, созданию LiveUSB образа и запаковку в образ виртуального жёсткого диска для Qemu/KVM (*qcow2*).

Дата написания: 27 июн. 2021

Дата принятия: 05 авг. 2021

Версия: 2021.1

Для версии Calmira: 2021.5 (LX4 1.1)

Если дате принятия больше года, значит, вероятно, есть более новая версия этого руководства. Посетите раздел документации, чтобы убедиться в этом.

Обратите внимание, что дистрибутив позиционируется для продвинутого пользователя, собственно, и это руководство тоже. Оно может быть непонятно начинающим пользователям, поэтому советуем чаще обращаться к поисковым системам [Google](<https://www.google.com>), [Yandex](<https://www.yandex.ru>) и другим для поиска дополнительной информации, которой нет в этом руководстве.

Содержание

1.0 [Принятые обозначения](#)

1.1 [О дистрибутиве](#)

1.2 [История развития Calmira GNU/Linux](#)

2.0 [Установка Calmira](#)

2.1 [Загрузка дистрибутива из репозитория GitHub](#)

2.2 [Распаковка дистрибутива, форматирование раздела и копирование на него системы](#)

2.3 [Установка системы на ПК](#)

2.4 [Установка системы в Qemu/KVM](#)

2.5 [Первичная настройка дистрибутива после копирования](#)

2.6 [Завершение установки. Выход из chroot и перезагрузка](#)

3.0 [Настройка дистрибутива](#)

3.1 [Настройка окружения](#)

3.2 [Настройка часов](#)

3.3 [Настройка клавиатуры и шрифта в TTY](#)

3.5 [Настройка inittab](#)

4.0 [Управление пакетами](#)

4.1 [Введение в порты](#)

4.2 [Введение в srkg](#)

4.3 [Создание порта](#)

4.4 [Создание бинарного пакета для srkg](#)

1.0. Принятые обозначения

В данном руководстве используется несколько типографских обозначений.

Используется для отображения кода или команды, которую нужно ввести в терминал/редактор

Используется для отображения возможного вывода программы в терминал/файл или отображения содержимого предполагаемого/существующего файла

Используется для всевозможных пояснений

Используется для выделения важной информации

Используется для выделения каких-либо терминов, команд или имён файлов (только в абзацах, списках и таблицах).

Такой вариант так же приемлем.

Для этого используются теги `<tt> ... </tt>` и `<pre> ... </pre>`



Смотрите также:

Используется для акцентирования внимания на каких-либо важных или дополнительных сведениях.

1.1. О Calmira GNU/Linux

Дистрибутив Calmira GNU/Linux задумывался как классический легковесный дистрибутив для гиков. Предоставляется минимальное окружение, предназначенное для дальнейшей настройки. Calmira создавалась и сейчас разрабатывается одним человеком. Больше в команду никто не требуется, однако от помощи никто не откажется. На данный момент в процессе сборки новый релиз - Calmira LX4 1.1. В нём будет очень много изменений по сравнению с предыдущими релизами.

1.1.1. Причины появления дистрибутива

Calmira создавалась с несколькими целями. Во-первых, предоставление пользователю легковесной системы, которая смогла бы работать на оборудовании начала двухтысячных. Большинство "легковесных" дистрибутивов начинало отжирать столько ресурсов ПК, сколько не снилось даже самому жирному дистрибутиву. Разработчику Calmira же это не нравилось.

Во-вторых, предоставление классического "олдскульного" окружения, которое нравилось в своё время матёрым линуксоидам. В некоторых случаях это даже лучше современного.

В-третьих, недовольство современными дистрибутивами. Одни не уважают частную жизнь и свободу пользователя, другие очень медленные и нестабильные, у третьих ужасные системные компоненты и т.д. Поэтому неплохо было бы сделать дистрибутив "для себя", "для души", и поделиться наработками с сообществом. Так же дистрибутив должен быть надёжным. Конечно, на данный момент, Calmira не такая надёжная, как планировалось, но все баги и недоработки исправляются, дистрибутив активно развивается.

В-четвёртых, в большинстве дистрибутивов Linux было установлено большое число абсолютно ненужного ПО, которое только занимало место, но не более. В новой системе Calmira этого же быть не должно. Она должна быть максимально лёгкой и без лишнего мусора, но, одновременно, и не в ущерб функциональности. Более того - в Calmira предоставляется только свободное/открытое ПО, что так же неплохо опытным пользователям.

Calmira была названа в честь одноимённой графической оболочки для Windows 3.11. Первоначально планировалось назвать дистрибутив "CalmiraOS", но через некоторое время было принято название "Calmira GNU/Linux".

Больше информации можете узнать в [истории](#) дистрибутива.

1.1.2. Позиционирование

Сейчас дистрибутив позиционируется для опытных пользователей или гиков, а так же владельцев слабого железа, так как Calmira очень слабо нагружает ПК. В дистрибутиве предоставлена большая свобода действий для пользователя. Большое число компонентов системы поддаётся настройке. Вы можете удалить один компонент системы, заменив его на другой.

1.1.3. План выхода релизов

Совсем недавно был сформирован план выхода релизов. Точных дат выхода нет, так как не всегда можно уложиться в поставленные сроки. Однако, тестовая версия дистрибутива выходит 5 числа определённого месяца, а стабильная - 15 числа (не обязательно того же месяца). Перед выходом тестовой версии на GitHub выкладывается временный инструментарий, с помощью которого собиралась система (если таковой инструментарий был

создан, чаще всего используются давно созданные). 5-го числа выкладывается тестовая версия системы, а 15 (не обязательно того же месяца) - окончательный релиз. Всё это можно найти в разделе [Releases](#).

Кандидаты в релизы обычно не выкладываются на GitHub, а сохраняются в архиве у автора. Тестирование релиз-кандидатов закрытое. О внесении правок и изменений не известно никому.

1.1.4. Обновления пакетов

На данный момент, обновления пакетов предоставляются только для текущего релиза. Наладить подобный механизм в предыдущих версиях затруднительно.

[Назад](#)

Принятые обозначения

[Домой](#)

[Далее](#)

История развития Calmira
GNU/Linux

1.2. История развития Calmira GNU/Linux

Calmira GNU/Linux - молодой дистрибутив, разрабатываемый в одиночку бывшим владельцем нескольких небольших сообществ. Этот дистрибутив не такой удобный и функциональный, однако он развивается, хоть и медленно, а так же активно используется некоторыми людьми в своих целях.

1.2.1. Цели появления

Если дистрибутив создавался "просто так", без целей, то он не будет нужен никому.

Calmira создавалась с несколькими целями. Во-первых, предоставление пользователю легковесной системы, которая смогла бы работать на оборудовании начала двухтысячных. Большинство "легковесных" дистрибутивов начинало отжирать столько ресурсов ПК, сколько не снилось даже самому жирному дистрибутиву. Разработчику Calmira же это не нравилось.

Во-вторых, предоставление классического "олдскульного" окружения, которое нравилось в своё время матёрым линуксоидам. В некоторых случаях это даже лучше современного.

В-третьих, недовольство современными дистрибутивами. Одни не уважают частную жизнь и свободу пользователя, другие очень медленные и нестабильные, у третьих ужасные системные компоненты и т.д. Поэтому неплохо было бы сделать дистрибутив "для себя", "для души", и поделиться наработками с сообществом. Так же дистрибутив должен быть надёжным. Конечно, на данный момент, Calmira не такая надёжная, как планировалось, но все баги и недоработки исправляются, дистрибутив активно развивается.

В-четвёртых, в большинстве дистрибутивов Linux было установлено большое число абсолютно ненужного ПО, которое только занимало место, но не более. В новой системе Calmira этого же быть не должно. Она должна быть максимально лёгкой и без лишнего мусора, но, одновременно, и не в ущерб функциональности.

Calmira была названа в честь одноимённой графической оболочки для Windows 3.11. Первоначально планировалось назвать дистрибутив "CalmiraOS", но через некоторое время было принято название "Calmira GNU/Linux".

1.2.2. Первые версии

2021.1 - первый блин комом

Версия 2021.1 вышла 23 апреля 2021 года. Собиралась тогда по LFS 9.0. На данный момент образ с системой является утерянным. Система выдачи версий дистрибутиву была простой. До точки - год выпуска дистрибутива, после (мажорная версия) - номер версии. В некоторых релизах добавляется ещё одна точка и число после неё (минорная версия). Просто небольшое обновление с исправлением багов (например, 2021.2.1).

Первая версия (2021.1) содержала в себе очень много багов и недоработок, начиная от нерабочего `gmp`, заканчивая отсутствия какого-либо инструмента для работы с пакетами (кроме систем сборки, конечно). Потом вскрылись и другие проблемы: полуробочий `wget` и невозможность сборки `Xorg`. Чуть позже был написан `cpkgi-tools` - то, из чего впоследствии получился автоматизированная утилита для работы с пакетами `cpkg`. А `cpkgi-tools` (Calmira Package Installer Tools) в то время мог только устанавливать ПО, об удалении и прочих

функциях не было и речи. Хотя были какие-то концепты для добавления жизненно важных функций, но они не были реализованы окончательно ввиду большой разрозненности - разные программы для установки, удаления и просмотра информации о пакете, а так же абсолютно разные общие функции для этого.

Самым главным достоинством `srkg` было то, что пакет он ставил в отдельную директорию (`/usr/user`). Но были и минусы - сильное раздувание `PATH`, поэтому чуть позже от этого отказались. Были очень жёсткие ограничения на имя пакета (так, например, в нём не должно было содержаться ничего, кроме имени пакета и расширения `tar.xz`, которое в будущем сократилось до `txz`), а так же кол-во файлов. Были серьёзные проблемы с добавлением пакета в базу данных. Однако `srkgi-tools` задал формат пакетов для будущего `srkg`. Этот формат, кстати, используется до сих пор, только немного дополнялся.

