

**DVD ДАРОМ!**



**ЦЕЛЫХ 4 ДИСТРИБУТИВА!**

# LINUX FORMAT

**Главное в мире Linux**

**63** страницы  
учебников  
и статей!

- » Зарядим планшет Ubuntu
- » Вещаем с Pi Zero
- » Заходим в терминал Linux



Все о Raspberry Pi 3 и эксклюзивные интервью с командой

Май 2016 № 5 (209)

# КРЕПЧАЕМ!

## Встроим Linux, и устройства взлетят!

- » Потокное вещание с брелка
- » Ультра-быстрая загрузка
- » Пробуем машины без шофера
- » Управляем дронами



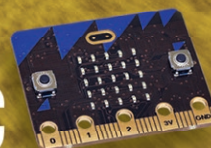
### Rust — не гниль

« С точки зрения C, избежать неопределенностей — задача программиста »

Джим Блэнди — о том, что в C все плохо с. 38

### BBC Micro:bit

Помогите своим детям программировать на новом гаджете BBC!



Сравнение

### Спасение утопающих

» Тукс в шаговой доступности и с лучшими инструментами наготове



Octave 4.0

### Обсчитаем Большие Данные

» Долой дорогущий Matlab — есть свободная альтернатива

ПОДПИСНЫЕ ИНДЕКСЫ В КАТАЛОГАХ  
Агентство «Роспечать» — 36343  
«Почта России» — 11932, «Пресса России» — 90959

Linux center  
www.linuxcenter.ru



# iTeleRadio

ИНТЕРАКТИВНОЕ РАДИО&ТЕЛЕВИДЕНИЕ

iteleradio.ru





## Что мы делаем

- » Мы поддерживаем открытое сообщество, предоставляя источник информации и площадку для обмена мнениями.
- » Мы помогаем всем читателям получить от Linux максимум пользы, публикуя статьи в разделе «Учебники»: здесь каждый найдет что-то по своему вкусу.
- » Мы выпускаем весь код, появляющийся на страницах раздела «Учебники», по лицензии GNU GPLv3.
- » Мы стремимся предоставлять точные, актуальные и непредвзятые сведения обо всем, что касается Linux и свободного ПО.

## Кто мы

Вот о чем мы спросили наших экспертов: с каким самым лучшим применением встраиваемых Linux вы сталкивались?



### Джонни Бидвелл

На новых эсминцах ВМС США класса Zumwalt стоят 16 блейд-серверов, где будет работать проприетарная ОС на базе Linux, под названием LynxOS. В один прекрасный день она будет контролировать рельсотроны и лазеры. Среди других ободряющих новостей — весь потенциал ядерного сдерживания Великобритании по-прежнему сидит на Windows XP.



### Нейл Ботвик

Linux сейчас в моде, так что трудно выбрать единственный удачный случай. Но портирование Linux в линейку процессоров Cyberduple и их встраивание в T-800 было/будет значительным шагом. Хотя попытки загнать их под GPL дождутся решения суда разве что в Судный день.



### Ник Пирс

Для меня это, конечно, Pi MusicBox. Ее несложно настроить, и она предназначена для запуска без монитора и позволяет невероятно легко заделать Pi Zero дешевым плеером потоковой музыки. Это здорово, если рассудить, что какая-нибудь система Sonos ободрала бы вас на сотни фунтов.



### Лес Паундер

Я помню, как в 2004 г. меня восхитила линейка устройств Sharp Zaurus. Там не было ни Wi-Fi, ни Bluetooth, оперативная память — всего 4 Гб, но эти устройства произвели революцию. По нынешним меркам это карманный калькулятор, но тогда это было блестящее применение Linux.



### Маянк Шарма

Этой чести удостоится Motorola A780. Как ни странно, в ней есть стандартные утилиты GNU, типа *glibc* и *fileutils*, и это был первый раз, когда я смог добыть оболочку root в телефоне. С нежностью вспоминаю о Сети в форм-факторе раскладушки и повторении подвигов Харальда Вельте [Harald Welte] в телефоне.



### Валентин Синицын

Если понимать встраивание в широком смысле слова, то, конечно, я голосую за Ubuntu в Windows 10. То, что когда-то казалось безумной фантазией, теперь реальность. Интересно, apt-get install mssql-server работает?



## Стандарт индустрии

» На страницах этого номера вы найдете несколько примеров использования Linux во встроженных системах. Диапазон подобных применений чрезвычайно широк — от «тяжелых» систем хранения данных до популярных видеокамер GoPro. К примеру, я не видел ни одной «читалки» для электронных книг, использующей другую ОС.

Причина популярности понятна. С одной стороны, ядро портировано на все современные процессорные архитектуры. Если появляется новая, то первой системой для нее становится именно Linux. Пример — семейство микропроцессоров «Эльбрус». С другой стороны, прошивки абсолютного большинства устройств являются по сути «системой одной задачи», на которой и должен сосредоточиться автор. Всё остальное — управление памятью и процессами, работа с сетью и файловыми системами — уже готово. О богатейшем выборе средств разработки и отладки и говорить нечего. Осмелюсь утверждать, что именно в области встроженных систем Linux доминирует абсолютно. Кроме тех особых случаев, где требуется «жесткое реальное время» (с «мягким» модифицированное ядро справляется без проблем). Но если посчитать в «штуках», то самолетов и атомных электростанций в мире значительно меньше, чем «читалок» и точек доступа. И что бы ни говорили маркетологи одной известной фирмы с Северо-Запада США, Linux сегодня и есть та самая «стандартная для индустрии ОС».

### Кирилл Степанов

Главный редактор

» [info@linuxformat.ru](mailto:info@linuxformat.ru)

## Как с нами связаться

Письма для публикации: [letters@linuxformat.ru](mailto:letters@linuxformat.ru)

Подписка и предыдущие номера: [subscribe@linuxformat.ru](mailto:subscribe@linuxformat.ru)

Техническая поддержка: [answers@linuxformat.ru](mailto:answers@linuxformat.ru)

Общие вопросы: [info@linuxformat.ru](mailto:info@linuxformat.ru)

Проблемы с дисками: [disks@linuxformat.ru](mailto:disks@linuxformat.ru)

Вопросы распространения: [sales@linuxformat.ru](mailto:sales@linuxformat.ru)

Сайт: [www.linuxformat.ru](http://www.linuxformat.ru), группа «ВКонтакте»: [vk.com/linuxform](https://vk.com/linuxform)

» Адрес редакции: Россия, Санкт-Петербург, пр. Медиков, 5, корп. 7

» Телефон редакции: (812) 309-0686. Дополнительная информация на с. 112



# ГНУ/Линуксцентр

*Ваш поставщик свободного программного  
и аппаратного обеспечения*

**Комплекты  
легализации СПО**

**Дистрибутивы  
GNU/Linux и СПО**  
на DVD и загрузочных  
флэшках

**Дистрибутивы  
GNU/Linux  
и СПО**  
с сертификатами  
ФСТЭК, ФСБ  
и Минобороны

**Межсетевые  
экраны**  
с сертификатами  
ФСТЭК, ФСБ  
и Минобороны



**Свободное  
аппаратное  
обеспечение**  
Arduino, oLinuxino,  
Cubieboard, Raspberry Pi,  
Intel Edison, Digilent,  
3D-принтеры  
и робототехнические  
конструкторы

**Аппаратное  
обеспечение  
с прошивками  
на базе СПО**

**Обучающая  
литература**

**Атрибутика**

## Фирменный магазин и сервис-центр

*Санкт-Петербург, пр. Медиков, 5, корп. 7*

*+7 812 309 06 86 | [www.linuxcenter.ru](http://www.linuxcenter.ru)*





# Samsung Z3

## Лучший Tizen-смартфон

Samsung Z3 — смартфон под управлением сертифицированной ОС Tizen для предприятий, предъявляющих повышенные требования к безопасности. Смартфон является составной частью решения SAFE — с сертифицированными MDM и VPN.

- Яркий экран HD sAMOLED Display 5" 293 PPI, насыщенность 100%, контраст 100 000:1
- Мощная батарея емкостью 2600 мА·ч. Время работы — 4–5 дней, до 13 часов проигрывания видео, до 33 часов в режиме ожидания на 10% заряда)
- Основная камера 8 Мпикс. Нулевая задержка затвора, f/2.2
- Фронтальная камера 5 Мпикс и многое другое...

Уже доступен для предзаказа!



[www.linuxcenter.ru/tizen/samsung-z3](http://www.linuxcenter.ru/tizen/samsung-z3)

## Предзаказ Samsung Z3

КОМПАНИЯ «ГНУ/ЛИНУКСЦЕНТР» является официальным партнером Samsung по продвижению смартфона Samsung Z3. На сайте [www.linuxcenter.ru/tizen/samsung-z3](http://www.linuxcenter.ru/tizen/samsung-z3) вы уже сейчас можете оформить заявку на приобретение Samsung Z3 из первой официальной поставки в Россию с гарантийной поддержкой, сертификацией и государственной таможенной декларацией.

Оформите заявку на приобретение Samsung Z3 прямо сейчас и получите:

- Скидку 15% при оплате заказа до 30 июня 2016 года.
  - Возможность стать участником конкурса проектов на базе ОС Tizen и получить демо-версию смартфона для реализации проектной идеи.
- Для оформления заявки необходимо:
1. Заполнить форму на сайте [www.linuxcenter.ru](http://www.linuxcenter.ru) с указанием контактных данных (фамилия, имя, телефон, адрес электронной почты).
  2. После размещения заявки на указанный адрес электронной почты будет направлено подтверждение о регистрации.

## Конкурс Tizen.Ru

КОМПАНИЯ СМАРТ КАПИТАЛ — основатель IoT Smart Center при поддержке ИТ-кластера фонда Сколково, и Ассоциация Тайзен.РУ совместно с партнерами объявляют конкурс на лучшие разработки на платформе ОС Tizen.

Команды-финалисты и победители конкурса получают призы от компании, а также возможность принять участие в 10-м «большом» Акселераторе ФРИИ и получить инвестиции на дальнейшее развитие своего бизнеса от институтов развития — фонда Сколково и ФРИИ — в размере от 1,5 до 15 млн руб.

Для участия в конкурсе необходимо зарегистрироваться на сайте [www.linuxcenter.ru/tizen/samsung-z3](http://www.linuxcenter.ru/tizen/samsung-z3)

## Конкурс проектов

С 1 ПО 30 ИЮНЯ 2016 ГОДА компания Samsung совместно с «ГНУ/Линуксцентром» проводит конкурс проектов на базе ОС Tizen. Для участия в конкурсе необходимо:

1. Написать статью о своем проекте, включающую информацию о том, какое место в нем занимает аппаратное устройство Samsung Z3 и ОС Tizen.
2. До 30 июня заполнить анкету разработчика на сайте [www.linuxcenter.ru/tizen/samsung-z3](http://www.linuxcenter.ru/tizen/samsung-z3) или прислать по электронной почте [konkurs@linuxcenter.ru](mailto:konkurs@linuxcenter.ru) заявку, содержащую:
  - Ссылку на вашу статью.
  - Информацию о составе вашей команды или наименование работодателя, если вы участвуете от имени группы или работодателя соответственно.

Самые интересные по мнению организаторов конкурса проекты получают демо-версию Samsung Z3 для проведения исследовательских работ по интеграции идей в ОС Tizen. **Главный приз конкурса** — грант на старт бизнеса до 2 млн руб. от Фонда содействия инновациям.

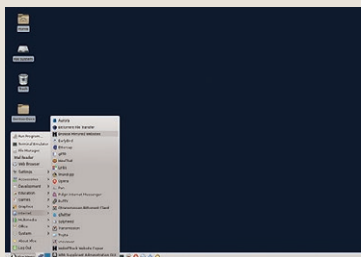
# Содержание

«Если Microsoft возьмется за приложения для Linux, то я победил.» Линус Торвальдс

## Обзоры

### exGENT ..... 12

Очередной дистрибутив на базе Gentoo не желает, чтобы его оценивали чисто из любви к его родителю, но готовит козыри...



› Задуман для новичков, замахнувшихся на Gentoo. А что вышло?

### Rosa Fresh KDE R7 ..... 13

Дистрибутив для энтузиастов и опытных пользователей идеально подойдет тем, кто в восторге от настраиваемости KDE и достаточно хорошо освоил Linux.

### Nelum OS ..... 14

Что выйдет, если дистрибутив, производный от Ubuntu, минималистский дистрибутив и самообновляемый дистрибутив на базе Debian сообразят на троих? Да вот это!

### Toshiba Chromebook 2 ..... 15

Toshiba стала первой крупной компанией, изготовившей хромбук высокого качества — и развивает свой успех, повышая планку спецификаций на оборудование-2016.



› Жаждали процессора Intel Core i3 и экрана 1080p? Получайте.

### Zotac NEN Steam PC ..... 16

Вам были обещаны Steam-машины, и наконец-то они появились. Попробуем на ней поиграть и посмотреть, на что способна Steam OS.

### XSOM 2 ..... 17

А мы предупреждали: угроза чужакам здесь реальная. Бежать в горы! Опять!

## ЗАПУСК!

Настала пора встраивать Linux во всё подряд, от квадрокоптеров до автомобилей. Как — см. с. 30



## Сравнение:

Спасательные дистрибутивы с. 24



## Интервью



« Это танец с саблями на льду. Rust перекидывает мост через эту пропасть. »

Джим Блэнди — о нестабильности традиционных языков с. 38



# На вашем бесплатном DVD

**Крутой медиа-центр** **Построим!**

**OpenELEC 6.0** для ПК и Pi

- Смотрите и упорядочивайте все свои фильмы
- Встроенные телевид и видеозапись
- Наслаждайтесь музыкой, фото и прочим!

**Ubuntu** для планшетов

**Rescatux 0.40b5** Лучший дистрибутив для Linux

**Debian 8.3.0** Многозадачный дистрибутив

**LINUX LIVE ДИСК: ГОТОВ К РАБОТЕ** ВСЕ НЕОБХОДИМОЕ ДЛЯ СТАРТА В LINUX

**OpenELEC для ПК и Pi, Debian 8.3 LXF 64-битный, Rescatux 32- и 64-битный**

» Каждый месяц — только лучшие дистрибутивы

**ПЛЮС:** Ubuntu для планшетов, и не только! **с. 106**

Побалуйте себя и любимых подпиской на LXF!



Доступно в AppStore!



[www.linuxformat.ru/subscribe](http://www.linuxformat.ru/subscribe)

## Пользователям Raspberry Pi



### Raspberry Pi стукнуло 4! ..... 88

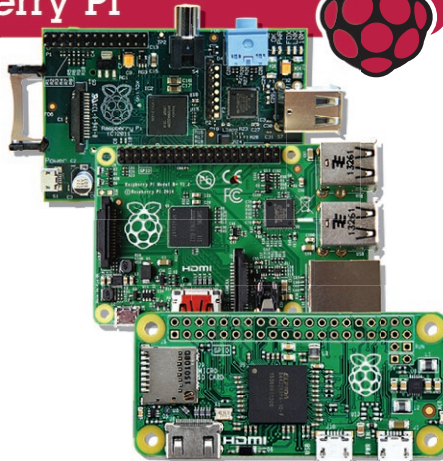
Итожим то, что Pi прожил за эти годы, и разбираемся, как он сумел стать рекордсменом по продажам плат микро-ПК и покорить мир!

### Комплект для экрана от Капо .... 93

Переносный экран, который дружит с Raspberry Pi, придется собрать самим, но ведь это истинное удовольствие.

### Списки в Scratch ..... 94

С помощью Scratch можно изучать концепции языков программирования, а затем перенести эти навыки на Python. Сейчас настала очередь списков.



## Ищите в номере

### Разработка для Tizen ..... 20

OS Tizen располагает полноценным инструментарием для разработчиков — присоединяйтесь.

### PHP на Android ..... 22

Вы убедитесь, что мобильное устройство отлично справляется с ролью web-сервера. И вы можете программировать прямо в дороге, пользуясь своим планшетом.

### BBC micro:bit ..... 42

Эта штука научит ваших детей программировать! Она от BBC, работает с Pi и в образовательных учреждениях Великобритании бесплатна. Получить бы такую и нашей школе...

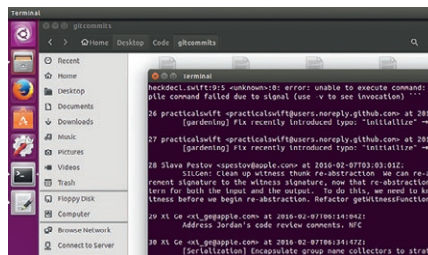
## Академия кодига

### Проекты на Swift ..... 80

Swift пока не полностью реализован на Linux, но это не повод уклоняться от его изучения! Бравые умельцы уже придумали обходные пути, так что не упускайте возможности поработать на перспективу.

### Сайт на MongoDB ..... 84

Завершаем серию про замечательную базу данных NoSQL построенную на ней сайта с блогем. А себе в помощь привлечем змеиные «бутылки».



## Учебники

### Основы Linux

**Ключевые утилиты ..... 52**  
Выводим тексты, работаем с числами и убиваем процессы, и всё это — не без оглядки на Bash.

**Основы терминала Привет, терминал! ..... 56**

Стартуем новую серию о том, как использовать терминал, чтобы творить всякие чудеса, да и время экономить.

**Встраиваемое оборудование Планшеты с Ubuntu ..... 58**

Интересно, будет ли Ubuntu работать на недорогом планшете x86? А то нет! Мы покажем, как его заставить.

**Gentoo Вникаем в Portage ..... 62**

Отметим обвинения, что мы пишем учебники только для Ubuntu — нате вам Gentoo.

**Математика! Octave 4.0 ..... 64**

Разберемся, как последний релиз свободного Octave побивает недешевый Matlab.

**OpenELEC Вещает брелок с Pi Zero ..... 68**

Пробуем построить лучший медиа-центр всех времен на микрокомпьютере.

**MySQL Fabric Работа с шардами ..... 72**

В погоне за высокой доступностью данных, эти самые шарды разделяем и перемещаем.

**CRUI Воссоздание системы ..... 76**

Перенос работающего приложения с сервера на сервер становится реальностью.

## Постоянные рубрики

### Новости ..... 6

В полку Raspberry Pi прибыло, сделан открытый ПК размером с монету, Samsung осваивает 10 нм, Intel вставляет компьютер в телевизор и выпускает новые чипы, LibreOffice работает на оборону, а Linux разъезжает на авто.

### Вести мобильных ОС .... 18

Wi-Fi уперся в ограничение, Google замахнулся на 3,5 ГГц и прослыл душителем, а злобный троянец не унимается.

### Сравнение ..... 24

Героический пингвин спешит на помощь! И с ним — славная когорта спасателей: Finnix, Rescatux, SystemRescueCd, Trinity Rescue Kit, UltimateBootCD.

### Интервью LXF ..... 38

Джим Блэнди ждет не дожидается появления хороших учителей: уж тогда-то язык Rust наконец обретет популярность.

### Рубрика сисадмина .... 46

Пока м-ра Джелиона Брауна не заменили искусственным интеллектом, он спешит порадоваться открытости AMD и выпасает многочисленные контейнеры Rancher.

### Ответы ..... 96

**ВАШИ ПРОБЛЕМЫ РЕШЕНЫ!**  
Нейл Ботвик — про первую установку, проблемы с Midnight Commander, игры в Linux с Wine, настройку твердотельного накопителя, перезагрузку роутера на удаленном сайте, Linux в Pine A64+.

### HotPicks ..... 100

Отдайте горяченького! Лучшие в мире новинки свободного ПО: ABCDE, Duckhunt-JS, FET, KeePassX, KNemo, Moksha, MyPaint, Opus, Retext, WebcamStudio, Widelands.

### Диск Linux Format ..... 106

Содержимое двустороннего DVD этого месяца.

### Пропустили номер? ... 108

Не сдавайтесь хакерам без боя: изучите их подлые трюки, которые описаны в LXF208.

### Через месяц ..... 112

Берем KDE 5 Plasma за основу и создаем наилучший рабочий стол Linux для всех вариантов оборудования.



**В ЭТОМ НОМЕРЕ:** Третий — не лишний » Не только Pi » Наномощь » ПК как ТВ-приставка » Старые кони вновь в борозде » Что готовит день грядущий » LibreOffice в армии » Пингвин под капотом

## НОВОСТИ PI

# Прибыл Raspberry Pi 3!

И он считается готовым для Интернета Вещей (IoT).



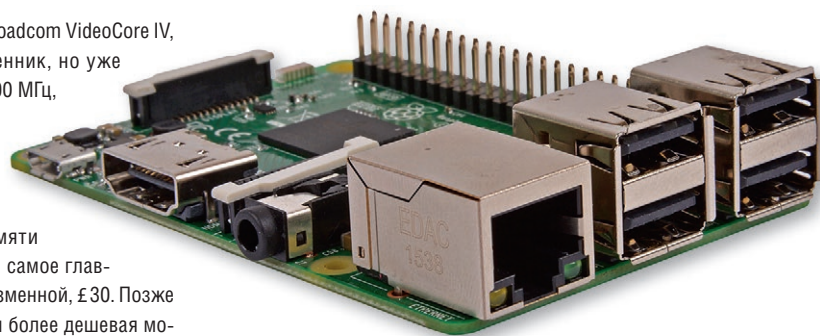
» Рубрику готовил  
**АНДРЕЙ  
ГОНДАРЕНКОВ**

Через четыре года после выпуска первого Pi, компании Raspberry Pi Trading и element14 совместно анонсировали новое и захватывающее приращение семейства Pi. Новая модель щеголяет мощным 64-битным ARM Cortex-A53 с тактовой частотой процессора QuadCore на частоте 1,2 ГГц, и это самый быстрый из всех Pi. Мало того, Pi 3 позиционируется как IoT-готовое устройство, с беспроводным адаптером 802.11b/g/n на борту и Bluetooth 4.1. Да и без всякого IoT потенциальных пользователей порадует, что беспроводная связь больше не требует тратить порт USB.

Но это еще не все новинки. Форм-фактор кредитной карты сохраняется, и разводка платы в основном не менялась после Pi 2, за исключением светодиодов, которые переместили к югу от SD-карты, чтобы уместить антенну. Зато слот microSD больше не использует пружинный механизм фиксации, ограничиваясь трением — одним шансом меньше что-то сломать или выбить. Pi 3 использует прежнюю (отныне

открытую) графику Broadcom VideoCore IV, как и его предшественник, но уже на тактовой частоте 400 МГц, а 3D-ядро работает на 300 МГц; то и другое — против бывших 250 МГц. Плюс 1 Гб оперативной памяти на борту и, возможно, самое главное: цена остается неизменной, £30. Позже в этом году ожидается более дешевая модель A Pi 3, без порта Ethernet, а также новый модуль Compute.

Хотя процессор 64-битный, Raspbian пока держится в 32-битном пространстве пользователей, ради совместимости разных моделей. В будущем это подлежит пересмотру: появятся приложения, которые выиграют от AArch64, но по большей части A53 является духовным преемником A7, хотя и только в 32-битном режиме. Кроме того, увеличение тактовой частоты на 33% и улучшения в BCM2387 означают, что при некоторых сценариях нагрузки скорость повысится. Согласно рекламе,



» Новая модель получила 64-битный четырехъядерный процессор ARM Cortex-A53 с тактовой частотой 1,2 ГГц, став самым быстрым из всех Pi.

Pi 2 был раз в шесть быстрее своего предшественника (B+), а на сей раз Фонд Pi сообщает, что новая плата вдесятеро быстрее B+, следовательно, где-то на 66% быстрее, чем Pi 2. Беглый сравнительный тест Whetstone был близок к этому, показав прирост скорости на 65% по сравнению с 900-МГц Pi 2. Новый процессор способен варьировать частоту, и при простое системы будет замедляться до 600 МГц. При такой частоте потребляемая мощность падает до жалких 2,5 Вт, на радость тем, кто применяет Pi в условиях ограничений на питание.

Беспроводная связь и Bluetooth обеспечиваются через комбинированное устройство связи BCM43438, которое поддерживает и классический Bluetooth, и режим Bluetooth с низким энергопотреблением. Ко времени вашего чтения драйверы для них добавят в Raspbian, так что все должно работать из коробки.

С момента выхода было отгружено восемь миллионов Pi, и Pi стал самым продаваемым ПК в Великобритании. При таких темпах этот показатель посягает на рекорд Commodore 64 — машины с наивысшим объемом продаж всех времен, более 17 миллионов (точно никто не считал).

» Форм-фактор кредитной карточки сохранен, но Pi 3 обзавелся беспроводным адаптером 802.11b/g/n и Bluetooth 4.1.





МИНИАТЮРИЗАЦИЯ

# Не Raspberry Pi единым

VoCore выпустил крохотный открытый Linux-компьютер за \$20.

**И**щите для своего проекта маленький и недорогой Linux-компьютер? Устроит ли вас девайс размером с монету по цене \$20? Это VoCore — компьютер с полностью открытыми аппаратной начинкой и ПО, идеально подходящий для устройств IoT или создания маршрутизатора с индивидуальной настройкой.

И не думайте, что за \$20 вы получите убогий набор-конструктор «Сделай сам». VoCore оснащен

CPU MIPS 360 МГц, 32 МБ ОЗУ, 8-МБ флэш-памятью, и оборудован разъемами для вывода звука, USB 2.0, Ethernet и даже Wi-Fi. Загрузка операционной системы OpenWrt с ядром Linux 3.10.44 занимает около 30 сек. Электропитание возможно от любого источника в диапазоне 3,2–6 В; опционально предлагается внешняя док-станция, позволяющая соединить крошку VoCore с другой платой, предоставляющей разъем для наушников, порт USB 2.0,

RJ45 и слот для карт micro SD. В магазине VoCore (<http://vocore.io/store>) цена одного VoCore составляет \$20, VoCore с док-станцией — \$45. Еще за \$40 предлагается модуль из камеры 1080p в комплекте со встроенным микрофоном, для подключения которого к VoCore не требуется никаких драйверов.

\$20 за VoCore слишком дорого? Тогда вам остается собственноручно собирать свой бесплатный (ну, кроме стоимости комплектующих) компьютер.

НАНОТЕХНОЛОГИИ

# В памяти моей скрыта мощь...

Samsung представила изготовленные по 10-нм технологии чипы DDR4.

**S**amsung начала массовое производство чипов памяти DDR4 на основе 10-нм процесса. Новая DRAM от южнокорейской компании значительно быстрее и энергоэффективнее, чем память DDR4, изготовленная с использованием процесса 20 нм.

DRAM, где временно сохраняются обрабатываемые данные, является критически важным элементом любой компьютерной системы. Новые чипы способны работать на максимальной частоте 3200 МГц, тогда как аналогичный показатель у DDR4 предыдущего поколения — только 2400 МГц.

Подобно ситуации с CPU и GPU, уменьшение чипов DRAM также повышает их энергоэффективность: новые модули будут потреблять на 10–20 % энергии меньше, чем прежние 20-нм DDR4.

На серверах память DDR4 появилась в 2014 г., и, по мнению Samsung, ее более быстрая DRAM улучшит работу крупномасштабных серверных приложений. Обработка приложения в ОЗУ (In-memory) — популярный тренд: она выполняется быстрее, к тому же уменьшается объем данных, перемещаемых между накопителями и DRAM. Новая DDR4 ускорит In-memory обработку баз данных.

10-нм чипы памяти DDR4 Samsung будут доступны уже в этом году, емкостью от 4 Гб для ноутбуков до 128 Гб для серверов предприятий. Также в этом году компания представит чипы 10-нм DRAM DDR4 для мобильных устройств, что, несомненно, укрепит лидирующие позиции Samsung на рынке ультра-HD смартфонов (смартфоны с DDR4 малой мощности, LPDDR4, уже существуют, так что речь идет, видимо, о более быстрой DDR4). Улучшения нового технологического процесса позволяют Samsung сделать обычную DDR4 такой же энергоэффективной, как и LPDDR4.

ОТЧЕТЫ И ИССЛЕДОВАНИЯ

# Future of Open Source Survey 2016

Будущее открытого ПО изобилует возможностями.

**B**lack Duck и North Bridge представили Future of Open Source Survey 2016 — 10-й ежегодный обзор, исследующий тенденции в мире открытого кода и составленный на основе анализа ответов от почти 3400 профессионалов. Отмечается повышенное внимание, уделяемое вопросам безопасности в ПО с открытым исходным кодом, внедрение новых технологий по мере их появления, а также рост популярности контейнеров: 76% респондентов заявили, что у их компаний есть планы использования контейнеров, а 59% респондентов уже применяют контейнеры во множестве развертываний, от разработки и тестирования до внутренней и внешней производственных сред.

Еще несколько положений Survey 2016:

- » Открытый код является существенным элементом стратегии развития более чем 65% респондентов, полагающихся на открытый код в целях ускорения разработки.
- » Более 55% респондентов предпочитают в своих производственных средах использовать СПО.
- » Финансирование открытых проектов частными лицами за 5 лет возросло почти в 4 раза.
- » У 50% компаний нет формальной политики отбора и санкционирования открытого кода.
- » 47% компаний не имеют формальных процессов отслеживания открытого кода, что ограничивает возможности контроля.

- » У более чем трети компаний нет процесса идентификации и отслеживания известных уязвимостей открытого ПО, а также исправления вызванных ими проблем.
- » 67% респондентов сообщили, что активно поощряют разработчиков участвовать и вносить вклад в открытые проекты.
- » 65% компаний вносят свой вклад в открытые проекты.
- » Каждая третья компания выделила под открытые проекты ресурс с полной занятостью.
- » 59% респондентов участвуют в открытых проектах с целью получения конкурентного преимущества.

ВНЕДРЕНИЯ

# В Италии вояки экономят

Проект LibreDifesa — крупнейшее развертывание *LibreOffice* в Европе.

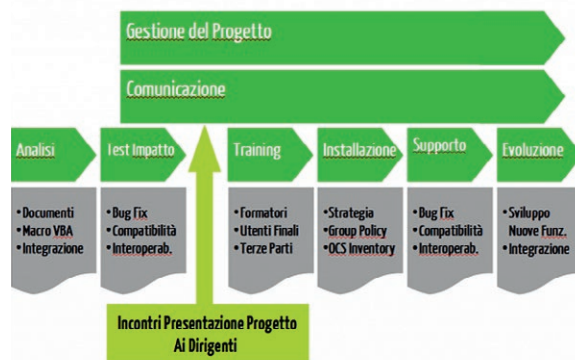
С помощью *LibreOffice* Минобороны Италии рассчитывает сэкономить в течение ближайших лет €26–29 млн: по проекту LibreDifesa все более чем 100 тыс. ПК военного ведомства перейдут на использование открытого офисного пакета. Учитывая крайние сроки текущих лицензий на *Microsoft Office*, к 2017 г. на *LibreOffice* будут переведены 75 тыс. (70%) пользователей, и к 2020 г. еще 25 тыс. Данное развертывание *LibreOffice* станет крупнейшим в Европе.

Проектная группа LibreDifesa включает представителей различных структур армии, флота и ВВС. Работа начиналась с оценки последствий, серии брифингов для персонала, обучения преподавателей, установщиков и представителей ИТ. Сейчас в тесном сотрудничестве с некоммерческой организацией Libreltalia для будущих пользователей создаются электронные обучающие курсы по *LibreOffice*, доступные на условиях copyleft-лицензии. Без особых проблем уже

выполнена миграция на *LibreOffice* 5 тыс. рабочих станций министерства.

Представитель Минобороны генерал Силео назвал основные причины миграции на *LibreOffice*:

- » в соответствии со ст. 68 Digital Administration Code Италии, ПО с открытым исходным кодом предпочтительнее проприетарного, если одинаково подходит и то, и другое;
- » принятие открытого формата ODF гарантирует функциональную совместимость и доступность документов даже после длительного периода времени, и делает государство независимым от поставщиков ПО;
- » принятие ODF за стандартный формат по всей стране — как в Великобритании — увеличит эффективность и снизит затраты;
- » переход на *LibreOffice* во всем ведомстве предоставит всем пользователям единую версию ПО, что в средне- и долгосрочной перспективе увеличит производительность;
- » экономия на лицензионных выплатах (поддержка Microsoft истекает через 5 лет).



» Схема проекта LibreDifesa по внедрению *LibreOffice* в итальянском Минобороны.

Минобороны Италии строго придерживается рекомендаций протокола миграции на *LibreOffice* от Document Foundation (TDF). Правильность этого подхода уже продемонстрировал проект LibreUmbria, успешно переместивший на *LibreOffice* тысячи рабочих станций в различных правительственных организациях итальянской провинции Умбрия.

LINUX ЗДЕСЬ, LINUX ТАМ...

# Мы едем, едем, едем!

Основной ОС в автомобилях XXI века будет GNU/Linux.

Взрывной рост AGL свидетельствует о спросе на технологии «подключенного автомобиля». Современный автомобиль — это комплексное компьютерное устройство, и ОС Linux задействована в информационно-развлекательных системах таких крупных производителей, как Toyota, Nissan и Jaguar Land Rover, а в ближайшей перспективе придет в модели Ford, Mazda, Mitsubishi и Subaru. Вскоре после анонса AGL Unified Code Base (UCB) — нового дистрибутива Linux, основанного на наработках AGL и двух других open-source проектов, Tizen и GENIVI Alliance — к Automotive Grade Linux (AGL), совместному открытому проекту по созданию базового программного стека на Linux для «подключенного» автомобиля, присоединились такие производители ПО, как Movimento, Oracle, Qualcomm, Texas Instruments, UIEvolution и VeriSilicon.

UCB — автомобильный Linux второго поколения, который был создан «с нуля»

и ориентирован на специализированные приложения (навигация, связь, безопасность, информационно-развлекательная функциональность). «Автомобильной промышленности необходимы стандартная открытая операционная система и платформа, позволяющие производителям и поставщикам быстро внедрять в автомобили смартфоноподобные возможности, — заверяет руководитель автомобильной группы Linux Foundation Дэн Коши [Dan Cauchy]. — Лучшие компоненты AGL, Tizen, GENIVI и связанного открытого кода интегрированы в едином дистрибутиве AGL Unified Code Base, что позволяет автопроизводителям использовать общую платформу для быстрых инноваций. Дистрибутив AGL UCB будет играть огромную роль в адаптации Linux-систем для всех функций транспортного средства».

AGL создан в январе 2016 г., и за прошедшее время к проекту присоединились 4 автомобильных компании и 10 новых



» Все усилия по внедрению Linux и открытого ПО на автомобили теперь объединены в рамках проекта Automotive Grade Linux.

компаний — производителей программных продуктов. Известный рекламный слоган 1965 г. от нефтяной корпорации Esso (ныне Exxon) гласил: “Put a tiger in your tank [Запусти тигра в бензобак!]” Происходящее сейчас можно, по аналогии, назвать “Put a penguin under your hood [Вставь пингвина под капот!]”: Linux уверенно занимает позицию главной автомобильной операционной системы XXI века.



СКАЖИ УЧЕБЕ

**ДА!**



**СЕРВЕРНЫЕ РЕШЕНИЯ  
LINUX  
SOLARIS  
ХРАНЕНИЕ ДАННЫХ  
JAVA  
ANDROID  
БЕЗОПАСНОСТЬ  
ЗАЩИТА ДАННЫХ**

Санкт-Петербург,  
ул. Яблочкова, 12, литер Ц  
(812) 611 1575  
[unixedu.ru](http://unixedu.ru)

**UnixEducationCenter**

В МИРЕ МИКРОКОМПЬЮТЕРОВ

# Compute Stick на Skylake

Online-магазины уже предлагают новые микрокомпьютеры от Intel с процессорами Core M3 и M5.

Компания Intel 29 апреля начала поставки трех моделей микрокомпьютеров Compute Stick с чипами Intel Core M3 и M5, анонсированными в январе в ходе выставке потребительской электроники CES-2015. Эти малыши превращают телевизор или дисплей с портом HDMI в полноценный ПК; надо всего лишь вставить Compute Stick в порт HDMI.

Цены на Compute Stick с чипами Skylake чуть выше, чем у прежних моделей, однако и их вычислительная мощность уже сопоставима с возможностями легких ноутбуков. Stick'и обладают высоким уровнем мобильности, хотя и ограничены по памяти, накопителям и портам, а пользователям нужны беспроводные клавиатура и мышь.

Начальная модель Intel Compute Stick STK2m364CC с процессором Core M3-6Y30 2,2 ГГц, без предустановленной ОС, предлагается по цене \$299. Модель STK2m3W64CC (тоже с процессором Core M3-6Y30), за \$395, поставляется с ОС Windows 10 (на [Amazon.com](http://Amazon.com) этот ПК можно заказать с 6 мая за \$359,95). Самый дорогой про-



Микрокомпьютер Intel Compute Stick с процессором M3.

дукт линейки, Compute Stick STK2mv64CC, с процессором Core M5-6Y57 и тактовой частотой 2,80 ГГц, стоит \$485 и также поставляется без ОС (на сайте BH Photo and Video предлагается с 7 мая за \$479). Эта модель поддерживает технологию Intel vPro.

Все три новые Compute Stick имеют накопители 64 Гб, 4 Гб памяти LPDDR3, интегрированную графику Intel HD Graphics 515, порты HDMI 1.4b, 802.11ac и Bluetooth 4.2, слот USB 3.0 и карт-ридер MicroSDXC. На модели без ОС пользователи смогут сами установить Windows или GNU/Linux (поддержка Chrome OS пока не заявлена).

Уже в продаже — представленные Intel на CES две другие модели Compute Stick, с процессорами Atom (Cherry Trail): STK1A32SC (\$129) и STK1AW32SC (\$139).

ПРОВЕРЕННЫЕ ВРЕМЕНЕМ

# Снова Pentium и Celeron

Появились Apollo Lake — новые чипы Intel на основе архитектуры Goldmont.

Процессоры Intel Core доминируют в мире ПК, но компания не собирается отказываться и от таких своих брендов, как Pentium и Celeron: во второй половине нынешнего года Intel выпустит новое поколение этих процессоров с кодовым именем Apollo Lake.

Чипы построены на новой архитектуре с низким энергопотреблением — Goldmont, и будут устанавливаться в недорогих ноутбуках и настольных ПК. На прошедшем в Шэньчжэне Форуме разработчиков Intel отмечено, что Apollo Lake обеспечит улучшенную производительность, а также большее время автономной работы. Новые процессоры Intel Atom также будут построены на Goldmont.

Apollo Lake приходит на замену нынешних чипов Pentium и Celeron N3000 Braswell.



Новые модели процессоров Intel Pentium и Celeron на архитектуре Goldmont.

Процессоры Braswell, построенные на старой архитектуре Airmont, применялись в недорогих ноутбуках, хромбуках и даже в некоторых планшетах. Во второй половине года Intel также начнет поставки новых процессоров Core на новой архитектуре Kaby Lake, по сравнению с которыми чипы Apollo Lake будут иметь меньше памяти, а также функций мультимедиа и I/O. LXF

Новости короткой строкой

Проект Phuctor предлагает публичный сервис анализа открытых RSA-ключей на потенциальную опасность компрометации закрытого ключа. Источник: [news.ycombinator.com](http://news.ycombinator.com)

В экспериментальных сборках движка V8 и Chrome для Chrome/Chromium 52 реализована поддержка пп. 6 и 7 спецификаций ECMAScript. Источник: [blog.chromium.org](http://blog.chromium.org)

Вышла бета-версия Devuan Jessie 1.0 — ветки дистрибутива Debian GNU/Linux 8.0 Jessie без *systemd*. Источник: [beta.devuan.org](http://beta.devuan.org)

За разработку GCC Ричард Столмен удостоен премии от Ассоциации вычислительной техники (ACM). Источник: [www.acm.org](http://www.acm.org)

В релизе GCC 6.1 по умолчанию в компиляторе C++ применяется стандарт C++14 и улучшенная поддержка C++17, платформ ARM, процессоров AMD Zen, Intel Skylake, IBM z13 и IBM POWER 9. Источник: [gcc.gnu.org](http://gcc.gnu.org)

В рамках реализации протокола Wayland в версии Firefox 46.0 для GNU/Linux перешли на GTK3+, но поддержка GTK2+ пока сохранена. Источник: [www.mozilla.org](http://www.mozilla.org)

Фонд СПО и проект GNU сочли Github и SourceForge неприемлемыми для их кода из-за элементов дискриминации по стране пользователя и работы на проприетарном JavaScript. Источник: [www.fsf.org](http://www.fsf.org)

Релиз свободного дистрибутива от сообщества OpenIndiana 2016.04, сменившего OpenSolaris, стал последним для 32-битных систем. Источник: [www.openindiana.org](http://www.openindiana.org)

Новая ОС DC/OS для дата-центров создана компанией Mesosphere под лицензией Apache 2.0 и поддерживается, в т.ч., Cisco, Equinix, HP, Microsoft, Canonical, Citrix, EMC и NetApp. Источник: [www.marketwired.com](http://www.marketwired.com)

Компания BitMover разработала свободный язык программирования L (Little), как альтернативу языку Tcl и его графическому инструментарию Tk. Источник: [sourceforge.net](http://sourceforge.net)

Лидером Debian выбран Мехди Доггуй [Mehdi Dogguy], известный адаптацией Debian для систем из TOP-500. Источник: [permalink.gmane.org](http://permalink.gmane.org)





Новинки программного и аппаратного обеспечения в описании наших экспертов



**АЛЕКСЕЙ ФЕДОРЧУК**  
Тэг <сарказм>  
по умолчанию,  
смайлики по вкусу.

## ubuntuBSD: опыт гибридации

Некогда в этой колонке (см. LXF118) говорилось о Debian GNU/kFreeBSD — системе с инфраструктурой Debian, переосаженной на ядро FreeBSD. Семь лет назад она не выглядела законченной, и развивалась с тех пор ни шатко, ни валко. Казалось, что идея гибридации FreeBSD и Linux'ового userspace близится к летальному исходу. Однако в марте этого года она неожиданно реанимировалась: Джон Боден [Jon Boden] выпустил первую бета-версию системы ubuntuBSD под девизом «Unix для гуманоидов [Unix for human beings]». Подобно kFreeBSD, на разработках которой в значительной мере основывается, она получила ядро FreeBSD, пользовательское окружение и пакетную инфраструктуру Ubuntu 15.10.

Система распространяется в виде ISO-образа чуть меньше 900 МБ, снабженного установщиком (клон «альтернативного» текстового инсталлятора Ubuntu). И, в отличие от kFreeBSD, позволяет «из коробки» получить не только базовую систему, но и рабочую среду Xfce с минимальным или полным (как у Xubuntu) набором приложений. Да и варианты чисто серверной системы не запрещены. Нехватка приложений после установки восполняется через собственный репозиторий.

Считать ubuntuBSD вполне готовой к употреблению пока нельзя, но недоработки активно исправляются — менее чем за полтора месяца вышло еще четыре бета-версии. И Джон начал борьбу за официальное членство в семье Ubuntu, в чем хотелось бы пожелать ему удачи.  
alv@posix.ru

## Сегодня мы рассматриваем:

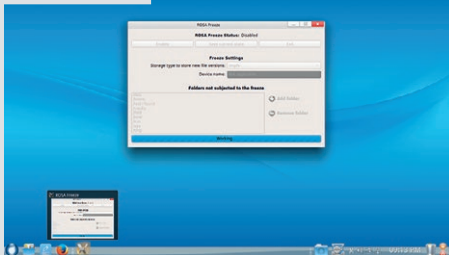
- exGENT** ..... 12  
Gentoo традиционно считается территорией «шибко умных», но было бы негуманно лишать остальное человечество его преимуществ, решили создатели exGENT.
- Rosa Fresh KDE R7** ..... 13  
Понятно, что KDE-дистрибутив делает упор на настраиваемость, а это требует опыта. Но рук не опускайте: Rosa легко адаптируется.
- Nelum OS** ..... 14  
Проект Nelum размахнулся аж на три весьма несходных между собой дистрибутива — объединяет их только оконный менеджер *Openbox*. Увы, вся троица не лишена мелких, но досадных недочетов.
- Toshiba Chromebook 2** ..... 15  
Этот хромбук превзошел своего предшественника более мощным процессором, качественным экраном и клавиатурой с подсветкой, а следовательно, и ценой. Но при таком качестве нельзя сказать, что цена зашкаливает.
- Zotac NEN Steam PC** ..... 16  
В Zotac сумели запихнуть лучшее игровое оборудование в крохотную упаковку. А Steam OS, предустановленный дистрибутив на базе Debian, способен на серьезные подвиги. Кабы не проблема с драйверами — Windows в пору бы забеспокоиться о конкуренции.



➤ Столь качественный экран на хромбуках встретишь не часто.

**XCOM 2** ..... 17  
Земля захвачена инопланетянами. Основная масса населения более или менее смирилась с господством пришельцев, но отважные бойцы Сопротивления не сдаются...

### Rosa KDE R7



➤ Узнаваемая синева KDE означает, что вы сможете настроить дистрибутив под любую цель.

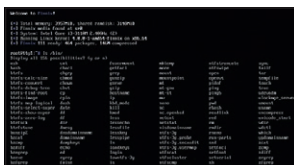
### XCOM 2



➤ После драки бластерами не машут? Нет! Дадим достойный отпор завоевателям!

## Сравнение: Спасательные диски с. 24

### Finnix



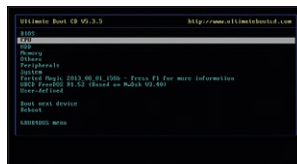
### Trinity Rescue Kit



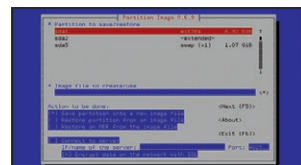
### Rescatux



### UltimateBootCD



### SystemRescueCd



Ситуация, когда волей-неволей придется выручать свою систему из беды, практически неизбежна: ее возникновение — всего лишь вопрос времени. Встретим же проблемы во всеоружии.

# exGENT

Шашанк Шарма испытывает еще один дистрибутив на базе Gentoo, решив не дать своей любви к предку дистрибутива затуманить свое суждение.

## Вкратце

» Live-дистрибутив на базе Gentoo с возможностью установки для пользователей с минимальным опытом. Надеется привлечь в свое лоно пользователей, чтобы попробовать установить Gentoo. В дистрибутиве масса приложений и легковесный рабочий стол Xfce. См. также: Gentoo, Sabayon.

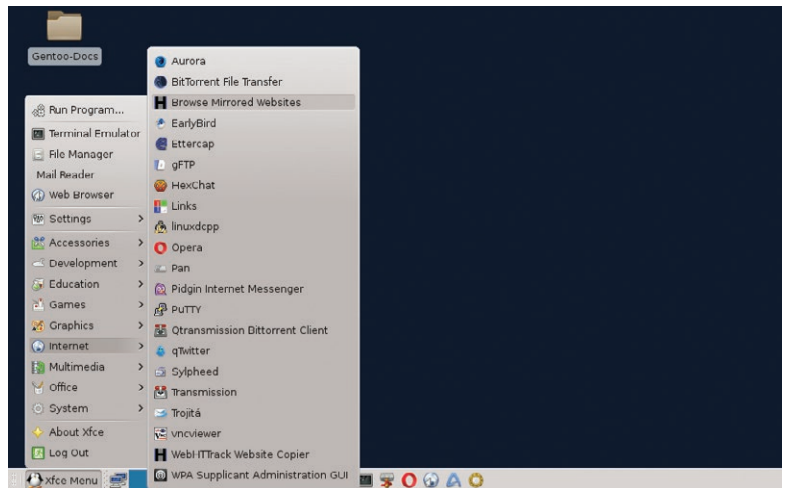
Все live-дистрибутивы на базе Gentoo стремятся преодолеть разрыв между знакомством с Linux начинающего пользователя и необходимым опытом, позволяющим играть с Gentoo. Особенным exGENT делает то, что он предназначен для привлечения новых пользователей на Gentoo, без полного отказа от обучения.

Доступный только для 64-разрядных машин, дистрибутив имеет широкий спектр приложений и программного обеспечения, способный удовлетворить даже самых требовательных пользователей. Дистрибутив можно запустить с live-диска или USB-накопителя, а несложный процесс установки позволяет даже начинающим пользователям получить рабочую систему Gentoo в считанные минуты.

Однако на нашей двуядерной тестовой машине с 2 ГБ ОЗУ на загрузку в живой среде потребовалось несколько минут — несмотря на заявление разработчика, что дистрибутив быстрее большинства подобных. После входа в систему дистрибутив преобразуется, становясь невероятно быстрым.

Многим пользователям, повидавшим последние версии Gnome или KDE, рабочий стол Xfce может показаться пресным, но преимущество в скорости легкого рабочего стола становится очевидным в течение первых нескольких минут работы дистрибутива.

Gentoo — один из наиболее документированных дистрибутивов в экосистеме Linux, но это не отражается на exGENT, который на своем сайте дает лишь ограниченные указания. Они включают инструкции по установке дистрибутива на жесткий диск или USB-накопитель. Есть также



» Стабильный и невероятно быстрый, дистрибутив exGENT обладает всеми качествами, которыми славен Gentoo, но хотелось бы видеть больше документации.

некоторые подробности о поддерживаемых картах Nvidia.

Как и его родитель, exGENT требует ручного создания и форматирования корневого раздела и раздела подкачки [swap]. Однако, чтобы облегчить процесс, дистрибутив поставляется с *GParted*, который может выделить место для дистрибутива и форматировать разделы.

## Gentoo — в массы

Сама установка запускается через скрипт оболочки, который берет на себя все остальное. Вам не нужно беспокоиться о выборе пакетов и настройке часового пояса или аппаратного обеспечения. Дистрибутив также может похвастаться отличным автоматическим распознаванием и настройкой оборудования. Все графические карты, адаптеры беспроводной связи, web-камеры и разное другое оборудование были безупречно обнаружены exGENT и настроены для использования без каких-либо проблем. Жаль, что скрипт не обеспечивает прогресс-индикатор или какой-либо вывод на экран, чтобы хотя бы видеть, что дистрибутив устанавливается. Последним шагом установки является настройка загрузчика. Для этого необходимо обратиться к инструкции по установке, если у вас уже установлен загрузчик или планируется установить по умолчанию унаследованный *Grub*.

Особое удовольствие в Gentoo доставляет его невероятно мощная система управления пакетами *Portage*. В попытке

облегчить пользователям освоение Gentoo, дистрибутив поставляется с графическим интерфейсом для этого, который называется *Porthole*. Хотя он и похож по внешнему виду и функциональности на многие аналоги, типа *Synaptic*, *Porthole* невероятно быстр и при необходимости поможет легко устанавливать дополнительное программное обеспечение. Новые пользователи также оценят утилиту *Settings Manager*, которая обеспечивает единую точку доступа к настраиваемым элементам.

По сравнению с другими дистрибутивами единственное, чего в exGENT не хватает — это документации, но в остальном он рекомендуется для всех, кто жаждал идеального дистрибутива со скользящим релизом. **LXF**

### Свойства навскидку

**Простая установка**  
Скрипт установки и обнаружение оборудования упрощает самые сложные элементы установки Gentoo.

**Скользящий релиз**  
Не надо беспокоиться о необходимости переустанавливать систему с каждым основным релизом.

### LINUX FORMAT Вердикт

**exGENT**  
**Разработчик:** Arne Exton  
**Сайт:** <http://exgent.exton.net>  
**Лицензия:** GPL и другие

<b>Функциональность</b>	10/10
<b>Быстродействие</b>	10/10
<b>Удобство в работе</b>	8/10
<b>Документация</b>	9/10

» При его скорости и настраиваемости, этот дистрибутив никоим образом не порочит величие Gentoo.

## Рейтинг 9/10



# Rosa Fresh KDE R7

Потерпев крах как алхимик XXI века, **Шашанк Шарма** одобряет попытку объединить KDE с передовым программным обеспечением.

## Вкратце

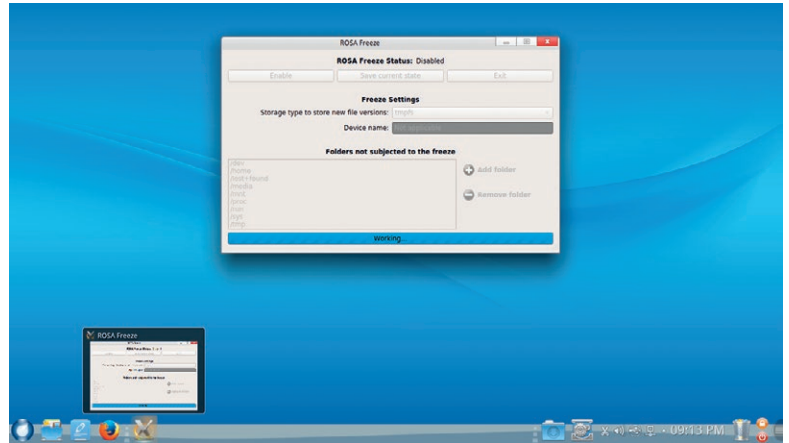
» Rosa Desktop Fresh нацелен на энтузиастов и опытных пользователей. Идеален для тех, кто очарован гибкостью KDE, имеет опыт работы с Linux и хочет поиграть с новейшими программными обеспечениями и технологиями. См. также: Mageia, openSUSE.

С первого своего Linux-дистрибутива Rosa Labs [ныне ООО «НТЦ ИТ РОСА», — прим. пер.] старается предоставить пользователям все возможности KDE. Помимо редакций для серверов и рабочих станций на предприятиях, они также производят линейку дистрибутивов Desktop Fresh, нацеленную и на любителей, и на опытных пользователей, и предлагающую современное программное обеспечение.

Разработанный при значительной помощи со стороны сообщества, дистрибутив подвергается серьезному тестированию, чтобы гарантировать, что любая его программа не приведет к краху системы. Линейка Desktop Fresh с годовым циклом выпуска включает дистрибутивы с разными окружениями рабочего стола, такими как KDE, Gnome, LXDE и т. д.

Основанный на платформе LTS ROSA 2014.1 с поддержкой до осени 2016 г., дистрибутив имеет своеобразный KDE — это отличительная черта всех выпусков Rosa. Он доступен для 32- и 64-разрядных машин; ISO-образ чуть меньше 2 ГБ и наполнен повседневными приложениями, играми и другим софтом, чтобы порадовать домашних пользователей и энтузиастов.

Последнее издание предлагает улучшенную поддержку графики AMD и Intel, что необходимо для поддержки Steam. Также включена поддержка Microsoft Hyper-V, что позволяет развертывать Rosa в публичных Azure Cloud. Как дистрибутив, который стремится к самосовершенствованию, последний релиз также включает новый rasterizer Adobe и Google, что значительно



» Свежий, передовой... эталон для новых дистрибутивов с KDE.

улучшает сглаживание шрифтов и создает более приятное впечатление от рабочего стола.

С первого релиза Rosa сосредоточена на предоставлении KDE собственной сборки. Помимо множества настроек, Rosa предлагает свое программное обеспечение, такое как *TimeFrame* и *StackFolder*, которые с тех пор нашли свое место в KDE.

## Свежесть гарантирована

Rosa разработал способ быстрого доступа пользователей к часто используемым папкам и файлам — *StackFolder*. Для начала пользователям надо только перетащить файлы или папки из файлового менеджера *Dolphin* на панель под названием *RocketBar* в нижней части рабочего стола и выбрать опцию *StackFolder*. После создания стека папки можно быстро просматривать подкаталоги или даже миниатюры фотографий и видеофайлов.

Система также включает программу запуска [launcher] *SimpleWelcome*, которая позволяет быстро найти нужное приложение из огромного числа установленных. Все приложения сгруппированы по функциям, таким как Интернет, офис, графика и т. д. Launcher в этой версии также более отзывчив: отказ от Plasma привел к снижению потребления памяти.

Наряду с *SimpleWelcome*, инструмент визуализации контента *TimeFrame* был заново переписан на QML. Он полагается на возможности упорядочивания метаданных *Perotuk* и дает возможность легко отслеживать активность в определенные даты. Он также поддерживает социальные

сети и предоставляет доступ к Facebook без запуска браузера.

Наряду с этими прекрасными новыми дополнениями дистрибутив также имеет несколько важных исправлений ошибок, таких как установка дистрибутива с помощью устройств с USB3 и улучшенную поддержку USB-клавиатур ThinkPad.

Благодаря своим богатым репозиториям программного обеспечения дистрибутив легко адаптируется для любых целей, будь то развлекательная станция для домашних пользователей или рабочая станция для разработчиков, и хотя дистрибутив предназначен для опытных пользователей, нет никаких причин не подойти также и новичкам в Linux. В самом деле, с его многочисленным сообществом это идеальная среда для новых пользователей, чтобы начать. **LXF**

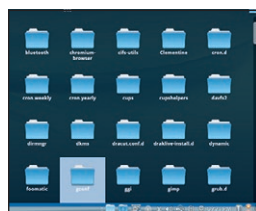


## Свойства навскидку



### TimeFrame

Собственный инструмент визуализации, пригодный для работы с изображениями и различными офисными документами.



### StackFolder

Удобное приложение для открытия файлов в ассоциированных приложениях и предпросмотра изображений и видеофайлов.

## LINUX FORMAT Вердикт

### Rosa Desktop Fresh KDE R7

Разработчик: Rosa Labs  
Сайт: [www.rosalab.com](http://www.rosalab.com)  
Лицензия: GPL и другие

Функциональность	9/10
Быстродействие	9/10
Удобство в работе	9/10
Документация	7/10

» Устанавливает очень высокую планку для дистрибутивов с KDE; почти идеальный релиз, цельный и превосходно настраиваемый.

## Рейтинг 9/10

# Nelum OS

Заходят в бар дистрибутивы: производный от Ubuntu, минималистский и обновляемый, на базе Debian... **Шашанк Шарма** счел это началом дурной шутки.

## Вкратце

» Хотя и основанный на все еще разрабатываемой Ubuntu 16.04, Nelum OS легкий и быстрый. Со своим набором приложений по умолчанию и возможностью воспроизводить мультимедиа из коробки, дистрибутив стремится обеспечить систему, пригодную для новичков и неопытных пользователей Linux. См. также: **Lubuntu, Puppy Linux.**

Для нового участника экосистемы Linux выбор Ubuntu в качестве своей основы — не трудное решение. Однако проект Nelum в невероятном сочетании амбиций и выдержки выпустил три дистрибутива. Рассмотренный дистрибутив Nelum OS базируется на Ubuntu 16.04 и ориентирован на новых пользователей Linux. Есть также версия на базе Debian Sid, которая называется Nelum Openbox и является роллинг-релизом с экспериментальным программным обеспечением, и поэтому предназначена для более опытных пользователей. Третий вариант — NelumBang — заявлен как минималистский дистрибутив. Вместо полнофункциональной среды рабочего стола все три варианта Nelum используют оконный менеджер *Openbox* в комплекте с предустановленным монитором системы *Conky*.

Nelum OS основана на разрабатываемой ветке Ubuntu 16.04. Хотя это позволяет предположить, что дистрибутив будет полон ошибок, конечный продукт достаточно стабилен. Некритичные заскоки возникают лишь изредка: например, при запуске в режиме Live Nelum не может правильно определить ваш часовой пояс, с чем у большинства других дистрибутивов нет никаких трудностей. Более того, нет способа исправить эту ошибку через графический интерфейс. Надо запустить терминал и выполнить команду `sudo date -s xx:xx` для установки правильного времени. К счастью, эта проблема исчезнет после установки Nelum на диск. Хотя дистрибутив легко нашел графические и беспроводные карты на наших тестовых машинах, он не смог правильно отобразить шрифты в некоторых



» В среде рабочего стола Nelum OS внизу есть док повседневных приложений, а также панель сверху.

приложениях, таких как Terminal. Кроме того, *VLC Media Player* работал без нареканий с мультимедийными клавиатурами, но сам рабочий стол Nelum не реагировал на специальные кнопки управления.

## Работа продолжается

Помимо этих проблем, Nelum OS пронизана некоторыми досадными неприятностями: например, первый экран установки направляет пользователей к информации о выпуске, но перейдя по ссылке, вы попадете на хостинг домена/продажу сайта. Вдобавок, хотя дистрибутив воспроизводит различные видео и MP3-файлы из коробки, звук пропадает без видимых причин.

В довершение всего наши попытки запуска любого из приложений *LibreOffice* выдали ошибку, а механизма отчетов об ошибках нет. У дистрибутива также нет списка рассылки, форумов, вики или достаточной документации. Тем не менее, меню *Openbox* дает ссылки на документацию Debian и Arch в дополнение к руководствам для *Conky* и *Openbox*.

В отличие от большинства производных Ubuntu, Nelum не использует *Центр приложений* Ubuntu. Вместо этого он поставляется с *Synaptic* и *App Grid*. Последний написан с нуля как альтернатива *Центру*, предоставляет рейтинги, обзоры и скриншоты программного обеспечения и является невероятно легким и быстрым.

Для пользователей, недовольных необходимостью навигации по меню приложений, дистрибутив имеет программу запуска [launcher] с полем поиска, чтобы помочь найти приложения. Также имеется инструмент поиска файлов *Catfish*, который может найти любой файл в системе за секунды.

В настоящее время Nelum OS не предлагает каких-либо веских причин для выбора его вместо других облегченных дистрибутивов, таких как *Puppy Linux*. Хотя в целом дистрибутив собран неплохо, он должен пройти долгий путь, прежде чем сможет даже надеяться создать вокруг себя сообщество пользователей. **LXF**

### Свойства навскидку

**Молниеносный**  
Имеет ресурсосберегающее ПО, такое как *App Grid* и *Catfish*, в дополнение к оконному менеджеру *Openbox*.

**Настраиваемый**  
Включает *Openbox* и *Conky*, которые предоставляют вам широчайшие возможности настройки дистрибутива.

## LINUX FORMAT Вердикт

**Nelum OS**

**Разработчик:** Nelum Project  
**Сайт:** <http://bit.ly/NelumOS>  
**Лицензия:** GPL и др.

<b>Функциональность</b>	6/10
<b>Быстродействие</b>	7/10
<b>Удобство в работе</b>	8/10
<b>Документация</b>	5/10

» Проект распылен на три редакции. Впереди долгая и напряженной работа.

## Рейтинг 6/10



# Toshiba CB 2

Смотреть видео в Full HD на Chromebook стало еще лучше, говорит **Кевин Ли**, радостно потирая руки.

## Спецификации

- » **ОС** Chrome OS
- » **Процессор** Intel Core i3-5015U, 2,1 ГГц (двухъядерный, кэш 3 МБ)
- » **Графика** Intel HD Graphics 5500
- » **Память** 4 ГБ DDR3L (1600 МГц)
- » **Экран** 13,3-дюймовый, 1920×1090
- » **Хранилище** 16 ГБ eMMC
- » **Порты** 1×USB3.0, 1×USB 2.0, HDMI, слот SD-карты, разъем для наушников и микрофона
- » **Связь** Intel Dual-Band Wireless-AC 7260, Bluetooth 4.0
- » **Габариты** 32×21,3×1,93 см
- » **Вес** 1,35 кг



» В Toshiba сделали ставку на более мощный процессор Core i3 и крутой дисплей Full HD.

**T**oshiba Chromebook 2 был одной из первых моделей, перевернувшей формат хромбуков своим классным 1080-пиксельным экраном, и вот он снова здесь, уже с процессором Broadwell Core i3 и новой клавиатурой с подсветкой. Увы, из-за добавления этих первоклассных компонентов подскочила и цена, поставив его в ряд ноутбуков премиум-класса на Chrome OS, таких как Google Chromebook Pixel и Dell Chromebook 13.

Обновленный Toshiba во многом напоминает своего предшественника из 2014 г., на базе Celeron. На самом деле, корпус почти не изменился, и размеры те же. Но если снаружи разницы не видно, наличие процессора Core i3 внутри означает, что Toshiba пришлось добавить вентилятор.

Клавиатура более чем удобная, с традиционной раскладкой, а подсветка клавиш — нечастая функция для устройств на Chrome OS. Перед клавиатурой расположен довольно большой трекпад с поддержкой точных щелчков. Проблем насчет производительности Chromebook 2 у нас практически не возникло. Ноутбук работал гладко, мы слушали Google Music при наличии кучи открытых вкладок в браузере и на фоне запущенного видео с YouTube, которое мы позабыли выключить.

Результат нового процессора Core i3 5015U в Octane составил 21554, а в Mozilla Kraken — 1,535 мс.

Два года назад Chromebook 2 стал первым ноутбуком Chrome OS с отличным экраном Full HD. Конечно, с тех пор 1080-пиксельные дисплеи стали встречаться чаще, но Toshiba по-прежнему остается одним из лидеров по этой части.

В этом дисплее Toshiba не только больше пикселей, но и сам он, в целом, еще качественнее. Вместо TN-панелей, используемых в большинстве хромбуков, в Toshiba предпочли TFT-экран, который передает яркость тонов и привносит глубину в темные оттенки.

## Новинка за разумные деньги

Естественно, платой за увеличение количества пикселей и мощности процессора является сокращение ресурса аккумулятора. Toshiba протянул всего шесть часов и две минуты непрерывного открывания вкладок, воспроизведения Google Music, часового просмотра YouTube и работы в текстовом редакторе.

Тест проводился при яркости экрана чуть ниже 50% и на громкости колонок около 20%. По сравнению с первым Toshiba Chromebook, дополнительное энергопотребление процессора Core i3 сократило жизнь батареи на 24 минуты.

В стандартном тесте с воспроизведением видео Chromebook 2 также сдался через 6 часов и 2 минуты. Тогда как у модели

на Celeron батарея продержалась 6 часов и 26 минут. Для сравнения, из Pixel нам удавалось выжать максимум 8 часов 22 минуты, а Dell Chromebook 13 дотянул аж до 12 часов.

В общем и целом, Toshiba Chromebook 2 превосходит модель 2014 г., под его «капотом» скрывается большая производительность, и его немало украшает подсветка клавиатуры. Выкладывать за него придется несколько побольше, но он тем не менее остается значительно доступнее рассчитанного на разработчиков Pixel и ориентированного на коммерческое применение Dell Chromebook 13. **LXF**

## LINUX FORMAT Вердикт

### Toshiba Chromebook 2

**Разработчик:** Toshiba  
**Сайт:** [www.toshiba.co.uk](http://www.toshiba.co.uk)  
**Цена:** £ 370 (Core i3, 4 ГБ)

<b>Функциональность</b>	8/10
<b>Производительность</b>	8/10
<b>Удобство в работе</b>	9/10
<b>Справданность цены</b>	8/10

» Обновленный, тонкий и легкий 13-дюймовый хромбук — лучший и полноценный для рядового пользователя.

**Рейтинг 8/10**

# Zotac NEN SN970

Новая Steam Machine от Zotac знает свое дело, но, как выяснил Дейв Джеймс, крошечную игровую систему подводят ее же инструменты.

## Спецификации

- » **ОС** Steam OS
- » **Процессор** Intel Core i5-6400T 2,2 ГГц (2,8 ГГц Turbo)
- » **ОЗУ** 8 ГБ, одноканальная, DDR3 @ 1600 МГц
- » **HDD** 1 ТБ
- » **Графический процессор** Nvidia GTX 960
- » **Связь** Wi-Fi 802.11ac/b/g/n, Bluetooth 4.0, 2xGigabit Ethernet
- » **Порты** 2xUSB 2.0, 2xUSB 3.0, 1xUSB-C, 4xHDMI 2.0
- » **Габариты** 210x203x62,2 мм
- » **Дополнительно** Steam Controller

**П**омните, как Steam Machines показали нам неплохой идеей? Помните, как Valve собирались спасти нас от игровой гегемонии, вернее, диктатуры злобного Microsoft? Да. И что-то пока не очень получается, правда? На самом деле, чем больше времени мы проводим, играя на специальных Steam Machines и их облегченных ОС Linux, тем с большим удовольствием потом возвращаемся на полноценную Linux-систему.

Та же история с последней Steam Machine от Zotac, со странным именем NEN. И это очень досадно, ведь Zotac создали одну из лучших игровых систем, которую нам довелось тестировать, сумев захватить лучшее игровое оборудование в крошечную упаковку.

Для начала, очень приятный черно-белый дизайн в стиле Stormtrooper. А внутри этой миниатюрной штуковины двухдверный процессор Skylake, 35-Вт Core i5-6400T.

Дополняет этот впечатляюще мощный процессор оперативная память DDR3 объемом 8 ГБ и частотой 1600 МГц; однако системная память NEN состоит из одного SODIMM, а значит, работает только в одноканальном режиме. Тем не менее, это значит, что ему еще есть куда развиваться. Но есть гораздо более насущные обновления, чтобы выжать максимум из Zotac NEN... до этого мы еще дойдем.

Отличительная черта NEN — мощь его графики. Внутри черно-белой вентилируемой коробочки с пластиковой отделкой прячется полноценный настольный GTX 960, с внушительной мощностью и пассивным охлаждением, способный работать



» Потрясающая игровая мини-система, еще бы игры и драйверы ей соответствовали...

в небольшом корпусе, не требуя громадных турбинных вентиляторов, чтобы поддерживать нормальную температуру.

Даже когда игра идет полным ходом, NEN остается удивительно тихим. А это главное для маленького игрового ПК, место которого рядом с широкоэкранным телевизором в гостиной, а не на столе вашего домашнего офиса. Вам же не нужна комнатная машина, ревущая в углу каждый раз при загрузке игры в XCOM 2, тогда как вы пытаетесь спасти свою задницу от очередного воинственного пришельца. Ну и вообще это отвлекает.

## ОС набирает ход

А если вы хотите ощутить всю производительную мощь оборудования, мы вам расскажем пару секретов о Steam OS. Предустановленный дистрибутив на базе Debian способен на многое. В сущности, он может сделать из этой игровой мини-системы доступную домашнюю консоль, с прямой загрузкой в Big Picture Mode в Steam. Благодаря входящему в комплект поставки Steam Controller, для взаимодействия не нужно даже мыши и клавиатуры. Конечно, контроллер выглядит немного слишком пластиковым, но вполне сойдет.

Из Big Picture Mode у вас должен быть доступ ко всем вашим играм Steam, чтобы вы могли играть во что угодно, с комфортом сидя на диване — такая роскошь доселе была доступна только нашим собратьям из консольного класса.

Только вот пока не получается. Steam OS по-прежнему не хватает собственных

игр; и хотя их число растет, все равно расстраивает, что в ходе поиска по библиотеке приходится пропускать большую часть игр, потому что они не пойдут на твоей новой игровой установке за £800.

Большая проблема в том, что даже если находится игра, совместимая со Steam OS, в Windows игровая производительность все равно гораздо выше. В крайних случаях, частота кадров сокращается вполтину, но в ходе наших тестов на производительность, NEN в целом резко замедлился при запуске драйверов Linux. Это разница между неоптимизированными играми и драйверами Linux и оптимизированными сборками и драйверами Windows. Пока Valve с этим не справится, Steam OS всегда будет медленнее. **LXF**



## Свойства навскидку



### Контроллер

Нам Steam Controller пришелся по душе, а вам, может, и нет (но тогда вы не правы).



### Тихий ход

Zotac Steam Machine работает практически беззвучно, что для гостиной — идеально.

## LINUX FORMAT Вердикт

### NEN Steam Machine SN970

Разработчик: Zotac  
Сайт: [www.zotac.com](http://www.zotac.com)  
Цена: £ 800

Функциональность	9/10
Производительность	7/10
Удобство использования	9/10
Оправданность цены	7/10

» Обещание Steam OS остается в силе, и эта система — прекрасный способ это воплотить, осталось только оптимизировать игры.

## Рейтинг 8/10



# XCOM 2

За прошедший год выяснилось, что с нелегальными пришельцами в игре-стратегии нужно действовать иначе, чем думал **Том Сениор**.

## Спецификации

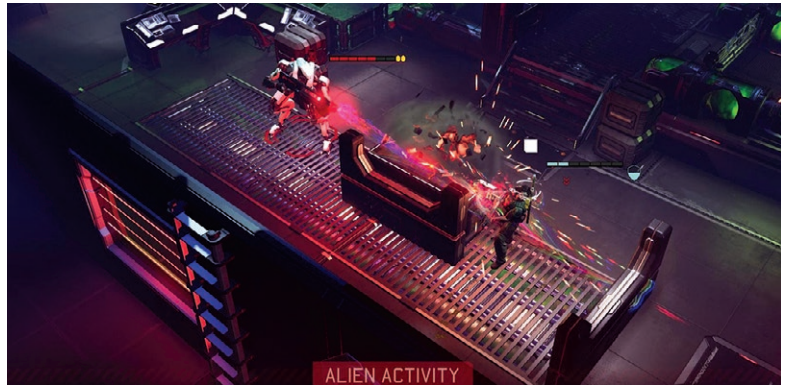
Рекомендуемые:

- » **ОС** Steam OS, Ubuntu 14.04 64-bit
- » **Процессор** Intel Core i7
- » **ОЗУ** 8 ГБ
- » **Графический процессор** Nvidia 960, 2 ГБ VRAM (Intel и AMD не поддерживаются)
- » **Диск** 45 ГБ

**В** XCOM 2 человечество обречено с самого начала. Предполагается, что вы не смогли сдержать вторжение инопланетян в *Enemy Unknown*. Теперь граждане Земли уютно живут под тоталитарным контролем инопланетян и набранной ими армии — Адвента [Advent]. Спротивление уцелело только в лице немногих отважных солдат, ученых и инженеров, которым удалось перепрофилировать огромный инопланетный корабль, Мстителем [Avenger]. Это ваш дом. Детальный поперечный вид корабля позволяет заглядывать в комнаты и затевать исследовательские и строительные проекты. Комнаты центрального блока можно расчистить под строительство новых объектов, а на мостике вы получаете доступ к Geoscape, карте мира, которая позволяет выбрать, где посадить свой космический корабль. Чтобы противостоять пришельцам, вы должны расширить свой первоначально малый охват территории, связавшись с ближайшими группами Сопrotивления. Время на Geoscape заморожено, но стоит вам начать действовать — устанавливая контакт с Сопrotивлением; приобретать ресурсы; выходить на черный рынок — вы активируете таймер, и счет идет на дни.

Это нервует. В любой момент ваши поиски может прервать атака пришельцев или миссия, которая позволит вам атаковать инопланетян. Некоторые миссии можно игнорировать, но это неразумно: они приносят важные ресурсы, дают вашим солдатам шанс обрести опыт и противостоять Темным событиям — всяческим проискам инопланетян, способным уполонинить ваш месячный доход, или вдогонку за Мстителем отправят перехватчик. Создатели умело используют нехватку возможностей в игре, чтобы ставить перед вами трудные задачи. Вам нужны новобранцы, нужен инженер, который создал бы

» **Важен каждый боец, и терять их мучительно больно.**



» **Непродуманный шаг или случайное невезение может стоить солдату жизни.**

средство связи с другими территориями, нужны применяемые пришельцами сплавывы, чтобы обновлять оружие. А получить, вероятно, удастся только что-то одно.

## Приговоренные

Что блестяще, вам придется сканировать, даже чтобы собрать накопленные вами за месяц ресурсы, спрятанные под ландшафт, чтобы избежать обнаружения инопланетянами. Мы оставили запасы на земле на неделю, потому что нам надо было нанять инженера. Нам нужно уничтожить базу пришельцев, чтобы приостановить реализацию их проекта Аватар [Avatar] — а дни наши сочтены, и если времени не хватит, конец будет плохим. Нам нужны были трупы солдат Адвента, чтобы получить жизненно важную модернизацию брони. Нужно было выпить чашку чая, так как все это слишком выматывало. Столь малые возможности идеально соответствуют ситуации. Вы берете все, что можно. Вы готовы выскрести еду и топливо из грязи, только бы Мститель держался в воздухе.

Стоит таймеру замереть во время сканирования, и вы покойник. Есть экран уведомлений, по которому надо щелкать, чтобы узнать, какая напасть вам грозит. Если вам повезет, то это союзник решил вас порадовать и сообщить, что они вам оставили пару сэндвичей в Южной Америке. Если нет, то вас ждет миссия в духе *Enemy Unknown Terror*, только от XCOM 2. Бой ведется пошагово и происходит на отлично выполненных полях для длительных сражений. Заснеженные леса, трупщобы, городские центры и базы пришельцев разнообразны как в плане окружающих деталей, вроде обтекаемых футуристических автомобилей

и пушистых деревьев, так и в вертикальной проекции, в виде утесов и многоэтажных зданий. Взрываются они тоже красиво.

Как только вас раскрыли, жизнь намного усложняется. Удачный выстрел зависит от случайности, и порой шансов гораздо больше, если остаться в надежном укрытии и не высовываться. Неловкий шаг или невезение может уничтожить солдата или выбить из строя на пару дней. Ограниченные по времени цели, вроде взлома терминала или спасения/уничтожения VIP-персоны, заставят вас проявить безрассудство. Не будем раскрывать, каких сюрпризов и ужасов мы натерпелись от более продвинутых инопланетных войск, но после парочки таких мы были в полном отчаянии после боины.

Благодаря различию ваших начальных позиций, продолжительным миссиям и тактической глубине, в XCOM 2 можно и нужно играть несколько раз. **LXF**

## LINUX FORMAT Вердикт

### X-COM 2

**Разработчик:** Firaxis  
**Сайт:** <https://xcom.com>  
**Цена:** £ 35

<b>Сюжет</b>	9/10
<b>Графика</b>	8/10
<b>Увлекательность</b>	8/10
<b>Оправданность цены</b>	8/10

» Необычайно трудная, достойная игра-стратегия, и мастерская переработка формата XCOM.

**Рейтинг 9/10**





# Мобильные НОВОСТИ

## СЛАБОЕ ЗВЕНО

# Тормозит Wi-Fi? Дело в проводах

Последняя версия 802.11ac исчерпала возможности кабеля.

Так называемый Gigabit Wi-Fi, или 802.11ac, медленно, но верно начинает проникать в частные дома, производственные помещения и общественные точки доступа. Однако, согласно анализу исследовательской компании OpenSignal, зачастую соединение происходит на скоростях, значительно ниже заявленных. Этот факт отражает производительность проводных сетей, сопряженных с Wi-Fi. Используя данные бесплатного приложения OpenSignal для мониторинга соединения, аналитики выяснили, что 802.11ac, последняя версия Wi-Fi, обеспечивает пользователям среднюю скорость 32,4 Мб/с. Это более чем вдвое быстрее, чем у предыдущей версии, однако новейшая технология предполагает 400 Мб/с даже в самом базовом виде, при наличии только одной антенны, как у всех современных смартфонов. Оказалось, что в большинстве случаев реальное ограничение скорости создают возможности кабеля, подключенного к точке доступа. Так, если данные проходят через широкополосный кабельный канал с пропускной способностью 25 Мб/с, такая скорость и будет на выходе.

В то время как точки доступа Wave 2 поддерживают скорости до 7 Гб/с, что и делает 802.11ac



Хаб Wi-Fi сети LinkNYC на улице Нью-Йорка.

