Справочник коммиттера

The FreeBSD Documentation Project
Дмитрий Морозовский

Издание: 43184

2013-11-13 hrs.

Аннотация

Данный документ содержит информацию для сообщества коммиттеров FreeBSD. Все новые коммиттеры должны изучить его перед началом работы; прочим коммиттерам также рекомендуется время от времени просматривать его.

Содержание

1. Административные детали

Хост основного репозитория	`ncvs.FreeBSD.org`
Способ авторизации	ssh(1), только протокол 2
Основной корень репозитория (CVSROOT)	`ncvs.FreeBSD.org :/home/ncvs` (см. также Раздел 3, «Работа с CVS»).
Администраторы Главного CVS Репозитория <*cvsadm@FreeBSD.org* >	Peter Wemm <`peter@FreeBSD.org` > и Mark Murray <`markm@FreeBSD.org` >, а также Josef Karthauser <`joe@FreeBSD.org` > и Joe Marcus Clarke <`marcus@FreeBSD.org` > для иерархии `ports/`
Списки рассылки	Список рассылки разработчиков документации FreeBSD, Список рассылки коммиттеров документации FreeBSD; Список рассылки разработчиков портов FreeBSD, Список рассылки коммиттеров портов FreeBSD; Список рассылки разработчиков FreeBSD, Список рассылки коммиттеров исходных текстов FreeBSD. (Каждому репозиторию проекта соответствуют отдельные списки рассылки с суффиксами -developers и -committers. Архивы этих списков хранятся в файлах `/home/mail/repository-name-developers-archive` и `/home/mail/`

	repository-name-committers-archive на машинах кластера FreeBSD.org).
Отчеты Правления	/home/core/public/monthly-report на машинах кластера FreeBSD.org .
Наиболее значимые метки CVS	RELENG_4 (ветвь 4.X-STABLE), RELENG_5 (ветвь 5.X-STABLE), RELENG_6 (ветвь 6.X-STABLE), HEAD (ветвь -CURRENT)

Для авторизации на машины проекта вы должны использовать протоколы ssh(1) или telnet(1) с включенным Kerberos 5. В случае ssh(1), допустим только протокол версии 2. Эти протоколы являются значительно более защищенными по сравнению с telnet(1) или rlogin(1), поскольку информация об авторизации передается в зашифрованном виде. По умолчанию, протокол ssh(1) также шифрует весь трафик. Учитывая наличие таких утилит, как ssh-agent(1) и scp(1), протокол ssh(1) значительно удобнее прочих в использовании. Если вы ничего не знаете об ssh(1), загляните в раздел Раздел 9, «SSH: быстрый старт».

2. Типы коммит битов

CVS Репозиторий FreeBSD состоит из нескольких разделов, охватывающих исходные тексты базовой операционной системы, документацию, инфраструктуру построения внешних приложений (портов), а также различные служебные утилиты. Право записи в репозиторий («коммит бит») подразумевает указание области дерева, в которое оно может быть применено. Как правило, области напрямую связаны с группой, подтвердившей право коммиттера на бит. В дальнейшем, область действия коммит бита может быть расширена; в этом случае, коммиттер должен следовать стандартным правилам нового для коммиттера в данной области, в частности, получая подтверждения на каждый коммит и, возможно, в течение какого-то времени работу с ментором.

Тип коммит бита	*Ответственные*	*Области репозитория*
src	core@	src/ и соответствующие части doc/
doc	doceng@	doc/, www/, документация дерева src/
ports	portmgr@	ports/

Биты, выделенные до разделения дерева на области могут использоваться во всех частях дерева. Однако, с точки зрения здравого смысла, коммиттер, не имеющий опыта работы в какой-либо части дерева, должен предоставлять предлагаемые изменения для рассмотрения другими коммиттерами, получать подтверждения от ответственных за различные части репозитория, а также, возможно, работать совместно с ментором. Поскольку правила ведения различных областей кода различаются, указанные нормы скорее направлены на благо коммиттера, не имеющего достаточного опыта работы в данной области.

Вне зависимости от области приложения усилий, запросы коммиттеров на рассмотрение предлагаемых изменений в процессе разработки могут только приветствоваться.

2.1. Правила для коммиттеров документации (doc/) при работе с деревом исходных текстов (src/)

- Коммиттеры документации могут самостоятельно изменять документацию к исходным текстам, например, страницы справочника, файлы README, базы данных утилиты fortune, календарей, а также исправлять комментарии в исходных текстах без дополнительного согласования с коммиттерами исходных текстов, при условии сохранения здравого смысла и манеры коммитов.

- Коммиттеры документации могут вносить незначительные изменения и исправления в исходные тексты (такие как исправления к процессу сборки, добавление малых дополнительных возможностей и т.п.) при наличии одобрения от коммиттера исходных текстов.

- Коммиттер документации может расширить область действия своего бита на область исходных текстов (и стать, таким образом, коммиттером исходных текстов), найдя себе ментора, который предложит это расширение Правлению (Core). После одобрения, строка с его именем пользователя вносится в файл 'access', и применяются обычные правила периода работы с ментором, подразумевающие получение одобрения на каждый коммит.

- Одобрение коммита ("Approved by") может производиться только "полновесным" (не работающим с ментором) коммиттером исходных текстов; последние могут рассматривать предлагаемые изменения и указываться в строке "Reviewed by".

3. Работа с CVS

Подразумевается, что вы уже имеете опыт базовой работы с CVS.

Администраторы Главного CVS Репозитория <`cvsadm@FreeBSD.org`> являются «владельцами» репозитория CVS и ответственны за все прямые его изменения (в целях чистки или исправления каких-либо вопиющих ошибок коммиттеров при работе с CVS). Если в результате ваших действий с частью репозитория произошел несчастный случай, например, после неверной операции `cvs import` или `cvs tag`, пошлите письмо соответствующей подгруппе Администраторы Главного CVS

Репозитория <cvsadm@FreeBSD.org > (см. следующую таблицу) и сообщите о проблеме. В наиболее серьезных случаях, касающихся не только какой-либо части репозитория, а дерева CVS в целом, вы можете написать Администраторы Главного CVS Репозитория <cvsadm@FreeBSD.org >. Пожалуйста, *не надо* писать группе Администраторы Главного CVS Репозитория <cvsadm@FreeBSD.org > по поводу репозиторного копирования и прочих вопросов, которые может решить соответствующая подгруппа.

Напрямую изменять содержимое репозитория может только группа CVS-мастеров; для обеспечения этого, только CVS-мастера имеют учетные записи на машинах, поддерживающих основной репозиторий.

Примечание

Адреса, на которые следует посылать запросы, зависят от области репозитория, которую требуется поправить:

- ncvs@ - репозиторий `/home/ncvs`, основные исходные тексты
- pcvs@ - репозиторий `/home/pcvs`, порты
- dcvs@ - репозиторий `/home/dcvs`, документация
- projcvs@ - репозиторий `/home/projcvs`, прочие проекты

Дерево CVS в настоящее время разделено на четыре независимых репозитория: `doc`, `ports`, `projects` и `src`. Для удобства работы пользователей при распространении через CVSup различные деревья комбинируются в одно, с одним служебным каталогом `CVSROOT`.

Примечание

Обратите внимание, что модуль `www`, содержащий исходные тексты веб-сайта FreeBSD, расположен в репозитории `doc`.

В настоящее время, все репозитории CVS располагаются на одной машине, `ncvs.FreeBSD.org`, однако, для обеспечения возможности в будущем разнести их по физически различным машинам, для каждой поддерживается отдельное имя хоста. Их и следует использовать коммиттерам. Наконец, каждый репозиторий расположен в отдельном каталоге. В итоге, общая картина выглядит так:

Таблица 1. Репозитории CVS FreeBSD, хосты и каталоги

Репозиторий	Хост	Каталог
doc	dcvs.FreeBSD.org	/home/dcvs
ports	pcvs.FreeBSD.org	/home/pcvs
projects	projcvs.FreeBSD.org	/home/projcvs
src	ncvs.FreeBSD.org	/home/ncvs

Операции с CVS производятся удаленно, путем установки переменной окружения `CVSROOT` (она должна указывать на соответствующий хост и каталог верхнего уровня, например `ncvs.FreeBSD.org :/home/ncvs`) и последующего выполнения команд выгрузки и коммита. Многие коммиттеры определяют команды-синонимы, разворачивающиеся в запуск cvs с правильными параметрами. В частности, пользователи оболочки tcsh(1) могут добавить следующие строки в свой скрипт начальной загрузки `.cshrc`:

```
alias dcvs cvs --d user@dcvs.FreeBSD.org:/home/dcvs
alias pcvs cvs --d user@pcvs.FreeBSD.org:/home/pcvs
alias projcvs cvs --d user@projcvs.FreeBSD.org:/home/projcvs
alias scvs cvs --d user@ncvs.FreeBSD.org:/home/ncvs
```

Теперь все операции с CVS могут выполняться на локальной машине, а для внесения изменений в официальное дерево CVS следует использовать команду `Xcvs commit`. Если вам нужно добавить в проект что-либо совершенно новое (например, исходные тексты сторонних разработчиков), нужно использовать команду `cvs import`; обратитесь к странице справочника по cvs(1) за подробностями.

Примечание

Пожалуйста, *не используйте* команды `cvs checkout` или `cvs update` для синхронизации ваших исходных текстов. Протокол CVS не оптимизирован для удаленной работы и требует значительных накладных расходов со стороны сервера. Пожалуйста, используйте метод синхронизации посредством `cvsup`, а основной хост используйте только для собственно коммитов. Наша распределенная сеть серверов cvsup достаточно развита. При необходимости синхронизации с самыми свежими изменениями вы можете пользоваться машиной `cvsup-master` , которая обладает достаточными ресурсами для удаленной работы с CVS; за нее отвечает Jun Kuriyama <`kuriyama@FreeBSD.org` >.

Если вам нужно использовать команды CVS `add` и `delete`, так чтобы в реальности переместить часть исходных текстов подобно действию команды mv(1), нужно запросить операцию «репозиторного копирования» (repository copy). При этом кто-либо из CVS-мастеров скопирует необходимые файлы внутри репозитория на нужное место и даст вам знать об этом. Репозиторное копирование производится для сохранения истории (журналов изменения). Возможность отследить историю изменений очень ценна для всего проекта FreeBSD.

Документация по CVS, учебные материалы и FAQ можно найти по адресу: `http://www.cvshome.org/docs/`. Очень полезна также книга Карла Фогеля (Karl Fogel) Open Source Development with CVS. Некоторая полезная информация о CVS на русском языке может быть найдена здесь.

Dag-Erling C. Smørgrav <`des@FreeBSD.org`> написал такой «мини-пример» работы с CVS:

1. Извлечение нужного модуля из репозитория: команда `co` или `checkout`.

   ```
   % cvs checkout shazam
   ```

   Эта команда извлечет копию модуля `shazam`. Если модуль с таким именем не существует (не описан в файле modules), будет произведена попытка извлечь директорию верхнего уровня `shazam`.

   Таблица 2. Полезные опции команды `cvs checkout`

`-P`	Не создавать (точнее, удалить после завершения выполнения) пустые каталоги
`-l`	Извлекать один уровень каталогов (без подкаталогов)
`-r`*`rev`*	Извлечь ревизию, ветвь или тег *rev* для указанного модуля
`-D`*`date`*	Извлечь состояние модуля в репозитории на момент *date*

   Примеры в применении к FreeBSD:

   - Извлечь модуль `miscfs`, расположенный в каталоге репозитория `src/sys/miscfs`:

     ```
     % cvs co miscfs
     ```

     После выполнения вы получите каталог `miscfs`, содержащий подкаталоги `CVS`, `deadfs`, `devfs` и т.д. Один из них (`linprocfs`) будет пустым.

- Извлечь те же файлы, но с полным путем:

  ```
  % cvs co src/sys/miscfs
  ```

  Теперь у вас есть каталог `src`, содержащий подкаталоги `CVS` и `sys`. Каталог `src/sys` содержит подкаталоги `CVS` и `miscfs` и т.д.