Что и говорить - сначала у Calmira не было и собственного логотипа! Он появился почти перед выходом версии 2021.2 и являлся перерисованным изображением одноимённой оболочки для Win3.11. Логотип неофициальный, и до сих пор не считается логотипом дистрибутива, хотя некоторые его компоненты (например, просмотр информации о версии дистрибутива) используют его, а на официальном сайте выложен скриншот дистрибутива с выводом в терминал ASCII-версии логотипа.

Как говорится, первый блин комом. Но Calmira 2021.1 дала очень большой опыт в создании дистрибутивов (до этого автор "забавлялся" пересборкой Debian, AntiX и Mint), а так же после выхода этой версии появилось очень много планов, некоторые из которых не реализованы и сегодня.

Как назло, первая версия была утеряна при создании `sqsh`-снимка для отправки дистрибутива знакомым для их оценки. Поэтому этот релиз не был выложен на GitHub. Возможно, в скором времени будет воссоздана эта версия.

2021.2 - формирование новой редакции с сервером Xorg и учёт некоторых старых ошибок

Версия 2021.2 выгодно отличается от предыдущей. Из неё был убран `srkgi-tools` и добавлен новый `srkg`, исправлен баг с пакетом `gmp`, из-за небольшого упущения в сборке которого невозможно было скомпилировать 90% программ на Calmira. В минимальную поставку было добавлено несколько новых пакетов. Так же начались работы над созданием репозитория с бинарными пакетами для `srkg`.

Автоматизированная утилита для работы с пакетами `srkg` тогда многого не умел. Работал он нестабильно, но был намного лучше предыдущих `srkgi-tools`. Добавили нормальную базу данных пакетов, установка и удаление пакета работали намного быстрее и надёжнее, было снято большое число ограничений. Добавили `pre-` и `postinstall` скрипты, а ближе к релизу Calmira 2021.3 `srkg` научится собирать ПО из исходного кода самостоятельно на основе предварительно приготовленных инструкций для сборки.

Да, пока ещё `srkg` - АУРП. Пакетным менеджером его в то время назвать было трудно - функционал был небольшим. Но скоро всё изменится в лучшую сторону. Релиз `srkg 1.0` `ra4` поменяет скромное "Автоматизированная утилита для работы с пакетами" на гордое "пакетный менеджер". `ra4` означало "pre-alpha 4". У `srkg` была немного другая система наименования версий, но очень похожая на то, что будет через релиз в будущем.

Однако баги так же были (а как же без них?). Например, сломанный `make-ca` и невозможность скачивания файлов с помощью `wget` без ключа `--no-check-certificates`, нерабочий `git` и полурбочий `uptime` (возможно, сломанными были и другие утилиты).

Было обновлено большое число пакетов и конфигов, а так же дистрибутив начал потреблять меньше оперативной памяти.

Помимо этого, была подготовлена отдельная редакция с сервером Xorg, а так же бинарный пакет с оконным менеджером Fluxbox (который нужно было установить вручную посредством нового `srkg`).

2021.3 - небольшое обновление

В этом релизе было подготовлено несколько мелких, но довольно важных обновлений - новая предварительная версия `srkg`, некоторые изменения в поставке ПО из минимальной сборки, отказ от редакции с `Xorg`, так как в планах было создание бинарных пакетов с этим сервером.

Начиная с этой версии начинает меняться позиционирование дистрибутива. Теперь это не просто легковесный дистрибутив, а минималистичная гибкая и быстрая система для решения узкого круга задач. Был создан "Манифест разработчиков Calmira" (он же "Несколько принципов построения ОС Calmira GNU/Linux"), в котором были установлены жёсткие ограничения на потребление дистрибутивом ресурсов ПК и установленный в нём софт.

В этом манифесте было описано несколько принципов: делать систему максимально компактной и минималистичной, но не в ущерб функциональности; ориентировать систему не для новичка, а на продвинутого пользователя; обеспечивать максимальную свободу пользователю; не использовать наработки Calmira во вред; делиться наработками с сообществом, если автор наработок считает это нужным.

В `srkg` добавили массу новых функций, таких как скачивание пакета с репозитория, обновление списка пакетов, а так же создание локального репозитория. `srkg` научился собирать ПО из исходного кода. В этот релиз дистрибутива вошёл `srkg 1.0ra4` с массой новых изменений. На то время это был самый крупный релиз за всю историю `srkg`.

Но увы - разработка этой версии была окончена во время тестирования. Несмотря на то, что это был относительно неплохой релиз, изменений в нём было довольно мало, а предыдущая версия 2021.2 всех устраивала. Поэтому все изменения было принято отложить на будущее.

1.2.3. Последние версии

LX4 1.0 - смена структуры директорий на упрощённую, повышение стабильности дистрибутива и куча других изменений

Самое большое изменение - это отказ от LFS в сторону молодого Linux4Yourself, сокращённо называемым LX4Y, LX4U либо же совсем коротко - LX4 (Linux4Yourself, кстати, появился через некоторое время после старта разработки Calmira 2021.1). Вместе с этим переходом изменилась и структура каталогов системы. В LX4 (и будущих LFS 11, кстати, тоже) отсутствуют `/bin`, `/sbin`, `/usr/sbin` и `/lib` - они заменены ссылками на `/usr/bin` и `/usr/lib` соотв. Это менее безопасно, а так же делает систему не такой гибкой (предыдущие релизы могли вообще без `/usr` загружаться и нормально работать). Это противоречит философии Calmira. В релизе LX4 1.1 это будет исправлено.

Вы могли заметить, что и система наименования версий дистрибутива поменялась. Вместо запутанной "годовой" (так называет разработчик Calmira предыдущую систему наименования версий) используется более простая. Мажорная версия (число до точки) значит крупное изменение, новый крупный релиз; минорная версия (число после точки) значит небольшие изменения, новый промежуточный релиз с мелкими обновлениями и изменениями. Приставка LX4 означает, что используется *Linux4Yourself*.

Из поставки дистрибутива был удалён пакетный менеджер `srkg`, ввиду его нестабильности. Он часто приводил систему в полурбочее состояние, а с упрощённой структурой директорий может убить систему полностью. В релиз LX4 1.0 планировалось ввести систему портов (например, как в FreeBSD), но работа над ней не была закончена, поэтому введение этой системы было отложено на релиз LX4 1.1.

Так же некоторое ПО было заменено на более совершенные форки или аналоги (например, `zlib` на `zlib-ng`).

Окончательно утвердился метод сжатия пакетов и образов системы. Теперь используется `xz` как основной ввиду своей эффективности сжатия, хотя

сжимает чуть медленнее всех остальных.

На этапе формирования новый цикл разработки дистрибутива. Но об этом позже.

Что потом?

Сейчас LX4 1.0 - последняя версия дистрибутива. В неё вошло много исправлений и улучшений. Система работает быстрее и надёжнее, а так же скоро появится много возможностей. Через пару месяцев ожидаем окончательную версию Calmira LX4 1.1 с огромным числом изменений. За ходом разработки вы можете посмотреть у нас на GitHub. А пока подумайте - что будет в новой версии дистрибутива?

[Назад](#)

О Calmira GNU/Linux

[Домой](#)

[Далее](#)

Установка дистрибутива

2.0. Установка дистрибутива

В данном разделе пойдёт речь об установке системы. Установка Calmira не похожа на установку других дистрибутивов. Вам нужно самому распаковать образ, скопировать данные и настроить конфиги, а некоторые написать самостоятельно (в т.ч. конфиг загрузчика GRUB).

Скоро будет готов установочный iso/img образ системы, и процесс установки будет более предсказуемым и понятным - сейчас же для установки Calmira нужен другой дистрибутив GNU/Linux, с помощью которого будет распакован снимок с системой, подготовлены необходимые разделы и скопированы на них данные из снимка. С помощью установочного iso/img не нужно будет загружаться в уже установленный GNU/Linux для подготовки ПК к установке Calmira.

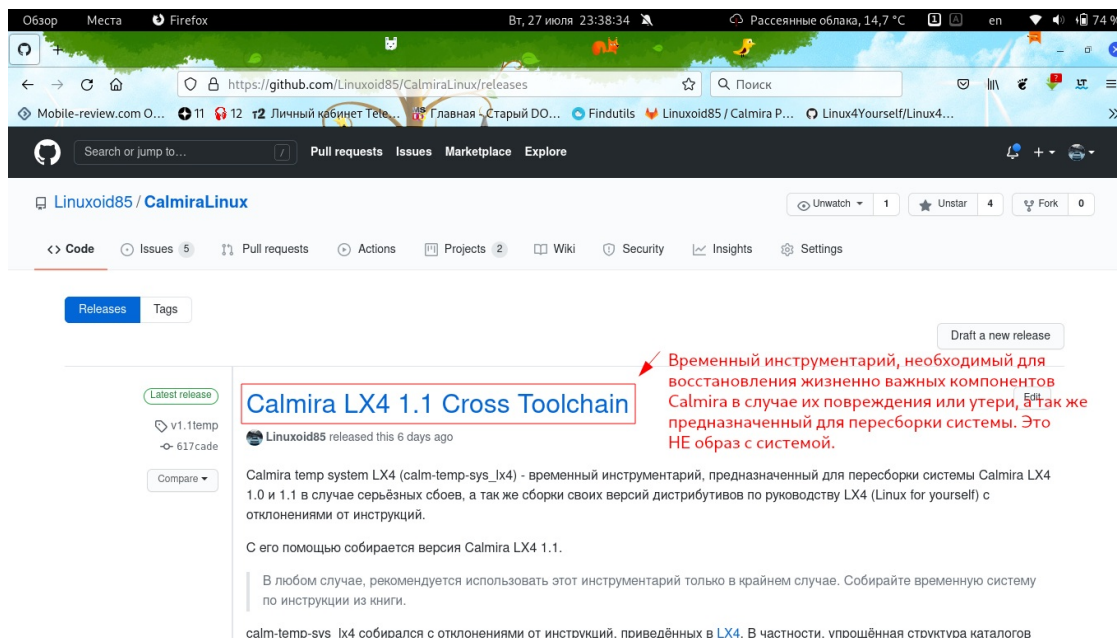
Содержание

- 2.1 [Загрузка дистрибутива из репозитория GitHub](#)
- 2.2 [Распаковка дистрибутива, форматирование раздела и копирование на него системы](#)
- 2.3 [Установка системы на ПК](#)
- 2.4 [Установка системы в Qemu/KVM](#)
- 2.5 [Первичная настройка дистрибутива после копирования](#)
- 2.6 [Завершение установки. Выход из chroot и перезагрузка](#)