еще быстрее, Ethernet начинает отставать. Некоторые новые интерфейсы LAN поддерживают скорость 5 Гб/с, но в целом соединения Gigabit Ethernet недостаточно быстры для точек доступа Wave 2, а полный переход на 10-Gigabit Ethernet — удовольствие недешевое.

В настоящее время как телефоны, так и сети, поддерживающие протокол 802.11ac, встречаются довольно редко. Даже в Норвегии (по данному показателю — стране №1, согласно исследованию

OpenSignal), пользователи провели на 802.11ac только 11,4% времени. На втором месте — США, 7,9%. Большую часть времени, когда смартфон использует Wi-Fi, это более медленное соединение 802.11n. Однако выявлен и один безусловный лидер: пользователи Google Nexus 6P в сетях 802.11ac провели 45% времени в своих соединениях Wi-Fi. Гики, пользующиеся флагманом Android от Google, окружают себя последними, самыми быстрыми технологиями.

## БЕСПРОВОДНЫЕ СЕТИ

# Эволюция Wi-Fi

Google начала тестирование инновационной сети 3,5 ГГц.

После создания год назад Федеральной комиссией по связи (FCC) платформы Citizens Broadband Radio Service, работающей в диапазоне 3,5 ГГц и допускающей его совместное динамическое использование, это первый крупномасштабный тест подобного рода в США. Он продлится 18 месяцев, а результатом станет повышение скорости беспроводных соединений малой дальности для обслуживания районов, не охваченных Google Fiber. Антенны установлены на столбах освещения и других сооружениях в восьми районах Канзас-Сити, штат Миссури.

Представитель комиссии назвал диапазон частот 3,5 ГГц «инновационным», и не исключил

возможность его эволюционирования в новую разновидность Wi-Fi или даже «нелицензированный диапазон LTE».

Коммерческий потенциал и функциональные возможности диапазона 3,5 ГГц огромны, однако для доступа к нему новым смартфонам и планшетам потребуется соответствующая антенна; в то же время, на ноутбуке подключение обеспечивает простой адаптер, вставленный в USB-порт. Теоретически, в этой полосе поддерживается скорость беспроводного соединения до 300 Мб/с (для сравнения, средняя скорость многих 4G LTE от 10 Мб/с до 20 Мб/с). Спектр 3,5 ГГц могут задействовать для беспроводных соединений и устройства Интернета

вещей. Google Fiber впервые установил соединение с Интернетом на скорости 1 Гб/с в домах Канзас-Сити еще в 2012 г., однако число подписчиков этого сервиса Google не открывает.

Тестирование диапазона 3,5 ГГц, которое проводит входящая в созданный в прошлом году холдинг Google Alphabet организация Access, никак не связано с виртуальным мобильным оператором Project Fi, еще одним проектом Google, однако, по мнению некоторых аналитиков, эти два направления в конечном счете могут быть объединены. Успех эксперимента в Канзас-Сити позволит Google создавать высокоскоростные беспроводные сети по всему миру.



## ЗА ЧЕСТНУЮ КОНКУРЕНЦИЮ

# Google сочли душителем

## ЕС обвиняет Google в навязывании производителям смартфонов своих приложений и сервисов.

**В** апреле Европейская комиссия предприняла новый антимонопольный выпад против Google. Компанию обвиняют в навязывании производителям смартфонов на базе Android своего поискового приложения и браузера *Chrome* как условия лицензирования других приложений и сервисов Google.

Кроме того, Еврокомиссия заподозрила Google в воспрепятствовании продажам производителями устройств под управлением вариантов или ответвлений ОС Android, а также в предоставлении материальных стимулов производителям телефонов и операторам мобильной связи, согласным на предустановку Google Search в своих устройствах.

По данным StatCounter, в марте Android занимал более 66% европейского рынка мобильных устройств. В целом же, Google принадлежит свыше 90% в поиске, требующих лицензирования «умных» мобильных ОС и магазинах приложений для Android на каждом из рынков Европейской экономической зоны (EEA).

Комиссия объявила о начале расследования, которое изучит поведение Google относительно ОС Android, в том числе, имело ли место незаконное препятствование Google разработке и доступу к рынкам конкурирующих мобильных приложений или сервисов, выразившееся в требованиях или предоставлении стимулов производителям за предустановку исключительно собственных продуктов Google. Еврокомиссар по вопросам конкуренции Маргрете Вестагер [Margrethe Vestager] указала, что альтернативные поисковые системы, мобильные ОС и web-браузеры были искусственно исключены из конкуренции: «Google перекрыл новым приложениям один из основных путей доступа к клиентам».

По правилам Еврокомиссии, предъявленное Statement of objections [официальное уведомление ЕС юридическому лицу о начале антимонопольного расследования] есть формальный шаг, которым заинтересованным сторонам сообщается о выдвинутых претензиях в письменном виде. Окончательное



➤ Маргрете Вестагер, комиссар по вопросам конкуренции Еврокомиссии, выступает на пресс-конференции в Брюсселе 20 апреля 2016 г.

решение принимается после того, как «стороны осуществили свои права на защиту». Если Google признают виновной, компании грозит штраф в размере до 10% от ее глобального годового дохода, т.е. штраф составит \$7,5 млрд, поскольку доходы Google за последний год приблизились к \$75 млрд.

## «ДОКТОР ВЕБ» ПРЕДУПРЕЖДАЕТ

# Android.SmsSpy.88.origin наглеет

## Известный с 2014 г. троянец все еще атакует клиентов банков по всему миру.

**О**дна из главных особенностей троянца *Android.SmsSpy.88.origin* заключается в том, что благодаря ему киберпреступники могут атаковать клиентов фактически любого банка. Для этого им необходимо лишь создать новый шаблон мошеннической формы аутентификации и отдать вредоносному приложению команду на обновление конфигурационного файла, в котором будет указано название банковского клиентского ПО соответствующей кредитной организации.

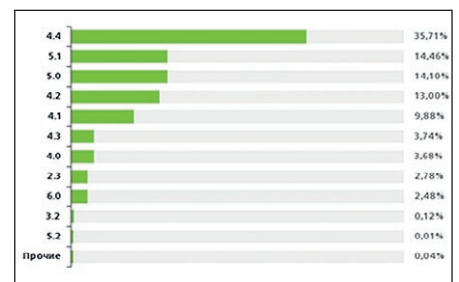
Новая версия троянца по-прежнему пытается похитить у пользователей информацию об их кредитных картах. Для этого *Android.SmsSpy.88.origin* отслеживает запуск ряда популярных приложений, а также некоторых системных программ, и после того, как данное программное обеспечение начинает свою работу, троянец показывает поверх его окна

фишинговую форму настроек платежного сервиса Google Play.

Троянец способен выполнять и другие вредоносные действия: перехватывать и отсылать SMS- и MMS-сообщения, отправлять USSD-запросы, рассылать SMS по всем номерам из телефонной книги, передавать на сервер все имеющиеся сообщения, установить пароль на разблокировку экрана или заблокировать экран специально сформированным окном.

Обновленный *Android.SmsSpy.88.origin* обладает функцией самозащиты: вредоносное приложение пытается помешать работе целого ряда антивирусных программ и сервисных утилит, не позволяя им запускаться.

С начала 2016 г. специалисты компании «Доктор Веб» получили доступ к более чем 50 бот-сетям, которые состояли из мобильных устройств,



➤ Версия ОС Android, установленная на устройствах, зараженных *Android.SmsSpy.88.origin*.

зараженных различными модификациями *Android.SmsSpy.88.origin*, и установили, что киберпреступники атаковали пользователей более 200 стран, а общее подтвержденное число инфицированных устройств достигло почти 40000. LXF

# Разработка приложений



**Виталий Сороко** анализирует возможность поучаствовать в раскрутке альтернативной ОС для мобильных устройств.

Операционная система Tizen является одной из альтернатив чрезвычайно распространенной на различных мобильных устройствах ОС Android. В целом эти системы довольно похожи и служат одной цели. Тем не менее, несмотря на давний выход (первый выпуск ОС Tizen состоялся 5 января 2012 г.), Tizen пока не завоевала такую же популярность, как Android. Но некоторые эксперты возлагают на нее большие надежды и предполагают, что в будущем данная ОС способна заменить Android. Так это или нет, на 100% сейчас утверждать нельзя, но одним из важных показателей этого является сложность или простота процесса разработки приложений, что сильно влияет на потенциальное количество будущих приложений для ОС. Популярность любой ОС неразрывно связана с наличием приложений для нее: если не будет достаточного количества популярных приложений или их аналогов, то в итоге это отпугнет конечных пользователей, а количество приложений зависит в том числе и от наличия удобных инструментов для разработчиков. Итак, в этой статье будет вкратце рассмотрен процесс программирования приложений для ОС Tizen и нужные для этого инструменты.

Начнем с главного, а именно — с того, что приложения для ОС Tizen могут быть двух основных типов: нативные («родные») приложения, которые пишутся на C, и web-приложения, которые пишутся с использованием различных web-технологий (HTML, JavaScript/jQuery, CSS). Допускается также создание гибридных приложений, которые представляют собой совмещение обоих этих типов внутри одного проекта с целью добиться более широкой функциональности. Кроме этого, в процессе разработки еще используются специальные профили для различных типов устройств: например, есть отдельные профили смартфонов, умных часов и других устройств. И, как и в случае любой другой ОС, у Tizen есть свой собственный



Так выглядит оффлайн-эмулятор смартфона HD Mobile, в котором можно проверить полученный результат.

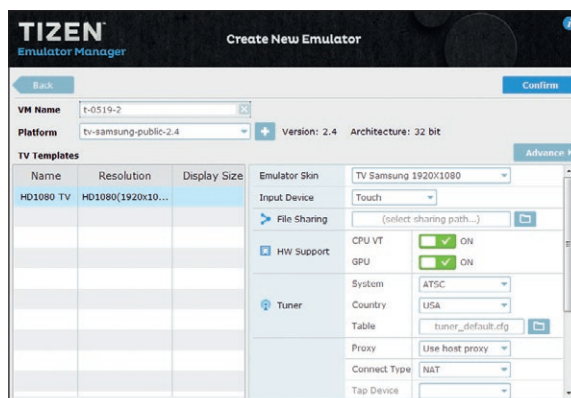


интерфейс программирования приложений (API) и модель программирования. Основные инструменты разработчиков — среда разработки Tizen IDE и интерфейс командной строки (CLI), который позволяет разрабатывать приложения для Tizen без использования Tizen IDE. Интерфейс командной строки будет необходим в основном в случаях, когда вы по каким-то причинам не можете использовать Tizen IDE или просто хотите редактировать исходный код в другой среде или редакторе без использования Tizen IDE.

Скачать Tizen IDE и CLI можно с официального сайта — <https://developer.tizen.org/development/tools/download>. Для установки также может потребоваться Java SE Development Kit 8, доступный на <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>. После успешной установки у вас по умолчанию будут доступны инструменты для разработки только web-приложений, а запуск и тестирование полученного результата будут производиться при помощи web-сервиса Tizen Web Simulator. И если вы хотите разрабатывать нативные приложения или установить не требующий соединения с Интернетом эмулятор, понадобится установка дополнительных пакетов при помощи Tizen Update Manager, запуск которого предлагается в диалоговом окне, автоматически появляющемся по окончании установки. То же надо сделать и при необходимости установки дополнительных профилей устройств (например, для TV-устройств). Tizen IDE основана на Eclipse, т.е. унаследовала все плюсы и минусы этой среды разработки.

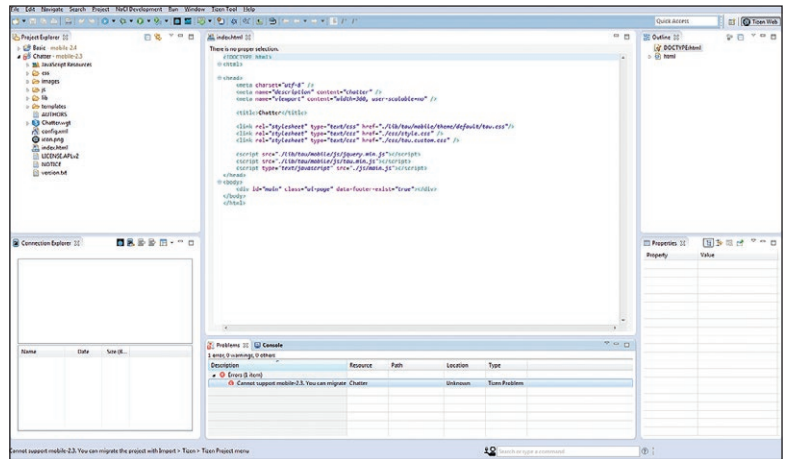
Чтобы начать разработку своего приложения, необходимо запустить Tizen IDE и при ее старте выбрать **Workspace** (основную папку, где будут сохраняться данные самой IDE). Затем вам надо будет создать новый проект, воспользовавшись пунктом меню File

Менеджер, предназначенный для настройки и управления эмуляторами устройств.





> New > Tizen Web Project. В появившемся диалоговом окне следует выбрать нужный вам профиль устройства. В том же диалоговом окне, перейдя на соответствующую вкладку, можно также скачать доступные примеры приложений, такие как Bluetooth Chat, Contacts Exchanger, (Circle) Music Player, Heart Rate Monitor и др. В процессе создания или по завершении работы над своим проектом вы сможете запустить полученный результат по кнопке Run, находящейся на основной панели инструментов. После нажатия на нее надо будет выбрать тип эмулятора. Как уже отмечено, если вы ничего дополнительно не устанавливали, то по умолчанию вам будет доступен только web-эмулятор, который называется Tizen Web Simulator. Он предоставляет доступ только к двум основным типам устройств: mobile (смартфоны) и wearable («носимое» устройство — например, умные часы). Если же вам необходимо протестировать свое приложение для какого-либо ТВ-устройства, то для этой цели также существует web-эмулятор под названием Samsung TV Web Simulator, который необходимо установить отдельно, при помощи Update Manager'a. Перечисленные web-эмуляторы очень удобны — за счет того, что при высокоскоростном интернет-соединении их загрузка происходит очень быстро. Но если ваш интернет-канал не слишком быстр либо интернет-соединение ограничено или отсутствует, можно воспользоваться оффлайн-эмуляторами (не требующими интернет-соединения), которые, как и в предыдущем случае, устанавливаются при помощи Update Manager'a. После установки оффлайн-эмулятора(ов) у вас в основном меню ОС должен появиться специальный инструмент для управления ими, под названием Emulator Manager. В нем вы увидите список всех доступных на данный момент оффлайн-эмуляторов и сможете произвести требуемые настройки доступных виртуальных устройств. Заметим, что оффлайн-эмуляторы работают на базе виртуальных машин QEMU, а значит, в процессе своей работы будут потреблять вычислительные ресурсы вашего компьютера (нагружать процессор и расходовать оперативную память). Кроме того, загружаются оффлайн-эмуляторы медленнее, чем web-эмуляторы, так как VM нужно некоторое время на загрузку ОС Tizen и подготовку виртуального устройства к работе. Зато при использовании оффлайн-эмуляторов на базе VM вы получаете практически точную программную копию реального устройства, а это значительно улучшает процесс тестирования приложения и в ряде случаев позволяет избежать дальнейших проблем в работе приложения на реальных физических устройствах. В общем, у обоих типов эмуляторов есть свои преимущества и недостатки, и какой из них будете использовать вы — решать только вам. Лично я установил себе и использую оба: первый — в процессе разработки, а второй — для тестирования после завершения основных работ над проектом.



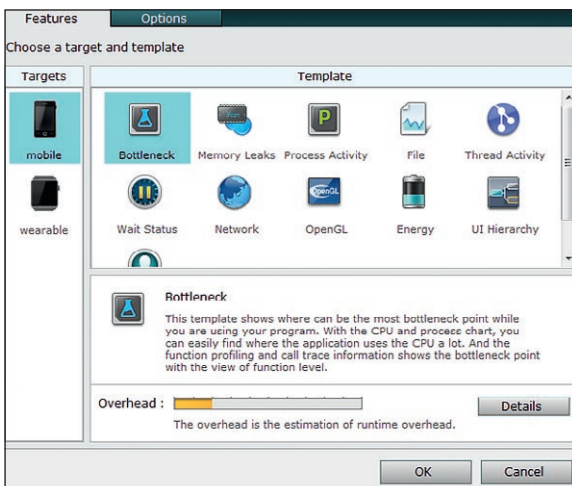
➤ Внешний вид основного окна IDE Tizen с открытым web-проектом.

В состав Tizen IDE входит еще один важный и удобный инструмент для тестирования приложений: это Dynamic Analyzer. Он запускается по нажатию одноименной кнопки на основной панели инструментов и предназначен для сбора данных о расходуемых приложением ресурсах на тестируемом устройстве. Данный инструмент предоставляет такие данные, как загрузка процессора и ОЗУ, расход заряда батареи и сетевого трафика, операции с файловой системой, активность процессов и многое другое. Помимо этого, с помощью Dynamic Analyzer'a вы сможете найти в своем приложении так называемые «узкие места», которые вызывают пиковую нагрузку на процессор, и оптимизировать производительность в OpenGL. В общем, Dynamic Analyzer — очень полезный, а иногда и просто незаменимый инструмент для тестирования и оптимизации созданных приложений.

## Среди доступных разработчикам инструментов есть всё необходимое и полезное.

Отметим, что поскольку ОС Tizen — открытое ПО, вы можете разрабатывать не только приложения для нее, но также стать и разработчиком компонентов самой ОС, то есть Platform Developer'ом. Необходимую документацию по процессу разработки компонентов ОС Tizen вы найдете на официальном сайте платформы: <https://source.tizen.org/documentation/developer-guide>. Ну и не обойдется без небольшой ложки дегтя. После установки и запуска Tizen IDE, как только я решил попробовать установить дополнительные компоненты (первым из них должен был стать оффлайн-эмулятор), Update Manager отказался это делать по причине того, что версию дополнительного компонента нельзя установить из-за более новой версии IDE среды. В итоге я нашел простой и быстрый способ решения этой проблемы — деинсталляция всех уже установленных компонентов с последующей установкой их заново средствами Update Manager'a. После этого дополнительные компоненты установились без каких-либо ошибок и неприятностей. И если вы вдруг столкнетесь с аналогичными проблемами, просто поступите, как я.

В целом, процесс разработки приложений для ОС Tizen не сильно отличается от такового для других ОС. Среди доступных разработчикам инструментов есть всё необходимое и полезное. Поэтому определенный потенциал у данной ОС однозначно есть. Тем не менее сейчас ее перспективы зависят во многом от других факторов, и основной из них — это появление в продаже широкого спектра устройств на базе Tizen. При условии, что такие устройства будут обладать хорошим соотношением цена/качество и будет проведена грамотная рекламная кампания, ОС Tizen сможет занять достойное место на рынке. LXF



➤ Панель управления Dynamic Analyzer, где можно выбрать целевое устройство и требуемый вид анализа.

# PHP на Android

Виталий Сороко признается в любви к web-программированию и пытается заразить ею и вас.

Эта статья посвящается всем тем, кто любит web-программирование так же сильно, как и я, и ни дня не может провести без него. Идея написания этой статьи появилась в тот момент, когда у меня возникла необходимость написать на PHP несколько web-страниц на планшете по пути домой. Сразу после этого мне стало интересно узнать, что выйдет из этой затеи и сможет ли потянуть мой старый, но верный друг-планшет такие задачи; и я немедленно начал действовать. Процесс оказался не таким простым, как я думал. В Интернете я нашел полным-полно статей про программирование на Android-устройствах, однако большая их часть была посвящена программированию на Java и созданию приложений для Android-устройств. Поэтому мне пришлось самому придумать подробный план действий, а затем опробовать множество Android-приложений и вариантов того, как это можно сделать лучше, проще, удобнее и быстрее.

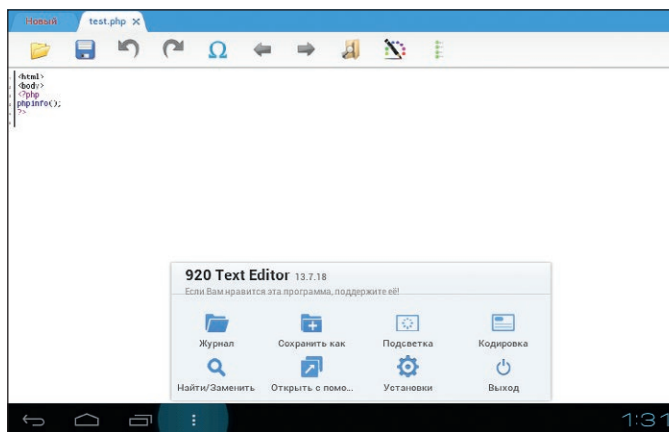
Итак, в соответствии с моим планом, для программирования на языке PHP и создания web-страниц на Android устройстве требовались следующие вещи:

- » Удобный текстовый редактор.
- » Web-сервер с необходимыми компонентами.
- » Удобный браузер и компоненты для верификации полученного результата.

Для установки всего этого я использовал семидюймовый планшет на базе Android 4 с внешней клавиатурой и следующими параметрами: однокорневой 1-ГГц процессор, 512 МБ ОЗУ и 500 МБ



» Вот на этом устройстве и выполнялось тестирование описанных здесь приложений для Android 4.



» Так выглядит текстовый редактор 920 Text Editor, который имеет подсветку синтаксиса и открытый исходный код.

свободного места на системном диске. Несмотря на то, что у меня в пользовании имеются и более мощные девайсы, я специально выбрал именно это устройство — потому, что оно обычно всегда у меня под рукой благодаря своему небольшому размеру, и мне хотелось проверить, насколько хорошо и удобно заниматься PHP-программированием именно на низкопроизводительном планшете.

## Текстовые редакторы

Естественно, что для удобного программирования и создания web-страниц понадобится текстовый редактор с подсветкой синтаксиса. При помощи приложения F-Droid, доступного для установки из Google Play, в стандартных репозиториях программ с открытым исходным кодом вы можете выбрать вариант текстового редактора на ваш вкус. Лично мне приглянулись и понравились следующие три варианта:

- » Turbo Editor (по умолчанию имеет черный фон, а подсветку синтаксиса нужно включать в настройках; этот редактор умеет подсвечивать HTML-теги в коде PHP);
- » 920 Text Editor — подсветка синтаксиса включена по умолчанию, дизайн и интерфейс приятный и напоминает классические текстовые редакторы из мира настольных ПК;
- » VIMTouch (отличный редактор Unix-way; подойдет только тем, кто знает, как этим пользоваться, и любит работать в Vim).

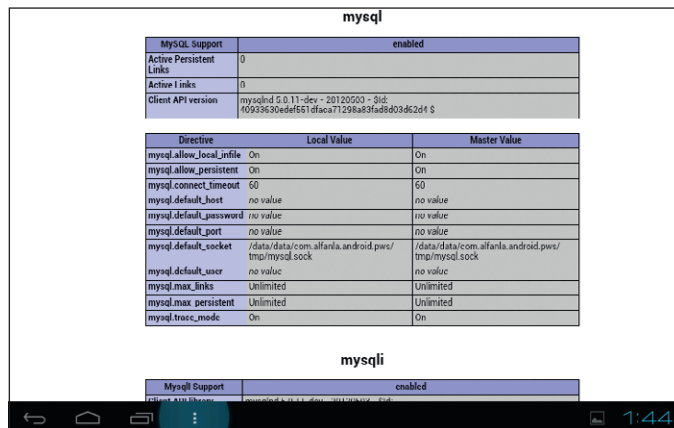
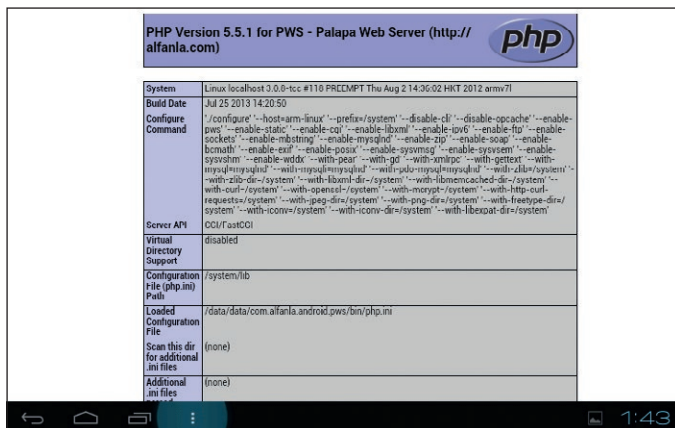
## Web-сервер

Есть два способа, при помощи которых можно установить web-сервер с поддержкой PHP, а также базой данных и другими необходимыми вещами.

Начнем с описания сложного способа, который подразумевает установку внутри Android дополнительной операционной системы (например, Debian, которую можно попробовать установить, воспользовавшись компонентом Debian Kit). Однако эта система будет загружаться вместе с Android и работать параллельно, нагружая тем самым ваше устройство. И вообще, на мой взгляд, у этого подхода больше минусов, чем плюсов. Судите сами: из-за дополнительной нагрузки медленные устройства будут тормозить, будет быстрее разряжаться аккумулятор, что очень важно в дороге, и самое главное — вам надо уметь настраивать эту дополнительную систему и все необходимые компоненты, что несомненно отнимет у вас дополнительное время и силы. Я уже опробовал данный способ на себе и своем устройстве и поэтому не могу его рекомендовать как простой, быстрый и удобный, так как в моем случае для этого пришлось не только потратить много времени, но еще и сделать устройство рутованным, что, на мой взгляд, не очень хорошо. Кроме этого, в процессе установки Debian было израсходовано много мобильного трафика, а это может быть неприемлемо, если у вас не безлимитный тарифный план.

В итоге я нашел более простой и быстрый способ превратить свое устройство в web-сервер с поддержкой PHP и MySQL — установить Palara Web Studio, которая бесплатна и используется для организации работы web-сервера в основном свободные компоненты, а именно web-сервер *lighttpd*, PHP, *MySQL*, *phpmyadmin* и т.д. Чтобы всё установить, достаточно просто загрузить и установить эту студию при помощи Google Play. Сразу же после установки вам станет доступна удобная





➤ Phpinfo на web-сервере *Lighttpd*, входящем в состав набора приложений Palapa Web Studio и поддерживающем PHP версии 5.5.1.

➤ Благодаря phpinfo мы также можем узнать, что внутри Palapa Web Studio есть СУБД *MySQL* версии 5.0.11 с поддержкой *mysqlq*.

локальная панель управления, которая появляется при открытии студии (сам web-сервер по умолчанию будет доступен по адресу <http://127.0.0.1:8080>). Помимо нее, в составе Palapa Web Studio имеется также и web-панель управления, доступная по адресу <http://127.0.0.1:9999> (имя пользователя и пароль — admin), а также СУБД *MySQL* (имя пользователя *MySQL* по умолчанию — root, а пароль — adminadmin) с web-панелью для управления базами данных *phpmyadmin*, доступной по адресу <http://127.0.0.1:9999/phpmyadmin>. Директория по умолчанию для web-документов находится на встроенной SD-карте по адресу `/mnt/sdcard/pws/www/`, а логи web-сервера и *MySQL*, необходимые для процесса отладки, хранятся в директории `/mnt/sdcard/pws/logs/`.

Важно также знать, что при наличии подключения по Wi-Fi web-сервер может стать доступным по выданному маршрутизатором IP-адресу. Если это для вас нежелательно, то перед подключением к какой-либо беспроводной сети лучше отключайте web-сервер вручную через панель Palapa Web Studio в приложении.

## Web-браузер

Мир программного обеспечения с открытым исходным кодом дарит вам полную свободу в выборе мобильного браузера для ОС Android. Только в базовых репозиториях F-Droid их можно найти более пяти. Самым известным из них, пожалуй, является *Firefox*, и он неплохо подходит для процесса web-разработки по следующим причинам:

➤ К нему есть плагин для просмотра исходного HTML-кода web-страницы, под названием *View Source Mobile*.

➤ Есть также плагины для валидации HTML и CSS.

Но *Firefox*, к сожалению, не лишен недостатков: он занимает больше места относительно других браузеров, потребляет много ресурсов и поэтому не очень быстро работает на медленных устройствах, да еще способствует быстрой потере заряда батареи и не всегда сразу открывается. Поэтому лично я нашел для себя и моего не очень быстрого планшета несколько альтернатив:

➤ *mBrowser* Очень легковесный вариант, который при открытии спрашивает, нужно ли загружать скрипты и изображения.

➤ *Lighting* [англ. Молния] Быстрый браузер с поддержкой Flash.

➤ *AOSP* Еще один быстрый браузер, который установлен по умолчанию на многих устройствах и будет необходим в том случае, когда вам понадобится просмотреть страницу в полной версии.

## Тестирование и валидация

Как уже говорилось, в процессе разработки web-страниц нередко возникает необходимость просмотреть получившийся в результате HTML-код, но ни один из вышеописанных альтернативных *Firefox*'у вариантов не дает такой возможности. Чтобы ее получить, вам прежде всего надо будет установить дополнительно еще одно приложение с открытым исходным кодом, которое называется *HTML Source Viewer*.

## Мобильное устройство пригодно не только как тестовый, но и как полноценный web-сервер.

На первый взгляд, пользоваться отдельным приложением для просмотра HTML-кода вне браузера неудобно, но у *HTML Source Viewer* есть одно важное преимущество по сравнению с тем же плагином *Firefox*: это форматирование кода. Каждый раз после получения исходного кода web-страницы *HTML Source Viewer* форматирует его (с подсветкой тэгов), делая приятным для чтения. Именно поэтому я его рекомендую установить, даже если у вас уже есть *Firefox* с плагином *View Source Mobile*. Кроме того, в *HTML Source Viewer* есть функция предпросмотра страницы, благодаря которой вы можете оценить внешний вид страницы, не запуская браузер.

Что касается валидации кода, то, как уже упоминалось выше, можно применить соответствующие плагины для *Firefox* или воспользоваться напрямую онлайн-сервисами валидации W3C. Второй вариант, на мой взгляд, предпочтительнее, поскольку найденные мной на момент написания статьи плагины используют в своей работе эти же сервисы, так что особого смысла в их установке нету. А еще вы можете установить из Google Play бесплатный

аналог *HTML Source Viewer* под названием *VT View Source*, который не форматирует при просмотре HTML-код страницы, но зато имеет много других полезных функций, одна из которых — валидация при помощи сервисов W3C.

## Заключение

В порядке заключения хочу отметить: после многих дней тестирования я убедился, что мобильное устройство пригодно к использованию не только как тестовый, но и как полноценный web-сервер. Например, на него можно поместить не очень высоконагруженный сайт для дальнейшей раздачи в локальной сети или Интернете. И если с использованием в локальной сети особых вопросов возникнуть не должно, так как к устройству можно будет просто обращаться по выданному маршрутизатором

Wi-Fi локальному сетевому адресу, то при использовании в Интернете вам, скорее всего, понадобится фиксированный IP-адрес и желательно безлимитный в части передачи данных тарифный план. Кроме того, в данном случае я порекомендую использовать стационарный интернет через Wi-Fi-маршрутизатор (опытным путем я обнаружил, что при работе по 3G/4G могут возникнуть проблемы со стабильностью соединения), а чтобы всё было доступно из внешнего мира, надо будет построить на маршрутизаторе Wi-Fi виртуальный сервер с перенадресацией соединений с порта 80 на порт 8080 (при условии, что web-сервер в вашем мобильном устройстве работает именно на этом порту).

После выполнения этих действий в случае правильной настройки вы сможете получать доступ к вашему сайту на мобильном устройстве из любой точки глобальной сети по внешнему IP-адресу, а при желании вы сможете привязать к нему и доменное имя.

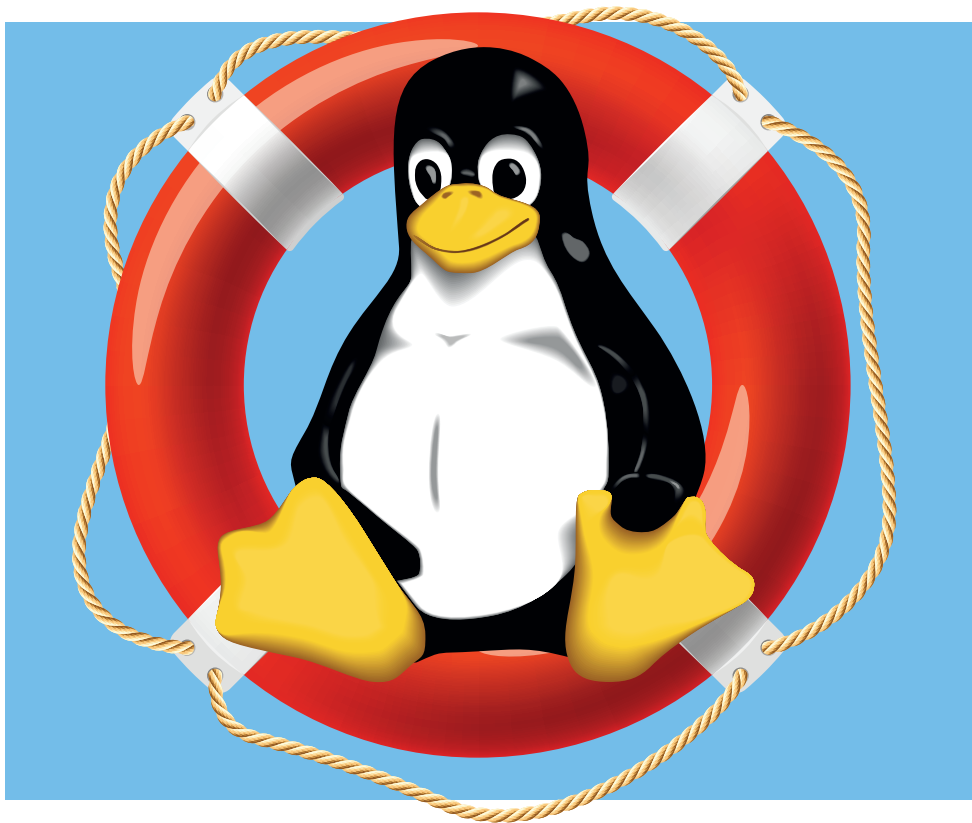
В общем, как видите, современные мобильные устройства могут заменить как ПК, так и сервер, и помогут вам писать код даже в дороге. LXF

# Сравнение

» Каждый месяц мы сравниваем тонны программ — а вы можете отдыхать!

## Спасти и сохранить

Не позволяйте сбою компьютера выбить вас из седла. **Маянк Шарма** тестирует спецдистрибутивы, помогающие навести порядок.



### Про наш тест...

Мы начнем с оценки продуманности инструментариев дистрибутивов — являются ли они супермаркетом, где найдется любой тип восстановления системы? Мы также сравним эффективность их подготовки к возможной катастрофе с оборудованием, вкуче с их способностями справиться с ситуацией, если катастрофа разразится.

Хотя оценка производительности — дело непростое, зависящее от множества факторов, мы сосредоточимся на ценности дистрибутива в подготовке пользователя к работе. Мы также попытаемся определить, есть ли у каждого дистрибутива стратегия относительно включения или не включения определенных инструментов. Восстановление системы — прерогатива сисадмина, а в руках небрежного и несведущего пользователя такие инструменты могут стать оружием массового поражения; поэтому мы также выделяем проекты с хорошей документацией — либо на сайте, либо в самом дистрибутиве.

### Наша подборка

- » Finnix
- » Rescatux
- » System-RescueCd
- » Trinity Rescue Kit
- » Ultimate-BootCD

**Linux** часто — и совершенно справедливо — объявляют стабильной и отказоустойчивой операционной системой. Но дистрибутивы работают на оборудовании, которое отказывает чаще, чем нам хотелось бы думать, и им управляют люди, склонные делать дурацкие ошибки, например, забывать пароли или неумышленно портить программу загрузки.

Занимаетесь ли вы самобичеванием, поскольку случайно удалили важные файлы, или проклинаете жесткий диск за повреждение системных файлов, сделавшее систему незагружаемой, есть специальные

### Разработаны, чтобы собрать в одной посуде всё необходимое для восстановления системы.

инструменты, которые помогут исправить ситуацию и решить проблему. Хотя многие из этих инструментов имеются в репозиториях большинства дистрибутивов Linux, определить подходящий к случаю инструмент может быть весьма непросто, учитывая колоссальный выбор. Вот здесь-то и вступает в игру спасательный дистрибутив. Такие спецдистрибутивы разработаны

для того, чтобы собрать в одной посуде всё необходимое для восстановления системы. Самые популярные из них идут еще дальше, предлагая даже индивидуальный UI, представляющий инструменты и их функции в организованном виде. В нашем Сравнении мы рассмотрим лучшие дистрибутивы, чтобы вы не отчаивались, если ваш компьютер начнет своевольничать.



# Инструментарии

Что в них содержится?

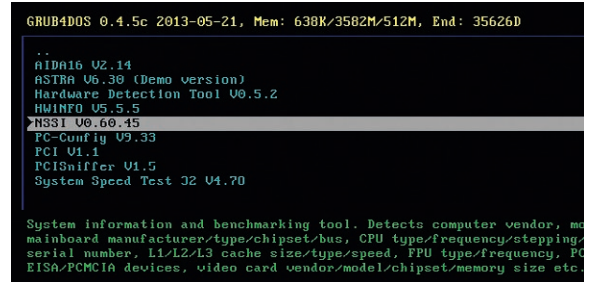
**F**innix — один из самых маленьких дистрибутивов в нашем Сравнении, но даже при весе чуть больше 400 МБ его список инструментов вполне исчерпывающий. В дистрибутиве имеются более крупные утилиты, такие как *partimage* и *smart boot manager*, но в нем масса и инструментов помельче — для оценки и поддержки всех основных файловых систем, а также для восстановления, резервного копирования и исправления поврежденных томов. Есть простые инструменты, обеспечивающие обычный доступ к сети и настройку Wi-Fi, и можно использовать Finnix в качестве файлового сервера, использующий FTP, Samba или NFS. Ну и более эзотерические инструменты для сетевой диагностики и мониторинга.

Однако самый крошечный дистрибутив в нашем Сравнении — это 150-МБ Trinity RescueKit (TRK), который не предлагает графической среды рабочего стола. Дистрибутив не поддерживает философию «один инструмент для одной задачи», применяемой некоторыми из его конкурентов, и, например, предлагает пять разных

сканеров вирусов, а помимо обычного набора задач восстановления TRK умеет также клонировать компьютеры в сети с помощью мультикаста [multicast — многоадресная рассылка].

Ultimate Boot CD (UBCD) — еще один дистрибутив без графического рабочего стола; подобно TRK, UBCD не придерживается концепции привязки к одной задаче единственного инструмента. Цель этого 600-МБ дистрибутива — вместить в загрузаемый CD побольше инструментов диагностики. С этой целью UBCD включает инструменты для: восстановления настроек BIOS; нагрузочные испытания оборудования и разных менеджеров загрузки; инструменты управления диском; и множество других полезных утилит.

Другой 600-МБ дистрибутив, Rescatux, предлагает минимальный графический рабочий стол и включает все важные и полезные инструменты для исправления проблем при отказе загрузки Linux и Windows, в том числе *testdisk*, *photorec*, *GParted* и т.д. Вы можете использовать Rescatux для восстановления загрузчика и файловых



Вы можете использовать Ultimate Boot CD для опроса и стресс-тестов оборудования и периферийных устройств, соединенных с системой.

систем, исправления таблиц разделов и изменения паролей в Linux и Windows.

И последний по очереди, но не по значимости, SystemRescueCd на базе Gentoo предлагает минимальный рабочий стол Xfce и ряд графических приложений, таких, как браузер *Midori* и программа просмотра *ePDF*. При этом 450-МБ дистрибутив содержит немало инструментов восстановления и исправления и утилит, и предлагает поддержку всех важных файловых систем, включая ext4, xfs, Btrfs, NTFS и reiserFS, а также сетевых файловых систем, таких, как Samba и NFS. Более того, для таких инструментов, как *FreeDOS*, *Gag*, графический менеджер загрузки, *Ranish Partition Manager* и т.д., дистрибутив включает образы диска.

**Вердикт**

- Rescatux ★★★★★
- UBCD ★★★★★
- Finnix ★★★★★
- SysRescCd ★★★★★
- TRK ★★★★★

» Выберите Rescatux и SysRescCd с их GUI.

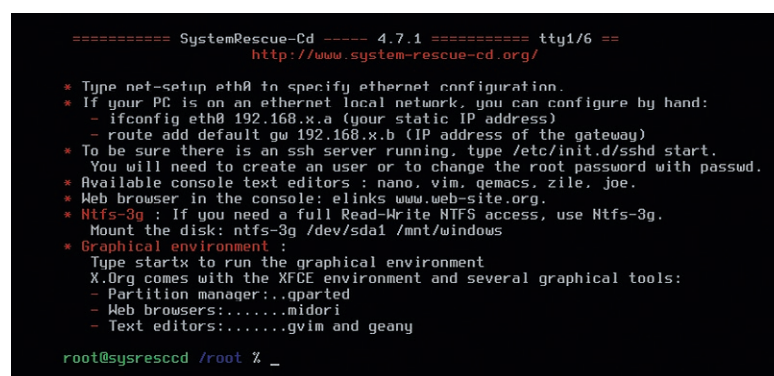
# Настраиваемость

Можете ли вы сделать его исключительно своим?

**O**снованный на Debian Finnix включает несколько опций индивидуальной настройки, в том числе сегменты перекрытия [overlay] и скрипт ремастеринга. На сайте есть подробные инструкции по редактированию структуры ISO-образа и использованию индивидуальных скриптов для переупаковки дистрибутива в пакеты. Но даже и при наличии инструкции эта процедура предназначена исключительно для пользователей с солидным опытом.

Rescatux тоже основывается на Debian и включает менеджер пакетов *Synaptic* для установки дополнительных программ, но здесь нет механизма сохранения изменений в дистрибутиве.

В разделе обратной связи сайт SysRescCd содержит пошаговое руководство, содержащее процедуру добавления индивидуальных пакетов и создания нового ISO. Дистрибутив поставляется с четырьмя ядрами, и имеется руководство по компиляции SysRescCd с вашим собственным ядром. В отличие от руководства Finnix,



руководство SysRescCd более подробно и пригодно в том числе и для относительно неопытных пользователей.

Подобно Finnix и SysRescCd, сайт UBCD предлагает руководство, детально описывающее процедуру декомпиляции ISO-образа, индивидуальной настройки среды — посредством добавления собственных образов дисков и приложений на основе FreeDOS — и затем компиляции нового ISO-образа.

Пользователи Gentoo смогут без особых усилий создавать индивидуальные версии SystemRescueCd, следуя инструкциям в обширном руководстве.

TRK поставляется с утилитой, которая превращает работающую в данный момент среду в ISO-образ. На сайте также предусмотрено руководство, где перечисляются процессы и области, которые необходимо изменить с целью создания индивидуальной сборки.

**Вердикт**

- SysRescCd ★★★★★
- TRK ★★★★★
- Finnix ★★★★★
- UBCD ★★★★★
- Rescatux ★★★★★

» Большинство дистрибутивов настраиваются легко, кроме Rescatux.

# Удобство пользователя

Как увязать всё это вместе?

Программные репозитории вашего дистрибутива кишат инструментами, которые помогают исправить ошибки и восстановить Linux и Windows и данные в них после сбоя. Дистрибутивы, о которых мы рассказываем в нашем Сравнении, предлагают самые лучшие и самые полезные инструменты в удобном пакете. Однако

соседство всех инструментов в одном пакете пригодится только пользователям, уже имеющим опыт работы с подобными инструментами. Среднему пользователю понадобится помощь: сначала — чтобы добраться до инструментов, потом — чтобы их использовать. Причина в том, что у многих инструментов отсутствует графический интерфейс,

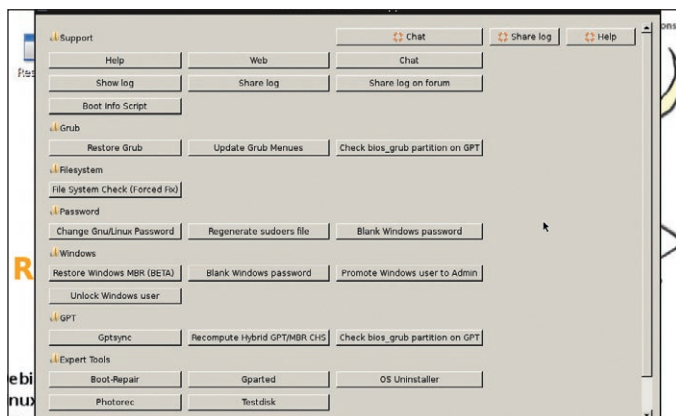
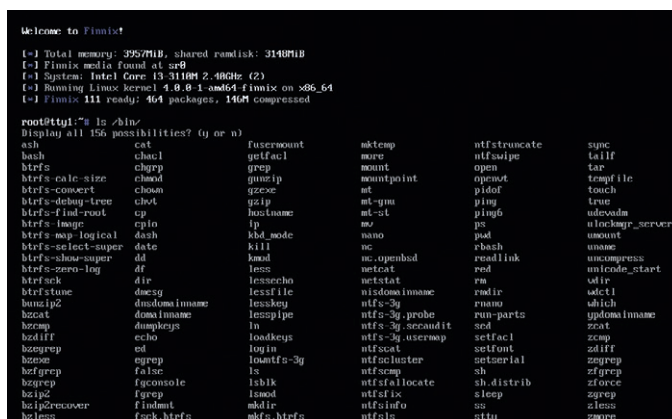
и если их употребить неправильно, это разве что усугубит и без того скверную ситуацию.

По этой причине мы будем обращать особое внимание на дистрибутивы, потратившие время на создание интерфейсов, которые помогут пользователю и уделяют больше внимания встроенной документации.

## Finnix ★★★★★

Меню загрузки дистрибутива дает несколько опций. По умолчанию загружается 64-битное ядро, но предлагается также опция загрузки 32-битного. Остальные опции приведут вас в оболочку FreeDOS или в *Smart Boot Manager*, либо запустят инструмент распознавания оборудования. В опции по умолчанию дистрибутив загружается очень быстро и, проведя вас по процессу, оставляет в командной строке. Для начинающих пользователей это способно стать сюрпризом, особенно с учетом полного отсутствия указаний, что там делать дальше.

Список инструментов Finnix обширен, однако его удобство в использовании ограничено из-за того, что это — дистрибутив командной строки. Сайт проекта в этом отношении тоже не особо полезен. Есть общая документация по некоторым аспектам дистрибутива, однако отсутствие руководства для начинающих является досадным упущением и камнем преткновения для новичков.



## Rescatux ★★★★★

После загрузки дистрибутив открывает минималистский графический рабочий стол на базе LXDE и сам запускает свое индивидуальное приложение помощи под названием *Rescapp*. Интерфейс приложения хорошеет от релиза к релизу, и в последней версии в нем имеется несколько кнопок, разделенных на категории, такие, как Grub, Файловая система [Filesystem] и Пароль [Password]. Кнопки внутри каждой категории снабжены описательными ярлычками, помогающими определить их функцию. При нажатии на кнопку та выводит документацию с подробным объяснением, какие шаги предпримет Rescatux и какую информацию он рассчитывает получить от пользователя. Просмотрев иллюстрированную документацию и поняв, чего ожидать, вы нажимаете на кнопку с надписью Run!, чтобы запустить утилиту. Опытные пользователи могут пропустить *Rescapp*, запустить терминал и получить доступ непосредственно к инструменту восстановления. Сайт дистрибутива также изобилует видеоруководствами.

# Поддержка & документация

Потому что вам наверняка потребуется помощь.

Finnix предлагает документацию по ряду аспектов дистрибутива, таких как параметры загрузки; разрозненные руководства по восстановлению загрузчика; и как использовать дистрибутив для криминальной экспертизы. Большая часть документации рассчитана на опытных пользователей, и на сайте нет руководства для начинающих. Раздел руководств сайта Ultimate Boot CDs тоже не слишком хорошо организован. Он указывает на список предоставленных пользователями руководств, за которыми следят на форумах,

а документация на wiki проекта не особенно полезна. В основном она не завершена и не предназначена для начинающих.

Зато документация на сайтах трех остальных дистрибутивов не разочаровывает. Rescatux вывесил массу руководств и обучающих видео. Имеется также и документация, помогающая новичкам разобраться с дистрибутивом. SysRescueCd предлагает руководство по быстрому старту для начинающих, а также подробные инструкции по обычному и расширенному использованию. Вдобавок на сайте есть инструкции

для ветеранов — например, руководство по созданию индивидуальной версии дистрибутива и резервному копированию данных с незагружаемого ПК под Windows.

Раздел документации сайта TRK начинается с руководства по быстрому старту, переходящее к подробному описанию возможностей его инструментов и индивидуальных утилит. Имеется также очень удобный раздел с подробным описанием процедур, которым надо следовать в определенных ситуациях, например, когда начинает отказывать диск.

**Вердикт**

Rescatux ★★★★★

SysRescueCd ★★★★★

TRK ★★★★★

Finnix ★★★★★

UBCD ★★★★★

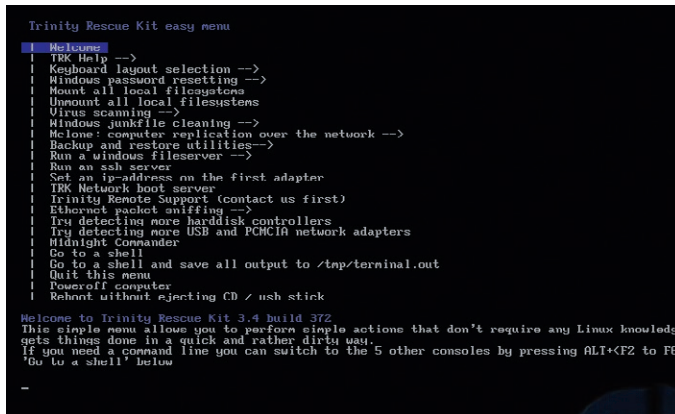
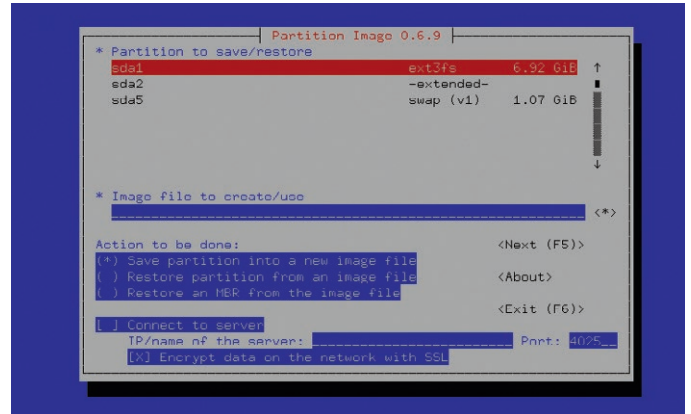
» Rescatux, SysRescueCd и TRK предлагают справочники и руководства.



## SystemRescueCd ★★★★★

Этот дистрибутив предлагает самое всестороннее меню загрузки из всех нами виденных. Помимо десятков опций, перечисленных на основном экране, имеется также подсказка, объясняющая разные расширенные параметры загрузки в нескольких виртуальных консолях. Опция загрузки по умолчанию загружается в консоль, где перечисляются основные команды для настройки сети, монтирования разделов NTFS и запуска графической среды. Однако внутри работающего на *JWM* минимального рабочего стола вы оказываетесь предоставлены сами себе.

Дистрибутив не предлагает никакой справочной документации, и начинающим пользователям придется обращаться к документации на сайте проекта. Вам также придется вызывать инструменты восстановления и исправления вручную, как в *Finnix*, хотя у вас есть некий комфорт графического рабочего стола и *web*-браузер для доступа ко вводной информации на сайте проекта.



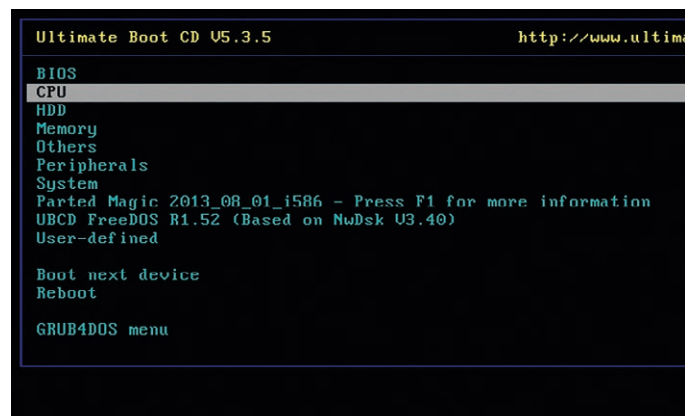
## Trinity Rescue Kit ★★★★★

Подобно SysResCd, TRK потрудился над опциями меню и предлагает их около двух десятков. По умолчанию загружается текстовое меню. Хотя TRK — дистрибутив командной строки, индивидуальное меню отлично справляется с задачей помощи пользователю в нахождении и запуске нужного инструмента. Перечисленные опции также достаточно подробно поясняются, чтобы неопытный пользователь нашел инструмент, пригодный для решения проблемы.

Начальные опции направляют пользователя в встроенную в дистрибутив информацию, однако некоторые опции обращаются прямо к онлайн-документации на сайте через браузер *Links*. Некоторые пункты меню предлагают больше опций для настройки поведения отдельных инструментов. Подменю также указывают на файлы подсказки, которые в основном выводят *man*-страницу. Опытные пользователи могут перейти в оболочку и вызвать инструменты вручную.

## Ultimate Boot CD ★★★★★

Несмотря на массу инструментов, UBCD загружается мигом. В отличие от некоторых его соперников с тщательно продуманными меню загрузки, этот дистрибутив загружается прямо в свое индивидуальное текстовое меню. Меню делит имеющиеся в пакетах утилиты на категории в соответствии с областями их действия, например, BIOS, HDD и память, и т.д., а каждая категория подразделяется на пункты для отдельных инструментов. Ряд категорий, например, HDD, далее подразделяется на разные задачи, например, Data Recovery [Восстановление данных], Disc Cloning [Клонирование диска] и Disk Wiping [Очистка диска], и т.д. При просмотре каждого инструмента в любой категории меню отображает краткую, но важную информацию об инструменте. Как и в TRK, меню может быть текстовым, но понятные категории и полезное описание означают, что навигация по дистрибутиву осуществляется легко и он может быть применен неопытными пользователями для решения любых проблем.



# Функции безопасности

### Помогут ли они вам сохранить конфиденциальность?

**Х**отя исходная цель каждого из этих дистрибутивов заключается в том, чтобы помочь вам восстановиться после сбоя, они также предлагают утилиты для обеспечения безопасности вашей системы и предотвращения утечек конфиденциальности. В *Finnix* вы найдете утилиту командной строки *wipe* для безопасного удаления файлов и *GNU Privacy Guard (GPGV)* для подтверждения подписей GPG. *Rescatux* тоже предлагает *GPGV* и включает *shred* для полного удаления файлов. В *SysResCd* предусмотрены инструменты

для безвозвратного удаления файлов с жесткого диска, *chkrootkit* для поиска руткитов и, наконец, утилита *md5deep*, применяемая в сообществах компьютерной безопасности, системного администрирования и криминальной экспертизы для пропуска большого количества файлов через несколько криптографических модулей.

В TRK имеется несколько движков для сканирования на вирусы, в том числе *ClamAV*, *F-Prot*, *BitDefender*, *Vexira* и *Avast*, а также инструмент создания контрольных сумм *md5sums* для всех файлов в системе

и утилиту *winclean* для очистки ненужных файлов, например, временных файлов на компьютере с Windows.

Кроме инструментов, предназначенных для его изначальной цели, UBCD содержит инструменты для безопасности и обеспечения конфиденциальности. Помимо *ClamAV* и сканера *F-Prot*, здесь есть множество инструментов для очистки диска, в том числе *DBAN*, *HDDerase*, *HDS shredder* для всех видов файловых систем и несколько низкоуровневых утилит для безопасной очистки различных жестких дисков.

### Вердикт

- SysResCd** ★★★★★
- TRK** ★★★★★
- UBCD** ★★★★★
- Finnix** ★★★☆☆
- Rescatux** ★★★☆☆

» SysResCd, TRK и UBCD предлагают несколько инструментов защиты приватности.

# Индивидуальные инструменты и UI

Могут ли они пойти еще дальше?

**Д**ело в создании успешного дистрибутива восстановления, а не в том, чтобы упаковать в него как можно больше инструментов. Именно небольшие и индивидуальные настройки помогают дистрибутиву выделиться из толпы.

Хорошим примером могут послужить индивидуальные функции Finnix: скрипт для облегчения сетевой загрузки [netbooting] дистрибутива с другого компьютера. Скрипт генерирует соответствующий initrd, который включает сетевые и NFS-модули. Утилита также настроит сервисы NFS и TFTP, так что работающий Finnix станет сервером загрузки через сеть. Дистрибутив

включает утилиту под названием *finnix-hw-submit*, которая позволяет вам предоставить подробности технических характеристик вашей аппаратной платформы разработчику, чтобы помочь исправить ошибки.

SysRescCd представил поддержку загрузки SRM (System Rescue Modules — Модулей Восстановления Системы). Это файловые системы squashfs, которые содержат дополнительные файлы, являющиеся частью системы. Эта функция полезна для добавления новых приложений к дистрибутиву. В дополнение к приложениям, модули можно также использовать для добавления

индивидуальных файлов данных. Имеется также утилита *sysresccd-backstore*, пригодная для создания закольцованной файловой системы, где можно хранить индивидуальные файлы.

Как мы уже упоминали, Rescatux, TRK и UBCD загружают индивидуальные текстовые или графические меню, которые помогают пользователю найти утилиту для решения своей проблемы с системой. В Rescatux меню ведет в индивидуальный интерфейс для утилит командной строки, использующий мастер для прохождения разных стадий процесса. Подобным же образом в TRK меню можно использовать для настройки и запуска нескольких задач, таких, как сканирование на вирусы и перенастройка паролей Windows через индивидуально настроенные текстовые мастера. Дистрибутив также содержит предоставленное пользователями приложение резервного копирования под названием *pi* и индивидуальные скрипты для облегчения выполнения некоторых административных задач, таких, как возможность находить и монтировать все локальные файловые системы.



➤ Trinity Rescue Kit предлагает аж пять разных сканеров вирусов, интегрированных в одну утилиту командной строки.

## Вердикт

- Rescatux ★★★★★
- TRK ★★★★★
- UBCD ★★★★★
- SysRescCd ★★★★★
- Finnix ★★★★★

➤ Если вы не знакомы с инструментами восстановления, избегайте Finnix.

# Целительные свойства

С какими недугами они способны справиться?

**Д**аже если вы стерли главную загрузочную запись (MBR) или заблокировали себе вход в собственную учетную запись, инструменты в этих дистрибутивах помогут вам выпутаться из неприятностей. Однако некоторые дистрибутивы более сведущи, чем остальные.

Например, Finnix загружается с инструментами. Он может все, от создания нового пароля root до управления, восстановления и исправления поврежденных файловых систем и томов. Но этот дистрибутив нельзя использовать для восстановления загрузчика Grub 2.

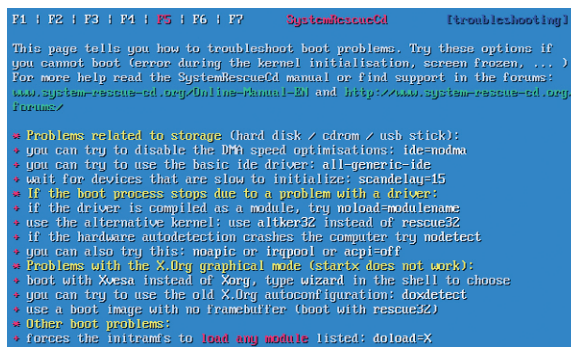
В Rescatux нет подобных ограничений: он может восстановить все виды загрузчиков; помочь вам восстановить MBR; изменить пароль в Windows и Linux; и исправить все популярные файловые системы. Дистрибутив также предлагает инструменты для восстановления данных и файлов и может удалить Windows и Linux.

SysRescCd умеет сохранять и восстанавливать таблицу разделов и сохранять

содержимое файловой системы в сжатом архивном файле с помощью *FSArchiver*. Помимо восстановления удаленных файлов и определения и устранения вирусов, он также может создавать образы дисков и клонировать диски; определять и выводить список поврежденных секторов; и извлекать поддающиеся восстановлению данные с физически поврежденных дисков.

В нашей версии TRK имеется улучшенный инструмент *Winpass*, способный легко

менять пароли Windows. И, конечно, вы можете использовать TRK для восстановления удаленных файлов и даже потерянных разделов. Однако больше всех предлагает UBCD. Дистрибутив пригоден для восстановления настроек BIOS, а также восстановления и резервного копирования CMOS. Он может исправлять загрузчики, восстанавливать потерянные пароли и удаленные файлы, а также поможет исправить реестр Windows, не загружаясь в вашу систему.



➤ Меню загрузки SystemRescueCd предлагает решения проблем с загрузкой.

## Вердикт

- UBCD ★★★★★
- Rescatux ★★★★★
- SysRescCd ★★★★★
- TRK ★★★★★
- Finnix ★★★★★

➤ Верный своему названию, UBCD превосходит других по инструментам восстановления и исправления.

Спасательные дистрибутивы

# Вердикт

**Е**ще раз напомним: наше Сравнение не оценивает дистрибутив просто по числу инструментов в нем. Тогда Ultimate Boot CD (UBCD) или Finnix, вероятно, превзошли бы всех. Но нашей целью было найти дистрибутив, пригодный для наибольшего количества людей независимо от их навыков администрирования и прошлого опыта.

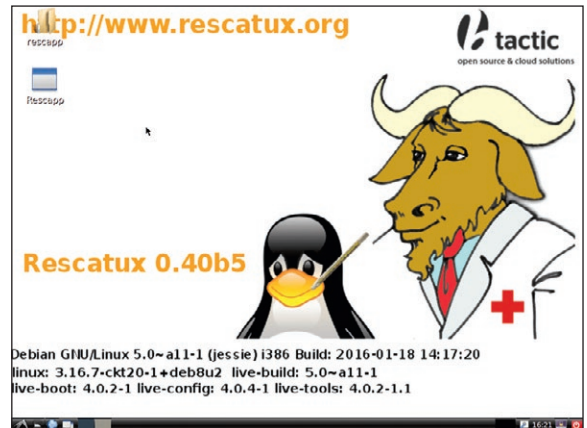
В этом отношении Finnix проигрывает, потому что предназначен для опытных пользователей и является исключительно дистрибутивом командной строки. С его помощью можно сделать многое, но только если вам точно известно, что надо делать. Подобным же образом, SystemRescueCd предназначен для людей с технической подготовкой, способных понять информацию в его меню загрузки и на сайте.

TRK проигрывает, потому что его будущее неопределенно. Уже некоторое время дистрибутив не обновлялся, но пока работает хорошо. Его индивидуальное меню,

хоть оно и текстовое, делает дистрибутив доступным для немалого числа людей.

И в итоге борьба сократилась до тесного соперничества между UBCD и Rescatux за первое место. И если у первого больше инструментов, то второй более дружелюбен к начинающим и неопытным пользователям. А если говорить о неопытных пользователях, то предлагать им сразу много опций — не лучшая идея, поскольку они просто могут выбрать неверный инструмент для работы.

Принимая во внимание все эти факторы, мы в нашем Сравнении должны присудить победу Rescatux, поскольку этот дистрибутив разработан для использования людьми, которые должны быть в состоянии использовать инструменты, не тратя много времени и не мороча головы излишком деталей. Победой Rescatux обязан своей чудесной утилите *Rescapp*, индивидуальным интерфейсам для инструментов командной строки и отлично иллюстрированной



встроенной и онлайн-документации. Возможно, Rescatux недостает столь же глубоких инструментов, как в некоторых других дистрибутивах, таких, как UBCD, но он отлично справляется с задачей ознакомления пользователей с имеющимися в нем инструментами. Кроме того, его разработчик довольно активен и отзывчив к запросам на расширение дистрибутива.

➤ **Может, он и не идеально красив с эстетической точки зрения, но будьте уверены, что Rescatux спасет вас и ваши данные.**

**Победой Rescatux обязан чудесной утилите Rescapp и интерфейсам.**

**I Rescatux** ★★★★★  
 Версия: 0.40b5 Сайт: <http://bit.ly/Rescatux> Лицензия: GNU GPL  
 » Самый дружелюбный к пользователю дистрибутив для выхода из проблемной ситуации.

**IV SystemRescueCd** ★★★★☆  
 Версия: 4.71 Сайт: [www.sysresccd.org](http://www.sysresccd.org) Лицензия: GNU GPL  
 » Хорош для опытных пользователей, способных переваривать документацию.

**II Ultimate Boot CD** ★★★★★  
 Версия: 5.3.5 Сайт: [www.ultimatebootcd.com](http://www.ultimatebootcd.com) Лицензия: GNU GPL  
 » Согласно своему названию, набит инструментами восстановления и исправления.

**V Finnix** ★★★★☆  
 Версия: 111 Сайт: [www.finnix.org](http://www.finnix.org) Лицензия: GPL и др.  
 » Проигрывает из-за отсутствия индивидуальных инструментов помощи пользователям.

**III Trinity Rescue Kit** ★★★★☆  
 Версия: 3.4 Сайт: [www.trinityhome.org/trk](http://www.trinityhome.org/trk) Лицензия: GNU GPL  
 » Чудесный дистрибутив, который пока работает, но имеет неопределенное будущее.

**Обратная связь**  
 Вам случилось терять сон из-за сбоев компьютера? Расскажите нам, как вы вышли из положения: [lxf.letters@futurenet.com](mailto:lxf.letters@futurenet.com).

## Рассмотрите также...

**П**омимо дистрибутивов, о которых мы рассказали в нашем Сравнении, восстановлением и исправлением занимается не так уж много других. Итальянский дистрибутив *PoliArch* с виду неплохо дополнил бы наш список. Увы, его домашняя страница и большая часть документации написаны на итальянском, что резко ограничивает пользовательскую базу. Другой опцией

будет взять liveCD для конкретного инструмента. Ряд популярных инструментов восстановления доступен на отлично настроенных минималистских liveCD. Если вам нужен только один такой инструмент, всегда можно обойтись версией liveCD вместо целого дистрибутива общего назначения.

Таким примером будет инструмент *Boot-Repair*, у которого есть liveCD, отлично подходящий для

восстановления и исправления разного рода проблем с загрузкой. Кроме самого *Boot-Repair*, дистрибутив также включает приложения *GParted* и *OS-Uninstaller*. *GParted* тоже выпустил свой liveCD, заодно содержащий утилиту *TestDisk*. Вы можете найти *TestDisk* в ряде других дистрибутивов, таких, как *Hiren Boot CD*, в котором есть несколько инструментов для исправления Windows. **LXF**



# Запуск!

Джонни Бидвеллу Linux мерещится повсюду — потому что так и есть. Наша любимая операционная система оказывается намного популярнее, чем предполагают скептики.



**П**ерсоналок с Linux немного — точную цифру назвать трудно, однако большинство оценок говорит о том, что эта цифра менее 3%. Использование Linux на Steam доходит до 1%. И это нормально: для Linux нет никакой крайности занимать верхние позиции в этих таблицах, его существованию не угрожает то, что людей — по необъяснимым причинам — по-прежнему тянет братья за *Microsoft Office*, чтобы правильно заполнить свои налоговые декларации. На самом деле, многим пользователям нравится быть в меньшинстве и даже поглядывать свысока на десктопные массы.

## Присмотревшись, вы обнаружите встроенный Linux в самых разных местах.

Однако настольный Linux — лишь побочный эффект беспрецедентного роста Linux в других областях. Для серверов и суперкомпьютеров надежность, возможности индивидуальной настройки и отсутствие цены делают Linux идеальным выбо-

ром, и здесь он господствует с конца девяностых и начала нулевых. Да еще, на минуточку, два миллиарда смартфонов Android. Однако Linux проник куда

глубже — присмотревшись внимательнее, вы обнаружите встроенный Linux в самых разных местах: от домашних роутеров, смарт-ТВ и плееров Blu-ray до электронных указателей, транспортных блоков управления и промышленных систем управления.

Поскольку Linux портирован на такое количество всевозможных архитектур CPU, это привлекательная опция для производителей. Альтернативой применению Linux является необходимость разработать (и поддерживать) специализированную ОС, что в лучшем случае будет повторным изобретением велосипеда, а в худшем — дорогим, рискованным и затратным по времени удовольствием.



# Linux везде

На Linux работает все, от роутеров до MP3-плееров и принтеров. Узнайте, как он туда попал и почему используется.

**К**огда столько производителей выбирают Linux, неудивительно, что появилась масса специализированных дистрибутивов, нацеленных на рынок встраиваемых устройств. Возможно, вы слышали о *DD-WRT* [см. Учебники, стр. 72 LXF198] и *OpenWRT*, которые можно установить на впечатляющее число роутеров. Таким образом пользователи могут получить более привлекательный интерфейс, добавочные функции, и, что самое важно, улучшить ситуацию с ленью производителей по части обновлений безопасности. *DD-WRT* появился вследствие прихоти LinkSys применять индивидуальные драйверы и компоненты Linux в своей серии роутеров *WRT54G*, не выпуская исходный код, как того требует GPL.

После нескольких жестко сформулированных электронных писем в LinkSys скомпилировали и открыли код. Первыми его ухватили в Sveasoft — и выпустили свою прошивку Alchemy и Talisman. Когда в Sveasoft решили брать деньги за «поддержку сообщества», *DD-WRT* ([www.dd-wrt.com](http://www.dd-wrt.com)) сделал ответвление последнего релиза и с тех самых пор предлагает его бесплатно.

Помимо роутеров, еще один успешный проект — Rockbox (<http://rockbox.org>), который приводит Linux на ваш MP3-плеер. Прошивки доступны для впечатляющего количества устройств: от дорогих iPod'ов до дешевых и забавных Sandisk Sansa. И опять-таки, они добавляют новые функции в виде: дополнительной или более надежной поддержки форматов файлов; улучшенной производительности; поддержки Unicode, а также возможности воспроизводить *Doom* (см. врезку *Он запускает Doom*, внизу). Rockbox появился в 2002 г. как продукт обратного инжиниринга прошивки (ошибочной) для Archos Player. Установка на все устройства потребует замены загрузчика. После этого (тут не обойдется без сложностей, поскольку продукты Apple требуют подписанного образа) можно установить Rockbox OS в основное хранилище устройства, что облегчит обновления и добавление дополнений.

Проект Yocto ([www.yoctoproject.org](http://www.yoctoproject.org)) существует для упрощения процесса внедрения Linux на встраиваемые устройства. Сам по себе Yocto не является дистрибутивом, он скорее направлен на переделку дистрибутивов под определенные устройства. Один из ключевых компонентов Yocto — инструмент компиляции *BitBake*. Его вдохновителем является *Portage*, менеджер пакетов Gentoo, и он компилирует индивидуальные образы из исходника с помощью вкусного *Recipes*. Yocto использует систему сборки *OpenEmbedded*, а та, в свою очередь, основана на технологии под названием *Buildroot*, которая все еще в активной разработке. *Buildroot* предлагается

в преднастроенном состоянии для готового оборудования, как, например, Raspberry Pi, и используется в *OpenWRT*.

Еще один проект, возникший из *OpenEmbedded* (и иже с ним) — *Angstrom Distribution*, целью которого является многогранность и удобство для пользователя. Если ему надо втиснуться в 4-МБ флэш-хранилища, он это сделает. Если ему надо быть больше похожим на полнофункциональный дистрибутив, он может и это. Он хорошо поддерживается на одноплатных компьютерах, включая мощный MinnowBoard от Intel.

В сердце Linux лежит библиотека *GNU C Library (glibc)*. Практически все пакеты на вашем компьютере зависят от нее, поскольку она содержит все низкоуровневые элементы, на которые опирается скомпилированный C/C++ код. В результате она огромна и, как правило, для встраиваемых технологий не годится. Однако ряд диетических вариантов *glibc* (включая *dietlibc*) в наши дни используются и во встраиваемых технологиях. Одна из самых распространен-



➤ Знаковый LinkSys WRT54G в 2003 г. дал старт движению самодельных прошивок.

## Появилась масса специализированных дистрибутивов, нацеленных на рынок встраиваемых устройств.

ных — *uclibc*, для работы ей даже не требуется устройства управления памятью, и потому она портируется на любые маломощные устройства, в частности, на микроконтроллеры. *Uclibc* не имела новых релизов с 2012 г., однако ее ветка, проект *uclibc-ng*, находится в активной разработке. Android использует разрабатываемую Google библиотеку *Bionic C*, которая опять же подходит для более мелких, менее мощных устройств. Но еще одной причиной ее разработки была изоляция проприетарных частей экосистемы Android от проблем копиленфта, возникающих при использовании ядра Linux. *Bionic* выходит под лицензией BSD, поэтому ее производные, которые, в силу малоубедительного обоснования, включают построенные на ней приложения Android, не обязаны предоставлять исходный код.

### Он запускает Doom

*PrBoom* — один из немногих исходных портов *Doom* (чей исходный код вышел в 1997 г.), доступных для Windows, Linux и множества других платформ. Он намного более способный и настраиваемый, чем оригинал, и при этом сохраняет совместимость с изначальными файлами WAD (поэтому можно использовать уровни и графику оригинала). Теперь, когда

Linux работает в столь неортодоксальных местах, сделалось своего рода штампом устанавливать на подобные устройства *PrBoom*.

Одним из наших любимых примеров является то, как Майкл Джордон [Michael Jordan] заставил его работать на LCD-экране принтера Canon Pixma. Брешь в системе обновлений безопасности прошивки

сделала этот прославленный прием возможным (<http://bit.ly/PixmaDoomed>).

Помимо этого, разные виды *Doom* работают на RockBox (хотя в монохроме выглядят довольно странно), графических калькуляторах, цифровых камерах... ах да, даже на осциллографе (правда, там работала Windows 95, так что мы этого не одобряем).



➤ *Doom* на крошечном MP3-плеере.

# Дорогая, я уменьшил Linux

Встраиваемый Linux — тоже Linux, но в целом это не тот вид, который работает на вашем ПК. Давайте посмотрим, в чем разница.

**Т**о, что вокруг так много устройств Linux, вовсе не означает, что вы можете запросто подключить к ним клавиатуру, сетевую кабель и монитор и начать вбивать туда команды *Bash*. Во-первых, все эти устройства просто не к чему подключать, во-вторых, встроенный Linux — это в первую очередь полный минимализм, и в ядре может и не быть поддержки клавиатур PS/2, кадровых буферов или сетевых пакетов.

Калибр оборудования тоже может сильно отличаться от типичного многоядерного ПК — большинство встраиваемых устройств основаны на архитектуре ARM или MIPS, и обычно на них установлен голый минимум ОЗУ, необходимого им для работы. Хранилище тоже может быть очень жестко ограничено: например, у многих домашних роутеров менее 2 МБ NVRAM, поэтому особо вдохновенные пользователи OpenWRT иногда вынуждены использовать

во-первых, все ломалось, а во-вторых, менеджер пакетов начинал рекомендовать переустановку всего удаленного плюс установку всякой ерунды. Причем эти два результата вовсе не взаимоисключающие. Но дело не в том, что данный подход плох, а всего лишь в том, что настольные пакеты имеют сложную паутину зависимостей, и ее нельзя нарушать. То, что можно удалить, зависит от того, с чего нам пришлось начать, и от того, что нам нужно: мы можем решить, что нам не нужны инструменты области пользователя для экзотических файловых систем, RAID и управления логическим разделом. Не забывая, что все это гипотетически, мы можем двигаться в этом направлении дальше. Многие дистрибутивы идут с Perl и Python, и уж эти оба вполне годны для сноса. Это высвобождает несколько сот мегабайт, и хотя мы с успехом можем продолжить упражняться в удалении пакетов, отдача будет невелика. Поэтому мы сменим тактику и внесем изменения, которые не одобрит бы менеджер пакетов.

## Цель — посмотреть, что можно поудалять, пока всё не откажет.

специальные минимизированные образы. Помните, как MS-DOS 5 поставлялась на трех флоппи-дисках? Ну так вот, оказывается, можно втиснуть ОС в небольшое пространство, если только «операции», которых ожидают от «системы», узко определены. Займемся-ка мы небольшим упражнением на сообразительность: как же нам урезать Linux?

Мы начнем с общей установки Linux без GUI: это может быть Arch или Debian или OpenELEC; неважно — у всех у них «отпечаток» после установки менее 1 Гб. Наша цель — посмотреть, что можно поудалять, пока все не откажет. Начнем с низко висящих плодов: вполне можно избавиться от ненужных пакетов. Многие пользователи испытывали соблазн сделать то же самое на своем настольном ПК, однако (в частности, на Ubuntu) получали два результата:

## Сгоняем вес

Документация — легкая добыча: при чистой установке */usr/share/doc* и */usr/share/man directories* вряд ли будут заполнены, но нам от них все равно проку нет. Подобным же образом можно избавиться от всех ненужных локалей *glibc*, ибо интернационализация занимает не самое высокое место в нашем списке приоритетов. Подобные удаления всегда можно отменить при обновлении пакетов, однако притворимся, что с нашим устройством его не происходит. Вообще-то, если нам хватит смелости, мы вполне обойдемся и без управления пакетами. Хорошо это или плохо, но многие встраиваемые устройства предполагают, что их ОС никогда не увидит никаких программных обновлений или заплаток безопасности. Для устройств, которые не будут соединены с Сетью и не предполагают произвольного ввода информации пользователем, это не проблема; а вот для новомодных гаджетов Интернета Вещей — весьма прискорбно. Но будь что будет: продолжим удалять скачанные пакеты из */var/cache*. Беспочвенные нападки на *Systemd*,

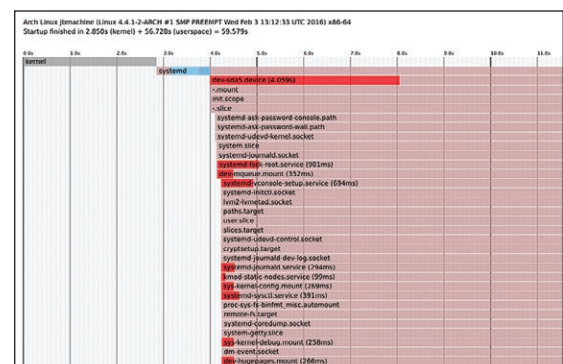
## Быстрая загрузка

Для определенных систем быстрая загрузка крайне важна — представьте, например, что вам надо ждать 20 секунд, пока ваш дефибриллятор просто сообщит «идет зарядка». Пользователи настольных ПК избалованы улучшенным за долгие годы управлением питанием, поэтому проблемы с медленным запуском можно обойти с помощью режима ожидания программ, из которого компьютер может выйти почти мгновенно.