- Извлечь те же файлы, удалив при этом пустые подкаталоги:

  ```
  % cvs co --P miscfs
  ```

  Вы получите каталог `miscfs` с подкаталогами `CVS`, `deadfs`, `devfs`... однако без подкаталога `linprocfs`, поскольку он не содержит файлов.

- Извлечь каталог `miscfs` без подкаталогов:

  ```
  % cvs co --l miscfs
  ```

  Теперь в каталоге `miscfs` будет только один подкаталог `CVS`.

- Извлечь модуль `miscfs` из ветви 6.X:

  ```
  % cvs co --rRELENG_6 miscfs
  ```

  Теперь вы можете изменить исходные тексты и произвести коммит в эту ветвь.

- Извлечь модуль `miscfs` по состоянию на момент выхода 6.0-RELEASE:

  ```
  % cvs co --rRELENG_6_0_0_RELEASE miscfs
  ```

  В этом случае вы не сможете внести изменения в репозиторий, поскольку `RELENG_6_0_0_RELEASE` описывает момент времени, а не ветвь разработки.

- Извлечь модуль `miscfs` по состоянию на 15 января 2000 г:

  ```
  % cvs co --D'01/15/2000' miscfs
  ```

  Как и в предыдущем случае, изменения не могут быть записаны.

- Извлечь модуль `miscfs`, каким он был неделю назад:

  ```
  % cvs co --D'last week' miscfs
  ```

  И вновь, изменения не могут быть записаны.

Обратите внимание, что мета-данные хранятся в подкаталогах `CVS`.

Аргументы опций `-D` and `-r` сохраняются (являются «клейкими», sticky), например, при последующем использовании команды `cvs update`.

2. Проверка состояния извлеченных файлов: команда `status`.

```
% cvs status shazam
```

Эта команда покажет статус файла `shazam` или каждого файла в директории `shazam`. Для каждого из файлов статус может быть одним из:

Up-to-date	Файл соответствует репозиторию и не модифицировался
Needs Patch	Файл не изменялся, но репозиторий содержит обновленную версию
Locally Modified	Файл соответствует репозиторию, но был изменен локально
Needs Merge	Файл изменен локально; вместе с тем, файл изменен и в репозитории
File had conflicts on merge	После последнего обновления возникли конфликты, и они все еще не устранены

Кроме того, будут показаны локальная версия и дата модификации, версия и дата последней из доступных (если вы применяли «клейкие» дату, тег или ветвь, последняя доступная версия может отличаться от вашей), а также клейкие теги, временные метки и опции.

3. Обновление извлеченного модуля: команда `update`.

```
% cvs update shazam
```

Эта команда обновит состояние файла `shazam` или файлов в каталоге `shazam` до наиболее свежих версий выбранной вами при извлечении ветви. Если выбирался «момент времени», не произойдет ничего, если только за истекшее время в репозитории не был перемещен тег или не произошло чего-нибудь еще непредвиденного.

Полезные опции в дополнение к уже описанным для команды `checkout`:

`-d`	Извлечь вновь появившиеся или пропущенные ранее подкаталоги
`-A`	Обновиться до текущего состояния головной ветви
`-j`*rev*	магическая опция (см. ниже)

Если вы извлекали модуль с опциями `-r` или `-D`, выполнение команды `cvs update` с другими параметрами `-r` или `-D` или с опцией `-A` приведет к выбору новой ветви, ревизии или даты. Использование опции `-A` удаляет использованные ранее клейкие свойства; опции `-r` и `-D`, наоборот, фиксируют их.

Теоретически использование `HEAD` в качестве аргумента опции `-r` должно дать тот же результат, что и указание опции `-A`, однако это верно лишь в теории.

Опция `-d` полезна, если:

- после извлечения вами модуля кем-либо еще в него были добавлены дополнительные каталоги;
- вы извлекали верхний уровень модуля при помощи опции `-l`, а в дальнейшем решили извлечь и подкаталоги;
- вы удалили какие-либо подкаталоги и теперь хотите вновь извлечь их.

Обращайте внимание на вывод команды `cvs update`. Действие, произведенное с файлом, обозначается буквой перед его именем:

`U`	Файл был успешно обновлен.
`P`	Файл был успешно обновлен (произведен успешный патч из удаленного репозитория).
`M`	Файл был изменен, и при этом обновлен успешно.
`C`	Файл был изменен, и при объединении изменений возникли конфликты.

Объединение (merging) производится, если вы выгрузили рабочую копию какого-то модуля, изменили его, затем кто-либо еще произвел коммит собственных изменений, и, наконец, вы выполняете команду `cvs update`. CVS знает, что производились локальные изменения, и пытается объединить ваши изменения с теми, что произошли в репозитории (от состоянии версии, которую вы выгружали, до версии, до которой вы пытаетесь обновиться). Если изменения происходили с различными частями файла, объединение почти всегда произойдет успешно (хотя результат при этом может не быть синтаксически или семантически корректным).

CVS выводит букву `M` перед именем всех локально измененных файлов, даже если у них нет новых версий в репозитории, так что команда `cvs update` удобна для быстрого получения списка файлов, которые вы изменяли.

Если в результате вы видите букву `C`, ваши изменения конфликтуют с изменениями, внесенными в репозиторий (изменения были в одних и тех же или рядом расположенных строках, либо вы изменили файл настолько, что при сравнении `cvs` не смогла удержать контекст и приложить изменения из репозитория). Вам необходимо устранить конфликты, вручную редактируя файл. Конфликтующие фрагменты помечаются строками из знаков <, = и >. В начале каждого из конфликтов присутствует строка из семи знаков < и имени файла, затем идет фрагмент, содержащий внесенные вами изменения, разделитель из семи знаков =, соответствующий фрагмент из версии файла, содержащейся в репозитории, и, наконец, строка из семи знаков > совместно с номером версии, до которой вы обновляли файл.

Опция `-j` содержит некоторое количество черной магии. При ее наличии локальный файл обновляется до указанной версии так же, как и при использовании опции `-r`, но отслеживаемые номер версии или ветвь не изменяются. Эта опция умеет смысл лишь при парном использовании: при этом делается попытка применить изменения между двумя указанными версиями к локальной копии файла.

К примеру, вы внесли изменения и произвели коммит в файл `shazam/shazam.c` в FreeBSD-CURRENT, а позднее хотите перенести обновления в FreeBSD-STABLE (Merge-From-Current, MFC). Версия, которая требует переноса — 1.15:

- Извлеките текущую версию модуля `shazam` для ветви FreeBSD-STABLE:

  ```
  % cvs co --rRELENG_6 shazam
  ```

- Приложите изменения между версиями 1.14 и 1.15:

  ```
  % cvs update --j1.14 --j1.15 shazam/shazam.c
  ```

Почти наверняка вы получите конфликт в строках, содержащих идентификатор файла (`$Id: article.xml,v 1.19 2007-05-09 06:08:50 bvs Exp $` или, в случае FreeBSD, `$FreeBSD: head/ru_RU.KOI8-R/articles/committers-guide/article.xml 43184 2013-11-13 07:52:45Z hrs $`). Вам потребуется отредактировать файл для устранения конфликта (в данном случае достаточно убрать строки-разделители и вторую строку `$Id: article.xml,v 1.19 2007-05-09 06:08:50 bvs Exp $` , оставив лишь строку с `$Id: article.xml,v 1.19 2007-05-09 06:08:50 bvs Exp $` для FreeBSD-STABLE).

4. Просмотр изменение между локальной версией и версией из репозитория: команда `diff`.

```
% cvs diff shazam
```

Эта команда покажет все отличия локального состояния файла (или файлов модуля) `shazam` от состояния, сохраненного в репозитории.

Таблица 3. Полезные опции команды `cvs diff`

`-u`	Использовать унифицированный (unified) формат.
`-c`	Использовать контекстный (context) формат.
`-N`	Показывать отсутствующие или созданные файлы.

Всегда имеет смысл пользоваться опцией `-u`, поскольку унифицированный формат гораздо удобнее и лучше читаем, чем почти все другие (в некоторых случаях контекстный формат, генерируемый опцией `-c` может быть несколько лучше, но он гораздо более громоздок). Унифицированный формат различий состоит из серии фрагментов, каждый из которых начинается со строки, состоящей из двух символов `@` и номеров строк, описывающих положение изменившегося участка. Затем следует группа строк: те, что начинаются с пробела, описывают контекст, начинающиеся с символа `-` определяют удаленные строки, наконец, начинающиеся с символа `+` — добавленные.

Вы можете сравнивать текущее состояние с версией, отличающейся от той, с которой вы извлекали файл, указав опцию `-r` или `-D` подобно командам `checkout` и `update`, или даже получить список изменений между любыми двумя версиями (вне зависимости от того, что лежит в вашей локальной копии), указав *две* версии при помощи опций `-r` или `-D`.

5. Просмотр журнала изменений: команда `log`.

```
% cvs log shazam
```

Если `shazam` является обычным файлом, эта команда выдаст на экран *заголовок* с информацией о файле, в частности, его местоположении в репозитории, какая версия соответствует текущему состоянию (`HEAD`), в каких ветвях разработки файл присутствует, а также перечислит теги, которыми он помечен. Затем, для каждой версии файла выводится соответствующее ей журнальное сообщение, включающее дату, время и автора коммита, количество добавленных и удаленных строк и собственно журнального сообщения, написанного коммиттером.

Если `shazam` является каталогом, вышеописанная процедура выполняется для каждого файла в каталоге. Если при этом команде `log` не был указан флаг `-l`, процедура рекурсивно повторяется для всех подкаталогов.

Команда `log` используется для просмотра истории одного или нескольких файлов в том виде, как она сохранена в репозитории CVS. Используя опцию `-r`*rev*, вы можете посмотреть журнальное сообщение к одной определенной версии:

```
% cvs log --r1.2 shazam
```

Эта команда покажет журнальное сообщение для версии `1.2` файла `shazam` (или для версий `1.2` каждого из файлов в каталоге `shazam`).

6. Кто что делал: команда `annotate`. Эта команда показывает перед каждой строкой указанного файла (файлов) имя пользователя, вносившего последние изменения в эту строку.

   ```
   % cvs annotate shazam
   ```

7. Добавление новых файлов: команда `add`.

   Создайте файл, выполните для него команду `cvs add`, затем произведите запись в репозиторий (коммит): `cvs commit`.

   Точно так же, новые каталоги добавляются в репозиторий путем создания и последующего выполнения команды `cvs add`. Заметьте, что выполнять коммит после добавления каталога не надо.

8. Удаление устаревших файлов: команда `remove`.

   Удалите файл, затем выполните команду `cvs rm` с его именем в качестве параметра, наконец, выполните для него `cvs commit`.

9. Внесение изменений в репозиторий: команда `commit` или `checkin`.

   Таблица 4. Useful `cvs commit` options

`-f`	Форсировать внесение изменений для не модифицированного файла.
`-m`*msg*	Указать сообщение для журнала в командной строке (не запускать текстовый редактор).

   Опцию `-f` следует использовать, если вы поняли, что забыли указать какую-либо важную информацию в журнале изменений.

   Хорошие журнальные сообщения очень важны. Они дают возможность другим узнать, зачем вы производили изменения, причем не только в момент их произведения, но и месяцы или годы спустя, когда кто-либо заинтересуется, почему выглядящий нелогично или неэффективно фрагмент кода попал в каши исходные тексты. Кроме того, это очень помогает в оценке того, нужно ли переносить соответствующий код в FreeBSD-STABLE (MFC).

Сообщения должны быть ясными, краткими, четкими, и представлять из себя разумную аннотацию, какие изменения были произведены и почему.

Сообщения должны достаточно ясно показывать сторонним разработчикам, насколько их касаются изменения и нужно ли им исследовать изменения подробно.

Избегайте внесения нескольких не связанных друг с другом изменений за один раз. Это затрудняет объединение изменений, а также, при обнаружении ошибок, усложняет поиск ответственного за ошибки участка.