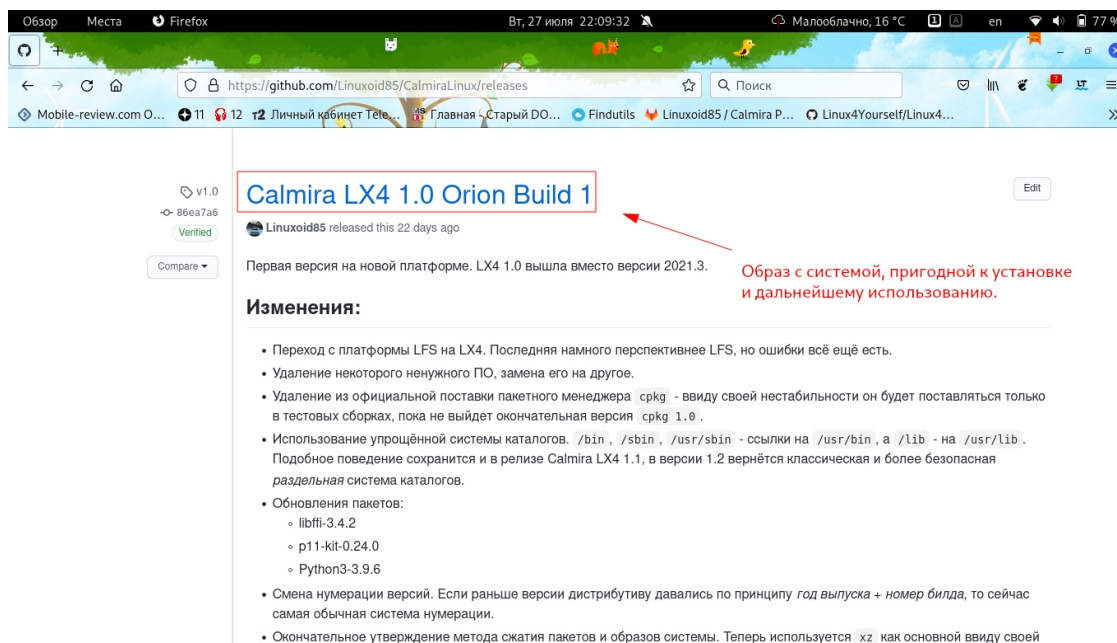
2.1. Скачивание дистрибутива из репозитория GitHub

На главной странице приведён раздел загрузок, но там приведены последние версии системы. А пользователь вправе выбирать нужную только ему.




Все релизы находятся в разделе [Releases](#). Здесь выложены как образы самой системы, так и временные инструменты для пересборки или восстановления Calmira GNU/Linux. Распознать одно от другого просто - у временного инструментария в заголовке написано "Calmira \$VERSION Cross Toolchain". Это архивы с временным инструментарием (кросс-компилятором и другими инструментами, необходимыми для восстановления или пересборки системы). ЭТО НЕ ЗАГРУЗОЧНЫЕ СИСТЕМЫ!



Всё остальное же - образы с системой. Вам следует скачать образ `sqsh`. Впоследствии, он будет распакован, а данные из него будут скопированы на определённый раздел жёсткого диска. Архивы `zip` и `tar`, подписанные как "Source code" являются копией данных из репозитория. Вам следует скачивать образ `sqsh`.



Нужные файлы, описанные в предыдущем параграфе, находятся в разделе "Assets":

Assets 3	
 calmira-1.0.sqsh	207 MB
 Source code (zip)	
 Source code (tar.gz)	

[Назад](#)
Установка дистрибутива

[Домой](#)

[Далее](#)
Распаковка дистрибутива

2.2. Распаковка дистрибутива

После скачивания дистрибутива нужно его распаковать. Это не *raw*-образ, который можно записать утилитой *dd* или любыми другими аналогами. Тут нужно сделать всё самостоятельно.

2.2.1. Распаковка

Так как это squashfs-образ, то распаковка производится утилитой *unsquashfs* из состава пакета *squashfs-tools*. Пример использования:

```
sudo unsquashfs calmira-1.1.sqsh
```

Пакет *squashfs-tools* может быть скомпилирован в Calmira GNU/Linux из системы портов. На других системах может быть установлен из бинарного пакета. Учитывайте то, что, например, на Fedora Linux (и других rpm-дистрибутивах) этот пакет НЕ собирается с поддержкой формата сжатия *xz*. Вам необходимо будет пересобрать его с поддержкой этого формата сжатия, так как образ с дистрибутивом Calmira сжимается с помощью *xz*.



ВНИМАНИЕ!

squashfs очень сильно сжимает данные. Сжатый образ весит 207 Мб, а данные, которые распакуются из него - 1 Гб. Учитывайте это.

Файлы распакуются в директорию *squashfs-root*. Введите:

```
cd squashfs-root
```

2.3. Установка системы на ПК

В данной инструкции описаны действия по установке системы на реальное железо.

2.3.1. Форматирование раздела, на который установится Calmira GNU/Linux

Предполагается, что Calmira будет установлена на один раздел, а не несколько. Хотя вы без проблем можете выделить столько разделов под установку, сколько вам надо.

Минимальный раздел корневого раздела должен равняться 1.5 Гб. Этого вполне хватит для установки дистрибутива и хранения некоторых пользовательских файлов (например, документов). Отформатируйте раздел в файловую систему *ext4*:

```
sudo mkfs.ext4 /dev/sdX
```

Замените *sdX* на метку раздела.

Теперь смонтируйте этот раздел в */mnt/calmira_system*:

```
sudo mkdir /mnt/calmira_system  
sudo mount -v /dev/sdX /mnt/calmira_system
```

Замените *sdX* на метку раздела.

2.3.2. Копирование системы

После того, как смонтировали раздел, скопируйте на него данные из распакованного образа дистрибутива. Во-первых убедитесь, что находитесь в директории *squashfs-root*:

```
pwd
```

Если всё верно, приступайте к копированию:

```
sudo cp -rvx * /mnt/calmira_system/
```

Во время копирования на экран будут выведены скопированные файлы. Внимательно прочитайте вывод: в нём не должно содержаться *error*, *fail* и пр. Если вы не хотите видеть подробный результат о каждом скопированном файле, то уберите ключ *-v* из команды. В таком случае на экран будут выведены сообщения об ошибках и предупреждения (если они есть).

2.4. Установка системы в Qemu/KVM

Вы можете захотеть установить Calmira не на реальное железо, а в виртуальную машину Qemu/KVM. Создайте виртуальную машину со следующими параметрами:

1. Тип ОС: Generic Linux/Generic Default;
2. Процессор: qemu64/kvm64/копирующий конфигурацию ЦП хоста (в случае если на хосте процессор x86_64);
3. Оперативная память: 64 Мб;
4. Жёсткий диск: qcow2 1.5 Гб;
5. Шина диска: SATA.

Всё это создавалось в virt-manager. Параметры рекомендуемые.

2.4.1. Монтирование

Теперь нужно скопировать распакованный [в этой](#) инструкции дистрибутив в только что созданный образ жёсткого диска (образ сгенерировался во время создания виртуальной машины). Нужно смонтировать этот образ. Для этого подключите модуль qemu-nbd и смонтируйте образ ЖД:

```
# Подключение модуля ядра:
sudo modprobe nbd max_part=8

# Подключение образа:
sudo qemu-nbd --connect=/dev/nbd0 /var/lib/libvirt/images/$NAME.qcow2

sudo fdisk /dev/nbd0 -l
mkdir /mnt/calmira_system
```

Замените *\$NAME* на имя виртуальной машины (и, по совместительству, имя диска).

2.4.2. Разметка разделов

Необходимо разметить новый "жёсткий диск". Для этого выполните:

```
sudo cfdisk /dev/nbd0
```

cfdisk запросит у вас, какую таблицу разделов выбрать. Выбирайте *mbr* (она же *dos*), либо новомодную *gpt*. В данном случае нет никакой разницы в выборе. Советуется выбрать *mbr* (*dos*).

Если вы всё-таки выбрали *gpt*, то создайте в начале диска раздел объёмом 1 Мб без файловой системы. Тип: *BIOS Boot*.

Потом (и для *gpt*, и для *mbr*) создайте корневой раздел. Он должен занимать всё оставшееся место. Тип: *Linux filesystem*. При желании можете создать ещё и раздел со *swap*. Его объём должен равняться половине от объёма оперативной памяти (к примеру, в виртуальной машине установлено значение ОЗУ, равное 64 Мб. Для подкачки тогда нужно выбрать объём 32 Мб). Тип раздела с подкачкой: *Linux swap*.