К сожалению, подобная роскошь не для встраиваемых устройств — даже если бы они умели засыпать, они все равно, вероятно, будут потреблять больше энергии, чем может позволить себе их окружение. Поэтому ведется большая работа по ускорению загрузки, и плодами данного труда

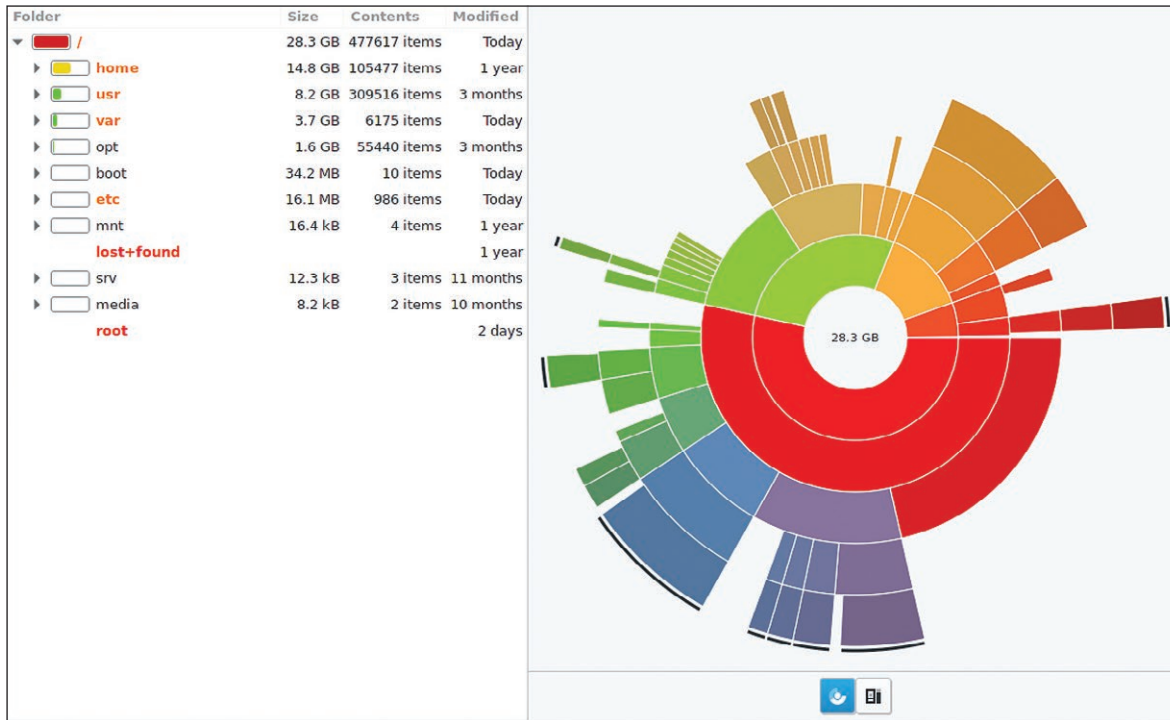
является то, что при правильных оборудовании и оптимизации Linux вполне может загружаться менее чем за секунду. Здесь можно кое-что добиться легко (удаляя неиспользуемые функции, не проверяя образ ядра, приглушая вывод информации о загрузке), чего-то — аналогично действиям тех, кто разгоняет загрузку настольного ПК (используя *bootchartd* или *system-analyze* для определения медленно запускаемых сервисов); а кое-что требует больших усилий и немалых изысканий (например, написать специальный загрузчик для своего оборудования).

Ян Альтенберг [Jan Altenberg] из Linutronix выложил чудесное видео здесь: <http://bit.ly/BootOneSecond>.



Systemd-analyze умеет создавать такие вот ценные графики загрузки, чтобы вы могли тратить часы, ускоряя загрузку на доли секунды.





» Наша установка Arch Linux занимает 28,3 ГБ. Подобное расточительство неприемлемо в мире встраиваемого Linux.

похоже, никогда не выйдут из моды, но для встраиваемых устройств он и вправду чрезмерный. Нам в любом случае потребуется система `init` и в крайнем случае может понадобиться менеджер сервисов, однако все остальные действия `Systemd` нам ни к чему. Так что мы можем втиснуть сюда самопальный `init` и высвободить 30 МБ за счет `Systemd`. Мы могли бы пойти дальше и заменить систему `init`, `udev`, `Bash` и основные утилиты Linux на `Busybox`. А то и еще дальше — избавиться и от привычного `init`, и получить скрипт, загружаемый прямо в основное приложение устройства. К этой стадии наш Франкен-Linux уже мало похож на оригинал, и мы можем взять его прямо за горло и начать возиться с ядром. Опять же — это гипотетически — нам, вероятно, незачем устанавливать на свою ОС все инструменты компилятора, поэтому наше оптимизированное ядро должно быть создано и (кросс)скомпилировано на другом компьютере.

## Отделка ядра

Большинство дистрибутивов упаковывают свое ядро всеми поддерживаемыми драйверами, скомпилированными в виде модулей — при обнаружении оборудования загружается соответствующий модуль. Но, зная, что мы не намерены менять свое оборудование, преспокойно можно избавиться от ненужных драйверов. Пакеты `linux` и `linux-firmware` вместе занимают около 200 МБ. Естественно, ядро нам нужно, плюс может понадобиться пара файлов прошивки, но большая часть этих пакетов для нас бесполезна. Итак, нашей первой целью становится компиляция индивидуального ядра, которое будет использовать только нужные нам драйверы. Мы даже можем вкомпилировать требуемые драйверы в ядро, чтобы улучшить время загрузки. Убедившись, что это работает, можно удалять пакеты; и мы освободили 150 МБ из своей установки. Можно и еще урезать ядро, если нам не нужны сетевые функции.

Следуя этим указаниям, можно в итоге получить ОС объемом около 500 МБ, что уже неплохо; но это не предел. Было бы упущением с нашей стороны не сказать о Damn Small Linux (который в настоящее время пребывает в спячке) и о Tiny Core Linux, которые

благополучно втиснули полноценный Linux в одну десятую этого места, однако они жульнически распаковывают программы в ОЗУ, а это для многих устройств не годится, так что давайте продолжим.

Повозившись с пользовательской областью, системой `init` и ядром, мы теперь добрались до загрузчика. `Grub` весьма интересен, но ему нет места во встраиваемом мире. Здесь предпочтительным инструментом является `Das U-Boot`. Он создан для обеспечения скорости и переносимости, и портирован на многие устройства. В отличие от ПК x86, где о том, чтобы привести оборудование в должное состояние, заботится BIOS (а позднее — прошивка UEFI), на встраиваемые устройства обычно первой загружается программа `U-Boot`. И как таковая, она подвержена еще большим ограничениям, нередко будучи вынужденной втискиваться во флэш-хранилище, которое иногда составляет всего лишь 128 КБ.

Однако у `U-Boot` есть то преимущество (по крайней мере, на архитектуре ARM), что тягомотина с определением оборудования полностью игнорируется. Причина в том, что `U-Boot` дополняется так называемым Device Tree [Дерево устройств], который весьма удобно сообщает ОС обо всем оборудовании, присоединенном к системе, и о требуемых драйверах. На настольном ПК, где любой компонент можно спокойно заменить (обычно тем, чему требуется совершенно другой драйвер) или вообще убрать, подобное бессмысленно. По сути, определение и перечисление оборудования в Linux является слегка недетерминистским:

жесткие диски определяются в порядке подачи к ним питания, и то, что было `/dev/sda` при одном запуске, вполне может стать `/dev/sdb` при следующем. То же применимо к сетевым интерфейсам, вот почему мы сегодня используем UUIDS разделов и постоянные имена устройств. Отсутствие заботы обо всей этой неопределенности значительно облегчает существование встраиваемых систем. »

**Как встроить Linux на планшет, см. на стр. 58**



# Дроны, дроны повсюду

Одно из самых интересных и популярных применений встраиваемого Linux сегодня — разработка любительских дронов.

**К**огда-то упоминание о беспилотном летательном аппарате (БПЛА) в основном относилось к работе спецподразделения ЦРУ на Ближнем Востоке. Сегодня они становятся частью нашей повседневной жизни. Возможно, один из них нарушил привычное спокойствие вашей воскресной прогулки, или, возможно, вы слышали, что они использовались для доставки контрабандных «посылок» в тюрьмы. Однако у них есть и иное применение: в некоторых частях Китая курьеры используют дроны, чтобы доставить ваш заказ до ближайшего магазина на углу. Вам сообщат точное время и место его прибытия, что всяко лучше тамошних сетей доставки (иногда они не в состоянии позволить в звонок).

Из-за странных правил допуска к отправке в небо мы вряд ли увидим в ближайшее время подобный способ доставки в Великобритании. Но благодаря росту спроса на развлекательные БПЛА мы определенно увидим больше дешевых дронов на рынке,

а вскоре после этого — и в небе. В ваших местных магазинах полно готовых дронов. Есть множество моделей мини-дронов, однако давайте ограничимся определенными устройствами, т.е. предназначенными для использования на улице, заряда батарей которых хватает больше чем на пару секунд. В этой лиге квадрокоптер начального уровня опустошит ваш карман на £400, тогда как высокотехнологичное устройство может обойтись раз в десять дороже. Здесь вы найдете поистине потрясающие летательные аппараты с камерами 1080p, закрепленными на шарнирах и способными вращаться на 360° по трем осям, и все это во время передачи на базу потокового видео (хотя и сильно сжатого) почти в реальном времени. Подобное создание может находиться в воздухе до 30 минут, прежде чем возвращаться на базу, что оно делает по собственному усмотрению благодаря технологии автопилота. Перед взлетом таким дронам можно даже запрограммировать курс, указав несколько точек GPS.



➤ Дрон 3DR Solo — один из самых чудесных на рынке. Помимо автопилота Pixhawk 2, ему добавляет ума вспомогательная плата Cortex A9 на Linux.

## Проект Dronocode

В октябре 2014 г. под эгидой Linux Foundation и в сотрудничестве с индустриальными лидерами 3D Robotics и Yuneec был введен в действие некоммерческий проект Dronocode. Это совместный труд с индустриальными партнерами по созданию платформы с открытым кодом с распределенным доступом для содействия основанному на Linux ПО для БПЛА.

Целеустремленные создатели ПО для воздухоплавания могут получить вознаграждение за свое участие в разработке платформы

в меритократичной манере. На данный момент над проектами, связанными с Dronocode, трудятся более 1200 разработчиков.

Цель Dronocode — стандартизация протоколов (типа MAVLink), кодов полета (таких, как APM и PX4) и ПО наземного контроля (QgroundControl, APM Planner и т.д.) ради расширения и улучшения доступа к экосистеме дронов на основе Linux. По мере увеличения мощности бортового оборудования аппарата Dronocode стремится подстегнуть разработку таких расширенных функций, как потоковое

видео (Pi 2 значительно улучшил данную ситуацию), предупреждение столкновений и компьютерное зрение.

Ждем не дождемся появления этих функций, особенно если реализуют недавние предложения Национальной полиции Голландии. Они предложили использовать дрессированных орлов (нет, серьезно: <http://bit.ly/EagleSquad>) для нейтрализации дронов со сбойми, и метеорологи уже выразили заинтересованность в вооружении собственным призывом пернатых стражей.



## Воспарить с Linux

Удерживать объект в небе нелегко, особенно на ветру. Чтобы оставаться в воздухе, дрон должен уметь мигом реагировать на изменения летных условий. Стоит ему определить нежелательный поворот относительно заданной оси, соответствующий мотор должен немедленно ускориться, чтобы выровнять аппарат, пока он не рухнул с неба. Подобные корректировки могут возникать несколько тысяч раз в секунду, и они управляются диспетчером полетов дрона.

Диспетчер полета оснащен сенсорами, требуемыми для этой задачи — как минимум это гироскоп, а на более продвинутых моделях еще и акселерометр. Там может также быть барометр, GPS, компас и датчики расстояния (на основе ультразвуковых импульсов или лазерного дальномера [lidar]). Однако нам требуется нечто способное переваривать всю эту информацию, так что в сердце авиадиспетчера находится CPU. Поскольку основным показателем является заряд батарей, традиционно это 8-битный чип на основе Arduino (или близких эквивалентов), которые в плане расширяемости ничего особенного предложить не могут.

Один из самых первых примеров — ArduPilot Mega (APM), который появился в 2007 г. и располагал 8-МГц CPU с 8-КБ ОЗУ и 256-КБ флэш-хранилищем. Оборудование APM объединено с ПО Mission Planner, которое предлагает все необходимое наземной станции управления, включая возможность прокладывать курс с помощью Google Maps.

Сейчас всё это меняется, и мы видим более сложные контроллеры, способные работать с большим числом сенсоров и осуществлять более сложную навигацию. Один из них — 3D Robotics Pixhawk, работающий на 32-битном ARM Cortex M4 и на ОС реального времени NuttX.

Более мощные одноплатные компьютеры, вроде Raspberry Pi, стали повсеместными — и тоже проложили себе путь в небо. Обычное ядро Linux никогда не предназначалось для приложений реального времени. Однако спрос на них рос, причем в самых разных областях, от производства аудио до промышленной нарезки на станках с ЧПУ. В результате появился широко используемый патч PREEMPT\_RT, который перерабатывает большие секторы ядра, превращая его в источник выгружаемой мощности с низкой латентностью. Прошивка APM, ранее доступная только для устройств

Arduino, была успешно портирована на Linux. Так что теперь наша любимая ОС может служить в авиации.

Это подход применен в автопилоте Erle-Brain 2, который соединяет Raspberry Pi 2 с платой Pixhawk Fire Cape 2.0 (PXF 2.0), и все это тщательно упаковано в защищенный от вибрации корпус в компактном форм-факторе. Плата PXF имеет полностью открытую структуру и обеспечивает все необходимые сенсоры, а также выводит соединения на шину I2C на случай, если вы захотите добавить больше. Она может контролировать до 12 выводов (моторы/роторы/рули/камеры) через PWM и имеет 4 ЖК-диода для отображения рудиментарной диагностической информации. Производители предоставили основанный на Debian образ (с заплатками для реального времени), а также более современную версию на Ubuntu Snappy Core. Snappy — легковесная ОС, разработанная для устройств IoT [Интернет Вещей] и облачных установок. Ее отличие от традиционной методологии обновлений Linux, основанной на пакетах — транзакционные обновления, когда базовая система обновляется как

Создадим брелок потокового вещания, см. стр. 68

## Чтобы оставаться в небе, дрон должен мигом реагировать на изменения летных условий.

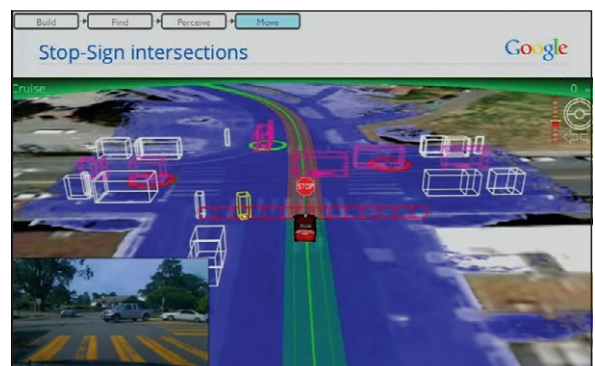
единое целое. Если с этим обновлением что-то пойдет не так, легко вернуться к предыдущему образу. Snappy также поддерживает приложения ('snaps'), которые можно скачать в магазине Erle Robotics (<http://erlerobotics.com/blog/snappy-store>). Erle-Brain можно приобрести отдельно или как часть набора дрона компании. Плата PXF 2.0 отдельно недоступна, однако обладатели умелых рук могут заинтересоваться последним предложением Erle — PXFmini. Это щиток для более новых Raspberry Pi, в частности, Zero, который втиснул все необходимые сенсоры и коннекторы в 15-граммовую, полностью APM-совместимую плату. Navio2 Autopilot Shield делает то же самое, но содержит еще и модуль GPS (который умеет работать также и с российской сетью Глонасс и китайской Бэй-Дуо) и модуль питания.

»

## Внутри автомобиля на Ubuntu

Беспилотные автомобили уже тестируются на дорогах, и хотя действующие законы запрещают им покидать гараж без надзирающего за ними «человечешки», но можно ожидать, что более юное поколение наших читателей увидит снятие этого ограничения. Эта технология уже невероятно передовая, она включает использование лидара, камер, работающих во всех направлениях, нейронных сетей из GPU, облака и зловещей магии. Иными словами, на данный момент целью для машин является возможность независимой работы в простых ситуациях на шоссе, а не на нерегулируемых перекрестках и непонятных дорогах в какой-нибудь «дыре». Доминирующее в прессе мнение уходит корнями в глубокие карманы Google, Tesla и, в последнее время, Nvidia. Однако перед Рождеством 2015 г. эта ситуация несколько изменилась. Произошло это благодаря некому Джорджу Хотцу [George Hotz], который в прошлой жизни был geohot — хакером, взломавшим первый iPhone и Playstation 3. Хотц объявил, что модифицировал свою Honda Acura домашним автопилотом, который работает при помощи

водителя. И вся эта магия базируется на нашем любимом Ubuntu Linux. Хотц надеется через свою компанию, comma.ai, продавать свою систему автопромышленности за \$1000, то есть преодолеть текущую монополию Mobileye (партнера многих автопроизводителей, предлагающего автопилоты Tesla). Хотя система выглядит неказисто, поскольку по всему салону размещаются провода и множество камер, в бардачке куча всяких мелочей, и все это дополняется пурпурной жутью темы по умолчанию Ubuntu, Хотц утверждает, что она невероятно прогрессивна: все, что ей известно об управлении автомобилем, это результат самообучения, а не введенные человеком правила и инструкции.



Целый флот робокаров Google тоже использует Linux. Другие транспортные средства и пешеходы показываются в виде квадратов — пока что автомобили не нацеливаются на эти квадраты сознательно.



# Точка доступа Arch Wi-Fi

Всеми фибрами прочувствуйте встраиваемый Linux, создав собственную точку доступа на Pi.

**В** заключение нашей статьи мы проведем вас по пути создания собственного устройства с псевдовстроенным Linux, а именно: беспроводной точки доступа на Raspberry Pi. Pi понадобится проводное соединение с роутером; идеальным в данном случае будет соединить Pi и роутер через соединение Powerline. Тогда мы получим беспроводной сигнал в непосредственной близости от адаптера Powerline, куда, вероятно, не доходит сигнал вашего роутера. Далее мы предположим, что ваш роутер предоставляет сервисы DNS и DHCP в вашу локальную сеть — он почти наверняка это делает, если вы находитесь в Великобритании или если у вас уже имеется несколько устройств, присоединенных к нему. Тем, у кого соединение через проводной модем, не повезло, поскольку мы применим простой сетевой мост, который использует эти сервисы с минимумом хлопот. Вам также понадобится беспроводной адаптер, способный перейти в режим Access Point (AP) — того, что нормально работает (т.е. соединен с вашей домашней

сетью) с Pi, может быть недостаточно. Вы можете проверить это, подключив адаптер к системе Linux и запустив

```
$ iw list
```

Если в списке поддерживаемых интерфейсов появляется AP, всё хорошо. Если нет, то подумайте о приобретении нового адаптера. Адаптер с большой антенной сильно увеличит диапазон вашей точки доступа, а информацию по совместимости вы найдете на <http://elinux.org/RPI-Wireless-Hotspot> или поискав в Сети по номеру модели. Мы бы не советовали применять чипы Realtek 8188EU или 8188CU (используемые во многих бюджетных беспроводных адаптерах). Их можно перевести в режим AP, но вам придется скомпилировать драйвер вне ядра и использовать соответствующую взломанную версию *hostapd* от Realtek. Подробности см. на <http://bit.ly/RTL8188AccessPointOnPi>.

Даже если у вас нет Raspberry Pi, всё равно можете поразвлекаться — программа *hostapd*, которой мы воспользуемся, имеется во всех репозиториях, так что подойдет любой компьютер, где работает Linux, хотя настройка сети будет отличаться. Для пущего разнообразия мы будем использовать ARM-порт Arch Linux для Pi. Raspbian уделяется немало внимания, но всегда полезно посмотреть, что еще у нас есть. В духе встраиваемых систем, изначальная установка обеспечивает только самый минимум. Кроме того, как свойственно встраиваемым системам, установка отчасти нетривиальна (см. инструкции во врезке *Настройка Hostapd*, внизу — следуйте им).

## Arch на Raspberry Pi

Вам понадобится SD-карта, на которой не жалко затереть все данные; 2 ГБ вполне хватит. Мы подготовим носитель на другом компьютере с Linux, используя стандартные инструменты. Установите SD-карту и найдите имя устройства — это будет нечто вроде `/dev/sdc` или `/dev/mmcbk0`; мы будем именовать его `/dev/sdX`. Убедитесь, что оно определилось правильно: если вы ошибетесь, то вполне можете уничтожить свой жесткий диск. Все эти команды должны выполняться от имени root, поэтому используйте `su` или `sudo -i`, чтобы повысить свои привилегии. Затем запустите *fdisk*:

```
# fdisk /dev/sdX
```

Введите `p` и нажмите `enter`, чтобы увидеть список разделов. Если это выглядит, как ваш жесткий диск (размер столбцов должен вам подсказать) — полный назад! В ином случае введите `o`, чтобы стереть все разделы. Потом введите `p`, чтобы создать новый раздел, введите `p`, чтобы выбрать первичный раздел, и введите `1`,



➤ Почему бы не добавить дисплей (например, взять Display-O-Tron 3000) к вашему роутеру, чтобы показывать число соединенных хостов и сетевые условия.

## Настройка hostapd

Настройка точки доступа освежающе проста. Отредактируйте `/etc/hostapd/hostapd.conf` и просто заполните файл:

```
interface=wlan0
driver=nl80211
bridge=br0
ssid=LXF Wireless
hw_mode=g
```

```
channel=11
auth_algs=3
wpa=2
wpa_passphrase=acupoftea
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
wmm_enabled=1
```

```
ieee80211n=1
```

Это позволит вам настроить точку доступа WPA2, и вы сможете перейти к соединению точки доступа с нашим проводным интерфейсом через мост. `ssid`, `channel` и `wpa_passphrase` настройте по своему усмотрению. Последние две строки включают более скоростные соединения 802.11n (если позволяет аппаратура).

чтобы сделать его первым разделом (нашим загрузочным), нажмите Enter, чтобы принять первый сектор по умолчанию (обычно 2048) и затем введите +100M, чтобы сделать этот раздел 100-МБ. И, наконец, нажмите w, чтобы записать таблицу и выйти из *fdisk*.

Теперь создадим и смонтируем файловую систему на карте:

```
# mkfs.vfat /dev/sdX1
# mkfs.ext4 /dev/sdX2
# mkdir /mnt/sdboot
# mkdir /mnt/sdroot
# mount /dev/sdX1 /mnt/sdboot
# mount /dev/sdX2 /mnt/sdroot
```

Тут нам понадобится образ ОС. Версии для Pi и Pi 2 различаются, и если вы — владелец первого, то вам повезло: образ есть на **LXFDVD**. И вы можете скопировать его в текущую директорию следующим образом:

```
# cp /run/media/LXFDVD209/ArchHotspot/ArchLinuxARMrpi-latest.tar.gz
```

Если у вас Pi 2, не переживайте — образ всего 300 МБ, и добыается он так:

```
# wget http://archlinuxalarm.org/os/ArchLinuxARM-rpi2-latest.tar.gz
```

Далее, распакуем файловую систему root на SD-карту — не забудьте установить стратегическое 2, если у вас Pi 2:

```
# bsdtar -xpf ArchLinuxARM-rpi-latest.tar.gz -C /mnt/sdroot
# sync
```

Последняя команда обеспечивает запись всех кэшированных данных на SD-карту, чтобы потом не плакать. Затем надо переместить соответствующие файлы из раздела root в раздел загрузки:

```
# mv /mnt/sdroot/boot/* /mnt/sdboot
```

Наконец, остается размонтировать разделы SD-карты и удалить временные точки монтирования, которые мы создали:

```
# umount /mnt/sdboot /mnt/sdroot
# rm -rf /mnt/sdboot /mnt/sdroot
```

Итак, Arch Linux установлен; мы можем продолжить и протестировать его на нашем Pi. В этот момент понадобятся дополнительный монитор и клавиатура: в отличие от Raspbian, Pi не соединен с сетью по умолчанию.

Установите свою свеженькую SD-карту с Arch в Pi и присоедините кабель Ethernet и беспроводной адаптер. При загрузке Pi на несколько секунд покажет вам рисунок в радужных тонах, а затем должен появиться *Systemd* и представить вам экран приглашения. Войдите от имени root с паролем root; неплохо было бы сразу же поменять его с помощью команды *passwd*. У вас должно установиться быстрое, но некачественное проводное соединение благодаря

```
# dhcpcd eth0
```

После этого обновите систему с помощью

```
# pacman -Syu
```

и установите требуемые пакеты:

```
# pacman -S hostapd bridge-utils iw
```

Базовая установка настроена на получение IP-адреса по DHCP через проводное соединение, как только оно подключено, однако это не идеально для нашего сценария — нам нужно раздобыть статический IP-адрес, чтобы легко войти через ssh, если будут проблемы. Мы воспользуемся программой *netctl*, чтобы создать профиль для простого сетевого моста, соединяющего проводной и беспроводной интерфейсы на Pi. Данная структура является самой простой по концепции, поскольку все устройства, соединенные беспроводным образом с Pi, в конечном итоге окажутся в той же самой подсети, что и устройства, соединенные напрямую с вашим роутером. А следовательно, никакой необходимости возиться с *iptables*, переадресацией пакетов или маскардингом DNS. Сначала запустите *# ip*, чтобы выяснить имена ваших проводных и беспроводных интерфейсов. У нас они именуются

*eth0* и *wlan0* соответственно — не забудьте их переименовать, если у вас они другие. Для создания и редактирования профиля запустите

```
# nano /etc/netctl/bridge
```

и заполните файл такими настройками:

```
Description="LXF Bridge connection"
Interface=br0
Connection=bridge
BindsToInterfaces=(eth0)
IP=static
Address='192.168.0.100/24'
Gateway='192.168.0.254'
```

после чего выйдите из системы с помощью *Ctrl+x*, при выходе сохранив файл.

Здесь мы предположили, что ваш роутер выдает IP-адреса в виде *192.168.0.xx*; информация от *ip* а сообщит вам, так ли это. Если нет, используйте в файле другие цифры; разумно настроить ваш роутер так, чтобы он избегал присваивать адрес, который вы дали своему Raspberry Pi (здесь — *192.168.0.100*) через DHCP, однако это выходит за рамки данной статьи. (Следуйте инструкциям во врезке *Настройка Hostapd*.)

## Ваше устройство с псевдо-встроенным Linux — точка доступа на Raspberry Pi.

Прежде чем проверять, работает ли это, надо включить наш сетевой мост и сервис *hostapd*, чтобы они запускались автоматически:

```
# systemctl enable netctl@bridge hostapd
```

Теперь перезагрузитесь и перекреститесь: вы должны увидеть точку доступа “LXF Wireless” с любого устройства с беспроводной картой. Более того, вы должны быть в состоянии соединиться с ней, используя пароль *асирфтеа*. В случае проблем попробуйте запустить *hostapd* в режиме отладки:

```
# hostapd -dd /etc/hostapd/hostapd.conf
```

Этот проект очень легко расширить в разных направлениях: те, кто озабочен своей конфиденциальностью, могут захотеть включить сюда поддержку Tor или VPN; клиенты, недовольные своим интернет-провайдером, могут решить автоматически отправлять твит при каждом падении скорости их соединения (см. код *AlexseyP* на <http://pastebin.com/WMEh802V>); или вы можете добавить жесткий диск, чтобы он мог обслуживать файлы. Ни в чем себе не отказывайте. **LXF**

```
alarm 35.8 KiB 3.49M/s 00:00 [#####] 100%
aur is up to date
:: Starting full system upgrade...
resolving dependencies...
looking for conflicting packages...

Packages (24) ca-certificates-mozilla-3.22-1 coreutils-8.25-1 curl-7.47.1-1
dialog 1:1.3 20160209 1 gcc 5.3.0 4 gcc-libs 5.3.0 4 git 2.7.1 1
gnutls-3.4.9-1 libcap-2.25-1 libcrypto-1.6.5-1 libsystemd-228-4
linux-raspberrypi-4.1.17-3 linux-raspberrypi-headers-4.1.17-3
nano-2.5.2-1 nlp-4.2.8.p6-3 pacman-mirror-list-20160207-1
python-packaging-16.2-1 python-pyparsing-2.1.0-1
python-setuptools-1:20.1.1-1
raspberrypi-firmware-bootloader-20160209-1
raspberrypi-firmware-bootloader-x-20160209-1
raspberrypi-firmware-tools-20160209-1 systemd-228-4
systemd-sysvcompat 228 4

Total Download Size: 80.33 MiB
Total Installed Size: 288.58 MiB
Net Upgrade Size: 1.22 MiB

:: Proceed with installation? [Y/n]
```

➤ Менеджер пакетов *pacman* можно использовать для обновления своей точки доступа к Wi-Fi, работающей на Arch Linux и Raspberry Pi.



**Джонни Бидвелл** узнает всё  
о новом языке, выпущенном  
Mozilla, от Rust'амана со стажем —  
Джима Блэнди.

# Алмазы и Rust







Джим Блэнди (Jim Blandy, он же jimbl) приложил руку к *Emacs*, *GNU Project Debugger (gdb)* и другим проектам GNU. Он основал компанию Red Bean, которой было не суждено обогатиться, но не суждено и прогореть.

Ныне он — разработчик программ для Mozilla и ярый апологет языка Rust [rust — *англ.* ржавчина]. Мы пересеклись с ним на OSCON, чтобы узнать об этом новом языке, а также о причудах старых. А также проблемах в обучении нового поколения растюльбов.

**Linux Format:** Итак, только что [интервью состоялось в июне 2015 г.] вышел Rust 1.0. И для начала вот такой прицельный вопрос: все эти новые языки от ключевых игроков — Go, Swift, Rust — зачем они нам?

**Джим Блэнди:** Отличный вопрос. Особенно когда ты, выучив несколько языков программирования, вдруг понимаешь, что в общем-то все они одинаковы. Поняв суть одного, куда быстрее освоить другие. Потому-то языки вроде Haskell и доставляют столько удовольствия — они не так легко учатся. Prolog тоже неплох, с ним можно выполнять программы задом наперед.

**LXF:** Даже я, математик, не особо верю, что функциональное программирование работает. Так и не понимаю монады.

**ДБ:** Это просто моноиды в категории эндоморфизмов.

**LXF:** Ах, вот оно что... Спасибо, это всё объясняет. Давайте вернемся к Rust.

**ДБ:** Определяющими чертами C и C++ является их отношение к неопределенному поведению. В Python и ему подобных, при обращении к элементу, выходящему за пределы массива, выдается исключение. Исключение это описано в документации, где говорится, что будет вот так-то и так-то. То есть даже если вы навредили, в самом языке прописана реакция на это.

Почти то же самое с JavaScript. В каком-то смысле, такие языки пытаются быть всеобъемлющими: любая программа, которую вы им дадите, обретает некое значение — она может быть и бесполезной, и просто выдавать ошибку — но свое значение есть у всего. В языках C и C++ вам скажут, что есть ошибки, которые можно отследить сразу, или во время компиляции, и показано, как. Но со всем, что требует чуть больше ресурсов для обнаружения, вам придется разбираться самим». По факту же, если ваша программа выполняет какое-то нештатное действие, компилятор может создать всё что угодно.

Поэтому свой доклад я начал с запуска демо, где простая трехстрочная программа на C объявляет массив и один его элемент, присваивает значение третьему элементу (которого нет) и выходит. При выполнении выдается странное сообщение об ошибке, где говорится, что ваш пароль общедоступен и всё такое. А на самом деле, программа затоптала адрес возврата для `main()`, так что `main()`



при завершении попадает в какую-то мелкую библиотеку C и просто падает.

**LXF:** И это не какая-нибудь ошибка в компиляторе или вроде того?

**ДБ:** Не-а. С точки зрения C и C++, все абсолютно логично; просто там изначально предполагается, что избегать неопределенностей — это задача программиста. За 30 лет мы насмотрелись, насколько это удастся. В 1988 году вирус Morris, используя переполненный буфер, проникнул в компьютеры через протокол Finger. С тех пор шел постоянный поток таких эксплоитов. В базе уязвимостей Open Source есть небольшая диаграмма уязвимостей, так вот там они всё время порядка 10%. Теперь появились

машин для других языков. И все эти системные языки небезопасны и заставляют вас делать такие вещи, которые людям в принципе не под силу.

Суть любого языка можно четко определить: Python, JavaScript, Haskell, Ruby, как и все остальные, стремятся к полноте. И получается этакая странная дихотомия, что те самые языки, которым мы доверяем столько недостоверных данных, принуждают нас исполнять танец с саблями на льду. Rust перекидывает мост через эту пропасть. Этот язык программирования предупреждает вас, когда вы нарушаете правила. В Rust, если я заявляю структуру с двумя 32-битными целыми числами, то это будет 64-битное значение и ничто другое. Только два слова, никаких метаданных или чего-то

## ПРО 30 ЛЕТ ПЕРЕПОЛНЕНИЙ БУФЕРА

# Суд постановил — эксперимент доказал, что люди такой код писать не могут, им доверять нельзя.

SQL и PHP-инъекции, то есть конкуренция высока, но эксплоиты держатся, и это неудивительно: суд постановил — эксперимент доказал, что люди такой код писать не могут, им доверять нельзя.

В блоге безопасности Google недавно был пост о целочисленных уязвимостях. Даже невинные вещи, вроде: «Ну, я просто вкину это 32-битное целое число в 16-битное, зачем терять время, извлекая 16 младших бит, я-то знаю, что делаю». Не знаете! Ну, то есть могли бы знать, но не знаете настолько часто, что есть чем поживиться русской мафии. И это плохо.

В результате получается, что языки системного программирования надежны на низком уровне, в плане ядра, шифрования и создания виртуальных

там динамического, а просто структура данных. Мы добавили программу сборки мусора для Servo (проект для портирования Firefox в Rust), но она не является частью самого языка. Создавая программу в Rust, вы точно знаете, когда освобождается каждая переменная, и не нужно ждать сборщика мусора.

Таким образом, в управлении памятью всё четко и под контролем, представление данных простое и недвусмысленное — в точности такое, как у машины для этих величин, а операции, выражаемые лаконичным кодом, действительно лаконичны. Когда в C++ при операции присваивания присваивается вектор, это означает его копирование в назначенное место. А если это у вас вектор строк, »

то копировать надо каждую строку. В итоге код может выйти невероятно громоздким, отводящим огромный объем памяти. Обычно это не проблема, но вы всё же не этого ждете от языка системного программирования, ведь его главный козырь — то, что он дает вам контроль над машиной.

## **LXF:** А чем в этом плане отличается Rust?

**ДБ:** В Rust к таким вещам подход другой. Здесь для тяжеловесных величин используются ходы: операция присваивания переносит значение из источника в пункт назначения, а сам источник деинициализируется. В результате, если вы имеете дело с крупной структурой, вроде вектора или хэш-таблицы, в любой момент времени у данного значения есть только один владелец. Вы можете передать его функции, тогда оно будет принадлежать ей, а вам, как инициатору вызова, оно будет уже недоступно: вы изменили владельца, а он может быть только один. Вы можете взять это большое значение и сохранить его в другую таблицу, теперь она будет им владеть, а у вас, опять же, доступа не будет. Если владелец только один, всегда

понятно, куда именно уходит значение. В чем и состоит управление памятью в Rust.

Конечно, то, что владелец может быть только один — очень существенное ограничение, поэтому-то и появляются программы с неоднозначной идентификацией принадлежности. Поэтому в Rust есть так называемые «заимствованные указатели [borrowed pointers]», позволяющие вам временно чем-то попользоваться, но затем вернуть владельцу. Такой указатель выдает доступ к значению без передачи права владения им. Вы можете строить на нем вычисления или изменять его, но должны выдавать свои временные ссылки так, чтобы это было понятно компилятору. Компилятор должен уметь точно распознать, когда прекращают действовать ваши заимствованные указатели. Итак, их два типа: разделяемые ссылки [shared references], коих у вас может быть миллионы, и вы можете передавать их кому угодно, но по возвращении все они перестают действовать. Плюс изменяемое заимствование [mutable borrow] — вам разрешается только одно, по схеме «несколько читателей — один писатель [multiple

reader single writer pattern]». Только через него возможен доступ к объекту, вы не сможете даже вызвать ее по исходному имени переменной. Изменяемое заимствование становится единственной точкой доступа к соответствующему объекту.

За счет жесткого разделения совместного и изменяемого доступа, Rust способен во время компиляции исключить применение обоих одновременно. Почитайте спецификацию Java и взгляните в интерфейс хэш-таблицы: там правило гласит, что при переборе ее элементов, менять хэш-таблицу может только сам итератор. Если это делает кто-то другой, ваш итератор выдает исключение.

В классическом стиле C++ при изменении хэш-таблицы, где есть итераторы, и все они нерабочие, это означает неопределенное поведение — о чем вас и оповещают, и вся ответственность за поддержание целостности программы ложится на вас, а вам, как я уже говорил, доверять нельзя. Хорошо хоть Java выдает исключение, да и другие языки тоже это распознают. Но Rust идет еще дальше, предупреждая вас о возникновении такой ситуации в ходе компиляции.

Так что мы сразу отсекаем кучу программ, которые, будь они написаны на других языках, считались бы правильными и качественными. Но природа статического анализа, как и любого анализа во время компиляции, без возможности увидеть работающую программу, такова, что если он разрешает писать правильные программы, разрешает нормальные программы, то, значит, разрешает и небезопасные. Нельзя точно сказать, где граница между нормальными и ненормальными — это невычислимо. Так что анализ Rust консервативен, он отвергает правильные программы — и будет отвергать, но на поверку это не так уж плохо. Следует только попривыкнуть и принять его логику, она вполне удобна и мало что запрещает.

**LXF:** Думаю, здесь уместна аналогия с теоремой Гёделя о неполноте, согласно которой возможна либо полнота, либо непротиворечивость данных.

**ДБ:** Именно так. Но система контроля за заимствованиями со временем будет улучшаться; увидев способ улучшения такого анализа, ради создания более правильных программ, мы его применим, но для этого нам надо убедиться в его здравомыслии. Хотя опыт работы с Rust — это что-то потрясающее. Пройдя через это обучение — я бы сказал, мучение, или, скорее, чистилище будет подходящим словом... так или иначе, приходится карабкаться на эту гору несколько недель, но вид оттуда стоит трудов. Ко мне как-то зашел приятель, чтобы показать алгоритм добавления значения в двоичное дерево, и сказал: «Я сам еще не проверял, но он не падает». И вот вы видите этот код, который находит значения, заменяет узлы, проверяет их и так далее. А в C или C++ вы бы думали: «Где-то тут не без повисшего указателя», и пришлось бы тщательно всё прочитывать и искать.

**LXF:** А как насчет многопоточных программ?

**ДБ:** А это самое интересное. Маттиас Фелляйзен [Matthias Felleisen] — он профессор Северо-Восточного университета — это один из первых





лиц в компании PLT Group, создателей Racket, DrScheme и кучи других вещей, таких как крутая функциональная реактивная система под названием Father Time. Кроме того, он занимается методикой преподавания информатики и активно работает над тем, чтобы обучение этому предмету стало реально доходчивым. Пару недель назад у него была дискуссия с Гиладом Браха [Gilad Bracha], на тему «Чему учить: куда движется информатика?» Много говорили про типы. Короче говоря, Маттиас поделился своим опытом с одним классом: он набрал кучу студентов, никогда раньше не занимавшихся параллельным программированием, и учил их, как это делается в Rust. Маттиас — известный иконоборец и из вредности громит всё, что ему не по нутру; он человек искренний, но жесткий. По его словам, первые две недели студентам всё казалось сложным, потому что давалось слишком медленно, и сообщения об ошибках они считали невнятными. Но после этих двух недель проблем не возникало, программы компилировались и запускались и работали точно так, как задумано. И это, по-моему, здорово. Смысл в применении параллельного программирования в первую очередь, а не в последнюю.

И это полная противоположность тому, что мы делаем сейчас — оптимизируем недолговечный однострочный код до тех пор, пока из него уже ничего не выжать, и уже потом принимаемся за параллелизм.

**LXF:** Очень часто люди любят делить языки программирования на быстрые и медленные. К быстрым относят C, C++ и Java, а к медленным — все остальные. К каким относится Rust?

**ДБ:** Rust — быстрый язык. Естественно, мы опираемся на потрясающие достижения команды Clang. Clang — сам по себе прекрасный интерфейс, да и любой рабочий интерфейс на C++ — уже достижение, но Clang особенно хорош, и поддерживается инфраструктурой оптимизации LLVM. Мало того, люди из LLVM просто фанатики архитектуры ПО. Иметь движок, подходящий к вашему интерфейсу — это общее место, но LLVM отлично изолирован, а значит, такие языки, как Rust, от этого выигрывают.

**LXF:** Трудно ли будет программисту-любителю, знакомому, допустим, с Python и PHP, освоить Rust?

**ДБ:** Трудно сказать. Лично я с ним работаю так. Я запускаю программу Rust, с виду очень простую и ясную, но она не компилируется. Минут пятнадцать я борюсь с компилятором, процесс всё усложняется, а код становится неуклюжим. Потом я осознаю, что все эти усложнения пошли на пользу, и начинаю постепенно убирать сложности и шероховатости, и в итоге наступает красота. Так что конечный результат часто выглядит, как хорошая программа на Python. Как будто я составил словарь, потом добавил туда кое-что еще, потом всё пересмотрел и лишнее убрал. Язык поддерживает логический вывод типа, так что надо использовать типы только на границах функций, а не внутри функций, и язык их вычислит.



Так что в этом смысле я думаю, что Rust прекрасно подойдет тем, кто до этого занимался динамическими языками. В то же время, я обычно беспокоюсь за тот промежуточный этап, когда все наметки «разбросаны по полу». Там мне приходится задействовать все свои знания, и для менее опытных людей это может стать существенным препятствием. Трудность изучения Rust может оказаться его главной слабостью. У мета-программистов C++ и знатоков Haskell, я полагаю, проблем не будет, но таких очень мало.

пшло не так. Rust тоже обладает этим качеством, он говорит вам, что происходит — и вы не попадаете в Странную Петлю [англ. Tombolia, — прим. пер.]. В книге «Гёдель, Эшер, Бах» [Д. Хофштадтера; по выражению издателей — «метафорическая fuga о разумах и машинах в духе Льюиса Кэрролла», — прим. пер.] есть такая фраза: «Нарушили правила — и вот вы в Странной Петле, где все теряет смысл». А в Rust нет Странных Петель. И это должно привлечь людей. Думая о типах и создавая их, многие уже мыслят в этой логике, и осваиваются легко.

## О НАДЕЖНОСТИ ПОТОКОВ В RUST

### Смысл в применении параллельного программирования в первую очередь, а не в последнюю.

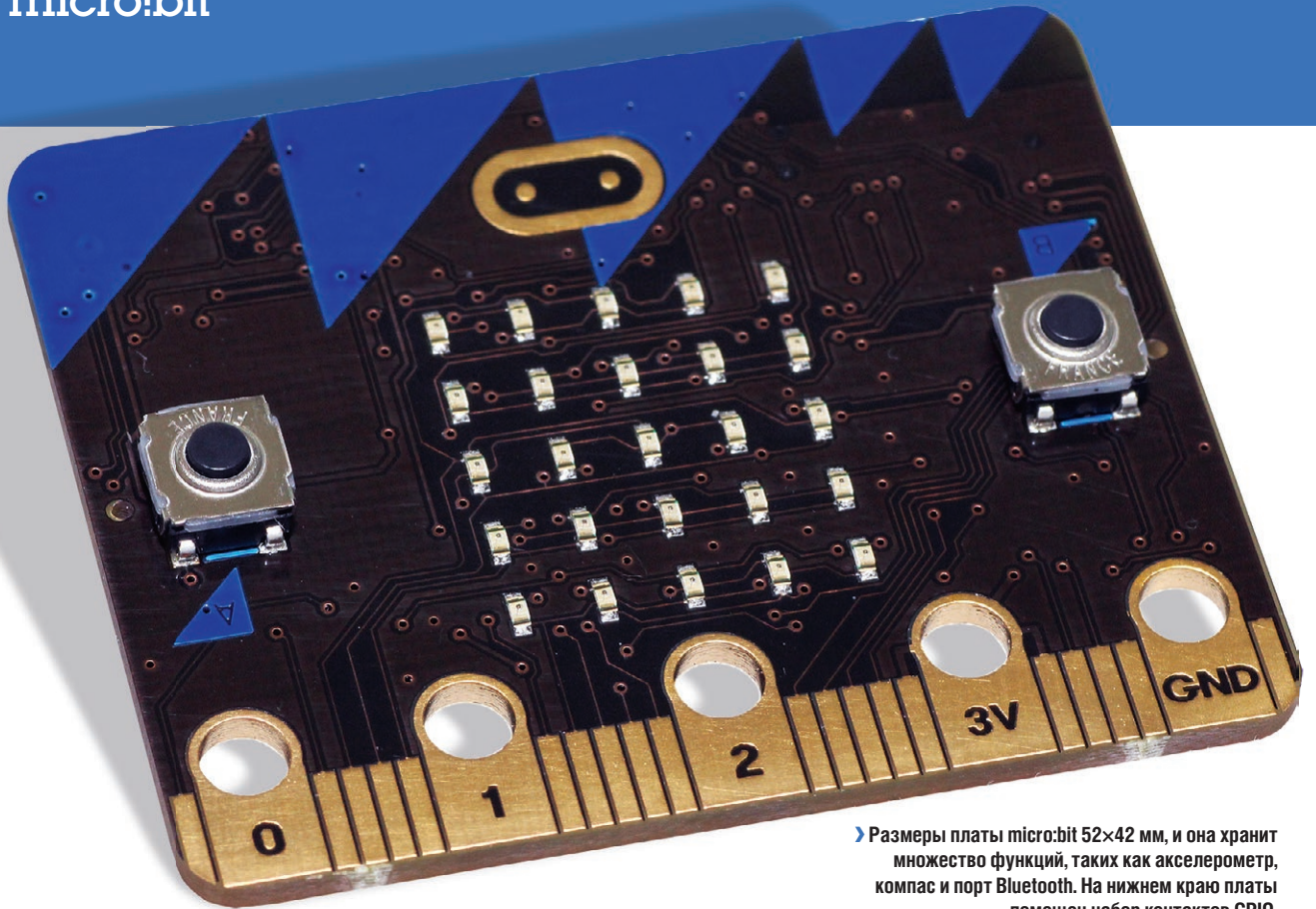
Я ищу, как лучше объяснить принцип работы Rust. Во вчерашней презентации были кое-какие неудачные моменты, но лучше всего прошли как раз самые сложные вещи. В частности, про передачу прав владения и получение временных прав, похоже, я нашел способ объяснить. В конце года выходит книжка от издательства O'Reilly, и если всё пойдет, как задумано, надеюсь, там будут качественные объяснения.

Python и Rust роднит надежность, в программах Python не бывает странных повреждений, когда система вдруг начинает вести себя необъяснимо и, в худшем случае, выдает исключение. Python очень дружелюбен, он просто сообщает вам, что именно

Некоторые об этом не думают, уж не знаю, как они программируют, но им придется трудновато.

Здорово то, что при всей низкоуровневости языка, близкого к железу, вам не надо переживать о битах и байтах. Не надо думать «Ой, вышло переполнение, теперь у меня неверный размер сегмента, я завалил программу». Вы получаете исключение, когда преобразуете 64-битное значение в 32-битное, а оно не влезает. Так что иногда это будет приветствоваться, а иногда создаст серьезные проблемы. Я не хочу голословно утверждать, что это нечто неподъемное, ведь всё дело в том, как этому учить; надо дожидаться появления хороших учителей, что я и собираюсь делать. **LXF**





› Размеры платы micro:bit 52×42 мм, и она хранит множество функций, таких как акселерометр, компас и порт Bluetooth. На нижнем краю платы помещен набор контактов GPIO.

# BBC micro:bit

## Назад в будущее компьютеров!

Лес Паундер прикладывает руки к новому BBC micro:bit — устройству, нацеленному на следующее поколение творцов.

**П**охоже, начинается целый потоп устройств, утверждающих, что они изменят будущее компьютерного образования. Чтобы приступить к программированию физических устройств, у нас есть куча вариантов, от Arduino до Raspberry Pi, но в конце 2014 г. ходил слух, что BBC (British Broadcasting Corporation) стремились повторить свое успешное выступление на сцене кодирнга 1980-х в Великобритании, поскольку тогда лидировал BBC Micro, ее собственный микрокомпьютер.

В 2011 г. состоялся ряд докладов от советников по образованию и членов парламента об отставании Великобритании в области информатики и о том, что многие дети верили, что для занятия

этой деятельностью надо переехать в другую страну. Особенно это относилось к игровой индустрии, где Великобритания сохраняла шестую позицию, но число разработчиков из Великобритании сокращалось. Поэтому образовательный сектор

**Дать детям и учителям устройство за ноль копеек и с максимумом отдачи.**

слушал стоя, когда BBC в 2015 г. объявила о партнерстве с рядом поставщиков сервиса, оборудования и ПО для доставки платформы, питаемой одноплатным микроконтроллером.

Целью проекта micro:bit не является просто введение очередного одноплатного компьютера/

микроконтроллера: проект намерен совершить прорыв и дать в руки детям и учителям устройство за ноль копеек и с максимумом отдачи. Прорыв также затронет наш подход к объединенному миру с его Интернетом Вещей, IoT. Micro:bit разработан для работы с мобильными устройствами, чтобы подстегнуть творчество в классе. С помощью micro:bit каждый сможет создать свое собственное интеллектуальное устройство с малым количеством кода.

У micro:bit также есть проекты и документация, разработанные в соответствии с учебной программой Великобритании по информатике. Все партнеры надеются возродить успех 1980-х и помочь детям понять пользу компьютерных наук в создании новых должностей в будущем.

# Micro Python: Приступим

Настройте фантастическое шоу с помощью своего micro:bit.

**Д**ля этого проекта вам понадобится подключить свой micro:bit к ПК с Linux или Raspberry Pi. Вам также понадобится светодиод, резистор на 220 Ом (КРАСНЫЙ-КРАСНО-КОРИЧНЕВЫЙ-ЗОЛОТОЙ) и три зажима-«крокодила».

В программировании физических устройств аналогом «Hello World» является управление светодиодом (LED). Это помогает убедиться, что плата и компоненты работают корректно, прежде чем перейти к чему-то более сложному.

Начнем мы со скачивания ПО Python, известного как *Mu*, с <http://bit.ly/LXF209-Microbit-Software>. Выберите самую свежую версию ПО для вашей ОС. Надо будет сделать скачанный файл выполнимым; в большинстве дистрибутивов Linux для этого можно щелкнуть правой кнопкой мыши по файлу, выбрать Properties и проделать все там. Если вы предпочитаете терминал, можете набрать `$ chmod +x <insert name of file>`

Теперь откройте приложение *Mu*, дважды щелкнув по скачанному файлу. Редактор *Mu* кажется примитивным, но над ним постоянно работают члены Python Software Foundation. В редакторе вы увидите ряд кнопок, и особого внимания достойны Flash и Repl. Flash используется для переноса вашего кода на прикрепленную micro:bit, а Repl (Read Eval Print Loop) — для интерактивного взаимодействия с micro:bit. Мы начнем наш проект с написания нескольких строк кода, который включит и выключит подсветку LED с полусекундным интервалом между этими состояниями. В верхнем окне мы импортируем всю библиотеку micro:bit

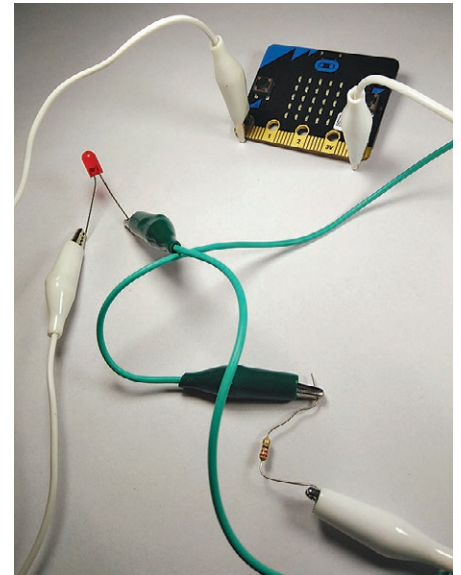
для Python: `from microbit import *`. Затем создадим бесконечный цикл, который будет содержать код, который мы хотим запустить: `while True:`. Следующая строка кода содержит отступ, согласно требованиям Python, чтобы показать, что этот код находится внутри цикла.

Вначале мы меняем значение контакта 0, который на данный момент выключен. Чтобы включить контакт, мы устанавливаем его в 1. Далее мы устанавливаем его на полусекундный сон перед выключением контакта 0, используя 0, после чего он засыпает еще на полсекунды для создания плавного цикла.

Вы заметите, что мы не импортировали библиотеку *time*, но тем не менее удается использовать функцию `sleep`. У Micro Python внутри библиотеки micro:bit есть своя собственная функция `sleep`, измеряющая длительность в миллисекундах, так что значение 500 равно половине секунды.

```
pin0.write_digital(1)
sleep(500)
pin0.write_digital(0)
sleep(500)
```

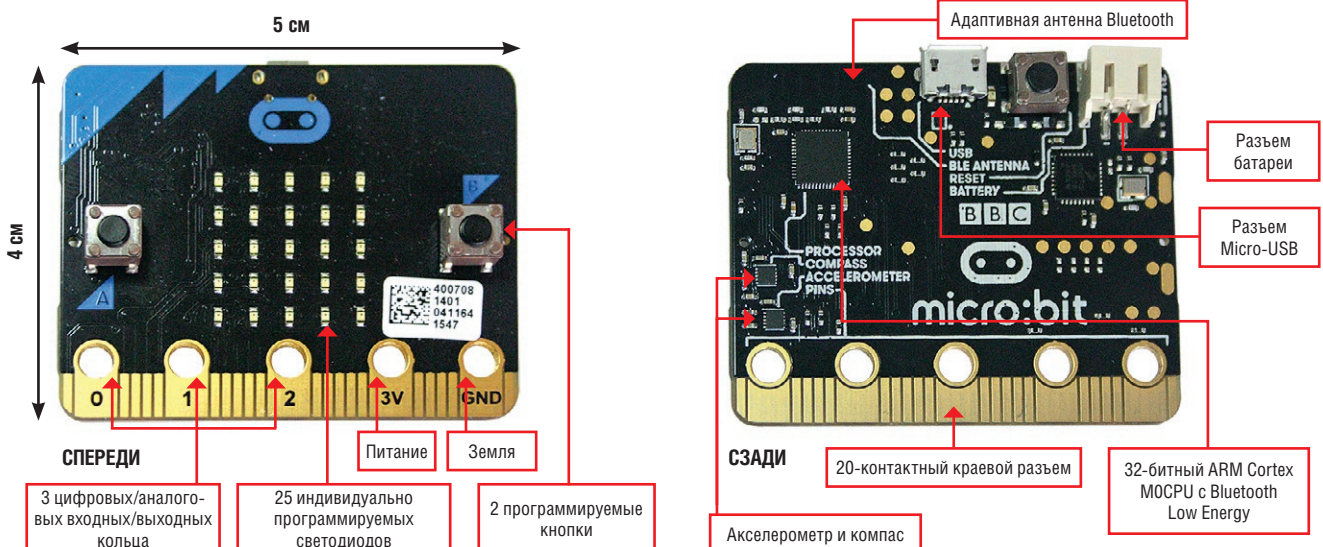
Код готов; настало время переслать его на прикрепленный micro:bit. Нажмите на Flash и подождите, пока желтый светодиод на тыльной стороне micro:bit перестанет мигать. С помощью загруженного на micro:bit кода подключим компоненты. Прикрепите один конец «крокодила» к контакту 0, а другой — к длинной ножке светодиода. Подключите другой «крокодил» к заземлению micro:bit и затем прикрепите другой конец к одной



► С помощью разъемов GPIO на micro:bit и нескольких «крокодилов» мы можем быстро построить схему для проверки нашей платы.

из ножек резистора. Подключите третий «крокодил» к другой ножке резистора и к короткой ножке светодиода. Теперь вы должны увидеть, как горит светодиод. Если это не так, проверьте, что ваша схема подключения проводов верна, удалив «крокодил» с контакта 0 и подключив его к 3 В. Если светодиод загорается, значит, подключение правильное. Bravo!

## Анатомия micro:bit





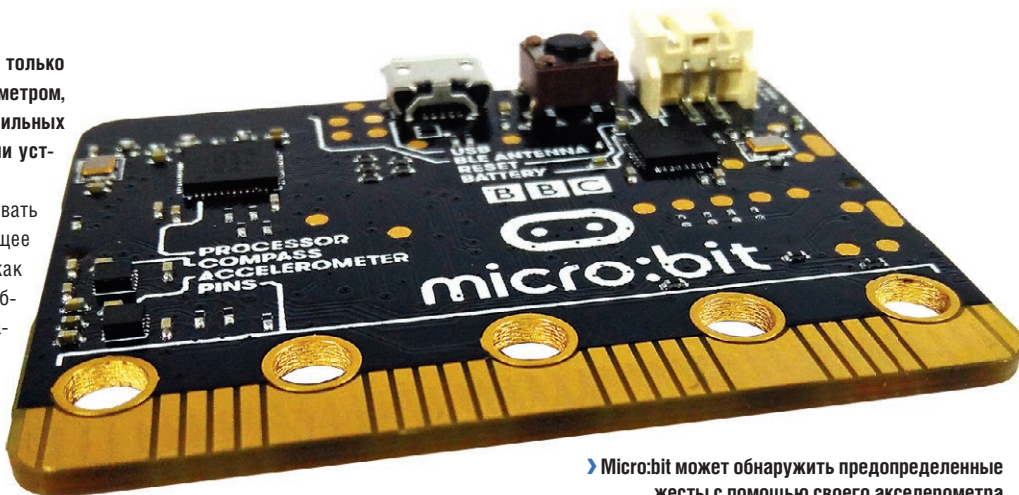
# Забавы с акселерометром

Что тут думать — micro:bit надо трясти!

**Д**ля этого проекта вам понадобится только micro:bit, поставляемый с акселерометром, который обычно применяется в мобильных устройствах для определения ориентации устройства и вращения экрана.

В этом руководстве мы будем использовать micro:bit как устройство ввода, реагирующее на жесты. Мы начнем в редакторе *Mu* и, как всегда, наша первая строка импортирует библиотеку micro:bit: `from microbit import *`. Затем используем бесконечный цикл, содержащий код, который мы хотим запустить, наподобие этого: `while True:`.

Акселерометр, встроенный в micro:bit, имеет свой собственный набор функций, пригодный для запроса положения платы в пространстве. Мы можем отследить полные x, y, z координаты платы для тонкой настройки, но бывает, что нам не нужна подобная точность, и как раз в этом случае жесты предлагают быстрое решение. Жесты — это предопределенные движения, например, встряхивание, наклон и переворачивание micro:bit. Мы можем использовать эти жесты для



► Micro:bit может обнаружить предопределенные жесты с помощью своего акселерометра и доложить о них с помощью библиотеки Python.

простого ввода, и в данном проекте употребим условные утверждения, проверяющие, какие жесты были сделаны, и соответственно реагирующие. Первый тест — увидеть, было ли устройство наклонено вверх. Вывод этого теста — либо True, либо False, но если True, то код (ниже) активируется. Перед прокруткой текста экран очищается через LED матрицу micro:bit. Наконец, делается 0,1-секундная пауза, и код снова проводит проверку.

```
if accelerometer.was_gesture('up'):
    display.clear()
    display.scroll("Dogs cannot look up")
    sleep(100)
```

Наш следующий тест называется Else...If, сокращенный в Python до elif. В этом тесте мы увидим, направлен ли micro:bit к полу.

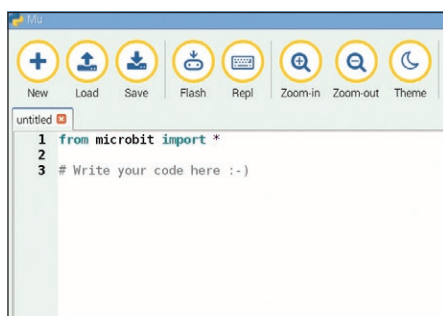
```
elif accelerometer.was_gesture('down'):
    display.clear()
    display.scroll("I feel sick")
```

```
sleep(100)
```

Мы повторяем этот процесс для еще двух тестов, охватив наклон micro:bit влево и вправо; синтаксис этого кода идентичен жесту вниз, но относится к 'left' и 'right'. Наш последний жест — встряхивающее движение, которое при обнаружении запускает активацию последнего раздела кода:

```
elif accelerometer.was_gesture('shake'):
    display.clear()
    display.scroll("Stop shaking me!")
    sleep(100)
```

Завершив код, сохраните свою работу и нажмите на Flash для отправки кода на прикрепленный micro:bit. Когда желтый светодиод на тыльной стороне micro:bit перестанет мигать, вы будете готовы использовать контроллер. Начните, наклонив micro:bit вперед, назад и из стороны в сторону. Наконец, встряхните micro:bit для проверки жеста встряхивания.



► *Mu* — пока единственный доступный оффлайн-редактор для micro:bit.

## Партнеры micro:bit

Спецификация оборудования micro:bit определяется потребностями его целевой аудитории — детьми и партнерами-поставщиками, такими как ARM, Freescale и Nordic Semiconductor, которые все вместе отвечают за CPU, акселерометр, магнитометр и Bluetooth LE. Вклад этих партнеров превратил плату из простого микроконтроллера в платформу для весьма продвинутых экспериментов.

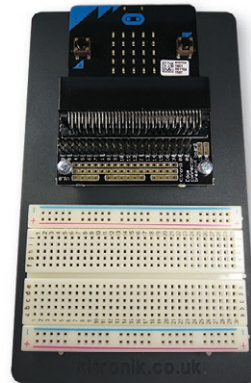
В одном пакете мы можем создать беспроводной контроллер для камеры или робота и затем на следующий день,

создать прокручивающуюся табличку с именем. Единственное ограничение micro:bit состоит в том, что он обеспечивает доступ только к пяти контактам GPIO. Но это преодолевается с помощью добавочной платы от Kitronix, другого партнера проекта micro:bit, который предоставляет полноценный GPIO для использования в проектах. Информацию об этом и ряде других продуктов можно найти на официальном сайте: <http://bit.ly/MicroBitAccessories>.

Общим дизайном micro:bit занимается Technology Will Save Us, которая

работала с детьми над созданием платы подходящих размеров для маленьких рученок и больших коннекторов.

Среди прочих партнеров проекта — Samsung, где создали мобильное приложение, способное программировать micro:bit через подключение Bluetooth. Это приложение идет с тремя проектами для проверки, а проект камеры для селфи весьма забавен и полезен, как и триггер запуска удаленной камеры для съемок жизни природы. Полный список партнеров — здесь: <http://bit.ly/MicroBitPartners>.



# Контроллер жестов Minecraft

Отправьте Стива полетать с вашим micro:bit.

**Д**ля этого проекта вам понадобится micro:bit, подключенный к Raspberry Pi, самая свежая ОС Raspbian и ПО *Mu* на Raspberry Pi (<http://bit.ly/LXF209-Microbit-Software>).

Основываясь на нашем предыдущем руководстве, мы используем micro:bit как контроллер для *Minecraft* на Raspberry Pi. Это руководство состоит из двух частей: код для micro:bit и код для Raspberry Pi. Начнем с кодирования micro:bit, и сперва откроем приложение *Mu*. Как обычно, импортируем библиотеки micro:bit — `from microbit import *`. Далее мы используем бесконечный цикл, чтобы проверить сделанный пользователем жест.

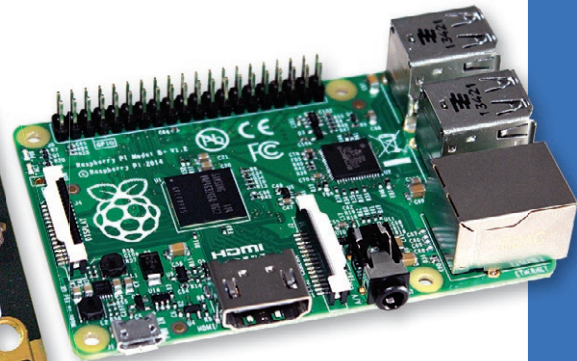
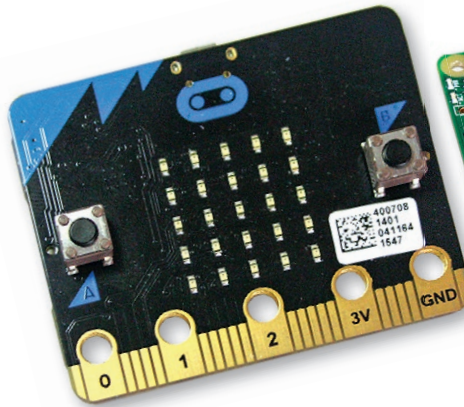
```
while True:
    if accelerometer.was_gesture('shake'):
        Если сделан правильный жест, код напечатает 'shake' оболочке и очистит матрицу светодиодов на micro:bit. Затем он прокрутит Teleport через экран перед перезагрузкой micro:bit, готового к следующему жесту.
        print("shake")
        display.clear()
        display.scroll("Teleport")
        sleep(100)
        accelerometer.reset_gestures()
        sleep(100)
```

Сохраните свою работу и отправьте код на прикрепленный micro:bit.

Теперь переключимся на Python 3, найденный в меню Programming. Нажмите на File > New Window и в новом окне нажмите на File > Save и сохраните файл как **mb-red-gestures.py**.

Мы начнем код с импорта *Minecraft* и библиотек *time* и *serial*. Библиотека *serial* будет применяться для связи с micro:bit. Также импортируем функцию `randint` из библиотеки *random*.

```
import serial, time
from mcpi.minecraft import Minecraft
from random import randint
```



➤ Новый micro:bit и Raspberry Pi отлично работают в связке.

Затем создадим три переменные: `port`, `baud` и `mc`. Они будут хранить: номер порта, к которому подключен micro:bit; скорость в бодах, с которой нам надо взаимодействовать; и сокращение от функции `create` в *Minecraft*, чтобы включить соединение с *Minecraft*.

```
port = "/dev/ttyACM0"
baud = 115200
mc = Minecraft.create()
```

Теперь используем бесконечный цикл как контейнер нашего кода. В цикле мы используем библиотеку *Serial* для подключения к порту micro:bit, настроим `baudrate` [частота в бодах], `parity` [четность], `databits` [биты данных] и `stopbits` [биты окончания], а затем попытаемся прочитать данные, которые передаются через последовательное соединение:

```
s = serial.Serial(port)
s.baudrate = baud
s.parity = serial.PARITY_NONE
s.databits = serial.EIGHTBITS
s.stopbits = serial.STOPBITS_ONE
```

Прочитав данные и сохранив их в переменной с именем `data`, мы используем задержку в 0,1 секунды, чтобы притормозить код перед конвертированием данных в строку. Затем воспользуемся

библиотекой *Minecraft* для получения текущей позиции игрока:

```
data = s.readline()
time.sleep(0.1)
data = str(data)
x,y,z = mc.player.getPos()
```

Последний раздел нашего кода выясняет, присутствует ли слово 'shake' в переменной `data`, содержащей данные, переданные через последовательное соединение USB с нашего micro:bit. Если слово найдено, то в окне чата *Minecraft* размещается сообщение, и игрок телепортируется по воздуху на случайно выбранную дистанцию через карту местности:

```
if "shake" in data:
    mc.postToChat("Teleport")
    mc.player.setTilePos(x+(randint(-50,50)),
        y+(randint(1,50)),z)
```

Сохраните созданный код. Откройте приложение *Minecraft* и его мир. После загрузки вернитесь в Python 3 и нажмите на Run > Run Module. Вернитесь к *Minecraft* и теперь встряхните micro:bit. Может потребоваться несколько попыток, но вы увидите Teleport на своем экране и micro:bit перед тем, как ваш игрок телепортируется в другое место на карте! **LXF**

## Micro Python

В конце 2013 г. краудфандинговый проект, созданный Демиеном Джорджем [Damien George], увидел более сжатую версию Python 3, написанную специально для микроконтроллеров и поддерживающую плату под названием PyBoard. Micro Python стал очень мощной платформой, предлагающей опытным разработчикам на Python возможность разобраться в программировании физических устройств, включая популярную Wi-Fi плату ESP8266, которая получит Micro Python весной 2016 г.

Использование Micro Python на PyBoard не требует дополнительного ПО, а плата распознается как

USB-диск. Код может быть написан в любом редакторе, сохранен на плату и запущен при перезапуске. Для проекта micro:bit обратились к Python Software Foundation (PSF) по вопросу создания реализации Python для micro:bit.

После ряда дискуссий между Николасом Толлервеем [Nicholas Tollervey], представляющим PSF, и BBC было принято решение, дающее доступ членам сообщества PSF, включая Демиена Джорджа, доступ к плате на ранних стадиях ради создания реализации. После долгих месяцев работы сообщества Python, теперь у нас есть крепкая

реализация Micro Python, которая постоянно совершенствуется сообществом. Одно из коронных достижений этого проекта — приложение *Mu*, которое мы использовали здесь. *Mu* — это возможная замена устаревающему редактору *IDLE* Python, в котором используется язык Python, но не очень-то учитываются потребности конечного пользователя.







# По советам м-ра Брауна

**Джوليон Браун**

В свободное от консультаций по Linux/DevOps время **Джوليон** обуздывает стартап. Его самая большая амбиция — найти причину пользоваться *Emacs*.

**Эзотерическое системное администрирование из таинственных закоулков серверной.**

## Вытеснены ИИ?

**В**озможно, вы заметили, что искусственный интеллект (ИИ) исподволь становится одним из самых заметных подводных течений в новостях за последнюю пару лет. Уже есть роботизированные автомобили, рассчитывающие на широкое применение после прохождения юридических процедур; политическая элита обсуждает хаос, который ИИ и роботы внесут в мировую экономику (хотя этот эффект сглаживается технократами в Давосе), а Стивен Хокинг [Stephen Hawking] предупреждает об опасностях, которые ИИ может внести в наш мир (рекомендую отличную книгу по этой теме — «Суперинтеллект [Superintelligence]» Ника Бострома [Nick Bostrom]).

Просвещенные читатели *Linux Format*, разумеется, знают, что термин «ИИ» обычно описывает систему с машинным обучением (более модный термин — «глубокое обучение») для решения конкретной задачи. Собственно интеллект и самосознание не имеют к этому отношения.

Выражаясь проще, находимся ли мы в той точке, когда растущая автоматизация (включая машинное обучение) может оставить сисадминов без работы? Как школьнику 1980-х, мне пришлось провести 15 минут с консультантом по выбору профессии. В эпоху господства 8-битных компьютеров я сообщил ему, что определился и стану программистом. «Зачем же братья за это, — спросил меня профессиональный консультант, — ведь через несколько лет компьютеры будут сами программировать себя?»

Хотя в своем прогнозе он и ошибся на несколько лет, DevOps — идеальное место, где обучение машины расширяет возможности сисадмина. Иногда мы ломаем голову над тем, что же на самом деле происходит на неработающей платформе. И иногда переменных слишком много — почему же обучающаяся система не смогла интерпретировать их и выдать ответ? Совсем немного осталось до той поры, когда компьютер сможет сам решать свои проблемы.

[jolyon.brown@gmail.com](mailto:jolyon.brown@gmail.com)



## AMD говорит: «Пора открыть графический сопроцессор»

AMD объявила о планах устранить препятствия для инноваций, выпустив GPUOpen.

**У**помянув о трудностях, испытываемых разработчиками при использовании API из «черного ящика», AMD выпустила набор утилит, кода и документации под лицензией MIT и разместила это на своем новом сайте GPUOpen (<http://gpuopen.com>). По словам AMD, это «позволит разработчикам создавать впечатляющие игры для ПК, компьютерную графику и вычислительные приложения для GPU, обладающие превосходной производительностью и жизнеспособностью без затрат и с помощью открытых инструментов и программ».

В блоге на сайте сотрудник AMD Ник Тибьерос [Nick Thibieroz] рассказывает о различиях между играми для консолей и ПК, где проприетарные библиотеки и инструментарию препятствуют обслуживанию, портированию и оптимизации кода разработчиками. Однако эта инициатива также адресована профессиональным вычислительным решениям, в которых роль GPU все более возрастает.

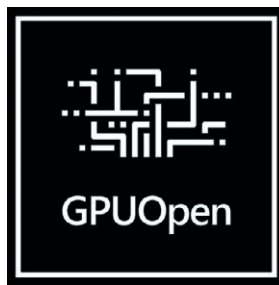
В основу GPUOpen положат три принципа. Во-первых, код и документация (которой очень не хватало в прошлом), которые позволяют разработчикам

получить больший контроль над устройствами. Во-вторых, AMD сделала ПО открытым, чтобы взаимодействовать инновациям и разработке. В-третьих, компания будет взаимодействовать с более широким сообществом разработчиков, и код уже появляется на GitHub (под лицензией MIT).

Для Linux, где графические драйверы имеют сложную историю, будучи проприетарными и рас-

пространяемыми только в виде двоичных файлов, этот код создаст открытую основу, позволяющую разрабатывать как открытые, так и закрытые драйверы. С принятием Vulkan (который должен стать наследником OpenGL) разработчики Linux должны получить в свое распоряжение открытую высокопроизводительную платформу для различных применений.

GPUOpen также содержит сведения о вычислительных проектах, которые AMD разработала для конкуренции с CUDA от Nvidia в области HPC (высокопроизводительных вычислений). Конкуренция на рынке видеокарт очень сильна, и лидирующую долю рынка занимает Nvidia. AMD надеется, что данная стратегия позволит сократить разрыв.



## Часть 2: Приватные контейнеры Rancher

В этом месяце **Джолион Браун** глубже зарывается в платформу *Rancher* и наблюдает за безопасными обновлениями из компактного интерфейса.

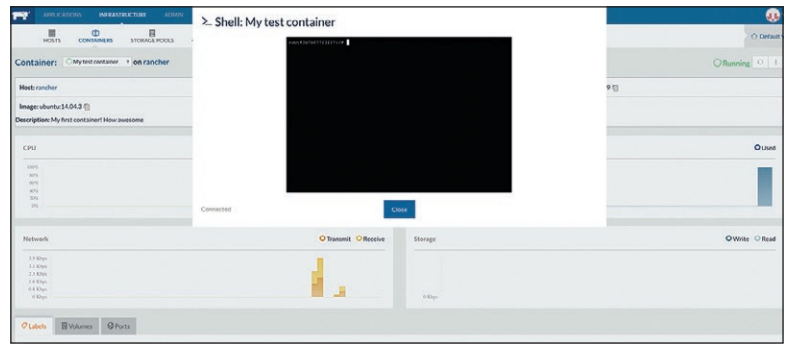
Месяц назад я рассказывал о *Rancher*, открытой платформе для запуска сервиса частных контейнеров. В конце того урока у меня было несколько виртуальных машин RancherOS, запущенных и готовых к использованию. Темп разработки проекта Rancher таков, что с момента выхода прошлого урока вышли новые обновления. Это дает мне право чуть подробнее рассказать о том, как легко обновить ОС в *Rancher* (или откатить обновления). На предыдущем уроке я настроил три виртуальных машины, поэтому могу подключиться по SSH к каждой из них и проверить, какая версия RancherOS на них запущена.

```
$ vagrant ssh rancher-01
$ sudo ros -v
```

В моей локальной системе эта команда возвращает `rancherctl version v0.4.1`. Это довольно свежая версия, но доступны и варианты обновления. Их можно просмотреть с помощью команды

```
$ sudo ros os list
rancher/os:v0.4.0 remote
rancher/os:v0.4.1 remote
rancher/os:v0.4.2 remote
rancher/os:v0.4.3 remote
```

Будь это обычный компьютер или виртуальная машина, для обновления было бы достаточно скомандовать `$ sudo ros os upgrade` и получить предложение (через Y/N в командной строке) обновиться до последней доступной версии. Последовала бы быстрая загрузка ОС и перезагрузка, и все было бы закончено. Но недостаток использования *Vagrant* для быстрых и легко запускаемых примеров типа нашего состоит в том, что так мы, к сожалению, теряем возможность подключиться обратно к виртуальной машине с помощью `vagrant ssh` (если бы в *Linux Format* разрешились эмодзи, я бы вставил сюда грустное личико). Похоже, дело в том, что в виртуальную машину *Vagrant* во время ее создания необходимо встроить ключи и т.п. Простой способ это обойти — запустить обновление компьютера с *Vagrant*; но для этого надо удалить и снова создать окружение *Vagrant*. Тем не менее, о простоте обновлений в «реальной» среде стоит знать. Вернуться к предыдущей версии также возможно. Для этого используется параметр `-i` команды `ros upgrade` (например, `$ sudo ros upgrade -i rancher/os:v0.4.1`). Это позволит быстро откатиться до доступной версии. В остальной части



урока я обновил версию RancherOS, указанную в файле *Vagrantfile* как `=>0.4.3`, и переустановил сам *Rancher*. Подробности описаны в последнем номере [стр. 47, LXF208].

Еще одно короткое замечание о RancherOS — при подключении к виртуальной машине или хосту, где она запущена, можно выбрать различные «консоли» для взаимодействия с системой. Наряду с командной строкой *Busybox* по умолчанию также доступны *Ubuntu* и *Debian*. Они являются «персистентными», т.е. сохраняют изменения после перезагрузок, тогда как консоль по умолчанию таким свойством не обладает. Команда `$ sudo ros service list` выведет список доступных вариантов (часть строк опущена):

```
disabled debian-console
disabled ubuntu-console
```

Переключиться в них очень легко: команда `$ sudo ros service enable debianconsole` включает эту консоль, а после перезагрузки она вступает в действие. Лично я думаю, что здесь стоит воспользоваться более «быстротечной» консолью — в конце концов, эти виртуальные машины — «скот», и с ними мы будем взаимодействовать не слишком часто. Думаю, что настоящая контейнерная инфраструктура должна справляться с отключением/перезагрузкой хоста и потерей его конфигурации. Пусть инфраструктура позаботится о себе — масштабируйте свои сервисы горизонтально и проектируйте их так, чтобы потеря одного узла не влияла на конечных пользователей. Мы все знаем, что это сложнее, чем кажется, но посмотрим, как *Rancher* поможет нам достичь цели. »

» На скромных началах родилась империя частных контейнеров. Здесь мы видим состояние контейнера и даже можем открыть оболочку, если вам это на самом деле надо.