Избегайте смешивания в одном коммите изменений функциональности со стилистическими правками или исправлениями в пробелах. Это усложняет объединение, и, кроме того, затрудняет понимание того, какие именно функциональные изменения были внесены. В случае коммита в файлы документации, это затруднит работу групп поддержки перевода, поскольку становится сложнее отделить изменения, требующие перевода.

Избегайте коммита большой группы файлов за один раз с одним общим и невнятным сообщением. Напротив, вносите изменения в отдельные файлы (или небольшие группы связанных файлов) с адекватными сообщениями для журналирования.

Перед коммитом, *обязательно*:

- проверьте, что вы будете выполнять коммит в правильную ветвь, посредством команды `cvs status`.
- проверьте ваши изменения при помощи команды `cvs diff`

Кроме того, ВСЕГДА указывайте, в какие именно файлы вы вносите изменения, так чтобы не включить в этот список лишних файлов. Команда `cvs commit` без аргументов включит все измененные файлы в текущем каталоге и всех подкаталогах.

Еще несколько полезных советов:

1. Часто используемые опции можно занести в файл `~/.cvsrc`, например:

```
cvs --z3
diff --Nu
update --Pd
checkout --P
```

   Для данного случая:

- всегда использовать компрессию уровня 3 для связи с удаленным сервером CVS. В случае медленного соединения это избавит вас от лишней головной боли.

- всегда использовать опции `-N` (показывать добавленные или удаленные файлы) и `-u` (унифицированный формат) для diff(1).

- всегда использовать опции `-P` (удалять пустые каталоги) и `-d` (добавлять новые каталоги) при обновлении.

- всегда использовать опцию `-P` (удалять пустые каталоги) при извлечении файлов и модулей.

2. Пользуйтесь скриптом Эйвинда Эклунда (Eivind Eklund) `cdiff` для просмотра изменению унифицированного формата. Он является оберткой для less(1), добавляющей цветовые коды ANSI для выделения заголовком, добавленных и удаленных строк; прочие строки не модифицируются. Помимо этого, скрипт корректно разворачивает табуляции (которые часто выглядят неправильно в изменениях из-за дополнительного символа в начале строки).

   textproc/cdiff

   Просто используйте его вместо more(1) или less(1):

   ```
   % cvs diff --Nu shazam -| cdiff
   ```

   Помимо этого, некоторые текстовые редакторы, такие как vim(1) (editors/vim) поддерживают цветовую синтаксическую разметку многих типов файлов, в том числе файлов изменений и журналов CVS/RCS.

   ```
   % echo -"syn on" >> ~/.vimrc
   % cvs diff --Nu shazam -| vim --
   % cvs log shazam -| vim --
   ```

3. CVS — старая, загадочная и порой слабо предсказуемая в своем поведении программа. Ни один человек не способен удержать в голове все тонкости ее работы, так что не бойтесь спрашивать совета у Искусственного Интеллекта (а именно Администраторы Главного CVS Репозитория <`cvsadm@FreeBSD.org` >).

4. Не оставляйте компьютер в процессе работы команды `cvs commit` (в редакторе при написании журнального сообщения) слишком надолго (более чем на 2–3 минуты). Эта команда блокирует каталог репозитория, в котором она запущена, и не позволяет другим разработчикам изменять его содержимое. Если вам нужно написать длинное журнальное сообщение, подготовьте его заранее и вставьте в редакторе во время выполнения команды `cvs commit` , либо запишите его в файл и используйте опцию CVS `-F`:

```
% vi logmsg
% cvs ci -F logmsg shazam
```

Это самый быстрый способ передать журнальное сообщение CVS; однако, вы должны быть внимательны при редактировании файла `logmsg`, поскольку при выполнении коммита у вас не будет шансов его поправить.

5. Вы можете существенно ускорить скорость работы CVS с центральным репозиторием, используя постоянное соединение с репозиторием. Для этого добавьте в файл `~/.ssh/config` строки

```
Host ncvs.FreeBSD.org
      ControlPath -/home/user/.ssh/cvs.cpath
Host dcvs.FreeBSD.org
      ControlPath -/home/user/.ssh/cvs.cpath
Host projcvs.FreeBSD.org
      ControlPath -/home/user/.ssh/cvs.cpath
Host pcvs.FreeBSD.org
      ControlPath -/home/user/.ssh/cvs.cpath
```

Затем откройте постоянное соединение с машиной repoman:

```
% ssh --fNM ncvs.FreeBSD.org
```

Теперь команды CVS должны выполняться быстрее, поскольку используют существующее соединение с репозиторием. Учтите, что регистр в именах хостов имеет значение.

4. Соглашения и традиции

Став коммиттером, вы должны прежде всего произвести некоторые стандартные действия.

- Добавьте себя в список «SGML сущностей» авторов в файл `doc/en_US.ISO8859-1/share/xml/authors.ent` ; это изменение должно быть сделано прежде прочих, поскольку в противном случае следующий ваш коммит неизбежно разрушит процесс построения дерева doc/.

  Это довольно простая задача, но при этом она является неплохим первым тестом ваших навыков работы с CVS.

- Также добавьте свою «SGML сущность» в `www/en/developers.xml` .

- Добавьте себя в раздел «Разработчики» статьи Участники проекта FreeBSD (`doc/en_US.ISO8859-1/articles/contributors/contrib.committers.xml`) и удалите свою запись из раздела «Прочие участники» (`doc/en_US.ISO8859-1/articles/contributors/contrib.additional.xml`).

- Добавьте новость о новом коммиттере в файл `www/share/xml/news.xml` . Используйте существующие записи вида «Новый коммиттер» как шаблон.

- Вам нужно добавить ваш PGP или GnuPG ключ в каталог `doc/share/pgpkeys` (а если у вас нет ключа, вам нужно его создать). Не забудьте изменить и произвести коммит в файл `doc/share/pgpkeys/pgpkeys.ent` .

  Dag-Erling C. Smørgrav <`des@FreeBSD.org`> написал скрипт для упрощения этого процесса. Дополнительную информацию можно прочесть в файле README.

> **Примечание**
>
> Очень важно, чтобы в Руководстве пользователя был записан актуальный PGP/GnuPG ключ, поскольку он может потребоваться для идентификации коммиттера (например, его будет проверять группа Администраторы FreeBSD.org <`admins@FreeBSD.org`> для аварийного восстановления учетной записи). Полный набор актуальных ключей пользователей домена `FreeBSD.org` можно найти по адресу http://www.FreeBSD.org/doc/pgpkeyring.txt.

- Добавьте себя в файл `src/share/misc/committers-репозиторий.dot` , где репозиторием будет являться либо doc, либо ports, либо src в зависимости от полученных вами коммиттерских привилегий.

- Некоторые коммиттеры добавляют информацию о своем местоположении в файл `ports/astro/xearth/files/freebsd.committers.markers` .

- Некоторые добавляют данные о дне своего рождения в файл `src/usr.bin/calendar/calendars/calendar.freebsd` .

- Представьтесь другим коммиттерам, иначе никто не будет знать, кто вы и чем занимаетесь. От вас не требуется писать подробное резюме или биографию: будут достаточны один-два абзаца о себе и областях FreeBSD, в которых вы планируете работать. Пошлите это письмо в Список рассылки разработчиков FreeBSD — и все!

- Зайдите на машину `hub.FreeBSD.org` и создайте файл `/var/forward/user` (замените *`user`* на ваше имя пользователя). Этот файл должен содержать адрес электронной почты, на который будет переправляться вся почта на адрес *`yourusername`* @FreeBSD.org, в том числе сообщения о коммитах и другая почта на адреса Список рассылки коммиттеров FreeBSD и Список рассылки разработчиков FreeBSD. Слишком большие почтовые ящики на машине `hub` могут быть «нечаянно» удалены или обрезаны без предупреждения, так что, чтобы не потерять почту, регулярно читайте ее либо перенаправьте куда-нибудь еще.

Из-за ощутимой загрузки, возникающей на серверах, обрабатывающих списки рассылки, из-за большого количества незапрошенной почты (спама), сервер, принимающий почту для домена FreeBSD.org, производит некоторые основные проверки и на основании их отвергает некоторые письма. На настоящий момент единственным проверяемым параметром является корректность информации DNS для хоста, доставляющего почту, но в будущем список может вырасти. Эти проверки временами обвиняют в том, что они отвергают правильную почту. Если вы хотите отключить проверки для своего адреса, создайте файл `~/.spam_lover` в своей домашней директории на машине `freefall.FreeBSD.org` .

- Если вы были подписаны на Список рассылки сообщений об изменениях в главном дереве исходных текстов FreeBSD, вам, скорее всего, следует отписаться от него, чтобы не получать дубликатов каждого сообщения о коммитах.

Все новые коммиттеры первоначально работают под руководством ментора. Ваш ментор отвечает за обучение вас правилам и соглашениям, принятым в проекте, и помогает вам сделать первые шаги в среде коммиттеров. Он(а) также персонально отвечает за ваши действия в этот начальный период. До тех пор, пока ваш ментор не решит (и не анонсирует это посредством форсированного коммита файла `access`), что вы достаточно освоились и готовы работать самостоятельно, перед любым коммитом вы должны получить одобрение (approval) ментора и указать это в журнальном сообщении коммита строкой `Approved by:` .

Все коммиты в дерево `src` сначала должны производиться в ветвь FreeBSD-CURRENT и лишь затем интегрироваться в FreeBSD-STABLE. Никакие серьезные изменения, новые возможности или рискованные модификации не должны производиться напрямую в ветви FreeBSD-STABLE.

5. Предпочтительная лицензия для новых файлов

В настоящее время Проект FreeBSD считает предпочтительной формой лицензии на исходные тексты следующий текст:

```
/*-
* Copyright (c) [year] [your name]
* All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
*    notice, this list of conditions and the following disclaimer.
```

```
* 2. Redistributions in binary form must reproduce the above ↺
copyright
*    notice, this list of conditions and the following disclaimer ↺
in the
*    documentation and/or other materials provided with the ↺
distribution.
*
* THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ``AS ↺
IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, ↺
THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A ↺
PARTICULAR PURPOSE
* ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE ↺
LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR ↺
CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE ↺
GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS ↺
INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN ↺
CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ↺
ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE ↺
POSSIBILITY OF
* SUCH DAMAGE.
*
* [id for your version control system, if any]
*/
```

Проект FreeBSD крайне не рекомендует так называемый «третий пункт», или «пункт о рекламе» в лицензии на новый исходный код. В связи с большим количеством участников проекта FreeBSD, выполнение этого пункта большинством коммерческих производителей все более затруднительно. Если ваш код в дереве исходников содержит «пункт о рекламе», рассмотрите возможность его удаления. На самом деле, рассмотрите возможность перехода на приведенную лицензию.

Проект FreeBSD не рекомендует использование полностью новых лицензий или вариаций стандартных лицензий. Новые лицензии перед использованием в репозитории проекта требуют утверждения группой <core@FreeBSD.org >. Большое число различных лицензий затрудняет использование кода, в основном из-за ненамеренных неверных выводов из плохо сформулированных формулировок лицензии.

Политика проекта требует, чтобы код под НЕ-BSD лицензиями располагался только в определённых местах репозитория, а в некоторых случаях компиляция должна быть условной по умолчанию или вообще отключена. К примеру, ядро GENERIC должно состоять только из лицензий идентичных или в значительной степени схожих с BSD лицензией. Программное обеспечение под лицензиями GPL, APSL, CDDL и др. не должно включаться в состав GENERIC.

Разработчикам напоминается, что в open source правильное понимание "open" также важно, как правильное понимание "source", ибо некорректное использование интеллектуальной собственности имеет серьезные последствия. Какие-либо вопросы или беспокойства на этот счёт должны быть немедленно вынесены на обсуждение главной команде разработчиков (core team).