2.4.3. Форматирование разделов

После разметки отформатируйте нужные разделы. Для форматирования корневого раздела выполните:

```
sudo mkfs.ext4 /dev/nbd0pX
```

Замените *nbd0pX* на нужную метку раздела. Обычно это *nbd0p1*, если таблица разделов *mbr (dos)*, или *nbd0p2*, если таблица разделов - *gpt*.

Для форматирования подкачки (если вы её создали):

```
mkswap /dev/nbd0pX
```

Замените *nbd0pX* на нужную метку раздела. Обычно это *nbd0p2*, если таблица разделов *mbr (dos)*, или *nbd0p3*, если таблица разделов - *gpt*.

2.4.4. Копирование системы

Смонтируйте корневой раздел:

```
sudo mount /dev/nbd0pX /mnt/calmira_system
```

Замените *nbd0pX* на нужную метку раздела. Обычно это *nbd0p1*, если таблица разделов *mbr (dos)*, или *nbd0p2*, если таблица разделов - *gpt*.

После того, как смонтировали раздел, скопируйте на него данные из распакованного образа дистрибутива. Во-первых убедитесь, что находитесь в директории *squashfs-root*:

```
pwd
```

Если всё верно, приступайте к копированию:

```
sudo cp -rvx * /mnt/calmira_system/
```

Во время копирования на экран будут выведены скопированные файлы. Внимательно прочитайте вывод: в нём не должно содержаться *error*, *fail* и пр. Если вы не хотите видеть подробный результат о каждом скопированном файле, то уберите ключ *-v* из команды. В таком случае на экран будут выведены сообщения об ошибках и предупреждения (если они есть).

[Назад](#)

Установка дистрибутива
на ПК

[Домой](#)

[Далее](#)

Первичная настройка
после копирования

2.5. Первичная настройка дистрибутива после копирования

Вы скопировали содержимое снимка себе на компьютер. Сейчас же необходимо настроить дистрибутив для корректной работы на нём. Без выполнения этих действий Calmira GNU/Linux не будет работать.

2.5.1. Смена корня

Первым делом необходимо загрузиться в только что скопированную Calmira. Сейчас же нельзя это сделать, так как система не загрузится - нужно настроить fstab и загрузчик. Поэтому на хост-системе произведём операцию смены корня.

Первым делом вы должны смонтировать раздел жёсткого диска или образ виртуального диска qcow2 в вашу систему. Точка монтирования: `/mnt/calmira_system`. О монтировании сказано в инструкциях по копированию системы на раздел жёсткого диска ([здесь](#)), либо в виртуальную машину ([здесь](#)).

```
for DIR in "dev" "dev/pts" "proc" "sys"; do
sudo mount -v --bind /$DIR /mnt/calmira_system/$DIR
done

if [ -h /mnt/calmira_system/dev/shm ]; then
mkdir -pv /mnt/calmira_system/$(readlink /mnt/calmira_system/dev/shm)
fi

chroot "/mnt/calmira_system" /usr/bin/env -i \
HOME=/root TERM="$TERM" \
PS1=(chroot) \u:~$ \
PATH=/bin:/sbin:/usr/bin:/usr/sbin \
/bin/bash --login
```

Вы окажетесь в скопированной Calmira GNU/Linux. Вы изолированы от хост-системы, и можете настраивать дистрибутив так, как вам удобно.

2.5.2. Настройка file system table

fstab (file systems table) — один из конфигурационных файлов, который содержит информацию о различных файловых системах и устройствах хранения информации компьютера, описывает, как диск будет использоваться или как будет интегрирован в систему. Файл `/etc/fstab` делает возможным автоматическое монтирование определенных файловых систем, что особенно нужно при загрузке системы. Он содержит ряд строк, описывающих файловые системы, их точки монтирования и другие параметры.

Каждая строка содержит:

1. устройство монтируемой файловой системы;
2. точку монтирования;
3. тип файловой системы;
4. параметры монтирования;
5. флаг для `dump`, утилиты создания резервных копий;
6. порядок проверки для `'fsck'` (File System Check).

Здесь всегда есть запись о корневой файловой системе. Раздел `swar` является специальным, поэтому его не видно в древовидной структуре, и в поле точки монтирования для таких разделов всегда содержится ключевое слово `swar`.

Необходимо настроить `fstab`, чтобы система корректно загружалась. Настройка `fstab` будет примерно одинаковой как для способа установки системы на раздел жёсткого диска, так и для способа установки системы в виртуальную машину Qemu.

2.5.2.1. Настройка fstab

Создайте файл `fstab`:

```
cat > /etc/fstab << "EOF"
# Begin /etc/fstab

# file system mount-point type options dump fsck order

/dev/sdX / ext4 defaults 1 1
proc /proc proc nosuid,noexec,nodev 0 0
sysfs /sys sysfs nosuid,noexec,nodev 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
tmpfs /run tmpfs defaults 0 0
devtmpfs /dev devtmpfs mode=0755,nosuid 0 0
EOF
```

Замените `/dev/sdX` из первой строки `fstab` на метку корневого раздела, на который устанавливалась система. Например, `/dev/sda3`. Если вы устанавливаете систему в виртуальную машину, то метка корневого раздела будет называться примерно так: `/dev/nbd0pX`, где `X` - номер нужного раздела.

Если у вас UEFI, то добавьте соотв. запись в `fstab`:

```
echo "/dev/sdN /boot/efi vfat umask=0077 0 0" >> /etc/fstab
```

Замените `/dev/sdN` из команды выше на нужную метку раздела EFI. Повторимся, что его объём должен быть равен 256 Мб, файловая система - Fat32 (vfat).

Если вы используете `swap`-раздел, то и его пропишите в `fstab`:


```
echo "/dev/sdM swap swap pri=1 0 0" >> /etc/fstab
```

Замените `/dev/sdM` из команды выше на нужную метку раздела с подкачкой (swap).

2.5.2.2. Монтировать в ОЗУ или не монтировать? - вот в чём вопрос.

Как вы могли заметить в `/etc/fstab`, некоторые директории монтируются в оперативную память (посредством `tmpfs`). У такого способа главное достоинство в том, что при монтировании этих каталогов в `tmpfs`, система будет работать несколько быстрее. На современном оборудовании это не так заметно, но на старом и слабом может сделать систему немного быстрее.

Недостаток в том, что на несколько Мб система будет потреблять больше. Так, например, если система потребляла 28 Мб, то с монтированием в `tmpfs` будет потреблять 30 Мб. Потребление зависит от содержимого тех директорий.

Владельцам оборудования с небольшим объёмом ОЗУ (меньше 64 мегабайт) просьба учитывать этот факт и трезво оценить достоинства и недостатки выноса некоторых директорий в ОЗУ.

Дополнительно о метке корневого раздела с системой

Мы настоятельно советуем использовать вместо метки корневого раздела, на который установлена Calmira GNU/Linux (например, `/dev/sda1`, `/dev/hdc2`, etc.), его UUID. Если метка диска, прописанного в `/etc/fstab` изменится, то могут возникнуть проблемы с загрузкой ОС, либо же она не загрузится вообще.

Узнать UUID для нужного раздела можно, выполнив:

```
blkid /dev/sdX
```

Замените `/dev/sdX` на метку корневого раздела. Например, `/dev/sda3`.

Вывод будет примерно таким:

```
/dev/sda3: UUID="1ed63fa7-e4c9-43cc-96f4-9f951224a338" BLOCK_SIZE="4096" TYPE="ext4" PARTLABEL="Calmira data" PARTUUID="56ae06ac-dbd8-11eb-96fc-e8039ae29c93"
```

Вам понадобится что-то вроде:

```
UUID="1ed63fa7-e4c9-43cc-96f4-9f951224a338"
```

Запишите эту строку вместо `/dev/sdX` в `fstab` (но без кавычек). Например, `fstab` с UUID корневого раздела вместо его метки будет выглядеть так:

```
cat > /etc/fstab << "EOF"
# Begin /etc/fstab

# file system mount-point type options dump fsck order
UUID=1ed63fa7-e4c9-43cc-96f4-9f951224a338 / ext4 defaults 1 1
proc /proc proc nosuid,noexec,nodev 0 0
sysfs /sys sysfs nosuid,noexec,nodev 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
tmpfs /run tmpfs defaults 0 0
devtmpfs /dev devtmpfs mode=0755,nosuid 0 0
EOF
```

Замените `UUID=1ed63fa7-e4c9-43cc-96f4-9f951224a338` на полученное значение.

Этот способ является более надёжным, чем использование обычных меток дисков/разделов, которые могут в определённых ситуациях меняться.

2.5.3. Установка загрузчика GRUB

В Calmira используется классический загрузчик GRUB2. Без него система не сможет загрузиться. Процесс установки для MBR BIOS, Legacy BIOS/GPT BIOS, UEFI различаются. Здесь будут описаны все эти три случая.

Установка загрузчика в MBR

Для того, чтобы установить загрузчик, выполните:

```
grub-install /dev/sdX
```

Замените `sdX` на метку диска. Например, `/dev/sda` или `/dev/nbd0`.



ЗАМЕЧАНИЕ.

В предыдущих инструкциях нужно было заменить `sdX` на метку раздела (например, на `sda3`). Сейчас же нужно заменить на метку ДИСКА (например, `sda`). Полученная команда будет `grub-install /dev/sda`, если вы выбрали установку загрузчика на этот диск. НЕ ВЫБИРАЙТЕ установку в раздел (`sda3`) - ни к чему это не приведёт.