### И что, это можно запускать в рабочей среде?

Несомненно, многие из вас уже думают, что *Rancher* решит все ваши проблемы, связанные с контейнерами, и позволит вам заработать массу похвал. Но в вашей голове уже крутится разговор с начальником: «А как насчет контрактов на поддержку?» и «Когда же выйдет версия 1.0?». Я связался с *Rancher*, чтобы узнать об их планах в этой сфере, и компания любезно отправила мне следующую информацию: «Rancher Labs планируют выпустить версию 1.0 платформы *Rancher* в первом квартале. Вместе с ней выйдет коммерчески лицензируемая

и поддерживаемая версия продукта с опциями поддержки *Standard* и *Platinum*. Время ответа для соглашений об уровнях обслуживания и время работы инженеров поддержки по телефону определяется пакетом *Standard* или *Platinum*, который выберет клиент. Rancher Labs уже предоставила полнофункциональную систему отправки запросов с web-интерфейсом, обладающую достаточным количеством функций и возможностей. Портал поддержки доступен компаниям с любым уровнем поддержки в режиме 24×7. На портале компании

могут получить свежую информацию, ввести информацию о своих тестовых случаях, добавить информацию к существующим случаям, получить информацию и обновления, закрыть тестовые случаи и вывести информацию о закрытых и открытых тестовых случаях. Система поддержки Rancher Labs также предоставляет постоянно улучшающуюся базу знаний с web-интерфейсом самообслуживания в режиме 24×7».

Мне кажется, это вполне пристойно. Подробно-сти см. на сайте *Rancher* — [www.rancher.com](http://www.rancher.com).



И последнее отступление. Я запускаю *Rancher* на одном из хостов, находящихся под его собственным управлением. Это не идеально (не рекомендуется), но для нашего урока сойдет. Самые хваткие из вас уже кричат про одиночную точку отказа. Вы правы. *Rancher* можно запускать в конфигурации HA, но на данный момент это не слишком просто и требует использования внешнего (общего) экземпляра *MySQL* и некоторых других программ (*Redis*, *Zookeeper*). Подробности см. на <http://bit.ly/RancherMultiNodes>, но я ожидаю/надеюсь, что проект станет гораздо удобнее для пользователя, когда приблизится к версии 1.0 (см. врезку «И что, это можно запускать в рабочей среде?», стр. 47).

Конечно, основная задача *Rancher* — запуск контейнеров. Запустить контейнер через web-интерфейс очень просто. На странице с хостами и инфраструктурой выберите Add container [Добавить контейнер], после чего появится меню создания контейнера. Там я могу создать контейнер, задав всего несколько параметров (имя, описание и используемый образ). В качестве образа по умолчанию используется Ubuntu 14.0.4.3. Всякие сложные параметры пока можно игнорировать. После этого снова откроется окно с хостами, где будет показан уже запущенный контейнер. Обратите внимание, что также запустился контейнер «сетевого агента» — это автоматическая система контейнеров, созданная *Rancher* для обработки таких задач, как организация сети между хостами и проверка работоспособности. Щелкнув на созданном контейнере, я увижу несколько приятных графиков, касающихся потребления процессора, памяти, сети и хранилища. К тому же, с помощью меню в правой части окна можно решать базовые задачи, такие как перезапуск, остановка, удаление, просмотр журналов и открытие оболочки.

Конечно, это можно сделать и в командной строке. Подключившись по SSH к любому из хостов с помощью *Vagrant*, я смогу

запустить приведенную ниже команду, которая установит и запустит контейнер (а затем я проверю, что он запущен, командой ps):

```
$ docker run -l io.rancher.container.network=true -itd ubuntu bash
$ docker ps
```

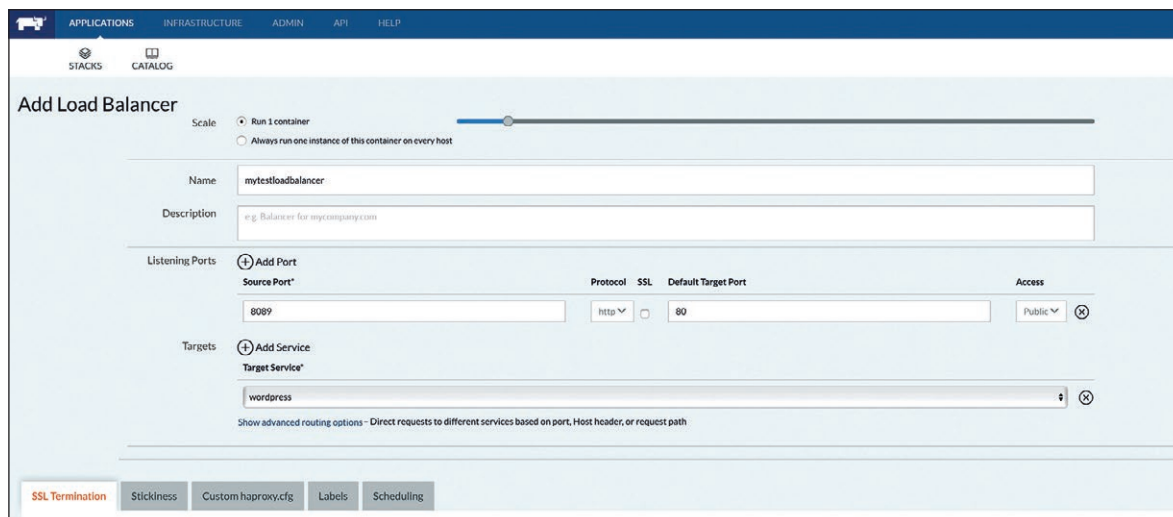
Параметр `io.rancher.container.network=true` присоединяет контейнер к сети под управлением *Rancher*. Вернувшись в браузер, я увижу новый контейнер в том же диапазоне IP-адресов, что и остальные контейнеры (с IP-адресом `10.42.X.X`). Запуск команды `docker inspect` для того же контейнера покажет более привычный IP-адрес `172.17...`. В чем дело? В сети *Rancher* контейнерам назначаются обычный IP-адрес моста `docker`, а также IP-адрес *Rancher* для моста по умолчанию `docker0`, который позволяет обращаться к ним по управляемой сети.

## Факторы окружения

В *Rancher* можно определить различные окружения. Думаю, эта идея знакома большинству читателей — иметь развернутое окружение в виде одного набора ресурсов и рабочую среду в виде другого очень удобно, не правда ли? Кто не согласен, теряет все очки «Советов» и возвращается обратно к LXF1. Вы не пройдете уровень и не наберете свои £200. Если на вашей работе нет такой конфигурации и у вас есть фактические данные конечных пользователей, работу советую поменять. Но я отклонился.

Для создания среды в *Rancher* достаточно нескольких щелчков мыши (используйте выпадающее меню справа). При создании новой среды снова открывается окно Adding your first host [Добавление первого хоста], которое я увидел при первом запуске *Rancher*. Пользователей можно раскидать по разным средам — это удобно, да и поставит на место этих надоедливых экспертов по безопасности. Но здесь я буду использовать только окружение по умолчанию.

» А вы помните, что для добавления балансировщика нагрузки придется плестись к сетевой команде и ждать шесть недель, пока это произойдет?



## Что еще следует знать?

Под колпаком *Rancher* скрыт ряд отличных функций. Все они доступны через встроенный API, и к нему можно обратиться на любом стоящем скриптовом языке. Это удобно для подключения к другим инструментам (например, для мониторинга, о котором мы подробнее поговорим через месяц) и годится как основной метод связи с *Rancher*.

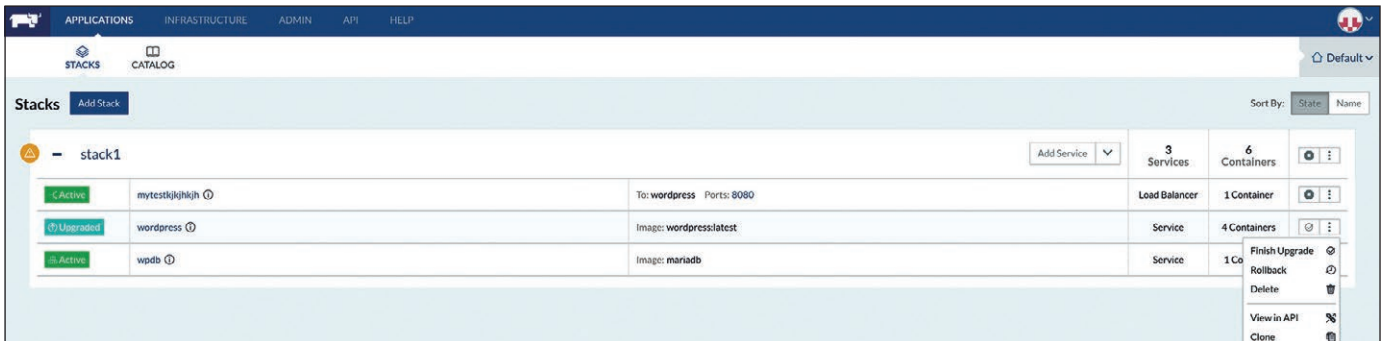
*Rancher* также поддерживает пользовательские реестры для локальных образов *Docker*, и вполне вероятно, что многие компании не захотят загружать свои внутренние приложения в Docker Hub или

другую службу, предпочитая, чтобы они надежно разместились в их внутренних сетях. Прекрасно было бы иметь автоматизированный процесс сборки, который помещает образы *Docker* в реестр для использования *Rancher*.

При определении служб можно пометить их как «внешние», т.е. не находящиеся под управлением *Rancher*. Где-то может размещаться база данных или SMTP-шлюз, с которыми контейнеры в *Rancher* могут захотеть связаться. Их можно определить в *Rancher* (иногда достаточно указать имя, IP-адрес

и порт), позволяя контейнерам найти их с помощью встроенного обнаружения служб, которое есть в *Rancher*.

Наконец, в окне «каталога» *Rancher* есть огромное количество приложений, специально настроенных для работы в среде *Rancher*. К ним относятся базы данных, файловые системы вроде Gluster и системы сборки типа Jenkins. Их можно установить одним щелчком мыши — по сути, это заранее настроенные файлы для `docker/rancher-compose`. Очень удобно!



Уверен, сейчас вы думаете, что всё это здорово, но симпатичный графический интерфейс позволит запустить разве что чистую установку *Docker* на любой старой системе. Верно. Настоящее преимущество *Rancher*, на мой взгляд, проявляется в его работе со «стеками». Во вселенной *Rancher* стек определяется как «отображение той же концепции, что и проект *docker-compose*. Он представляет группу сервисов, которые составляют обычное приложение или рабочую нагрузку».

## Стек контейнеров

Со стеками можно взаимодействовать различными способами. Разумеется, через интерфейс пользователя, а также с помощью утилиты под названием *rancher-compose*. Это версия *docker-compose* для нескольких хостов. Ее можно загрузить прямо из установки *Rancher*. В правом нижнем углу каждого окна есть ссылка `Download CLI [Загрузить командную строку]` (я кратко рассказывал о ней на прошлом уроке). Поместив получившийся исполняемый файл где-нибудь по пути `$PATH`, я смогу очень быстро создавать файлы настройки для своих сервисов (а постоянным читателям известна моя страсть добавлять файлы настройки для таких систем в системы управления версиями). Кстати, очень рекомендуется поддерживать версию *rancher-compose* в соответствии с версией самого *Rancher* (например, загружать новую версию при каждом обновлении). Однако прежде чем использовать *rancher-compose*, надо пройти некоторую аутентификацию. Эта утилита общается напрямую с API *Rancher*, и у всех, кто ее вызывает, должен быть соответствующий секрет или ключ. Но создать их просто. В среде, с которой я собираюсь работать, достаточно щелкнуть на вкладке API в верхней части окна, затем нажать `Add API Key [Добавить ключ API]`, и откроется окно для ввода имени пользователя (ключ доступа) и пароля (секретный ключ). Они будут показаны только один раз — после ввода имени, описания и выбора кнопки `Продолжить` секретный ключ исчезнет навсегда. Единственный способ сбросить его — удалить запись API и создать ее снова. Для использования с *rancher-compose* эти значения нужно экспортировать как переменные окружения (их также можно указать в качестве аргументов).

```
$ export RANCHER_URL=http://server_ip:8080/
$ export RANCHER_ACCESS_KEY=<username_of_key>
$ export RANCHER_SECRET_KEY=<password_of_key>
```

Для примера я на этом уроке создам простой стек с помощью двух файлов. Первый из них — файл `docker-compose.yml`. Здесь нет ничего сложного, я просто создам определения для своего любимого примера, стека Wordpress:

```
wordpress:
  image: wordpress:4.2
  links:
    - db:mysql
wpdb:
  image: mariadb
environment:
  MYSQL_ROOT_PASSWORD: example
```

Второй — действительно небольшой файл `rancher-compose.yml`:

```
wordpress:
  scale: 2
wpdb:
```

Создав эти файлы и экспортировав переменные окружения, я могу запустить свой стек:

```
$ rancher-compose -p stack1 up
```

Командная строка дает немало полезного вывода, но было бы здорово взглянуть на интерфейс пользователя *Rancher* и посмотреть, как запускаются контейнеры. Перейдя к приложениям и стекам, вы увидите, как ваша новорожденная платформа для блога борется за жизнь. Даже борется буквально — во время тестирования *MariaDB* горько жаловалась на недостаток памяти для запуска. В моем случае оказалось, что *Rancher* пытается запуститься на моем, подготовленном по лучшим рекомендациям узле, где был уже запущен *Rancher*. Однако *Rancher* смог выявить эту ошибку и переместился на другой узел, где успешно запустился (пожалуй, всем, кто хочет выполнить действия этого урока, стоит зайти в *Vagrantfile* и увеличить значение параметра `vm_mem`).

В своем примере конфигурации я определил серверы приложений *Wordpress* (оба разворачиваются *Rancher*, согласно запросу) немного необычным образом — без указания открытых портов и со старой версией *Wordpress* (так определенно нельзя делать в современном мире, когда с ними смогут разобраться и малыши).

Давайте обратимся к вопросу с портами. Я хочу выполнить балансировку нагрузки для трафика к этим двум узлам с помощью встроенной функции *Rancher*. На нашем уроке я буду делать это через интерфейс пользователя. Вернувшись на страницу со стеками, можно щелкнуть по `Add Service [Добавить сервис]`, а затем `Add Load Balancer [Добавить балансировщик нагрузки]`. После определения портов и целевого сервиса, который я хочу сбалансировать (см. рис. на стр. 48), я могу легко запустить балансировщик нагрузки (чтобы он запустился, надо щелкнуть по `Запустить`). К сожалению для меня, *Wordpress* не захотел запускаться в тестовой настройке (я получал много ошибок 504). Может, и правда надо обновить *Wordpress*?

Вернувшись в файл `docker-compose`, я изменю строку

```
image: wordpress:4.2
```

на

```
image: wordpress:latest
```

Теперь я могу показать, как происходят обновления (для этого удобно открыть представление приложения/стека).

```
$ rancher-compose -p stack1 up --upgrade --pull
```

Через некоторое время (которое уйдет на загрузку последней версии образа Wordpress) *Rancher* запустит два новых контейнера с новым образом и пометит сервис как обновленный. В этот момент (и после некоторого тестирования) вы сможете щелкнуть по `Finish upgrade [Завершить обновление]`, чтобы подтвердить изменения, и *Rancher* приберется за собой, удалив старые контейнеры. Впечатляет! Как всегда, доступного мне места хватает только на то, чтобы создать впечатление о платформе, но *Rancher* мне определенно нравится, и я буду приглядывать за ним. **LXF**

» Впечатлите своих коллег и клиентов своим мастерством в сложных обновлениях. Но не показывайте им это окно, чтобы они не увидели, как всё просто.



# 7 советов быстрого поиска работы от hh.ru

- 1. Определите цель.** Решите, кем вы хотите работать, как бы смешно это ни звучало. Точно сформулируйте вашу должность. Работодатель не найдет вас, если название резюме будет общим: «Менеджер» или «Начальник». Лучше уточните: «Менеджер по закупкам» или «Начальник строительной бригады».
- 2. Узнайте о своих способностях.** Вы все еще в поиске своего призвания? Пройдите онлайн-тест «Профориентация»\* [hh.ru/article/proforientation\\_promo](https://hh.ru/article/proforientation_promo) и узнайте, какая работа вам больше всего подойдет.
- 3. Составьте резюме.** Сделать это на hh.ru легко. Главное — заполните все предлагаемые поля. Уделите особое внимание опыту работы и вашим достижениям — так вы покажете работодателю вашу компетентность.
- 4. Настройте процесс.** Подпишитесь на подходящие вакансии и получайте самые свежие на почту. А также скачайте мобильное приложение HeadHunter, чтобы искать работу в любое время, в любом месте.
- 5. Действуйте.** Откликайтесь на все интересующие вас вакансии. Пишите сопроводительные письма работодателю, поясняя, почему вас интересует эта вакансия.
- 6. Сделайте резюме заметным.** [hh.ru/applicant/services](https://hh.ru/applicant/services) Подключите «Яркое резюме»\*, чтобы выделить резюме цветом, и «Автообновление»\*, чтобы поднимать его в результатах поиска. Работодатели обратят на вас внимание.
- 7. Подготовьтесь к собеседованию.** Поздравляем, вас пригласили! Самое время подготовить ответ на вопрос: «Почему мы должны взять именно вас?». Узнайте максимум информации о компании и подготовьте небольшую речь о том, какой вы классный специалист.

**И помните, что работа найдется для каждого!**



Наши эксперты помогут вам с любым приложением Linux!



**ЕВГЕНИЙ БАЛДИН**  
Подтвердивший  
свою квалификацию  
физик.

## GNU/Windows? Вы серьезно?

Кирпич ни с того ни с сего никому и никогда на голову не свалится.  
*Воланд. «Мастер и Маргарита»*

**П**риложения Ubuntu теперь можно прозрачно устанавливать и запускать в Windows 10!

Неправда, конечно, в том смысле, что всё прозрачно и работает, но для Microsoft интеграция Ubuntu в Windows 10 — это действительно поступок. Немного непонятно, в каком направлении, и экранные снимки пока демонстрируют в основном *Bash* под пользователем *root*, да и то предупреждают, что с цветами могут иметь место проблемы из-за ограничений родного для Windows терминала.

Безусловно, практически все свободные программы, доступные в рамках GNU/Linux, могут быть скомпилированы и запущены в Windows, но все это делалось, как правило, через усилия сторонних надстроек и костылей (спасибо Cygwin за то, что он есть). А здесь даже пресловутый `fork()` переписали под такое счастье. Так что GNU/Windows, скорее всего, быть, и это лучше, чем Windows без приставки GNU, хотя я не стал бы переходить на него с GNU/Linux.

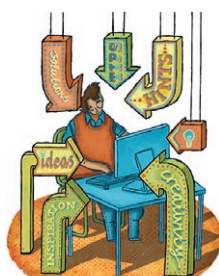
**PS** Неожиданно для себя обнаружил на LOR, что есть такая программа под названием *Valentina*, а именно параметрическая 2D-САПР для проектирования выкроек одежды. Автор (он из Житомира), кстати, устраивал мастер-класс на Libre Graphics Meeting 2016.  
[E.m.Baldin@inp.nsk.su](mailto:E.m.Baldin@inp.nsk.su)

## В этом месяце вы научитесь...



### Выводить тексты и числа ..... 52

С присущей ему кропотливостью, **Дмитрий Пантелеичев** разбирает возможные способы вывода текстовой и числовой информации на экран.



### Работать в терминале .... 56

**Ник Пирс** проведует теорию, что работа в терминале во многих отношениях эффективнее, чем в GUI. Проверим это на практике.



### Ставить Ubuntu ..... 58

Дешевых планшетов с Windows развелось видимо-невидимо, и **Ник Пирс** решил таким воспользоваться для установки любимой ОС.



### Разбираться в Portage ..... 62

Почитесь у **Нейла Ботвика** по максимуму использовать гибкость Gentoo: если вам мало программ, добудьте их из дерева портежей.



### Запросто брать Octave ..... 64

**Афна Рехмана** привела в восхищение программа, умеющая превратить беспорядочные числовые данные в элегантные графики.



### Вещать потоком с брелка ..... 68

Да, потоковое вещание можно организовать даже с USB-флешки! **Ник Пирс** строит медиа-центр на Pi Zero. Затрат — минимум, удовольствия — море.



### Оперировать шардами ..... 72

**Лада Шерышова** подключает коннекторы Python для написания приложений, способных работать с шардированными данными.



### Делать снимки программ ..... 76

**Павел Емельянов** занялся созданием утилиты, которая позволяет снимать информацию о состоянии программы прямо во время выполнения.

## АКАДЕМИЯ КОДИНГА



**И новичкам, и гуру!  
Всегда полезно будет познать  
нечто доселе неведомое**

**API на Swift ..... 80**  
Swift для Linux все еще находится в стадии альфа-тестирования, но **Пол Хадсон** очертя голову принимается за реальный проект для Github.

**MongoDB и блоги ..... 84**  
Под занавес своей серии статей **Михалис Цукалос** учит на сайте блог, употребив скрипты на Python — поскольку без бутылки тут не разобраться.



# GNU Core Utilities

Путешествие **Дмитрия Пантелеичева** по компонентам пакета ключевых утилит продолжается...



**Наш эксперт**

**Дмитрий Пантелеичев** считает, что любую технологию надо изучать от простого к сложному, и каждый шаг закреплять практически примерами.

Сегодня нас ждут средства для вывода текста (*echo*, *print* и *yes*), для числовых операций (*factor*, *seq* и *numfmt*) и одна утилита для принудительного завершения работы (*kill*).

## coreutils и bash

Но вначале необходимо сделать одно отступление. Если ввести в консоли слово `echo`, то, скорее всего, вызвана будет не эта утилита, а встроенная команда `echo`, входящая в командную оболочку *Bash*. Такая ситуация возникает со многими программами, имена которых совпадают с командами *Bash*.

Проверить, откуда вызывается команда, поможет команда `type`, также входящая в *Bash*. Давайте, например, поинтересуемся, является ли `cd` отдельной программой:

```
type cd
```

Мы получим сообщение, что `cd` — это встроенная команда:

```
cd is a shell builtin
```

А как насчет `mkdir`?

```
type mkdir
```

А вот `mkdir` является внешней программой. Мы даже увидим, по какому пути она находится:

```
mkdir is /usr/bin/mkdir
```

Итак, зная теперь, как это делается, поинтересуемся насчет `echo`.

```
type echo
```

Что же мы видим? Оказывается, `echo` — тоже встроенная команда.

```
echo is a shell builtin
```

Тем не менее, существует и внешняя программа с таким именем, и она входит в пакет *coreutils*. Чтобы гарантированно вызвать именно ее, а не команду *Bash*, необходимо воспользоваться другой утилитой *coreutils* — *env*. На следующих уроках мы разберем и ее. А пока просто будем всегда вызывать `echo` через *env*. Кстати, команду `cd` вы через *env* не вызовете. Можете попробовать: тогда появится сообщение «Нет такого файла или каталога».

## echo

Утилита `echo` выводит текстовую строку, добавив в конец символ переноса строки. У нее три ключа: `-n` запрещает добавлять символ переноса строки, `-e` требует интерпретировать экранирующие последовательности, `-E`, наоборот, запрещает интерпретацию экранирующих последовательностей (по умолчанию их интерпретация также запрещена).

Текст может быть заключен в одиночные кавычки, двойные кавычки, а также может быть без кавычек. Пробуем.

```
env echo - Кто там стучится в поздний час?; env echo "- Конечно я, Финдлей."; env echo '- Ступай домой, все спят у нас.'; env echo - "Не все", сказал Финдлей.
```

Здесь мы четыре раза вызвали `echo` (и увидели, что команды можно вводить в одной строке, отделяя друг от друга точкой с запятой). Следовательно, в результате увидим на экране четыре строчки.

```
- Кто там стучится в поздний час?
Конечно я, Финдлей.
- Ступай домой, все спят у нас.
Не все, сказал Финдлей.
```

Как видим, все кавычки полностью проигнорированы. Дело в том, что, если вводить после `echo` текстовые строки друг за другом, они будут объединены в единую строку. При этом фрагмент, заключенный в кавычки, воспринимается программой как одна строка из списка объединяемых строк. Строки объединяются, и поэтому кавычки игнорируются.

Чтобы вывести в консоли кавычки, надо текст, содержащий их, окружить кавычками другого типа.

```
env echo - Как ты прийти ко мне посмел?; env echo "Посмел", сказал Финдлей.; env echo - Небось наделаешь ты дел.; env echo 'Могу', сказал Финдлей.
```

Результат:

```
- Как ты прийти ко мне посмел?
"Посмел", сказал Финдлей.
- Небось наделаешь ты дел.
'Могу', сказал Финдлей.
```

Другой вариант решения — экранировать все кавычки с помощью обратного слэша. В этом случае сам текст в кавычки заключаться не должен.

```
env echo - Тебе калитку отвори...; env echo '\A ну', сказал Финдлей.; env echo - Ты спать не дашь мне до зари.; env echo '\Не дам', сказал Финдлей.
```

Результат:

```
- Тебе калитку отвори...
'\A ну', сказал Финдлей.
- Ты спать не дашь мне до зари.
'\Не дам', сказал Финдлей.
```

Если открывающая кавычка (любая) не будет закрыта парной кавычкой, то при нажатии `Enter` консоль будет ожидать ввода следующей строки, пока не появится закрывающая кавычка. Таким способом можно ввести многострочный текст.

```
env echo '- Попробуй в дом тебя впустить.
"Впустить", сказал Финдлей.
- Всю ночь ты можешь прогостить.
"Всю ночь", сказал Финдлей.'
```

Результат:

```
- Попробуй в дом тебя впустить.
"Впустить", сказал Финдлей.
- Всю ночь ты можешь прогостить.
"Всю ночь", сказал Финдлей.
```

Подобно тому, как кавычку можно отобразить, экранируя ее обратным слэшем, так же можно экранировать и любой символ, например, точку с запятой, которая без экранирования служит для разделения команд. Экранировать можно и сам обратный слэш. То есть, его надо ввести два раза, чтобы он отобразился один раз.

```
env echo '\- С тобою ночь одну побудь.'; env echo '\Побудь\.', сказал Финдлей.;
```

Результат:

```
- С тобою ночь одну побудь.
"Побудь", сказал Финдлей.;
```

Если же текст заключен в кавычки (любые), то внутри них обратный слэш ничего не экранирует. Выводится ровно столько символов, сколько их введено.

```
env echo "\- Ко мне опять найдешь ты путь.\."; env echo "\Найду\". сказал Финдлей.;
```

Результат:

```
\- Ко мне опять найдешь ты путь.\
\Найду\". сказал Финдлей.
```

Обратный слэш также используется в экранирующих последовательностях. Это такие сочетания символов, при интерпретации которых происходят некоторые действия. Например, вводится непечатаемый символ: перенос строки, удаление символа слева. Или вводится редкий символ, которого нет на клавиатуре. Или даже звучит звук динамика. Интерпретация таких последовательностей включается ключом `-e`. Если текст вводится без кавычек, то экранирующие последовательности должны вводиться двумя обратными слэшами, идущими подряд, потому что один обратный слэш в этой ситуации просто будет экранировать соседний символ. Рассмотрим эту ситуацию на примере последовательности `\n`, означающей перенос строки:

```
env echo -e " - О том, что буду я с тобой, \n\"Со мной\", сказал Финдлей.
```

Результат:

```
- О том, что буду я с тобой,
\"Со мной\", сказал Финдлей.
```

Если текст заключен в кавычки, то экранирующая последовательность вводится одним слэшем.

```
env echo -e "' - Молчи до крышки гробовой.\n\"Идет\", сказал Финдлей.'
```

Результат:

```
- Молчи до крышки гробовой.
\"Идет\", сказал Финдлей.
```

Полный список экранирующих последовательностей можно увидеть, вызвав справку по команде `info coreutils 'echo invocation'`. Поэкспериментируйте с ними.

Утилита `echo` также позволяет прямо в строку вставлять команды и выводить результаты их выполнения. Эти команды должны быть окружены символами ``` («гравис» или «тупое ударение»). Давайте введем такую строчку:

```
env echo Меня зовут `whoami`. Я работаю под `uname` версии `uname -r` на машине `hostname`.
```

Результат выдаст все ваши секреты:

```
Меня зовут dima. Я работаю под Linux версии 3.19.0-32-generic на машине dima-System-Product-Name.
```

Вот мы заодно повторили материал самого первого урока. Здорово, правда?

## printf

С утилитой `printf` — та же история, что и с `echo`: ее тоже надо вызывать через `env`. Она выводит строковые или числовые данные в форматированном виде, то есть, с указанием, как именно они должны быть представлены.

Первым параметром `printf` является формат, все остальные — аргументы, которых может быть сколько угодно. Выведем число 5 в форме числа с фиксированной точкой:

```
env printf %f\n 5
```

В данном случае `%f\n` — это формат, `5` — аргумент. Входящая в формат подстрока `%f` — это управляющая последовательность. Она указывает, в какой вид преобразовать аргумент. Символы, не входящие в управляющую последовательность, просто выводятся «как есть» рядом с преобразованным значением аргумента. Здесь мы добавили экранирующую последовательность `\n`, означающую перенос на новую строку. Без нее вывод будет некрасивым, так как `printf` не добавляет перенос строки. Результат должен получиться таким:

```
5,000000
```

Можно подставить в `printf` не только один, но и несколько аргументов.

```
env printf "%f " 5 6 7; env echo ""
```

Вот что получится:

```
5,000000 6,000000 7,000000
```

И даже можно каждому из аргументов подставить свой формат. В этом случае все форматы следует сосредоточить в единой строке:

```
env printf "%f %i %e %s\n" 5 6 7 OK
5,000000 6 7,000000e+00 OK
```

Числовые аргументы можно вводить в десятичной, восьмеричной или шестнадцатеричной записи. При этом восьмеричные числа должны начинаться с `0`, шестнадцатеричные — с `0x`.

```
env printf "%o(oct)=%d(dec), %x(hex)=%d(dec)\n" 0200 0200
0x200 0x200
```

Результат:

```
200(oct)=128(dec), 200(hex)=512(dec)
```

Рассмотрим структуру управляющей последовательности. Она включает следующие элементы:

```
%[флаги][ширина][точность][размер]тип
```

Из них обязательны только символ `%` и тип.

Флаги бывают следующие: `+` (плюс), `« »` (пробел), `#` (решетка), `0` (ноль), `-` (дефис).

Флаг «плюс» означает, что перед положительным числом обязательно ставится знак `+`, подобно тому, как перед отрицательным ставится `-`.

```
env printf %+f\n 5
+5,000000
```

Флаг «пробел» означает, что, если первый символ результата — не знак (то есть не `+` и не `-`), то на это место ставится пробел.

```
env printf "% f\n 5 -5
5,000000
-5,000000
```

Флаг «решетка» выводит результат в «альтернативной форме»: для типов `g`, `G` сохраняются конечные нули, для типов `f`, `F`, `e`, `E`, `g`, `G` всегда выводится десятичный разделитель, для типов `o`, `x`, `X` перед ненулевым значением подставляются символы `0`, `0x`, `0X`. (Что означают эти буквенные обозначения типов — см. ниже.)

```
env printf %g\n 5; env printf %#g\n 5
5
5,000000
```

```
env printf %1.0f\n 5; env printf %#1.0f\n 5
5
5,
```

```
env printf %X\n 500; env printf %#X\n 500
1F4
0X1F4
```

Флаг «ноль» дополняет результат нулями до ширины, указанной в поле «ширина».

```
env printf %7i\n 5; env printf %07i\n 5
5
0000005
```

Флаг «дефис» выравнивает результат по левому краю в пределах минимальной ширины поля. По умолчанию, он выравнивается по правому краю.

```
env printf %7i\n 5; env printf %-7i\n 5
5
5
```

В последних примерах мы использовали поле «ширина». Если ширина записана двумя числами, разделенными точками, то первое число обозначает общую минимальную ширину, а второе число — количество символов после десятичного разделителя.

```
env printf %7i\n 5; env printf %7.3f\n 5.5
5
5,500
```

Значения ширины могут задаваться звездочками. В этом случае, перед аргументом должны подставляться дополнительные параметры, указывающие значения звездочек.



При выводе строки посредством `echo` есть возможность внедрить в текст консольные команды и в отображенной строке получить результаты выполнения этих команд.



```
env printf "%i\n" 7 5; env printf "%.*f\n" 7 3 5.5
5
5,500
```

Поле «точность» пишется после точки, и указывает на следующее:

- » для типов d, i, o, u, x, X — минимальное число символов всего;
- » для типов a, A, e, E, f, F — минимальное число символов после десятичного разделителя;
- » для типов g, G — максимальное число значащих символов;
- » для типов s, b — максимальное число выводимых символов.

```
env printf "%.7i\n" 5; env printf "%.2f\n" 5.123456; env printf
"%3e\n" 0.0005; env printf "%.3g\n" 5.123456; env printf "%3s\n"
'Hello everybody'
```

Результат:

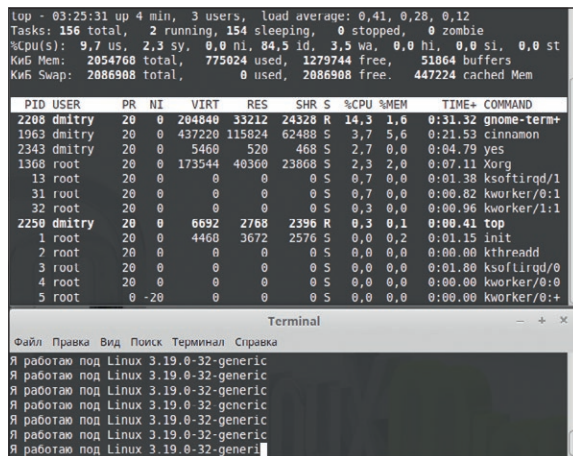
```
0000005
5,12
5,000e-04
5,12
Hel
```

Поле «размер» передает размер числовых данных. Точный тип аргумента не всегда можно определить по его значению. Например, 5 может быть int, а может быть long. Если указать значение поля «размер», то тип аргумента будет приведен к указанному типу. Это поле используется большей частью при программировании на C/C++ (с использованием библиотеки *Glibc*, где форматирование осуществляется точно так же). При работе с утилитой *printf* это поле вряд ли будет использоваться широко. Но, тем не менее, знайте о нем. Его значения могут быть следующие:

- » Нет значения — int, unsigned int
- » l — long int, unsigned long int
- » hh — signed char, unsigned char
- » h — short int, unsigned short int
- » ll — long long int, unsigned long long int
- » j — intmax\_t, uintmax\_t
- » z — size\_t
- » t — ptrdiff\_t
- » L — \_\_int64, unsigned \_\_int64

Наконец, последнее поле «тип» указывает, в каком виде вывести результат:

- » % — просто вывести символ %. Этот тип не принимает ни флага, ни ширины, ни точности, ни размера, всегда используется в виде %%.
- » d, i — вывести как целое число.
- » u — вывести как целое беззнаковое число.
- » f, F — вывести как число с фиксированной точкой (даже если дробная часть будет представлена только нулями).
- » e, E — вывести как число с плавающей точкой (5,000e-04).
- » g, G — вывести как целое число, число с фиксированной или с плавающей точкой, в зависимости от аргумента.



» Рис. 1. Работающая программа *yes* и процесс, который она создала.

- » x, X — вывести как целое число в шестнадцатеричном виде.
- » o — вывести как целое число в восьмеричном виде.
- » s — строка, завершающаяся «нулевым» символом. При этом экранирующие последовательности выводятся «как есть».
- » b — строка, завершающаяся «нулевым» символом. При этом экранирующие последовательности интерпретируются.
- » c — вывести символ.
- » a, A — вывести как число с плавающей точкой в шестнадцатеричном виде.

## yes

Утилита *yes* выводит текстовую строку бесконечное количество раз, пока ее работа не будет прервана извне. Если ввести эту команду без параметров, она будет выводить строки из одного символа *y*.

```
yes
y
y
...
```

Если после *yes* добавить какой-нибудь текст, то будет бесконечно выводиться этот текст. Внутри этого текста можно вставлять команды, обрамленные символами ```.

```
yes Я работаю под `uname` `uname -r`.
Я работаю под Linux 3.19.0-32-generic.
Я работаю под Linux 3.19.0-32-generic.
...
```

Прервать программу можно, например, сочетанием клавиш **Ctrl+C** — это известный нам способ принудительного завершения работающей программы. Но давайте найдем способ поинтереснее. Заодно познакомимся еще с одной программой пакета *Core Utilities* — *kill*.

## kill

При запуске любой программы операционная система создает отдельный процесс, который и выполняет эту программу. Процесс — это единица работы, создаваемая операционной системой. Он должен иметь идентификатор процессора (Pid) и адресное пространство, назначаемое операционной системой.

Чтобы посмотреть, какие же процессы выполняются в операционной системе, воспользуемся программой *top*. Эта программа входит в пакет *procps-ng*, содержащий средства для работы с процессами. Чтобы воспользоваться программой *top*, убедитесь, что пакет *procps-ng* (или его старая версия *procps*) установлен в вашей системе.

Программа *top* показывает текущие процессы в реальном времени, сортируя их по времени использования процессора в порядке убывания (отсюда название “top” — *англ.* вершина). Она имеет псевдографический интерфейс, и на время ее работы консоль станет недоступной. Чтобы остановить работу *top* и вернуться в командную оболочку консоли, надо нажать на клавишу *q*.

Давайте запустим программу *yes* (с любым текстом), затем откроем вторую консоль и запустим в ней программу *top*. В ней мы видим, среди прочих, процесс с названием *yes*. Обратите внимание, что там присутствует и процесс с названием *top*. Ведь эта программа тоже требует отдельного процесса (рис. 1).

Посмотрим на самый первый столбец списка процессов. В нем содержатся идентификаторы процессов (Pid). Как мы видим, Pid процесса программы *yes* — 2343.

Теперь откроем третью консоль и вызовем утилиту *kill*, указав ей в параметре этот идентификатор. (Да-да, и тут приходится использовать *env*.)

```
env kill 2343
```

Результат мы увидим на рис. 2. Выполнение *yes* остановилось, а из списка процессов исчез процесс для *yes*.

Принудительное завершение процесса означает, что ему был послан сигнал **TERM** (завершение). Этот сигнал посылается

по умолчанию. Но *kill* может послать процессу и некоторые другие сигналы. Самые используемые из них — это HUP (обрыв терминальной линии), INT (эмуляция нажатия Ctrl+C), KILL (безусловное завершение), STOP (остановка), CONT (продолжение работы после STOP) и 0 (ничего не делать, просто проверить, имею ли я право посылать сигналы).

Сигнал TERM отличается от сигнала KILL тем, что сигнал TERM может быть обработан программой, которая получила этот сигнал. И, обработав, она может ему не подчиниться. Сигнал KILL такую возможность исключает.

Полный список возможных сигналов можно просмотреть в справочной системе по команде `info coreutils 'Signal specifications'`.

Сигнал, посылаемый процессу, задается с помощью ключа `-s`. Давайте слова запустим программу `yes`, посмотрим, какой теперь будет PId ее процесса (у меня он получился 5271), и отправим этому процессу сигнал остановки. Обратите внимание, что его можно указать полным именем, сокращенным именем или номером: SIGSTOP, STOP или 19.

```
env kill -s STOP 5271
```

Мы увидим, что строки перестали бежать. Процесс остановлен, но не «убит». Если мы сейчас пошлем этому процессу сигнал CONT, строки побегут опять.

```
env kill -s CONT 5271
```

## factor, seq, numfmt

Итак, средства для работы со строками мы сегодня уже разобрали. Рассмотрим теперь утилиты для работы с числами. Первая из них, *factor*, раскладывает целое число на простые множители, и выводит эти множители.

```
factor 12
12: 2 2 3
factor 34 48
34: 2 17
48: 2 2 2 2 3
```

Вторая, *seq*, выводит последовательности целых чисел. Параметры располагаются в следующем порядке: наименьшее число, шаг, наибольшее число. Если параметров только два, то шаг принимается за единицу. Если параметр один, наименьшим числом считается единица.

```
seq 3
1
2
3
seq 15 17
15
16
17
seq 20 3 30
20
23
26
29
```

По умолчанию, полученные числа разделяются символом новой строки. Но с помощью ключа `-s` (`--separator`) можно задать другой символ.

```
seq -s " " 20 3 30
20 23 26 29
```

С помощью ключа `-w` (`--equal-width`) можно сделать все числа одинаковой ширины, заполнив нулями отсутствующие разряды.

```
seq -w -s " " 7 12
07 08 09 10 11 12
```

А ключ `-f` (`--format`) позволяет отформатировать числа тем же способом, который мы рассматривали, разбирая утилиту *printf*.

```
seq -f "%.2f" -s " " 7 12
7,00 8,00 9,00 10,00 11,00 12,00
```

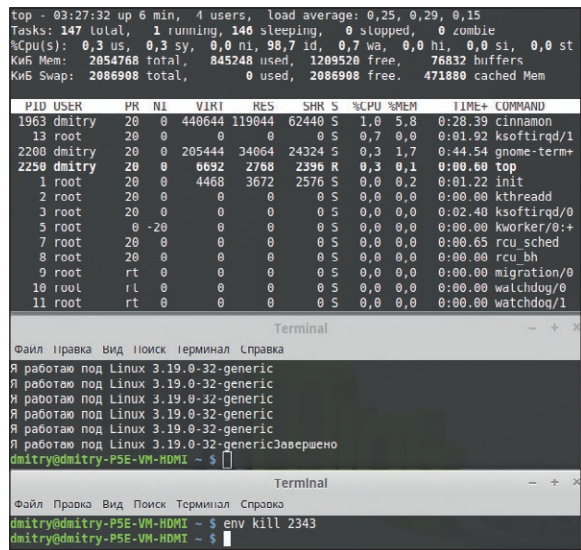


Рис. 2. Результат работы программы *kill*.

Третья утилита, *numfmt*, распознает числа, в которых большие разряды заменены буквенными суффиксами, и/или выводит результат с буквенными суффиксами.

Ключи `--from` и `--to` задают, соответственно, в каких единицах читать аргумент и в какие единицы записывать. Эти единицы следующие: none, si, iec, iec-i, auto.

- » none указывает, что числа вводятся или выводятся без суффиксов. Все разряды пишутся полностью.
- » si указывает, что суффиксы соответствуют степеням числа 1000: K=1000<sup>1</sup> (Kilo), M=1000<sup>2</sup> (Mega), G=1000<sup>3</sup> (Giga), T=1000<sup>4</sup> (Tera), P=1000<sup>5</sup> (Peta), E=1000<sup>6</sup> (Exa), Z=1000<sup>7</sup> (Zetta), Y=1000<sup>8</sup> (Yotta).
- » iec указывает, что суффиксы соответствуют степеням числа 1024: K=1024<sup>1</sup> (Kibi), M=1024<sup>2</sup> (Mebi), G=1024<sup>3</sup> (Gibi), T=1024<sup>4</sup> (Tebi), P=1024<sup>5</sup> (Pebi), E=1024<sup>6</sup> (Exbi), Z=1024<sup>7</sup> (Zebi), Y=1024<sup>8</sup> (Yobi).
- » iec-i указывает, что суффиксы также соответствуют степени 1024, но обозначаются двумя буквами: Ki, Mi, Gi, Ti, Pi, Ei, Zi, Yi.
- » auto используется только при ключе `--from`, указывая, что однобуквенные суффиксы интерпретируются по типу si, двухбуквенные — по типу iec-i.

Покажем на примерах:

```
numfmt --from=none --to=si 345654
346K
numfmt --from=si --to=iec-i 200M
191Mi
```

Ключи `--from-unit` и `--to-unit` задают размер единиц. По умолчанию, их значение равно 1. Но в тех случаях, когда вводимые или выводимые данные сами представлены в других единицах, бывает необходимо задать и другие значения. Например, давайте переведем восемь тысяч пар обуви в общее количество обуток ног:

```
numfmt --from=si --from-unit=2 --to=none 8K
16000
```

Ключ `--grouping` разделяет группы разрядов специальным разделителем, который зависит от локали системы. Для русского языка этот разделитель — пробел. Так гораздо удобнее просматривать большие числа, например, количество байт в двадцати гигабайтах:

```
numfmt --from=iec --to=none --grouping 20G
21 474 836 480
```

Ключ `--suffix` позволяет принимать или задавать какой-нибудь нестандартный суффикс:

```
numfmt --from=iec --to=none --grouping --suffix=" байт" 20G
21 474 836 480 байт
```

Вот мы изучили еще несколько утилит, и жизнь сделалась еще чуть-чуть понятнее и интереснее. Теперь можно немного отдохнуть. Но не очень расслабляйтесь: вас ждет продолжение путешествия. LXF



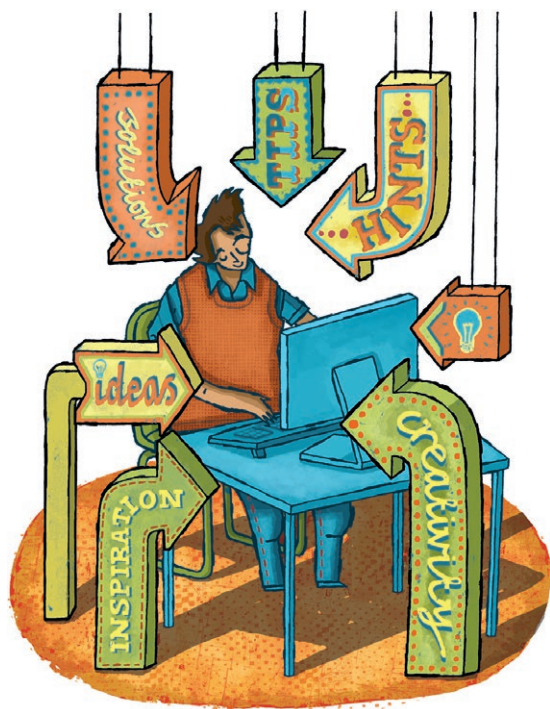
# Терминал: Как начать

Очертя голову, Ник Пирс ныряет в чернильную тьму терминала, чтобы показать вам, как обращаться с командами.



Наш эксперт

Ник Пирс считает терминал уютной и безопасной темной пещеркой, где он прячется после десятилетия работы техническим журналистом.



Терминал является невероятно важной частью рабочего стола Linux. Как бы вы ни привыкли наводить и щелкать в командной строке, в какой-то момент вам придется окунуться в темные воды терминала и задействовать его. Не бойтесь, терминал не так страшен, как его малюют, и уделив время на изучение основ, вы обнаружите, что определенные задачи в нем решаются гораздо быстрее и эффективнее.

Как и следует ожидать, терминал по сути выдает доступ к командной оболочке Linux, то есть работает так же и использует

тот же язык (*Bash*). Это означает, что в терминале можно сделать то, что вы обычно делаете в командной строке, причем не расставаясь с относительно комфортным комфортом рабочего стола. Так что обучение работе с терминалом — и *Bash* — вдвойне выгодно, поскольку дает вам первое представление о работе с базовой оболочкой Linux. И на следующих нескольких уроках вы именно этим и займетесь: как освоить работу с терминалом.

Эта серия уроков основана на Ubuntu, так что для начала откройте Dash и наберите 'terminal' в поле поиска. Терминал вы, естественно, найдете, но заодно увидите и два пункта под названием *UXTerm* и *XTerm*. Это проливает свет на тот факт, что существует несколько эмуляторов терминала, которые можно запускать для взаимодействия с оболочкой. Различия между ними, конечно, есть, но в принципе они делают одно и то же.

На этом уроке мы придерживаемся терминала по умолчанию, который на самом деле является эмулятором *gnome-terminal* — технически это эмуляция сессии TeleType (TTY). Он обладает всей необходимой функциональностью, но стоит отметить и *XTerm* и *UXTerm* как более минималистские инструменты, не требующие для запуска никаких зависимостей. И если основной терминал почему-либо отказывается, взамен можно использовать их. Кстати, единственная разница между ними состоит в том, что *UXTerm* поддерживает расширенный набор символов Unicode.

## Как работает Bash

Оболочка Linux использует оболочку *Bash* и командный язык для выполнения задач, с относительно простым синтаксисом для каждой команды: имя утилиты команда -опция.

Часть команды имя утилиты — это отсылка к инструменту, который вы хотите запустить: например, *ls* служит для просмотра содержимого каталога, а *apt-get* — для запуска инструмента

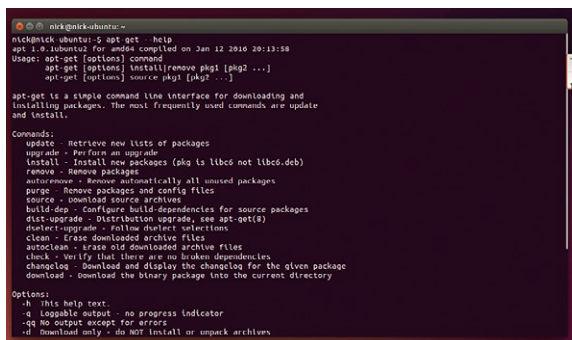
## Ускорение ввода текста

Невзирая на степень вашего умения быстро печатать, командная строка все равно может оказаться удручающе затратной по времени. К счастью, в терминале немало удобных сочетаний клавиш. Давайте посмотрим, как легко получить доступ к ранее использованным командам и увидеть предложения:

- » Стрелки вверх/вниз Просмотр истории команд.
- » history Просмотр истории команд.
- » Ctrl+g Поиск по истории команд. Введите буквы, чтобы сузить поиск до ближайшего совпадения, и придержите клавиши Ctrl+g, чтобы увидеть другие совпадения.
- » Tab Посмотреть подсказку или автозавершение слова или пути, если вы его целиком не помните. Нажатие ~+Tab автоматически вставит имя пользователя, @+Tab — имя хоста, а \$+Tab — переменную.

## Скорая помощь

Если для ввода требуемой команды в окне уже нет места, окно можно очистить, просто введя *clear* и нажав Enter. Обратите внимание, это не повлияет на вашу историю команд.



» Флаг `--help` можно использовать с любой командой, для выяснения, что она делает и какие аргументы использовать.

## Ваши первые команды терминала

Хотя для установки и управления программным обеспечением можно обойтись комбинацией *Software Center* и панели управления *Software & Updates* Ubuntu, часто быстрее воспользоваться семейством инструментов *Advanced Package Tool (APT)*. Вот несколько основных способов их применения (применение `sudo` см. ниже):

- » `$ apt-cache pkgnames` Перечисляет все доступные пакеты в исходниках, указанных в списках источников в файле `/etc/apt/sources.list`.
- » `$ sudo add-apt-repository ppa:<имярепозитория>` Добавляет в список исходников указанный репозиторий Launchpad PPA.
- » `$ sudo apt-get update` Получает свежие списки пакетов (включая обновленные версии) из всех перечисленных репозиториев.
- » `$ sudo apt-get install <пакет>` Устанавливает названный пакет. Также скачивает и устанавливает все требуемые зависимости пакетов.
- » `$ apt-get remove <package>` Удаляет установленный пакет. Используйте `apt-get purge <пакет>`, чтобы удалить все файлы настройки пакета, и `apt-get autoremove`, чтобы удалить уже не нужные пакеты, установленные другими пакетами.
- » `$ sudo apt-get upgrade` Обновляет все установленное программное обеспечение — перед этим запустите `sudo apt-get update`.

Другие полезные команды `apt-get` включают `apt-get check` — инструмент проверки нарушенных зависимостей, и `apt-get autoclean`, который ликвидирует Deb-файлы удаленных пакетов.

```
nick@nick-ubuntu:~$ apt-cache show vlc
Package: vlc
Priority: optional
Section: universe/graphics
Installed-Size: 3765
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original Maintainer: Debian Multimedia Maintainers <pkg-multimedia-maintainers@lists.alioth.debian.org>
Architecture: amd64
Version: 2.1.6-0ubuntu14.04.1
Replaces: vlc-data (<< 1.1.5), vlc-nox (<< 2.0.2)
Provides: mp3-decoder
Depends: fonts-freefont-ttf, vlc-nox (= 2.1.6-0ubuntu14.04.1), libaa1 (>= 1.4p5), libc6 (>= 2.15), libcacao0 (>= 0.99.beta17-1), libfreetype6 (>= 2.2.1), libfribidi0 (>= 0.19.2), libgcc1 (>= 1:4.1.1), libgl1-mesa-glx | libgl1, libqtcore4 (>= 4:4.8.0), libqtgui4 (>= 4:4.8.0), libstdc++6 (>= 4.6), libtar0, libva-x11-1 (>= 1.3.0-1), libvorbis0c (>= 1.2.10), libx11-6, libxcb-composite0, libxcb-keysyms1 (>= 0.3.9), libxcb-randr0 (>= 1.11.2), libxcb-shm0, libxcb-xv0 (>= 1.2), libxcb1 (>= 1.6), libxext6, libxinerama1, libxpm4, zlib1g (>= 1:1.2.3.3)
Pre-Depends: dpkg (>= 1.15.6-)
Recommends: vlc-plugin-notify (= 2.1.6-0ubuntu14.04.1), vlc-plugin-pulse (= 2.1.6-0ubuntu14.04.1), xdg-utils
Suggests: vlcdean-doc
Breaks: vlc-data (<< 1.1.5), vlc-nox (<< 2.0.2)
Filename: pool/universe/v/vlc/vlc_2.1.6-0ubuntu14.04.1_amd64.deb
Size: 1212144
MD5sum: fbb7933ada01d9ccddd319ddea71bd89
SHA1: 4bb0e71315956d97ce18b67d7eeb27ee1b968fbc
SHA256: 636997ae393297dd5afdba39b9cb3a0958b3df724651a47660cc3f5993ca35f
Description-en: multimedia player and streamer
```

» Пакет `apt-cache` также умеет отыскивать указанные пакеты или выявлять зависимости пакета.

управления пакетами *APT*. В части команда вы конкретно указываете, что утилита должна сделать, например, `apt-get install` предписывает утилите управления пакетами выполнить установку именovanного пакета, например: `apt-get install vlc`.

В части -опция можно установить один или несколько параметров-«флажков», чтобы задать определенные параметры. Каждому флагу предшествует один или два дефиса (--), а полезнее всех опция --help, которая дает краткое описание утилиты плюс перечисляет все доступные команды и параметры, например, `ls -l`.

Флаг -l велит инструменту вывода каталога предоставить подробную информацию о содержимом папки, включая: права доступа; владельца файла; время последнего изменения и его объем в байтах. Утилиты можно запускать без каких-либо команд или параметров — например, `ls` сама по себе выдает простой список всех папок и файлов в каталоге. Можно также запустить утилиты с комбинацией команд и/или параметров.

### Ограниченный доступ

Откройте терминал, и вы увидите нечто вроде `username@pc-name:~$`. Это показывает, что вы вошли в оболочку под своей учетной записью. А значит, у вас есть доступ к ограниченному числу команд — можно, например, напрямую работать с `ls`, но нельзя установить пакет с помощью `apt-get`, поскольку эта команда требует прав root. Права достигаются одним из двух способов — если вы пользователь с правами администратора, как пользователь по умолчанию в Ubuntu, то можно предварить вашу команду командой `sudo`, например, `sudo apt-get install vlc`. Вам предложат ввести пароль учетной записи, а затем команда выполнит-ся. Вы обнаружите, что можете и дальше запускать команды через `sudo` без повторного ввода пароля (пять минут), пока терминал открыт. В некоторых дистрибутивах можно войти в терминал пользователя root с помощью `su` — вам предложат ввести пароль пользователя root, после чего вы увидите следующее приглашение: `root@pc-name:~$`. Войдя, вы сможете вводить команды без ограничений.

Мы рекомендуем использовать команду `sudo`, а не этот вариант, и если вы работаете в Ubuntu, то обнаружите, что `su` вообще не работает, поскольку пароль учетной записи root заблокирован по соображениям безопасности.

При установке некоторых дистрибутивов или добавлении новых пользователей в Ubuntu вы можете обнаружить, что ваша учетная запись пользователя по умолчанию в группе `sudo` не добавлена. Чтобы решить эту проблему, откройте терминал в учетной записи, которая действительно имеет доступ root (или, если поддерживается, воспользуйтесь командой `su`) и введите `sudo adduser <имяпользователя> sudo`. Той же командой можно добавить пользователя в другие группы, если перечислить все группы, в которые вы хотите его добавить, например: `sudo adduser <username> adm sudo lpadmin sambashare`.

Еще один удобный инструмент — `gksudo`; он позволяет запускать настольные приложения с привилегиями root. Это наиболее полезно, когда требуется просмотреть систему с правами root с помощью диспетчера файлов: `gksudo nautilus`. Обязательно оставляйте терминал открытым на все время выполнения приложения, в противном случае оно при закрытии терминала закроется тоже. Закончив, закройте окно программы, затем нажмите `Ctrl+c` в терминале, что прерывает запущенную программу и возвращает вас в командную строку.

Мы уже обсуждали флаг --help, но есть и другие инструменты помощи, которые тоже можно использовать. Во-первых, есть `whatis` — который можно вести с любой командой, чтобы получить краткое описание его и любых указанных элементов: например, `whatis apt-get install vlc` опишет инструмент `apt-get`, аргумент `install` [установка] и что такое пакет `vlc`. Флаги игнорируются.

Если вы ищете полноценное руководство, то инструмент `man` предоставляет доступ к онлайн-руководству вашего дистрибутива. Запуск `man intro` выдает большое и подробное введение в командную строку. После этого нажмите `q`, чтобы вернуться в терминал. Для дополнительных советов по навигации по руководству наберите `man man` или в сочетании с именем инструмента, например, `man ls`.

Теперь, когда вы сделали свои первые шаги в мир терминала, посмотрите врезку (*Ваши первые команды терминала*, вверху), где вы найдете ряд полезных для работы команд управления пакетами. На следующем уроке мы рассмотрим перемещение по файловой системе из терминала и запуск программ, и изучим полезные сочетания клавиш, помогающие ускорить взаимодействие с командной строкой. **LXF**



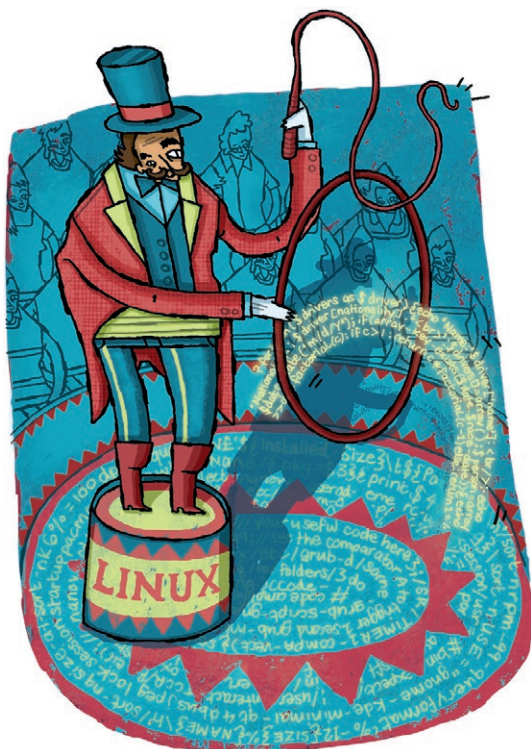
# Ubuntu: Linux на планшете

Ник Пирс глубоко копает, узнавая, как успешно установить рабочую версию Ubuntu на планшет Windows 2-в-1.



Наш эксперт

Ник Пирс недавно купил себе планшет Linux 1010 специально для запуска на нем Linux. Это признак либо большой увлеченности, либо большой глупости — решать вам.



преодолеть ряд затруднений. Во-первых, эти планшеты сочетают 64-битный процессор с 32-битным EFI, а большинство дистрибутивов ожидают 64-битный процессор с 64-битным EFI или 32-битный процессор с традиционным BIOS, так что они не распознают USB-диск при загрузке. Во-вторых, хотя по мере свежих выпусков ядра аппаратная поддержка быстро улучшается, она все-таки из коробки не очень полная. Но не волнуйтесь — если вы временно согласны жить с ограниченными возможностями (ситуация улучшается почти ежедневно), вы можете установить и запустить работающий Linux на планшете с процессором Bay Trail. Вот что надо делать.

Стоит изготовить полную резервную копию вашего планшета в его текущем состоянии, чтобы при необходимости восстановить его исходные настройки. Лучшим инструментом для этого однозначно является бесплатное Windows-приложение под названием *Macrium Reflect Free* ([www.macrium.com/reflectfree.aspx](http://www.macrium.com/reflectfree.aspx)). Установите его на планшет, затем создайте резервную копию всего диска на microSD-карте памяти планшета и после этого создайте предохранительный [failsafe] загрузочный USB-накопитель Macrium для восстановления из резервной копии. Учтите: слот карты памяти microSD не обнаруживается спасательным диском, и чтобы вернуть свой планшет в состояние по умолчанию, вам понадобится USB-чисталка для карт microSD, распознаваемая программой *Macrium*.

Подготовив резервную копию, беритесь за дело. Планшеты Bay Trail очень похожи, но не идентичны, поэтому стоит поискать модель вашего планшета в сочетании с соответствующими терминами ('Linux', 'Ubuntu', 'Debian' и т.д.) и посмотреть, что всплывет. Вы, вероятно, найдете таких энтузиастов, как Джон Уэллс [John Wells] ([www.jfwhome.com](http://www.jfwhome.com)), у которых есть подробные руководства и загружаемые скрипты для запуска Ubuntu на планшете Asus Transformer T100TA, и большая часть оборудования работает. Еще один хороший ресурс — вики DebianOn (<https://wiki.debian.org/InstallingDebianOn>), где вы найдете много других планшетов, снабженных указаниями, что работает и на какие проблемы нужно обратить внимание, и полезными ссылками для получения дополнительной информации.

К сожалению — нашему — для планшета Linux 1010 нет единого удобного инструментария, и нам пришлось немного поэкспериментировать, прежде чем мы нашли лучший способ (см. *Эксперименты с поддержкой Linux*, стр. 60).

## Установка Linux на Linx

Для планшета Linux 1010 мы решили остановиться на Ubuntu. Мы в долгу перед Яном Моррисоном за огромную работу по созданию модифицированной версии Ubuntu (14.04.3 LTS), которая работает не только как liveCD, но и как установщик. Мы экспериментировали с последними релизами Ubuntu — 15.10 и ежедневными

Завидуете быстрому распространению дешевых Windows планшетов 2-в-1? А не запустить ли на таком Linux? Испанский производитель смартфонов BQ, конечно, договорился с Canonical ради продаж планшета Aquarius M10 с предустановленной Ubuntu, но ожидаемая цена — более £200; а зачем переплачивать, когда, оказывается, можно — правда, со значительной настройкой — установить Linux на одно из тех дешевых устройств Windows?

Все эти устройства используют малобюджетный четырехъядерный процессор Intel Atom, известный под общим названием Bay Trail, и нам удалось найти один такой планшет; на нем мы и сфокусировались на нашем уроке. Это устройство — Linx 1010, с процессором Atom Z3735F, 2 ГБ ОЗУ, 32-ГБ встроенной EMMC (плюс слот для дополнительной карты microSD), двумя полноразмерными портами USB и сенсорным экраном с поддержкой мультитач. Его можно приобрести со съемной клавиатурой и трекпадом на площадках типа [www.ebuyer.com](http://www.ebuyer.com) за £150. Эти устройства поставляются с предустановленной Windows 10, но, как вы поймете, на них можно и запустить, и установить разновидности Linux.

В идеальном мире вы бы просто создали живой USB-диск с Linux, подключили его к компьютеру — и вперед; но придется

Скорая помощь



Ян Моррисон [Ian Morrison] проделал огромную работу по созданию версии Ubuntu 14.04.3 LTS для устройств на процессоре Z3735F, таких как Linx 1010. Если вы хотите, чтобы он и дальше развивал свою работу, рекомендуем пожертвовать через его сайт [www.linuxium.com.au](http://www.linuxium.com.au).

## Поддержка оборудования

Каково текущее состояние аппаратной поддержки планшета Bay Trail? Оно, конечно, варьируется от устройства к устройству, но есть и различия. Вот что вы должны искать при тестировании планшета:

» **ACPI** Это касается управления питанием. «Из коробки» оно практически отсутствует, но последние ядра имеют тенденцию поддерживать отображение состояния батареи — Linux, похоже, является здесь исключением из правил. Приостановки и спящего режима следует избегать.

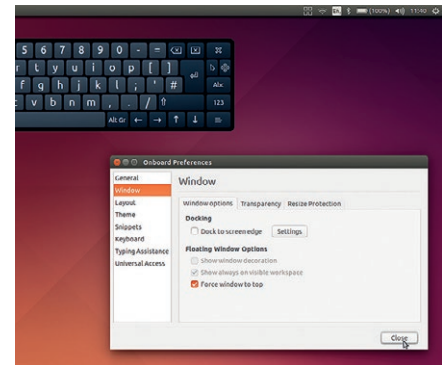
» **Wi-Fi** Последние ядра снова улучшили поддержку, но многие устройства используют беспроводные адаптеры SDIO, не поддерживаемые без заплаток или заказных драйверов вроде найденных на <https://github.com/hadess/rtl8723bs>.

» **Bluetooth** Последние ядра часто нуждаются в заплатках, хотя наш планшет Linux держал связь по Bluetooth, даже когда внутренний адаптер Wi-Fi перестал работать.

» **Звук** Проблема многих планшетов; даже если драйвер распознан и загружен, требуемая прошивка может отсутствовать. Будьте здесь осторожны — есть сообщения пользователей о повреждении звуковых карт при попытке активировать их.

» **Сенсорный экран** Как мы видели, старые ядра их не поддерживают, но обновление до ядра 4.1 или новее должно дать положительные результаты, хотя и с некоторой настройкой.

» **Камера** Здесь мало что сделано до сих пор. В большинстве случаев вам придется подождать появления драйверов.



» Обновите ядро до 4.1 или новее, чтобы Ubuntu работал с сенсорным экраном вашего планшета.

сборками 16.04 — но хотя live-дистрибутивы работают нормально, их установка оказалась невозможной. Впрочем, еще не все потеряно, как вы узнаете попозже. Итак, самый простой и легкий способ установить Ubuntu на планшет с процессором Z3735F — взять неофициальную «официальную» сборку квази-Ubuntu 14.04.3 LTS от Яна. Она поставляется с поддержкой 32-битной UEFI, встроенной в ISO, и включает самодельные драйверы для ключевых компонентов, в т.ч. процессора Z3735F и внутреннего адаптера Wi-Fi. Однако нет поддержки сенсорного экрана, поэтому к планшету придется подключить съемную клавиатуру и тачпад.

Перейдите на [www.linuxium.com.au](http://www.linuxium.com.au) на вашем основном ПК и посмотрите соответствующий пост (от 12 августа 2015 г., но последнее обновление в декабре) в разделе Latest. Нажмите на ссылку Google Drive и выберите ссылку синего цвета Download [Загрузить], чтобы сохранить файл **Ubuntu-14.04.3-desktop-linuxium.iso** в папку **Загрузки**.

После этого берите свежееотформатированную флешку — она должна быть емкостью не менее 2 ГБ и отформатирована в FAT32. Простейший способ создания диска — использовать *UNetbootin*, выбрать флешку, отыскать ISO-образ Ubuntu и создать USB-накопитель. Записав, извлеките накопитель. Подключите его к одному из USB-портов в Linux, затем включите, одновременно удерживая кнопки включения и + уровня звука. Секунд через пять вы должны увидеть подтверждение, что меню загрузки вот-вот появится — когда это произойдет, стукните пальцем по Boot Manager [Менеджер загрузки]. Клавишей курсора выберите пункт EFI USB Device и нажмите Enter, чтобы открыть меню *Grub*. Далее, выберите Try Ubuntu without installing [Попробовать Ubuntu без установки] и снова нажмите Enter.

Вы увидите экран загрузки Ubuntu, и после длительной паузы (и пустоты на экране) должен появиться рабочий стол. Вы должны также получить мгновенное уведомление, что обнаружен внутренний адаптер Wi-Fi — это один из ключевых признаков того, что данный ремикс дистрибутива Ubuntu создавался специально для устройств Bay Trail.

До сих пор вы должны были работать с планшетом в портретном режиме; пора переключить его в более удобный ландшафтный вид, и мы сделаем это, нажав кнопку Settings [Настройки] в правом верхнем углу экрана и выбрав System Settings [Системные настройки]. Выберите Displays [Экраны], задайте в раскрывающемся меню Rotation Clockwise [Вращать по часовой стрелке] и нажмите Apply [Применить] (сама кнопка большей частью за пределами экрана, но приглядевшись, вы обнаружите ее левый кусок сверху экрана).

Затем подключитесь к своей сети Wi-Fi, нажав кнопку беспроводной связи на панели меню, выбрав свою сеть и введя пароль.

Теперь вы готовы дважды щелкнуть по Install Ubuntu 14.04.3 [Установить...] и следовать знакомому мастеру для установки Ubuntu на свой планшет. Вы заметите утверждение установщика, что планшет не подключен к источнику питания, хотя вы должны были сделать это перед установкой — это симптом плохой поддержки ACPI в Linux для этих планшетов.

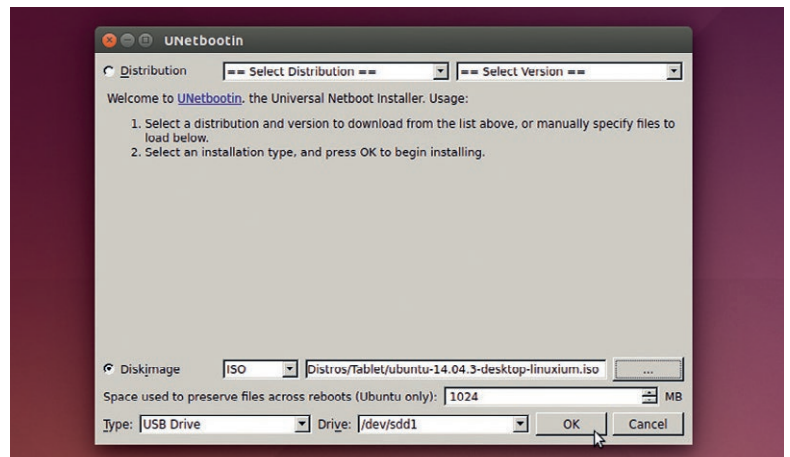
Рекомендуем перед нажатием кнопки Continue [Далее] отметить Download updates while installing [При установке загрузить обновления], после чего вы, вероятно, увидите ошибку ввода/вывода fsync/closing — просто нажмите Ignore [Пропустить], а затем Yes [Да], когда появится запрос на отмонтирование различных разделов.

На экране разбиения диска появится с виду отличная новость — Ubuntu предлагается установить рядом с Windows; но работать это не будет, в основном из-за попытки установиться на карту microSD вместо внутренней памяти. Эта карта не обнаруживается при загрузке, и установка в итоге потерпит неудачу. Вместо этого мы собираемся установить Ubuntu вместо Windows, так что выберите Something else [Другое].

Игнорируйте любые предупреждения о `/dev/sda` — вместо этого сосредоточьтесь на `/dev/mmcblk0`, который находится во внутренней флэш-памяти. Вы увидите четыре раздела — нам необходимо сохранить два первых (Windows Boot Manager и неизвестный) и удалить два раздела NTFS (`/dev/mmcblk0p3` и `/dev/mmcblk0p4` соответственно). Выберите поочередно каждый из них и нажмите кнопку «-», чтобы удалить.

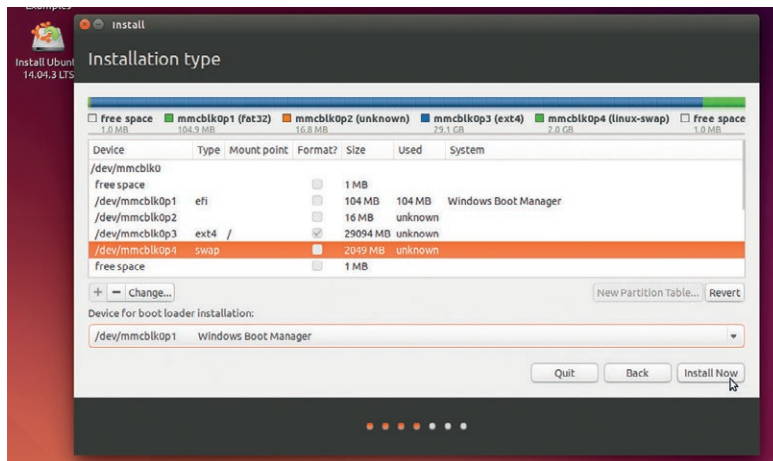
### Скорая помощь

Хотя может быть заманчиво обновить ядро до текущей версии (4.4.1 на момент написания), но могут возникнуть проблемы с тачпадом. Пока придерживайтесь ядра 4.3.3, пока эти проблемы не будут устранены.



» Можно создать себе установочный диск Ubuntu на настольном ПК с помощью утилиты *UNetbootin* — это быстро и (в данном случае) работает эффективно.





➤ При появлении запроса установите разделы вручную — вам надо сохранить исходный раздел EFI.

Далее выберите созданное свободное пространство (31145 МБ или около того) и нажмите кнопку «+». Сначала создайте основной раздел — уменьшите отведенное место на 2048 МБ, оставив их для раздела подкачки, установите точку монтирования /, все остальные параметры оставьте прежними — и жмите ОК. Теперь выберите оставшееся свободное место и снова нажмите кнопку «+». На этот раз установите Use as [Использовать как] в swap area [область подкачки] и нажмите ОК. Наконец, нажмите выпадающее меню Device for bootloader installation [Устройство для установки загрузчика], выберите раздел Windows Boot Manager [Менеджер загрузки Windows] и нажмите кнопку Install Now [Установить сейчас]. Остальной процесс установки должен пройти гладко. По окончании, однако, не нажимайте сразу кнопку Continue testing or Reboot now [Продолжить тестировать или сразу перезагрузить]. Во-первых, надо выполнить важный шаг, делающий вашу копию Ubuntu загружаемой — установить 32-битную версию загрузчика Grub 2. В пошаговом руководстве (см. внизу стр. 61) показан простой способ сделать это с помощью удобного скрипта, любезно предоставленного Яном Моррисоном.

## Аппаратная совместимость

После установки Ubuntu и первой перезагрузки в нее надо будет опять ориентировать рабочий стол как ландшафт через Screen

Display в System Settings. Теперь откройте на своем планшете Firefox и скачайте еще два скрипта с <http://bit.ly/z3735fpatch> и <http://bit.ly/z3735f-dsdt> соответственно. Они улучшают аппаратную поддержку для устройств Linx 1010 с процессором Z3735F Atom, и хотя как будто не расширяют функциональности Linx, но обеспечивают правильную идентификацию процессора.

Нужно chmod оба скрипта, как описано в шаге 2 пошагового руководства по Grub (см. внизу стр. 61), затем установите их один за другим, перезагружаясь после каждой установки. Наконец, загрузите и установите последние предлагаемые обновления Ubuntu.

Вы увидите, что при первом входе в систему экран входа возвращается в режим портрета — не волнуйтесь, режим ландшафта после входа восстановится, и станет можно посмотреть, что есть на вашем планшете, а что не поддерживается. В случае Linx 1010 на данном этапе работает немного. Нет поддержки ACPI, сенсорный экран не обнаружен и нет поддержки камеры или звука (хотя, по крайней мере, найден звуковой чип). Внутренний Wi-Fi, к счастью, поддерживается, как и порты USB, Bluetooth, клавиатура/трекпад и внутренний флэш.

Последние версии ядра должны улучшить совместимость — и нам было любопытно посмотреть, нельзя ли установить на Linx Ubuntu 15.10 или 16.04. Ничего не вышло: сенсорная-то поддержка есть, но нам пришлось вручную добавить файл bootia32.efi в папку EFI\Boot для получения живой среды для загрузки, а установка срывалась на разных этапах, видимо, из-за неполной поддержки внутреннего флэш-накопителя. Мы надеемся, что финальный релиз 16.04 даст больше возможностей, но если вы не в силах ждать и готовы пойти на риск снижения стабильности, читайте далее.

Если вы отчаялись получить поддержку сенсорного экрана для вашего планшета и у вас есть под рукой запасной USB-адаптер Wi-Fi (поскольку обновление ядра нарушает работу внутреннего Wi-Fi адаптера), то обновите ядро до 4.1 или более поздней версии. Мы выбрали ядро 4.3.3 — для его установки введите в терминале

```
$ cd /tmp
$ wget \kernel.ubuntu.com/~kernel-ppa/mainline/v4.3.3-wily/linux-headers-4.3.3-040303_4.3.3-040303.201512150130_all.deb
$ wget kernel.ubuntu.com/~kernel-ppa/mainline/v4.3.3-wily/linux-headers-4.3.3-040303-generic_4.3.3-040303.201512150130_ amd64.deb
```

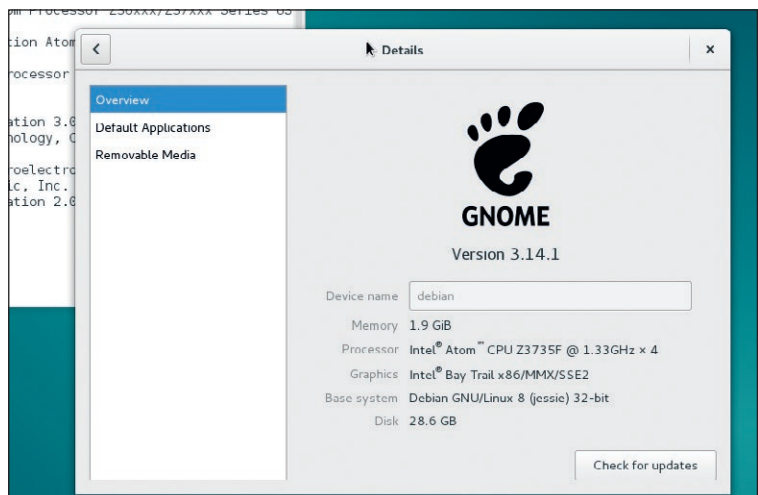
## Эксперименты с поддержкой Linux

Единственный другой дистрибутив, который мы смогли успешно установить на планшет Linx 1010, был Debian Jessie (8.3). Его уникальность в том, что и 32-, и 64-битные версии работают с 32-битным UEFI без каких-либо изменений, но нет поддержки живого режима: придется установить его прямо на жесткий диск.

Поддержка Wi-Fi не предоставляется из коробки — чтобы наша подключаемая карта распознавалась, пришлось добавить несвободный пакет прошивки для флэш-накопителя USB. Аппаратная поддержка была минимальной, хотя обновление ядра до 4.2, по крайней мере, позволило распознать внутренний адаптер Wi-Fi.