6. Взаимодействие между разработчиками

Если вы работаете над собственным исходным кодом, либо в области, в которой вы уже определены как ответственная персона, вам, скорее всего, не потребуется согласовывать коммит с кем-либо еще из разработчиков. Те же правила действуют, если вы нашли ошибку в той части системы, которой явно давно никто не занимается (к нашему стыду, существует несколько таких областей). Если же вы собираетесь модифицировать что-либо активно поддерживаемое (по-хорошему, узнать это можно только исследуя архивы списка рассылки `cvs-committers`), стоит послать предполагаемый патч ответственному за этот участок кода, как вы бы поступали, пока не были коммиттером. В случае портов нужно обращаться по адресу, указанному в строке `MAINTAINER` в файле `Makefile`. Для других частей репозитория, в случае если вам не очевидно, кто ведет данный участок кода, может помочь исследование вывода команды `cvs log`. Bill Fenner <`fenner@FreeBSD.org` > написал отличный скрипт для определения разработчиков, наиболее активно производивших коммиты, выводящий для каждого из указанных файлов имя пользователя вместе с количеством произведенных им коммитов в данный файл. Скрипт можно найти на машине `freefall` в файле `~fenner/bin/whodid` . Если найденная вами персона не отвечает на ваши вопросы или иным образом демонстрирует отсутствие интереса к проблеме, смело производите коммит самостоятельно.

Если вы по каким-либо причинам не уверены в своих изменениях, предложите их для оценки в списке рассылки `-hackers` перед коммитом. Будет лучше, если вас обругают там и тогда, чем когда предлагаемое изменение уже будет частью репозитория. Если случилось так, что ваш коммит встретил сопротивление, возможно, стоит его откатить (back out) до тех пор, пока не будет достигнут консенсус. Помните — с помощью CVS всегда можно вернуться к предыдущему состоянию.

Не принимайте в штыки мнения других разработчиков, с которыми вы не согласны. Если они предлагают иное решение проблемы чем вы, или даже иначе воспринимают проблему, это не значит, что они глупы, имеют сомнительное происхождение, хотят разрушить вашу работу, очернить ваше доброе имя, или развалить проект FreeBSD. Просто они смотрят на мир под иным углом. Различные взгляды — благо.

Будьте честны в спорах. Оценивайте свою позицию по заслугам, честно относитесь к ее слабым сторонам и будьте готовы принять другие точки зрения и пути решения. Будьте открыты.

Будьте терпимы, если вас поправляют. Все мы совершаем ошибки. Если вы ошиблись, извинитесь. Не обвиняйте ни себя, ни, тем более, других в ошибке. Не теряйте времени на смущение или упреки, просто исправьте ошибку и двигайтесь дальше.

Спрашивайте и просите о помощи. Предлагайте ваши изменения для рассмотрения коллегам и рассматривайте их изменения. Одним из преимуществ программного обеспечения с открытыми исходными текстами является открытость разработки. Если никто не будет исследовать чужой код, это преимущество исчезнет.

7. GNATS

Для отслеживания ошибок и запросов на изменения проект FreeBSD использует GNATS. Если вы исправили ошибку или внесли изменения, описанные в одном из сообщений об ошибках (PR), не забудьте закрыть это сообщение, используя команду `edit-pr pr-number` на машине `freefall`. Хорошо будет, если вы потратите немного времени на поиск и закрытие других PR по этой теме. Вы и сами можете пользоваться send-pr(1) для предложения изменений, которые, по вашему мнению, могут потребовать более подробного обсуждения с коллегами.

Более подробно о GNATS можно прочитать по адресам:

- `http://www.cs.utah.edu/csinfo/texinfo/gnats/gnats.html`
- http://www.FreeBSD.org/support.html
- send-pr(1)

Вы можете пользоваться локальной копией GNATS, поддерживая ее синхронность при помощи CVSup. При этом вы можете использовать команды GNATS локально, а также пользоваться другими интерфейсами, такими как `tkgnats`, что позволит вам работать с базой сообщений об ошибках без соединения с Интернет.

Процедура 1. Использование локальной копии GNATS

1. Если вы еще не поддерживаете зеркало дерева GNATS, добавьте в ваш `supfile` строку

   ```
   gnats release=current prefix=/usr
   ```

   Учтите, что эта строка должна предшествовать любым строкам, содержащим параметр «tag=», поскольку дерево GNATS не находится под управлением CVS и не имеет символьных меток.

После запуска cvsup в каталоге `/usr/gnats` будет создана копия дерева GNATS FreeBSD. Вы можете использовать файл *refuse* для копирования отдельных категорий. Например, если вас интересуют только сообщения категории `docs`, добавьте в файл `/usr/local/etc/cvsup/sup/refuse` [1] строку

```
gnats/[a-ce-z]*
```

Прочие примеры в этом разделе подразумевают, что вы синхронизируете только категорию `docs`.

2. Установите порт GNATS из `ports/databases/gnats` . После установки вы обнаружите различные служебные каталоги в дереве `$PREFIX/share/gnats` .

3. Создайте символьные ссылки на синхронизированные каталоги GNATS в служебный каталог GNATS:

```
# cd -/usr/local/share/gnats/gnats-db
# ln --s -/usr/gnats/docs
```

Проделайте эту операцию для всех синхронизируемых категорий.

4. Обновите служебный файл GNATS `categories` , расположенный в каталоге `$PREFIX/share/gnats/gnats-db/gnats-adm` :

```
# This category is mandatory
pending:Category for faulty PRs:gnats-admin:
#
# FreeBSD categories
#
docs:Documentation Bug:freebsd-doc:
```

5. Запустите `$PREFIX/libexec/gnats/gen-index` для создания индекса. Вывод этой команды должен быть перенаправлен в файл `$PREFIX/share/gnats/gnats-db/gnats-adm/index` . Эту операцию можно выполнять периодически при помощи cron(8) или запускать cvsup(1) из скрипта, который затем сгенерирует новый индекс:

```
# /usr/local/libexec/gnats/gen-index \
 > -/usr/local/share/gnats/gnats-db/gnats-adm/index
```

6. Протестируйте созданную конфигурацию запросом к базе данных GNATS. Следующая команда выведет список открытых сообщений об ошибках в категории `docs`:

```
# query-pr --c docs --s open
```

[1] Точный путь к файлу зависит от установок `*default base` в вашем файле `supfile`.

Другие интерфейсы, например, порт databases/tkgnats также должны работать.

7. Выберите PR и закройте его.

> **Примечание**
>
> Описанная процедура позволяет вам выбирать и просматривать сообщения об ошибках локально. Для редактирования или закрытия вам потребуется зайти на машину `freefall`.

8. Кто есть кто

Помимо мастеров репозитория, существует еще несколько участников и групп проекта FreeBSD, с которыми вам как коммиттеру может потребоваться общаться. Краткий и ни в коем случае не полный список приводится ниже.

John Baldwin <`jhb@FreeBSD.org`>
: Джон возглавляет проект SMPng и отвечает за архитектуру, дизайн и реализацию перехода на многонитевое ядро. Джон также является редактором статьи "Архитектура SMPng". Если вы работаете с тонкими блокировками многопроцессорного ядра, координируйте свою работу с Джоном.

Группа Менеджеров Дерева Документации FreeBSD <`doceng@FreeBSD.org`>
: doceng — группа, отвечающая за инфраструктуру построения документации, прием новых коммиттеров документации и актуальность информации относительно CVS на веб-сайте и FTP-сайте FreeBSD. Эта группа не разбирает конфликты. Большая часть обсуждений, связанных с документацией, происходит в Список рассылки Проекта Документации FreeBSD. Дополнительную информацию о деятельности группы можно найти в ее собственном документе. Коммиттеры, заинтересованные в обновлении документации, должны ознакомиться с Учебником по Проекту Документирования FreeBSD для новых участников.

Ruslan Ermilov <`ru@FreeBSD.org`>
: Руслан великолепно знает тонкости mdoc(7). Если вы пишете справочную страницу и нуждаетесь в совете по ее структуре или разметке, обратитесь к Руслану.

Bruce Evans <`bde@FreeBSD.org`>
: Брюс занимается общим стилем кода проекта. Если ваш коммит мог бы быть лучше оформлен, Брюс укажет вам на это. Радуйтесь, что такой человек вооб-

ще есть. Брюс также является знатоком различных стандартов, применимых к FreeBSD.

Murray Stokely <murray@FreeBSD.org >, Doug White <dwhite@FreeBSD.org >, Robert Watson <rwatson@FreeBSD.org >, Ken Smith <kensmith@FreeBSD.org >, Hiroki Sato <hrs@FreeBSD.org >, Maxime Henrion <mux@FreeBSD.org >, Bruce A. Mah <bmah@FreeBSD.org >

Таков состав группы Группа Выпуска Релизов FreeBSD <re@FreeBSD.org >. Эта группа отвечает за сроки и процесс выпуска релизов. В период заморозки кода, выпускающие инженеры принимают окончательные решения по поводу всех изменений системы в ветви, готовящейся к очередному релизу. Если вы хотите интегрировать какие-либо изменения из FreeBSD-CURRENT в FreeBSD-STABLE (какими бы они ни были в данный конкретный момент), вам предстоит общаться с этой группой.

Хироки, кроме того, ведет раздел документации к релизам (`src/release/doc/*`). Если ваши изменения стоят того, чтобы быть упомянутыми в информации о релизе, сообщите об этом Хироки. Еще лучше, если вы пошлете патч с предлагаемыми изменениями к документу.

Colin Percival <cperciva@FreeBSD.org >

Колин — FreeBSD Security Officer и отвечает за деятельность группы Группа Офицеров Безопасности <security-officer@FreeBSD.org >.

Garrett Wollman <wollman@FreeBSD.org >

Если вам нужен совет по поводу темных мест сетевой части ядра, или вы не уверены в планируемом изменении сетевой подсистемы, мудрым решением будет обратиться к Гаррету. Помимо того, он хорошо разбирается в различных стандартах, применимых к FreeBSD.

Список рассылки коммиттеров FreeBSD

cvs-committers — адрес, используемый CVS для посылки сообщений о коммитах. Вы *никогда* не должны посылать письма напрямую на этот адрес; следует лишь отвечать на него, когда вам нужно послать короткие комментарии, непосредственно относящиеся к коммиту.

Список рассылки разработчиков FreeBSD

Все коммиттеры подписаны на список рассылки -developers. Этот список создан для обсуждения вопросов, касающихся «сообщества» коммиттеров FreeBSD, таких как выборы Правления, анонсы и т.п.

Список рассылки разработчиков FreeBSD служит для только для использования FreeBSD коммиттерами. Коммиттеры должны иметь возможность публично обсуждать вещи, которые должны быть разрешены, перед тем, как они будут публично объявлены. Данные дискуссии не предназначены для широкой публики и могут нанести вред FreeBSD.

Все FreeBSD коммиттеры должны соблюдать авторские права оригинального автора или авторов писем из этого списка рассылки. Не публикуйте и не пересылайте сообщения из Список рассылки разработчиков FreeBSD вне подписчиков данного списка рассылки без согласия всех авторов.

Нарушители авторских прав будут удалены из списка подписчиков Список рассылки разработчиков FreeBSD, и будут приостановлены их коммиттерские привилегии. Повторяющиеся или вопиющие нарушения приведут к полному лишению коммиттерских прав.

Этот список *не* предназначен для обсуждения кода, и *не является* заменой списков Список рассылки, посвящённый архитектуре и внутреннему устройству FreeBSD или Список рассылки, посвящённый аудиту кода FreeBSD. На самом деле, такое его использование вредит проекту, поскольку открытые обсуждения вопросов, касающихся всего сообщества пользователей FreeBSD в закрытом списке недопустимы. И, наконец: *никогда, действительно никогда не пишите в Список рассылки разработчиков FreeBSD с копией в другой список рассылки FreeBSD*. Никогда не пишите в какой-либо другой список рассылки с копией в Список рассылки разработчиков FreeBSD. Подобные действия серьезно подрывают смысл существования данного списка рассылки.

9. SSH: быстрый старт

1. Если вы используете FreeBSD версии 4.0 или более позднюю, OpenSSH включен в базовую поставку системы. Для более ранних версий обновите и установите OpenSSH из порта security/openssh.