2.5.3.1. Установка загрузчика Legacy BIOS

Если вы по каким-то причинам используете Legacy BIOS, либо же обычный BIOS, но у жёсткого диска таблица разделов GPT, то для установки загрузчика вам необходимо иметь раздел объёмом 1 Мб без файловой системы в начале диска. Если у вас на ПК уже установлена какая-либо ОС, то этот раздел должен присутствовать.

Установите загрузчик командой:

```
grub-install /dev/sdX
```

Замените *sdX* на метку диска. Например, */dev/sda* или */dev/nbd0*.



ЗАМЕЧАНИЕ.

В предыдущих инструкциях нужно было заменить *sdX* на метку раздела (например, на *sda3*). Сейчас же нужно заменить на метку ДИСКА (например, *sda*). Полученная команда будет `grub-install /dev/sda`, если вы выбрали установку загрузчика на этот диск. НЕ ВЫБИРАЙТЕ установку в раздел (*sda3*) - ни к чему это не приведёт.

2.5.3.1. Установка загрузчика EFI

На данный момент все современные ПК имеют UEFI, а не BIOS. Для установки системы на UEFI, вы должны иметь доп. раздел объёмом 256 Мб с файловой системой *fat32* и названием *ESP*. Если у вас на ПК уже установлена какая-либо ОС, то этот раздел уже имеется. Смонтируйте его в */boot/efi* командой `mount`. После чего необходимо установить несколько пакетов из системы портов для правильной работы загрузчика.

Установите из все порты из *base/grub-efi* в следующем порядке:

- `efivar`;
- `popt` (находится не в *base/grub-efi*, а в *general_libs*);
- `efibootmgr`;
- `grub`.

Для этого выполнить:

```
cd /usr/ports/$КАТЕГОРИЯ/$ПАКЕТ
./install
```

- *\$КАТЕГОРИЯ* - категория, в котором располагается порт. Например, *base/grub-efi* или *general_libs*;
- *\$ПАКЕТ* - имя нужного пакета. Например, ``efibootmgr``.

После чего можно устанавливать `grub`:

```
grub-install /dev/sdX
```

Замените *sdX* на метку диска. Например, */dev/sda* или */dev/nbd0*.



ЗАМЕЧАНИЕ.

В предыдущих инструкциях нужно было заменить *sdX* на метку раздела (например, на *sda3*). Сейчас же нужно заменить на метку ДИСКА (например, *sda*). Полученная команда будет `grub-install /dev/sda`, если вы выбрали установку загрузчика на этот диск. НЕ ВЫБИРАЙТЕ установку в раздел (*sda3*) - ни к чему это не приведёт.

2.5.4. Создание конфига GRUB (актуально для установки загрузчика на MBR или Legacy BIOS)

Создайте конфиг следующей командой:

```
cat > /boot/grub/grub.cfg << "EOF"
# Begin /boot/grub/grub.cfg
set default=0
set timeout=5

insmod ext2
set root=(hd0,2)

menuentry "Calmira LX4 1.1 GNU/Linux, Linux" {
    linux /boot/vmlinuz root=/dev/sda2 ro
}
EOF
```

Строка `set timeout=5` устанавливает таймаут (время ожидания автоматической загрузки системы, если пользователь не выбрал ни одного элемента из загрузочного меню GRUB), равный пяти секундам. Измените это значение при необходимости.

Строка `set root=(hd0,2)` устанавливает нужный жёсткий диск. Если вы не знаете какой выбрать, наберите в консоли GRUB строки:

```
ls
ls ($DISK)/boot/grub
```

Первая команда выдаст список дисков, а вторая выдаст список файлов загрузчика, если выбран нужный диск с системой (замените *\$DISK* на нужный диск из вывода первой команды).

Строка `linux /boot/vmlinuz root=/dev/sda2 ro` указывает нахождение ядра Linux (`linux /boot/vmlinuz`) и его нахождение на определённом разделе (`root=/dev/sda2`). Замените `/dev/sda2` на ваш корневой раздел, куда копировалась система в предыдущей инструкции.

2.6. Завершение установки. Выход из chroot и перезагрузка

Установка и загрузка системы завершена. Теперь вы можете выйти из chroot и перезагрузить компьютер.

2.6.1. Выход из chroot

Для того, чтобы выйти из chroot и отмонтировать виртуальные ФС ядра, выполните:

```
logout

sudo umount /mnt/calmira_system/dev{/pts,}
sudo umount /mnt/calmira_system/{sys,proc,run}
```

2.6.2. Размонтирование

Теперь можно отмонтировать раздел/образ жёсткого диска из системы.

2.6.2.1. При установке в раздел

```
sudo umount /dev/sdX
```

Замените *sdX* на метку раздела.

2.6.2.2. При установке на виртуальную машину

1. Отмонтируйте образ виртуального жёсткого диска;
2. Отключите образ ВЖД от системы;
3. Отключите модуль ядра nbd.

```
sudo umount /mnt/calmira_system
sudo qemu-nbd --disconnect /dev/nbd0
sudo rmmmod nbd
```

2.6.3. Перезагрузка

Введите:

```
sudo reboot
```

Для перезагрузки.

3.0. Настройка дистрибутива

В данном разделе приведены сведения о настройке системы Calmira GNU/Linux.

3.1 [Настройка окружения](#)

3.2 [Настройка часов](#)

3.3 [Настройка клавиатуры и шрифта в TTY](#)

3.4 [Настройка inittab](#)

3.1. Настройка окружения и конфигурационных файлов

В данном разделе содержится описание конфигурационных файлов Calmira.

3.1.1. Редактирование текста приветствия

```
* Activating all swap files/partitions... [ OK ]
Mounting root file system in read-only mode... [ 2.756238] EXT4-fs (sda
2): re-mounted. Opts: (null). Quota mode: none.
* [ OK ]
Checking file systems...[ 2.767473] random: crng init done
[ 2.767980] random: 3 urandom warning(s) missed due to ratelimiting
* [ OK ]
Remounting root file system in read-write mode...[ 2.853062] EXT4-fs (s
da2): re-mounted. Opts: (null). Quota mode: none.
* [ OK ]
* Mounting remaining file systems... [ OK ]
* Cleaning file systems: /tmp [ OK ]
* Retrying failed uevents, if any... [ OK ]
* Setting up Linux console... [ OK ]
INIT: Entering runlevel: 3
* Starting system log daemon... [ OK ]
* Starting kernel log daemon... [ OK ]

Welcome to Calmira GNU/Linux! Developer ONLY version!
Do not use this version permanently. Do not leak assembly data
to third parties!

Build 18.07.1.2021

test login:
```

Перед тем, как система запросит ваш логин и пароль, она выведет приветственное сообщение с базовой информацией о дистрибутиве. Если вам оно не нравится, либо вы хотите его заменить на нужное вам, отредактируйте файл `/etc/issue`:

```
# Очистка файла, если вам не надо, чтобы
# отображалось приветственное сообщение:
> /etc/issue

# Редактирование файла:
vim /etc/issue
```

3.1.2. Редактирование общесистемных настроек окружения

Основной упор в Calmira GNU/Linux на работу в TTY, а не графике. Поэтому пользователям было бы неплохо отредактировать настройки по умолчанию для более комфортной работы. Общесистемные настройки окружения находятся в файле `/etc/profile`.

Функции `pathremove`, `pathprepend`, `pathappend` предназначены для работы с PATH. Функция `ver` отображает информацию о релизе дистрибутива Calmira, а `system_welcome` показывает информацию о базовых командах и предназначен для вставки в `/etc/bashrc`, `/etc/skel/.bashrc` и/или `~/.bashrc`.

По умолчанию в PATH содержатся директории `/bin`, `/sbin`, `/usr/bin`, `/usr/sbin`. Если вы создаёте обычного пользователя без привилегий, специфичных пользователю `root`, то **обязательно** уберите из PATH директории `/sbin` и `/usr/sbin`. Обычному пользователю программы оттуда не понадобятся, так же

эта операция обезопасит вас и непривилегированного пользователя от всевозможных ошибок и вирусов. Для этого в `~/.bashrc` или `~/.profile` обычного (не root) пользователя добавьте строки:

```
unset PATH
PATH=/bin:/usr/bin
```

Если вы создали нового пользователя, то скопируйте все скрытые файлы из `/etc/skel/*`, если они не скопировались автоматически при создании этого пользователя. В конфигурационных файлах из `/etc/skel` уже произведены все настройки по умолчанию, в т.ч. настроен PATH для более безопасной и надёжной работы.

По умолчанию системное приглашение к вводу безцветное и непривлекательное. Если вы хотите переделать его, то отредактируйте значение переменной `PS1`. Возможно использование escape-последовательностей для изменения цвета приглашения. Так же дополнительные последовательности, из которых строится приглашение к вводу в консоль:

Последовательность	Значение	Последовательность	Значение
\a	Издать звуковой сигнал	\@	Текущее время в 12-часовом формате <i>AM/PM</i>
\d	Текущая дата в формате <i>день_недели</i> <i>месяц</i> <i>число</i> : Mon May 16	\A	Текущее время в 24-часовом формате <i>часы:минуты</i>
\h	Имя локальной машины -- имя домена	\r	Имя пользователя
\H	Полное имя хоста	\v	Номер версии командной оболочки
\j	Число заданий, действующих в текущем терминале	\V	Номер версии и выпуска командной оболочки
\l	Имя текущего устр-ва терминала	\w	Имя текущего рабочего каталога
\n	Переход на новую строку	\W	Последняя часть имени текущего рабочего каталога
\r	Возврат каретки	\!	Имя текущей команды в истории
\s	Имя программы в программной оболочке	\#	Число команд, введённых в текущем сеансе
\t	Текущее время в 24-часовом формате	\\$	Выводит \$, если пользователь не является root, если это root, то выводит #

\T	Текущее время в 12- часовом формате	[и \]	Отмечает начало и конец (соотв.) последовательности непечатаемых символов
----	-------------------------------------	--------	---

3.1.3. Редактирование настроек bash

Интерпретатором по умолчанию в Calmira является bash. Его настройки расположены в `/etc/bashrc`, `/etc/skel/.bashrc` и `~/.bashrc`. Рекомендуем вам редактировать файл `~/.bashrc`.