Еще мы попробовали Fedlet — ремикс Fedora (<http://bit.ly/fedora-fedlet>) в качестве live-USB, но чтобы загрузиться, пришлось использовать утилиту Windows (Rufus) для создания USB-флешки. Производительность была крайне низкой, а внутренний адаптер Wi-Fi не распознал. Зато работал сенсорный экран.

Нам также удалось загрузка из специализированного Arch Linux ISO, имеющего поддержку SDIO WiFi и 32-битного UEFI. Его можно взять на <http://bit.ly/arch-baytrail>, но его установка быстро обрывается. Версия Porteus с <http://build.porteus.org> запускается и работает, но требует много возни, а затраченные усилия не дали результатов лучше, чем все остальные варианты, которые мы пробовали.



➤ С подключением внешнего адаптера Wi-Fi, установка Debian на нашем планшете Linx 1010 была довольно простым процессом.

```
$ wget \kernel.ubuntu.com/~kernel-ppa/mainline/v4.3.3-wily/linux-image-4.3.3-040303-generic_4.3.3-040303.201512150130_i386.deb
$ sudo dpkg -i linux-headers-4.3*.deb linux-image-4.3*.deb
```

По завершении перезагрузите планшет. Вы обнаружите, что теперь у вас есть поддержка сенсорного экрана при входе в систему (точно, а не мультитач), но после входа экран поворачивается и больше не работает правильно. Мы скоро исправим эту оплошность.

Прежде всего вам надо знать о недостатках. Вы потеряете поддержку внутренней беспроводной карты SDIO (пришлось вставить подручный Wi-Fi-адаптер USB, чтобы вернуть подключение к Интернету), и звук больше не распознается. Также могут быть проблемы со стабильностью, которые можно исправить с помощью шукарского приема настройки *Grub*:

```
$ sudo nano /etc/default/grub
Найдите строку, отмеченную GRUB_CMDLINE_LINUX_DEFAULT, и измените на:
GRUB_CMDLINE_LINUX_DEFAULT="intel_idle.max_cstate=0 quiet"
```

Сохраните файл, выйдите из *nano* и наберите

```
$ sudo update-grub
```

После перезагрузки вероятность зависания системы уменьшится, но учтите, что параметр ядра увеличивает энергопотребление и влияет на заряд батареи. Это позор, поскольку функции ACPI все равно не работают, а значит, параметры питания остаются неточными: заряд батареи всегда оценивается в 100%, даже когда это явно не так.

## Наладка сенсорного экрана

Далее давайте заставим нормально работать сенсорный экран. Сначала определим его тип, используя *xinput*. В случае с Linx 1010 это емкостной сенсорный экран Goodix. Нам нужно заставить сенсорный экран поворачивать свою матрицу, когда поворачивается дисплей, чтобы она работала и в портретном, и в ландшафтном режимах. Сделать это поможет команда *xinput*:

```
xinput set-prop "Goodix Capacitive TouchScreen" 'Coordinate Transformation Matrix' 0 1 0 -1 0 1 0 0 1
```

Теперь сенсорный экран должен правильно работать в горизонтальном режиме ландшафта. Пока вам потребуется делать это вручную при каждом входе в Ubuntu, а сенсорный экран перестанет работать, если вы повернете его в режим портрета. Если вы хотите иметь возможность поворачивать и изображение, и сенсорный экран вместе, надо адаптировать скрипт **rotate-screen.sh** с <http://bit.ly/RotateScreen> (переключитесь в вид Raw [Необработанный]), затем щелкните правой кнопкой мыши и выберите **Save page as** [Сохранить страницу как], чтобы сохранить его на свой планшет. Затем откройте скрипт в *Gedit* или *nano* и внесите изменения в следующие строки:

```
TOUCHPAD="pointer:SINO WEALTH USB Composite Device"
TOUCHSCREEN="Goodix Capacitive TouchScreen"
```

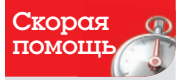
Сохраните и выйдите, затем запустите скрипт:

```
$ ./rotate_desktop.sh <option>
```

Замените *<option>* на *normal* (портрет), *inverted*, *left* или *right*, чтобы повернуть и изображение, и матрицу сенсорного экрана. Перед запуском скрипта надо сначала отменить текущий поворот экрана с помощью *Screen Display* — восстановить вид по умолчанию, а затем запустить *./rotate\_desktop.sh* вправо, чтобы совместить сенсорную панель и сенсорный экран.

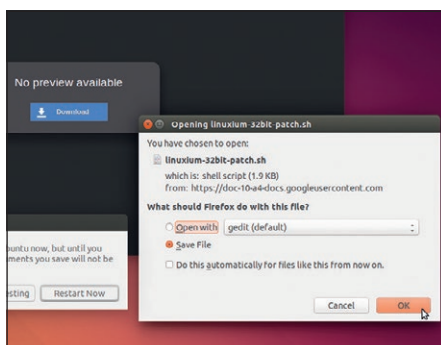
Теперь советуем создать скрипт, запускаемый при загрузке: откройте *dash* и наберите *startup* для запуска Startup Applications [Приложения, запускаемые при загрузке]. Нажмите **Add** [Добавить]. Введите подходящее осмысленное название, нажмите **Browse** [Просмотр], найдите и выберите свой скрипт — когда закончите, щелкните в поле **Command** [Команда] и убедитесь, что добавили *right* в конец скрипта. Нажмите **Save** [Сохранить], перезагрузитесь — и после входа в систему обнаружите, что планшет и сенсорный экран прекрасно работают с внешней клавиатурой и тачпадом.

Вы успешно установили Ubuntu на свой планшет Bay Trail. Что дальше? Следите за последними обновлениями ядра и форумами, чтобы увидеть, когда предприимчивые люди найдут обходные пути и хитрости, необходимые для нормальной работы аппаратной части планшета. Ну, а мы пойдем смотреть, удастся ли заставить снова работать встроенный звук и Wi-Fi, а затем займемся настройками ACPI... **LXF**



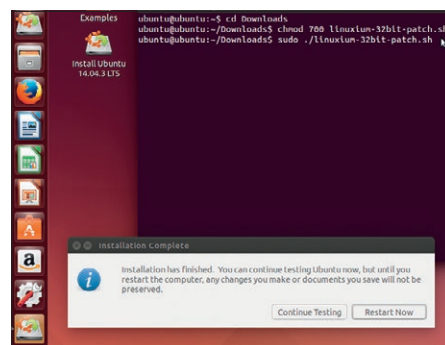
Откройте вкладку **Settings > Universal Access > Typing** [Настройки > Универсальный доступ > Набор] и сдвиньте переключатель **On Screen Keyboard** [Экранная клавиатура] в положение **On** [Вкл], чтобы он запускался с Ubuntu автоматически. Далее откройте **Onboard Settings** [Автономные настройки] через *dash* и отметьте **Start Onboard Hidden** [Запуск скрытых настроек], а также настройте клавиатуру на свой вкус. Теперь вы легко получите доступ к сенсорной клавиатуре через меню состояния.

# Установка 32-битного загрузчика Grub



## 1 Скачайте установочный скрипт

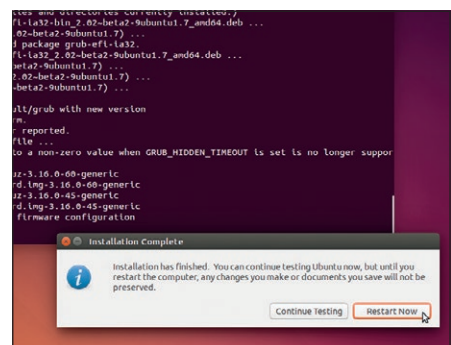
По завершении установки Ubuntu на планшет оставьте на экране диалог с предложением продолжить повторное тестирование или перезагрузить ваш ПК. Затем откройте браузер *Firefox* и перейдите на <http://bit.ly/grub32bit>, откуда вас перенаправят на страницу загрузки Google-диска. Нажмите **Download**; скрипт **linuxium-32bit-patch.sh** сохранится в папке **Downloads**.



## 2 Установите скрипт

Полученный файл **linuxium-32bitpatch.sh** — это скрипт, который автоматизирует процесс установки 32-битной версии загрузчика *Grub 2*. Теперь вам необходимо нажать комбинацию клавиш **Ctrl+Alt+T** и ввести следующие команды:

```
$ cd Downloads
$ chmod 700 linuxium-32bit-patch.sh
$ sudo ./linuxium-32bit-patch.sh
```



## 3 Перезагрузите ПК

Вы увидите, как некоторое количество пакетов автоматически загружается и устанавливается, что впоследствии позволит 32-битному UEFI планшету распознать загрузчик *Grub* и автоматически запускать Ubuntu при включении. Нажмите кнопку **Restart Now** [Перезагрузить сейчас], чтобы завершить процесс, и дождитесь, пока ваш компьютер впервые перезагрузится в Ubuntu.



# Portage: Сеты и оверлеи

Нейл Ботвик углубляется в управление Gentoo и Portage...



Наш эксперт

У Неила Ботвика море опыта по части загрузки, ведь у него в каждой комнате по компьютеру, а вот в перезагрузке, после перехода на Linux, он не силен.



**В** LXZF208 мы рассмотрели, как выжать побольше из *Portage*, менеджера пакетов Gentoo. Тогда мы описали самое основное, а теперь разберемся с другими функциями, чтобы вы могли по максимуму использовать гибкость Gentoo. Дерево портежей [portage tree] содержит множество программ: на момент написания статьи, это 39394 сборки [ebuild] для 19120 пакетов (большинство пакетов в дереве имеют более одной версии). Это очень много, но это не значит, что там есть всё. Иногда вам надо будет установить то, чего нет в дереве портежей, и здесь вам понадобятся оверлеи [overlay — *англ. зд.* приложение]. Оверлей — это просто дополнительный репозиторий к дереву портежей, наподобие PPA в Ubuntu. Отличие заключается в том, что оверлей легко сделать самому. Имея сборку пакета — возможно, скачанную с сайта Gentoo Bugzilla, или предоставленную вашей программой — вы можете

```

tmacedo [git] (git://github.com/tmacedo/portage-git
tocaro [git] (git://anonigt.gentoo.org/user/tocaro.git, https://anonigt.gentoo...
TomaJ [git] (git://anonigt.gentoo.org/dev/TomaJ.git, https://anonigt.gentoo...
toolchain [git] (git://anonigt.gentoo.org/proj/toolchain.git, https://anonigt.gent...
torbrowser [git] (git://github.com/Meister/torbrowser-overlay.git, https://github.com/...
too-overlay [git] (git://github.com/Too/gentoo-overlay-too.git, https://github.com/...
tranquility [git] (git://anonigt.gentoo.org/epo/dev/tranquility.git, https://anonigt.gent...
trash [git] (https://github.com/batolsman/trash-overlay.git
triqueta [git] (git://anonigt.gentoo.org/user/triqueta.git, https://anonigt.gent...
tristelume [git] (https://github.com/tristelume1/tristelume-overlay.git
trollerlay [git] (https://github.com/fatroll/trollerlay.git
tryton [git] (https://hg.tryton.org/tryton-overlay
twister [git] (https://github.com/dorians/gentoo-twister-overlay.git, git://gi...
twitch3ss [git] (git://github.com/twitch3ss/ebuilds.git
ulm [git] (git://anonigt.gentoo.org/epo/dev/ulm.git, https://anonigt.gentoo...
ultrabug [git] (git://anonigt.gentoo.org/dev/ultrabug.git, https://anonigt.gentoo...
underlay [git] (git://anonigt.gentoo.org/user/underlay.git, https://anonigt.gent...
undesktop [git] (https://github.com/undesktop/undesktop-overlay.git, git://github.com/...
unity-gentoo [git] (git://github.com/sh2nix/unity-gentoo.git
v-fec [git] (git://github.com/v-fec/gentoo_overlay.git, http://github.com/v-f...
vaca [git] (git://github.com/hashtash/gentoo-vaca-overlay.git
vala [git] (https://code.google.com/p/vala-overlay/
vapoursynth [git] (git://github.com/ere/vapoursynth-portage.git
vayexr [git] (https://github.com/vayexr/vayexr-gentoo.git, git://github.com/va...
vdr-devel [git] (git://anonigt.gentoo.org/proj/vdr-devel.git, https://anonigt.gent...
vdr-testing [git] (git://anonigt.gentoo.org/proj/vdr/testing-git, https://anonigt.g...
vincent [git] (git://anonigt.gentoo.org/dev/vincent.git, https://anonigt.gentoo...
virtualization [git] (git://anonigt.gentoo.org/proj/virtualization.git, https://anonigt...
vmacs [git] (git://anonigt.gentoo.org/user/vmacs.git, https://anonigt.gentoo...
vmware [git] (git://anonigt.gentoo.org/proj/vmware.git, https://anonigt.gentoo...
voip [git] (git://anonigt.gentoo.org/proj/voip.git, https://anonigt.gentoo.o...
vortex [git] (git://github.com/n8b8tght/vortex-overlay.git, https://github.com/...
voyageur [git] (https://cfehalli.fr/git/voyageur-overlay/
    
```

» Оверлеев доступно множество. Помеченные зеленой звездочкой поддерживаются официально.

добавить его в свой локальный оверлей, чтобы он стал доступным *Portage*. Оверлей — это обычный каталог, имеющий тот же формат, что и дерево портежей. Создайте каталог — чаще всего, по умолчанию, это `/usr/portage/local`, но может быть и любое другое место на вашем жестком диске или в сети; затем создайте файл `/etc/portage/repos.conf/local.conf` с таким текстом:

```

[local]
location = /usr/portage/local
priority = 100
auto-sync = No
    
```

Расстановка приоритетов используется, если одна и та же версия пакета содержится в нескольких оверлеях, включая главное дерево. Настройка `auto-sync` определяет, что происходит при запуске `emerge --sync`; в случае локального оверлея — ничего. Структура каталогов в оверлее та же, что и в основном дереве; также должны использоваться те же категории каталогов, с поддиректориями для каждого пакета, которым сборка затем присвоит имена по названию программы и номеру версии. Так что если у вас есть сборка для `foo 1.0`, то она будет сохранена как `/usr/portage/local/app-misc/foo-1.0.ebuild`. Заглянув в главное дерево портежей, вы увидите, что для каждого пакета есть файл Manifest, содержащий контрольные

## Сеты

Мы упоминали сеты в прошлом месяце, говоря о `@system` и `@world`, но есть и другие. Если вы устанавливаете новое ядро, любые сторонние модули (такие как Nvidia и некоторые беспроводные драйверы), также необходимо обновить. Все эти драйверы объединяются в сет при установке, так что вы можете объединить их все с помощью

```
$ emerge @module-rebuild
```

Аналогично, `@x11-module-rebuild` делает то же самое для любых графических драйверов после обновления *X.Org*. Вы также можете составлять собственные сеты, что весьма полезно, если вам надо установить одни и те же приложения на несколько машин.

Создайте файл в `/etc/portage/sets` и добавьте туда список пакетов. У нас есть сет под названием `base`, куда входят программы, необходимые на любой новой системе, такие как `eix` или `conf-update`. Мы копируем его в эту систему и затем выполняем `emerge @base`, чтобы установить все программы.

суммы для защиты от несанкционированного доступа. Чтобы сгенерировать такой для вашей сборки, выполните

```
$ ebuild /usr/portage/local/app-misc/foo-1.0.ebuild manifest
```

Часто оверлей используется для добавления сборки, когда новая версия программы уже вышла, но в дереве портежей ее еще нет. Так что если *foo-1.0* в дереве есть, а новая *foo-1.1* отсутствует, можно выполнить

```
$ cp -a /usr/portage/app-misc/foo /usr/portage/local/appmisc
$ mv /usr/portage/local/app-misc/foo-1.0.ebuild /usr/portage/local/app-misc/foo-1.1.ebuild
$ ebuild /usr/portage/local/app-misc/foo-1.1.ebuild manifest
$ emerge -1a foo
```

Хотя свои оверлеи создавать удобно, существует куча готовых, которые вы можете добавить. Для этого используется инструмент *layman*, который сначала надо вызвать. Вы можете получить список оверлеев с помощью `$ layman --list`, и добавить любой к себе на компьютер через `$ layman --add overlay-name`. Чтобы просмотреть установленные оверлеи, используйте `$ layman --list-local`. При добавлении нового оверлея в `/etc/portage/repos.conf/layman.conf` приписывается строка, а содержимое скачивается на вашу систему и обновляется, когда вы выполняете `emerge --sync`.

Одни оверлеи предоставляют добавочные пакеты, другие требуются для тестирования новых версий, прежде чем те будут добавлены в дерево — например, можно попробовать оверлей последней бета-версии Gnome или KDE. Эти оверлеи считаются официальными и проходят тот же контроль качества, что и основное дерево, но в *layman* также присутствует немало неофициальных оверлеев. При добавлении такого вы получите предупреждение. Некоторых оверлеев в *layman* пока нет; чтобы добавить один из них, надо создать файл в `/etc/portage/repos.conf`, как вы это делали для локального.

## Поиск программ

Итак, в дереве портежей есть богатый выбор программ, плюс все эти доступные оверлеи; как же понять, что и где искать? Команда `emerge --search`, как следует из названия, выполняет поиск по дереву портежей — но очень медленно. Одна из первых утилит *Portage*, которую вам нужно установить — это *eix*, которую месяц назад мы вскользь упоминали. Она использует базу данных доступного ПО, которую следует держать в актуальном состоянии, выполняя `eix-update` после каждой синхронизации. Это гораздо быстрее, чем `emerge --search`, а если потом запустить `$ eixremote update`, можно скачать и добавить в базу содержимое оверлеев *layman*. Тогда поиск по дереву портежей можно будет осуществлять посредством `$ eix foo`, а по дереву и всем оверлеям — с помощью `$ eix -R foo`.

Утилита *eix* имеет множество всяких параметров, и это, увы, означает, что с тап-страницами придется повозиться. Наиболее важными из параметров являются: `-c` для компактного, в одну строку, вывода для каждого пакета, `-A` для поиска имен категорий, `-S` для поиска описания пакетов и `-s` для поиска имен пакетов. Последний используется по умолчанию, если другие не указаны. Например, чтобы найти пакеты, в названии или описании которых присутствует 'apache', и получить компактный вывод результатов, поскольку таких много, воспользуйтесь `$ eix -csS apache`. Заданная строка поиска по умолчанию интерпретируется как регулярное выражение, а если вам нужны только точные совпадения, добавьте `-e`. Вывод *eix* сообщит вам о: доступных версиях; доступных флагах USE; номере установленной версии; используемых в ней флагах USE; а также дату установки — всё это вам может пригодиться для устранения неполадок.

Для новичков проблемной областью может стать управление файлами конфигурации. Допустим, у вас установлена *foo-1.0* (да, это очень распространенная программа), а во время общего обновления станет доступной и установится *foo-1.1*. В программе *foo-1.0*

## Смешанный архив

Gentoo, по сути, содержит два дерева портежей в одном: стабильное и тестовое. Если вы используете стабильное дерево, указав `ACCEPT_KEYWORDS=amd64` в `make.conf`, вы получите проверенную и протестированную сборку, но не самую свежую. При выборе `-amd64` это будет тестовая сборка. Однако отчасти возможно их соединить. Допустим, у вас стабильная

система, но вам надо, чтобы она была еще и актуальной. Просто добавьте пакет в `etc/portage/package.accept_keywords`. Как и в случае с `package.use` — и другими подобными файлами — это может быть и отдельный файл, и каталог, где таковых множество. Соединение тестового и стабильного контента способно вызвать проблемы, и лучше прибегать к этому пореже.

в `/etc/foo.conf` есть файл конфигурации, который вы долго и старательно настраивали. Но в *foo-1.1* уже другой файл конфигурации, поскольку параметры там немного другие. Если *Portage* не тронет старый файл, не появится новшества, а если установить новые значения по умолчанию, вас это не обрадует. Поэтому *Portage* устанавливает новый файл как `/etc/_cfg000foo.conf` и по завершении процесса выдает предупреждение, что файлы конфигурации необходимо обновить. Как правило, это делается с помощью инструмента *etc-update*, который выдает вам список доступных для обновления файлов конфигурации, показывает, чем они отличаются, и позволяет выбрать, какой вы хотите использовать или отредактировать.

Сделать это можно либо с помощью *dispatch-conf*, также входящего в состав *Portage*; либо *cfg-update*, у которого есть вариант с графическим интерфейсом, либо *conf-update*, на котором остановились мы. Что бы вы ни выбрали, сообщения об обновлении файлов конфигурации важно не упустить, в противном случае у вас будут проблемы.

Основной файл конфигурации — `/etc/portage/make.conf`, его вы отредактируете во время установки. Тут можно изменить пути, используемые *Portage* — к примеру, многие люди предпочитают не хранить большие объемы данных в `/usr`; и лучше заменить `DISTDIR` на другой адрес, чтобы не захламлять `/usr/portage/distfiles` своими загрузками. Если у вас в сети несколько компьютеров на Gentoo, неплохо будет сделать общую папку NFS, чтобы файл скачивала только та система, которой он требуется. У настройке `FEATURES` свои параметры. Если вы добавите в список `buildpkg`, *Portage* сохранит двоичные пакеты устанавливаемых файлов, которые потом можно установить по `emerge -k package`. Так гораздо проще откатиться к предыдущей версии, если с новой возникнут проблемы. **LXF**

```

# /etc/rc.conf
##+ /etc/_cfg0000_rc.conf      2016-01-29 12:32:07.000000000 +0000
## -13,7 +13,7 00
# boot so you can choose to start specific services. Set to "NO" to disable
# this feature. This feature is automatically disabled if rc_parallel is
# set to YES.
+rc_interactives="YES"
+rc_interactives="YES"

# If we need to drop to a shell, you can specify it here.
# If not specified we use $SHELL, otherwise the one specified in /etc/passwd,
## -27,7 +27,7 00
# both will be started, but services that depend on 'net' will work if either
# one comes up. With rc_depend_strict="YES" we would require them both to
# come up.
+rc_depend_strict="YES"
+rc_depend_strict="YES"

# rc_hotplug controls which services we allow to be hotplugged.
# A hotplugged service is one started by a dynamic dev manager when a matching
## -48,7 +48,7 00
# /var/log/rc.log
# NOTE: Linux systems require the devfs service to be started before
# logging can take place and as such cannot log the sysinit runlevel.
+rc_logger="YES"
+rc_logger="YES"

# Through rc_log_path you can specify a custom log file.
# The default value is: /var/log/rc.log
## -168,7 +168,7 00
#
# MOST: "stdin"
Press 'q' to quit, 'h' for help, and SPACE to scroll.

```

► *Conf-update* — и другие менеджеры конфигурации — показывают вам изменения, вносимые в файлы, и вы можете их принять, отклонить или изменить.



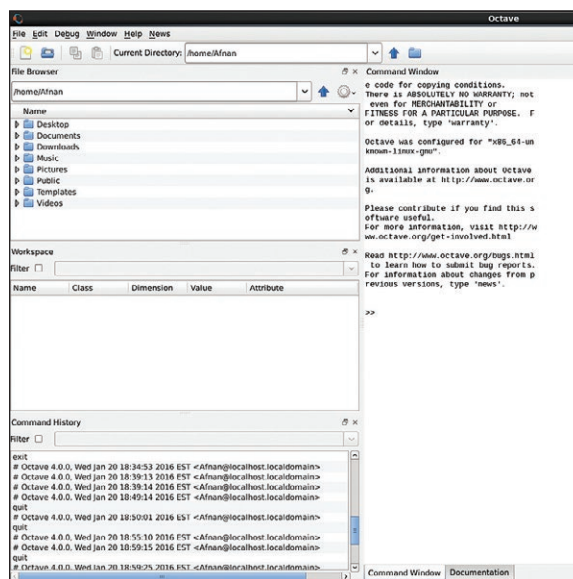
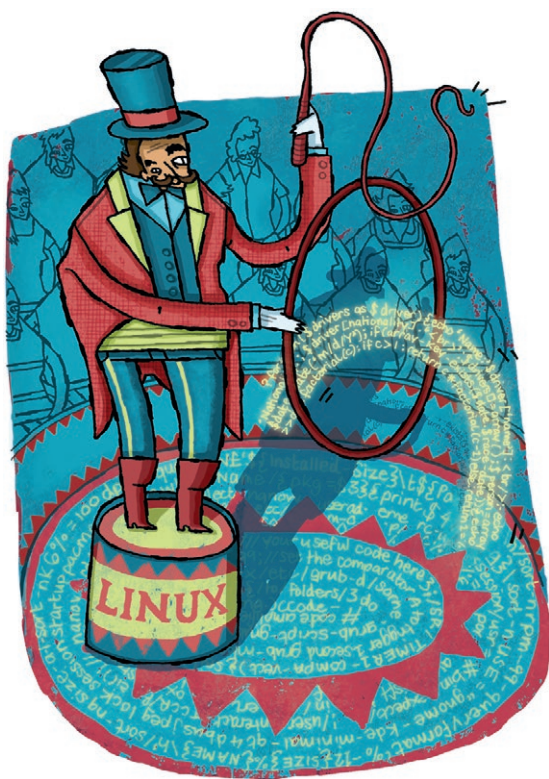
# Octave 4: Будем строить графики

Афнан Рехман визуализирует данные с помощью графического интерфейса пользователя в Octave 4 и его новых функций, уже ставших стандартом.



Наш эксперт

Афнан Рехман любит разбираться и ковыряться в настройках, да и вообще занудствовать по части Linux и других систем. Сейчас он добился до описания своей деятельности.



» GUI Octave, введенный в версии 3.8, стал нормальным интерфейсом пользователя, заменив устаревшую командную строку и сделав программу удобнее для обычных людей.

совместимость с *Matlab*, гарантировав работу ваших программ и в *Matlab*, и в *Octave*. Как же мне добыть эту славную программу, спросите вы? Некоторые дистрибутивы Linux включают ее в свои репозитории ПО, что очень упрощает процесс. Если вы, как и я, любите повозиться с построением из исходника, зайдите на официальную страницу GNU Octave на [www.gnu.org/software/octave](http://www.gnu.org/software/octave) и скачайте сжатые исходные файлы.

## Компиляция Octave

Мы будем собирать *Octave* из исходника в CentOS 6.6, урезанной платформе чисто для ковыряния с настройками. *Octave* версии 3.4.3 там уже есть, но старье нас не интересует. Скачаем свежесжатый TAR.GZ и распакуем папку, чтобы получить файлы внутри нее. Это можно сделать из терминала, командой `tar -xvf filename.tar.gz`. Большинство дистрибутивов также поставляется с менеджером архивов, который извлечет эти файлы за вас.

После этого перейдите туда, где расположатся файлы *Octave*, по команде `cd`. Оказавшись в главной папке, скомандуйте `./configure` для создания файла конфигурации, который поможет в сборке программы. Если при создании файла конфигурации возникли ошибки, убедитесь, что у вас установлены все необходимые пакеты ПО и библиотеки. Командная строка скажет вам, чего не хватает. Мы обнаружили, что в базовую установку

Бывают моменты, когда человек спрашивает себя: выполнять расчет или не выполнять? Если вы — некто вроде автора перед финалом инженерного проекта, сдача которого через несколько часов, и тремя чашками кофе на вашем столе, то ответ — да. И в такие моменты вы обращаетесь к *Octave*, прекрасной программе, которая превратит неряшливые строки нацарапанных на салфетке уравнений в визуальное пиришество.

*Octave* — программа с открытым кодом, вызванная к жизни необходимостью управлять данными и визуализировать их в функциональной, но простой в использовании среде. Будучи основана на языке GNU Octave, она очень похожа на популярный *Matlab*, но без столь значительной цены.

*Octave 3.8.0* вышла в декабре 2013 г. вместе со своим экспериментальным — но столь необходимым — графическим интерфейсом пользователя (GUI), который приблизил *Octave* к превращению в простое для освоения приложение. В мае 2015 г. релиз *Octave 4.0.0* вышел с новым официальным и завершенным GUI, ставшим частью ПО, так что больше не надо запускать его после загрузки программы вручную. Эта версия также значительно улучшила

Скорая помощь

При установке убедитесь, что ваш дистрибутив Linux находится в актуальном состоянии и что все предыдущие версии *Octave* были загодя удалены командой `yum remove`.

CentOS не включено довольно много библиотек, таких как *BLAS* (Basic Linear Algebra Libraries) и *LAPACK* (Linear Algebra PACKage), и надо добавлять их вручную с помощью команды `yum` либо инструмента `Add/Remove Software`. По завершении этого процесса введите `make`, чтобы запустить сборку. Это займет некоторое время — вы успеете употребить две фирменные чашки чая от *Linux Format*. Наш компьютер извел на этот этап более получаса (мы еще и посмотрели небольшое телешоу). Когда все закончится, останется только набрать `make install`, это свернет процесс построения и установки действительно быстро. Поздравляем, теперь вы готовы пользоваться *Octave 4.0!*

Заполучив *Octave*, давайте устроим небольшой обзор GUI, одного из ярчайших нововведений этого ПО. Он очень похож на экспериментальный GUI, имевшийся в версии 3.8; в интерфейсе по умолчанию большую часть экрана занимает командное окно. Как и в предыдущих версиях, GUI также включает знакомое утверждение «гарантия отсутствует» от создателя, прямо под копирайтом. Слева вы найдете проводник по файлам, рабочее пространство и вкладки истории команд. Вкладка рабочего пространства предоставляет подробную информацию обо всех переменных, используемых в данный момент в любом запущенном вами скрипте. Это бывает полезно при отладке вашей программы, поскольку можно видеть значения переменных на каждом шаге программы. Другая полезная вкладка — `Command History`, она покажет, какие команды использовались ранее; это очень помогает при отладке скрипта или функции.

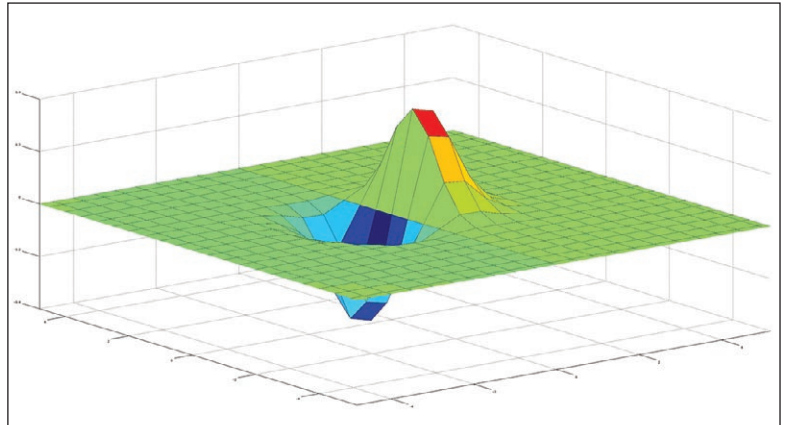
## Основы

Ну вот, вы узнали, как сюда добраться; пора изучить основы. В самом примитивном случае программа может применяться как обычный калькулятор. Ввод чего-нибудь вроде `10+20` выдаст `30` в командной строке, стоит вам нажать `Enter`. Можно также задавать значения переменных и использовать эти переменные для чуть более сложных вычислений, например:

```
>> a = 14/2
>> a = 7
>> b = 13/8
>> b = 1.6250
>> a - b
>> ans = 5.3750
```

Вы можете заметить, что командное окно выдает значение каждой переменной, которое вы вводите. И поскольку это не влияет на окончательный результат, такое засорение вам ни к чему. Чтобы прекратить это, просто добавьте точку с запятой в конце присвоения переменной, чтобы подавить вывод.

*Octave* также предоставляет множество таких же математических операторов, которые вы найдете в *Matlab* или в других



► Команда `surf` преобразит ваш 3D-график, натянув на сетку симпатичную цветную поверхность.

математически ориентированных языках: `sin`, `cos`, `tan`, `max`, `min`, `pi` и т. п. Полный список вы всегда можете найти в Интернете или получить его, просто набрав `help log` в командной строке. Но где *Octave* действительно блещет, так это в области матриц и векторов. Создать вектор не сложнее, чем написать в квадратных скобках последовательность чисел, разделенных пробелами. Чтобы создать матрицу с несколькими строками, просто разделите строки точкой с запятой, например, `>> [1 2; 3 4]`.

Создастся матрица  $2 \times 2$  с верхней строкой, содержащей числа 1 и 2, и нижней строкой, содержащей числа 3 и 4. Однако иногда нужно создать большой набор данных для ваших вычислений. Команда `linspace()` — полезный инструмент, где вы можете указать первое и последнее число массива, а также желаемое количество элементов, и *Octave* создаст соответствующий по длине вектор с равноотстоящими элементами.

Эти векторы и матрицы также могут употребляться в математических операциях. В основном *Octave* выполняет эти операции поэлементно. Это означает, что если вы попросите *Octave* перемножить два вектора, при условии, что они одинаковой размерности, *Octave* умножит первый элемент первого вектора на первый элемент второго вектора, затем второй элемент первого вектора на второй элемент второго вектора, и так далее. Ключевое синтаксическое отличие между поэлементным умножением и перемножением векторов или матриц — это точка, добавляемая перед знаком операции. Проиллюстрируем это на следующем примере:

```
>> x = [1 2 3];
>> y = [4 5 6];
>> x .* y
>> ans = [4 10 18]
```

## Скорая помощь

При написании скриптов отличной идеей является добавление комментариев к вашему коду, с пояснениями, что вы делаете. Это поможет вам и другим людям лучше понять код.

## Octave и Matlab

Мы много говорили о совместимости *Octave* с *Matlab*, но что именно совместимо, а что отличается? Много ли вы сумеете сделать, прежде чем упресть в стену, разделяющую эти два приложения?

Начнем с совместимости. *Octave* был разработан ради хорошего сотрудничества с *Matlab*, и, следовательно, имеет с ним много общих функций. Оба используют матрицы как фундаментальный вид данных; имеют встроенную поддержку комплексных чисел; и у обоих есть мощные встроенные

математические функции и библиотеки. Подавляющая часть синтаксиса пересекается, и открытие файла скрипта в одной программе даст тот же результат, что и открытие в другой.

Есть несколько умышленных, но незначительных дополнений синтаксиса со стороны *Octave*, которые потенциально способны вызвать проблемы. Одно из них — то, что для определения строк в *Octave* символ `"` может использоваться так же, как и символ `'`. Другой пример — комментарии в коде *Octave*

могут начинаться и символом `#`, и символом `%`. Помимо этих косметических различий, есть куда более значимые изменения, такие как цикл с условием завершения (`do-until`), который на данный момент в *Matlab* отсутствует. С другой стороны, в *Matlab* множество функций, отсутствующих в *Octave*, что заставит *Octave* выдать паническое сообщение об ошибке невыполнимости, когда программа не будет иметь представления, как с функцией поступить.

» Подпишитесь на печатную или электронную версии на [www.linuxformat.ru/subscribe!](http://www.linuxformat.ru/subscribe!)



Теперь немного о том, как отображать ваш вывод. Для простого вывода, как в примере выше, есть отличный способ: просто не подавляйте вывод и сразу получите отображаемый результат автоматически. Однако, став искуснее в использовании *Octave*, вы приметесь за более сложные вычисления, и подавление вывода превратится в необходимость. Здесь пойдут в дело команды `disp` и `printf`. Команда `disp` является самой основной доступной командой отображения, и она просто выведет значение любой переменной, которую вы укажете, как если бы вы просто ввели переменную в окно команд. Однако имя переменной не отображается.

Команда `printf` значительно более мощная, чем `disp`; она и сложнее в использовании. С помощью `printf` можно точно определить, как должен выглядеть вывод, и вы можете указать такие вещи, как: количество отображаемых цифр; формат чисел; имя переменной; и даже добавить вывод до или после переменной, например, строку текста. Команда `printf` и ее вывод будут выглядеть следующим образом (при условии  $x = 40$ ):

```
>> printf('Ответ: %i', x);
>> Ответ 40
```


## Скрипты и функции

Теперь, зная некоторые основы, мы можем перейти к более продвинутым вещам, которые умеет делать *Octave*. В отличие от базового калькулятора, в *Octave* можно писать скрипты для выполнения сложных функций, нечто вроде примитивного программирования, где вы можете запрашивать вводимую информацию, выполнять задачи и выводить результаты в командное окно.

Как и его недешевый кузен *Matlab*, *Octave* может использовать скрипты и функции для создания программы, способной производить вычисления самостоятельно с помощью вводимой пользователем информации или даже с помощью чтения текстовых файлов для выполнения задачи.

Нажав на кнопку **New Script** в левом верхнем углу главного окна, вы откроете свежий файл скрипта, где можно написать программу, запускаемую независимо от пользователя, то есть вам не нужно будет перенабирать каждую команду, когда вы захотите что-то вычислить. Файлы скриптов сохраняются в формате `.m`, что гарантирует дальнейшую совместимость с *Matlab*. Все созданное этим способом в *Octave* можно использовать в *Matlab*, и наоборот.

Скорая помощь



Чтобы получить полный список доступных команд стиля графика, перейдите на <http://bit.ly/OctaveGraphStyles>.

Файл скрипта при первом вашем использовании будет пустым и будет располагаться во вкладке под названием `editor`. Над файлом вы найдете ряд опций для запуска, сохранения и редактирования скрипта. Начав печатать свои команды, вы заметите, что редактор также маркирует для вас многие элементы цветом, как и полагается хорошему текстовому редактору.

Итак, вы ознакомились с раскладкой и основными функциями; что именно вы сможете делать? С помощью этого математически-ориентированного языка вы с легкостью сделаете массу вещей, которые вы могли бы делать в другом языке программирования, таком как Java или C. Можно использовать циклы `for` и циклы с предусловием, чтобы позаботиться о повторяющихся задачах, типа генерирования массивов чисел; присвоения переменных; запроса ввода информации от пользователя; принятия решений с условиями `if/else`, и т.д.

Одной из многих вещей, которые вы можете попробовать с вашей новоприобретенной способностью к скриптам, является создание удобного маленького инструмента для вычисления значения у по прилагаемой формуле и заданному значению  $x$ .

```
X = 3.2;
Y = 3*x+24;
disp(Y);
```

Именно таким образом мы здесь вычислили значение  $Y$  (33.6, если вам это интересно) и добавили команду вывода результата в командное окно. Конечно, это пример незатейливый, но *Octave* предлагает множество инструментов для создания более сложных скриптов и функций, таких как `case`; `do-until`; `break`; `continue`; и все математические операторы для сложения, вычитания, умножения и деления; а также корни и показатели степени. Он включает все обычные логические операторы и операторы сравнения, такие как `<`, `<=`, `>`, `>=`, `==`, `!=`, `&&` и `||` и т.д.

Функции — это файлы, содержащие математические выражения и, как правило, отдельный фрагмент кода, вызываемый главным файлом скрипта. Для тех, кто знаком с программированием, это работает как метод или функция в языке программирования Java. В *Octave* функции обычно используют следующий синтаксис:

```
Function [return value 1, return value 2, ...] = name([arg1,arg1,...])
<тело кода>
endfunction
```

Чтобы вызвать функцию в файле скрипта, мы добавим следующую строку:

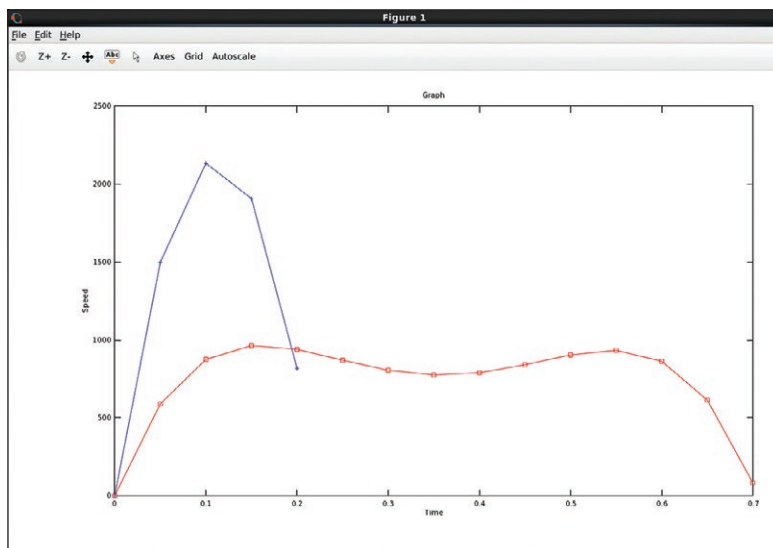
```
value1= name(7);
```

В этой строке 7 будет функционировать в качестве входной переменной, и любое возвращенное значение присваивается переменной `value1`. Функции помогут организовать код таким образом, чтобы его было легко понять, легко изменить и в большинстве случаев сделать короче.

## Сплошные графы

Теперь перейдем к потехе: построению графиков! Мы можем нарисовать множество завлекательных картинок с помощью встроенного *FLTK* (*Fast Light ToolKit*), графического инструментария. *FLTK* заменил *gnuplot* в качестве основной графической библиотеки, поскольку он лучше масштабирует и вращает 3D-графики, а это полезная функция, когда вы пытаетесь визуализировать данные. 2D-графики очень просты и часто требуют лишь набора данных  $x$  и  $y$  в некоторой форме и функции `plot`. Введение массивов (как показано ниже) построит вам простой линейный график с наклоном 0,5:

```
>> X = [0:0.5:200];
>> Y = [0:0.25:100];
>> plot(x,y);
```



➤ По осям X и Y графики могут быть по-разному окрашены и иметь разную длину. И можно указать на графике условные обозначения, командой `legend()`.

➤ Пропустили номер? Узнайте на с. 108, как получить его прямо сейчас.

## Ключ к успеху

Будь то в жизни или в Linux, успех одного построен на силах многих. За рамками этой статьи есть множество источников, которые могут помочь вам в расширении ваших знаний об Octave. Вся документация доступна на <https://www.gnu.org/software/octave/doc/interpreter>. Это руководство, опубликованное создателем ПО, Джоном Итоном [John Eaton], покрывает почти все, что вам нужно знать для задействования всего потенциала ПО. Единственный

недостаток этого руководства в том, что, поскольку оно всеобъемлющее, временами оно весьма плотное. Для более дружелюбного к новичкам набора руководств проверьте руководство Wikibooks на <http://bit.ly/WikibooksOctave>.

Еще одно полезное руководство написано Tutorialspoint на [http://www.tutorialspoint.com/matlab/matlab\\_gnu\\_octave.htm](http://www.tutorialspoint.com/matlab/matlab_gnu_octave.htm). Хотя внешне это руководство предназначено для *Matlab*, почти весь синтаксис

переносится в *Octave*, и руководство отмечает, где они несовместимы. Хотя информация в этих источниках может частично перекрываться, каждый из них предоставляет свои собственные примеры, а наличие множества различных примеров будет действительно полезным для начинающих, желающих охватить различные способы применения конкретных команд или фрагмента кода для достижения требуемого результата.

```
Alfian@localhost:~$ octave --no-gui
warning: docstring file '/usr/local/share/octave/4.0.0/etc/built-in-docstrings'
not found.
GNU Octave, version 4.0.0
Copyright (c) 2015 John W. Eaton and others.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. For details, type 'warranty'.

Octave was configured for 'x86_64-unknown-linux-gnu'.

Additional information about octave is available at http://www.octave.org.
Please contribute if you find this software useful.
For more information, visit http://www.octave.org/get-involved.html
Read http://www.octave.org/bugs.html to learn how to submit bug reports.
For information about changes from previous versions, type 'news'.
>> █
```

► При желании вы можете запустить *Octave* в старомодном режиме одной только командной строки, просто войдя в терминал и напечатав `octave --no-gui`; тогда GUI выключится.

На заметку: взгляните, как полезна функция подавления вывода с помощью точки с запятой. Забыв о ней, вы получите в массиве, появляющемся в вашем командном окне, все 400 или около того условий. Конечно же, есть возможность построить несколько 2D-графиков одновременно, просто разделив параметры запятой, например: `>> plot(x,y,x2,t2);`

Этот кусочек кода (выше) позволит вам создать два графика с различными наборами значений  $x$  и  $y$ . Учтите: так можно создавать графики разного размера. Желая открыть несколько окон для разных графиков, поставьте `;` перед каждой командой `plot()`. Чтобы различать графики, вы можете придавать им разные цвета и стили, добавляя части к команде `plot`. Вы даже можете добавлять команды для вставки названия графика и обозначения осей. Взгляните на пример ниже:

```
>> plot(x,y,'r-s',x2,y2,'b-+'); title('Graph Name'); xlabel('x-axis');
ylabel('y-axis');
```

Эта строка команд, разделенных запятыми, вначале строит график, обозначает первый график в виде красной линии с квадратными точками ввода данных, а второй график делает синим с обозначенными «+» точками ввода данных, и затем добавляет название и обозначения осей. Эти команды можно без изменения использовать в *Matlab*.

## Использование 3D-графиков

3D-графики несколько тяжелее. Простейшая форма линейных 3D-графиков выполняется с помощью функции `plot3` — например, следующий код создаст спиральный 3D-график:

```
>> z = [0:0.05:5];
>> plot3(cos(2*pi*z), sin(2*pi*z), z);
```

Используя скрипты и функции, вы можете сократить этот процесс до простого введения значений для  $x$ ,  $y$  и  $z$ . Ну, а если мы хотим создавать эти необычные 3D-графики плоскости? Команда `meshgrid`, разумеется! Эта команда создает матрицы координат

и  $u$ , чтобы помочь построить данные оси  $z$ . Вы также можете попробовать функцию `ngrid`, которая позволяет строить  $n$ -мерные графики, не ограничиваясь двумя или тремя размерностями.

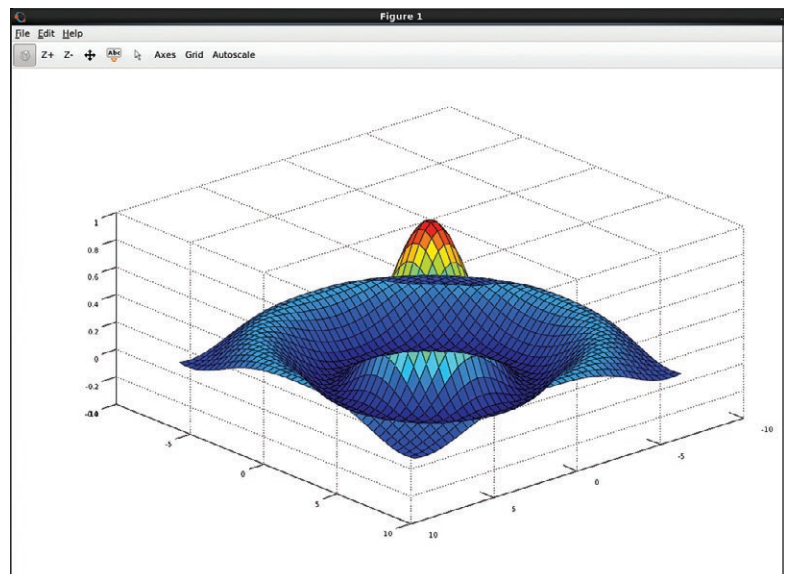
Типовая структура команды для графика `meshgrid` будет выглядеть так:

```
>> [x,y] = meshgrid(-20:0.5:20);
>> z = x.*exp(-x.^2-y.^2);
>> mesh(x,y,z);
```

Это создаст 3D-плоскость с расчетной сеткой, прозрачной между линиями. Команда `meshgrid` в первый раз создает 2D-плоскость. Вторая линия определяет функцию для  $z$ , а третий график, «сетка» 2D, который мы создали ранее по отношению к функции  $z$ . Чтобы заполнить пробелы, вы можете приписать к командам выше такую команду `surf`:

```
>> surf(x,y,z);
```

Итак, мы рассмотрели все основы, которые необходимо знать, чтобы начать работу с *Octave*. Это знание также пригодится в *Matlab* или других похожих программах, в зависимости от того, какое ПО вы используете для выполнения своих задач. Однако у *Octave* намного больше возможностей, чем мы рассмотрели на нашем уроке. В конце концов, изучение, как использовать весь потенциал *Octave*, займет куда больше, чем четырехстраничное руководство. Чтобы узнать больше об *Octave* и о том, как его можно использовать в решении системы линейных уравнений, дифференциальных уравнений и других приложений, вы можете перерыть огромное множество руководств и документации, и лучшей отправной точкой будет <http://bit.ly/OctaveSupport>. **LXF**



► Мощь недавно внедренных графических библиотек *FLTK* означает, что вы теперь можете с легкостью вращать 3D-график и изменять его масштаб, прямо как с известным «Сомбреро *Octave*», показанном здесь.



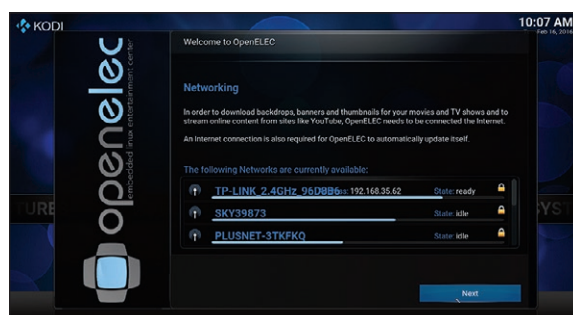
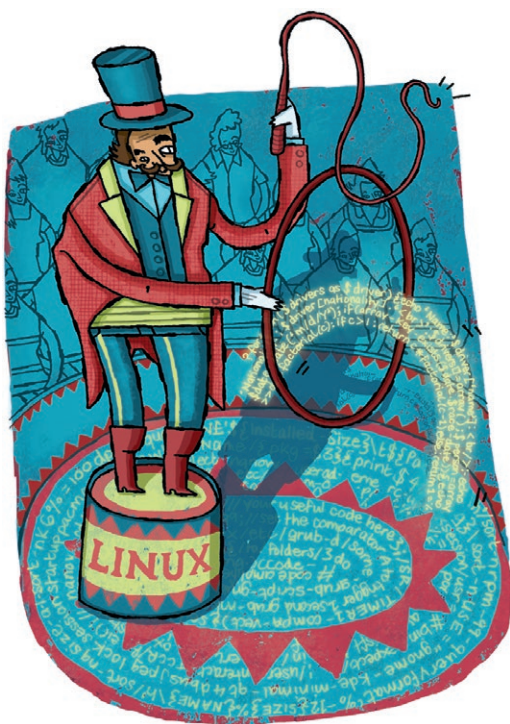
# OpenELEC. СВОЙ МЕДИА-ЦЕНТР

Ник Пирс показывает, как создать себе флешку с умным потоковым вещанием через Raspberry Pi — и для персональных медиа, и для Интернета.



## Наш эксперт

Ник Пирс провел последние полтора года, пытаясь превратить свой Pi в медиа-сервер для быстрого доступа к своей неуклонно растущей коллекции DVD: она изрядно разбухает и выдает явную одержимость певицей Дорис Дей [Doris Day].



» OpenELEC запускает мастера начальной настройки, чтобы помочь вам подключиться — а если у вас Pi Zero и вы хотите организовать сеть, понадобится USB-хаб.

неофициальные сборки для кракнутых устройств Apple TV марки 1 (перейдите на <http://chewitt.openelec.tv/appletv>), а также для оборудования, основанного на AMLogic (<http://bit.ly/amlogic-oe>).

## Выберите себе оборудование

Самый дешевый способ построить устройство потокового вещания OpenELEC с нуля — сделать его на основе Raspberry Pi Zero. Есть только одно небольшое затруднение, связанное с тем, что у него всего один USB-порт, и вам потребуется мощный питаемый хаб, чтобы поддерживать клавиатуру и Wi-Fi адаптер во время фазы изначальной установки. Будьте готовы заплатить £30–40 за весь необходимый вам набор от таких производителей, как PiHut ([www.thepihut.com](http://www.thepihut.com)) или Pimoroni (<https://shop.pimoroni.com>). Вам понадобится Pi Zero (очевидно), системный блок, адаптер питания, адаптер Wi-Fi, карта microSD, USB-хаб с собственным блоком питания и аксессуары.

Если вы готовы потратить несколько больше, то Raspberry Pi Model B+ стоит £19,20, а четырехъядерный Pi 2 Model B — £28 (<http://uk.rsonline.com>), без учета стоимости адаптера питания, Wi-Fi-адаптера, карты microSD и системного блока. Оба идут с портом Ethernet для проводного подключения сети, четырьмя USB-портами и полноразмерным HDMI-портом — выбирайте Pi 2, если планируете запускать медиа-сервер.

Для изначальной настройки OpenELEC потребуется клавиатура, но по завершении этих этапов вы сможете управлять OpenELEC удаленно через ваш веб-браузер или бесплатное мобильное приложение. Вам также понадобится где-то хранить ваши медиа. Если ваша коллекция невелика (менее 50 Гб), то разоритесь на 64-Гб карту microSD и храните ее локально; в ином случае присоедините жесткий диск USB или даже храните свои медиа на диске NAS и подключайтесь через сеть. Обратите внимание, что последний вариант значительно замедлит ход событий, и вы столкнетесь с буферизацией, особенно при подключении через Wi-Fi.

## Скорая помощь

Потратьте время на OSMC (<http://osmc.tv>). Это еще один дистрибутив на базе Kodi, оптимизированный для меньших устройств (включая Pi) и идущий со своим собственным минималистичным скином. Есть небольшая разница в производительности, и хотя OSMC в стандартном варианте проще в использовании, OpenELEC предлагает больше Kodi-подобного опыта.

Зачем раскошелиться на дорогую телевизионную приставку, когда можно создать свою собственную со значительно меньшими затратами? Благодаря мощному ПО медиа-центра с открытым кодом Kodi (<http://kodi.tv>) вы можете получить доступ к локальным персональным медиа, а также к широкому спектру интернет-сервисов потокового вещания, включая телевидение «вслед за эфиром».

Успех Kodi — ранее известного как XBMC — привел к разработке Kodi-содержащих дистрибутивов. Если вы ищете полномасштабный основанный на Ubuntu дистрибутив с Kodi поверх, то Kodibuntu (<http://kodi.wiki/view/Kodibuntu>) вам подойдет.

Для нужд большинства людей Kodibuntu — это перебор, и здесь на сцену выходит OpenELEC ([www.openelec.tv](http://www.openelec.tv)). Это встроенная ОС, построенная вокруг Kodi, оптимизированная для менее мощных установок и задуманная максимально простой в работе и администрировании. К ОС, лежащей в основе, можно получить доступ через SSH, но по большей части вы можете ограничиться исключительно средой Kodi.

На данный момент доступны четыре официальных сборки: «универсальная» покрывает 32- и 64-битные графические устройства Intel, Nvidia и AMD; две разновидности Raspberry Pi — одна для Pi 2, а другая для всего остального, включая Pi Zero; и одна финальная сборка для устройств Freescale iMX6 ARM. Есть также

## SSH-доступ

Переключитесь на SSH, и у вас появится доступ к лежащей в основе установке Linux через Terminal (используйте `ssh root@192.168.x.y`, заменив `192.168.x.y` на IP-адрес устройства OpenELEC. Пароль `openelec`). Основная причина для этого — возможность настройки OpenELEC без лишнего захода в System > OpenELEC. Вначале наберите `ls -all` и нажмите Enter — вы увидите ключевые папки, по умолчанию скрытые.

Поддерживаются базовые команды — такие как `ifconfig` для проверки настроек вашей сети и `top` для просмотра текущего использования памяти и CPU. Делать здесь вы можете не так уж и много — идея в том, чтобы дать вам доступ только к полезным инструментам. Настройки сети в OpenELEC управляются демоном `connman`; например — для

проверки этого перейдите в `storage/cache/connman`, где вы найдете длинное имя файла, начинающееся с `wifi_`. Войдите в эту папку с помощью `cd wifi_` и затем наберите `nano settings`, чтобы получить доступ. Если вы бы хотели установить отсюда статический IP-адрес, измените следующие строчки:

```
IPv4.method=manual
```

Затем добавьте следующие три строчки под `IPv6.privacy=disabled`:

```
IPv4.netmask_prefixlen=24
```

```
IPv4.local_address=192.168.x.y
```

```
IPv4.gateway=192.168.x.z
```

Замените `192.168.x.y` на выбранный вами IP-адрес, а `192.168.x.z` — на IP-адрес вашего роутера (вы получите его с помощью `ifconfig`). Сохраните ваши изменения и перезагрузитесь.

```
nick@nickubuntu:~$ ssh root@192.168.35.45
root@192.168.35.45's password:
#
# http://openelec.tv
#
OpenELEC (official) version: 6.0.1
openelec:~# ls -all
total 24
drwxr-xr-x 13 root root      1024 Jan 16 15:26
drwxr-xr-x 16 root root      256 Jan 27 08:26
-rw-r--r-- 1 root root       774 Feb 16 15:26 ash_history
drwxr-xr-x  8 root root      4096 Jan 1 1970 cache
drwxr-xr-x 11 root root      1024 Jan 1 1970 config
drwxr-xr-x  8 root root      1024 Jan 1 1970 kodi
drwxr-xr-x  2 root root      1024 Jan 1 1970 ssh
drwxr-xr-x  2 root root      1024 Jan 1 1970 update
drwxr-xr-x  2 root root     12288 Jan 27 08:26 lost+found
drwxr-xr-x  2 root root      1024 Jan 1 1970 music
drwxr-xr-x  2 root root      1024 Jan 1 1970 pictures
drwxr-xr-x  2 root root      1024 Jan 1 1970 screenshots
drwxr-xr-x  2 root root      1024 Jan 1 1970 sshkeys
drwxr-xr-x  2 root root      1024 Jan 1 1970 videos
openelec:~# cd /cache/connman/wifi
openelec:~/cache/connman/wifi/connman09338f4_46502d4c4940b5f32263447487a5f93644384236_managed_psk # nano set
ttings.xml
```

➤ Решив настраивать OpenELEC через Terminal, вы обнаружите ограниченное количество доступных команд.

Мы поместили обе сборки Raspberry Pi и универсальную 64-битную сборку OpenELEC 6.0.1 на **LXF DVD**, чтобы помочь вам приступить к работе — или же вы можете скачать последнюю версию с <http://openelec.tv/get-openelec>. Файлы сжаты в формат TAR или GZ, так что вначале вам нужно извлечь их. Простейший способ сделать это — использовать GUI вашего дистрибутива Linux; в Ubuntu, например, скопируйте файл на свой жесткий диск, затем щелкните по нему правой кнопкой мыши и выберите пункт Extract Here.

## Сборка, установка и настройка

Теперь подключите свою карту microSD к ПК через подходящий карт-ридер (вы можете приобрести его онлайн за £3) и используйте команду `$ dmesg | tail` или утилиту `Disks` для определения ее точки монтирования. После этого введите следующие команды — при условии, что ваш диск — `sdс` и ваш файл образа находится в папке **Downloads**:

```
$ umount /dev/sdc1
```

```
$ cd Downloads
```

```
$ sudo dd if=OpenELEC-RPi.arm-6.0.1.img of=/dev/sdc bs=4M
```

Если вы устанавливаете OpenELEC на Raspberry Pi 2, понадобится скомандовать `sudo dd if=OpenELEC-RPi2.arm-6.0.1.img of=/dev/sdc bs=4M`. Подождите, пока образ запишется на вашу карту microSD — это может занять некоторое время, к тому же здесь нет прогресс-индикатора, так что запаситесь терпением (возможно, пришла пора для чашечки чая?).

По завершении размонтируйте диск и извлеките его. Вставьте карту microSD в Pi, подключите его к монитору и клавиатуре и включите. Вы должны немедленно увидеть зеленый огонек и включившийся экран.

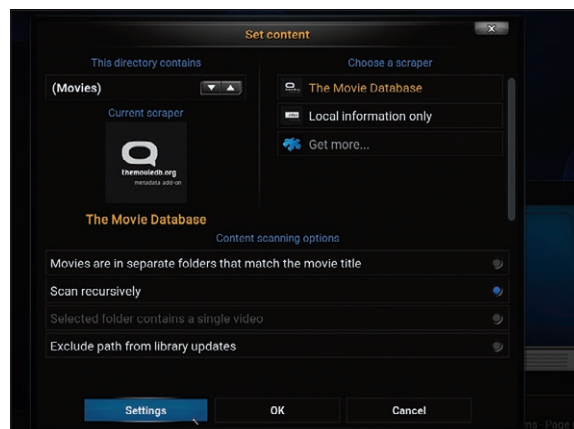
Появится заставка, и в этот момент OpenELEC скажет вам, что изменяет размер карты — по существу, это создание раздела данных, где вы можете локально хранить медиа, если захотите. После второй перезагрузки вы наконец-то окажетесь в мастере изначальной настройки самого Kodi.

Если у вас не подключена мышь, используйте Tab или клавиши со стрелками для навигации между опциями и Enter для их выбора. Начните с просмотра имени хоста — OpenELEC — и его изменения, если вы собираетесь запускать медиа-сервер, а имя все еще недостаточно очевидно. Затем подключитесь к вашей сети Wi-Fi, выбрав ее из списка и введя пароль. Затем вы можете добавить поддержку удаленного SSH доступа, а также *Samba* (см. врезку *SSH-доступ* внизу).

Теперь вы можете удаленно управлять Kodi через свой веб-браузер, если захотите: напишите `192.168.x.y:80` в вашем браузере (заменяв `192.168.x.y` на IP-адрес своего Pi). Переключитесь на вкладку Remote, и вы найдете удобный пульт управления с системой указания и щелчка на экране — и, что не совсем очевидно, ваша клавиатура теперь тоже управляет Kodi, поскольку была подключена к вашему Pi напрямую. Вы также увидите вкладки для фильмов, ТВ-шоу и музыки — заполнив свои медиа-библиотеки, вы сможете просматривать и настраивать содержимое, чтобы воспроизводить его отсюда.

Данный подход основывается на пребывании вашего ПК или ноутбука в прямой видимости от вашего ТВ — если это не практично, используйте вместо этого свой планшет или телефон в качестве пульта дистанционного управления. Поищите Kore в Google Play (Android) или Kodi Remote в App Store (iOS), и вы обнаружите, что оба приложения с легкостью найдут ваш Pi и позволят управлять им через интерфейс, подобный дистанционному пульта управления.

По умолчанию OpenELEC использует DHCP для подключения к вашей локальной сети — если локальный IP-адрес вашего Pi меняется, может быть трудно отследить его в вашем браузере для дистанционной настройки. Измените это, выбрав System > OpenELEC > Connections, выбрав ваше подключение и нажав Enter. Выберите Edit из списка и отметьте IPv4 для присвоения статического IP-адреса, который вы сможете использовать, чтобы всегда иметь возможность получить доступ к Kodi в будущем. Вы легко можете оставить текущий присвоенный адрес или выбрать другой.



## Скорая помощь

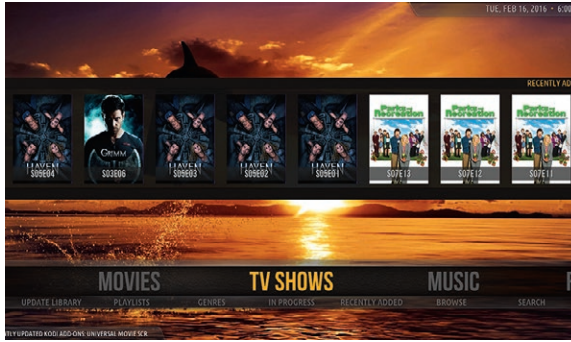
Чтобы управлять Kodi через удаленный ПК или мобильник, по умолчанию требуется только имя пользователя. Но не мешает добавить также пароль — для этого перейдите в System > Settings > Services > Web Server, где можно заодно и сменить имя пользователя.

»

➤ Для автоматического захвата иллюстраций и метаданных из ваших медиа-файлов, основанных на именах файлов и структуре папки, Kodi применяет «скребки».



➤ Скин Amber — красивая альтернатива более функциональному Confluence по умолчанию. К сожалению, из него нет доступа к меню настройки OpenELEC.



Не забудьте нажать Save, чтобы изменения вошли в силу. Если все это выглядит большой морокой, прочитайте врезку об SSH (SSH-доступ, стр. 69), чтобы вместо этого получить способ изменить лежащие в основе файлы конфигурации.

## Настройка библиотек

Первым делом надо добавить ваши медиа в вашу же библиотеку. Kodi поддерживает широкий спектр хранилищ и форматов, так что проблем у вас не должно быть, если только вы не работаете с каким-то малоизвестным форматом. Ознакомьтесь со врезкой (см. *Добавление содержимого в библиотеку* внизу) для совета по именованию и организации ваших медиа, чтобы позволить Kodi признать их и отобразить дополнительную информацию о ТВ-шоу и фильмах. Он использует помощь специальных «скребков [scrapers]» — инструментов, которые извлекают метаданные из онлайн-баз данных, такие как названия, краткие содержания ТВ-эпизодов и графическое оформление, чтобы совместить их с вашими медиа-файлами для идентификации.

Где стоит хранить свой локальный контент для Kodi, чтобы иметь к нему доступ? Если ваша карта microSD достаточно вместительна — мы бы предложили 64 ГБ или больше — на ней можно хранить изрядное количество видео и музыки. Вы можете передавать файлы по локальной сети — откройте File Manager и выберите для просмотра свою сеть. Должно показаться ваше устройство OpenELEC; дважды щелкните по записи общего доступа к файлам, и вы увидите папки для Музыка, Изображений, ТВ-шоу и Видео. Просто скопируйте свои файлы сюда, чтобы добавить их в библиотеку. После этого просмотрите Видео или Музыка, и медиа-файлы уже должны отображаться и учитываться, хотя пока им еще не был присвоен скребок, помогающий вам определить их.

Копирование файлов через сеть бывает медленным — вы можете передавать файлы прямо на карту, когда она монтирована

в карт-ридер вашего ПК, но для этого надо иметь доступ к File Manager с правами root; в Ubuntu, к примеру, вы получите требуемый доступ, набрав \$ `gksudo` и нажав Enter. Вариант попроще — если на вашем Pi есть запасной USB-порт, храните свои медиа на внешней флешке или жестком диске. Просто вставьте в Pi накопитель, перейдите в Видео или Музыка и выберите опцию Add... Нажмите Browse и выберите верхнюю папку, содержащую тот тип медиа, который вы добавляете — ТВ-шоу, фильмы или музыку. Если вы подключили USB-устройство, вы найдете его под `root/media`, тогда как накопители NAS обычно находятся под Windows Network (SMB). Выбрав, нажмите OK. Появится диалоговое окно Set Content — используйте клавиши со стрелками вверх и вниз, чтобы выбрать тип медиа, который вы вносите в каталог, и подтвердите, что хотите использовать именно выбранный скребок. Проверьте опции проверки содержимого — по умолчанию они должны подходить большинству людей — и нажмите Settings, чтобы просмотреть продвинутые настройки (например, для фильмов вы можете захотеть изменить страну сертификации на Великобританию). Нажмите OK дважды и выберите Yes при предложении обновить библиотеку.

Когда все будет сделано, вы найдете новый элемент — Library — добавленным в меню медиа на основном экране. Он дает вам доступ к вашему контенту с такими фильтрами, как жанр, название или год, для помощи в навигации в больших коллекциях. Теперь повторите все это для других имеющихся у вас типов медиа. Если вы хотите включить множественное расположение папок в пределах одной библиотеки, следует перейти к виду Files и щелкнуть правой кнопкой мыши по имени библиотеки (или выбрать ее и нажать с на клавиатуре), чтобы вызвать контекстное меню. Выберите Edit Source, чтобы добавить местоположений, а если надо — Change Content, чтобы изменить тип медиа и скребок.

Умнейшая вещь, которую можно сделать с любой цифровой медиа-библиотекой — хранить ее на медиа-сервере, что позволит вам с легкостью получить к ней доступ с других устройств вашей сети и — в некоторых случаях — через Интернет. Kodi обладает умениями медиа-сервера UPnP, и это блестяще сработает с другими экземплярами Kodi в сети, а также даст доступ к вашим медиа с других совместимых клиентов. Медиа-серверы бывают весьма требовательны, так что не рекомендуем использовать Pi Zero или Pi Model B+. Вместо этого настройте его на вашем самом мощном ПК (или Pi 2) и используйте OpenELEC, чтобы подключиться к нему как клиент.

Как медиа-сервер, Kodi довольно примитивен. Если вам нужен привлекательный и гибкий сервер, см. наше руководство по Emby [стр. 28 LXF204]. Совместите это с примочкой [addon] Emby for Kodi, и вы сможете получить доступ к вашим хранимым Emby

### Скорая помощь

Хотите обновить OpenELEC до свежей сборки? Вначале скачайте файл последнего обновления (в формате TAR) с <http://openelec.tv/get-openelec>, откройте File Manager и щелкните по Browse Network. Дважды щелкните по своему устройству OpenELEC и скопируйте TAR-файл в папку Update. Перезагрузите OpenELEC, и вы обнаружите, что обновление будет применено.

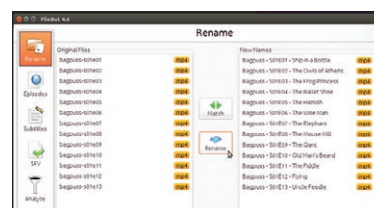
## Добавление содержимого в библиотеку

Kodi лучше работает с локально хранимыми цифровыми медиа, но чтобы распознавать в вашей музыкальной коллекции ТВ-шоу, следует правильно наименовать свои медиа, а также распределить их по правильно названным папкам.

Kodi поддерживает то же соглашение об именовании, что и его конкуренты Emby

[см. стр. 30 LXF204] и Plex — используйте приведенную внизу таблицу для помощи в этом деле:

Нужно быстро переименовать файлы? Тогда Filebot ([www.filebot.net](http://www.filebot.net)) — ваш новый лучший друг [более подробные указания по его использованию — см. стр. 30 LXF204].



➤ Дайте своим медиа-файлам правильные имена, если хотите, чтобы они появились в медиа-библиотеке в полноценном виде.

Тип	Структура папки	Синтаксис	Пример
Музыка	Музыка/исполнитель/альбом	Исполнитель — название трека	Music\David Bowie\Blackstar\David Bowie - Lazarus.mp3
Фильмы	Фильмы/Жанр/Название фильма	Название (год)	Movies\Sci-Fi\Star Trek\Star Trek (2009).mkv
ТВ-шоу	ТВ/Жанр/Название шоу/Сезон	tvshow — s01e01	TV\Sci-Fi\Fringe\Season 5\Fringe - s05e09.mkv
Клипы	Клип/Исполнитель	исполнитель — название трека	Music Videos\A-ha\A-ha - Velvet.mkv

медиа, не добавляя их в свой Kodi. Похожий аддон, PlexBMC (<http://bit.ly/PlexBMC>), предлагающий привлекательный интерфейс, существует также для пользователей Plex Media Server.

Желая получить доступ к другим серверам UPnP через Kodi без особых прибамбасов, перейдите в System > Settings > Services > UPnP/DLNA и выберите Allow remote control via UPnP. Здесь можно также настроить Kodi в качестве медиа-сервера: выберите Share my libraries, они должны быть видимы любому клиенту UPnP в вашей сети, хотя тут не исключена перезагрузка.

Очевидно, что на маломощных устройствах типа Pi производительность не обойдется без проблем; Pi 2 очень отзывчив, а вот Pi Zero временами буксует. Следовательно, имеет смысл попытаться оптимизировать ваши настройки, чтобы передать Pi столько ресурсов, сколько ему нужно для бесперебойной работы.

Начните с отключения ненужных сервисов — загляните под System > OpenELEC > Services (скажем, Samba вам ни к чему, если вы не делитесь файлами с и из Kodi) и под System > Settings > Services (обычно AirPlay не требуется). Заодно, пока вы в System > Settings, нажмите на Settings level: Standard, чтобы сперва выбрать пункт Advanced > Expert для открытия большего числа настроек.

Слабым местом устройств Pi является взаимодействие с большими библиотеками — протяните Pi руку помощи: перейдите в Settings > Music > File lists и отключите чтение тэгов. Также перейдите в Settings > Video > Library и отключите Download actor thumbnails [Отключить миниатюры исполнителя]. Еще можно отключить Extract thumbnails and video information под File Lists, но тогда вы потеряете массу декоративных элементов и кэширование эскизов для дальнейшего употребления.

Скин по умолчанию Confluence весьма резвый, но если просмотр домашнего экрана страдает спотыканием, обдумайте, не отключить ли показ недавно добавленных видео и альбомов: выберите Settings > Appearance, затем нажмите на Settings на правой панели под Skin. Переключитесь на Home Window Options и снимите выделение с обеих опций Show recently added....

Кстати о Confluence: если вам не нравится скин по умолчанию, попробуйте Amber — он приятен глазу и не требователен к системным ресурсам. Запустив его, вы потеряете доступ к OpenELEC, но всегда можно временно переключиться обратно на Confluence или использовать SSH для изменения настроек при необходимости. LXF

## Добавьте ТВ «вслед за эфиром» на свою «потокую» флешку



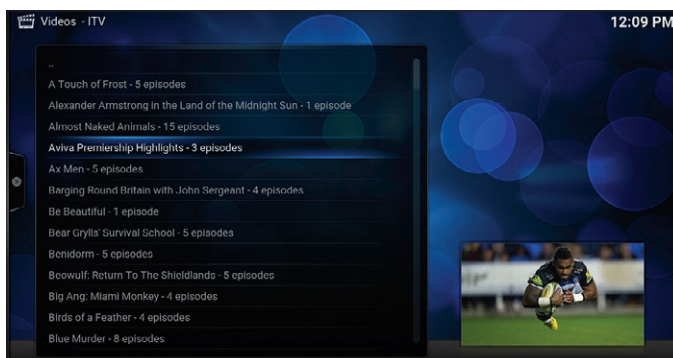
### 1 Добавьте BBC iPlayer

Перейдите в Videos > Add-ons. Выберите Get more..., затем прокрутите список и найдите iPlayer WWW. Выберите его и нажмите Install. После установки вы сможете получить доступ к нему через Videos > Add-ons для доступа к потоковому вещанию в режиме live или «вслед за эфиром». Настройте субтитры и другие предпочтения через System > Settings > Add-ons > iPlayer WWW.



### 2 Получите ITV Player

Перейдите в System > File Manager. Выберите Add Source с последующим <None>, введите <http://www.xunitytalk.me/xfinity> и выберите Done, а затем OK. Нажмите Esc, затем выберите System > Settings > Add-ons > Install from ZIP file. Выберите в списке местоположений xfinity, выберите XunityTalk\_Repository.zip, нажмите Enter и подождите, пока он установится.



### 3 Завершите установку

Теперь выберите Install from repository, а затем XunityTalk Repository > Video add-ons, прокрутите вниз и выберите ITV. Выберите Install, и он должен быстро скачаться, установиться и включиться. Перейдите в Videos > Add-ons для доступа к потоковому вещанию в режиме live шести каналов ITV channels, а также доступ к программам через ITV Player в течение последующих 30 дней — значительное преимущество по сравнению с семью днями, предлагаемыми большинством платформ.



### 4 UKTV Play

Следуйте инструкциям ITV Player по добавлению <http://srp.nu> в качестве источника, затем добавьте основной репозиторий через пункты меню SuperRepo > isengard > repositories > superrepo. После этого выберите Install from repository > SuperRepo > Add-on repository > SuperRepo Category Video. И наконец, добавьте UKTV Play из репозитория SuperRepo Category Video для получения доступа к открытым каналам UKTV Play.



ЧАСТЬ 3

# MySQL Fabric: Еще о шардинге

Лада Шерышова принимается за разнообразные операции с шардами и шардированными данными.



Наш эксперт

Лада Шерышова долгие годы работала на коммерческие корпорации, создавая промышленные высоконадежные информационные системы. Но пришло время сбросить оковы и применить свои знания и опыт в работе со свободным ПО.

В предыдущей части учебника мы познакомились с основами шардирования данных на базе *MySQL Fabric* и научились создавать карту шардирования, регистрировать в ней таблицы и добавлять шарды. В этой части мы научимся выполнять операции над шардами (разделять и перемещать их), читать данные из шардированной схемы, а также рассмотрим более подробно возможности коннектора Python для написания приложений, работающих с шардированными данными.

## Разделяем шарды

Если шард становится перегруженным, можно разделить его на части, поместив их в другие высокодоступные (ВД) группы. Для этого в соответствии с тестовой конфигурацией (см. Учебники, стр. 72 LXF208) создадим еще одну группу group-3, добавив туда 2 сервера — localhost:3319 и localhost:3320:

```
>> mysqlfabric group create group-3 --description='MySQL Fabric Group-3'
>> mysqlfabric group add group-3 localhost:3319
>> mysqlfabric group add group-3 localhost:3320
```

Назначим группе главный сервер (например, localhost:3320) и подключим регистратор отказов для включения автоматического режима обнаружения сбоев:

```
>> mysqlfabric group promote group-3 --slave_id=localhost:3320
>> mysqlfabric group activate group-3
```

Разделим шард 1, расположенный в группе group-1, на два по значению ключа шардирования emp\_no = 500. Это значит, что в первом шарде останутся данные, соответствующие значениям ключа от 1 до 499, а во втором шарде, который мы разместим в группе group-3, будут храниться данные, соответствующие номерам сотрудников от 500 до 999. Для разделения шарда предназначена следующая команда:

```
Linux-63@p:/usr/local/mysql # mysqlfabric sharding split_shard 1 group-3 --split_value=500
Fabric UUID: 5ca1abe-a007-feed-f00d-cab3fe13249e
Time-To-Live: 1

uid finished success result
1e20a7b0-cf6c-4545-99d5-734c4e1bb125 1 1 1

state success when description
3 2 1460375815,48 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375815,95 Executed action (_check_shard_info)
5 2 1460375815,88 Executed action (_check_shard_info)
3 2 1460375815,78 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375815,86 Executing action (_backup_source)
5 2 1460375815,47 Executed action (_backup_source)
3 2 1460375815,38 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375815,47 Executing action (_restore_shard)
5 2 1460375845,47 Executed action (_restore_shard)
3 2 1460375845,36 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375845,47 Executing action (_setup_repl)
5 2 1460375846,62 Executed action (_setup_repl)
3 2 1460375846,48 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375846,62 Executing action (_set)
5 2 1460375849,38 Executed action (_set)
3 2 1460375849,29 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375849,38 Executing action (_setup_resharding)
5 2 1460375853,0 Executed action (_setup_resharding)
3 2 1460375852,65 Triggered by <mysql.fabric.events.Event object at 0x
4 2 1460375853,0 Executing action (_prune_shard_tables_afte)
5 2 1460375857,49 Executed action (_prune_shard_tables_afte)
```

Рис. 1. Разделяем шард.

```
Linux-63@p:/usr/local/mysql # bin/mysql --port=3316 --socket=/tmp/mysql6.sock
-u group -p -e "select * from fabrictest.employees;"
Enter password:

+----+-----+-----+-----+-----+
| emp_no | first_name | last_name | birth_date | hire_date |
+----+-----+-----+-----+-----+
| 10 | Pavel | Ivanov | 1972-10-01 | 2005-02-01 |
| 30 | Elena | Orlova | 1978-03-22 | 2008-04-15 |
| 120 | Sergey | Stepanov | 1982-08-09 | 2010-10-16 |
+----+-----+-----+-----+-----+

Linux-63@p:/usr/local/mysql # bin/mysql --port=3320 --socket=/tmp/mysql10.sock
-u group -p -e "select * from fabrictest.employees;"
Enter password:

+----+-----+-----+-----+-----+
| emp_no | first_name | last_name | birth_date | hire_date |
+----+-----+-----+-----+-----+
| 550 | Olga | Petrova | 1980-01-11 | 2011-11-14 |
| 780 | Alex | Plotnikov | 1979-05-18 | 2011-11-08 |
+----+-----+-----+-----+-----+
```

Рис. 2. Результат разделения шардов.

```
>> mysqlfabric sharding split_shard <SHARD_ID> <GROUP_ID>
[--split_value=SPLIT_VALUE]
```

Здесь <SHARD\_ID> — идентификатор шарда, который требуется разделить; <GROUP\_ID> — наименование ВД-группы, в которую должны попасть разделенные данные; --split\_value=SPLIT\_VALUE — значение, задающее диапазон для разделения данных. Параметр не указывается, если схема шардирования использует функцию HASH, а если функцию RANGE — обязан быть определен.