2. Для тех, кто не хочет набирать свой пароль каждый раз при использовании ssh(1) и использует для авторизации ключи RSA или DSA, удобной будет утилита ssh-agent(1). Если вы собираетесь использовать ее, убедитесь, что она запущена раньше прочих приложений. Пользователи X Window, например, обычно запускают ее из файлов `.xsession` или `.xinitrc`. Подробнее смотрите в справочной странице ssh-agent(1).

3. Создайте пару ключей при помощи ssh-keygen(1). Ключи появятся в каталоге `$HOME/.ssh/`.

4. Пошлите ваш публичный ключ (содержимое файла `$HOME/.ssh/id_dsa.pub` или `$HOME/.ssh/id_rsa.pub`) вашему будущему ментору, чтобы он мог быть помещен в файл `yourlogin` в каталоге `/c/ssh-keys/` на машине `freefall`.

Теперь вы можете пользоваться утилитой ssh-add(1) для авторизации один раз за сессию. Утилита запросит кодовую фразу для вашего секретного ключа и затем сохранит ее в агенте авторизации (ssh-agent(1)). Если вы хотите удалить сохраненный секретный ключ из агента, используйте команду `ssh-add -d`.

Для теста используйте команду типа `ssh freefall.FreeBSD.org ls /usr` .

За дополнительной информацией обращайтесь к security/openssh, ssh(1), ssh-add(1), ssh-agent(1), ssh-keygen(1) и scp(1).

10. Большой Список Правил Коммиттера FreeBSD

1. Уважайте других коммиттеров.
2. Уважайте других участников проекта.
3. Обсудите любые значимые изменения *до* коммита.
4. Уважайте существующих мейнтейнеров (указанных в поле `MAINTAINER` файлов `Makefile` или в файле `MAINTAINER` в корневом каталоге репозитория).
5. Любое спорное изменение необходимо откатить (back out) в ожидании решения, если того требует мейнтейнер. Вопросы безопасности могут перекрывать мнение мейнтейнера, если так решит Security Officer.
6. Изменения вносятся в ветвь FreeBSD-CURRENT до FreeBSD-STABLE, за исключением случаев, прямо разрешенных выпускающими инженерами или неприменимости изменения к FreeBSD-CURRENT. Любое нетривиальное и не срочное изменение должно быть выдержано в FreeBSD-CURRENT в течение по крайней мере 3 дней перед переносом, чтобы его могли адекватно протестировать. Выпускающие инженеры обладают той же властью в ветви FreeBSD-STABLE, что и мейнтейнеры (см. правило 5).
7. Не пререкайтесь с другими коммиттерами публично: это дурно выглядит. Если вам необходимо с чем-либо «категорически не согласиться», делайте это личной почтой.
8. Соблюдайте все периоды заморозки кода (core freeze), а также своевременно читайте списки рассылки `committers` и `developers` , чтобы быть в курсе расписания таких периодов.
9. Если вы сомневаетесь в какой-либо процедуре, сначала спросите!
10. Тестируйте свои изменения перед коммитом.
11. Не производите коммит в деревья `src/contrib` , `src/crypto` и `src/sys/contrib` без *прямого* разрешения (approval) соответствующего мейнтейнера(ов).

Невыполнение этих правил может служить основанием для приостановки или, в случае рецидивов, полного лишения коммиттерских прав. Члены Правления (Core) имеют право временно приостановить права коммиттера до момента, когда Правление в целом сможет решить вопрос. В «аварийном» случае (коммиттер разрушает репозиторий) такие права имеют также ответственные за репозиторий. Для приостановки прав коммиттера более чем на неделю или для полного лишения таковых прав требуется квалифицированное большинство (2/3) голосов Правления. Это правило существует не потому, что Правление состоит из злобных диктаторов, разбрасывающихся коммиттерами словно банками из-под колы, но ради предоставления проекту аварийного выключателя. Если кто-то выходит из-под контроля, важно иметь возможность справиться с ситуацией немедленно, а не быть втянутыми в дебаты. В любом случае, коммиттер, чьи права приостановлены, имеет право на «слушания Правления», на которых определяется срок приостановки или лишения коммиттерских прав. Коммиттер, права которого приостановлены может запросить пересмотр своего вопроса через 30 дней и каждые последующие 30 дней (если общий период приостановки превышает 30 дней). Коммиттер, полностью лишенный прав, может запросить пересмотр по истечении 6 месяцев. Правила пересмотра являются *полностью неформальными* и во всех случаях Правление имеет право отвергнуть запрос на пересмотр, если считает свое первоначальное решение верным.

Во всех прочих аспектах деятельности проекта, Правление является подмножеством коммиттеров и ограничено *теми же правилами*. Само по себе членство в Правлении не дает права преступать описанные правила. Правление обладает «особой силой» только в случае деятельность как целое, а не на индивидуальной основе. Члены Правления — в первую очередь коммиттеры.

10.1. Подробности

1. Уважайте других коммиттеров.

   Вы должны относиться к другим коммиттерам как к коллегам по разработке (кем они и являются). Несмотря на возникающие временами попытки утверждать обратное, никто не стал коммиттером по своей или чьей-либо еще глупости, и мало что обижает сильнее, чем подобные обвинения от коллег. Вне зависимости от того, всегда ли мы чувствуем уважение друг к другу или нет (у каждого бывают не лучшие дни), мы всегда должны *выказывать* уважение к другим коммиттерам, как в публичных форумах, так и в личной почте.

   Способность совместной работы в течение длительного времени — одно из главных достижений проекта, много более важное, чем любой набор изменений в коде, и никакие аргументы относительно кода не стоят потерь в возможности гармонично работать вместе.

   Чтобы не противоречить этому правилу, никогда не посылайте писем, когда вы злы или каким-либо иным образом можете спровоцировать других на конфронтацию. Сначала успокойтесь, затем подумайте о том, как наиболее эффективно убедить оппонента(ов) в правильности ваших аргументов; не подливайте масла в огонь ради краткого мига злорадства, если ценой будет долгая ругань. Это не просто крайне «энергетически неэффективно»: повторяющиеся прецеденты публичной агрессии, влияющие на нашу способность работать вместе, будут всерьез рассмотрены лидерами проекта, и могут привести к приостановке или потере прав коммиттера. Во внимание будут приниматься как публичные высказывания, так и личная переписка; это не означает, что Правление будет требовать раскрытия тайны переписки, однако, все предоставленные затронутыми коммиттерами материалы будут рассмотрены.

Описанные процедуры никого не могут порадовать даже в малом, однако «целостность проекта превыше всего». Никакой объем кода не стоит потери целостности.

2. Уважайте других участников проекта.

   Вы не всегда были коммиттером. В свое время вы были простым сторонним участником (contributor). Помните об этом все время. Помните, сколь важно было добиться внимания и помощи. Не забывайте, насколько ваше участие было важным для вас. Помните свои ощущения. Не препятствуйте другим участникам и не унижайте их. Относитесь к ним с уважением. Возможно, они наши будущие коммиттеры, и они настолько же важны для проекта, как и коммиттеры. Их вклад в проект настолько же ценен и важен, как и ваш. В конце концов, вам пришлось приложить немало усилий для проекта, чтобы стать коммиттером. Всегда помните об этом.

   Обдумайте первое правило 1 и применяйте его и к другим участникам проекта.

3. Обсудите любые значимые изменения *до* коммита.

   Репозиторий CVS — не место для анонса изменений или обсуждения их. Все изменения должны обсуждаться в списках рассылки, и лишь после достижения консенсуса вносится в репозиторий. Это не означает, что вы должны спрашивать разрешения на исправление очевидной синтаксической ошибки в коде или опечатки в странице справочника. Вы должны ощутить, когда предполагаемое изменение требует предварительного обсуждения и обратной связи. Как правило, никто не станет возражать против обширных изменений, если результат очевидно лучше предыдущего состояния, однако никто не любит, когда эти изменения *неожиданны*. Лучший способ убедиться, что вы на правильном пути — дать ваш код просмотреть кому-либо еще из коммиттеров.

   Если вы сомневаетесь, просите отзыва!

4. Уважайте существующих мейнтейнеров.

   Многие части кода FreeBSD не являются чьей-либо «собственностью»: ситуация, когда некто подпрыгнет и завопит, если вы внесете изменения в «его» код, редка;

однако, всегда стоит предварительно проверить. Одним из используемых соглашений было добавление строки MAINTAINER в файл `Makefile` пакета или части дерева, которая активно поддерживается одним или несколькими коммиттерами; см. также соответствующий раздел http://www.FreeBSD.org/doc/ru_RU.KOI8-R/books/developers-handbook/policies.html. В случае, если какой-то участок системы имеет несколько мейнтейнеров, изменение его одним из них должно быть одобрено по крайней мере одним из других. В случаях, когда «принадлежность» кода неясна, вы можете взглянуть на историю коммитов, чтобы понять, кто наиболее активно либо в последнее время работал в этой области.

Отдельные области FreeBSD попадают под контроль коммиттеров, занимающихся поддержкой целых категорий на пути эволюции FreeBSD, таких как локализация или сетевая подсистема. Для дополнительной информации смотрите http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributors/staff-who.html

5. Любое спорное изменение необходимо откатить в ожидании решения, если того требует мейнтейнер. Вопросы безопасности могут перекрывать мнение мейнтейнера, если так решит Security Officer.

   Это может быть нелегко, особенно в период конфликта (когда каждый участник уверен, что прав именно он). К счастью, CVS дает возможность, вместо того чтобы вести бушующую перебранку, просто откатить внесенные изменения, успокоиться всем участникам конфликта, а затем попробовать найти взаимоприемлемый путь. Если в конце концов окажется, что изменение стоит того, оно может быть легко применено вновь. В противном случае, пользователям не придется жить с неправильным состоянием дерева исходных текстов, пока стороны заняты выяснением отношений. Запросы на откаты возникают *крайне* редко, поскольку обсуждение обычно выявляет неверные или спорные моменты до коммита; однако, если такой запрос все же возник, он должен быть безусловно удовлетворен, чтобы мы могли спокойно выяснить, было изменение неверным или нет.

6. Изменения вносятся в ветвь FreeBSD-CURRENT до FreeBSD-STABLE, за исключением случаев, прямо разрешенных выпускающими инженерами или неприменимости изменения к FreeBSD-CURRENT. Любое нетривиальное и не срочное изменение должно быть выдержано в FreeBSD-CURRENT в течение по крайней мере 3 дней перед переносом, чтобы его могли адекватно протестировать. Выпускающие инженеры обладают той же властью в ветви FreeBSD-STABLE, что и мейнтейнеры (см. правило 5).

   Это еще одно «не обсуждаемое» правило: выпускающий инженер безусловно отвечает за последствия, если выясняется что внесенные изменения неверны. Уважайте эти права, и помогайте группе выпуска релизов в работе с ветвью FreeBSD-STABLE. На первый взгляд может показаться, что ветвь FreeBSD-STABLE развивается чересчур консервативно. Не забывайте, однако, что разумный консерватизм — отличительное свойство FreeBSD-STABLE, и что эта ветвь развивается по законам, отличным от законов FreeBSD-CURRENT. Кроме того, нет смысла тестировать изменения в FreeBSD-CURRENT, если они немедленно переносятся в FreeBSD-STABLE. Разработчики FreeBSD-CURRENT должны иметь возможность протестировать внесенные изменения, так что оставьте время для такого тестирования, если только речь не идет о критическом исправлении или о чем-либо очевидно не требующем тестирования (например, исправления опечаток в страницах справочника, очевидных ошибок или опечаток в исходных текстах и т.п.) Иными словами, исходите из соображений здравого смысла.

Изменения в ветви поддержки безопасности (security branches, например, `RELENG_6_0`) должны быть одобрены членом группы Группа Офицеров Безопасности <security-officer@FreeBSD.org > или, в некоторых случаях, одним из выпускающих инженеров (Группа Выпуска Релизов FreeBSD <re@FreeBSD.org >).

7. Не пререкайтесь с другими коммиттерами публично: это дурно выглядит. Если вам необходимо с чем-либо «категорически не согласиться», делайте это личной почтой.