3.1.4. Редактирование настроек Vim

В Calmira GNU/Linux по умолчанию используется редактор Vim. Его настройки расположены в файле `/etc/vimrc`. Строки:

```
set nu
set lbr
```

Предназначены для отображения номеров строк сбоку и переноса строк по словам соотв.

3.1.5. Редактирование настроек file system table

В файле `/etc/fstab` приведены базовые настройки монтирования файловых систем. Не изменяйте следующие строки, без которых система не сможет загрузиться корректно:

```
proc      /proc      proc      nosuid,noexec,nodev 0 0
sysfs     /sys       sysfs     nosuid,noexec,nodev 0 0
devpts    /dev/pts   devpts    gid=5,mode=620 0 0
tmpfs     /run       tmpfs     defaults        0 0
devtmpfs  /dev       devtmpfs  mode=0755,nosuid 0 0
```

3.1.6. Установка имени хоста

```
echo "calmira_pc" > /etc/hostname
```

Замените `calmira_pc` на имя хоста.

[Назад](#)

Настройка дистрибутива

[Домой](#)

[Далее](#)

Настройка часов

3.2. Настройка часов

При загрузке считывается информация из аппаратных часов - CMOS. Программно нельзя определить, какой часовой пояс используют часы CMOS, однако вы можете выполнить команду `hwclock --localtime --show` и сравнить результат с местным временем. Если он не совпадает - ваши часы, скорее всего, используют UTC.

3.2.1. `/etc/sysconfig/clock`

Создайте нужный файл, определяющий, использует ли CMOS UTC. Если не использует, то замените значение переменной `UTC` на 0.

```
cat > /etc/sysconfig/clock << "EOF"
# Begin /etc/sysconfig/clock

UTC=1

# Set this to any options you might need to give to hwclock,
# such as machine hardware clock type for Alphas.
CLOCKPARAMS=

# End /etc/sysconfig/clock
EOF
```

3.2.2. Выбор часового пояса.

Для выбора нужного часового пояса выполните следующий скрипт:

```
tzselect
```

Для сохранения нужного часового пояса выполните:

```
In -svf /usr/share/zoneinfo/<xxx> /etc/localtime
```

Замените `<xxx>` на ваш часовой пояс. Это значение было получено из вывода `tzselect`.

3.3. Настройка клавиатуры (раскладки) и шрифта в ТТУ

Дистрибутив Calmira GNU/Linux предназначен для работы в ТТУ в первую очередь. Некоторым пользователям может понадобиться переключение раскладки клавиатуры с английской на свою родную. В системе Calmira уже предустановлен нужный софт для настройки раскладок.

3.3.1. Файлы

Настройка раскладок производится в файле `/etc/sysconfig/console`.

Раскладки клавиатуры содержатся в директории `/usr/share/keymaps`. Шрифты в `/usr/share/consolefonts`.

3.3.2. Параметры в конфигурационном файле

См. предыдущий пункт.

- `UNICODE` - при значении 1, yes или true переводит консоль в режим UTF-8, что важно для некоторых раскладок и шрифтов, например, русских.
- `KEYMAP` содержит аргументы для программы `loadkeys`, которая загружает раскладки клавиатуры. Значение этого параметра - имя загружаемой раскладки.
- `FONT` - имя шрифта консоли. Содержит аргументы для программы `setfont`. Значение этого параметра - имя загружаемого шрифта.

3.3.3. Установка раскладки клавиатуры

Для поиска нужной раскладки выполнить:

```
find /usr/share/keymaps --type f
```

Выберите нужную раскладку. Например, для русской qwerty-раскладки, это будут варианты:

1. `ru` - переключение раскладки не установлено автором (либо же вовсе отсутствует).
2. `ruwin_alt_sh-UTF-8` - переключение раскладки по Alt+Shift.
3. `ruwin_cplk-UTF-8` - переключение раскладки по Caps Lock.
4. `ruwin-ct_sh-UTF-8` - переключение раскладки по Ctrl+Shift.

Как вы могли заметить, значение раскладки берётся из вывода предыдущей команды, но без расширения на конце и полного пути. Подставьте полученный результат как значение переменной `KEYMAP`, например:

```
KEYMAP="ruwin_alt_sh-UTF-8"
```

3.3.4. Установка шрифта

Для корректного отображения символов в родной для пользователя локализации (в т.ч. и для смысла настройки раскладок клавиатуры) нужно установить шрифт, поддерживающий эту локализацию.

Для просмотра всех установленных раскладок клавиатуры выполните:

```
ls /usr/share/consolefonts
```

Из списка выберите нужный шрифт. Например, для корректного отображения кириллицы и латиницы неплохо подходит шрифт *cyr-sun16*. Подставьте нужное значение в переменную FONT, например:

```
FONT="cyr-sun16"
```

3.3.5. Пример готового конфигурационного файла

```
# Begin /etc/sysconfig/console

UNICODE="1"
KEYMAP="ruwin_alt_sh-UTF-8"
FONT="cyr-sun16"

# End /etc/sysconfig/console
```

[Назад](#)
Настройка часов

[Домой](#)

[Далее](#)
Настройка inittab

3.4. Настройка inittab

Система инициализации SystemVinit читает параметры из файла `/etc/inittab`. Так же там описан запуск программ `sulogin` (программа для логина пользователей), `agetty` (альтернатива `getty` - программа, управляющая доступом к ТТУ).

По умолчанию этот файл уже создан. Для того, чтобы отредактировать его, выполните:

```
vim /etc/inittab
```



ВНИМАНИЕ!

Не рекомендуется редактировать этот файл!

Редактирование этого файла может быть полезно, если нужно изменить число ТТУ, например. По умолчанию их 6:

```
1:2345:respawn:/sbin/agetty --noclear tty1 9600
2:2345:respawn:/sbin/agetty tty2 9600
3:2345:respawn:/sbin/agetty tty3 9600
4:2345:respawn:/sbin/agetty tty4 9600
5:2345:respawn:/sbin/agetty tty5 9600
6:2345:respawn:/sbin/agetty tty6 9600
```

Для добавления новых ТТУ добавьте строку:

```
$NUMBER:2345:respawn:/sbin/agetty tty$NUMBER 9600
```

Замените `$NUMBER` на номер ТТУ.

Переключаться между ТТУ можно сочетанием клавиш `Ctrl + Alt + Fx` (где `x` - номер функциональной клавиши, равный номеру ТТУ), или `Alt + Стрелки влево/вправо`.

4.1. Введение в порты

Порты - относительно молодой (в Calmira), но, в тоже время, очень удобный инструмент управления ПО. На данный момент, с помощью портов можно только устанавливать и удалять ПО, но в скором времени будут доступны новые функции.

4.1.1. Введение

Система портов находится в */usr/ports*. Всё ПО разделено на категории:

1. base - базовые порты, входящие в минимальную поставку дистрибутива;
2. kernel - ядро и его компоненты;
3. console - консольные утилиты;
4. optional - дополнительный софт, не вошедший в базовую поставку дистрибутива, но который может понадобиться после установки системы;
5. xorg - сервер X и другое ПО, такое как оконные менеджеры и рабочие окружения;

В каждой категории несколько подкаталогов с портом программы. Первые версии портов включают в себя только файл *install* с инструкциями по сборке, либо ещё *info* с информацией о пакете. В новых портах вместо *info* используется файл *config.sh*, который предназначается ещё и для занесения пакета в базу данных срkg.

4.1.2. Установка программы из портов

Для того, чтобы установить какую-либо программу из порта, нужно выполнить:

```
# Переход в нужную директорию
cd /usr/ports/$КАТЕГОРИЯ/$ПРОГРАММА

# Выполнение инструкций сборки и установки
./install
```

Всё ПО из портов собирается из исходного кода. В некоторых случаях сборка может занять довольно много времени (так, например, пакет *gcc* на очень слабом железе может собираться до трёх дней). Однако, у сборки очень много преимуществ:

- Возможность оптимизации программы под своё железо;
- Большой контроль при установке;
- Выбор нужных функций и удаление ненужных (если программа это поддерживает).

Иногда пакеты предоставляют к установке дополнительную документацию. Она не всем нужна, поэтому мы оставили в портах возможность выбора пользователем: ставить её или нет. Система портов всегда спрашивает пользователя об этом. Если вы отказались от установки такой документации в будущем, но она вам пригодилась позже - перейдите в директорию с портом и выполните:

```
chmod +x install_doc.sh
./install_doc.sh
```

Этими командами вы установите документацию пакета.

Настоятельно рекомендуется использовать порты вместо установки бинарных пакетов посредством срkg.

Кстати, портами скоро можно будет управлять и с помощью пакетного менеджера `crkg`. Ко второй версии Calmira планируется добавить просмотр информации и удаление пакета, установленного из порта, а так же множество других функций.