Перед выполнением операции разделения шарда необходимо соединиться с главным сервером группы, к которой принадлежит разделяемый шард, и выполнить команду reset\_master:

```
>> bin/mysql --port=3316 --socket=/tmp/mysql6.sock -u group -p
-e "reset master;"
```

Вспомним, что текущий главный сервер в ВД-группе может быть определен с помощью команды

```
>> mysqlfabric group lookup_servers <GROUP_ID>
```

После этого выполним команду по разделению шарда (рис. 1):

```
>> mysqlfabric sharding split_shard 1 group-3 --split_value=500
```

Убедимся, что разделение шардов произошло. Выполним запрос к таблице employees, соединившись с одним из серверов в группе group-1 (например, localhost:3316) и группе group-2 (localhost:3320):

```
>> bin/mysql --port=<PORT_NUM> --socket=<PATH> -u group -p
-e "select * from fabrictest.employees"
```

Как видим (рис. 2), в группу group-3 перенеслись данные, имеющие значение ключа emp\_no от 500 до 999.

Теперь посмотрим, как изменилась схема шардирования. Соединимся с узлом *MySQL Fabric* (localhost:3311) и выполним следующий запрос:

```
mysql> select shard_ranges.shard_mapping_id, shards.shard_id, group_id, lower_bound, state from shards join (shard_ranges) using (shard_id);
```

Видим (рис. 3), что схема шардирования теперь содержит 3 шарда, которые хранят данные в соответствии с диапазоном значений, имеющих нижнюю границу, указанную в поле lower\_bound.

```
mysql> select shard_ranges.shard_mapping_id, shards.shard_id, group_id, lower_bound,
state from shards join (shard_ranges) using (shard_id);
+-----+-----+-----+-----+-----+
| shard_mapping_id | shard_id | group_id | lower_bound | state |
+-----+-----+-----+-----+-----+
| 1 | 1 | 2 | group-2 | 1000 | ENABLED |
| 1 | 1 | 3 | group-1 | 1 | ENABLED |
| 1 | 1 | 4 | group-3 | 500 | ENABLED |
+-----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

mysql>
```

Рис. 3. Схема шардирования после разделения шардов.

## Перемещаем шарды

Если текущий набор серверов, входящих в шард, не справляется с нагрузкой, можно переместить этот шард в другую, более мощную группу серверов.

Перемещение шарда выполняется командой

```
>> mysqlfabric sharding move_shard <SHARD_ID> <GROUP_ID>
где <SHARD_ID> — идентификатор шарда, которые требуется переместить; <GROUP_ID> — наименование группы, в которую будет перемещаться данный шард.
```

Переместим шард 2 (shard\_id=2) из группы group-2 в новую группу group-4. Для этого создадим группу group-4 и добавим в нее два сервера в соответствии с тестовой конфигурацией localhost:3321 и localhost:3322 (назначим его главным).

Перед тем, как переместить шард в новую группу, необходимо соединиться с ее главным сервером и выполнить команду reset\_master:

```
>> bin/mysql --port=3322 --socket=/tmp/mysql12.sock -u group
-p -e "reset master;"
```

После этого мы можем выполнить команду по перемещению шарда (рис. 4):

```
>> mysqlfabric sharding move_shard 2 group-4
```

Убедимся, что перемещение действительно произошло. Выполним запрос к таблице employees, соединившись с сервером, входящим в группу group-2 (например, localhost:3317), из которой мы делали перемещение, и увидим сообщение об ошибке, сигнализирующее о том, что данная таблица не существует (рис. 5). Это говорит о том, что схема данных и сами данные были удалены из этого шарда.

Теперь проверим, что данные с шарда 2 были перемещены в новую группу. Выполним команду

```
>> mysqlfabric sharding lookup_servers <TABLE_NAME>
<KEY_VALUE>
```

которая показывает шард, соответствующий заданному ключу:

```
>> mysqlfabric sharding lookup_servers fabrictest.employees 1000
```

В качестве значения ключа укажем нижнюю границу диапазона, по которому данные помещались в шард 2.

Рис. 6 демонстрирует — теперь эти данные хранятся на серверах, которые входят в группу group-4, что нам и требовалось доказать.

```
Linux-63@p:/usr/local/mysql # mysqlfabric sharding move_shard 2 group-4
Fabric UUID: 5ca1abe-a007-feed-f00d-cab3fe13249e
Time-To-Live: 1

          uuid finished success result
-----
b493749c-d0e-4508-a82d-c0dba7167434      1      1      1

state success      when      descriptio
-----
3      2      1460386064.1 Triggered by <mysql.fabric.events.Event object at 0x9bf420c>
4      2      1460386064.19      Executing action (<check_shard_information>)
5      2      1460386064.59      Executed action (<check_shard_information>)
3      2      1460386064.4 Triggered by <mysql.fabric.events.Event object at 0x9bf420c>
4      2      1460386064.59      Executing action (<backup_source_shard>)
5      2      1460386065.3      Executed action (<backup_source_shard>)
3      2      1460386065.15 Triggered by <mysql.fabric.events.Event object at 0x9bf430c>
4      2      1460386065.3      Executing action (<restore_shard_backup>)
5      2      1460386065.99      Executed action (<restore_shard_backup>)
3      2      1460386065.99 Triggered by <mysql.fabric.events.Event object at 0x9bf432c>
4      2      1460386065.99      Executing action (<setup_replication>)
5      2      1460386067.03      Executed action (<setup_replication>)
3      2      1460386066.94 Triggered by <mysql.fabric.events.Event object at 0x9bf434c>
4      2      1460386067.03      Executing action (<setup_sync>)
5      2      1460386068.03      Executed action (<setup_sync>)
3      2      1460386068.06 Triggered by <mysql.fabric.events.Event object at 0x9bf435c>
4      2      1460386068.63      Executing action (<setup_resharding_switch>)
5      2      1460386068.63      Executed action (<setup_resharding_switch>)
```

Рис. 4. Перемещаем шард.

```
Linux-63@p:/usr/local/mysql # bin/mysql --port=3317 --socket=/tmp/mysql7.sock -u group
-p -e "select * from fabrictest.employees;"
Enter password:
ERROR 1146 (42502) at line 1: Table 'fabrictest.employees' doesn't exist
Linux-63@p:/usr/local/mysql #
```

Рис. 5. Результат перемещения шарда.

```
Linux-63@p:/usr/local/mysql # mysqlfabric sharding lookup_servers fabrictest.employees 1000
Fabric UUID: 5ca1abe-a007-feed-f00d-cab3fe13249e
Time-To-Live: 1

server_uuid      address      status      mode weight
-----
ddb07ba0-fff0-11e5-9916-0021634dd154 localhost:3322 PRIMARY READ_WRITE 1,0
ddb0e5aa-fff0-11e5-9683-0021634dd154 localhost:3321 SECONDARY READ_ONLY 1,0
```

Рис. 6. Перемещение данных в новый шард.

А как стала выглядеть схема шардирования после перемещения шарда, можно увидеть, подключившись к узлу MySQL Fabric и выполнив запрос к таблице shards (рис. 7).

## Операции чтения

MySQL Fabric позволяет выполнять над шардами многотабличные запросы. Выберем данные о сотрудниках и их зарплатах, соединив таблицы employees и salaries по внешнему ключу emp\_no. В свойствах подключения укажем соответствующие таблицы, ключ шардирования и режим доступа fabric.MODE\_READONLY для включения режима балансировки нагрузки операций чтения между подчиненными серверами в ВД-группах.

Создадим файл read\_data\_gl.py и последовательно добавим туда следующий код.

Создаем подключение к узлу MySQL Fabric:

```
import mysql.connector from mysql.connector
import fabric from mysql.connector
conn = mysql.connector.connect(
    fabric={"host": "localhost", "port": 32274, "username":
"admin", "password": "admin", "report_errors": True},
    user="group", password="group", autocommit=True
)
```

Определяем процедуру чтения данных:

```
def find_employee(conn, emp_no)
```

Установим свойства подключения. В свойствах подключения необходимо указать: tables — шардированная таблица или список таблиц (через запятую), участвующих в операциях чтения данных (в нашем случае, 2 таблицы: employees и salaries); key — ключ шардирования; mode — режим обращения к серверу. Т.к. мы выполняем операцию чтения данных, то параметр mode устанавливаем в значение fabric.MODE\_READONLY. Это говорит Fabric-коннектору о том, что он может выполнять балансировку нагрузки запросов между подчиненными серверами в ВД-группах, не посылая их на главные сервера.

```
conn.set_property(tables=["fabrictest.employees", "fabrictest.
salaries"], key=emp_no, mode=fabric.MODE_READONLY)
```

Определяем запрос к шардированным таблицам employees и salaries, используя оператор JOIN (объединение двух таблиц по ключу):

```
cur = conn.cursor()
cur.execute("USE fabrictest")
cur.execute(
    "SELECT emp_no, first_name, last_name, birth_date, salary
FROM salaries "
    "JOIN employees USING (emp_no) "
    "WHERE emp_no = %s", (emp_no, )
)
```

»

```
Linux-63@p:/usr/local/mysql # bin/mysql --port=3311 --socket=/tmp/mysql11.sock
-u fabric -p -e "select * from fabric.shards;"
Enter password:
+-----+-----+-----+
| shard_id | group_id | state |
+-----+-----+-----+
| 1 | 2 | group-4 | ENABLED |
| 1 | 3 | group-1 | ENABLED |
| 1 | 4 | group-3 | ENABLED |
+-----+-----+-----+
```

Рис. 7. Схема шардирования после перемещения шарда.



```
linux-63np:/usr/local/mysql # python mysqlfab/read_table_gl_ha.py
Retrieved (10, u'Pavel', u'Ivanov', datetime.date(1972, 10, 1), 12000)
Retrieved (30, u'Elena', u'Orlova', datetime.date(1978, 3, 22), 24000)
Retrieved (550, u'Olga', u'Petrova', datetime.date(1980, 1, 11), 35000)
Retrieved (2340, u'Veera', u'Andreeva', datetime.date(1980, 1, 29), 11000)
```

► Рис. 8. Чтение данных из шардированных таблиц.

```
)
for row in cur:
    print "Retrieved ", row
    Выполняем операции чтения, выбирая данные по ключам,
    принадлежащим разным диапазонам (и, соответственно, разным шардам):
    find_employee(conn, 10)
    find_employee(conn, 30)
    find_employee(conn, 550)
    find_employee(conn, 2340)
```

Запустим скрипт: `>> python read_table_gl_ha.py` и проверим результат (рис. 8).

## Полезные команды работы с шардами

- `disable_shard` Делает шард недоступным:  
`>> mysqlfabric sharding disable_shard <SHARD_ID>`  
 где `<SHARD_ID>` — номер шарда, который должен быть отключен.
- `enable_shard` Активирует шард в схеме шардирования:  
`>> mysqlfabric sharding enable_shard <SHARD_ID>`  
 где `<SHARD_ID>` — идентификатор шарда, который должен быть активирован.
- `remove_definition` Удаляет определение карты шардирования:  
`>> mysqlfabric sharding remove_definition <SHARD_MAPPING_ID>`,  
 где `<SHARD_MAPPING_ID>` — идентификатор карты шардирования.
- `remove_shard` Удаляет шард из схемы шардирования:  
`>> mysqlfabric sharding remove_shard <SHARD_ID>`  
 где `<SHARD_ID>` — идентификатор шарда, который должен быть удален.
- `remove_table` Удаляет спецификацию заданной таблицы из карты шардирования:  
`>> mysqlfabric sharding remove_table <TABLE_NAME>`  
 где `<TABLE_NAME>` — наименование шардированной таблицы, спецификация которой должна быть удалена.

## Connector/Python в приложениях

Connector/Python реализует следующие возможности *MySQL Fabric*: автоматический выбор сервера на основании предоставляемой информации о шардинге (таблицы и ключи) и разделение операций записи и чтения внутри ВД-группы.

Для работы с шардингом данных Connector/Python обладает следующими ключевыми возможностями:

- Запрашивает подключение к серверу, управляемому *MySQL Fabric*.
- Получает соединение с *MySQL*-сервером в заданной ВД-группе с установлением режима доступа к серверу `read/write` (чтение/запись) или `read-only` (только чтение).
- Находит соответствующий сервер для заданной шардированной таблицы (или таблиц) и ключа в зависимости от типа операций (`local` или `global`) и режима доступа к серверу (`read/write` или `read-only`) на основании заданной стратегии шардирования (`RANGE` или `HASH`).
- Поддерживает балансировку нагрузки запросов между подчиненными серверами в ВД-группе на основе заданных *MySQL*-серверам весов.
- Для ускорения операций чтения/записи кэширует информацию о ВД-группах и шардинге, полученную от *Fabric*. Всякий раз при поиске данных коннектор Python сначала просматривает свой кэш [`cache`] и берет актуальную информацию из него. Когда

информация в кэше устареваает, он становится недействительным, и должен быть обновлен. По умолчанию, время жизни кэша (`time-to-live` или `TTL`) равно 1 минуте. Также это значение может быть задано в конфигурационном файле *Fabric* и использоваться им для работы с кэшем.

Для работы с *MySQL Fabric Connector/Python* поддерживает следующие ключевые библиотеки и классы:

- Библиотеку `mysql.connector.fabric`: содержит все классы, функции и константы, относящиеся к *MySQL Fabric*.
- Класс `fabric.MySQLFabricConnection`: создает подключение к *MySQL*-серверу на основе информации, предоставляемой приложением.
- Класс `fabric.Fabric`: управляет подключением к узлу *MySQL Fabric*; используется классом `MySQLFabricConnection`.

## Подключимся к узлу MySQL Fabric через Connector/Python

Для создания подключения к узлу *MySQL Fabric* сначала необходимо импортировать библиотеку `mysql.connector.fabric`:

```
import mysql.connector from mysql.connector
import fabric from mysql.connector
    И далее установить MySQL-соединение с помощью метода
    mysql.connector.connect(), используя необходимые параметры
    подключения, например:
    conn = mysql.connector.connect(
        fabric={"host": "localhost", "port": 32274, "username":
        "admin", "password": "admin", "report_errors": True},
        user="group", password="group", database="fabrictest",
        autocommit=True
    )
```

Запрос на соединение с *Fabric* передает методу `connect()` следующие аргументы:

- `fabric` является словарем и содержит набор параметров, используемых для подключения к узлу *Fabric*, такие как:
  - `host` Имя хоста, к которому производится подключение.
  - `port` Номер TCP/IP порта для установки соединения к заданному хосту (по умолчанию, 32274).
  - `username` Имя пользователя, через которого производится подключение (опционально).
  - `password` Пароль пользователя, через которого производится подключение (опционально).
  - `connect_attempts` Количество попыток подключения (опционально; по умолчанию 3).
  - `connect_delay` время задержки (в секундах) между попытками подключения (опционально; по умолчанию 1).
  - `report_errors` Указывает на то, нужно ли сообщать *Fabric* об ошибках при подключении к узлу (опционально; по умолчанию `false`).
  - `ssl_ca` Файл, содержащий информацию о центре сертификации SSL (опционально).
  - `ssl_cert` Файл, содержащий файл сертификата SSL (опционально).
  - `ssl_key` Файл, содержащий ключ SSL (опционально).
  - `protocol` протокол соединения (опционально; по умолчанию `xmldrpc`). Допустимыми значениями являются `xmldrpc` (при использовании XML-RPC протокола) и `mysql` (при использовании протокола `mysql client/server`). Если используется протокол `mysql`, то значение параметра `port` по умолчанию становится 32275.
- `user` Имя пользователя для соединения с *MySQL*-сервером.
- `password` Пароль пользователя для соединения с *MySQL*-сервером.
- `database` Имя базы данных, к которой необходимо подключиться.
- `autocommit` Включение автоматической фиксации транзакций (может принимать значения `true` или `false`).

## Метод `set_property()` коннектора Python

Для доступа к базе данных коннектору необходимо предоставить следующую информацию:

- » Наименование ВД-группы, входящей в систему *Fabric*.
- » Наименование шардированной таблицы (или таблиц) и ключ для выбора соответствующего шарда.

Эта информация может быть предоставлена коннектору с помощью метода `set_property()` объекта `Fabric connection`. Метод `set_property()` имеет следующие аргументы:

- » `group` Наименование ВД-группы
- » `tables` Шардированная таблица (или список таблиц)
- » `mode` Режим обращения к серверу (`read/write` или `read only`)
- » `scope` Тип операций (`local` или `global`)
- » `key` Ключ шардирования для выбора строк

Аргументы `group` и `tables` являются взаимоисключающими, поэтому в методе `set_property()` должен быть определен только один из них. Использование остальных аргументов — `mode`, `scope` и `key` — зависит от того, какой из ключевых аргументов задается: `group` или `tables`.

При использовании аргумента `group` задаются следующие параметры:

- » `mode` — опционально. Если значение не задано, то по умолчанию устанавливается значение `fabric.MODE_READWRITE`.
- » `scope` — не применяется.
- » `key` — не применяется.

При использовании аргумента `tables` задаются следующие параметры:

- » `mode` Опционально. Если значение не задано, то по умолчанию устанавливается значение `fabric.MODE_READWRITE`.
- » `scope` Опционально. Если значение не задается, то по умолчанию устанавливается `fabric.SCOPE_LOCAL`.
- » `key` Если значение параметра `scope = fabric.SCOPE_LOCAL`, требуется указать ключ, по которому выбираются записи. Если значение параметра `scope = fabric.SCOPE_GLOBAL`, параметр `key` не задается.

Значениями аргумента `mode` могут быть:

- » `fabric.MODE_READWRITE` Задаёт подключение к главному серверу. Используется как значение по умолчанию.
- » `fabric.MODE_READONLY` Задаёт подключение к одному из доступных подчиненных серверов (если ни один из них не является доступным, то к главному серверу). В случае наличия нескольких подчиненных *MySQL*-серверов выполняется балансировка нагрузки запросов.

В качестве значений аргумента `scope` допускаются следующие:

- » `fabric.SCOPE_LOCAL` Локальные операции, которые применяются к записям с заданным ключом. Используется как значение по умолчанию.
- » `fabric.SCOPE_GLOBAL` Глобальные операции, которые применяются ко всем записям.

Чтобы указать *Fabric* набор серверов, с которыми будет осуществляться взаимодействие, достаточно просто указать наименование ВД-группы, к которой принадлежат эти сервера, вызвав метод `set_property()` с аргументом `group`:

```
conn.set_property(group='group-1')
```

Здесь, а также в примерах, приведенных ниже, `conn` является объектом `Fabric connection`.

Для указания таблиц и ключей шардирования используются аргументы `tables` и `key` метода `set_property()`. Шардированная таблица задается в формате `'db_name.tbl_name'`, где `db_name` — имя базы данных, `tbl_name` — имя таблицы. Если используется несколько шардированных таблиц, то они перечисляются через запятую: `['db_name.tbl_name', 'db_name.tbl_name']`. В аргументе `key` указывается значение ключа, по которому будут отбираться записи:

```
conn.set_property(tables=['fabrictest.employees'], key=40)
```

```
cur = conn.cursor()
```

```
< операции над таблицей employees с ключом emp_no=40 >
```

```
conn.close()
```

По умолчанию, все операции над данными выполняются в локальной области видимости (`local`), либо область видимости может быть явно задана аргументом `scope` метода `set_property()`. В следующем примере выполняются глобальные операции, которые производятся над всеми записями в таблице `employees`:

```
conn.set_property(tables=['fabrictest.employees'], scope=fabric.SCOPE_GLOBAL)
```

```
cur = conn.cursor()
```

```
cur.execute("UPDATE employees SET first_name =
```

```
UPPER(first_name)")
```

```
conn.commit()
```

```
conn.close()
```

В качестве режима подключения к серверу `mode` по умолчанию используется режим `read/write` (чтение/запись). Таким образом, коннектор знает, что необходимо подключиться к главному серверу. Однако режим подключения можно явно указать в методе `set_property()`, например:

```
conn.set_property(group='group-1', mode=fabric.
```

```
MODE_READWRITE)
```

```
cur = conn.cursor()
```

```
cur.execute("UPDATE employees SET first_name =
```

```
UPPER(first_name)")
```

```
conn.commit()
```

```
conn.close()
```

Приведем пример использования режима `read-only` (только чтение):

```
conn.set_property(group="group-1", mode=fabric.
```

```
MODE_READONLY)
```

```
cur = conn.cursor()
```

```
cur.execute("USE fabrictest")
```

```
cur.execute("SELECT emp_no, first_name, last_name, hire_date
```

```
FROM employees ")
```

```
conn.close()
```

Также `Connector/Python 2.0.1` или выше поддерживает такие типы ключей шардирования, как `RANGE_STRING` и `RANGE_DATETIME`. Они похожи на тип `RANGE`, но вместо целочисленного значения ключа используют значения следующих типов:

- » `RANGE_STRING` Задаёт значение ключа в формате строки в кодировке UTF-8. Например:

```
conn.set_property(tables=["fabrictest.employees"], key=u'first_
```

```
name', mode=fabric.MODE_READONLY)
```

Для этого типа ключа поддерживают строки только в формате `Unicode!` Использование любого другого типа кодировки приведет к возникновению ошибки.

- » `RANGE_DATETIME` Задаёт значение ключа в формате `datetime` или `date`. Пусть в карте шардирования для шардов установлены следующие диапазоны: `«group-1/2000-01-01, group-2/2010-01-01»`. Для вычисления шарда, который содержит информацию о сотрудниках, принявших работу после 2010 г., необходимо в качестве ключа `key` указать следующее:

```
conn.set_property(tables=["fabrictest.employees"], key=datetime.
```

```
date(2010, 1, 1), mode=fabric.MODE_READONLY)
```

Если в качестве границ диапазонов, кроме даты, указывается также и время, то вызов метода `set_property()` будет выглядеть следующим образом:

```
conn.set_property(tables=["fabrictest.employees"], key=datetime.
```

```
datetime(2010, 1, 1, 12, 0, 0), mode=fabric.MODE_READONLY)
```

Для ключей шардирования типа `RANGE_DATETIME` поддерживаются только форматы `datetime.datetime` и `datetime.date`. Любого другого типа, используемый в качестве значения ключа, вызовет ошибку исполнения. **LXF**



# CRIU: Практическое руководство

Павел Емельянов представляет вам инструмент создания снимков состояния системы, позволяющих эту систему восстанавливать.



Наш эксперт

Павел Емельянов — идеолог проекта живой миграции приложений в Linux. Мы писали о нем в LXF196.



Способ живой миграции не только виртуальных машин между различными платформами, но и самих приложений разработчики компании Parallels (ныне Odin) начали искать еще в 2012 г. Существует весьма популярный проект контейнерной виртуализации *OpenVZ*, поддерживаемый компанией Virtuozzo, и у него даже есть определенные возможности по резервному копированию и восстановлению работы приложений, но именно *CRIU* изначально была призвана создать универсальный и надежный инструмент перемещения приложений между любыми рабочими средами.

*CRIU* расшифровывается как Checkpoint-Restore In Userspace. Это проект, основной целью которого является создание системной утилиты для снятия как можно более полной информации о состоянии выполняющегося приложения с целью последующего восстановления приложения по полученной информации. Примеры применения такой технологии — живая миграция контейнеров или обновление ядра на сервере без перезапуска работающих серверов.

Первое время сообщество рассматривало данную инициативу с большим скепсисом. Все же решение задачи, которую поставили

перед собой программисты, было слишком многосторонним: следовало предусмотреть массу всяческих нюансов, чтобы приложение продолжило работать без сбоев, не теряло свои данные и сохраняло безопасность ИТ-инфраструктуры, принятые политики доступа к информации. Даже Линус Торвалдс поначалу считал, что «ничего не получится», но текущие достижения и упорство команды доказали обратное. Возможно эта фраза покажется вам слишком уж бравадной, но в действительности команда *CRIU* за прошедшее время внесла немало изменений и дополнений в мир GNU/Linux. И это касается не только самого кода *CRIU*, пакет с которым уже включают различные дистрибутивы, но и различных дополнительных интерфейсов и функций ядра [kernel], созданных командой программистов для поддержки работы *CRIU*. Конечно, можно было зашить всю технологию в отдельный модуль ядра, но разработчики OpenSource всегда предпочитают сделать что-то более общее и полезное, если возникает такая возможность.

Так, благодаря *CRIU* в ядре Linux появилась возможность получать исчерпывающую информацию о сетевых соединениях, следить за тем, какие участки памяти используют отдельные приложения. Кроме этого были расширены возможности получения информации из виртуальной файловой системы *proc*, все более широко используемой OpenSource сообществом. Более того, знаком зрелости *CRIU* можно считать тот факт, что в релизе Red Hat от ноября 2015 г. инструмент *CRIU* был включен в комплект как проба технологии [tech preview]. А это значит, что наработки *CRIU* стали вызывать интерес у более широкого спектра пользователей Linux — ведь с новой версией Red Hat все желающие смогут опробовать механизм *CRIU* в действии.

Вы спросите, чем же *CRIU* отличается от других решений для резервного копирования и восстановления приложений? Ответ будет прост: *CRIU* позволяет перенести выполняющееся приложение на другой сервер практически незаметно для самого приложения. Все данные работающего приложения сохраняются в виде набора файлов и могут быть переданы куда угодно и как угодно. Такой подход снимает ограничения приложений, не написанных в популярной сегодня микросервисной архитектуре — теперь приложение не привязано жестко к серверу, на котором выполняется, а это значит, что для восстановления его функций не нужно его перезапускать, теряя время и данные — можно просто восстановить приложение, которое не только будет работать корректно, но и «не поймет», что его кто-то куда-то перенес.

За время развития проекта ребята научились делать последовательные снимки [snapshot], добавляя к образу только изменившиеся данные, сделали возможной миграцию приложения с одного сервера на другой вообще без обращения к жесткому диску — сохраняя все необходимые данные прямо в оперативной памяти. В последних версиях *CRIU* появилась возможность обрабатывать сложные комбинации из абстракций ядра, что необходимо для работы с современными приложениями. А в ближайшем будущем появится функция миграции приложения без сохраненной памяти, с последующей ее подкачкой по сети для продолжения непрерывной работы.

Конечно, такая задача порождает массу вопросов и подзадач, которые команда *CRIU* решает и сегодня. Ежеквартально выходят релизы *CRIU*, в которых появляется немало новых функций

и огромное количество доработок и исправлений. Так, в последней версии *CRIU 1.8* появилась библиотека API Python, улучшено взаимодействие с ядром и OpenVZ, исправлен целый ряд недоработок, а также повышен уровень безопасности. Все это и привело к тому, что *CRIU* появился в составе Red Hat — пусть пока и в статусе tech preview. Похоже, российской команде программистов все же удастся решить свою задачу, и, возможно, скоро мы увидим *CRIU* в составе всех остальных дистрибутивов Linux.

Помимо основной, у проекта есть и вспомогательные цели. Например, для оптимизации живой миграции нужна возможность создания последовательных снимков состояний, передачи информации о состоянии по сети, снятия только содержимого памяти и ряд других. Для обновления ядра — возможность хранить большую часть информации в памяти, а не на диске.

## Установка

Есть два способа получить *CRIU* на своей машине: сборка из исходников и установка пакета. Для получения исходных текстов тоже есть два пути — либо клонировать себе весь репозиторий, либо скачать архив с определенной версией. В первом случае можно будет легко внести какие-нибудь изменения в код и, таким образом, поучаствовать в проекте как разработчик; второй способ больше подходит для сборки пакета.

### Репозиторий

Репозиторий с *CRIU* хранится на GitHub. Для его скачивания надо выполнить команду

```
$ git clone https://github.com/xemul/criu
Cloning into criu...
remote: Counting objects: 32754, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 32754 (delta 4), reused 0 (delta 0), pack-reused 32743
Receiving objects: 100% (32754/32754), 8.40 MiB | 814.00 KiB/s, done.
Resolving deltas: 100% (24026/24026), done.
Checking connectivity... done.
```

После этого в директории **criu/** появится код проекта.

### Архив

Архивы с исходниками лежат на сайте проекта openVZ. Чтобы скачать версию 1.8, выполняем

```
$ wget http://download.openvz.org/criu/criu-1.8.tar.bz2
--2016-01-25 13:50:36-- https://download.openvz.org/criu/criu-1.8.tar.bz2
Resolving download.openvz.org (download.openvz.org)...
199.115.104.11
Connecting to download.openvz.org (download.openvz.org)|199.115.104.11:443... connected.
```

```
HTTP request sent, awaiting response... 200 OK
Length: 534200 (522K) [application/x-bzip2]
Saving to: 'criu-1.8.tar.bz2'
criu-1.8.tar.bz2 100% [=====>] 521.68K 75.9KB/s in 6.9s
```

После этого распаковываем скачанный архив —

```
$ tar xjf criu-1.8.tar.bz2
и получаем исходники в директории criu-1.8/.
```

### Сборка

Для сборки понадобится компилятор *gcc*, утилиты *make* и библиотека для работы с google protocol buffers или protobuf. Protobuf — это двоичный формат хранения данных, разработанный Google. *CRIU* использует этот формат для хранения информации о состоянии приложений.

Точный список пакетов зависит от дистрибутива; нужны пакеты с основными библиотеками для работы, с компилятором файлов **.proto**, и пакеты для разработчиков (*-devel*). Полный список пакетов доступен на сайте <https://criu.org/Installation>.

После установки зависимостей достаточно будет выполнить команду *make* в директории с исходными текстами:

```
$ make
...
LINK criu
make[1]: Leaving directory `/root/criu'
make[1]: Entering directory `/root/criu'
make[2]: Entering directory `/root/criu/pycriu'
make[3]: Entering directory `/root/criu/pycriu/images'
GEN magic.py
make[3]: Leaving directory `/root/criu/pycriu/images'
make[2]: Leaving directory `/root/criu/pycriu'
make[1]: Leaving directory `/root/criu'
```

Результат работы — исполняемый файл **criu**.

## Установка пакета

В некоторых дистрибутивах *CRIU* присутствует в виде пакета. Надо воспользоваться пакетным менеджером, чтобы установить его. Обратите внимание, что *CRIU* все еще довольно молодой проект, и дистрибутивы не всегда успевают обновлять свои пакеты до последних версий.

### Проверка перед использованием

Во-первых, следует отметить, что вследствие политик безопасности, накладываемых ядром, полноценная работа с *CRIU* возможна только при наличии прав суперпользователя, так что надо будет либо позаботиться о том, чтобы текущий пользователь

»

» Подпишитесь на печатную или электронную версии на [www.linuxformat.ru/subscribe!](http://www.linuxformat.ru/subscribe!)



› Вывод команды `criu check` сразу после сборки.

```
CC      util.o
CC      uts_ns.o
CC      vdso.o
LINK    built-in.o
make[2]: Leaving directory '/root/criu/criu'
LINK    criu
make[1]: Leaving directory '/root/criu/criu'
make[1]: Entering directory '/root/criu/lib'
GEN     lib-py
GEN     c/built-in.o
make[2]: Entering directory '/root/criu/lib/py'
make[2]: Entering directory '/root/criu/lib'
make[3]: Entering directory '/root/criu/lib/py/images'
GEN     magic.py
DEP     c/criu.d
make[3]: Leaving directory '/root/criu/lib/py/images'
make[2]: Leaving directory '/root/criu/lib/py'
CC      c/criu.o
CC      /root/criu/images/rpc.pb-c.o
LINK    c/built-in.o
make[2]: Leaving directory '/root/criu/lib'
LINK    c/libcriu.so
make[1]: Leaving directory '/root/criu/lib'
[root@localhost criu]# ./criu/criu check
Warn (autofs.c:77): Failed to find pipe_ino option (old kernel?)
Looks good.
[root@localhost criu]#
```

обладал правами на расширение через `sudo`, либо войти в систему от имени `root`.

Во-вторых, для работы *CRIU* требуется определенная поддержка от ядра. За несколько лет существования проекта все требуемые расширения уже попали в основную ветку ядра Linux и принесли пользу не одному только проекту *CRIU*. Например, был существенно переработан (фактически переписан с нуля) интерфейс получения из ядра информации об открытых сетевых соединениях, что позволило утилите `ss` наконец-то показывать соединения сокетов Unix.

Тем не менее, не все эти изменения доступны во всех дистрибутивах. Например, уже можно работать в Fedora 19 и новее, в RHEL — начиная с версии 7, в Ubuntu — с Vivid Vervet. Для проверки готовности вашего ядра к работе с *CRIU* надо выполнить команду

```
# ./criu check
Looks good.
```

Если, как здесь показывает вывод команды, никаких проблем не было выявлено, то можно продолжать; в противном случае, какая-то функциональность может не заработать. Предупреждения, с которыми работа принципиально возможна, могут быть такими:

```
Error (cr-check.c:683): Dumping seccomp filters not supported:
Input/output error
```

Данное сообщение говорит о том, что не поддерживаются фильтры `seccomp`. Если приложение их не использует (а скорее всего это так), то работать с *CRIU* по-прежнему можно.

```
Error (cr-check.c:780): AIO remap doesn't work properly
```

Это — о том, что нет возможности восстановить область, куда ядро складывает события от AIO (асинхронный ввод-вывод). Опять-таки — это не критично, поскольку не все приложения используют AIO.

```
Error (cr-check.c:826): CLONE_PARENT | CLONE_NEWPID don't
work together
```

Ошибка означает, что ядро не поддерживает указанную комбинацию `clone`-флагов при создании процесса. *CRIU* пользуется этой комбинацией для восстановления контейнеров *LXC* или *Docker*, для простых экспериментов такое не потребуется.

## Первые запуски

Сразу начать работать с *CRIU* может быть проблематично — далеко не любое приложение можно просто взять и сохранить в формате *CRIU*. Может потребоваться подготовка окружения, в котором запущено приложение, или понадобится использовать для *CRIU* специальные опции. Поэтому для начала мы попробуем запустить тест.

### Запуск теста

Для тестирования *CRIU* используется большой набор разнообразных тестов, все они находятся в директории `test/`. Основные тесты — т.н. `Zdtm`-тесты. Они представляют собой набор небольших программ, каждая из которых делает нечто простое и, после сохранения состояния и восстановления, проверяет, что требуемое ей «состояние» не испортилось. Один из таких тестов мы и запустим.

Для начала тесты нужно собрать:

```
# cd test/ && make zdtm_ct && make -C zdtm/ && cd -
...
```

Ошибки компиляции если и возникают, то, как правило, связаны с отсутствием нужных библиотек для разработчиков, например `libaio-devel` или `libcap-devel`, но их легко установить через стандартный пакетный менеджер.

```
[root@localhost test]# ./zdtm.py run -t zdtm/static/env00 -f h --keep-img always
=== Run 1/1 ===
===== Run zdtm/static/env00 in h =====
Start test
./env00 --pidfile=env00.pid --outfile=env00.out --envname=ENV_00_TEST
Run criu dump
Run criu restore
Wait for zdtm/static/env00 to die for 0.100000
===== Test zdtm/static/env00 PASS =====
[root@localhost test]#
```

› Пример запуска простейшего теста.

После этого попробуем запустить простейший тест, проверяющий, что переменная окружения `[environment]` не потерялась в процессе `checkpoint-restore`. Обратите внимание на опцию `-f h` — она говорит, что тест должен быть запущен без какого-либо дополнительного окружения (аргумент `h`). Есть два окружения, в которых мы так же проверяем работоспособность тестов — в обычном контейнере (аргумент `ns`) и в безопасном контейнере (аргумент `uns`), это требуется для автоматической проверки способности *CRIU* не просто снимать состояние с теста и восстанавливать его, а и обеспечивать работу процессов, запущенных в контейнере.

```
# ./test/zdtm.py run -t zdtm/live/static/env00 -f h
=== Run 1/1 ===
===== Run zdtm/live/static/env00 in h =====
Start test
./env00 --pidfile=env00.pid --outfile=env00.out
--envname=ENV_00_TEST
Run criu dump
Run criu restore
Wait for zdtm/live/static/env00 to die for 0.100000
Removing dump/zdtm/live/static/env00/72
===== Test zdtm/live/static/env00 PASS =====
```

Если тест прошел, значит, у нас все получилось, но все промежуточные файлы, включая файлы с состояниями, уже удалены. Чтобы посмотреть, что там все-таки было, надо добавить к запуску ключ `--keep-img`:

```
# ./test/zdtm.py run -t zdtm/live/static/env00 -f h --keep-img
always
=== Run 1/1 ===
```

» Пропустили номер? Узнайте на с. 108, как получить его прямо сейчас.

```

===== Run zdtm/live/static/env00 in h =====
Start test
./env00 --pidfile=env00.pid --outfile=env00.out
--envname=ENV_00_TEST
Run criu dump
Run criu restore
Wait for zdtm/live/static/env00 to die for 0.100000
===== Test zdtm/live/static/env00 PASS =====

```

После этого в директории `test/dump/$(имя_теста)/$(номер_запуска)/$(номер_итерации)/` останутся промежуточные файлы. Какие? Посмотрим:

```

# ls test/dump/.../72/1/
core-72.img fs-72.img pagemap-72.img restore.cropt
stats-restore
dump.cropt ids-72.img pages-1.img restore.log
dump.log inventory.img ptree.img sigacts-72.img
fdinfo-2.img mm-72.img reg-files.img stats-dump

```

Файлы `dump.log` и `restore.log` содержат отладочную печать от самого *CRIU*, это обычные текстовые файлы. В файлах `dump.cropt` и `restore.cropt` содержатся опции командной строки, с которыми была запущена утилита для сохранения и восстановления состояний. Остальные файлы — двоичные, для их прочтения необходимо воспользоваться утилитой *CRIT*, о чем мы расскажем в одном из следующих уроков.

Остальные тесты запускаются так же. Чтобы узнать их имена, надо скомандовать

```

# ./test/zdtm.py list
zdtm/live/transition/ipc
zdtm/live/transition/ptrace
zdtm/live/transition/file_read
zdtm/live/transition/fork
...
zdtm/live/static/uptime_grow
zdtm/live/static/pipe02
zdtm/live/static/groups
zdtm/live/static/vt

```

Всего существует более 200 тестов.

## Опции командной строки — теория

Теперь попробуем разобраться в том, как вообще пользоваться утилитой. Вот `help`:

```
# ./criu --help
```

Синтаксис «стандартен» для современных многофункциональных утилит. Сначала надо указать требуемое действие, потом опции к нему. Самые интересные действия:

- » `dump` Выполняет снятие состояния с процессов.
- » `pre-dump` То же самое, но, во-первых, процессы в течение снятия состояния не остановлены, а во-вторых, снимается не полное состояние, а та его часть, изменения в которой можно легко обнаружить на следующем пре-дампе или дампе. Это оптимизация, требующаяся для живой миграции — на снятие полного состояния и передачу его по сети должно уйти как можно меньше времени, поэтому часть состояния предварительно считывается с неостановленных процессов.

- » `restore` Восстановление процессов.

- » `check` Проверка готовности ядра, это мы уже пробовали.

Остальные действия пока оставим. Теперь — опции:

- » `--tree $pid` Сообщает *CRIU* PID процесса, начиная с которого надо снимать состояние. *CRIU* сохраняет информацию не об одном процессе, а о поддереве, начиная с заданного.

- » `-D|--images-dir $dir` Директория для файлов-образов. По команде `dump` образы туда пишутся, по `restore` оттуда читаются. По умолчанию *CRIU* работает с текущей директорией, но файлов обычно много, и они способны затеряться среди существующих, так что

```

[root@localhost criu]# ./criu/criu
Usage:
  criu dump|pre-dump -t PID [<options>]
  criu restore [<options>]
  criu check [--feature FEAT]
  criu exec -p PID <syscall-string>
  criu page-server
  criu service [<options>]
  criu dedup
  criu lazy-pages -D DIR [<options>]

Commands:
  dump          checkpoint a process/tree identified by pid
  pre-dump      pre-dump task(s) minimizing their frozen time
  restore       restore a process/tree
  check         checks whether the kernel support is up-to-date
  exec         execute a system call by other task
  page-server   launch page server
  service       launch service
  dedup         remove duplicates in memory dump
  cpuinfo dump  writes cpu information into image file
  cpuinfo check validates cpu information read from image file

Try -h|--help for more info
[root@localhost criu]#

```

» Вывод команды `criu` без аргументов. Это короткая версия `help`-текста.

не указывать эту опцию имеет смысл только при запуске *CRIU* из пустой директории.

- » `-o|--log-file $file` Имя файла, куда *CRIU* будет писать информацию о своей работе. Это очень полезная опция, так как информации печатается много, и в случае ошибки `dump` или `restore` разобраться, что не заработало, без этой информации невозможно. При указании относительного имени файл будет помещен в директорию с образами.

- » `-v$NR` Уровень «разговорчивости» *CRIU*: от 0 (ничего не печатать) до 4 (печатать все).

## Опции командной строки — практика

Давайте посмотрим, с какими опциями *CRIU* было запущено во время тестов.

```

# cat test/dump/.../dump.cropt -o dump.log -D dump/zdtm/live/
static/env00/72/1 -v4 -t 72
# cat test/dump/.../restore.cropt -o restore.log -D dump/zdtm/live/
static/env00/72/1 -v4 --pidfile /root/criu/test/zdtm/live/static/
env00.pid --restore-detached

```

Опции `-t`, `-o`, `-D` и `-v` уже были рассмотрены выше. Опция `--pidfile` говорит *CRIU*, в какой файл сохранить PID процесса после восстановления. Дело в том, что запускатель тестов запускает их *всех* в микроконтейнерах, и при восстановлении на хосте PID процесса изменится. Опция `--restore-detached` заслуживает отдельного описания.

**Восстановленное дерево.** Когда *CRIU* восстанавливает дерево процессов, то первым делом создается процесс-корень этого дерева, а сам процесс *CRIU* «дирижирует» восстановлением. Таким образом, к концу процедуры восстановления в системе будет существовать как минимум два новых процесса — сам *CRIU*, который будет дочерним к запустившему его, и корень восстановленного дерева. Есть три варианта подселения этого процесса в систему. Вариант первый — корень становится дочерним к *CRIU*, и *CRIU* ждет его завершения. Так *CRIU* работает по-умолчанию. Вариант второй — корень отдается дочерним к `init`'у. Для этого достаточно, чтобы *CRIU* сразу после восстановления завершился. Именно это задает опция `--restore-detached`. Второй вариант позволяет вызвавшему *CRIU* процессу узнать о завершении восстановления, что и требуется для системы запуска тестов `zdtm.py` — после выхода *CRIU* она начинает «проверять», что восстановленный тест восстановлен правильно.

Оба этих варианта обладают одним и тем же недостатком: процесс, вызвавший *CRIU*, лишается возможности взаимодействовать с восстановленным поддеревом как со своим потомком. Это критично для, например, демонов *LXC* и *Docker*, для которых все контейнеры являются процессами-потомками, на чем и построена вся логика их работы. Для такого случая в *CRIU* есть возможность восстановить процесс-корень как дочерний для вызывающего процесса — `--restore-sibling`. Но для этого нужна поддержка ядра, о которой мы писали выше. **LXF**



# Swift: Опрос API для GitHub

Попробуйте свои силы в разработке утилиты для настоящего API на Swift для Linux, под руководством Пола Хадсона (который немного схитрит, где надо).



## Наш эксперт

**Пол Хадсон** — удостоенный наград разработчик и автор, который хочет заставить весь мир перейти на оболочку Z. Сдавайтесь, пользователи Bash!



Swift — язык программирования-мармит. На языке британцев это означает, что вы или полюбите, или возненавидите его, но вряд ли скажете: «Хм, ну и что». Для столь резкого разделения мнений есть веские причины: это язык от Apple (бэ-ээ!), но он открыт (ух ты!); он типобезопасный (ух ты!), но необязательные типы и замыкания путают пользователей (бэ-ээ!); его синтаксис часто меняется (бэ-ээ!), но он быстро компилируется (ух ты!); и так далее.

Раз вы читаете эту статью, то хотите познакомиться с новым языком программирования, и мы надеемся, что вы читали учебник Михалиса Цукалоса в прошлом номере [«Академия кодинга», стр. 80 LXF208]. Если нет, прочтите его на LXF DVD: мы предполагаем, что вы уже знакомы с этим руководством, и не будем пересказывать, как установить Swift или каковы его основные функции. Вместо этого мы воспользуемся вашими новообретенными знаниями в Swift, чтобы создать настоящий проект, который получает данные отправки GitHub для репозитория и интерактивно отображает их для пользователя. Предупреждение: Swift для Linux все еще находится на стадии раннего альфа-тестирования. Да, с ним весело повозиться, и мы считаем, что лучше сделать это загодя, но если вы когда-нибудь пользовались Swift в OS X или iOS, вы заметите огромные пробелы в функциональности, которые все еще заполняются. Но подробнее об этом позже!

Откройте окно терминала и создайте каталог `gitcommits`. Перейдите в него и выполните команду `touch Package.swift`. Этот файл объясняет Swift, как настроить сборку проекта, но если файл пуст, Swift использует разумные настройки по умолчанию.

## Стандартный код и необязательные типы

В подходящем текстовом редакторе создайте файл `main.swift` в каталоге `gitcommits`. Отдельные классы и структуры стоит помещать в собственные файлы, но чтобы упростить наш пример, мы поместим весь код в файл `main.swift`.

Мы создадим новый класс под названием `GitCommits`, который будет получать и выводить данные отправки. Для Swift в iOS эта задача была бы простой, но в Linux довольно много чего не реализовано, так что мы будем использовать обходные решения. Добавьте в начало файла `main.swift` две таких строки:

```
import Foundation
import Glibc
```

Первая включает подборку стандартных функций, на которые мы будем опираться, а вторая импортирует стандартную библиотеку C и будет использоваться для обхода некоторых недостатков реализации Swift в Linux.

Для начала объявим новый тип структуры `GitCommits`. При создании такой структуры мы передадим ей имя репозитория для считывания данных и заставим ее выводить сообщение о том, что все работает. Добавьте следующий код в файл `main.swift` под двумя строками, которые там уже есть:

```
struct GitCommits {
    init(repo: String) {
        print(«Fetching \(repo)...»)
    }
}
```

Разработчики Swift обычно предпочитают структуры классам, потому что структуры используют значения, а не ссылки. То есть, если я попробую скопировать свою структуру в другую переменную, у меня будут два независимых значения, а не две переменные, указывающих на одни и те же данные. В прошлой статье вы видели интерполяцию строк в Swift, и я не буду повторяться.

Мы хотим создать экземпляр этой структуры, передав имя репозитория. Его можно закодировать принудительно — например, `twostraws/hackingwithswift` — но гораздо лучше, чтобы его вводил пользователь. Поэтому поместите этот код сразу после добавленной структуры:

```
print(«Пожалуйста, введите репозиторий GitHub для запроса:»)
if let entry = readLine() {
    let commits = GitCommits(repo: entry)
}
```

Функция `readLine()`, которую мы видели выше, входит в стандартную библиотеку Swift, принимает строку ввода от пользователя и возвращает `String?`. Как вы помните по предыдущей статье,

## Функции первого класса

На нашем уроке вы видели строки и целые числа, но в Swift также можно хранить в переменных функции. Можно даже передавать функции в качестве параметров и получать их обратно в качестве возвращаемых значений как обычные строки или целые числа.

Рассмотрев код на **LXF DVD**, вы убедитесь, что я продвинул этот проект еще на шаг вперед, сделал так, что метод `present()` принимает необязательную функцию: если функция указана, она используется для фильтрации массива `filteredCommits`, чтобы отображались только некоторые значения.

В противном случае отображается весь массив. Синтаксис поначалу немного озадачивает:

```
func present(filter: ((Commit) -> Bool)?) {
```

Это означает следующее: «параметр фильтра может быть либо `nil`, либо функцией, которая принимает экземпляр `Commit` и возвращает `true` или `false`». Когда он пропускается через метод `filter()` Swift, то любая отправка, которая возвращает `false`, будет удалена из результирующего массива.

С помощью этого метода совсем несложно добавить в метод `mainLoop()` код для обработки различных вариантов ввода: мы просто перехватываем

каждое нажатие клавиши и затем вызываем `present()` с различными фильтрами. Swift знает, что каждая функция фильтра должна работать с одним экземпляром `Commit`, поэтому в коде фильтрации используется два удобных сокращения:

1) внутри функции фильтра отправка обозначается как `$0`, и 2) если вы напишете всего одну строку кода, ее значение автоматически считается возвращаемым значением фильтра, например:

```
$0.email.containsString("@apple")
```

Это выражение вернет `true`, если адрес электронной почты для отправки содержит `@apple`.

знак вопроса имеет значение: обычная строка может не содержать ни одного символа (""), может содержать одно слово ("Hello") или все сочинения Шекспира. Необязательная строка — которая записывается как `String?` — может иметь все эти значения, но также может содержать `'nil'`, что означает «значения нет». Важно понимать разницу между «пустой строкой» и «отсутствием значения».

В случае с `readLine()` вы можете получить пустую строку, и это означает, что пользователь нажал `Enter`, ничего не введя. Если пользователь что-то ввел, вы получите строку ввода. Но вы также можете получить значение `'nil'` — если пользователь нажал `Ctrl+d`, чтобы обозначить конец ввода.

Очевидно, что `'nil'` нельзя рассматривать как строку: ее нельзя измерить, нельзя прочесть и нельзя преобразовать. Это просто пустая память, а не строка. Поэтому Swift заставляет вас использовать необязательные типы безопасно, разворачивая их, что мы и делаем в строке `if let`. Эта строка означает «получить результат `readLine()`, и если результат содержит какое-то значение, развернуть его и записать в переменную». Поэтому, хотя `readLine()` возвращает `String?`, результатом будет обычная строка (`String`) — необязательный тип был развернут. Узнав, что мы получили настоящую строку, мы создаем экземпляр `GitCommits` с данной строкой, и это означает, что будет вызвана функция `print()` для вывода некоторой отладочной информации.

Вернувшись в терминал, выполните команду `swift build` для компиляции, а затем — `.build/debug/gitcommits` для запуска программы. Swift автоматически назовет двоичный файл по имени каталога. После запуска приложения вы можете набрать что-нибудь и нажать `Enter`, и вы должны увидеть, как тот же текст появится на экране.

В последнее время мы много работали с API GitHub, так как это один из немногих API с данными, которые интересны разработчикам (кто изменил исходный код и почему), и он полностью открыт — любой желающий сможет воспользоваться им без регистрации для получения логина и пароля. При таком подходе частота запросов ограничена 60 запросами в час, но для наших целей этого более чем достаточно.

Нам также понадобится внешняя библиотека для разбора JSON, поэтому найдите файл `TidyJSON.swift` на **LXF DVD** и скопируйте его в каталог `gitcommits`. `TidyJSON` — открытая библиотека, позволяющая легко разбирать JSON, и нам она отлично подходит. Для добавления библиотеки в проект просто скопируйте файл в каталог: Swift автоматически извлекает все файлы исходного кода, которые найдет.

Обработанный код JSON мы поместим в отдельные экземпляры новой структуры `Commit`. В ней будут храниться имя и адрес элек-

тронной почты отправителя, сообщение и дата. Добавьте следующий код в `main.swift` перед существующим классом `GitCommits`:

```
struct Commit {
    var name: String
    var email: String
    var message: String
    var date: String
}
```

Определив эту новую структуру, мы можем обновить структуру `GitCommits`, чтобы в ней хранился массив `Commits` — добавьте следующий код прямо перед методом `init()`:

```
var commits = [Commit]()
```

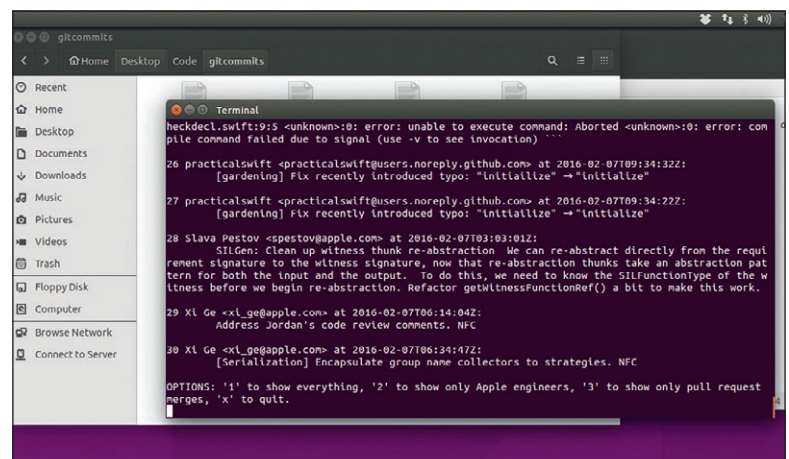
## Получение JSON

Пока метод `init()` делает только одно: выводит сообщение с именем репозитория, из которого будут получены данные. Нам надо добавить в этот метод следующие действия:

- 1) Очистка существующего массива данных для отправки.
- 2) Получение данных для отправки с помощью GitHub API.
- 3) Преобразование JSON в объекты `Commit`.

Первое действие простое, второе — чуть сложнее из-за ограничений Swift в Linux, а третье — сложное, потому что сначала придется кое-чему научиться. Разберемся с ними по порядку: оставив существующий вызов `print()` в `init()`, добавьте под ним следующий вызов:

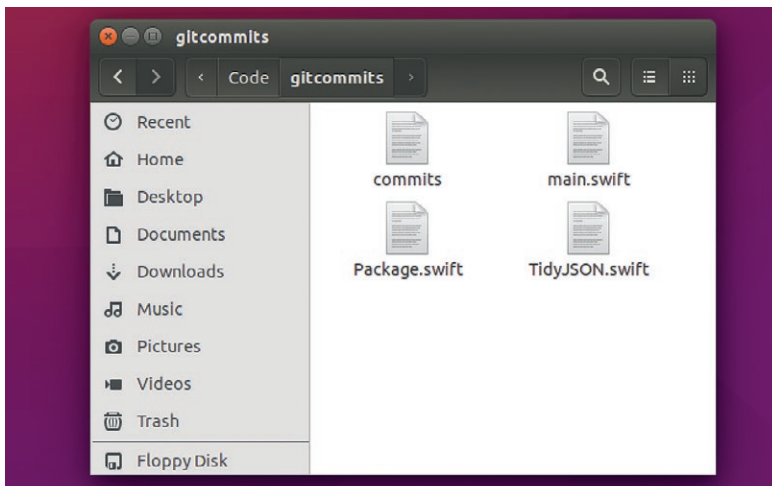
```
commits.removeAll()
```



» Наша законченная программа запускается из командной строки и позволяет просматривать и фильтровать отправки для GitHub. Дополнительные функции см. на **LXF DVD**!

» Подпишитесь на печатную или электронную версии на [www.linuxformat.ru/subscribe!](http://www.linuxformat.ru/subscribe!)





► В каталоге нашего проекта должен быть файл `Package.swift`, но если вы не хотите ничего делать, просто создайте пустой файл, и Swift использует разумные параметры по умолчанию.

У Swift есть отлично документированный API, и назначение этой функции должно быть понятно: она удаляет все элементы из массива отправок. С первой задачей мы разобрались! Мы сказали, что второе действие будет сложным из-за ограничений Swift в Linux. Как мы упомянули ранее, Swift для Linux находится на ранней альфа-стадии разработки, и следующий шаг усложняется из-за отсутствия некоторых функций. Обычно строки в Swift имеют встроенный метод загрузки данных с удаленного сервера, который отлично подходит для получения данных API GitHub.

Увы, этот метод пока что недоступен в Linux, как недоступны и несколько удобных альтернатив. Поэтому нам придется схитрить: мы воспользуемся `system()`, стандартной библиотечной функцией C, чтобы вызвать программу командной строки `curl` для получения текста. После этого текст можно загрузить в строку Swift, так как это API реализовано. Да, это всего лишь «хак», но он прост и решает нашу проблему — ура!

Добавьте такие две строки под той, которую вы только что написали:

```
let urlString = "https://api.github.com/repos/(repo)/commits"
system("curl -s \(urlString) > commits")
```

Оба метода используют интерполяцию строк для выполнения команды в командной строке. Следует подчеркнуть, что это далеко не идеальное решение, так как оно допускает выполнение пользователями почти любой команды, какую они захотят, с помощью функции `system()`. Впрочем, если они могут запустить программу Swift, то могут запустить и все остальное, так что это не такая большая проблема. Кстати, параметр `-s` команды `curl` означает «тихий режим»: в этом режиме она перестает выводить данные на экран.

Загрузив данные отправки, надо поместить их в строку, а затем передать ее для разбора в TidyJSON. Первое быстро и легко делается в Swift, второе — довольно необычным способом в TidyJSON, и вам надо познакомиться с тем, и с другим.

Добавьте следующие пять строк в метод `init()` под предыдущими строками:

```
if let contents = try? String(contentsOfFile: "commits", encoding:
NSUTF8StringEncoding) {
    if let json = JSON.parse(contents).0 {
        // Здесь должен быть код
    }
}
```

В первой строке Swift загружает содержимое текстового файла. Все было бы чуть проще, будь реализация Swift в Linux не такой поверхностной, но с текущим положением дел надо явно указывать Swift, что файл использует кодировку UTF-8. Часть `try?` означает следующее: «этот вызов может завершиться неудачно, например, если файла не существует; в таком случае надо вернуть `nil` вместо строки». Затем можно воспользоваться приведенным

ранее синтаксисом `if/let`, чтобы убедиться, что у нас на самом деле есть значение.

Вторая строка посвящена обработке ошибок в TidyJSON. Вместо использования `try` TidyJSON возвращает два значения для одного вызова метода: дополнительный объект JSON, представляющий разобранный строку, и дополнительную строку с сообщением об ошибке, если оно было. Нам некогда возиться с сообщением об ошибке, поэтому мы просто прочитаем первое значение, представленное как 0 в возвращаемом значении `parse()`: его обработка с помощью `if/let` позволяет быть уверенными в том, что мы получим действительные объекты после разбора строки отправки.

## Создание экземпляров отправки

Мы загрузили данные API из GitHub, поместили их в строку и разобрали ее с помощью TidyJSON. А значит, теперь мы можем преобразовать эти данные в экземпляры `Commit`, но для этого надо познакомиться с тремя вещами: быстрыми перечислениями, объединением с `nil` и поэлементной инициализацией.

Быстрое перечисление существует во многих языках, и это способ перебора массива. Обычно это означает «перебрать массив, поместив каждый элемент в переменную, чтобы я смог поработать с ней». Но в TidyJSON все немного сложнее, так как вы получаете две вещи: ключ и значение. Это вызвано тем, что JSON так же часто, как и массивы, использует словари. Наш JSON будет представлять собой массив, поэтому можно попросить Swift проигнорировать ключ, указав для него специальное имя переменной, подчеркивание `'_'`:

«Объединение с `nil`» звучит не слишком понятно, но это настолько просто и удобно, что вы будете часто им пользоваться. Как вы видели, необязательные значения в Swift очень распространены: «Это целое число может содержать возраст человека или же `nil`, так как его возраст нам неизвестен». Но для безопасного использования этих значений нужно разворачивать их с помощью `if/let`, что бывает утомительно. Поэтому в Swift есть особый оператор `??`, позволяющий указать значение по умолчанию, которое будет использовано, если другое значение равно `nil`. Поэтому можно написать такой код:

```
let age = getAge() ?? 21
```

Это означает: «выполнить функцию `getAge()` и поместить его результат в константу `age`. Но если эта функция возвращает `nil` вместо целого числа, использовать число 21 вместо `nil`». Это означает, что константа `age` всегда будет содержать `Int`, а не `Int?`. То есть, этот тип никогда не будет необязательным, так как либо `getAge()` вернет действительный результат, либо будет использовано значение по умолчанию.

Объединение с `nil` очень полезно при разборе JSON, потому что TidyJSON возвращает почти всё как необязательные переменные. Дело не в том, что TidyJSON противоречиво, а в том, что обработка текста по своей природе небезопасна: значение, которое есть здесь сейчас, завтра может исчезнуть или быть переименовано. Помните, что Swift хочет защитить ваш код от «падений», поэтому каждый раз при запросе значения TidyJSON мы будем использовать объединение с `nil`, чтобы при отсутствии значения возвращалось разумное значение по умолчанию.

Наконец, поэлементная инициализация — это полезная функция структур Swift: мы определили четыре свойства для нашей структуры (имя, адрес электронной почты, сообщение и дата), и Swift позволит автоматически создать экземпляры `Commit` с использованием этих четырех значений.

Но довольно разговоров: пора написать какой-нибудь код. Откройте адрес <https://api.github.com/repos/apple/swift/commits> в своем браузере, и вы увидите структуру данных GitHub, затем добавьте следующий код, взамен комментария «вставьте дополнительный код сюда» в методе `init()`:

```
for (_, value) in json {
```

### Скорая помощь

Git — система управления версиями; она отслеживает изменения в исходном коде, над которым совместно работают группа разработчиков. Каждый раз, когда кто-то выполняет изменения, он пишет небольшой комментарий о том, что изменилось, и затем отправляет эти изменения в GitHub в одной связке. Эта связка называется «отправкой [commit]».

```

let name = value["commit"]["committer"]["name"].string ??
"Anonymous"
let email = value["commit"]["committer"]["email"].string ??
"tcook@apple.com"
let date = value["commit"]["committer"]["date"].string ??
"1984-01-24T10:00:00Z"
var message = value["commit"]["message"].string ?? "Я улуч-
шил Linux Swift."
message = message.stringByReplacingOccurrencesOfString("\n",
withString: " ")
let commit = Commit(name: name, email: email, message: mes-
sage, date: date)
commits.append(commit)
}

```

Как видите, TidyJSON позволяет углубиться в отдельные отправки, мы ищем имя отправителя (`commit > committer > name`) в структуре JSON, а затем получаем для него строковое значение. Если любое из этих полей по какой-либо причине не существует, TidyJSON возвратит `'nil'`, и здесь наш оператор объединения с `nil` возьмется за дело и возвратит разумное значение по умолчанию: `"Anonymous"`, `tcook@apple.com` и т.д. Чтобы все было еще лучше, мы употребили длинный, но не требующий пояснений метод `stringByReplacingOccurrencesOfString()` для замены переносов строк на пробелы в сообщениях отправки. Если хотите, можете удалить этот метод, но тогда вывод будет гораздо сложнее читать!

## Создаем простой интерфейс

Сейчас в нашем массиве отправок много всего интересного, но программа еще ничего особенного не делает, так как конечному пользователю ничего не видно. Мы это исправим, выведя в терминале все отправки, а позже еще и улучшим.

В простейшей версии интерфейса мы должны выводить все отправки, отображать для пользователя какие-то варианты действий и затем спрашивать, что хочет сделать пользователь. Для считывания и обработки пользовательских команд мы воспользуемся `readLine()`, начав с нескольких команд.

Через минуту мы создадим два новых метода, но сначала вам надо познакомиться с методом `enumerate()`. Это особый вид быстрого перечисления, и вы особенно оцените его, если пытались делать нечто похожее в других языках: он перебирает каждый элемент массива, как и в обычном быстром перечислении, но заодно возвращает вам индекс этого элемента в массиве, как и при переборе массивов в C.

Добавьте следующие два метода под методом `init()`:

```

func present() {
let filteredCommits: [Commit]
filteredCommits = commits
for (index, commit) in filteredCommits.enumerate() {
print("(index + 1) \(commit.name) < \(commit.email) > at \
(commit.date):")
print("\t \(commit.message)\n")
}
showOptions()
}
func showOptions() {
print("OPTIONS: '1' to show everything, '2' to show only Apple
engineers, '3' to show only pull request merges, 'x' to quit.")
}
}

```

Метод `showOptions()` тривиален — он просто выводит инструкцию на экран. Зато метод `present()` делает несколько любопытных штук: вы видите метод `enumerate()`, с помощью которого я могу вывести на экран нумерованный список отправок. Но используемый массив отправок на самом деле представляет собой копию под названием `filteredCommits`. Да, сейчас никакой фильтрации не производится — это заглушка для кода, который вы найдете на **LXF DVD**.

## Как развивается Swift

Как вы видели на этом уроке, в Swift для Linux на данный момент не хватает некоторых базовых функций, которые считаются данностью в iOS и OS X. Здесь мы обошли их отсутствие, но Apple усиленно работает над заполнением пробелов. Но не расслабляйтесь: даже если эти пробелы будут

заполнены, Apple собирается внести серию изменений в API, которые направлены на то, чтобы сделать несколько тысяч вызовов API «более адаптированными для Swift». Надеюсь, это не повлияет на код, который мы написали, но чтобы знать наверняка, нужно подождать до июня 2016 г.

Добавив эти два метода, надо добавить третий, который обрабатывает ввод пользователя до тех пор, пока он не попросит завершить программу. При первом обращении он вызывает `present()`, затем входит в бесконечный цикл (который не завершится до тех пор, пока мы не попросим его остановиться), где будет запрашиваться и обрабатываться ввод пользователя.

Когда пользователь набирает `1`, мы выводим список отправок. Когда он набирает `x`, мы завершаем работу программы. Если он вводит что-то другое, мы (пока) просто снова выводим список вариантов. Обратите внимание, что мы преобразуем ввод пользователя в нижний регистр, чтобы избежать проблем. Добавьте третий метод под предыдущими двумя:

```

func mainLoop() {
present()
mainLoop: while true {
if let input = readLine() {
switch input.lowercaseString {
case "1":
present()
case "x":
break mainLoop
default:
showOptions()
}
}
}
}
}

```

Как видите, мы оставили небольшой фрагмент кода вам на закуску: у бесконечного цикла (`while true`) есть метка `mainLoop`. Когда пользователь вводит `x`, мы можем завершить цикл (`break mainLoop`), чтобы Swift вышел не только из блока `swift/case`, но и из всего цикла `while true` в одной строке кода. Обратите внимание, что в Swift нет явного фрагмента `case` для остальных вариантов, которым заражены такие языки, как C или PHP — от этого стоило избавиться!

Чтобы весь этот новый код заработал, достаточно изменить только первую строку кода:

```

if let entry = readLine() {
let commits = GitCommits(repo: entry)
commits.mainLoop()
}
}

```

Теперь вы должны собрать и запустить программу, чтобы увидеть ее в действии — попробуйте указать `apple/swift` в качестве имени репозитория (без кавычек), чтобы посмотреть на несколько примеров.

На этом уроке мы рассказали о многом, но есть еще больше вещей, о которых рассказать не успели. Увы, место у нас закончилось, поэтому придется схитрить: загляните во врезку «Функции первого класса» на стр. 81, и вы увидите нечто способное взорвать ваш мозг или сделать вас поклонником Swift. Как я уже говорил, очень немногие, глядя на Swift, говорят: «Хм, ну и что», и я надеюсь, он покажется вам восхитительным, полезным, современным, а главное — безопасным языком программирования. **LXF**

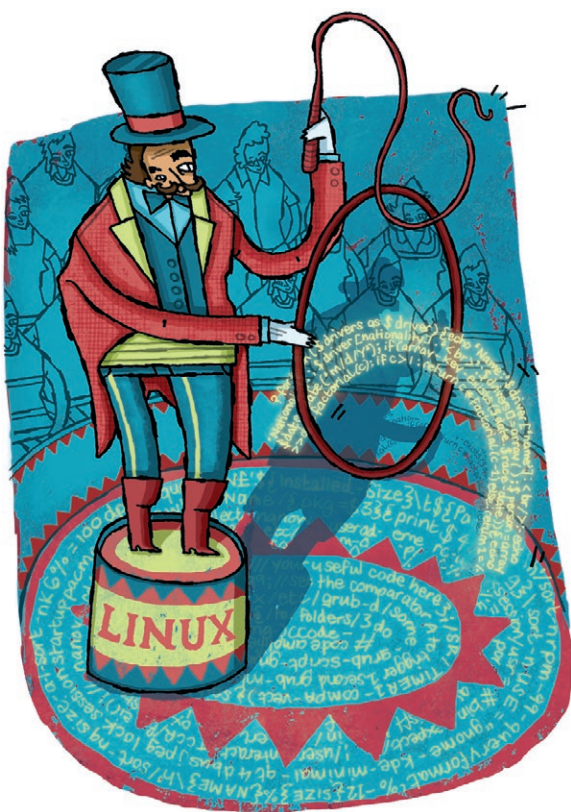
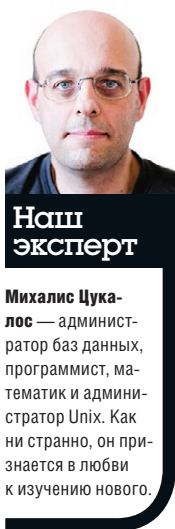
### Скорая помощь

Загрузите код для этой статьи из Интернета, зайдя на [linuxformat.com](http://linuxformat.com) или по ссылке <http://bit.ly/LXF209swift>.



# MongoDB: Создаем блог

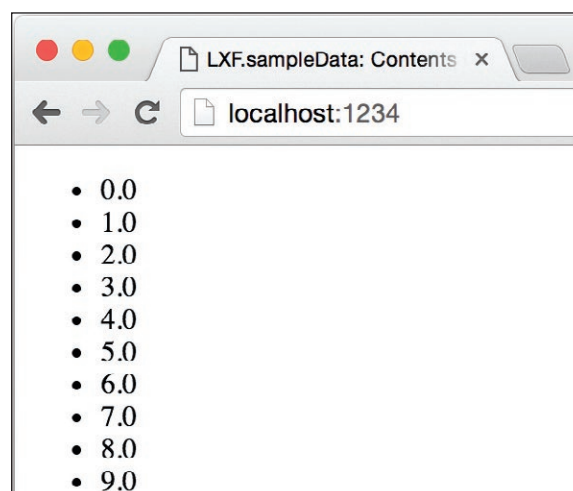
Чтобы веселее жилось, **Михалис Цукалос** добавляет в *MongoDB* быстрый и легкий фреймворк на Python под названием *Bottle*.



**В** этом руководстве мы воспользуемся накопленными знаниями о *MongoDB* и объединим их с *Bottle*, фреймворком Python, чтобы создать сайт блога, данные которого будут храниться в базе данных *MongoDB*. Фреймворк *Bottle* будет в основном отвечать за пользовательский интерфейс сайта и его логику. Далее предполагается, что *MongoDB* и *PyMongo* уже установлены в вашем дистрибутиве Linux. Если вам не очень комфортно с *MongoDB* и *PyMongo* — драйвером *MongoDB* для Python — советуем освежить в памяти две предыдущие статьи [«Академия кодинга», стр. 80 **LXF207** и «Академия кодинга», стр. 84 **LXF208**], чтобы подробнее узнать о драйвере и администрировании *MongoDB* соответственно). В конце этого урока у вас будет прекрасный блог на Python, который использует *MongoDB* для хранения данных и *Bottle* для их отображения.

## Фреймворк Bottle

*Bottle* [англ. бутылка] — быстрый, простой и легкий микро web-фреймворк WSGI [Web Server Gateway Interface], написанный на языке Python. Весь фреймворк представляет собой модуль,



➤ Это фрагмент вывода скрипта *sampleData.py*, в котором содержатся значения ключа 'x' из документов коллекции 'sampleData'.

состоящий из одного файла, и его единственной зависимостью является стандартная библиотека Python. На момент написания статьи последняя стабильная версия *Bottle* — 0.12.9, а для установки *Bottle* выполните следующую команду с правами root: `$ pip install bottle`. В Debian эта команда установит *Bottle* в файлы `/usr/local/lib/python2.7/dist-packages/bottle.py` и `/usr/local/bin/bottle.py`. Но поскольку *Bottle* не зависит ни от каких внешних библиотек Python, можно загрузить **bottle.py** и поместить его в каталог, который вы используете для разработки:

```
$ wget http://bottlepy.org/bottle.py
$ wc bottle.py
4107 15384 156332 bottle.py
```

Программа "Hello World!" в *Bottle* выглядит так:

```
from bottle import route, run, template
@route('/hello/<user>')
def index(user):
    return template('<h2>Hello World from {user}!</h2>', user=user)
run(host='localhost', port=1234)
```

Импортируемый метод `run` может использоваться для запуска приложения на сервере разработки, и это лучший способ проверить свое приложение в процессе его написания. Метод `route` сообщает приложению о поддерживаемых запросах URL, а также о том, как обрабатывать их с помощью функций Python. Маршрутизация в приложениях *Bottle* реализуется путем вызова одной функции Python для каждого поддерживаемого URL-адреса.

Это не простая программа "Hello World", поскольку она также отображает имя пользователя, которое включено в URL. Как видите, можно определить пользовательскую переменную и затем

## Базовые команды CRUD

CRUD — сокращение от “Create, Read, Update and Delete [Создание, чтение, обновление и удаление]”.

Это базовые операции, которые можно выполнить с любой базой данных. При чтении этого руководства по *MongoDB* будет полезно держать в памяти базовые команды CRUD, чтобы проверить, что умеют делать и чего не умеют скрипты на Python.

С помощью метода `findOne()` можно получить один случайным образом выбранный документ из коллекции:

```
> db.sampleData.findOne()
```

В коде далее первая команда возвращает все документы из коллекции, а вторая команда — все документы, у которых ключ `n` имеет значение 324:

```
> db.sampleData.find()
> db.sampleData.find({n: 324})
```

Основное различие между `find()` и `findOne()`, которое вы обнаружите — в том, что первая функция может возвращать несколько документов с помощью курсора, а вторая — случайным образом возвращает один документ BSON.

В следующем примере вывод функции `find()` сортируется по полю `x` по убыванию:

```
> db.sampleData.find().sort({x: -1})
```

Кроме того, в *MongoDB* можно вставить документ — это делается следующим образом:

```
> db.sampleData.insert({ "x": 23, "y": 13 })
```

Аналогично можно удалить документ, который соответствует определенным критериям:

```
> db.sampleData.remove({"y": 13})
WriteResult({"nRemoved": 1})
```

Помните, что самый безопасный способ найти и удалить документ — с помощью его поля `_id`. Также можно обновить существующий документ, следующей командой:

```
> db.sampleData.update({"x": 123}, {$set: { "z": 123}})
WriteResult({"nMatched": 1, "nUpserted": 0, "nModified": 1})
```

Удобный способ обновить несколько документов сразу — установить параметр `multi` в `true` (`(multi:true)`) при использовании функции `update()` следующим образом:

```
> db.sampleData.update({"x": 23}, {$set: {"anotherKey": 54321}}, {multi:true})
```

передать ее функции `template()`. Чтобы проверить эту простую web-страницу, запустите скрипт Python следующим образом:

```
$ python hw.py
Bottle v0.12.8 server starting up (using WSGIRefServer())...
Listening on http://localhost:1234/
Hit Ctrl-C to quit.
127.0.0.1 - - [25/Jan/2016 16:28:26] "GET /hello/tsoukalos
HTTP/1.1" 200 36
```

Откройте свой любимый браузер и введите в адресной строке <http://localhost:1234/hello/tsoukalos>, чтобы увидеть только что созданную web-страницу! Обратите внимание, что при попытке открыть адрес <http://localhost:1234/hello/tsoukalos/> (со слэшем на конце) появится сообщение об ошибке, так как этот адрес не настроен.

Как вы, возможно, догадались, имя хоста и порт для простого web-сервера задаются в файле `hw.py` следующим образом:

```
run(host='localhost', port=1234)
```

Если вы захотите выводить отладочную информацию, включите режим загрузки:

```
run(host='localhost', port=1234, debug=True)
```

## Подключение Bottle к MongoDB

Подключение *Bottle* к *MongoDB* возможно с помощью драйвера *MongoDB* для Python. В следующем примере (`sampleData.py`) с помощью *Bottle* осуществляется чтение данных с сервера *MongoDB* и отображение значений ключа 'x', которые можно найти в документах коллекции 'sampleData' базы данных LXF. Сервер *MongoDB*, используемый в скрипте Python, расположен на том же компьютере, и скрипт использует порт *MongoDB* по умолчанию — 27017. Код Python файла `sampleData.py` таков:

```
import pymongo
from pymongo import MongoClient
from bottle import route, run, template
# Получаем данные из MongoDB
myData = []
client = MongoClient('localhost', 27017)
db = client.LXF
cursor = db.sampleData.find({}, {'_id':0, 'y':0})
for myDoc in cursor:
    myData.append(myDoc['x'])
@route('/')
def rootDirectory():
```

```
    return template('listContents', data=myData)
run(host='localhost', port=1234)
```

Код `myDoc['x']` используется для получения фактического значения ключа 'x', которое затем помещается в список `myData` — этот список затем передается в качестве параметра функции `template()`. Использованная команда `find()` не возвращает значений полей 'y' и '\_id' ради экономии процессорного времени, что очень удобно, когда в базе данных *MongoDB* много данных.

Как вы видите, для запуска скрипта `sampleData.py` необходим один внешний файл — так в *Bottle* организованы проекты. Хотя код внешнего файла *Bottle* можно включить в основной скрипт Python, лучше сохранить такой код отдельно, особенно если вы работаете с большими проектами, которые поддерживают несколько URL.

Использование шаблона 'listContents' с методом `template()` упрощает код файла `sampleData.py` — небольшой платой за это является необходимость редактировать несколько файлов. Содержимое файла `listContents.tpl` (расширение TPL применяется автоматически) таково:

```
<!DOCTYPE html>
<html><head>
<title>LXF.sampleData: Contents of 'x' key</title>
</head>
<body>
<ul>
%for myX in data:
<li>{{myX}}</li>
%end
</ul>
</body></html>
```

Хотя файлы шаблонов содержат преимущественно HTML-код, у них также есть доступ к переменным, таким как `data`, что позволяет создавать динамическое содержимое. Обращение к внутренней переменной `data` осуществляется в цикле `for`, который выглядит довольно просто. Данные форматируются с помощью списка HTML. Перед запуском файла `sampleData.py` по аналогии с файлом `hw.py` убедитесь, что в коллекции `sampleData` базы данных LXF есть данные. Если их нет, запустите следующий код из оболочки *MongoDB*, чтобы заполнить базу данных некоторыми данными:

```
> use LXF
switched to db LXF
> for (var i=0; i<100; i++) { db.sampleData.insert({'x':i, 'y':i}); }
WriteResult({"nInserted": 1})
```

### Скорая помощь

Подробную информацию о фреймворке *Bottle* можно найти на <http://bottlepy.org>. Подробнее о драйвере *MongoDB* для Python можно узнать на <https://docs.mongodb.org/ecosystem/drivers/python>.