   Для всех участников проекта очень важно поддержание его публичного образа; особенно это важно, если мы хотим продолжать привлекать новых участников. Случается, что, несмотря на все усилия по сохранению власти над собой, люди срываются и грубят друг другу. Лучшее, что здесь можно сделать — минимизировать эффект, пока все участники не успокоятся. Следовательно, вы не должны озвучивать свою ярость публично, а равно и пересылать частную переписку в общедоступные списки рассылки. Выражения, употребляющиеся в переписке один на один зачастую гораздо менее сдержанны, чем те, которые каждый участник употребил бы публично, так что подобной переписке нет места в публичных рассылках: это лишь усугубит и без того неприятную ситуацию. Если кто-либо, говорящий вам нелицеприятные слова, делает это в частной переписке, соблюдайте приличия и вы: отвечайте приватно. Если, по вашему мнению, кто-либо из разработчиков поступает с вами нечестно, и это настолько мучит вас, обратитесь к Правлению, а не выносите конфликт наружу. Правление приложит все силы к разрешению ситуации, выступая в роли третейского судьи. В случаях, когда спор затрагивает какие-либо части кода, и участники не могут прийти к взаимно приемлемому соглашению, Правление может привлечь независимого участника для разрешения вопроса. В этом случае все участники конфликта должны согласиться принять решение, вынесенное третьей стороной.

8. Соблюдайте все периоды заморозки кода (core freeze), а также своевременно читайте списки рассылки `committers` и `developers` , чтобы быть в курсе расписания таких периодов.

Внесение не одобренных изменений в период заморозки кода является довольно большой ошибкой. Коммиттеры должны быть в курсе событий, прежде чем внести 10 мегабайт изменений после долгого отсутствия. Нарушающие это правило будут подвергаться заморозке коммиттерского бита до прохождения курса в Счастливом Лагере Повышения Квалификации FreeBSD, который организован в Гренландии.

9. Если вы сомневаетесь в какой-либо процедуре, сначала спросите!

   Множество ошибок совершается, когда кто-либо совершает поспешные действия, думая, что поступает правильно. Делая что-либо впервые, вы, скорее всего, не знаете принятых мелочей и тонкостей, и лучше всего будет сначала спросить, не то вы имеете шансы выставить себя не в лучшем свете. Не стоит стыдиться спросить «Как, черт возьми, это надо делать?» Мы и так знаем, что вы умны: иначе вы не стали бы коммиттером.

10. Тестируйте свои изменения перед коммитом.

    Это правило может показаться очевидным. Впрочем, если бы оно действительно было очевидно для всех, мы не так часто сталкивались со случаями явного его нарушения. Если ваши изменения затрагивают ядро, убедитесь, что после него нормально собираются ядра GENERIC и LINT. Если вы изменяете другую часть исходного кода, убедитесь, что код собирается (успешно завершается make buildworld). Если вы изменяете код в какой-либо ветви, убедитесь, что вы тестируете его на машине, которая работает именно на этой ветви кода. Если ваши изменения могут затронуть другие архитектуры, проверьте его на всех поддерживаемых архитектурах. Список доступных ресурсов можно найти на странице `http://www.FreeBSD.org/internal/` . По мере расширения списка поддерживаемых платформ в кластер будут добавляться соответствующие машины для тестирования.

11. Не производите коммит в деревья `src/contrib` , `src/crypto` и `src/sys/contrib` без *прямого* разрешения (approval) соответствующего мейнтейнера(ов).

    Описанные деревья содержат исходный код сторонних производителей, который, как правило, импортируется в соответствующие ветви. Любой коммит, даже не выводящий файл из ветви производителя, может стать головной болью для ответственных за эту часть проекта разработчиков. Так что, если у вас нет *прямого* разрешения от мейнтейнера, *ничего* не делайте с этой частью репозитория (если, конечно, вы не поддерживаете этот код сами).

    Отметим, что только что сказанное вовсе не означает, что вы не должны пытаться улучшить упомянутый код, наоборот, этому будут только рады. Лучше всего, если вы передадите ваши исправления вендору. Если изменения специфичны для FreeBSD, обсудите вопрос с мейнтейнером, возможно, он посчитает разумным применить их локально. Тем не менее, что бы вы ни делали, *не* производите коммит сами!

Если вы хотите стать ответственным за «ничей» участок дерева исходников, свяжитесь с FreeBSD Core.

10.2. Правила работы с различными архитектурами

Начиная с версии 5.0 проект FreeBSD начал поддерживать несколько новых вычислительных архитектур, и более не является «i386™-центричным». Для упрощения поддержки FreeBSD на базе всех этих платформ Правлением было сформулировано следующее заявление:

> Основной 32-битной платформой разработки является i386; основная 64-битная платформа — Sparc64. Крупные изменения в дизайне (в том числе основные изменения в API и ABI) до попадания в репозиторий должны быть отлажены по крайней мере на одной 32 и одной 64-битной платформе, желательно на основных поддерживаемых платформах.

Платформы i386 и Sparc64 были выбраны по причине широкой распространенности доступности для разработчиков; кроме того, они представляют принципиально разные подходы к дизайну процессора и системы в целом: порядок байт в слове, организация регистров, реализация DMA, кэша, страничной адресации и т.д.

Процессор Alpha, конечно, является 64-битным, однако он представляет более традиционный дизайн и потому не может служить достаточно хорошей тестовой платформой для отработки тонкостей, с которыми разработчик может столкнуться на других 64-битных платформах. Платформа ia64 во многом сложна так же, как и Sparc64, однако ее доступность для разработчиков пока оставляет желать лучшего.

Мы будем переформулировать эти правила по мере того, как будут меняться цены и доступность 64-битных платформ.

Кроме того, разработчики должны быть в курсе наших правил классов поддержки (Tier Policy) различных аппаратных архитектур. Эти правила предназначены для общего описание процесса разработки, и потому отличаются от вышеописанных требований к возможностям и архитектурам. Правила классов поддержки на период выпуска релизов много жестче, чем ограничения на изменения в процессе разработки.

10.3. Другие рекомендации

Перед коммитом в области документации используйте какие-либо средства проверки орфографии. Для документов SGML, кроме того, при помощи команды `make lint` следует проверить корректность форматирования.

Для страниц справочника, при помощи утилиты из коллекции портов `manck` проверяйте корректность перекрестных ссылок и ссылок на файлы, а также наличие всех необходимых ссылок на синонимы (переменная `MLINK`).

Не смешивайте функциональные изменения со стилистическими (не изменяющими функциональных свойств кода). Такое смешивание затрудняет вычленение изменений при использовании команды `cvs diff` и, таким образом, может скрыть появившиеся ошибки. Не смешивайте в коммите в деревья `doc/` и `www/` изменения текста и переформатирование: это затрудняет работу переводчиков. Производите все стилистические изменения или переформатирования отдельными коммитами, и четко обозначайте их как таковые в журнальных сообщениях к коммиту.

10.4. Удаление возможностей

При необходимости удаления какой-либо функциональной возможности из утилит базовой системы следует использовать следующую схему действий:

1. В странице справочника и, возможно, в комментариях к релизу опция, утилита или интерфейс объявляются устаревающими и не рекомендованными к использованию (deprecated); их использование выводит предупреждение.

2. Опция, утилита или интерфейс сохраняются до очередного основного релиза (релиз X.0).

3. Опция, утилита или интерфейс удаляются, в том числе из документации: теперь они являются устаревшими. Как правило, об этом стоит упомянуть в комментариях к релизу.

11. Поддержка различных архитектур

FreeBSD является хорошо портируемой операционной системой и предназначена для работы на самых разнообразных аппаратных архитектурах. Важной частью процесса обеспечения гибкости в поддержке современных тенденций развития оборудования является деление кода на машинно-зависимый (Machine Dependent, MD) и машинно-независимый (Machine Independent, MI), а также, по возможности, минимизация машинно-зависимой части кода. Каждая новая аппаратная архитектура, которую начинает поддерживать FreeBSD, ощутимо увеличивает работу по поддержке кода, инструментария и процесса выпуска релизов. Кроме того, становится значительно сложнее эффективно тестировать изменения в коде ядра. Все это делает необходимым введение различных классов поддержки для различных архитектур, при сохранении максимальной стабильности малого числа "основных платформ".

11.1. Основные намерения

Проект FreeBSD предназначен для работы на рабочих станциях, серверах и высокопроизводительных встроенных системах. Сохраняя ориентир на малое количество архитектур в интересах таких систем, проект FreeBSD остается способен поддерживать высокий уровень надежности, стабильности и производительности, а также уменьшить нагрузку на различные группы поддержки проекта, такие как группы поддержки портов, документации, безопасности и выпуска релизов. Разнообразие поддерживаемых аппаратных платформ расширяет область применимости FreeBSD за счет поддержки новых возможностей (например, поддержка 64-битных процессоров, использование во встроенных системах и т.п.); тем не менее, расширение этого списка всегда должно быть тщательно оценено с позиций увеличения затрат на поддержку дополнительной аппаратной платформы.

Проект FreeBSD делит различные аппаратные платформы на 4 класса. Для каждого класса описывается набор требований, необходимых для присвоения платформе данного класса, и обязательства разработчиков по отношению к платформе. Кроме того, определяется порядок смены класса для архитектуры.

11.2. Класс 1: Полностью поддерживаемые архитектуры

Платформы 1 класса полностью поддерживаются группой безопасности, группой выпуска релизов и мейнтейнерами инструментария. Новые возможности, добавляемые в код системы, должны быть полностью функциональны для всех архитектур первого класса для каждого из релизов (исключением могут быть архитектурно-зависимые возможности, такие как драйвера аппаратуры). Как правило, все платформы 1 класса должны поддерживаться системами сборки либо расположенными непосредственно в кластере FreeBSD.org, либо легко доступными для всех разработчиков.

Архитектуры первого класса должны быть готовыми к эксплуатации под управлением FreeBSD во всех аспектах, включая процесс установки и среду разработки.

В настоящее время платформами 1 класса являются i386, Sparc64, AMD64, and PC98.

11.3. Класс 2: Архитектуры для разработчиков

Платформы 2 класса не поддерживаются группами безопасности и выпуска релизов. Поддержка инструментария оставляется на усмотрение его мейнтейнеров. Новые возможности, реализуемые в FreeBSD, должны быть реализуемы на этих платформах, однако непосредственная реализация на момент добавления кода в дерево исходных текстов не требуется. Реализация порта на архитектуру 2 класса может быть добавлена в репозиторий, если она не входит в противоречие с текущим состоянием систем первого класса и не влияет в существенной степени на прочие платформы 2 класса. Для добавления архитектуры 2 класса в дерево исходных текстов FreeBSD система должна быть способна загрузиться хотя бы в однопользовательский режим на реальной аппаратуре. Некоторые исключения из последнего правила могут быть сделаны для новой аппаратуры, находящейся в состоянии разработки и временно не доступной для проекта.

Обычно архитектурами 2 класса являются те, которые планируются к переходу в 1 класс, но пока находятся в состоянии разработки. Также во втором классе могут находится платформы, перешедшие из 1 класса по причине потери актуальности, по мере того как уменьшается количество ресурсов, доступных для поддержки системы в состоянии готовности к промышленной эксплуатации.

В настоящее время платформами 2 класса являются PowerPC и ia64.

11.4. Класс 3: Экспериментальные архитектуры

Платформы 3 класса не поддерживаются группами безопасности и выпуска релизов. Поддержка инструментария оставляется на усмотрение его мейнтейнеров. Архитектурами третьего класса могут быть: те, для которых нет и в ближайшее время не предвидится доступного проекту оборудования; имеющие менее трех активных разработчиков; не способные загрузиться в однопользовательский режим на реальной аппаратуре (или под управлением эмулятора, если реальная аппаратура недоступна); наконец, системы, которые оцениваются как исчезающие, чья дальнейшая распространенность сомнительна. Поддержка систем 3 класса не вносится в основное дерево исходных текстов FreeBSD, однако работа над такими архитектурами может производиться в репозитории Perforce FreeBSD, для облегчения контроля версий и дальнейшей интеграции с основной массой кода.