4.1.3. port-пакеты `crkg`

Пакетный менеджер `crkg` умеет устанавливать не только бинарные пакеты, но и собирать ПО из исходного кода. Но у такого подхода несколько минусов:

- Отсутствие контроля. Если в портах каждую инструкцию можно изменить так, как нужно человеку, то в случае с port-пакетами `crkg` так не случится. Для редактирования инструкций сборки придётся распаковывать пакет, править его, а потом запаковывать обратно.
- Нестабильность. Несмотря на то, что механизм port-пакетов уже достаточно стабилен, могут возникать ошибки.
- Небольшое число пакетов. Для пакетного менеджера `crkg` очень мало port-пакетов. Выбор очень невелик, а единственный репозиторий с port-пакетами было принято решение закрыть после выхода полноценной системы портов.

Рекомендуется использовать систему портов, а не сборку port-пакетов с помощью `crkg`.

[Назад](#)

Управление пакетами

[Домой](#)

[Далее](#)

Введение в `crkg`

4.2. Введение в срkg

срkg - написанный с нуля пакетный менеджер для Calmira GNU/Linux. Впервые он вошёл в состав Calmira GNU/Linux 2021.2 на смену менее надёжному и функциональному срkgi-tools. Из состава Calmira LX4 1.0 GNU/Linux он был убран ввиду своей нестабильности, но вернулся в Calmira LX4 1.1.

4.2.1. Обзор функций

1. Установка;
2. Удаление;
3. Просмотр информации.

Эти 3 функции - основные в любом пакетном менеджере (далее - ПМ). Так же есть несколько дополнительных:

1. Создание локального репозитория с пакетами срkg. Уже неактуально, так как из этого ПМ удалили скачивание пакетов из репозиториев. В следующих релизах это будет удалено.
2. Вывод списка установленных пакетов;
3. Очистка. Очистка кеша: `/usr/src`, `/var/cache/cpkg/archives`.

Все остальные функции предназначены для разработчиков и не могут быть использованы для работы с пакетами.

4.2.2. Функции

1. `install` - установить пакет;
2. `remove` - удалить пакет;
3. `info` - просмотр информации о пакете;
4. `help` - просмотр справки об использовании.

4.2.3. Установка пакета

Для того, чтобы установить пакет, введите:

```
срkg install package.txz
```

Для краткости можно заменить опцию `install` на ключ `-i`.

Пакет копируется в `/var/cache/cpkg/archives` и распаковывается там. После чего данные пакета копируются в нужные папки и добавляются в базу данных срkg.

4.2.3.1. Удовлетворение зависимостей пакета

Учитывайте то, что срkg НЕ обрабатывает зависимости. Он может отображать зависимости определённого пакета, но устанавливать их вы должны сами. срkg выводит список зависимостей пакета перед тем, как установить его. Без наличия *необходимых* зависимостей пакет не будет работать (либо будет работать некорректно). Так же пакет не будет работать без некоторых *рекомендуемых* зависимостей. *Опциональные* зависимости нужны для добавления каких-либо новых функций в исходный пакет. Если вы не хотите их ставить (или вам они не нужны), то опциональные зависимости можно не устанавливать.

4.2.4. Удаление пакета

Для удаления выполните:

```
сpkg remove package
```

Для краткости можно заменить опцию `remove` на ключ `-r`.

Читается `config.sh` пакета из базы данных `сpkg`, и на основе данных из этого файла удаляется пакет. После чего он выносится из базы данных `сpkg`.

Просмотр информации о пакете

Для того, чтобы узнать версию, сборщика пакета, зависимости и другую информацию, выполните:

```
сpkg info package
```

Для краткости можно заменить опцию `info` на `-l`.

4.2.5. Дополнительная информация

Для просмотра доп. справки выполните:

```
сpkg help
```

Для просмотра бОльших технических особенностей и характеристик зайдите в репозиторий `сpkg` на [GitHub](#).

[Назад](#)

Введение в порты

[Домой](#)

[Далее](#)

Создание порта

4.3. Создание порта

От помощи в создании порта для сборки какого-либо пакета мы никогда не откажемся. Если есть какие-либо заинтересованные в помощи люди - вы нам нужны.

4.3.1. Начальный этап

Создайте у себя на GitLab форк [этого](#) репозитория, после чего приступайте к работе. Советуем вам клонировать этот репозиторий себе на ПК для удобной работы:

```
git clone https://gitlab.com/USERNAME/calmira_ports
cd calmira_ports
```

Замените *USERNAME* на имя пользователя GitLab.

После этих действий вы можете приступать к созданию порта. Хотим обратить ваше внимание на то, что строение порта должно повторять строение других портов.

4.3.2. Структура системы портов

Просим вас соблюдать структуру системы портов:

```
КАТЕГОРИЯ
├── makeport.sh
└── ИМЯ_ПАКЕТА
    ├── install
    ├── install_doc.sh
    └── config.sh
```

Здесь приведена общая структура системы портов. КАТЕГОРИЯ и ИМЯ_ПАКЕТА - директории. КАТЕГОРИЯ - имя категории, в которую входит пакет с именем ИМЯ_ПАКЕТА.

Файл *makeport.sh* нужен для автоматизации создания порта. Он создаёт директорию с именем порта, а так же файл *install* и открывает его в вашем текстовом редакторе, установленном по умолчанию. Наличие этого файла необязательно. Синтаксис:

```
./makeport.sh $ИМЯ_ПАКЕТА
```

Файл *install* необходим. Он содержит инструкции по сборке ПО из исходного кода. Файл *install_doc.sh* опционален. Если у пакета есть дополнительная документация, которая не устанавливается по умолчанию, то в файле *install_doc.sh* содержатся инструкции по установке документации. Запускать его или нет - зависит от выбора пользователя. Он выбирает нужный ответ в диалоговом окне.

Файл *config.sh* содержит информацию о пакете. Она может быть добавлена в базу данных cpkg для удобного управления пакетом (так как система портов не умеет отображать информацию о пакете и удалять его, для таких целей используется cpkg). Наличие этого файла **ОБЯЗАТЕЛЬНО**, так же как и файла *install*.



Смотрите также:

Для установки какого-либо текстового редактора по умолчанию (в ТТУ) экспортируйте переменную EDITOR, значение которой - путь до исполняемого бинарного файла редактора.

4.3.3. Создание нужных файлов

Перейдите в одну из нужных директорий, выполните:

```
./makeport.sh $PACKAGE_NAME
```

Замените \$PACKAGE_NAME на имя порта. Скрипт makeport.sh создаёт нужную директорию и записывает в файл нужную информацию, которая должна содержаться в файле порта.

Если вы не нашли этот скрипт, скопируйте его из любой другой директории.

Будет создан файл *install* в директории, имеющей имя порта, который вы хотите создать. Этот файл будет открыт для редактирования в вашем текстовом редакторе, установленном по умолчанию. Замените строки

```
# Port created by Linuxoid85  
#  
# (C) 2021 Michail [Linuxoid85] Krasnov
```

на строки с вашим именем.

Строки

```
wget  
tar -xf  
cd
```

Предназначены для того, чтобы подставить в их конец нужную информацию. wget скачивает архив с исходным кодом, tar -xf распаковывает его, а cd переходит в распакованную директорию.

Запишите в конец файла нужные инструкции для сборки. Доступны все команды bash.

Если можно установить дополнительную документацию, то в конце открытого в редакторе файла вызовите функцию print_document_dial:

```
print_document_dial
```

Он будет запрашивать у пользователя, ставить ли дополнительную документацию пакета.

Запишите нужные инструкции для установки документации в файл install_doc.sh и поместите его в ту же директорию, что и файл install.

После чего создайте файл config.sh и запишите в него информацию о пакете. Строение этого файла аналогично тому, что используется в пакетном менеджере cpkg. О создании этого файла читайте [в этой](#) странице руководства.

4.3.4. Отправка изменений на github

Если вы клонировали репозиторий себе на ПК, выполните:

```
# Фиксирование изменений
git add .
git commit -m "Добавление порта с пакетом $PACKAGE_NAME"

# Отправка изменений себе на GitLab
git push
```

Замените `$PACKAGE_NAME` на имя порта с пакетом.

Если вы делали всё в Web, то отметьте изменения с помощью графического интерфейса GitLab.

Создайте запрос на слияние (pull-request). Создание порта завершено!

[Назад](#)

Введение в сpkg

[Домой](#)

[Далее](#)

Создание бинарного
пакета для сpkg

4.4. Создание бинарного пакета для cpkg

За время использования ОС вы можете захотеть сделать бинарный пакет для cpkg с какой-либо программой для её удобной установки. В данной статье рассказано, как сделать это. Несмотря на то, что основной упор в Calmira делается на сборку ПО из исходного кода, разработчики не препятствуют установке готовых бинарных пакетов, хотя их официально не предоставляют.

4.4.1. Введение

Пакет для cpkg представляет собой tar архив, сжатый методом xz. В архиве находится директория PKG, в которой находятся файлы config.sh, preinst.sh/postremove.sh (опционально), postinst.sh/postremove.sh (опционально), а так же директория pkg, в которой находится сам пакет. Дерево каталогов пакета должно совпадать с деревом каталогов операционной системы.

Назначение файлов и директорий:

- PKG - в этой директории находятся сам пакет и файлы описания, а так же файлы пред- и послеустановочных настроек.
- PKG/pkg - директория с пакетом
- PKG/config.sh - описание пакета
- PKG/preinst.sh - скрипт, выполняющийся до установки пакета. Может быть полезен для настройки системы или окружения перед установкой пакета. Наличие этого файла **опционально**.
- PKG/postinst.sh - скрипт, выполняющийся после установки пакета. Может быть полезен для настройки пакета. Наличие этого файла **опционально**.
- PKG/preremove.sh - скрипт, выполняющийся до удаления пакета. Может быть полезен для настройки системы или окружения.
- PKG/postremove.sh - скрипт, выполняющийся после удаления пакета. Может быть полезен для окончательной настройки системы/окружения, а так же удаления некоторых конфигурационных (и других) файлов, не вошедших в список файлов пакета.
- PKG/metadata.xz - в данном архиве содержится информация о совместимости пакета с дистрибутивом Calmira.