» Не хотите пропустить номер? Подпишитесь на [www.linuxformat.ru/subscribe/!](http://www.linuxformat.ru/subscribe/)



```
> db.sampleData.count();
100
Если вы намерены удалить всю коллекцию sampleData целиком, чтобы начать с пустой коллекции, выполните
> db.sampleData.drop();
true
```

Так как хранение данных занимает место на диске, неплохо бы знать, как удалять в *MongoDB* целые коллекции и базы данных. Следующая команда удаляет всю базу данных LXF вместе с ее файлами данных:

```
> use LXF
switched to db LXF
> db.runCommand( { dropDatabase: 1 } )
{ "dropped" : "myData", "ok" : 1 }
```

Несмотря на похожесть между `sampleData.py` и `hw.py`, первая программа сложнее, так как для получения данных она подключается к базе данных *MongoDB*. Если вам удалось успешно запустить `sampleData.py` и получить ее вывод в своем любимом браузере, вы можете продолжить работу с остальной частью руководства; в противном случае постарайтесь исправить возможные ошибки в коде Python или конфигурации *MongoDB*.

## Обработчики, представления, формы и куки

Bottle следует шаблону программирования MVC (модель–представление–контроллер), чтобы разделить различные функции интерфейса пользователя. Модель отвечает за хранение, опрос и обновление данных, представление отвечает за отображение данных на экране и, наконец, контроллер содержит логику приложения.

Используя код Python, Bottle разделяет сайты на обработчики, представления и формы. Обработчик URL позволяет указать веб-страницы и привязать их к коду Python. Обработчиков URL может быть сколько угодно. Но если их окажется слишком много, надо заново продумать структуру веб-приложения. Как вы видели в `hw.py`, Bottle также поддерживает динамические маршруты и регулярные выражения в маршрутах. Обработчики URL определяются командой `@bottle.route()`.

Представление — это то, что видит пользователь. Это код, который связан с отображением информации и обычно представляет собой динамическое содержимое. Представление не имеет права содержать никакого кода, который должен быть в контроллере. Представления реализуются с помощью шаблонов Bottle. По умолчанию Bottle ищет шаблоны в каталогах `./` и `./views/`.

Благодаря форме приложение получает данные, вводимые пользователем. Bottle также поддерживает куки, которые бывают очень полезны.

Итак, с Bottle и структурой проектов вы познакомились; пора приступить к созданию сайта блога. Для создания структуры сайта блога выполните следующие команды:

```
$ mkdir blog
$ cd blog
$ mkdir views
$ touch blog.py
$ touch views/menu.tpl
$ touch views/listPosts.tpl
$ touch views/write.tpl
$ touch views/showsinglepost.tpl
```

Корневой каталог, содержащий все файлы и каталоги проекта, называется **blog**. Имя основного файла, содержащего правила URL — `blog.py`. Его код на Python таков:

```
import pymongo
from pymongo import MongoClient
from bottle import route, run, template, request, get, post, redirect
import datetime
from bson import ObjectId

@route('/')
def rootDirectory():
    return template('menu')

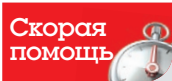
@route('/list/')
@route('/list')
def listAllPosts():
    # Получаем данные из MongoDB
    myData = []
    client = MongoClient('localhost', 27017)
    db = client.LXF
    cursor = db.blogposts.find()
    for myDoc in cursor:
        myData.append(myDoc)
    return template('listPosts', data=myData)

@get('/write/')
@get('/write')
def writeNewPost():
    return template('write', dict(subject="", body="", tags=""))

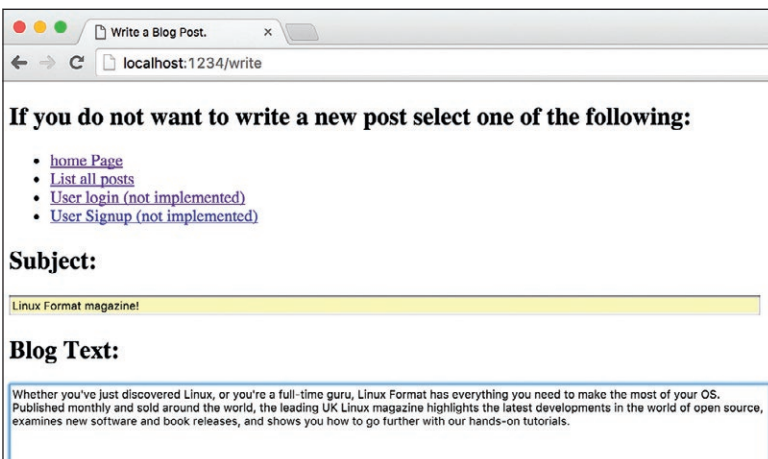
@post('/presentnewpost/')
@post('/presentnewpost')
def presentNewPost():
    # Извлекаем данные из write.tpl FORM
    title = request.forms.get("subject")
    post = request.forms.get("body")

    if title == "":
        title = "Нет заголовка!"
    if post == "":
        post = "Это текст записи."
    postDocument = { "title": title,
                    "text": post,
                    "date": datetime.datetime.utcnow()}

    client = MongoClient('localhost', 27017)
    db = client.LXF
    # Записываем их в базу данных
```



Bottle — не единственный веб-фреймворк Python. Другие фреймворки Python, на которые стоит обратить внимание — Django ([www.djangoproject.com](http://www.djangoproject.com)) и Flask (<http://flask.pocoo.org>).



➤ Возможность публиковать новые записи обязательна для блога. На рисунке показана веб-страница, которая позволяет добавлять новые записи.

➤ **Пропустили номер?** Узнайте на с. 108, как получить его прямо сейчас.



## Создание статического сайта с помощью Bottle

Хотя фреймворк Bottle разрабатывался с расчетом на динамические сайты, с его помощью можно создавать и статические. Следующий код Python (**static.py**) демонстрирует это:

```
from bottle import route, run
@route('/hello/')
@route('/hello')
def hello():
    return "<h1>Hi to you too!<h1>"
run(host='localhost', port=1234, debug=True)
```

Здесь определяются обработчики для двух URL ('/hello' и '/hello/') — это единственные адреса,

поддерживаемые **static.py**. Фактический HTML-код включен в файл PY, поэтому при изменении сайта не придется заглядывать в большое количество файлов. Кроме того, **static.py** показывает, что можно использовать несколько маршрутов с одной функцией (что и происходит в **static.py** с адресами '/hello' и '/hello/'), и это очень удобно. При попытке открыть URL-адрес, который не поддерживается **static.py**, вы получите обычную ошибку 404. Желая вывести свое сообщение об ошибке, используйте следующий маршрут, перехватывающий всё, что не было обработано предыдущими правилами:

```
@route('<mypath:path>')
def doSomething(mypath):
    print mypath
    return "Ваш URL - %s но такого не существует!"
% mypath
```

Основное преимущество этого подхода в том, что сайт легко превратить из статического в динамический, просто изменив обработчики URL.

Мы также обнаружили, что с помощью Bottle удобно и несложно создавать прототип динамического сайта, прежде чем приступить к процессу реализации.

```
db.blogposts.insert_one(postDocument)
# Получаем postID, это поле _id документа
postid = str(postDocument['_id'])
# print(postid)
# Представляем документ
redirect('/post/' + postid)

# Печатаем запись целиком
@get('</post/<postid>»)
def showPost(postid):
    client = MongoClient('localhost', 27017)
    db = client.LXF
    query = {'_id': ObjectId(postid)}
    post = db.blogposts.find_one(query)
    return template('showsinglepost', post=post)
```

Как видите, в файле **blog.py** определяются различные обработчики URL. Шаблон **menu.tpl** отображает основную страницу сайта блога. Шаблон **listPosts.tpl** выводит краткий список всех записей блога и позволяет выбрать и просмотреть одну из них. Шаблон **write.tpl** позволяет создать новую запись в блоге с помощью формы — этот шаблон можно как угодно изменять. Представление **showsinglepost.tpl** появляется автоматически после создания новой записи с помощью **write.tpl**. Как несложно догадаться, **showsinglepost.tpl** обычно используется для отображения одной записи блога.

Прежде чем продолжить с Bottle, стоит немного поговорить о схеме базы данных *MongoDB*, где будут храниться данные. Как вы, возможно, знаете, *MongoDB* схемы не имеет. Это означает, что два документа, принадлежащие к одной коллекции, могут иметь совершенно разное количество ключей, за исключением ключа **\_id**, который является обязательным.

При написании кода для *MongoDB* это очень важно, так как если неправильно указать имя коллекции в коде, то вы не получите сообщения об ошибке — вместо этого создастся новая коллекция! То же произойдет, если неправильно указать имя ключа документа.

### Создание записей блога

Как обычно, для создания новых записей блога понадобится шаблон. Файл шаблона Bottle (**write.tpl**) для создания новых записей выглядит так:

```
<!DOCTYPE html>
<html><head>
<title>Записываем в блог.</title>
</head>
<body>
<h2>Если вы не хотите создать новую запись, выберите из следующего:</h2>
<ul>
```

```
<li><a href="/">home Page</a> </li>
<li><a href="/list">Показать все записи</a></li>
<li><a href="/login">User login (not implemented)</a></li>
<li><a href="/newuser">User Signup (not implemented)</a></li>
</ul>
<form action="/presentnewpost" method="POST">
<h2>Subject:</h2>
<input type="text" name="subject" size="120"
value="{{subject}}"><br>
<h2>Blog Text:</h2>
<textarea name="body" cols="120" rows="20">{{body}}</
textarea><br><p>
<input type="submit" value="Submit">
</body></html>
```

Каждая запись блога сохраняется в базу данных *MongoDB*. После нажатия кнопки Submit [Отправить] ваши данные автоматически передаются на URL **/presentnewpost**, в котором сохраняется запись блога. Наконец, вы перенаправляетесь на URL **/post/postid** для просмотра записи. Идентификатор каждой записи (postid) — это строковое значение ее ключа **\_id**, которое является уникальным для этой коллекции. Шаблон Bottle для получения всех доступных записей блога выглядит так:

```
<!DOCTYPE html>
<html><head>
<title>List Blog Posts Page!</title>
</head>
<body>
<h1>Эта страница показывает существующие записи блога.</
h1>
<ul>
%for myX in data:
<%
link = str(myX['_id'])
%>
<li><a href="/post/{{link}}">{{myX['title']}}</a></li>
%end
</ul>
</body></html>
```

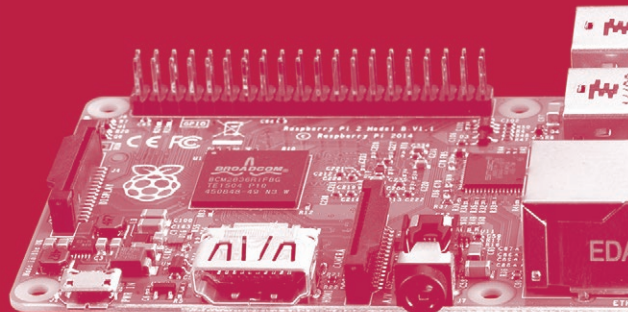
Следующий прием позволит выполнить код Python в файле шаблона Bottle:

```
<%
link = str(myX['_id'])
%>
```

Этот код используется для создания правильной ссылки на каждую запись блога. Было бы интересно посмотреть, как каждая запись блога хранится в коллекциях *MongoDB*. Так как *MongoDB* не имеет схемы, новые ключи в коллекции **blogposts** легко добавив в любое время и без простоя. **LXF**

# LINUX FORMAT Пользователям

# Pi



Ваша порция смачных новостей, обзоров и учебников от Raspberry Pi

**ПИТ ЛОМАС**  
Со-основатель  
и попечитель  
Raspberry Pi  
Foundation



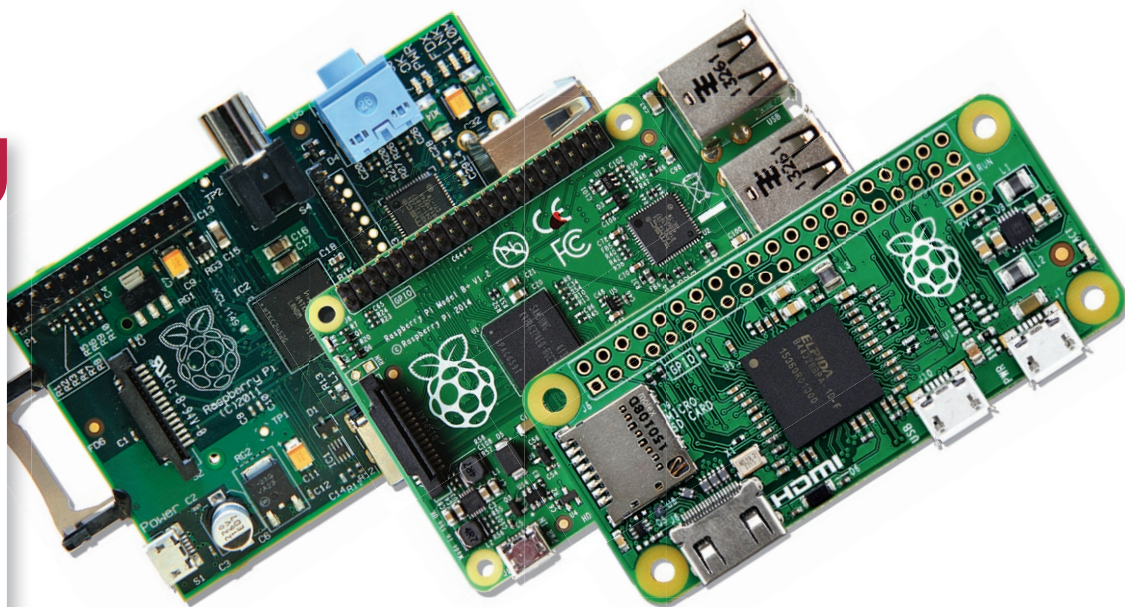
## Привет...

**R**aspberry Pi задумывался ради улучшения навыков «низкоуровневых» вычислений, во многом утраченных после того, как IBM PC покончила с эпохой домашнего программирования (как ее тогда называли). Как дешевая аппаратная платформа, совместимая с Linux, Pi отвечал этой цели. Но наше решение добавить к нему 26-контактный GPIO-выход воспало сообщество самоделкиных. Отныне они могли создавать нужное им оборудование, свободно публиковать и поддерживать свои продукты, чтобы другие также могли использовать их и настраивать под себя.

Благодаря улучшенному и расширенному разьему Pi-Hat/GPIO уже появились сотни других плат с готовыми программными модулями и богатым выбором компонентов для любительских и коммерческих проектов. С Pi можно от управления битами на порте ввода-вывода дойти до искусственного интеллекта и дальше. Подключаемость Pi открыла море опций для удаленного контроля и наблюдения: от скворечников до сбора данных в космосе!

Набирая новых людей, я всегда интересуюсь их хобби. Оригинальные изобретения демонстрируют навыки и способности лучше, чем оценки на бумаге. Компьютерное мышление, требуемое, чтобы разделить проект на достижимые цели и исследовательские решения, а затем реализовать их и собрать воедино — это базовый навык, полезный во многих аспектах нашей жизни.

А учатся ли создатели, работая над большим проектом? В моем представлении — несомненно: разве это не львиная доля удовольствия?



# Pi — ровно 4!

Как Raspberry Pi и Pi Foundation удалось завоевать мир? Вспомним, что мы писали об этом раньше, и узнаем, куда Pi движется теперь.

**R**aspberry Pi, прошедший путь от проекта, набросанного на салфетке, до популярного во всем мире устройства, стал феноменом. Теперь, когда число его продаж перевалило за 8 миллионов в год, он стал не просто быстрее всего продаваемым ПК за всю историю, но и оспорил у Amstrad PCW титул самого продаваемого британского ПК. Но как он появился? Как пришел к такому успеху?

(Подсказка: ему помог один симпатичный пингвинчик) Кто привел Raspberry Pi туда, где он сейчас находится?

Кажется, всё это промчалось за мгновение ока, и теперь, когда Raspberry Pi исполнилось 4 года, мы вновь кратко напомним его историю, подробно рассказанную Linux Format в основном от лица его

зачинателей. Хотелось бы подчеркнуть, что успех Pi легко принять как должное. Но если бы не дальновидность его авторов, не щедрость Foundation; не постоянное развитие ОС GNU/Linux и не безграничный энтузиазм сообщества, Raspberry Pi никогда бы не добился такого успеха и стал бы еще одной рядовой платой.

Мерилом этого успеха является то, что несмотря на непрекращающуюся конкуренцию со стороны подражателей Pi, несмотря на существование и развитие Arduino; несмотря на то, что Intel и прочие выпускают мини-ПК под x86, а Microsoft пытается поддержать их своей Windows 10, Pi всё равно остается лидером продаж и твердо хранит верность Linux.

**Pi остается лидером продаж и твердо верен Linux.**

цию со стороны подражателей Pi, несмотря на существование и развитие Arduino; несмотря на то, что Intel и прочие выпускают мини-ПК под x86, а Microsoft пытается поддержать их своей Windows 10, Pi всё равно остается лидером продаж и твердо хранит верность Linux.





# Дэвид Брейбен

Брейбен, один из создателей *Elite*, верил в успех Pi, поэтому был в проекте с самого начала.

Устройство называется Raspberry Pi. Устройство? Скорее, компьютер. «Устройство» обычно имеет какую-то одну функцию или назначение, а здесь все совершенно не так. Очевидный козырь — на нем можно запустить *Quake III Arena*, «но, я думаю, справедливо будет сказать, что это далеко не все его способности», — считает Дэвид Брейбен. — Он поладит и с вещами, куда более современными, причем в свободном доступе, которые мы сможем легко запустить». На самом деле, суть Pi не в этом. Да, он работает на Linux и загружается в привычный нам LXDE, который в более дешевую модель, со 128 МБ ОЗУ (по сравнению с 256 МБ), вписывается в обрез; но со временем появится аппаратное ускорение, и все будет очень гладко. И кстати, к Pi присматривается лидер в сфере открытых программ для работы с медиа, XBMC, и Pi умеет декодировать видео с разрешением 1080 пикселей благодаря графическому процессору Broadcom. И так, это довольно мощная штука, при этом меньше про размеру (и почти что дешевле), чем стандартный USB-хаб.

Но не стоит низводить Pi к рядовому самопалу. «Дело не в этом, — говорит Брейбен. — Между [использующими контент пользователя] *Halo Forge*, *Rollercoaster Tycoon* или *LittleBigPlanet*, и лидерами, вроде *XNA*, где нужно быть докой, чтобы принять участие, огромная пропасть. Для меня эту пропасть преодолел BBC Micro. На самом деле, нижней границы тогда даже не существовало, но сейчас она свидетельствует о том, что интерес к изучению программирования „по принципу Lego“ растет».

Компактный и дешевый, Raspberry Pi легко проложит дорогу в портфель каждого школьника в этой стране и вне ее, и потенциально может стать

конструкторским инструментом, годным для любого вида учебной деятельности. Можно научить детей создавать виртуальные объекты, совмещая учебный план и свои задумки, а в процессе — лишить ореола мистики код, скрытый в вещах, которыми они пользуются каждый день. Игр, гаджетах, телефонах и прочих повседневных машинах: как говорит Брейбен, все они обманчиво просты, надо только заглянуть под внешний интерфейс.

В плане которого, кстати, Pi еще многое предстоит поменять. Версия, появление которой намечено на декабрь [2012 г.], это плата для разработки, без корпуса и предустановленного ПО. Хотя со временем предполагается, что на ней будет загружаться что-то более для нас всех привычное.

«У нас есть одна штука от MIT, называется Scratch, мы ее недавно показывали в новостях. Такая, знаете... взглянешь на нее и скажешь: „Да я бы и сам мог такую сделать“. Персональный компьютер пугающе сложен, и его легко сломать, ну или, по крайней мере, есть такие страхи — что он больше в жизни своей не загрузится. И не совсем беспочвенные: удаляется не тот DLL, и вам будет трудно оживить свой ПК, не поняв, что натворил ваш ребенок. Школы изрядно с этим мучаются».

В Raspberry Pi нет дисковой памяти, только один слот для SD-карты, содержащей все, что нужно для его загрузки и работы. Хотя Брейбен (с улыбкой Чеширского Кота) заявляет, что BBC BASIC запускается прямо из основного кэша процессора со скоростью Ассемблера, важно знать, что грохнутая машинка восстанавливается всего лишь заменой SD-карты. Есть серьезные



► Брейбен в ноябре 2011 г. — EDGE #235.

## ON GETTING THE WORD OUT В последнем выпуске Newslight мы показали Scratch.

намерения добавить BASIC, «однако не исключено, что с этими волшебными тремя буквами будет проблема».

Избегая упоминать модное слово «краудсорсинг», Брейбен, тем не менее, признает, что ключевую роль в составлении библиотеки программ Raspberry Pi будет играть сообщество. Помимо 50, находящихся в открытом доступе, еще 10000 можно приобрести на сайте, от самого широкого круга лиц: от образовательных организаций и инженерных компаний и до тех, кто просто сам, из лучших побуждений, создал отличное ПО». Мы надеемся, что сообщество будет развиваться вместе с платой — и наступят времена, когда, как в юности Брейбена, под компьютерами понимались программы, а под программами — возможности. ►►

## Хронология Pi — когда какой Pi был готов к употреблению

### Пре-Pi

Меловой период (2006 г.)

У Эбена Аптона [Eben Upton] родился идея создать набор для сборки простого домашнего ПК. На основе системы Atmel ATmega644, размещенной на Veroboard с блоком SRAM-памяти. При частоте 22,1 МГц он должен был поддерживать еще и дисплей, оставляя время гашения для обработки. Аптон решил, что лучше уж чип (SoC), способный потянуть нормальную ОС, и молодец! <http://bit.ly/RaspPi2006Ed>

### Прото-Pi — Май 2011 г.

В начале 2011 г. Дэвид "Elite" Брейбен демонстрирует прототип USB-устройства, который он называет Raspberry Pi (<http://bit.ly/EarlyPiOnBBC>). От этой модели, похожей скорее на флешку, чем на нынешний Pi, отказались в пользу устройства с большим числом разъемов, в том числе с жизненно значимыми контактами GPIO.



### Alpha Pi — Август 2011 г.

Плата Alpha была запущена в производство и готова для тестирования в августе 2011 г. Ее употребляли как демо возможностей финальной сборки Raspberry Pi. Способная, как и раньше, запускать Debian и ускоренную Quake 3, она была немного крупнее итогового дизайна. Публике ее представили на (<http://bit.ly/PiDemo2011>) Transfer Summit в сентябре 2011 г.

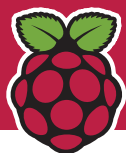


### Beta Pi — Декабрь 2011 г.

Beta PCB была спроектирована к ноябрю, и готовые платы появились во вдохновенном Pi HQ прямо перед Рождеством 2011 г. Вскоре обнаружилась проблема в ходе производства, потребовавшая замены компонента. Из-за этого финальный релиз несколько задержался.







## Пит Ломас

Пит Ломас спроектировал первую плату Raspberry Pi, и Нейл Мор беседует с ним об этой истории.

### Linux Format: Как вы пришли в проект?

**Пит Ломас:** Я разработал большой комплекс оборудования FPGA для Imperial College и отправился на день открытых дверей, чтобы посмотреть, как все работает. И уселся рядом с неким Аланом Майкромфтом [Alan Mycroft], оказавшимся профессором компьютерных наук из Кембриджа.

Он рассказал, что в Кембридже есть такой Эбен Аптон, который задумал создать небольшую макетную плату для обучения программированию. Я подумал, что это прекрасная идея, и отправился с ним поболтать. Всего через полчаса я сказал, что готов ее спроектировать: оборудование у меня есть, пришлите мне чипы, скажите, что плата должна уметь, а я ее изготовлю на своей производственной площадке.

Нас с Эбеном прозвали Билл и Бен [персонажи детского стишка: Bill and Ben are funny men, — прим. ред.]. Эбен взял на себя большую часть работы с BBC, так как он в командировке от Broadcom, и мы им очень благодарны: не поддержи они нас в самом начале, позволил Эбену включиться в проект, ничего бы не было... моя-то компания меня отпустила, но я же ее владелец.

### LXF: Это был ожидаемый успех?

**ПЛ:** Как на голову свалился. Сначала мы планировали изготовить две-три тысячи [плат Pi] за три года, и меня угораздило ляпнуть: «У меня есть свое производство, присылайте чипы. Я спроектирую

и сделаю платформу сам». С этого все и началось — мол, я сделаю это для вас, уж больно идея хорошая. Рори Селлан-Джонс [Rory Cellan-Jones] обратился к Дэвиду Брейбену [David Braben, один из создателей Elite], тоже попечителю, и сказал: «Мы тут кое-что придумали, не хотите глянуть?» Это была ранняя версия Pi, но уже с полноценным графическим процессором, и они сделали ролик для YouTube. Когда за несколько недель ролик набрал 600 000 просмотров, мы поняли, что чего-то да достигли. А потом вдруг как прорвало, и нам стали заказывать партии более чем по 200 штук.

### LXF: Кажется, Pi появился вместе с шокирующей новостью, что в английских школах не учат программированию.

**ПЛ:** В этом процессе важно вот что. Если у вас есть цель, суть не в том, как это запрограммировать, а в том, чтобы ее достигнуть. Таким образом, все это программирование впитывается осмотически, через жажду познания.

### LXF: То есть вы не велите изучать код, а обозначаете конечный результат?

**ПЛ:** Ну, например, с циклом for: все сидят и спрашивают, зачем он нужен? С чего мне им заниматься? А если у них нечто вроде Sonic Pi, вы им говорите: вот так получается музыка. Чтобы повторить музыкальный фрагмент, нужно сделать так-то, и это называется циклом for, а вот тут задается, сколько раз его повторять. И сразу становится ясно, как работает такой кусок кода и зачем он нужен. Это один из приемов, освоив которые, вы научитесь лучше писать музыку —



Ломас в феврале 2014 г. — LXF181.

и они его осваивают. Дойдя до программирования на Python, они уже будут кое-чем знакомы. А раз знакомы, им это будет легче даваться.

### LXF: Потом вы добавляете Minecraft...

**ПЛ:** Должен сказать, это прекрасная программа. Впервые ее увидев, я подумал — ну игра и игра. Но посидев за ней пару часов вместе с сыном — он показал мне, что там можно делать — я понял, что это штука потрясающая. Возможность полностью отстроить этот цифровой Lego, включая водопроводы, электронику — и сын похвастался, что может изготовить пушку, стреляющую тротилом...

Важно, что [в Pi] можно выйти за пределы экрана, благодаря GPIO. Конечно, до Grand Theft Auto по части графики ему далеко, но зато Pi умеет передвигать что-нибудь на вашем столе. Я бы сказал, что в такой встраиваемой среде незримо присутствует гораздо больше компьютеров, чем заметно миру.

### LXF: Сегодня у нас есть масса удивительных устройств, но их внутренняя работа скрыта.

**ПЛ:** Детям нужны инструменты для развития воображения. Это все равно что учить их писать, научив языку, но не разрешая сочинять свои истории. Просто зубрить грамматику. Разве это не подло?

## ОБ ОБУЧЕНИИ

### Чтобы понять, что не так с кодом, нужно в нем очень хорошо разбираться.

## Хронология Pi — 2012-2016

### Pi Model B

Февраль 2012 г.

В Pi Foundation планировали выпустить две модели, потому предзаказы на более популярную Model B начались в феврале 2012 г. В этой более дорогой версии был Ethernet и дополнительный порт USB. К сожалению, порт Ethernet был неправильно запитан — в первых 10 000 устройств его пришлось заменить — так что из-за ускоренного тестирования на соответствие выпуск пришлось отложить до апреля 2012 г.



Май 2012 г.

Отгружено 20 000 единиц

### Pi Model B версия 2

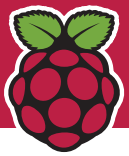
Сентябрь 2012 г.

Производство перенесено в Великобританию, и в связи с этим выходит Raspberry Pi B в версии 2 плюс готовится увеличение памяти до 512 МБ (в октябре 2012 г.). В этой плате устранили некоторые проблемы первоначальной конструкции, связанные с выбором компонентов и монтажных отверстий.

### Pi Model A

Ноябрь 2012 г.

Упрощенная Model A с самого начала предусматривалась в планах Raspberry Pi, но высокий спрос на Model B приводил к тому, что производство платы A вынуждены были отложить до конца 2012 г., и первые экземпляры появились только где-то в декабре. В исходной версии стояла оперативная память на 128 МБ, но в финальной модели ее объем в результате увеличили уже до 256 МБ.



**LXF:** Должен ли Raspberry Pi как-то этому способствовать?

**ПА:** Было бы абсолютно неправомерно сказать учителям: вот вам новые предметы в программу, потрудитесь освоить и внедрить. Откуда сам собой возьмется непрерывный профессиональный рост для этого? Мы обязаны им помочь, ведь они изо всех сил стараются учить детей. На мой взгляд, существование Pi и знание компьютерных наук призваны вдохновить их на творчество. Можно показать Pi в школе и этим их заразить. Но чтобы действительно внедрить Pi в учебную программу, нужно гораздо больше труда, и сейчас мы над этим работаем.

**LXF:** И насколько он туда вписывается?

**ПА:** Без сомнения, Pi это под силу, равно как и какому-то другому проекту. Мы не хотим ничего навязывать — по-моему, это было бы самой фатальной ошибкой. Я бы даже скорее хотел, чтобы учителя посещали бы Raspberry Jam... и наблюдали бы за детьми.

**LXF:** Функция ввода-вывода не требует особо мощного процессора.

**ПА:** Да, но если вы подключаете, например, фотоаппарат, вся обработка данных ведется графическим процессором. Почти вся нагрузка с центрального процессора перекладывается на него. Он создает дампы кадров. Если вам нужно сжатие JPEG, он и это сделает. Если говорить о чипах в плане нагрузки, то вот этот неказистый малыш в углу будет на первом месте, а уж потом — графический процессор и электропитание, но в связке они работают отлично. Однако в целом, там около 100 000 транзисторов.

**LXF:** Именно отсюда возникла идея разместить на устройстве порт ввода-вывода?

**ПА:** Не без этого. Думаю, многое можно было бы выжать и из USB, если изрядно помучиться, но идея была сделать его как можно более низкоуровневым. Чтобы своеобразным эквивалентом программы "Hello World" стала загоревшаяся лампочка, к которой потом можно добавить выключатель. И сделать ее проблесковым маячком.

# Эбен Аптон

В уже далеком июне 2013 г., когда Pi приближался к 2-миллионной отметке продаж, мы взяли интервью у самого «Мистера Pi».

**Linux Format:** Много ли трудностей пришло с успехом Raspberry Pi?

**Эбен Аптон:** Думаю, основную тяжесть логистики и масштабов приняли на себя RS и Element 14 (два изготовителя Raspberry Pi).

**LXF:** А вы заранее на них рассчитывали?

**ЭА:** Да, так и планировалось изначально, ведь еще до старта проекта мы осознавали, что его успех может оказаться нам не под силу, у нас были деньги на создание 10 000 образцов — четверть миллиона долларов... Наверное, этот момент наступил, когда мы выпустили первый образ SD-карты для устройства и получили 50 000 заказов, или глючную альфа-версию первой операционной системы, в жизнеспособности которой сомневались сами разработчики; тогда-то мы и поняли, что дело плохо [смеется] и пора подумать о нашей модели всерьез.

**LXF:** Это было в конце 2011 года?

**ЭА:** Да, и если бы мы тогда остановились на первой модели, то весь прошлый год занимались бы только заказами с того первого дня. И хотя нам кажется, что так и было, на самом деле это заняло месяца 3–4. Но нас бы это точно никуда не привело: в лучшем случае, сделали бы 100 000 за первый год, но никак не миллион.

**LXF:** А ваше личное отношение к проекту изменилось?

**ЭА:** Конечно, в каком-то смысле... Не то чтобы стало менее увлекательно, но гораздо серьезнее.



Эбен Аптон в июне 2013 г. — LXF173.

Ведь есть люди, которые этим зарабатывают. Изначально мы не думали создавать рабочие места, то есть нанимать кого-то помимо изготовителей и распространителей. Но сегодня уже много, порядка сотни, тех, для кого Raspberry Pi — основной источник дохода. А при нынешних »

Сентябрь 2012 г.  
Отгружено 400 000 единиц

Январь 2013 г.  
Отгружен 1 миллион

Апрель 2013 г.  
Выпущена камера Pi

Ноябрь 2013 г.  
Отгружено 2 миллиона

Февраль 2014 г.  
Отгружено 2,5 миллиона

## Pi Compute Module

Апрель 2014 г.  
При имеющемся спросе на Raspberry Pi от компьютерной отрасли появилась необходимость выпустить компактную модель встраивания в стационарные проекты. С Pi Compute Module, Pi перешел в класс плат памяти DIMM и отказался от многих портов.



Май 2014 г.  
Отгружено 3,14 миллиона

## Model B+

Июль 2014 г.  
Первое обновление дизайна. В основе своей Pi остается прежним — та же SoC и ОЗУ — но видоизменен порт GPIO, добавились долгожданные порты USB и упала цена.

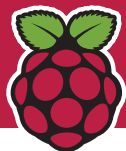


## Model A+

Ноябрь 2014 г.  
Была радикально пересмотрена исходная концепция Model A. Новая Raspberry Pi Model A+ представляет собой компактную, совместимую с Pi плату, с минимальными количеством портов и уровнем потребления памяти — хотя в Pi Zero он сделался еще экономнее.

Февраль 2015 г.  
Отгружено 5 миллионов





объемах, естественно, недельный выход составит 30, 40, а то и 50 тысяч единиц; если нанести урон бизнесу, люди потеряют работу. Так что увлекательно, но и ответственно.

### LXF: Вы задумали Raspberry Pi одновременно и как фонд, и как бизнес?

**ЭА:** Я считаю, что если вы действительно хотите выжить, так и должно быть. Есть мнение, что благотворительностью не обойдешься — вот мы хотели быть благотворительной организацией, и являемся некоммерческой. Просто вырученные деньги вновь идут в дело. Я не получаю зарплату. Мне повезло — я могу себе это позволить.

### LXF: И ваша позиция не изменится?

**ЭА:** Когда-нибудь — возможно. Ведь работаю я, на самом деле, более чем с полной занятостью — больше, чем для многих означает «на полный день» в этой организации.

Но я до сих пор числюсь сотрудником Broadcom. И это огромная щедрость с их стороны. По-моему, в этом всегда видели маркетинговый ход, а я считаю это подкормкой идеи. Нам просто повезло; но пришлось-таки поубеждать Broadcom, что это дело стоящее.

### LXF: Влияет ли это на возможность открытия схемы устройства?

**ЭА:** Схема открыта, кроме самой платы [PCB], и это вопрос интересный. Откроем ли мы когда-нибудь схему PCB? Такое намерение всегда было и остается. Проблема в том, что чипов-то не купить.

### LXF: Вы поэтому не раскрываете плату?

**ЭА:** Если мы ее раскроем, то сделать Raspberry Pi помешают только отсутствие чипов и... нарекания... думаю, мы мало кого обидели, не выдав всё целиком. А все нарекания сейчас валяются

на нас. Теперь я представляю, до чего бы они дошли, если бы мы опубликовали плату. И при той огромной поддержке, оказанной нам Broadcom, я считаю, что они этого не заслуживают. Так что пока я рад, что все эти кирпичи летят только в мой огород.

### LXF: И вы не хотите, чтобы кто-то со стороны создал собственный аналог...

**ЭА:** Именно. В конечном итоге, это привело бы к краху самого фонда, его возможности инвестировать в образование, в открытые проекты без осязаемой выгоды потребителей. Разве что в идейном плане почувствовать, какие мы молодцы: вот дернули за веревочку, и дверка от-

**ЭА:** Мы усиленно работаем над программным обеспечением.

### LXF: И сейчас все силы брошены на это?

**ЭА:** При частоте в 700 МГц [Pi] является необычайно мощным ускорителем медиа. Чип — это на 97% ускоритель.

### LXF: А что будет в установленном по умолчанию Raspbian?

**ЭА:** На данный момент, имеется видеовывод, USB-контроллер; то, что было до появления ARM. ARM сделал чип на 3% больше. Основную нагрузку несет именно ARM, некоторые элементы инфраструктуры, контроллер SDRAM и несколько крохотных периферийных устройств. Значительная часть системы по большей части в спящем режиме. И все равно, 700 МГц для ARM — это очень мощный процессор, хотя по общепринятым стандартам тяжеловатый. Но раз уж здесь

мы повязаны, то делаем больший упор на ПО, чтобы выжать из него максимум. Поэтому много времени было уделено оптимизации системных компонентов, будем надеяться, для общего блага, системных компонентов Linux — того же рixтар. Улучшению таких вещей, как memscoru и memset.

Любопытная была дискуссия по поводу ускорения графики. У нас нет ни ускорителя Иксов, ни драйвера ускорителя. Но есть множество компонентов чипа, множество подсистем, способных влиять на ускоритель X-сервера. И по сути этого достаточно. С ПО для X все хорошо. Я сам удивляюсь — насколько. Просто ARM подстраивает пиксель за пикселем — мы наладдили рixтар, но всё равно это делает ARM: гонит пиксели. Производительность ARM довольно низкая, а пиксели движутся шустро. Вот так мы подискутировали о том, нужно ли нам аппаратное ускорение графики. **LXF**

## О RASPBERRY PI 2

# Будет грустно и даже фатально, если в 2016-м мы будем продавать тот же Pi.

крылась — ну и всё. А потом на Broadcom обрушились бы все эти нарекания за то, что не предоставляют свои чипы.

### LXF: Вы уже обдумываете Raspberry Pi 2?

**ЭА:** Думаю, было бы очень грустно и даже фатально для нас продавать, скажем, в 2016-м всё тот же Raspberry Pi. Видимо, надо что-то сделать, но пока не представляю, что... Реальная проблема в том, что я представляю себе платы, которые мог бы создать, за любую цену от \$25 до \$85... И те новые, которые я мог бы создавать за каждые добавочные \$10. Пока ни одна из них не делается. Но как найти ту, что будет на самом деле хороша, которая добьется такого же признания, как Pi... А козырь Pi — в соотношении цены и качества.

### LXF: Как же вам удастся поддерживать постоянный разгон?

## Хронология Pi — 2012–2016

### Raspberry Pi 2

Февраль 2015 г.

На свой третий день рождения, Foundation выпустил свое самое мощное, на данный момент, устройство, Pi 2. Бесподобный гибрид недорогого мини-ПК и четырехъядерного процессора ARM.

Сентябрь 2015 г.

Отгружено 6,5 миллионов

Ноябрь 2015 г.

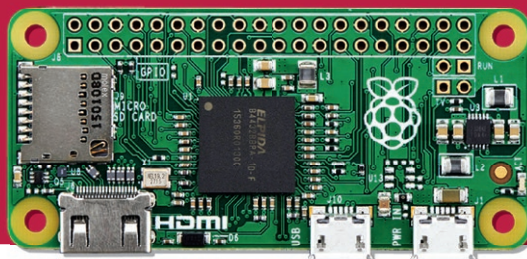
Отгружено 7 миллионов

### Raspberry Pi Zero

Ноябрь 2015 г.

Откуда ни возьмись, появляется Pi Zero, и этот последний Pi становится лидером гонки за миниатюрный

дизайн. Сохранив всю мощь модели B в крошечном формате, с минимальным энергопотреблением и невероятно низкой ценой, Pi Zero задрал планку.



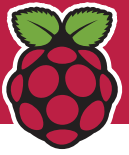
### Raspberry Pi 3

Февраль 2016 г.

Совсем недавно мы размышляли, каким же будет следующий Pi, и вот Pi Foundation его выпустил. Если вы еще этого не сделали, узнайте все детали в разделе новостей на стр. 6! Когда же теперь ждать Pi 4...







# Набор Kano Screen

В погоне за идеальной платформой для своих кодерских экспериментов с Raspberry Pi Лес Паундер нашел портативный экран, влезавший в его рюкзак.

## Вкратце

» Портативный экран для Raspberry Pi, с HD-разрешением, в удобном для детей корпусе. Идея Капо — позволить ученикам «обвесить» свой компьютер — подкрепляется серией учебников, написанных специально для детей.

С новыми аксессуарами для Raspberry Pi — как с автобусами: сначала ждешь не дожدهшься, когда появится какое-нибудь портативное решение, а потом они появляются все и сразу. Мы повидали много разных экранов для Raspberry Pi, включая официальный 7-дюймовый, и вот теперь есть еще Капо.

Эта компания знает толк в упаковке и маркетинге, поэтому у качественного корпуса Kano Screen Kit всё надежно и промаркировано. Также, специально для юных хакеров, в набор включена инструкция по сборке, наподобие таковых для Lego, чтобы можно было открыть коробку и сразу приняться за дело.

Капо имеет диагональ в 10,1 дюйма и выполнен из Gorilla Glass [стекло повышенного сопротивления ударам и царапинам, — прим. пер.] Экран поставляется в виде набора, и пользователю предлагается его собрать. В набор входит прочная пластиковая рамка, в которую вставляется экран и которую можно класть на стол либо поднимать вертикально, как обычный монитор, на манер старой школьной парты. На тыльной стороне экрана расположена контактная плата, через которую он подключается к Raspberry Pi с помощью обычного кабеля HDMI; к плате прилепает коммутационный кабель, подключаемый к плате управления на экране.

Там также есть место для крепления, чтобы можно было объединить ваш Капо и Raspberry Pi в хороший цельный прибор. Питание экрана осуществляется через специальный кабель микро-USB, который заодно питает и ваш Raspberry Pi.



» Этот экран — продуманный баланс между портативностью и функциональностью. С него удобно читать под любым углом, и его можно подключить к Pi через USB-адаптор. (Клавиатура в набор не входит!)

Собирается экран за минуты, единственная сложность — подсоединить к нему плату управления.

## Трудный соперник

Мы протестировали экран на последнем дистрибутиве KanoX и новейшем образе Raspbian. На обоих мониторах ориентация изначально была вертикальной, из-за чего нам пришлось его положить, как старую школьную парту, а затем повернуть изображение на 180° в ОС. В KanoX для этого надо зайти в Настройки системы и выбрать Display. Для Raspbian придется отредактировать файл `config.txt`, расположенный в разделе загрузки, и добавить последней строкой `display_rotate=2`. В обоих дистрибутивах размер рабочего стола был правильным, и ничего настраивать не пришлось, но при необходимости это можно было сделать через кнопки управления сзади экрана.

Экран не имеет ни аудиовыхода (поэтому колонки/наушники придется цеплять через разъем на Raspberry Pi), ни сенсорного ввода. Но качество картинки великолепное: 10,1-дюймовый экран 1280 × 800 дает 150 пикселей на дюйм (PPI). В результате изображение яркое и четкое, и представляет идеальное поле для экспериментов с кодом.

Капо — качественный и прочный продукт, что для его целевой аудитории, то есть

школьников и студентов, в самый раз, так как, в отличие от других экранов, способен выдержать неаккуратное обращение.

В плане цены Капо сейчас — золотая середина: он подороже официального экрана и не имеет сенсорного ввода, но долговечен, что, в сравнении с другими решениями, такими как Pi Top [см. Обзоры, стр. 89 LXF207], делает его хорошей альтернативой для пользователей и школ. Но отсутствие аудиовыхода — это проблема, особенно для тех, кто будет подключать экран к Pi Zero, где нет аудиоразъема. Капо недешев, но в своей ценовой категории — один из самых надежных. LXF



## Свойства навскидку



### Настраиваемый дисплей

Имеет два рабочих положения и удобные кнопки настройки сзади.



### Совместим со всем

Экран Капо имеет панель управления HDMI, а значит, может применяться с другими устройствами.

## LINUX FORMAT Вердикт

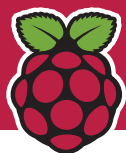
### Kano Screen Kit

Разработчик: Kano  
Сайт: kano.me  
Цена: £ 110

Функциональность	7/10
Производительность	9/10
Удобство в работе	8/10
Справедливость цены	7/10

» Kano Screen надежен и сочетает портативность с удобством, предлагая качественный экран по разумной цене.

## Рейтинг 8/10



# Scratch: Как кодируют списки

Лес Паундер показывает, как изучить важные концепции программирования с помощью Scratch на Pi и затем применить их в Python.



## Наш эксперт

Лес Паундер работает с Raspberry Pi Foundation, проводя тренинги с учителями по всей Великобритании. Лес ведет блог на <http://bigles.com> о работах, взломах и обо всем, что можно сделать самому.



## Скорая помощь

Собирать из блоков Scratch легко, но тем не менее можно создать достаточно сложный код. Если вы немного запутаетесь, просто вытяните раздел кода из основного тела и работайте с ним отдельно. Это простой способ проверки кода. По завершении верните код в основной раздел и снова проверьте на предмет совместимости.

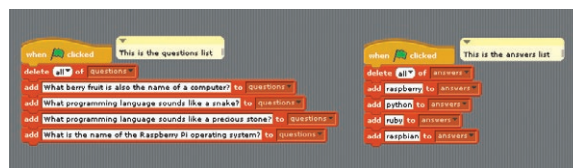
Мы воспринимаем списки как должное, но это фантастически мощные инструменты, ежедневно используемые нами для всего: они напоминают о наших покупках и заметках и вообще о том, что нам надо сделать. В программировании списки используются для сбора данных в последовательность (обычно разделенную запятыми).

На этом уроке мы будем использовать список в двух приложениях: вначале создадим в Scratch простую викторину, где используется два списка для управления вопросами и ответами. Затем будем случайным образом задавать вопросы и применим метод для проверки, что в списке ответов выбран правильный. Если игрок отвечает на вопрос правильно, игра его похвалит, а если нет, случится нечто иное. Так мы введем понятие списка. Во врезке на стр. 95 мы конвертируем викторину в код Python и покажем, как типизированный язык программирования обрабатывает списки.

Для этого проекта достаточно будет любой модели Raspberry Pi плюс самый свежий дистрибутив Raspbian. Весь код можно скачать с <http://bit.ly/LXF209-PiUser-Lists>, так что давайте запустим ваш Pi и войдем в рабочий стол. Мы начнем со Scratch, популярного и свободного языка визуального программирования, который находится в меню Programming, расположенном в левой верхней части рабочего стола Raspbian под главным меню.

Интерфейс Scratch использует ряд блоков, помещенных в левой части экрана, которые перетаскиваются в область команд в центре. Действия и вывод кода отображаются в верхней правой части экрана.

Мы начнем наш код с выбора спрайта Кота, щелчком левой кнопкой по спрайту. Теперь щелкните по палитре Variable [Переменные]: она будет пустой, но на ней есть две кнопки, нажмите на Make a list [Создать список], вызовите список questions [вопросы] и создайте другой список под названием answers [ответы]. Для каждого списка понадобится создать ряд значений; для questions мы создадим четыре вопроса и будем хранить их в списке. Из палитры Control [Контроль] захватите другой блок When Green Flag Clicked [Когда нажат зеленый флаг]. Под этим блоком стоит



Видя списки в виде блоков в области кодирования, можно быстро сравнить их содержимое, и гарантировать, что на вопрос был дан верный ответ.

добавить Delete #1 from questions [Удалить первый пункт из], найденную в палитре Variables. Измените этот блок, чтобы он читался Delete All from questions [Удалить все пункты из questions]. Это будет заново устанавливать вопросы при каждой игре. Под следующим блоком поместим Add thing to questions [Добавить thing к questions], которую также можно найти в палитре Variables. Затем измените thing [зд. нечто], чтобы она формулировала вопрос. Теперь создадим ответы к этим вопросам и сохраним их в списке answers. Процесс идентичен процессу с questions, но будьте внимательны, поскольку вам надо убедиться в том, что вы работаете с должным списком!

Теперь у нас имеется два списка, и в каждом по четыре записи, а первый вопрос в списке questions имеет верный ответ в первой записи списка answers. Чтобы связать верный ответ с вопросом, создадим переменную с именем item# [номер пункта].

## Создание переменной

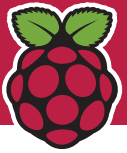
Теперь вернитесь в палитру Control и перетащите блок When Green Flag Clicked [Когда нажат красный флаг] в область кода. Затем захватите блок Repeat [Повторять] 10 и прикрепите его под новым блоком Green Flag [Зеленый флаг]. Нам надо, чтобы код повторялся для всех вопросов, находящихся в нашем списке, а в палитре Variables мы видим length of questions [длина списка questions]. Перетащите этот блок и поместите его над 10 из Repeat 10, 10 должно подсветиться, показывая, что наш блок войдет внутрь, заместите им 10. Внутри повторяющегося цикла мы поместим Set item# to 0 [Установить item# в 0] из палитры Variables. Заменим 0 на блок Pick Random 1 to 10 [Выбрать случайным образом число от 1 до 10], его можно найти в палитре Operators [Операторы], а 10 — на length of questions из палитры Variables. Теперь мы можем менять значение переменной item# на случайное число между 1 и длиной списка.

Все еще внутри цикла Repeat, мы используем блок Ask what's your name? and wait [Спросить ваше имя и ждать] из палитры Sensing [Сенсоры], который прикрепляется под блоком Set item#. Блок Ask нужен, чтобы задать пользователю вопрос, и в него нам надо поместить случайно выбранный вопрос.

Захватите блок Item 1 of questions и перетащите в пространство What's your name? блока Ask. Тогда будет задан конкретный вопрос. Так что давайте захватим блок Item# из Variables и используем его



Как видите, законченный проект выглядит структурно простым и позволит каждому узнать больше о списках в программировании.



## Работа со списками

Как мы уяснили из основного текста, списки — это мощный инструмент в Scratch и Python. Каждый элемент в списке имеет индекс — номер, который указывает на его местоположение. В Scratch и Python можно удалять элементы из списка, но не забывайте, что если вы удалите из списка элемент 2, значения индексов всех последующих элементов уменьшатся на 1, и тогда в викторине, которую мы создали, получится, что ответ на вопрос подсухнет

неверный, а это не слишком-то честно по отношению к игрокам.

Как вы можете ожидать, у Scratch очень упрощенный выбор инструментов для работы со списками, но они являются отличным введением к пониманию понятия списка детьми. Для дальнейшего изучения в Python 3 есть расширенный массив инструментов для работы со списками и другими структурами данных: <https://docs.python.org/3/tutorial/datastructures.html>.

```
File Edit Format Run Options Window Help
import easygui as eg
from random import randint

questions = ["What berry fruit is also the name of a computer?",
            "What programming language sounds like a snake?",
            "What programming language sounds like a precious stone?",
            "What is the name of the Raspberry Pi operating system?"]

answer = ["raspberry", "python", "ruby", "raspbian"]

for question in range(len(questions)):
    item = randint(0,3)
    response = eg.enterbox(msg=questions[item], title="Question")

    if response == answer[item]:
        eg.msgbox(msg="Correct")
    else:
        eg.msgbox(msg="Incorrect")
```

для замены '1'. Отныне у нас есть метод задавания случайного вопроса из списка questions.

Раз вопрос задан, воспользуемся условным оператором `if... Else`, который проверит данный ответ на предмет правильности. Надо написать условие для проверки, так что перетащите `_ = _` из палитры Operators и поместите его в шестиугольник после `if`. Теперь захватите блок Answer из Sensing и перетащите в первое пустое пространство `_ = _`. Блок Answer хранит ответ на только что заданный вопрос. В другом пустом поле нам надо поместить соответствующий ответ на вопрос. Так что перетащите блок Item 1 of answers из палитры Variables и вставьте его в оставшееся пустое поле. Потом замените '1' на блок item# из Variables. Теперь у нас есть готовый тест, который проверит данные ответы на их соответствие нашим ожиданиям. Далее создадим действия, которые будут выполняться при правильном ответе. Для моего проекта я использовал цикл repeat 24 внутри условия if. В цикле repeat 24 я поместил Turn clockwise 15 degrees [Повернуть по часовой стрелке на 15 градусов] из палитры Motion [Движение]. Мой спрайт развернется на 360 градусов (15 градусов — это 360/24). Когда поворот завершится, я проиграю коротенькое аудио, в награду игроку. Для импорта звука нажмите на вкладку Sounds [Звуки] в верхней части поля программирования. Затем импортируйте аудио, нажав Import и выбрав подходящий аудиоклип из диалогового окна. Чтобы использовать аудио, перетащите Play sound <select one> until done [Играть звук <выберите какой> пока не завершится] из палитры Sound и прикрепите его под циклом repeat 24, но все еще внутри условия if.

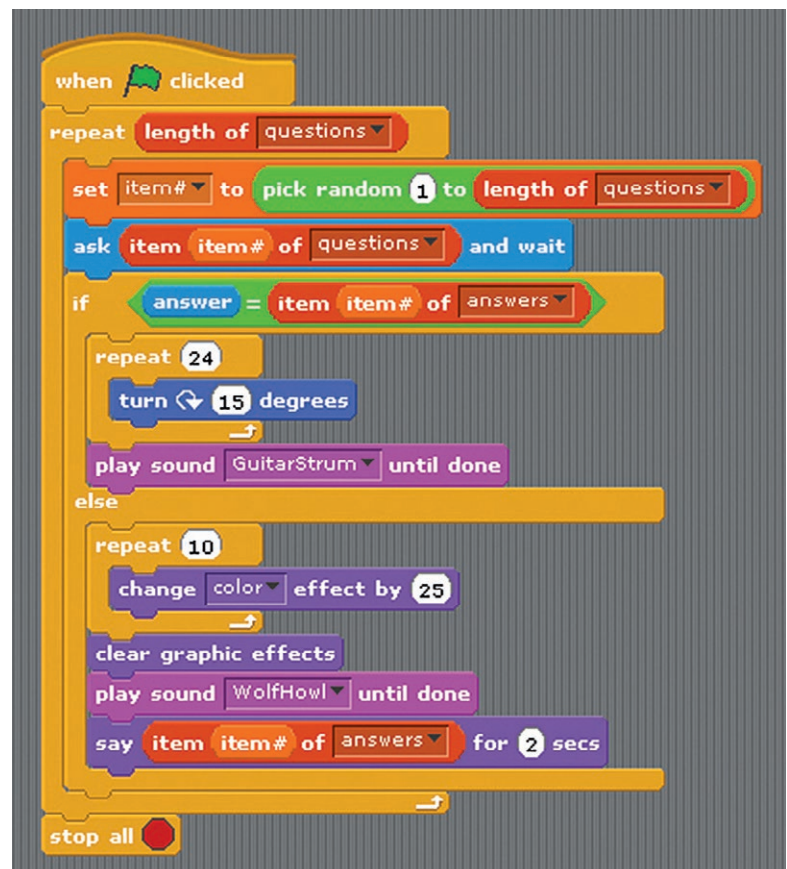
### If и else

Теперь выйдем из условия if и займемся частью else. Нам не нужно условие для проверки противоположного исхода, поскольку у нас есть If the answer is not correct, then else must be the only logical option [Если ответ неверный, else должна быть единственной логической опцией]. Внутри условия else я решил использовать другой цикл repeat 10 и внутри него — блок Change color effect by 25 [Изменить цветовой эффект на 25] из Looks [Вид] для изменения цвета спрайта. Уже вне цикла repeat 10, но все еще внутри условия else, мы перетащим Clear graphic effects [Убрать графические эффекты], что перезапустит наш спрайт в изначальном цвете. Теперь мы воспроизводим другой звук, перетаскивая блок Play sound <select one> until done из палитры Sound. Помните: звук можно менять, импортируя новый файл аудио. Наш последний шаг для условия else использует блок Say Hello! for 2 secs [Говорите Привет! 2 секунды] в палитре Looks. Затем мы захватываем блок Item 1 of answers из Variables и захватываем блок item# также из Variables, этот блок

заменяет 1. Теперь надо создать шаг, который выведет верный ответ на экран, если игрок ошибся.

Вы можете изменить вид вашего спрайта, нажав на Costumes [Костюмы] выше поля кода, и изменить задний план, нажав на Stage [Сцена], находящийся в нижней правой части экрана и нажав на Backgrounds [Фоны] выше поля кода. Я выбрал здание школы, но вы можете поставить на заднем плане любое строение. Мы также добавили блок When Green Flag clicked вместе с бесконечным циклом, играющим фоновую музыку для игры. Опять же, это излишество, но так смешнее.

Итак, код готов. Не забудьте сохранить свою работу. Нажмите на зеленый флаг и приготовьтесь отвечать на вопросы. **LXP**



› Иногда надо класть блоки поверх блоков, что бывает весьма хитро, но Scratch позволяет растащить блоки врозь и повторить попытку, прямо как в Lego.

» Подпишитесь на печатную или электронную версии на [www.linuxformat.ru/subscribe!](http://www.linuxformat.ru/subscribe!)



# ОТВЕТЫ

Есть вопрос по открытому ПО? Пишите нам по адресу [answers@linuxformat.ru](mailto:answers@linuxformat.ru), и мы найдем ответ.

» В этом месяце мы ответим на вопросы про...

- 1 Первую установку Ubuntu
- 2 Проблемы с Midnight Commander
- 3 Игры в Linux с Wine
- 4 Настройку твердотельного накопителя
- 5 Перезагрузку роутера на удаленном сайте
- 6 Linux в Pine A64+

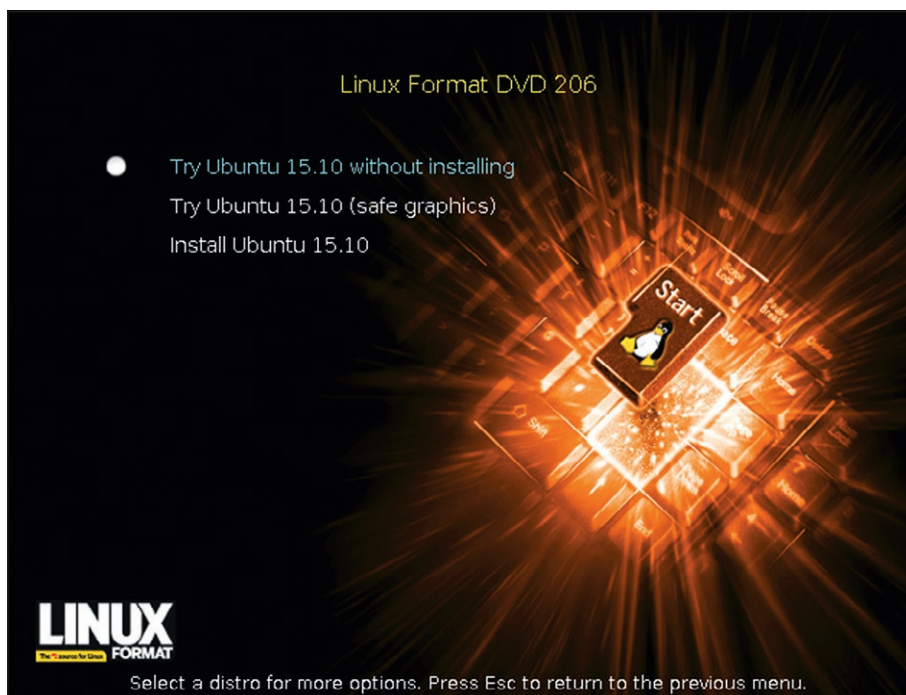
## 1 Первые шаги

В Я не могу понять инструкции по установке Ubuntu 15.10 с диска. Конечно, мне не хватает базовых знаний о компьютерах, не говоря уже о понимании того иностранного языка, на котором все это написано. Мои юные друзья предложили мне перейти на Linux с Microsoft Windows 7, и я бы рад, но они не оставили никаких подсказок. Я открыл файл `index.html`, чтобы разобраться, и был озадачен. Понял ли я хоть что-то? Нет. Вывод: наверное, я тупой. Я пробовал искать помощь в Интернете, у меня есть PDF-файл «Linux для чайников», который я читал снова и снова, и я уже начал сомневаться в своем интеллекте и рассудке. Умоляю, объясните мне все это самыми простыми словами! Не может же это быть настолько трудно. Или может? Какие программы нужно установить, чтобы смотреть DVD-диски, слушать CD-диски, играть в игры и т.д.?

Джим МакРоберт (Jim McRobert)

О Начало работы с новой операционной системой (ОС) похоже на переезд в другую страну: язык, традиции, почти всё — другое. Это даже еще сложнее, потому что в случае с новой ОС ее нужно установить. Так как на большинстве компьютеров Windows установлена заранее, то пользователи Windows обычно не сталкиваются с установкой. Но это не так сложно. Во-первых, чтобы попробовать Linux, не обязательно его устанавливать. У большинства дистрибутивов Linux (дистрибутив — это просто ОС Linux с набором полезных приложений) есть live-режим. Вам нужно загрузить свой компьютер с DVD, для этого удерживайте клавишу, которая открывает загрузочное меню, после включения компьютера. Обычно это Esc или одна из функциональных клавиш, часто F11 или F12, а на ноутбуках — F2. Если в руководстве по Вашему компьютеру не указана точная клавиша, попробуйте все эти клавиши по очереди.

Загрузившись с DVD, Вы увидите меню. Выберите дистрибутив, который хотите попробовать,



» Вы можете попробовать большинство дистрибутивов, загрузившись с DVD и выбрав live-среду. Если дистрибутив вам понравился, используйте опцию Install [Установить], чтобы поселить его на жестком диске своего компьютера.

и выберите пункт «попробовать без установки» (так этот вариант выглядит в Ubuntu; в других дистрибутивах может быть просто “live”). При этом операционная система будет загружена в оперативную память и работать прямо с DVD, и Вы сможете поэкспериментировать с ней. Вы не сможете сохранять файлы и настройки, и система будет работать медленнее, поскольку запускается с DVD-привода — но общее представление Вы получите.

Когда Вы будете готовы установить дистрибутив на свой компьютер, выберите пункт «установить» в меню загрузки или щелкните по иконке «установить» на рабочем столе live-дистрибутива. Запустится установщик дистрибутива, который проведет Вас по разным этапам процесса установки. Вам будет задано несколько простых вопросов и несколько вопросов, ответы на которые важны: например, создание пользователя и выбор надежного пароля. Самый важный вопрос — куда Вы хотите установить Linux? Обычно есть возможность очистить весь диск и выполнить установку на чистый диск (при этом будет удалена Windows и все сохраненные файлы) или установить Linux вместе с Windows. Вам, вероятно, требуется последний вариант — с ним будет изменен размер диска C, будет установлен Linux рядом с Windows и появится меню для выбора ОС при каждой загрузке

компьютера. Перед этим стоит загрузиться в Windows и дефрагментировать диск C, тогда изменение размера диска пройдет проще и быстрее.

Дополнительные программы загружаются из центра программ — Вам не придется скачивать EXE-файлы со случайных сайтов. Однако Вы увидите, что в таких дистрибутивах, как Ubuntu, большинство необходимых программ уже установлено. Удачи в экспериментах с новой ОС; а если Вам понадобится помощь, загляните на наши форумы [www.linuxformat.com](http://www.linuxformat.com).

## 2 Полуночный командир

В Я запустил несколько SME-серверов. На большинстве из них сейчас установлен на SME 9.1 на основе CentOS 6. На них установлен Midnight Commander 4.7.0.2 (MC), в котором, похоже, есть проблема с обработкой VFS. В частности, я пользуюсь MC для извлечения отдельных файлов из резервных копий, которые хранятся в TAR-архивах, упакованных GZIP. В версии 4.7.0.2 у меня возникли проблемы с резервными копиями большого размера. Если скопировать архив на другой компьютер и воспользоваться MC 4.8.15, никаких проблем с этими же файлами нет. Мне интересно, можно ли как-то получить RPM-файл последней версии MC для CentOS. Иначе

## Терминалы и суперпользователи

Мы часто предлагаем в качестве решения проблемы ввести те или иные команды в терминале. Хотя обычно то же самое можно сделать с помощью графических утилит дистрибутива, такие решения будут слишком конкретными (будут зависеть от дистрибутива). Команды в терминале более гибкие и — самое главное — ими можно пользоваться во всех дистрибутивах. Команды настройки системы часто нужно выполнять от имени суперпользователя, называемого также `root`. Существует два основных способа это делать, в зависимости от используемого дистрибутива. Во многих дистрибутивах, особенно в Ubuntu и его производных, перед командой можно написать `sudo` — при этом будет запрошен пароль пользователя, и ему будут предоставлены привилегии `root` только на время выполнения команды. В других дистрибутивах применяется команда `su`, для использования которой требуется ввести пароль `root` и которая предоставляет полный доступ `root` до того момента, пока вы не наберете `logout`. Если в вашем дистрибутиве используется `su`, запустите ее один раз и выполняйте любые заданные команды без предшествующей `sudo`.

**мне придется установить CentOS 6.x, установить средства разработчика и научиться собирать RPM-пакеты самому.**

Пол [Paul]

**О** RPM-пакет более свежей версии *MC* для Red Hat Вы должны найти на [www.rpmfind.net](http://www.rpmfind.net), однако не спешите делать вывод о том, что отсутствие ошибок на втором компьютере вызвано версией *MC*. Между двумя системами могут быть и другие различия, которые на это повлияли. При выборе архива в *MC* он распаковывается во временный каталог, а затем Вам показывают содержимое этого каталога. По умолчанию этот временный каталог находится в `/tmp`. Если в `/tmp` недостаточно места для распаковки всего архива, то попытка открыть этот архив в *MC* завершится неудачей. Поэтому различие могло быть просто в количестве свободного места в `/tmp`. Это удастся проверить, изменив переменную окружения

`TMPDIR` так, чтобы она указывала на какой-то каталог с большим количеством свободного места. Это можно сделать в своем профиле или временно в командной строке при запуске *MC* следующим образом: `$ TMPDIR=/путь_к_большому_каталогу mc`.

Если это не помогло и Вам действительно требуется более новая версия, то учиться собирать пакеты RPM не обязательно. Просто соберите *MC* из исходников на том компьютере, на котором собираетесь его запускать. (Собирать RPM-пакет надо только в том случае, если Вы собираетесь его распространять.) Сначала надо убедиться, что установлены все необходимые утилиты разработки. Для этого выполните команду

```
$ yum groupinstall "Development Tools"
```

или

```
$ yum install gcc gcc-c++ make automake
```

Затем загрузите и распакуйте исходный код *MC*, войдите в созданный каталог и выполните команду

```
$ ./configure && make && make install
```

Эта команда установит *MC* в `/usr/local`, поэтому во избежание конфликтов предыдущую версию нужно удалить с помощью *Yum* (предыдущая версия была установлена в `/usr`).

### 3 Винтажный Wine

**В** Я пытаюсь поиграть в игру *Dawn Of War: Soulstorm* с помощью *Wine*, и игра не запускается. Игра вроде бы установлена правильно. У меня есть запись для нее в каталоге `.wine` и значок на рабочем столе, который указывает на этот каталог. У меня двойная загрузка с Windows 8.1 (ее выбирал не я) и Ubuntu 14.04. Я установил игру на раздел с Windows, и там она работает прекрасно. Иногда у меня возникает (очень короткое) сообщение об ошибке `run-dll-32`. Когда я устанавливаю флажок для более подробного сообщения на экране или в лог-файле, то ничего не вижу. У меня Ubuntu 14.04 (64-битная), ядро 3.13.0-74-generic, KDE 4.13.3 и *Wine* 1.6.2.

Noob\_Computa\_Ninja

**О** Проблема почти наверняка связана с Вашей версией *Wine*. Эта программа развивается очень быстро, и ее последние версии поддерживают больше программ Windows. Базу данных *Wine* можно найти на сайте <https://appdb.winehq.org>, и она сообщает, что игра прекрасно работает с версией 1.9.1, последняя версия для разработчиков — 1.9.2, и даже стабильный релиз имеет версию 1.8 — все это гораздо новее, чем то, что у Вас есть.

В таких дистрибутивах, как Ubuntu, программы обычно оставляют в тех версиях, которые протестированы с релизом дистрибутива, и обновляют их только для исправления ошибок и уязвимостей безопасности. Для получения функциональных обновлений обычно нужно ждать следующей версии дистрибутива. Решение Вашей проблемы — установка новой версии *Wine*. Для этого добавьте персональный архив пакетов *Wine* в свой список программных репозиториях. Запустите терминал и выполните команду `$ sudo add-apt-repository ppa:ubuntu-wine/ppa`.

Последние сборки для разработчиков предоставят репозитории `ppa:wine/wine-builds`. После этого нужно обновить списки пакетов репозиториях и установить свежую версию *Wine*. Это можно сделать в графическом менеджере пакетов, таком как *Synaptic*, но так как у Вас открыт терминал, выполните следующие команды:

```
sudo apt-get update
```

```
sudo apt-get install wine1.9
```

Можно было бы просто загружаться в Windows каждый раз, когда Вы захотите поиграть, но настройка запуска с *Wine* позволит не только играть без перезагрузки, но и произвести полезное сравнение относительной производительности игры в двух средах на одном и том же «железе».

### 4 Настройка SSD

**В** Мне подарили твердотельный диск Samsung емкостью 250 Гб в расчете на то, что я поставлю его в свой настольный компьютер и буду наслаждаться, но я потратил

»



## Коротко про...

### Script

**П**еренаправление оболочки удобно, когда вы хотите перехватить вывод одной команды, а при некоторых навыках можно перехватить и канал ошибок. А если вам надо сохранить вывод нескольких команд или команд, которые запускаются интерактивно, то есть надо видеть сами команды, чтобы давать правильные ответы? Решение — программа под названием *script*. Запустите ее из терминала без аргументов. На экране не появилось ничего, кроме сообщения о том, что программа *script* запущилась. Теперь выполните еще пару команд, и на первый взгляд все будет как обычно.

Но на самом деле *script* открыла подоболочку и отображает и перехватывает все, что вы набираете — и это не только вывод команд, но и то, что вы вводите, и даже приглашения оболочки. Некоторые программы, когда их вывод происходит не в ТТУ (оболочку), начинают вести себя иначе: например, если перенаправить вывод команды в файл, он перестает выделяться цветом. Так как *script* запускает свою оболочку, то вывод команд ничем не отличается от того, что вы видите в оболочке.

Когда закончите, нажмите `Ctrl+d` для выхода из подоболочки, и вы вернетесь в исходную

оболочку, в которой появился файл `typescript`, содержащий весь текст вашего сеанса. Если указать после команды имя файла, *script* воспользуется им вместо `typescript`. Имя команды можно указать с параметром `-c`. *Script* запустит ее и завершится. Также есть и другие параметры, которые можно использовать, например, `--timing`, при котором информация о временных параметрах выводится в отдельный файл. Этот параметр удобно использовать с параметром `scriptreplay`, который выводит информацию `typescript`, при желании — с той же информацией о времени, без повторного запуска команд.

не один день на чтение о людях, которые сжигали свои новые SSD. Мне нужно простое решение.

lowy71

Вокруг SSD есть много путаницы, но работать с ними довольно просто. Хотя они используют флеш-память, о них не стоит думать как о дешевых устройствах на флеш-памяти, например, USB-брелках. В таких устройствах слишком большое число операций записи может вызывать сбои, хотя даже они сегодня работают гораздо лучше. В дисках SSD установлены сложные контроллеры, которые учитывают износ, и это означает, что современный SSD при обычном использовании может пережить традиционный жесткий диск.

При настройке SSD некоторые вещи надо делать определенным образом, но современные программы разбиения дисков позаботятся об этом автоматически. Если у Вас старая версия *fdisk*, Вы можете обнаружить, что разделы выровнены неправильно, и это способно привести к серьезным проблемам с производительностью, но такое бывает только с уж очень старыми версиями. Похожие проблемы возникли с появлением дисков объемом больше 2 ТБ, в которых использовались блоки по 4 КБ. Воспользуйтесь *gdisk*, *parted* или *GParted*, и о выравнивании думать не придется.

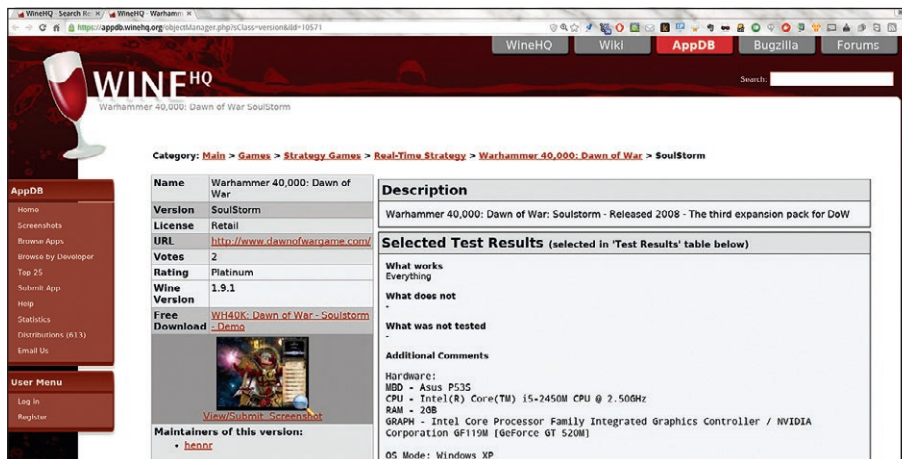
Существует несколько файловых систем, написанных специально для SSD, например, F2FS (Flash-Friendly File System), но ext4 сегодня тоже умеет работать с флеш-дисками. Единственное, что нужно учитывать — применение функции TRIM, которой обладают твердотельные диски. Она очищает блоки данных, содержащие удаленные данные; это своего рода сборщик мусора для диска. Со временем блоки данных могут снижать производительность диска, и TRIM решает эту проблему. Некоторые файловые системы, такие как ext4, позволяют делать это автоматически путем монтирования с параметром `discard` (просто добавьте его в `/etc/fstab`), но это не всегда лучший вариант, так как сама функция TRIM может замедлить диск в ходе своей работы. Другой вариант — запустить команду `fstrim`, которая работает со смонтированными файловыми системами и требует точку монтирования диска в качестве единственного аргумента; при этом также можно указать аргумент `-v` для команды `$fstrim`, чтобы получить более подробный вывод. Можно настроить еженедельный запуск этой команды, создав в каталоге `/etc/cron.weekly` файл со следующим содержанием:

```
#!/bin/sh
/sbin/fstrim /
```

Сделайте файл исполняемым. Если на диске несколько файловых систем, предусмотрите дополнительную команду `fstrim` для каждой из них.

## 5 Стойкий роутер

У меня есть Raspberry Pi на удаленном сайте в другой стране. К сожалению, сайт страдает от частых отключений питания. Когда питание восстанавливается, у моего модема/роутера часто не получается восстановить свое широкополосное подключение — вероятно,



Проверьте базу данных приложений Wine, чтобы узнать, какая версия нужна для вашей программы. Приложению может потребоваться более новая версия, чем та, которую предлагает ваш дистрибутив.

из-за того, что множество других роутеров попытаются одновременно сделать то же самое. Pi остается отключенным от Интернета, часто в течение нескольких месяцев, пока я не приеду туда и не сброшу роутер вручную. Решить проблему мог бы роутер, который периодически пытается подключиться снова, до тех пор, пока у него не получится. Не подскажете ли такое решение?

Дэвид Мэнли [David Manley]

Если роутеру не удалось установить соединение, он бы и сам попробовал установить его снова. Более вероятный сценарий — он считает, что установил соединение, но это соединение не работает. У некоторых модемов и роутеров есть опция регулярной перезагрузки. Если есть какое-то время дня, в которое он Вам вряд ли понадобится, можно перезагружать его каждый день.

Другой вариант — найти модем или роутер с ОС DD-WRT. Эта ОС доступна для широкого диапазона роутеров, но не для модемов, если разработчик сам не установил ее. Также можно найти модем или роутер, который разрешает доступ по SSH наряду с обычным web-интерфейсом. Это вполне возможно, поскольку многие современные роутеры работают на встроенном Linux. Затем можно запустить скрипт *Cron*, который проверяет подключение к Интернету, и если его нет, перезагружает роутер. Когда я столкнулся с похожей проблемой, при которой соединение вдруг переставало работать и приходилось перезагружать модем, чтобы оно начало работать снова, я использовал такой скрипт:

```
#!/bin/sh
HOSTS="mail.isp.com www.google.com www.yahoo.com"
reboot() {
  ssh user@router/sbin/reboot
}
for i in {1..5}; do
  for HOST in ${HOSTS}; do
    ping -c 1 -q ${HOST} 2>&1 >/dev/null && exit
  done
  echo "Все хосты недоступны на дату $(date)"
  [[ i -lt 5 ]] && sleep 3m
done
```

```
echo "All hosts failed"
reboot
```

Скрипт принимает список хостов, с которыми нужно связаться (первым укажите своего провайдера). Затем он пытается связаться с ними по очереди и завершается при первом успешном подключении. Если ни один из хостов не доступен, скрипт ждет и пробует снова. Если скрипт не может подключиться ни к одному из хостов в течение 15 минут, он предполагает худшее и перезагружает роутер.

Вам требуется настроить подключение по SSH без пароля для роутера, для чего следует скопировать открытый ключ SSH в web-интерфейс роутера или воспользоваться аппаратным выключателем, таким как USB Net Power 8800. Он выглядит как короткий удлинитель с USB-портом, с помощью которого им можно управлять. Вы подключаете модем в разъем питания, затем подключаете удлинитель к электросети и затем подключаете Pi к порту USB. Затем скачайте скрипт Python (<http://bit.ly/USBNetPower8800>) для управления устройством. В приведенном выше скрипте замените `ssh` на

```
/usr/local/bin/usbnetpower8800.py reboot 45
```

45 — это задержка между выключением и повторным включением питания. Прибор по умолчанию выключается при включении питания, поэтому нужно отправить команду включения из скрипта в `/etc/rc.local`. Если установить здесь короткую задержку, чтобы Вы не пытались подключаться одновременно со всеми остальными, то проблемы можно избежать, например, так:

```
#!/bin/sh
sleep 5m
/usr/local/bin/usbnetpower8800.py on
```

## 6 Pine A64

Недавно я купил мини-компьютер Pine A64+ на Kickstarter. Я новичок в одноплатных компьютерах и хочу получить ваш совет по программам, которые надо использовать. Какая программа, на ваш взгляд, будет похожа на Windows 10, и какие менеджеры изображений будут совместимы с Pine A64? Мне попала статья о менеджерах изображений в LXF206 и понравился первый вариант, описанный в этой



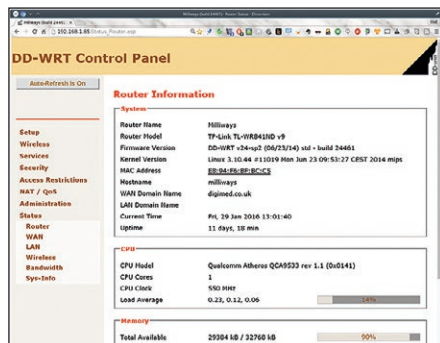
статье — *Digikam*. Можно ли запустить его на Pine A64?