В настоящее время единственной платформой 3 класса является S/390®.

11.5. Класс 4: не поддерживаемые архитектуры

Системы 4 класса никак не поддерживаются проектом.

К 4 классу относятся все архитектуры, не перечисленные выше.

11.6. Правила смены класса для архитектуры

Для переноса платформы из класса в класс требуется решение, утвержденное Правлением, которое, в свою очередь, согласует его с группами безопасности, выпуска релизов и поддержки инструментария.

12. FAQ по работе с портами

12.1. Добавление нового порта

Вопрос: Как добавить новый порт?

Ответ: Для начала прочитайте раздел, посвященный репозиторному копированию.

Самым простым будет использовать скрипт `addport` на машине `freefall`. Он добавит порт из указанного вами каталоге, определив нужную категорию из файла `Makefile`, добавит строку в файл `CVSROOT/modules` и в файл `Makefile` для нужной категории. Скрипт был написан Michael Haro <mharo@FreeBSD.org> и Will Andrews <will@FreeBSD.org>; вопросы и исправления по поводу `addport` следует отправлять Уиллу, как текущему мейнтейнеру.

Вопрос: Что еще следует сделать, добавляя новый порт?

Ответ: Проверьте его. Желательно убедиться в том, что порт и соответствующий пакет корректно собираются. Рекомендуемая последовательность действий такова:

```
# make install
# make package
# make deinstall
# pkg_add имя собранного пакета
# make deinstall
# make reinstall
# make package
```

Более подробные инструкции можно найти в Руководстве FreeBSD по созданию портов.

Пользуйтесь portlint(1) для проверки корректности порта. Не обязательно добиваться полного отсутствия предупреждений, но по крайней мере исправьте простейшие из них.

Если новый порт прислал человек, еще не упомянутый в Списке прочих участников, добавьте его имя туда.

Закройте PR, если новый порт пришел в виде PR. Для этого воспользуйтесь командой **`edit-pr PR#`** на машине `freefall` и измените значение в строке `state` с `open` на `closed`. Затем опишите причину смены статуса, и на этом работа закончена.

12.2. Удаление порта

Вопрос: Как удалить существующий порт?

Ответ: Для начала прочтите раздел о репозиторном копировании. Прежде чем удалить порт, вы должны проверить, что удаление не затронет другие порты коллекции.

- Убедитесь, что другие порты не зависят от удаляемого:
  - Имя пакета (PKGNAME) должно встречаться в свежем файле INDEX ровно один раз.
  - В файлах Makefile* других портов не должно встречаться ни одной ссылки на каталог удаляемого порта или имя его пакета (PKGNAME).
- Удалите порт:
  1. Удалите файлы порта командой `cvs remove` .
  2. Удалите строку `SUBDIR` для удаляемого порта из файла `Makefile` категории.
  3. Удалите запись для порта из файла модулей `CVSROOT/modules` .
  4. Добавьте соответствующую строку в файл `ports/MOVED` .
  5. Если порт упоминается в файле `ports/LEGAL` , удалите его оттуда.

Вы можете воспользоваться скриптом `rmport` из каталога `ports/Tools/scripts` . Этот скрипт написал Vasil Dimov <vd@FreeBSD.org >, и он же его поддерживает, так что вопросы, исправления и замечания по поводу `rmport` следует посылать непосредственно ему.

12.3. Репозиторное копирование

Вопрос: Когда требуется репозиторное копирование?

Ответ: При необходимости добавления порта, имеющего отношение к другому, уже находящемуся в репозитории в другом каталоге, необходимо произвести репозиторное копирование. В данном случае *имеющий отношение* означает другую версию или небольшую модификацию. Примерами могут служить различные версии `print/ghostscript*` и английская и локализованные версии `x11-wm/windowmaker*` .

Другим примером является необходимость перенести порт из одного подкаталога в другой, или переименовать каталог, когда автор меняет имя своей программы.

Вопрос: Когда репозиторное копирование *не* требуется?

Ответ: Если нет истории, которую стоило бы сохранять. Для порта, добавленного в неправильную категорию и сразу же перемещенного, будет вполне достаточно выполнить команды `cvs remove` для старого варианта и `addport` для нового.

Вопрос: Что нужно делать?

Ответ: Создайте в GNATS PR, описав причины репозиторного копирования. Поменяйте ответственного на `portmgr` и установите статус (`state`) в состояние `repocopy`. Если ваш запрос будет одобрен группой Группа Менеджеров Дерева Портов FreeBSD <`portmgr@FreeBSD.org`>, он будет переадресован на `pcvs`. Группа Менеджеров Дерева Портов FreeBSD <`portmgr@FreeBSD.org`> может произвести копирование каталогов самостоятельно; в противном случае группа Администраторы Репозитория Портов<`pcvs@FreeBSD.org`> произведет собственно копирование и вернет вам ваш PR. После этого, вам необходимо проделать следующее:

- После репозиторного копирования порта:

    1. Обновите новый вариант порта до новой версии. Не забудьте изменить строку `LATEST_LINK`, чтобы не получить двух портов с одним именем. В некоторых исключительных случаях может быть необходимо изменить переменную `PORTNAME` вместо `LATEST_LINK`, но это должно быть сделано только тогда когда это действительно нужно. Например, при использовании существующего порта в качестве основы для весьма похожей программы с другим именем или при обновлении порта до новой основной версии программы, при котором изменяется имя самого дистрибутива, как в случае перехода с `textproc/libxml` на `textproc/libxml2`. В большинстве случаев изменение `LATEST_LINK` должно быть достаточно.

    2. Добавьте новый каталог в список `SUBDIR` в родительском файле `Makefile`. Для проверки вы можете воспользоваться командой `make checksubdirs`.

    3. Если порт менял категорию, измените строку `CATEGORIES` в файле `Makefile`.

    4. Добавьте строку для нового модуля в `CVSROOT/modules`.

    5. Добавьте строку в файл `ports/MOVED`, в случае если вы удалили первоначальный порт.

- При удалении порта:

1. Тщательно проверьте коллекцию на предмет портов, зависящих от удаляемого и обновите их при необходимости. Выполнение команды `grep` по содержимому файла `INDEX` недостаточно, поскольку некоторые порты могут быть сконфигурированы на этапе сборки. Рекомендуется использовать полный поиск при помощи команды `grep -r`.

2. Удалите старый порт, запись `SUBDIR` и строку, описывающую модуль.

3. Добавьте строку в файл `ports/MOVED` .

- После репозиторного перемещения (операции «переименования», когда после копирования старый вариант удаляется):

  - Используйте процедуры из предыдущих двух пунктов для активации нового порта и удаления старого.

12.4. Заморозка портов

Вопрос: Что такое «заморозка портов»?

Ответ: Перед выпуском релиза для сохранения целостности различных частей системы требуется на некоторое время ограничить коммиты в дерево портов. Этот процесс и называется «заморозкой портов».

За дополнительной информацией по поводу правил поведения во время заморозки обращайтесь к документу Задачи контроля качества для Группы управления портами.

Вопрос: Сколько длится заморозка?

Ответ: Обычно неделю или две.

Вопрос: Что это значит для меня?

Ответ: Во время заморозки вы не можете производить какие-либо коммиты в дерево портов без прямого разрешения группы порт-менеджеров. «Прямое разрешение» здесь означает, что вы послали свой патч группе порт-менеджеров и получили ответ «Вперед, производите коммит».

В период заморозки не все изменения могут быть внесены в дерево. За подробностями обращайтесь к документу Задачи контроля качества для Группы управления портами.

Отметим, что у вас нет подразумеваемого разрешения исправлять неработающий порт в период заморозки только потому, что порт не работает.

Вопрос: Откуда я узнаю о начале периода заморозки?

Ответ: Обычно за 2-3 недели до начала периода заморозки кто-либо из группы порт-менеджеров посылает письмо с предупреждением об этом в Список рассылки, посвящённый Портам FreeBSD и Список рассылки коммиттеров FreeBSD. Точное время начала периода заморозки определяется за несколько дней до собственно релиза, поскольку фиксируемое дерево портов должно быть синхронизировано с релизом, а точная дата выпуска определяется по ходу дела.

Разумеется, после начала периода заморозки в Список рассылки коммиттеров FreeBSD будет отправлено еще одно предупреждение.

Вопрос: Как узнать, когда период заморозки портов закончился?

Ответ: Завершение периода заморозки анонсируется группой порт-менеджеров посылкой письма в Список рассылки, посвящённый Портам FreeBSD и Список рассылки коммиттеров FreeBSD через несколько часов после релиза. Отметим, что факт выпуска релиза не означает автоматического завершения заморозки. Нам потребуется убедиться, что в последние минуты не произошло ничего непредвиденного, что заставило бы перевыпускать релиз.

12.5. Создание новой категории

Вопрос: Какова процедура создания новой категории портов?

Ответ: Разработчик, предлагающий новую категорию, должен подготовить детальное обоснование ее создания, в том числе описание причин, по которым текущий список категорий недостаточен, а также список портов, переносимых в новую категорию.

Прежде чем отправлять запрос, помните, что процесс потребует приложения немалых сил от многих участников, затронет всякого, кто поддерживает актуальное состояние дерева портов целиком, и, наконец, что подобные предложения неизбежно вызовут споры и расхождения во мнениях.

Обратитесь к разделу Proposing a New Category Руководства по созданию портов. После передачи PR группе Группа Менеджеров Дерева Портов FreeBSD <`portmgr@FreeBSD.org`> решение о создании категории остается за ней. В случае утверждения новой категории кто-либо из Группа Менеджеров Дерева Портов FreeBSD <`portmgr@FreeBSD.org`> делает следующее:

1. Производит нужные репозиторные копирования.

2. Обновляет определения `VALID_CATEGORIES` в файле `ports/Mk/bsd.port.mk` .

3. Возвращает PR вам.

Вопрос: Как устроен процесс?

Ответ: Процедура является надстройкой над уже описанной процедурой репозиторного копирования отдельного порта.

1. Обновите файлы `Makefile` для всех перенесенных портов. Пока не добавляйте новую категорию в процесс построения индекса.

   Для этого вам необходимо:

   1. Сменить для всех портов значение переменной `CATEGORIES` (это и было нашей целью, не правда ли?) Новая категория должна быть указано в списке *первой*, это поможет проверить, правильно ли установлена переменная `PKGORIGIN` .

   2. Выполните команду `make describe` . Поскольку процедура построения главного индекса `make index` , которую вам предстоит выполнить несколько позже, использует именно `make describe` , обнаружение ошибок сейчас сэкономит вам немало времени в будущем.

   3. Если вы хотите быть совсем честным, самое время запустить portlint(1).

2. Проверьте корректность переменных `PKGORIGIN` . Система работы с портами использует значение переменной `CATEGORIES` для установки переменной `PKGORIGIN` , которая затем используется для связи установленных пакетов с каталогами дерева портов. Если эта связь установлена неправильно, перестанут правильно функционировать утилиты работы с портами, такие как pkg_version(1) и portupgrade(1).

   Для проверки следует использовать скрипт `chkorigin.sh` : `env PORTSDIR=/path/to/ports sh -e /path/to/ports/Tools/scripts/chkorigin.sh` . Эта команда проверит *каждый* порт в дереве, в том числе и те, что не включены в процесс сборки, так что ее можно использовать сразу после репозиторного копирования. Совет: не забудьте проверить `PKGORIGIN` для зависимых от изменяемых вами портов!

3. Протестируйте изменения локально, на вашей машине: закомментируйте строки `SUBDIR` для старых портов, затем разрешите обработку новой категории в файле `ports/Makefile` . Запустите `make checksubdirs` в затрагиваемых категориях. Наконец, выполните в каталоге `ports/` команду `make index` . Ее выполнение может занять до 40 минут даже на современной машине, однако, это необходимые затраты для того, чтобы не создать проблем для других.

4. После завершения этой операции вы можете вносить в репозиторий изменения `ports/Makefile` для включения новой категории в процесс сборки, а также производить коммит изменений `Makefile` для старых категорий.