Тогда дерево каталогов пакета будет примерно таким:

```
some_package.txz # <-- Архив с пакетом
├── PKG # <-- Директория, в котором находится пакет
│   ├── config.sh
│   ├── {post,pre}inst.sh
│   ├── {post,pre}remove.sh
│   ├── metadata.xz
│   └── pkg # <-- Директория, в котором находятся данные пакета
│       ├── usr
│       ├── etc
│       ├── var
│       └── ...
```

В случае, если это пакет с исходным кодом (port-пакет), в директории PKG наличие файла port.sh **обязательно**. В нём описываются инструкции по сборке и установке пакета.

Наличие шебанга обязательно:

```
#!/bin/bash
```

Пример:

```
#!/bin/bash

cd /var/cache/cpkg/archives # Переход в рабочую директорию
wget https://www.some.site/some_package.tar.gz # Скачивание пакета some_package.tar.gz с сайта https://www.some.site
tar -xvf some_package.tar.gz # Распаковка пакета с исходным кодом

cd some_package
./configure --prefix=/usr \
--bindir=/usr/bin \
--sysconfdir=/etc # Конфигурирование пакета
make # Сборка
make install # Установка
```



ВНИМАНИЕ!

Мы **НАСТОЯТЕЛЬНО НЕ** рекомендуем использовать port-пакеты.
Если хотите автоматизировать сборку ПО из исходного кода, то
используйте [систему портов Calmira GNU/Linux](#).

4.4.1.1. Строение файла config.sh

В этом файле описывается сам пакет. В нём указывается имя пакета, версия, мейнтейнер, описание и файлы пакета. Пример такого файла:

```
NAME=some
VERSION=1.0
MAINTAINER="Linuxoid85 "
REQ_DEPS="bash"
RECOM_DEPS="openssl freetype"
OPT_DEPS="coreutils"
TEST_DEPS="expect"
BEF_DEPS="cpkg"
CON_DEPS="wget make-ca"
PRIORITY=user
DESCRIPTION="Some package for test cpkg package manager"
FILES="/usr/bin/{some_pkg,test_cpkg} /usr/share/some_pkg/"
```

Описание переменных:

1. NAME - имя пакета;
2. VERSION - версия пакета/программы;
3. MAINTAINER - сборщик (сопровождающий) пакета;
4. REQ_DEPS - необходимые для работы пакета зависимости;
5. RECOM_DEPS - рекомендуемые зависимости;
6. OPT_DEPS - опциональные зависимости;
7. TEST_DEPS - необходимые для тестирования пакета зависимости (указывать только в port-пакетах);
8. BEF_DEPS - какие пакеты требуют, чтобы указанный пакет был собран и установлен до их сборки и установки;
9. CON_DEPS - зависимости, с которыми конфликтует данный пакет;
10. PRIORITY - приоритет пакета;
11. DESCRIPTION - описание пакета;
12. SITE - сайт пакета, либо ссылка, откуда можно скачать оригинал/архив с исходным кодом, из которого собирался пакет;
13. FILES - все файлы пакета.

Если в одной директории (чаще всего это /bin, /usr/bin, /etc и пр.) несколько файлов из пакета, то нет смысла дублировать пути до этих файлов. Проще объединить их в массив. К примеру, есть директория /usr/bin, в которую устанавливаются файлы some_pkg и test_cpkg. В строке FILES="..." они объединены в массив /usr/bin/{some_pkg,test_cpkg}. То есть, файлы перечисляются в фигурных скобках {}.

Так же, все отдельные файлы разделяются между собой пробелами.

В версии cpkg 1.0ra4 была добавлена функция просмотра зависимостей

пакета (при установке, удалении и просмотра информации о пакете).

Типов зависимостей несколько:

Наименование	Объяснение	В какой строке вывода
REQ_DEPS	Необходимые зависимости. Если вы устанавливаете бинарный пакет, то необходимые зависимости можно установить после, но если вы ставите port-пакет, то поставьте сначала необходимые зависимости, а только потом пакет.	Необходимые:
RECOM_DEPS	Рекомендуемые зависимости. Советуется устанавливать их	Рекомендуемые:
OPT_DEPS	Опциональные зависимости. Устанавливать их необязательно, они лишь служат для добавления определённого функционала. Порядок установки таких зависимостей в большинстве случаев не важен (кроме port-пакетов, где опциональные должны быть установлены पहले).	Опциональные:
TEST_DEPS	Зависимости, необходимые для тестирования пакета.	Для тестирования:
CON_DEPS	Зависимости, которые конфликтуют с данным пакетом. Их желательно удалить во избежание сбоев в работе как их самих, так и (в некоторых случаях) Calmira GNU/Linux	Конфликтует
BEF_DEPS	Необязательные зависимости, определяющие, что указанный port-пакет должен собран ДО установки/сборки тех зависимостей	Установлен перед ними:

Помимо этого, есть приоритет пакета - `PRIORITY`. Только два значения: `system`, если пакет системный и `user` - если пользовательский. Т.е., в первую группу входят все пакеты, которые обеспечивают корректную работу МИНИМАЛЬНОЙ системы. Системные пакеты удалить нельзя. Их можно лишь обновлять до новой версии и просматривать информацию о них. А с пользовательскими пакетами (приоритет `user`) можно делать всё, что угодно.

Настоятельно рекомендуем использовать пользовательский (`user`) пакета! Все системные пакеты собираются одним мейнтейнером для поддержания работоспособности системы. Как правило, большинство пакетов от сторонних сборщиков с приоритетом `system` не только не являются системными, но и при удалении этих пакетов в системе не произойдёт ничего.

4.4.2. Сборка пакета

Создайте некую директорию `/usr/PKG/pkg`, в которую, впоследствии, будет установлен пакет. И выполните процедуру сборки. Но тут загвоздка. Использовать альтернативный префикс `/usr/PKG/pkg` в процессе конфигурирования (напр., `configure`) сделает пакет нерабочим. Будут сделаны неправильные ссылки, файлы будут скопированы не в те директории. В каталоге `/usr/PKG/pkg` нужно создать зеркальную копию `/usr`, либо же корня (`/`) - в случае, если пакет системный. Поэтому при конфигурировании пакета (напр., создании `Makefile`) используйте префикс `/usr` или `/` (по умолчанию). А потом с помощью `make DESTDIR=/usr/PKG/pkg install` сфальсифицировать установку пакета в корень, либо же в указанные директории (когда как на самом деле пакет будет установлен в `/usr/PKG/pkg`).

В Calmira GNU/Linux LX4 1.1 используется стандартная структура директорий, где `/bin`, `/sbin`, `/usr/bin` и `/usr/sbin` - разные каталоги. Поэтому если бинарные файлы должны быть установлены в `/bin`, бинарные файлы для суперпользователя в `/sbin`, и пр. При необходимости (если некоторые файлы пакета должны быть установлены в `/bin`, `/sbin`, `/lib`) используйте ключи `--bindir=/bin`, `--sbin=/sbin`, `--libexecdir=/lib`.

Выполните процедуру конфигурирования:

```
./configure --prefix=$PREFIX \  
--bindir=$BINDIR \  
--sbindir=$SBINDIR
```

Если вы используете систему сборки `meson`, то замените процедуру конфигурирования на корректную для этой системы сборки.

Используйте нужные вам ключи. В любом случае, процедура конфигурирования должна быть точно такой, какой она является по умолчанию.

Если конфигурирование прошло успешно, выполните сборку:

```
make
```

Если используете `meson`, то сборка будет такой:

```
ninja
```

После чего установите пакет, только не в те директории, что были указаны `configure/meson`, а в `/usr/PKG/pkg`:

```
make DESTDIR=/usr/PKG/pkg install
```

Вместо `DESTDIR` могут быть переменные `INSTALL_DIR`, `prefix` и пр. Или может не быть вообще, тогда файлы придётся находить и копировать вручную. О подробной информации смотрите в документации к установке пакета.

Теперь создайте файл `config.sh` в директории `/usr/PKG` и запишите в него информацию о пакете:

1. Название пакета;
2. Версия пакета;
3. Описание пакета;
4. Сборщик пакета;
5. Зависимости пакета;
6. Файлы пакета.

И прочую информацию. Запишите её в соответствии с пунктом "Введение".

Для того, чтобы записать список файлов в раздел `FILE`, перейдите в директорию `pkg` и выведите список файлов:

```
cd pkg  
find
```

Запишите вывод в раздел `FILE`:

```
FILE="/usr/bin/file  
....  
....  
"
```

Удалите из вывода все точки в начале, а так же все файлы, относящиеся к другим пакетам. Так же удалите все ``/bin``, ``/usr/bin``, ``/usr/share`` и пр. Оставьте только файлы. Одна ошибка - и при удалении пакета можно сломать либо

какой-то отдельный пакет, либо всю систему.

Если у пакета должны быть *preinstall* и *postinstall* скрипты для настройки окружения и пакета соотв., то создайте их. Назовите `preinst.sh` и `postinst.sh` соотв.

И соберите пакет:

```
# Переход из /usr/PKG на уровень ниже (в /usr):  
cd ..  
  
# Сборка пакета  
tar -cf PKG $PACKAGE_NAME.txz -J
```

Замените `$PACKAGE_NAME` на имя. Оно должно включать в себя имя и версию пакета, например: `some_1.0.txz`.

[Назад](#)

Создание порта

[Домой](#)

[Далее](#)