Пол Тиррелл (Paul Tyrrell)

Хотя платы Pine кажутся интересной альтернативой Raspberry Pi, они все еще очень новые. На момент написания статьи они еще не поступили в широкую продажу и доступны только на Kickstarter. Поэтому они все еще находятся на этапе «сделай сам», и на данный момент для них нет настольного дистрибутива Linux — Вам придется собрать его самостоятельно. Пока доступна только сборка Android Lollipop, которую можно загрузить с <http://wiki.pine64.org>. Это ZIP-файл, содержащий IMG-файл. Его нужно записать на карту microSD с помощью dd в Linux. Эта сборка имеет интерфейс, знакомый владельцам телефонов или планшетов с Android.

Разработчики трудятся над релизом Ubuntu, который даст Вам более привычное окружение рабочего стола. На момент написания статьи он еще не доступен, но, возможно, уже будет доступен, когда Вы будете читать эту статью. Когда релиз выйдет в доступ, большинство программ смогут собрать свои пакеты для него (по крайней мере, в теории). Ограничивающие факторы здесь — количество разработчиков, желающих потратить свое время на создание пакетов для этой платы, и, что более важно, системные ресурсы.

*Digikam* использует среду рабочего стола KDE, хотя программу можно запустить и в Ubuntu.



➤ **Запуск DD-WRT (или OpenWRT) на роутере дает гораздо больше возможностей управления по сравнению с большинством прошивок по умолчанию.**

Однако KDE довольно требователен к оперативной памяти, а обработка изображений с большим разрешением тоже требует большого объема памяти, поэтому Ваш план может быть довольно амбициозным даже для 2-ГБ версии Pine64. Я не хочу сказать, что это невозможно, но процессору придется попрыгать.

Такие платы, как Pine и Pi, отлично подходят для более легких приложений, и судя по аппаратной начинке Pine, из нее может получиться прекрасный центр мультимедиа, но еще нужно время, чтобы плата получила признание и привлекла достаточное количество пользователей и разработчиков для создания такой среды. А пока на Pine

## Помогите нам помочь вам

Ежемесячно к нам поступает несколько писем, на которые мы не в состоянии ответить, поскольку проблема описана в них с недостаточной полнотой. Чтобы дать вам наилучший ответ, нам необходимо получить как можно больше информации о проблеме.

Если у вас появляется сообщение об ошибке, приведите его текст в точности и опишите конкретные условия, когда оно появляется. При возникновении проблемы с устройствами перечислите нам все устройства, которые у вас установлены.

Если Linux в вашей системе запущен, вы сможете применить для этого превосходную программу *Hardinfo* (<http://sourceforge.net/hardinfo.berlios>) — она сохранит подробную информацию об устройствах и о состоянии системы в HTML-файле, который вы сможете приложить к своему письму, отправляемому нам.

Не уступающий в удобстве альтернативный вариант — *lshw* (<http://ezix.org/project/wiki/HardwareLiSter>). Одна из указанных программ непременно должна быть включена в ваш дистрибутив (а иногда и обе).

Если у вас нет желания или возможности их установить, выполните в терминале от имени root приведенные ниже команды и приложите сгенерированный ими файл **system.txt** к письму. Это окажет неоценимую помощь в диагностике вашей проблемы.

```
uname -a > system.txt
lspci >> system.txt
lspci -vv >> system.txt
```

есть Android, и это хорошая платформа для изучения внутренних (низкоуровневых) аспектов вычислений. **LXF**



## Часто задаваемые вопросы

# Установка программ

➤ **Я новичок в Linux, где найти программы для этой ОС?**

Linux во многом не похож на Windows, и больше всего — в управлении программами. В дистрибутивах Linux есть централизованный менеджер ПО.

➤ **Как он работает и где его найти?**

В Linux используется система репозитория, содержащих все программы, и управляющая программа для установки, обновления и удаления программ. В большинстве дистрибутивов есть нечто под названием «Центр программ» или что-нибудь похожее, где всё и происходит.

➤ **Итак, в дистрибутивах хранятся собственные копии всех программ. Это немного неэффективно, не так ли?**

Каждый дистрибутив переупаковывает программы, чтобы убедиться,

что они будут работать с остальными пакетами дистрибутива, и избежать конфликтов программ. Все пакеты также снабжаются цифровой подписью, и если в какой-то пакет будет добавлено вредоносное ПО, он не будет установлен. То, что всё хранится в одном месте, также позволяет разрешить все зависимости.

➤ **Что такое зависимости?**

Когда для запуска программы нужна какая-то другая программа, обычно библиотека, это называется зависимостью. Менеджеры ПО позаботятся об этом автоматически и установят зависимости при установке программ.

➤ **Как насчет обновлений? Надо ли по-прежнему проверять их наличие на сайтах?**

Нет, те, кто поддерживает дистрибутив, делают это за вас. Поэтому

программам тоже нет нужды «звонить домой». Часть менеджера пакетов будет периодически загружать свежий список всех пакетов и оповещать вас обо всех обновлениях.

➤ **Я проверил сайт, и там есть новая версия; почему мне ее не предлагают?**

Помните, что дистрибутив следит за тем, чтобы все программы работали с другими пакетами? Это означает, что в некоторых дистрибутивах все обновления откладываются до следующего выпуска дистрибутива. Однако любые обновления, связанные с безопасностью или исправлением серьезных ошибок, становятся доступными сразу же.

➤ **А если мне нужна свежая версия?**

Некоторые программы сами предоставляют пакеты, поэтому

вы можете сами скачать последнюю версию. Менеджеры пакетов не ограничены репозиториями, которые предлагает дистрибутив, в них можно добавить дополнительные репозитории, через которые сторонние проекты обычно и предоставляют собственные пакеты. Чтобы узнать, как это сделать, зайдите на сайт проекта. Точные действия зависят от дистрибутива, но в дистрибутивах на основе Debian/Ubuntu дополнительные репозитории можно добавить как персональные архивы пакетов (PPA).

➤ **То есть все равно придется проверять пакеты?**

Нет. Если вы добавили внешний репозиторий, то менеджер пакетов позаботится о нем так же, как об остальных, и будет следить за обновлениями пакетов и разрешать зависимости.



# LXF Hot Picks



**Александр Толстой**

раскладывает перед вами еще один прилавок вкуснейших свободных и открытых приложений, с утра выуженных из глубин интернет-океана.

MyPaint » Moksha » Knemo » WebcamStudio » KeePassX » FET » Opus » Widelands » Duckhunt-JS » Retext » ABCDE

## Программа для рисования

# MyPaint

Версия: 1.1.2 Сайт: <http://mypaint.org>

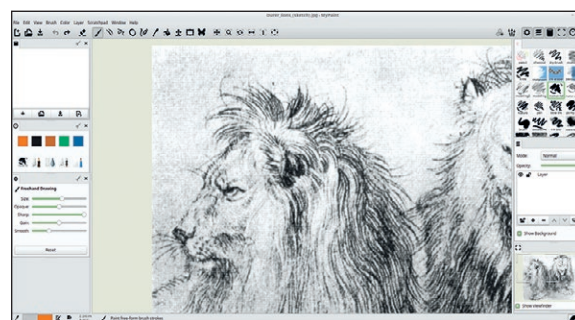
Деятели цифрового искусства имеют тенденцию собираться и обсуждать работы друг друга в web-сообществах, таких как DeviantArt и Behance. Там вы можете увидеть, какими программами пользуются мастера, и попробовать их сами. Очень неплохая идея — начать создавать свои любительские произведения, основываясь на опыте художников, которыми вы восхищаетесь. В мире Windows большинство художников используют *Corel Painter* и *Artage*, а любители Linux отжигают с *Krita*.

И при чем тут *MyPaint*?

Эта небольшая программка приобретает большое значение, если вам надо делать

чертежи и эскизы в Linux и требуется быстрое приложение на *GTK*.

*MyPaint* предлагает меньше хитроумных функций, чем *Krita*, но явно лучше подходит для тех, кто просто хочет порисовать и не требует слишком многого. Одна из самых ценных функций *MyPaint* — «бесконечный» холст, который можно перетаскивать, щелкнув по нему левой кнопкой и одновременно удерживая клавишу пробела,



Возможно, *MyPaint* и похож на *GIMP*, но программа рисования предлагает специальные художественные функции.

## Одна из ценных функций MyPaint — «бесконечный» холст.

а также уменьшать и увеличивать его масштаб с помощью колеса прокрутки.

С обеих сторон экрана имеются всплывающие панели, и их можно включать и выключать с помощью соответствующих кнопок на главной панели инструментов. Эти панели очень напоминают применяемые в *GIMP* (такие как слои), но в *MyPaint* куда больше внимания уделено художественным инструментам. Здесь есть наборы кистей, имитирующих карандаш, мелки, тушь, уголь, и ряд кистей с комбинированными эффектами. Для каждого инструмента рисования можно выбрать размер кисти, давление, тонкость и прочие настраиваемые параметры. Работа с *MyPaint* принесет еще более весомые результаты, если вместо мыши взять планшет или иное подобное устройство ввода. *MyPaint* автоматически определяет такие устройства и позволяет настроить их поведение в разделе Preferences [Предпочтения] приложения (ищите вкладку Devices [Устройства]).

Если раньше вы уже пользовались *MyPaint*, вы, безусловно, заметите положительные изменения по сравнению с прошлым релизом, который был три года назад. Появилась удобная панель истории кисти; все панели привязаны и поддерживают вкладки; поддерживаются новые векторные слои; в коллекцию добавлены новые кисти — короче, найдется, чем мотивировать энтузиастов графики Linux.

## Исследуем интерфейс MyPaint

### Значки панели инструментов

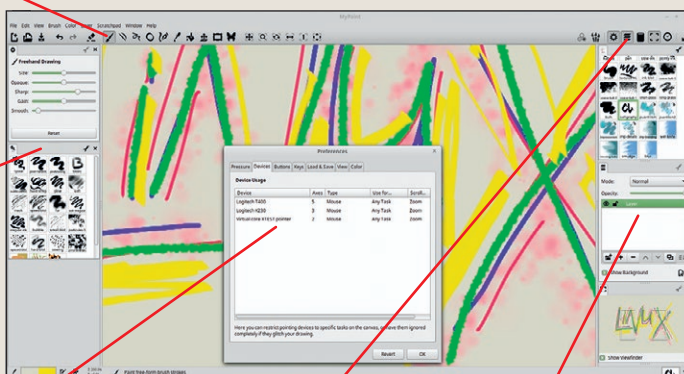
Теперь они монохромные и куда меньше отвлекают внимание.

### Левая боковая панель

По умолчанию здесь расположены опции свободного рисования и несколько кистей, но вы можете привязать сюда всё, что хотите.

### Предпочтения

Можно сообщить каждому устройству ввода, что именно ему следует делать в *MyPaint*.



### Триггеры панели

Управляющие панелями кнопки визуально сгруппированы в правой стороне панели инструментов.

### Слои

Некоторые панели выглядят так, словно их перенесли сюда из *GIMP*.

Среда рабочего стола

# Moksha

Версия: 0.2.0 Сайт: <http://bit.ly/MokshaDesktop>

**М**oksha — слово из санскрита, которое означает «эмансипацию, освобождение или выход» и отлично соответствует стратегии именования в дистрибутиве Bodhi Linux. Возможно, вы помните, что это наиболее употребительный из дистрибутивов Linux с рабочим столом *Enlightenment* по умолчанию. Точнее, был раньше, поскольку Bodhi сделал ответвление *Enlightenment 17 (E17)* и назвал его *Moksha*.

Джефф Хугланд [Jeff Hoogland], один из разработчиков *E17* и Bodhi, сказал, что причиной ответвления было большое число переделок и регрессий в *E18* и *E19*. И это привело, по его утверждению, к тому, что рабочий стол *Enlightenment* стал менее легковесным, чем раньше, и с ним стало гораздо тяжелее работать на состарившемся оборудовании.

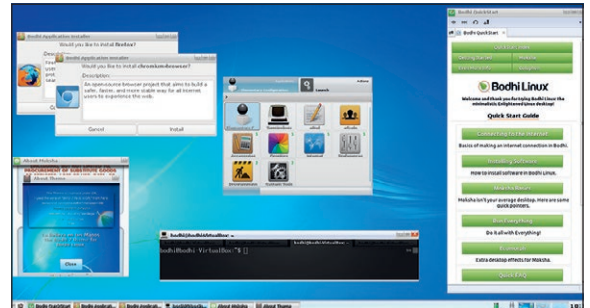
*Enlightenment* разветвился также из-за того, что ряд старых функций (например, основанные на XEmbed значки системного лотка) был удален, точно так же,

как рабочий стол Mate стал ответвлением Gnome 2, а Trinity — KDE 3.

*Moksha* — модульный проект с основным ядром, довольствующимся 300-МГц CPU и 128 МБ ОЗУ, и вспомогательными модулями, которые расширяют функции рабочего стола: например, пользователям рекомендуется установить *moksha-pulsemixer* для задания громкости, местá для управления подключаемыми устройствами хранения и модуль Engage для запуска приложений и переключения между работающими приложениями.

Разработчики *Moksha* планируют и далее полировать кодовую базу, портируя при этом полезные функции из более новых релизов *Enlightenment*. Самый очевидный способ свести знакомство с *Moksha* —

## Bodhi сделал ответвление Enlightenment 17 и назвал его Moksha.



► Если вы пробуете Bodhi Linux, ваш рабочий стол не обязан быть зеленым: для *Moksha* есть десятки тем.

использовать Bodhi Linux, начиная с версии 3.1. Сделав именно так, вы увидите, что дистрибутив быстр и дружелюбен к пользователю, и немало выигрывает от родства с Ubuntu. Можете также взглянуть на краткое описание *Moksha* на сайте <http://www.bodhilinux.com/moksha-desktop>, со ссылками на сборки *Moksha* для Debian, Sabayon и Arch Linux. Конечно, привыкание и к *Moksha*, и к *Enlightenment* потребует некоторого времени, но может оказаться, что такой рабочий стол вы давно уже искали.

Сетевой инструмент

# KNemo

Версия: 0.7.7 Сайт: <http://bit.ly/KNemoMonitor>

**У**многих из нас широкополосное интернет-соединение, а значит, в мониторинге сетевого трафика и в подсчете мегабайт нет особого смысла. Но даже в 2016-м бывают ситуации, когда по-прежнему важны сетевые инструменты для управления локальным соединением. Некоторые пользователи до сих пор живут на предоплаченном тарифном плане с ограниченным трафиком, да есть и желающие контролировать своего провайдера на предмет предоставления положенной скорости. Несколько лет назад, когда доступ в Интернет по телефонной линии был еще в ходу, популярен был инструмент *KPPP*, предоставлявший пользователям Linux большую мощь и полезную обратную связь. Ныне есть более современный инструмент, под названием *KNemo*.

Это небольшая утилита из мира KDE (хотя она неплохо работает на рабочих столах и на *GTK*), которая сидит в системном лотке и сохраняет подробную информацию

о ваших сетевых интерфейсах. По умолчанию значок *KNemo* — это просто картинка пары анимированных эмиттеров, однако можно показать и нечто более удобное, щелкнув по значку правой кнопкой и перейдя в *Configure KNemo... > Interfaces > Icon theme* [Настроить *KNemo... > Интерфейсы > Тема*]. Например, тема *Text* означает, что *KNemo* будет показывать исходящий и входящий уровень трафика прямо в лотке, а более подробная информация с обновлениями *live* доступна во вкладке *Traffic* главного окна *KNemo*.

Еще одна крутая функция — визуализация трафика, позволяющая оценить скорость загрузки или скачивания. Для доступа к ней просто щелкните правой кнопкой

## KNemo показывает скорость трафика прямо в системном лотке.



► *KNemo* отлично визуализирует трафик и показывает, что мы немало получаем через нашу точку беспроводного доступа.

по значку *KNemo* и выберите *Show Traffic Plotter* [Показать кривую трафика]. График обновляется каждую секунду и автоматически подправляет масштаб при наличии всплесков или спадов. *KNemo* вписывается во многих отношениях самую нужную информацию о сетевом соединении в аккуратный интерфейс, позволяя быстро выяснить: свой IP и диапазон подсетей; адрес вещания; шлюз; а также подробности вашего беспроводного соединения (если оно у вас есть). *KNemo* будет хорошим дополнением к любой системе Linux, где по каким-то причинам нет апплета лотка *NetworkManager*. А если он есть, оба инструмента отлично дополняют друг друга.



Live-видеомикшер

# WebcamStudio

Версия: 0.7.3 Сайт: <http://bit.ly/WebcamStudio>

**W**ebcamStudio занимает отдельную нишу среди приложений Linux — представьте, что вы записываете видео со своей web-камеры, микшируете эту съемку с другими видеофайлами, используя плавные переходы, и потом добавляете ко всему этому прекрасную фоновую музыку!

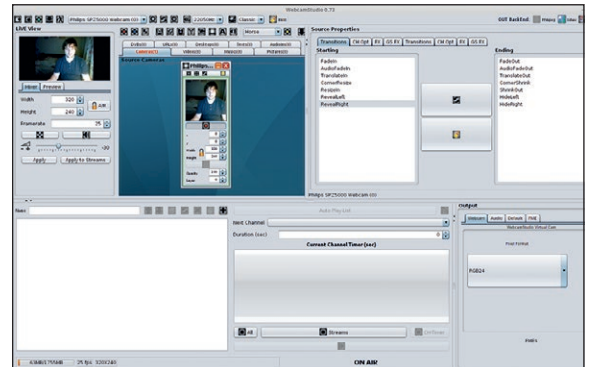
WebcamStudio позаимствовал функцию нелинейного видеоредактирования, соединил ее с программой записи экрана рабочего стола и приправил потоковым вещанием live... и, как нетрудно догадаться, на усвоение основ понадобится время.

В окне приложения — несколько областей, самая важная из которых — центральная область «рабочего стола» со множеством вкладок в верхней части окна. Из вкладок можно собрать все медиа-источники, которые вы планируете использовать в своем вещании, например, видеоклипы, аудиофайлы, изображения, записанные сессии рабочего стола и, конечно, живое изображение с web-камеры.

Хотя область «рабочего стола» невелика, это монтажный стол, позволяющий разместить свои источники, задать их размер и прозрачность, и т.д.

В нижней левой панели WebcamStudio размещены «каналы». Это просто ярлыки источников вашего видео. Создав для всех источников каналы, вы сможете легко переключаться между ними и воспроизводить их с комментариями. Для более профессионального подхода к пакету можно использовать переходы и видеоэффекты из панели Source Properties [Свойства Источников].

Вы можете дать другим шансы насладиться вашей работой, перенаправив поток на один или несколько имеющихся выходов. Список весьма обширен, и выходы



К счастью, все эти мелкие кнопки в верхней части окна снабжены удобными подсказками.

включают чистое аудио (в результате получается канал наподобие радио), Twitch, IceCast или иной поддерживаемый сообществом онлайн-сервис, и есть даже виртуальная камера-муляж для тестирования.

WebcamStudio — это приложение на Java, и, следует признать, сборок Linux пока не так уж много. Вы всегда можете попробовать его на совместимом с Ubuntu дистрибутиве, используя `ppa:webcamstudio/webcamstudiodailybuild`.

## Переключайте источники и воспроизводите с комментариями.

Генератор паролей

# KeePassX

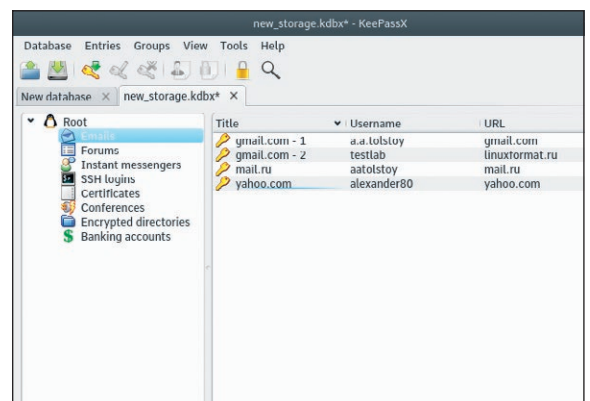
Версия: 2.0.2 Сайт: [www.keepassx.org](http://www.keepassx.org)

**Н**еобходимость запоминать десятки паролей электронной почты, форумов и кучи личных учетных записей в Сети вгонит в отчаяние кого угодно. Многие выбирают простейший путь — и самый опасный: задают один и тот же пароль для всего. Очевидно, что это очень плохая, хоть и весьма нередкая практика, которая подрывает всю идею безопасности персональной информации.

Куда лучше создать мнемоническое правило для длинных и сложных паролей, однако мало кто этим пользуется. KeePassX является выходом для любого, кому нужна усиленная безопасность при всех паролях, которые хранятся наготове. KeePassX — это инструмент, похожий на KeePass 2 [см. HotPicks, стр. 101 LXF183]. У обоих приложений функции практически одинаковые, но KeePass 2 для Linux использует среду Mono под управлением GTK, тогда как KeePassX написан на чистом C и применяется управление Qt.

Его основная идея — собрать все ваши пароли и прочие важные сведения в одной зашифрованной базе данных, защищенной мастер-паролем, и это — единственная последовательность символов, которую вам настоятельно не рекомендуется забывать! После первого запуска KeePassX вам придется создать новую базу данных с мастер-паролем и, опционально, с файлом ключа (XML с генерированными хаотическими помехами). Начиная с версии 2, KeePassX обзавелся полной поддержкой формата базы данных KDBX и умеет также использовать файлы DB из KeePass 2. Войдя в базу данных, вы можете начать создавать записи для сайтов, куда вы заходите с именем пользователя и паролем.

## KeePassX приобрел полную поддержку базы данных KDBX.



Укрепите свою безопасность с помощью надежного и владеющего шифрованием менеджера паролей.

KeePassX позволяет указать URL сайта и установить срок действия пароля по каждой записи. Кроме того, здесь есть стильный настраиваемый генератор паролей, где можно задать длину и сложность пароля.

Вспомогательные вкладки также помогут присоединять добавочные примечания и данные к записи, включая индивидуальные значки. Записи можно группировать и сортировать в отображении в виде дерева в левой части главного окна KeePassX, так что это — отличный способ классифицировать ваши учетные записи и поддерживать порядок в целом.

## Инструмент планирования

## FET

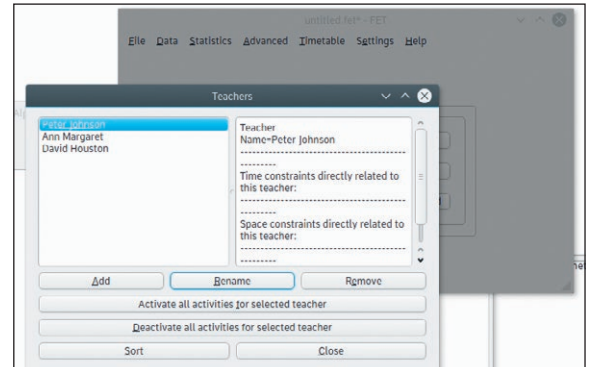
Версия: 5.28.4 Сайт: <http://lalescu.ro/liviu/fet>

**М**ы непроизвольно начали курировать в HotPicks еще одну серию программ по одной теме, и это — образовательные программы. В LXF205 мы рассмотрели *Intef-eXe* [она же *eXe-Learning*, см. HotPicks, стр. 102], которая помогает преподавателям тестировать своих студентов. На сей раз мы обращаемся к *FET* (*Free TimeTabling Software*), которая разработана в помощь организации образовательного процесса и составляет расписание с учетом обычных проблем — например, проблем с отведением аудиторий, конфликтами по времени, отсутствием учителей и пропуском предметов.

В сердцевине *FET* — оригинальный алгоритм автоматического разрешения конфликтов расписания и распределения всех занятий соответственно. *FET* — программа, с виду очень приятная и аккуратная, однако начать работать с ней сразу практически невозможно, не прочитав руководства пользователя (<http://bit.ly/FETManual>) — обширнейшего: даже что-то напортачив,

вы получите исчерпывающую подсказку. Для начала надо создать и сохранить файл настройки во вкладке File, и только после этого переходить в другие вкладки. Во вкладке Data содержатся основные записи — все они обязательны, и все требуется заполнить. Надо добавить название учебного заведения, группы учеников, учителей, аудитории и корпуса, и соединить всё это занятиями. Далее надо определить годы; поделить годы на месяцы, семестры или четверти; задать ограничения по времени и приписать их к уже созданным занятиям. На самом деле, следует очень многое сделать, прежде чем вы сумеете наконец-то создать расписание, и мы настоятельно рекомендуем ознакомиться с руководством, а не импровизировать.

## В сердце FET — алгоритм разрешения конфликтов расписания.



► Не пожалейте времени на начальную настройку *FET*, и сэкономите массу времени в будущем.

Но в итоге вы получите расписание без ошибок, которое можно будет просмотреть в меню Timetable с разными вариантами Отображения. Каждое расписание является документом HTML+CSS, разработанным в первую очередь для электронного использования, но его можно и распечатать из web-браузера (просто нажмите Ctrl+P). Внешний вид и работу полученных в результате файлов HTML можно настроить в Settings > Timetables > HTML level of generated timetables [Настройки > Расписания > HTML-уровень созданных расписаний].

## Аудиокодек

## Opus

Версия: 1.1.2 Сайт: <http://opus-codec.org>

**Б**ыли времена, когда опытные пользователи заботились о кодеках для своей локальной музыкальной библиотеки, потому что всё внимание уделялось уменьшению размера файлов, которое не слишком влияло на качество аудио.

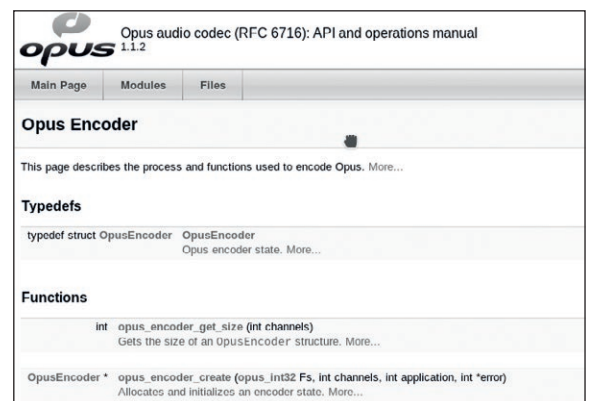
Два десятилетия царем горы был MP3, а его соперники — WMA и AAC — были менее популярны, хотя AAC предлагает лучшее качество при таком же битрейте [битрейт — скорость передачи данных]. Ныне музыка переехала в Сеть; но пожелав иметь собственный сервер потокового вещания или записывать подкасты, вы скоренько поймете, что пора вернуться к истокам. На самом деле, есть несколько эффективнейших аудиокодеков высшего качества с открытым кодом, таких, как Ogg Vorbis или FLAC, но стоит вам погрузиться в детали сжатия, как вы ощутите раздражение из-за ряда ограничений.

Основная проблема — в том, что существуют разные кодеки для кодирования

речи и музыки. Группа речи оптимизирована для более низких битрейтов, обеспечивающих лучшее качество человеческого голоса (например, Speex, AMR), а вторая группа блещет высокой точностью воспроизведения (например, OGG, AAC).

Кодек Opus — толковое сочетание того и другого. По умолчанию он производит очень достойный результат, сравнимый с другими кодеками и по качеству, и по степени сжатия. Вы можете попробовать его с помощью `$ opusenc input.wav output.wav`, но настоящая работа начинается при понижении битрейта. Например, попробуйте сравнить файлы, созданные командами `$ lame -b 16 /usr/share/sounds/alsa/Front_Center.wav ~/1.mp3`

## Сравним с другими по качеству и коэффициенту сжатия.



► Есть дополнительные модули и 97 страниц документации API для тех, кто хочет заниматься разработкой с Opus.

```
$ opusenc --bitrate 16 /usr/share/sounds/alsa/Front_Center.wav ~/1.Opus
```

Вы увидите, что файл *.opus* звучит отлично, несмотря на объем всего в 3К; и это делает кодек идеальным для передачи интерактивной речи и аудио через Интернет. Секрет его потрясающей производительности в том, что Opus использует кодек SILK от Skype для низких битрейтов и кодек CELT от Xiph.Org для высококачественных записей. В Linux кодированные в Opus файлы поддерживаются многими плеерами, включая те, что базируются на VLC и FFmpeg.

## HotGames Развлекательные приложения

## Стратегия

## Widelands

Версия: Build19 Сайт: <https://wl.widelands.org>

**В** Linux немало встроенных стратегий реального времени. Одни заточены на битвы, а другие — на перевозки и логистику. *Widelands* размещает поверх обычных схваток с другими племенами и войн некоторый экономический уровень.

Игра начинается на клочке земли, где вы начинаете обживать со своим племенем. Изначально можно выбрать одно из трех племен: варваров [Barbarians], жителей империи [Imperials] и атлантов [Atlanteans], и у каждого из них есть и положительные, и отрицательные характеристики. Обобщая, можно сказать, что варвары хорошо сражаются, но у них слабо развита наука; у имперцев налицо культурное превосходство; атланты умны и разбираются в технологиях.

*Widelands* напоминает многие классические стратегии, но больше всего — игру

*Settlers II* [Поселенцы] из середины 1990-х. Цель *Widelands* практически такая же, как и во многих других стратегиях: собрать природные ресурсы (камни, лес и руду); построить побольше домов; «вырастить» рабочих и солдат и, наконец, атаковать противника, чтобы выжить и стать повелителем мира.

Некоторые особенности *Widelands* проистекают из ограниченных ресурсов компьютера эры *Settlers*: здания можно строить только на определенных участках карты, что вынуждает тщательно продумывать землеотвод. Еще одна вещь, о которой следует помнить —

**Размещает экономический уровень над борьбой с другими.**



► *Widelands* заставит вас тщательно продумывать, как наилучшим образом использовать территорию для строительства и сбора ресурсов.

это дороги, необходимые для нормального развития вашей экономики. В *Widelands* можно использовать разные стратегии: например, быстро создать армию, чтобы физически уничтожить противника, или поспешить и занять больше земли, чтобы оттеснить его.

В *Widelands* очень подробная система подсказок и удобный режим руководства. Новым пользователям желательно ознакомиться с основами игры, поскольку в будущем это позволит сэкономить немало усилий.

## Аркадная стрелялка

## Duckhunt-JS

Версия: 2.0 Сайт: <http://bit.ly/DuckHunt-JS>

**Е**сть нечто удивительное в старых классических играх — которым был нужен телевизор, дорогая видеокарта с проводным джойстиком и видеоружьем, и, конечно, картридж игры — умещавшийся в крошечный архив объемом всего 1 МБ. Среди разработчиков открытого кода возникла тенденция воссоздавать эти стрелялки в виде браузерных игр. Вообще-то не только стрелялки, поскольку у нас имеется множество обычных платформенных игр и даже странные создания из прошлого (например, задайте поиск для Windows 95 в браузере), которые как родные работают внутри вашего браузера *Firefox* или *Chrome*. Во многих случаях это возможно благодаря сочетанию Node.js и кода JavaScript.

*Duckhunt-JS* — нестареющая классика, выпущенная Nintendo в 1984 г. для NES (Nintendo Entertainment System). Как и следует ожидать, цель игры — палить

по виртуальным уткам, которые появляются на экране, и заработать как можно больше очков. Подстрелив должное количество уток, вы переходите на следующий уровень, но если этого сделать не удастся, игре конец. Из классики также взяли охотничью собаку, которая радостно подбегает к подстреленным уткам и потешается над вами, если вы промахнулись.

Вам незачем покупать древнюю консоль и световое ружье NES Zapper, чтобы насладиться этой отличной игрой; не нужен вам и программный эмулятор NES для Linux или какой-то обходной путь, поскольку *Duckhunt-JS* — точный клон игры-оригинала: он написан на JavaScript

**Точный клон, совместим с любым современным браузером.**



► При подготовке этого обзора классической охоты на уток ни одна настоящая утка не пострадала.

и совместим с любым современным браузером. Перед запуском игры убедитесь в наличии у вас менеджера пакетов *NPM* для Node.js — он должен быть скачан и установлен; или клонируйте репозиторий со страницы GitHub, распакуйте и запустите `$ npm install` в его директории. Для запуска игры надо всего лишь открыть файл `index.html`. Кроме того, игра настраивается: можно менять количество волн, уток и т. д., и жульничать, продлевая таймер.



## Текстовый редактор

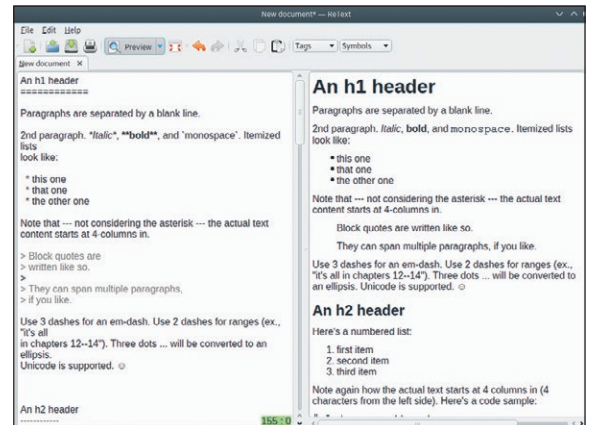
## Retext

Версия: 5.3 Сайт: <https://github.com/retext-project/retext>

**M**arkdown становится все популярнее среди web-разработчиков и технарей. Это простой язык разметки, который позволяет создавать отлично отформатированные и аккуратные документы HTML без написания тэгов вручную и, в то же время, без необходимости слепо полагаться на графический редактор WYSIWYG.

*Retext* — это текстовый редактор Markdown, с набором функций для написания документов Markdown легко и с удобствами. Он также поддерживает reStructuredText — похожий язык с расширенными функциями, такими как таблицы и создание стилей. Редактор выглядит минималистским: небольшая белая область для написания контента и скромная панель инструментов вверху. *Retext* поддерживает вкладки, как в браузере, и вы можете открыть несколько документов и переключаться между ними. Освоив синтаксис разметки, вы сможете начать печатать, но рано или поздно вам понадобится предпросмотр

результата. В *Retext* для этого есть несколько способов. Доступ к разметке HTML с выделенными тэгами можно получить через меню Edit > View HTML Code. Всплывающее окно возникнет, как только вы выберете код и скопируете его в буфер. Обратите внимание на кнопку Preview на панели инструментов *Retext*. Просто щелкните по ней для перехода из режима предпросмотра к включенной разметке и обратно; и можно также щелкнуть по стрелке рядом с кнопкой Preview, чтобы включить предпросмотр в режиме Live. В этом случае и разметка, и обработанная страница HTML будут существовать бок о бок, что весьма удобно, когда вы хотите посмотреть, как изменения в вашем коде влияют на рендеринг страницы. По умолчанию рендеринг страницы



➤ *Retext* в действии: зеленая подсказка удобно показывает номер строки и количество символов.

производит устаревший движок KHTML, но вы можете сменить его на WebKit через Edit > Use WebKit renderer.

*Retext* позволяет экспортировать свой документ через удобное меню File > Export, и не только в HTML, но также в форматы ODT и PDF. Сайт проекта на GitHub предлагает очень удобную страницу wiki с расширениями Markdown и общими подсказками по написанию, например, математических формул.

**По стрелке можно включить предпросмотр в режиме Live.**

## CD-риппер

## ABCDE

Версия: 2.7.1 Сайт: <http://abcde.einval.com>

**П**ростейший ныне способ послушать свою любимую музыку — выбрать web-сервис; но самый надежный, свободный от DRM и проверенный временем вариант — открыть футляр, достать CD и наслаждаться, совсем как в старые добрые времена.

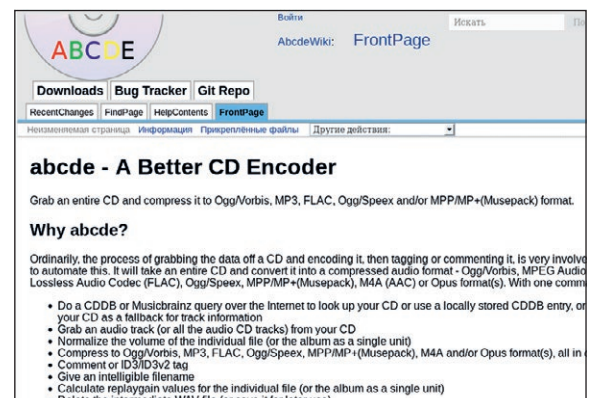
Всего несколько лет назад программы для обдиранья CD были исключительно популярны среди пользователей, переносивших свою музыку с физических дисков в локальную музыкальную библиотеку на жестком диске. Программы вроде *K3b* или *cdparanoia* живы до сих пор и работают приемлемо, но есть более элегантное решение, в котором нет столько зависимостей и которое делает свое дело, не задавая лишних вопросов; и это — *ABCDE*.

На самом деле данное название означает *A Better CD Encoder*. Суть этой программы для извлечения и кодирования в том, чтобы делать несколько работ одновременно, например, брать данные с CD (сразу со всего

диска или по трекам), перекодировать их (например, с помощью *Opus*), опросить сервисы типа CDDb или Musicbrainz, чтобы заполнить тэги, и дать осмысленные имена файлам и директориям. *ABCDE* также умеет нормализовать громкость перед сжатием и легко настраивается для любой работы: например, можно с помощью *ABCDE* перекодировать целый альбом в один файл FLAC со встроенной музыкальной справкой или осуществлять воспроизведение через другие программы перекодирования и извлекать каждый трек в отдельный файл.

*ABCDE* — инструмент командной строки, благодаря чему идеален для автоматизации разных процедур в оболочке. В нем также имеется впечатляющий набор опций

**Суть этой программы — делать несколько работ одновременно.**



➤ Прежде чем писать скрипт оболочки самому, рассмотрите вариант использования *ABCDE* для автоматического вскрытия и перекодирования CD.

в `/etc/abcde.conf`, который можно сменить на ваш персональный из `$HOME/abcde.conf`. На официальном сайте проекта есть очень полезные шаблоны настроек, которые можно просто скопировать и вставить в свою настройку.

*ABCDE* работает без проблем, и ему нужны только *cdparanoia*, рабочее интернет-соединение (для онлайн-запросов) и программа перекодирования, которую вы хотите использовать. **LXF**

# На диске

Дистрибутивы, приложения, игры, книги и всякое-разное...

Лучшее из Интернета, упакованное в 9 ГБ качественного DVD.



## Дистрибутивы

Как-то я был в гостях у друга, и его сын смотрел мультики на медиа-плейере. Когда он закончил и вернулся на домашний экран, я увидел, что это был *Kodi*, работающий на OpenELEC (по случайному совпадению это было буквально через пару дней после того, как я включил OpenELEC в DVD этого месяца). Тут нет ничего примечательного, вот только ИТ-знания этого человека в лучшем случае ограничиваются калькулятором. Linux долго был вотчиной лиц с определенным уровнем технических знаний или повышенной любознательностью. И вот у кого-то на бытовом оборудовании работает Linux. Да, технически Linux вот уже несколько лет находится в руках миллионов в виде устройств Android, но здесь это был GNU/Linux, тот, о котором мы обычно думаем, когда упоминаем Linux, а не какая-то другая ОС, работающая на ядре Linux. Linux уже давно присутствует на встраиваемых устройствах, например, роутерах, но сейчас мы видим устройства, где работает нечто близкое к настольному Linux, дополненное X, в руках людей, которые о Linux и не слыхивали. Может, этот год и не стал годом Linux на рабочем столе, но, кажется, план мирового господства Линуса Торвальдса реализуется со скоростью одного устройства за раз.

*Neil*

## » Важно ВНИМАНИЕ!

### Порченные диски

В маловероятном случае какого-то дефекта вашего LXF DVD обращайтесь, пожалуйста, по адресу [disks@linuxformat.ru](mailto:disks@linuxformat.ru) или телефону +7 (812) 309-0686.

## Медиа-центр

# OpenELEC 6.0.1

Вы можете установить *Kodi* (медиа-плейер, ранее известный как *XBM3*) практически на любой дистрибутив, но если вам нужен специальный медиа-центр, то общецелевой дистрибутив будет излишеством. Здесь-то и вступает в игру OpenELEC — это крошечный дистрибутив, который достаточно велик для Linux, способного запустить и отобразить *Kodi* — а именно это вам и нужно на медиа-системе. У нас здесь две версии, одна для 64-битного ПК и одна для Raspberry Pi. Обе в виде сжатых *Gzip* файлов образов. Версия для Pi копируется на карту с помощью *dd* обычным способом. ПК-версия копируется на USB-брелок точно так же, затем компьютер загружается с брелка.



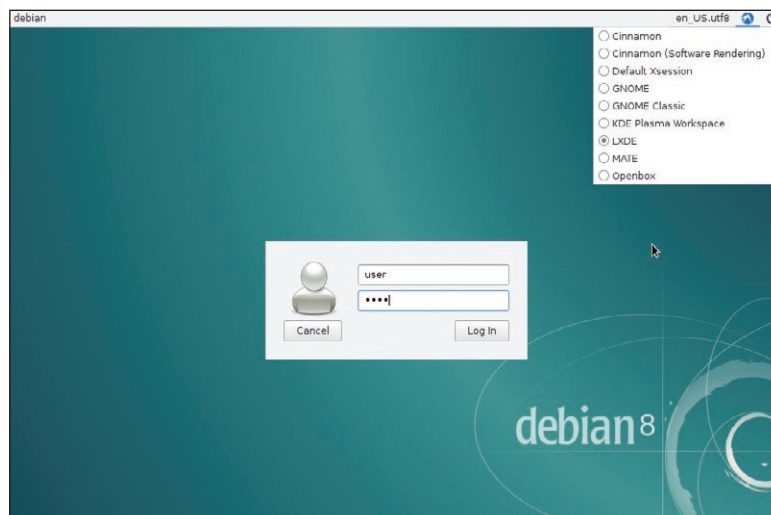
## Прародитель

# Debian 8.3.0 Remix

Может быть, у Debian и велики промежутки между основными релизами, но тем не менее разработчики периодически выпускают обновленные версии. Это Debian 8.3.0, который не особенно отличается от релиза Debian 8, но имеет достаточно обновлений, чтобы его стоило включить.

Debian производит несколько live DVD, каждый со своим рабочим столом. Мы рассмотрели альтернативы, постарались понять, какой лучше, а потом

решили, что у нас будет один из наших Ремиксов рабочих столов live с Cinnamon, Gnome, KDE, LXDE и Mate. По умолчанию DVD загружает Cinnamon, а чтобы попробовать какой-то другой рабочий стол, выйдите, выберите желаемый рабочий стол из меню сверху справа (рисунок внизу), затем снова зайдите от имени пользователя *user* с паролем *live*. Ремикс можно установить на ваш жесткий диск обычным способом; откройте установщик в разделе меню System Tools.



**Крутой медиа-центр** **Построим!**

**OpenELEC 6.0** для ПК и Pi

- Смотрите и упорядочивайте все свои фильмы
- Встроенный телевид и видеозапись
- Наслаждайтесь музыкой, фото и прочим!

**Ubuntu** для планшетов  
Создайте планшет на Linux-основе

**Rescatux 0.40b5**  
Лучший диск-образ для спасения

**Debian 8.3.0**  
Мини-дистрибутив для ПК

**LINUX LIVE-ДИСК: ГОТОВ К РАБОТЕ**  
ВСЕ НЕОБХОДИМОЕ ДЛЯ СТАРТА В LINUX

# Новичок в Linux? Начните отсюда!

- » Что такое Linux? Как его установить?
- » Есть ли в нем эквивалент MS Office?
- » Зачем нужна командная строка?
- » Как устанавливать программы?

Ответы приводятся в [Index.html](#) на диске.

## Минималист

# Arch Linux для Pi

Arch Linux — дистрибутив, популярность которого за последние несколько лет значительно выросла. За тот же период, несомненно, столь же выросла популярность и некоего интересного оборудования — а именно, Raspberry Pi, поэтому явно имеет смысл их объединить.

Хотя самым распространенным выбором дистрибутива в данном случае является Raspbian, порт Arch Linux на ARM как раз подходит для легковесного Pi и создает основу для одной из статей этого месяца.

Наша версия Arch также подходит для моделей Pi A, A+, B или B+, но не будет работать на Pi 2, потому что там требуется другая настройка ядра.

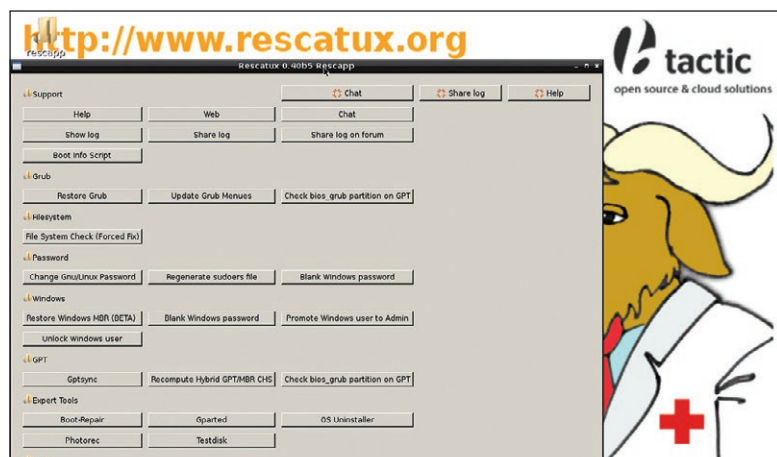


## Восстановление

# Rescatux 0.40b5

Без неприятностей не обойтись, такова правда жизни. И хорошо бы всегда иметь под рукой спасательный диск. В этом месяце Сравнение было посвящено пяти представителям этого вида, и победителем стал... Впрочем, не будем портить удовольствие тем, кто еще не читал обзор; а на LXF DVD этого месяца есть Rescatux. Большинство дисков аварийного восстановления концентрируются на включении всех инструментов диагностики и восстановления, способных вам понадобиться, и обычно загружаются в командную строку или очень минималистский рабочий стол, чего

достаточно для запуска браузера. Это неплохо, если вы хорошо знакомы с указанными инструментами командной строки, но если вы умудрились испортить свои настройки Grub и просто хотите, чтобы все снова загружалось как можно скорее, то чтение документации по опциям командной строки явно не будет вашим первым выбором. Rescatux загружает рабочий стол, содержащий окно с кнопками, и каждая из них исправляет одну из частых проблем, поэтому обычно вы можете исправить поврежденный загрузчик или обрести забытый пароль Windows за считанные секунды. LXF



## И еще!

### Системные инструменты

#### Главное

**Checkinstall** Установка tar-архива с помощью менеджера пакетов.

**GNU Core Utils** Основные утилиты, обязательные присутствовать в каждой операционной системе.

**Hardinfo** Инструмент тестирования системы.

**Kernel** Исходный код самого последнего стабильного релиза ядра.

**Memtest86+** Проверьте ОЗУ на предмет сбоев.

**Plop** Простой менеджер загрузки для запуска ОС с CD, DVD и USB.

**RaWrite** Создавайте загрузочные диски в MS-DOS в Windows.

**SBM** Независимый от ОС менеджер загрузки с несложным интерфейсом.

**WvDial** Соединяйтесь с Интернетом через телефонный модем.

### Чтение

#### Книжная полка

**Расширенное руководство по скриптам Bash** Изучите написание скриптов еще лучше.

**Руководство Bash для начинающих** Осваивайте написание скриптов Bash.

**Руководство по созданию скриптов Bourne Shell** Начните осваивать скрипты оболочки.

**The Cathedral and the Bazaar [Собор и Базар]** Классический текст Эрика С. Реймонда [Eric S Raymond], объясняющий преимущества открытой разработки.

**Справочник администратора Debian** Базовое руководство для системных администраторов.

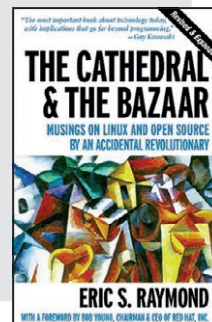
**Введение в Linux** Удобное руководство со множеством подсказок для начинающих пользователей Linux.

**Словарь Linux** Linux от A до Я.

**Linux Kernel in a Nutshell [Ядро Linux в Двух Словах]** Введение в ядро, написанное великим магистром ядра Греггом Кроа-Хартманом [Greg Kroah-Hartman].

**Руководство системного администратора Linux** Полностью контролируете вверенную вам систему.

**Обзор инструментов** Исчерпывающий обзор инструментов GNU.





## Пропустили номер?



Закажите его через сайт [www.linuxformat.ru](http://www.linuxformat.ru) в «ГНУ/Линуксцентре»! Журналы доставляются и в печатной, и в электронной форме, так что получение нужного вам выпуска LXF может занять всего пару минут с момента открытия браузера!

Прямо сейчас для заказа доступны следующие номера:

 <p><b>LXF205/206</b> Февраль 2016</p> <ul style="list-style-type: none"><li>» Умывальников начальник Управляем умным домом</li><li>» Видеоплейеры Лучше семь раз увидеть</li><li>» Raspberry Pi Zero Компьютер за 5 долларов!</li><li>» Организация данных Диски и как с ними бороться</li></ul> <p><b>LXFDVD:</b> Fedora, Korora, Kubuntu, Tails, Ubuntu, 10 книг о Linux (на английском языке), горячие новинки и прочее...</p> <p>Печатная версия: <a href="http://shop.linuxformat.ru/lxf_205-206/">shop.linuxformat.ru/lxf_205-206/</a> PDF-версия: <a href="http://shop.linuxformat.ru/elxf_205-206/">shop.linuxformat.ru/elxf_205-206/</a></p>	 <p><b>LXF207</b> Март 2016</p> <ul style="list-style-type: none"><li>» Разбегаемся из Окна Прощай, Windows!</li><li>» Резервное копирование Чтобы не было беды</li><li>» Говорят художники Свободным искусствам — свободные инструменты</li><li>» Нас хранит богиня Кали Тесты на вторжение</li></ul> <p><b>LXFDVD:</b> Linux Mint, openSUSE, Clonezilla, ROSA, Sabayon, Tiny Core, 10 книг о Linux (на английском языке), горячие новинки и прочее...</p> <p>Печатная версия: <a href="http://shop.linuxformat.ru/lxf_207/">shop.linuxformat.ru/lxf_207/</a> PDF-версия: <a href="http://shop.linuxformat.ru/elxf_207/">shop.linuxformat.ru/elxf_207/</a></p>	 <p><b>LXF208</b> Апрель 2016</p> <ul style="list-style-type: none"><li>» Разоблачаем хакеров Наладить оборону</li><li>» Кодирем видео Смотреть всё, смотреть везде</li><li>» Дистрибутивов много не бывает Мультизагрузка</li><li>» Что я знаю о системе Анализ среды</li></ul> <p><b>LXFDVD:</b> Bodhi, Fedora Security, GParted, Kali Light, LXLE, Manjaro, Robotlinux, 10 книг о Linux (на английском языке), горячие новинки и прочее...</p> <p>Печатная версия: <a href="http://shop.linuxformat.ru/lxf_208/">shop.linuxformat.ru/lxf_208/</a> PDF-версия: <a href="http://shop.linuxformat.ru/elxf_208/">shop.linuxformat.ru/elxf_208/</a></p>
---	--	--

Подпишитесь на печатную версию журнала через [www.linuxformat.ru/subscribe](http://www.linuxformat.ru/subscribe) или [www.linuxcenter.ru](http://www.linuxcenter.ru), и получите электронную версию в подарок!

На сайте [shop.linuxformat.ru](http://shop.linuxformat.ru) вы также сможете приобрести предыдущие выпуски LXF.

Телефоны отдела подписки

- » Санкт-Петербург: (812) 309-0686
- » Москва: (499) 271-4954



Linux Format ВКонтакте: [vk.com/linuxform](https://vk.com/linuxform)  
Вступайте в нашу группу!

MySQL — одна из самых популярных систем управления базами данных с открытым кодом

Оформите подписку на глобальную техническую поддержку Oracle для продуктов линейки MySQL, и вам будут доступны:

- 7500 специалистов в режиме 24×7
- Неограниченное количество запросов на обслуживание через Интернет или по телефону
- Патчи и обновления на портале MyOracleSupport.com
- Опыт использования продукта более чем у тысячи клиентов



+7 812 309 0686

[WWW.LINUXCENTER.RU/SHOP/MY\\_SQL](http://WWW.LINUXCENTER.RU/SHOP/MY_SQL) ● [INFO@LINUXCENTER.RU](mailto:INFO@LINUXCENTER.RU)

## Страница 1

## » Содержание



## ДИСТРИБУТИВЫ

## Debian 8.3.0 Jessie (64-битный)

OS Debian использует ядро Linux и поставляется более чем с 50000 пакетов (скомпилированное ПО, поставляемое в удобном формате для легкой установки на вашей машине). Проект Debian ставится своей надежностью: перед выпуском для пользователей всё его ПО тщательно тестируется.

## Rescatux 0.40b5 (64- и 32-битный)

Live-дистрибутив на базе Debian с графическим мастером для реанимации стюкманной установки GNU/Linux. Спасательные опции включают восстановление загрузки Grub после установки Windows, сброс пароля Linux и Windows и проверку файловой системы Linux.

## Axi для Raspberry Pi

Независимо разрабатываемый дистрибутив Linux со скрывающимися обновлениями. После установки будет автоматически поддерживаться в актуальном состоянии. В данной редакции оптимизирован для мини-компьютера Raspberry Pi. Обеспечивает прочную базу, позволяющую создавать индивидуальные системы.

## OpenMEEG 6.0.1 (64-битный и для Pi)

Встроенная ОС на базе Linux, созданная специально для запуска Kodi, развлекательного медиа-центра с открытым кодом. Идея OpenMEEG в том, чтобы использовать компьютер домашнего кинотеатра (HTPC) как любое другое устройство, прикреплённое к телевизору, например DVD-плеера. Максимально упрощает установку, управление и использование, поэтому, ближе к телевизору, чем к полноценному ПК.

## Ubuntu для планшетов 14.04.3

Ubuntu — полноценная ОС Linux для настольных компьютеров, свободно доступная с поддержкой сообщества и профессионалов — на сей раз выпущена для планшетов. Принцип компании-разработчика — предоставлять пользователям одноклассовый опыт работы на всех его устройствах благодаря унификации рабочего стола.

Описание на обороте »

## Информация о диске

## Что-то потеряли?

Часто случается, что новые программы зависят от других программных продуктов, которые могут не входить в текущую версию вашего дистрибутива Linux.

Мы стараемся предоставить вам как можно больше важных вспомогательных файлов. В большинстве случаев, последние версии библиотек и другие пакеты мы включаем в каталог «Essentials [Главное]» на прилагаемом диске. Поэтому, если в вашей системе возникли проблемы с зависимостями, первым делом следует заглянуть именно туда.

## Форматы пакетов

Мы стараемся включать как можно больше различных типов установочных пакетов: RPM, Deb или любых других. Просим вас принять во внимание, что мы ограничены свободным пространством и доступными двоичными выпусками программ. По возможности, мы будем включать исходные тексты для любого пакета, чтобы вы могли собрать его самостоятельно.

## Документация

На диске вы сможете найти всю необходимую информацию о том, как устанавливать и использовать некоторые программы. Пожалуйста, не забывайте, что большинство программ поставляются вместе со своей документацией, поэтому дополнительные материалы и файлы находятся в соответствующих директориях.

## Что это за файлы?

Если вы новичок в Linux, вас может смутить изобилие различных файлов и расширений. Так как мы стараемся собрать как можно больше вариантов пакетов для обеспечения совместимости, в одном каталоге часто находятся два или три файла для различных версий Linux и различных архитектур, исходные тексты и откомпилированные пакеты. Чтобы определить, какой именно файл вам нужен, необходимо обратить внимание на его имя или расширение:

- » **имя\_программы-1.0.1.i386.rpm** — вероятно, это двоичный пакет RPM, предназначенный для работы на системах x86;
- » **имя\_программы-1.0.1.i386.deb** — такой же пакет, но уже для Debian;
- » **имя\_программы-1.0.1.tar.gz** — обычно это исходный код;
- » **имя\_программы-1.0.1.tgz** — тот же файл, что и выше этажом по списку: «tgz» — это сокращение от «tar.gz»;
- » **имя\_программы-1.0.1.tar.bz2** — тот же файл, но сжатый bzip2 вместо обычного gzip;
- » **имя\_программы-1.0.1.src.rpm** — также исходный код, но поставляемый как RPM-пакет для упрощения процесса установки;
- » **имя\_программы-1.0.1.i386.FC4.RPM** — двоичный пакет RPM для x86, предназначенный специально для операционной системы Fedora Core 4;
- » **имя\_программы-1.0.1.ppc.Suse9.rpm** — двоичный пакет RPM, предназначенный специально для операционной системы SUSE 9.x PPC;
- » **имя\_программы-devel-1.0.1.i386.rpm** — версия для разработчиков.

## Если диск не читается...

Это маловероятно, но если все же прилагаемый к журналу диск поврежден, пожалуйста, свяжитесь с нашей службой поддержки по электронной почте: [disks@linuxformat.ru](mailto:disks@linuxformat.ru)

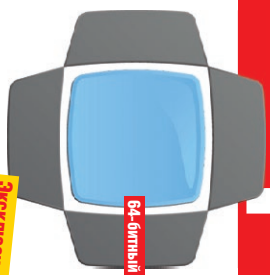
Внимательно прочтите это перед тем, как использовать LXF DVD!

# Крутой Медиа-Центр

Построим!

## OpenMEEG 6.0 для ПК и Pi

- » Смотрите и упорядочивайте все свои фильмы
- » Встроенные телегид и видеозапись
- » Наслаждайтесь музыкой, фото и прочим!



Создайте планшеты для планшетов на Linux немерцая!

## Ubuntu

Лучший дистрибутив для починки

## Rescatux 0.40b5

Мультиязычный от LXF

## Debian 8.3.0 64-битный

Экспресс!

**LIVE-ДИСК: ГОТОВ К РАБОТЕ**  
ВСЕ НЕОБХОДИМОЕ ДЛЯ СТАРТА В LINUX

# Содержание



## Сторона 2

### НОТРИСКИ

**AVCODE 2.71** CD-риппер

[about.einval.com](http://about.einval.com)

**DuckHunt-JS 2.0** Аркадная

стрелялка

[bit.ly/DuckHunt-JS](http://bit.ly/DuckHunt-JS)

**FET 5.28.4** Инструмент

составления расписаний

[lakeson.ro/iv/iv/fet](http://lakeson.ro/iv/iv/fet)

**KeepassX 2.0.2** Генератор паролей

[www.keepassx.org](http://www.keepassx.org)

**КМемо 0.17** Сетевой инструмент

[bit.ly/KMemoMonitor](http://bit.ly/KMemoMonitor)

**Moksha 0.2.0** Рабочий стол

[bit.ly/MokshaDesktop](http://bit.ly/MokshaDesktop)

**Мурмант 1.1.2** ПО для рисования

[muramint.org](http://muramint.org)

**Opus 1.1.2** Аудиокодек

[opus-codecs.org](http://opus-codecs.org)

**Retext 5.3** Текстовый редактор

[github.com/retext-org/retext](http://github.com/retext-org/retext)

**WebcamStudio 0.73** «Живой»

видеомикшер

[bit.ly/WebcamStudio](http://bit.ly/WebcamStudio)

**Widelands Build#19** Стратегия

[widelands.org](http://widelands.org)

### УЧЕБНИКИ

MongoDB

Respberry Pi

Swift

### ДОКУМЕНТАЦИЯ:

**11 КНИГ О LINUX**

**(НА АНГЛИЙСКОМ**

**ЯЗЫКЕ)**

**Advanced Bash Scripting Guide**

Подробное руководство

по программированию на Bash

**Bash Guide for Beginners**

Руководство по Bash

для начинающих

**Linux Kernel in a Nutshell**

Описание ядра Linux, созданное

одним из его выдающихся

разработчиков — Греггом Кра-

Хартоманом [Greg Kroah-Hartman]

**System Administrators Guide**

Руководство по базовому

администрированию Linux

**GNU Tools Summary** Руководство

по работе в командной строке

и обзор основных утилит GNU

### ГЛАВНОЕ

CheckInstall

Coreutils

HardInfo

Kernel

Memtest86+

Plor

SBM

WvDial

### ПОМОЩЬ

Руководство новичка

Руководства

Ответы

Чаво (FAQ)

### ДИСТРИБУТИВЫ

**openSUSE Leap 42.2 Alpha 1**

(64-битный)

Дистрибутив сообщества,

спонсируемый SUSE Linux

и другими компаниями, ставит

три основные цели: сделать

самым простым для получения

и наиболее широко используемым

дистрибутивом Linux; превратить

openSUSE в удобный в мире

дистрибутив Linux для новых

и опытных пользователей;

стать платформой выбора

для разработчиков Linux

и поставщиков ПО.

Это установочный образ: можно

произвести новую установку

или обновить существующую.

Все дистрибутивы представлены ISO-образами, которые можно записать на отдельный носитель, и загрузить в live-режиме прямо с LXF-DVD. У всех присутствует возможность установки на жесткий диск.

**Пожалуйста, перед использованием Аджунто Диска ознакомьтесь с инструкцией, опубликованной в журнале на стр. 109!**

**КОММЕНТАРИЙ** Присылайте ваши пожелания и предложения по электронной почте: [shobal@lxf.ru](mailto:shobal@lxf.ru)

**Две стрелки** в желтом треугольнике означают обнаруженные дефекты на данной странице, обращайтесь, пожалуйста, по адресу [shobal@lxf.ru](mailto:shobal@lxf.ru)

Настоящий диск тщательно тестировался и проверялся на всех стадиях производства, однако, как и в случае с любым новым ПО, мы рекомендуем вам использовать адаптивный сканер. Мы также рекомендуем всегда иметь под рукой актуальную резервную копию данных вашего жесткого диска. К сожалению, редакция Linux Format не в состоянии принимать на себя ответственность за любые повреждения, разрушения или иные убытки, которые могут повлечь за собой использование этого DVD, представленных на нем программ или данных. Прежде чем устанавливать какое-либо ПО на компьютер, пожалуйста, с нами, чтобы убедиться, что вы получаете именно то, что вам нужно.

Тираж издательства ООО «Марком», 186852, Россия, Ленинградская область, Всеволожский р-н, дер. Юрки, Школьная ул., 7-а, Лицензия ИДПТР ВАО № 77-03.

## Создание установочных дисков при помощи cdcrcord

Самый быстрый способ записать ISO-образ на чистую матрицу — это обратиться к программе *cdrecord*. Для всех перечисленных ниже действий потребуются права суперпользователя-root. Сначала определите путь к вашему устройству для записи дисков. Наберите следующую команду:

```
cdrecord -scanbus
```

После этого на экране терминала должен отобразиться список устройств, подключенных к вашей системе. SCSI-адрес каждого устройства представляет собой три числа в левой колонке — например, 0,3,0. Теперь вы можете с легкостью записать образ на диск:

```
cdrecord dev=0,3,0 -v /путь к образу/image.iso
```

Чтобы упростить дальнейшее использование *cdrecord*, сохраните некоторые настройки в файле */etc/default/cdrecord*. Добавьте по одной строке для каждого устройства записи (скорее всего, в вашей системе присутствует только одно такое устройство):

```
Plextor=0,3,0 12 16M
```

Первое слово в этой строке — метка; затем после адреса SCSI-устройства вы должны указать скорость и размер буфера. Теперь можете заменить SCSI-адрес в командной строке на выбранную вами метку. Все будет еще проще, если вы добавите следующее:

```
CDR_DEVICE=Plextor
```

Для записи ISO-образа вам осталось набрать команду

```
cdrecord -v /path/to/image.iso
```

Если вы не принадлежите к любителям командной строки, в таком случае вам придет на помощь утилита *gcombust*. Запустите ее от имени root и выберите вкладку Burn и ISO 9660 Image в верхней части окна. Введите путь к образу, который вы хотите записать на диск, и смело нажимайте на Combust! Пока ваш образ пишется на диск, можете выпить чашечку кофе.

### Другая ОС?

Использовать Linux для записи компакт-диска не обязательно. Программы вроде *cdrecord* просто переносят двоичные данные на чистую матрицу. Все необходимые файлы уже включены в ISO-образ, который распознается любой операционной системой, будь то Linux, Windows, Mac OS X или AmigaOS.

### Нет устройства для записи дисков?

Если у вас нет устройства, с помощью которого можно было бы записать образ на диск, можно найти какого-нибудь друга или организацию, у кого есть компьютер с дисководом, и прожечь диск у них. Опять-таки, вам подойдет любая операционная система, способная распознать пишущий привод (см. выше).

Некоторые дистрибутивы умеют монтировать образы дисков и выполнять сетевую установку или даже установку с раздела жесткого диска. Конкретные методы, конечно, зависят от дистрибутива. За дополнительной информацией обращайтесь на web-сайт разработчика дистрибутива.



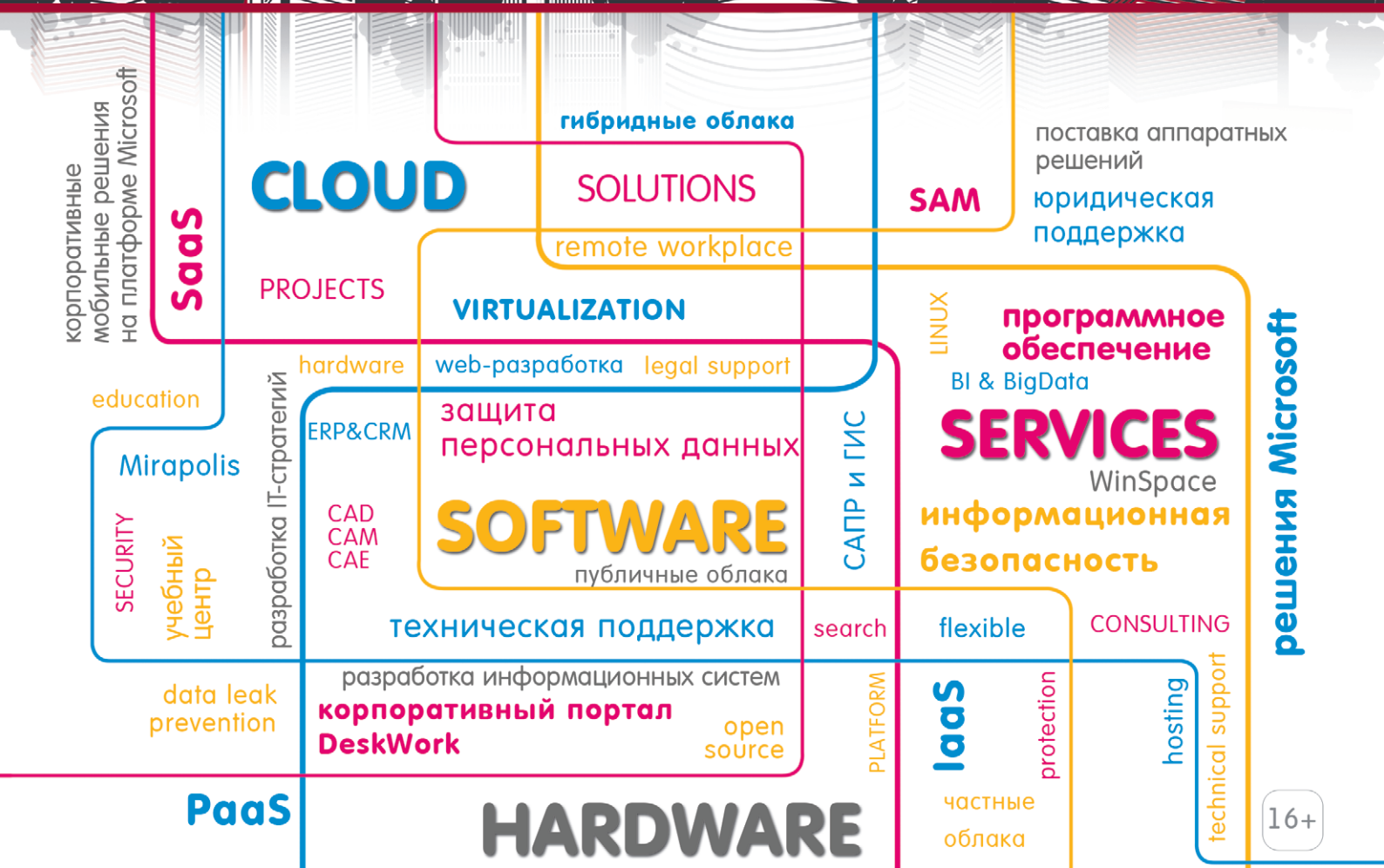
# softline®

Cloud Software Hardware Services

# 20+

Years in IT

## IT-архитектура вашего бизнеса





# LINUX FORMAT

Главное в мире Linux

Журнал зарегистрирован Федеральной службой по надзору за соблюдением законодательства в сфере массовых коммуникаций и охране культурного наследия ПИ № Ф077-21973 от 14 сентября 2005 года. Выходит ежемесячно. Тираж печатной версии 2000 экз., распространение электронной версии 30000 экз.

## РЕДАКЦИЯ РУССКОЯЗЫЧНОЙ ВЕРСИИ

Главный редактор

Кирилл Степанов [info@linuxformat.ru](mailto:info@linuxformat.ru)

Литературный и выпускающий редактор

Елена Толстякова

Переводчики

Елена Ессяк, Даниил Кривошеин, Светлана Кривошеина, Валентин Развозжаев, Валерий Смирнов, Елена Толстякова

Редактор диска

Александр Баракин

Верстка, допечатная подготовка

Сергей Рогожников

Технический директор

Андрей Смирнов

Директор по рекламе

Владимир Савельев [advert@linuxformat.ru](mailto:advert@linuxformat.ru)

Генеральный редактор

Павел Фролов

Учредители

Частные лица

Издатель

ООО «Линукс Формат»

Отпечатано в типографии ООО «ЛД-ПРИНТ»

196644, Санкт-Петербург, Колпинский р-н, пос. Саперный, территория предприятия «Балтика», д. 6/н, лит. Ф  
Тел. (812) 462-8383, e-mail: [office@ldprint.ru](mailto:office@ldprint.ru)  
Заказ 14177

## РЕДАКЦИЯ АНГЛОЯЗЫЧНОЙ ВЕРСИИ

Редактор Нейл Мор [Neil Mohr] [neil.mohr@futurenet.com](mailto:neil.mohr@futurenet.com)

Научный редактор Джонни Бидвелл [Jonni Bidwell]

[jonni.bidwell@futurenet.com](mailto:jonni.bidwell@futurenet.com)

Выпускающий редактор Крис Торнетт [Chris Thornett]

[chris.thornett@futurenet.com](mailto:chris.thornett@futurenet.com)

Художественный редактор Эфраин Эрнандес-Мендоса

[Efrain Hernandez-Mendoza] [efrain.hernandez-mendoza@futurenet.com](mailto:efrain.hernandez-mendoza@futurenet.com)

## ПОДГОТОВКА МАТЕРИАЛОВ

Джонни Бидвелл [Jonni Bidwell], Нейл Ботвик [Neil Bothwick], Джолион Браун [Jolyon Brown], Пол Хадсон [Paul Hudson], Дэйв Джеймс [Dave James], Кевин Ли [Kevin Lee], Пит Ломас [Pete Lomas], Нейл Мор [Neil Mohr], Ник Пирс [Nick Peers], Лес Паундер [Les Pounder], Афан Рехман [Atan Rehman], Том Сениор [Tom Senior], Маянк Шарма [Mayank Sharma], Шашанк Шарма [Shashank Sharma], Александр Толстой [Alexander Tolstoy], Михалис Цухалос [Mihalis Tsoukalos], Евгений Балдин, Андрей Гондаренков, Павел Емельянов, Дмитрий Пантеличев, Виталий Сороко, Алексей Федорчук, Лада Шерышова

Иллюстрации Шейн Коллиндж [Shane Collinge]

Иллюстрация с обложки [www.magictorch.com](http://www.magictorch.com)

## КОНТАКТНАЯ ИНФОРМАЦИЯ

UK: Linux Format, 30 Monmouth Street, Bath BA1 2BW

Тел. +44 (0) 1604 251045, email: [linuxformat@myfavouriteimagazines.co.uk](mailto:linuxformat@myfavouriteimagazines.co.uk)

РОССИЯ: Санкт-Петербург, пр. Медиков, 5, корп. 7

Тел. +7 (812) 309-0686, e-mail: [info@linuxformat.ru](mailto:info@linuxformat.ru)

По вопросам сотрудничества, партнерства, оптовых закупок:

[partner@linuxcenter.ru](mailto:partner@linuxcenter.ru)

**Авторские права:** статьи, переведенные из английского издания Linux Format, являются собственностью или лицензированы Future Publishing Ltd (Future plc group company). Все права зарегистрированы. Никакая часть данного журнала не может быть повторно опубликована без письменного разрешения издателя.

Все письма, независимо от способа отправки, считаются предназначенными для публикации, если иное не указано явно. Редакция оставляет за собой право корректировать присланные письма и другие материалы. Редакция Linux Format получает неэксклюзивное право на публикацию и лицензирование всех присланных материалов, если не было оговорено иное. Linux Format стремится оставлять уведомление об авторских правах всюду, где это возможно. Свяжитесь с нами, если мы не упомянули вас как автора предложенных вами материалов, и мы постараемся исправить эту ошибку. Редакция Linux Format не несет ответственности за опечатки.

Ответственность за содержание статьи несет ее автор. Мнение авторов может не совпадать с мнением редакции.

Все присланные материалы могут быть помещены на диски — CD или DVD, поставляемые вместе с журналом, если не было оговорено иное.

**Ограничение ответственности:** используйте все советы на свой страх и риск. Ни при каких условиях редакция Linux Format не несет ответственность за повреждение или ущерб, нанесенные вашему компьютеру и периферии вследствие использования тех или иных советов.

Linux — зарегистрированный товарный знак Линуса Торвальдса [Linus Torvalds].

©GNU/Linux заменяется на "Linux" в целях сокращения. Все остальные товарные знаки являются собственностью их законных владельцев. Весь код, опубликованный в журнале, лицензирован на условиях GPL v3. См. [www.gnu.org/copyleft/gpl.html](http://www.gnu.org/copyleft/gpl.html)

За информацией о журнале, издаваемом Future plc group company, обращайтесь на сайт [www.futurepic.com](http://www.futurepic.com)

## В июньском номере

# Соорудим свой... рабочий стол!

Настраиваем — потому что мы это можем! Создадим идеальное рабочее окружение, между KDE и Gnome.

## Строим Linux-ПК

Раз уж мы смастерили идеальный рабочий стол, отчего же не собрать из компонентов идеальный ПК — инструкция прилагается.

## Редактируем вместе

Объединим силы с друзьями и коллегами, чтобы создавать, обновлять и редактировать самые убедительные в мире документы.

## BBC Micro:bit опять здесь!

Продвинемся в изучении отличного образовательного инструмента от BBC и создадим крутые проекты.

Содержание будущих выпусков может меняться — вдруг нас завалит примочками рабочих столов...



© Linux Format 2005

© Future Publishing Ltd 2005

BATH • LONDON • MILAN • NEW YORK • PARIS • SAN DIEGO • SAN FRANCISCO



# Новое поколение средств защиты

## Межсетевые экраны ССПТ, не имеющие IP-адреса

ССПТ-2 — это сертифицированное ФСТЭК, ФСБ и ГАЗПРОМСЕРТ средство защиты информации нового поколения, реализующее функции межсетевого экрана, но при этом остающееся «невидимым» для любых протоколов и тестовых воздействий, что достигается за счет отсутствия физических и логических адресов на его фильтрующих интерфейсах. ССПТ-2 невозможно обнаружить никакими известными средствами удаленного мониторинга сети.

Скрытность функционирования межсетевого экрана повышает надежность системы защиты в целом и существенно упрощает процедуру установки ССПТ-2 в компьютерные сети и функционирующие на их основе информационные и телематические системы.

Защита для высокоскоростных корпоративных сетей Ethernet 100/1000 Мбит/с

Сертифицированы ФСТЭК и ФСБ (3-й класс защиты)

На базе процессоров с 64-разрядной многоядерной архитектурой



## Назначение устройства

Основное средство защиты для реализации различных политик информационной безопасности с помощью:

- фильтрации пакетов на канальном, сетевом, транспортном и прикладном уровнях;
- управления транспортными соединениями между отдельными узлами ЛВС или виртуальной ЛВС (VLAN);
- контроля контента данных на прикладном уровне с учетом направления, времени и типа протоколов передачи трафика.

Дополнительное устройство защиты для:

- обеспечения безопасности функционирования ранее установленных в компьютерной сети средств защиты и устройств маршрутизации;
- мониторинга трафика с возможностью анализа данных регистрации пакетов по различным критериям и интеграции с IDS;
- обеспечения функционирования сетевых распределенных телематических приложений и GRID-ресурсов.

Москва  
+7 (499)

271-49-54

Санкт-Петербург  
+7 (812)

309-06-86

Linux-эксперт для вашего бизнеса. [www.linuxcenter.ru](http://www.linuxcenter.ru)

Linux  center



**HETZNER**  
ONLINE

New Dedicated Server

Clever Solution.



Все цены указаны без учёта НДС и регулируются условиями предоставления услуг компанией Hetzner Online GmbH. Цены могут быть изменены. Все права защищены соответствующими производителями.



## Выделенный сервер EX51

Intel® Core™ i7-6700  
Quad-Core Skylake Processor  
64 ГБ DDR4 RAM  
2 x 4 ТБ SATA HDD Enterprise класс  
1 Гбит/с гарантированно  
100 ГБ место для резервных копий  
30 ТБ трафик\*  
Без минимального контракта  
Установка 8200 рублей

**4100**  
рублей в месяц

## Выделенный сервер EX51-SSD

Intel® Core™ i7-6700  
Quad-Core Skylake Processor  
64 ГБ DDR4 RAM  
2 x 500 ГБ SATA SSD Enterprise класс  
1 Гбит/с гарантированно  
100 ГБ место для резервных копий  
30 ТБ трафик\*  
Без минимального контракта  
Установка 8200 рублей

**4100**  
рублей в месяц

[www.ru.hetzner.com](http://www.ru.hetzner.com)

\* Нет платы за превышение. При превышении 30 ТБ/месяц скорость соединения ограничивается (подсчёт ведётся по исходящему трафику, входящий и внутренний трафик не учитывается). Опционально можно снять ограничение, подтвердив оплату 98 рублей за каждый дополнительный ТБ.