5. Добавьте в файл `CVSROOT-ports/modules` строку

   ```
   ports_categoryname      categoryname
   ```

   Поля должны быть разделены табуляцией.

   Если *categoryname* содержит дефисы, замените их на подчеркивания.

6. Поменяйте строки для затронутых модулей в файле `CVSROOT-ports/modules` .

7. Добавьте нужные строки в файл `ports/MOVED` .

8. Обновите инструкции для cvsup(1):

   - Добавьте категорию в файл `distrib/cvsup/sup/README`

   - Добавьте в каталог `distrib/cvsup/sup/ports-categoryname` два файла: `list.cvs` и `releases` .

   - Добавьте категорию в файл `src/share/examples/cvsup/ports-supfile`

   (Обратите внимание: эти файлы расположены в репозитории src, а не ports). Если вы не являетесь коммиттером src, вам потребуется создать PR.

9. Обновите список категорий, используемый в sysinstall(8) в `src/usr.sbin/sysinstall` .

10. Обновите документацию:

    - Руководство FreeBSD по созданию портов

- Файл `www/en/ports/categories`. Обратите внимание, что строки в них сгруппированы по категориям, описанным в файле `www/en/ports/categories.descriptions`.

- Раздел Руководства, перечисляющий cvsup коллекции.

  (Внимание: все эти файлы находятся в репозитории документации. Если вы не являетесь коммиттером в этой области, создайте PR в категории документации (doc).

11. Старые варианты портов могут быть удалены из репозитория только после того, как все описанные процедуры будут завершены, и никто не жалуется на новую структуру.

Специально обновлять веб-страницу портов при добавлении новой категории не нужно: изменение файла `www/en/ports/categories` будет учтено при ежедневной перестройке списка портов (`INDEX`) автоматически.

12.6. Прочие вопросы

Вопрос: Как мне проверить, что мой порт корректно собирается?

Ответ: В первую очередь проверьте свой порт по адресу `http://pointyhat.FreeBSD.org/errorlogs/`. Там вы найдете журналы сборки пакетов на всех поддерживаемых архитектурах для большинства последних ветвей разработки.

Впрочем, отсутствие вашего порта среди журналов с ошибками еще не значит, что он успешно собирается (например, может не собираться один из зависимых портов). Необходимую информацию вы можете найти на машине `pointyhat` в каталогах `/a/portbuild/<arch>/<major_version>`. Каждая пара архитектуры и базовой версии содержит следующие подкаталоги:

```
errors        журналы ошибок последней сборки версии ↺
<major_version> на платформе <arch>
logs          все журналы последней сборки версии ↺
<major_version> на платформе <arch>
packages      свежесобранные пакеты для версии <major_version> ↺
на платформе <arch>
bak/errors    журналы ошибок последней полной сборки версии ↺
<major_version> на платформе <arch>
bak/logs      все журналы последней полной сборки версии ↺
<major_version> на платформе <arch>
bak/packages  пакеты последней полной сборки версии ↺
<major_version> на платформе <arch>
```

Общее правило: пакет, присутствующий в каталоге `packages` или каталоге `logs`, и при этом отсутствующий в `errors`, собрался успешно. (Именно каталоги `errors` вы видите на веб-сервере `pointyhat`).

Вопрос: Я добавил новый порт. Нужно ли добавлять его в файл `INDEX`?

Ответ: Нет. `INDEX` больше не хранится в CVS репозитории. Данный файл может быть сгенерирован с помощью команды `make index` или уже сгенерированная версия может быть загружена с помощью `make fetchindex`.

Вопрос: Какие еще файлы я не должен трогать?

Ответ: Любой файл в на верхнем уровне `ports/`, а также все файлы в каталогах, имена которых начинаются с прописной буквы (например, `Mk/`, `Tools/` и т.п.). В частности, упаси вас Бог трогать файлы `ports/Mk/bsd.port*.mk`, если вы не хотите привести порт-менеджеров в ярость!

Вопрос: Каков корректный порядок обновления порта, когда его исходный архив поменялся, но не сменил имя?

Ответ: При возникновении ситуации, когда автор обновляет дистрибутивный архив без изменения идентификатора версии, сообщение о коммите должно содержать аннотацию различий между предыдущим и обновленным состоянием архива, чтобы можно было убедиться, что архив не испорчен и не подменен злоумышленником. Если текущая версия порта существовала достаточное время, копии архива будут доступны на ftp-серверах проекта; в противном случае следует связаться с автором или мейнтейнером порта для выяснения причин замены архива.

13. Пряники и прочие льготы

Увы, льгот, возникающих от того, что вы являетесь коммиттером, не так уж много. Пожалуй, единственным несомненным долговременным преимуществом будет признание вас как компетентного специалиста. Тем не менее, кое-какие льготы все же существуют:

Прямой доступ к машине `cvsup-master`

Будучи коммиттером, вы можете обратиться к Jun Kuriyama <kuriyama@FreeBSD.org>, чтобы получить доступ к машине `cvsup-master.FreeBSD.org`, приложив вывод команды `cvpasswd yourusername@FreeBSD.org freefall.FreeBSD.org`. Обратите внимание: в командной строке вы должны указать `freefall.FreeBSD.org`, хотя реальным сервером будет `cvsup-master`. Доступом к `cvsup-master` не следует злоупотреблять: это весьма загруженная машина.

Бесплатная подписка на комплект из 4 CD или DVD
: Компания FreeBSD Mall, Inc. предоставляет для всех коммиттеров FreeBSD возможность бесплатной подписки на выпуски FreeBSD. Порядок подписки появляется в списке рассылки <developers@FreeBSD.org> после каждого релиза.

14. Прочие вопросы

Вопрос: Почему не следует вносить малозначимые изменения в ветви разработчика (vendor branches)?

Ответ:

- После этого действия каждый новый релиз от разработчика требует ручного приложения и объединения патчей.
- Что хуже, каждый новый релиз от разработчика требует ручной *проверки* приложенных патчей.
- Опция CVS `-j` не всегда хорошо работает. Можете спросить David O'Brien <obrien@FreeBSD.org>, он расскажет вам жутких историй.

Вопрос: Как мне добавить файл в ветвь CVS?

Ответ: Для добавления файла в ветви просто обновите исходные файлы до нужной ветви, а затем используйте команду `cvs add`. Например, если мы хотите перенести файл `src/sys/alpha/include/smp.h` из ветви HEAD в ветвь RELENG_6, в которой он пока не существует, можно использовать следующую последовательность действий:

Пример 1. MFC для нового файла

```
% cd sys/alpha/include
% cvs update --rRELENG_6
cvs update: Updating -.
U clockvar.h
U console.h
...
% cvs update --kk --Ap smp.h > smp.h
===================================================================
Checking out smp.h
RCS:  -/usr/cvs/src/sys/alpha/include/smp.h,v
VERS: 1.1
***************
```

```
% cvs add smp.h
cvs add: scheduling file `smp.h' for addition on branch ↺
`RELENG_6'
cvs add: use -'cvs commit' to add this file permanently
% cvs commit
```

Вопрос: Какую «мета-информацию» я должен включать в сообщения для коммита?

Ответ: Помимо информативного описания содержания коммита вам может потребоваться включить в сообщение дополнительную информацию.

Она состоит из одной или нескольких строк вида: ключевое слово или словосочетание, двоеточие, табуляции для форматирования, собственно дополнительная информация.

Ключевыми словами могут быть:

`PR:`	Идентификатор сообщения об ошибке, затрагиваемого (как правило, закрываемого) данным коммитом.
`Submitted by:`	Имя и e-mail адрес приславшего исправление; для коммиттеров — просто имя пользователя в кластере FreeBSD.
`Reviewed by:`	Имя и e-mail адрес того или тех, кто рецензировал изменения; для коммиттеров — имя пользователя в кластере FreeBSD. Если изменения были посланы в список рассылки на рецензию и получили одобрение, имя списка рассылки.
`Approved by:`	Имя и e-mail адрес того или тех, кто одобрил изменение; как и прежде, для коммиттеров просто имя пользователя в кластере. Обычной практикой является получение одобрения для коммитов в новые для вас области дерева. Кроме того, в период перед каждым релизом все коммиты *должны* быть одобрены группой выпускающих инженеров. В случае ваших первых коммитов вы должны получить одобрение на них у вашего мен-

	тора, и упомянуть его в виде «*username-of-mentor* (mentor)».
`Obtained from:`	Имя проекта, из исходного кода которого было взято изменение.
`MFC after:`	Если вы хотите получать по почте напоминания об MFC, укажите число дней, недель или месяцев с момента изначального коммита, через которое вы планируете произвести MFC.
`Security:`	Если ваши изменения затрагивают вопросы безопасности или исправляют какие-либо уязвимости, укажите ссылки на опубликованные отчеты или описание проблемы.

Пример 2. Сообщение для коммита, основанного на PR

Вы собираетесь внести коммит, основанный на PR, присланном John Smith и содержащим патч для исправления проблемы. Ваше сообщение должно заканчиваться примерно такими строками:

```
...

PR:                   foo/12345
Submitted by:     John Smith <John.Smith@example.com>
```

Пример 3. Сообщение для коммита, требующего рецензии

Вы собираетесь изменить подсистему работы с виртуальной памятью. Вы опубликовали предполагаемые изменения в соответствующем списке рассылки (в данном случае `freebsd-arch`), и изменения были одобрены.

```
...
```

```
Reviewed by:       --arch
```

Пример 4. Сообщение для коммита, требующего одобрения

Вы намерены произвести коммит в область дерева, для которой определен ведущий (MAINTAINER). Вы скоординировали усилия с мейнтейнером, и он отреагировал «Отлично. Производи коммит.»

```
...

Approved by:     abc
```

Где *abc* имя пользователя, одобрившего ваш коммит.

Пример 5. Сообщение для коммита, использующего код OpenBSD

Вы собираетесь внести изменение, основанное на коде, использованном проектом OpenBSD.

```
...

Obtained from:      OpenBSD
```

Пример 6. Сообщение для коммита, планирующего интеграцию из FreeBSD-CURRENT в FreeBSD-STABLE через некоторое время

Вы хотите внести изменения, которые должны быть интегрированы из FreeBSD-CURRENT в ветвь FreeBSD-STABLE через две недели.

```
...
```

```
MFC after:      2 weeks
```

Где *2* является количеством дней, недель или месяцев, через которое вы планируете интегрировать (MFC) в FreeBSD-STABLE. В качестве *weeks* может быть использовано week, weeks, month, months, либо этот параметр может быть опущен (при этом подразумевается *X* дней).

В отдельных случаях вам потребуется комбинировать приведенные примеры.

Рассмотрим ситуацию, когда некто прислал сообщение об ошибке, содержащее код из проекта NetBSD. Вы заинтересовались этим случаем, но он расположен в той части дерева, в которой вы обычно не работаете, так что вы решаете выдать изменения на рассмотрение списка рассылки arch. Поскольку изменения были достаточно сложны, вы решаете интегрировать их (MFC) через месяц, чтобы обеспечить адекватное время для тестирования.

В описанном случае сообщения для коммита может выглядеть примерно так:

```
PR:                 foo/54321
Submitted by:       John Smith <John.Smith@example.com>
Reviewed by:        --arch
Obtained from:      NetBSD
MFC after:          1 month
```

Вопрос: Как мне получить доступ к people.FreeBSD.org для того чтобы разместить там персональную информацию или информацию о моих проектах?

Ответ: people.FreeBSD.org — синоним для freefall.FreeBSD.org. Просто создайте каталог public_html. Все, что вы разместите в нем, будет автоматически доступно по адресу http://people.FreeBSD.org/ .

Вопрос: Где расположены архивы списков рассылки?

Ответ: Списки рассылки архивируются в иерархию каталогов /g/mail, видимую на всех машинах кластера как /hub/g/mail (см. pwd(1)).

Вопрос: Мне бы хотелось стать ментором для нового коммиттера. Какого технологического процесса я должен придерживаться?

Ответ: Обратитесь к документу Процедура создания нового аккаунта.