Встроенная команда exec

Синопсис

```
exec [-ИМЯ] [-cl] [КОМАНДА] [АРГУМЕНТ ...] [ПЕРЕНАПРАВЛЕНИЕ...]
```

Описание

`exec` Встроенная команда используется для

- **замените** оболочку заданной программой (выполняя ее, а **не как новый процесс**)
- установите перенаправления для выполняемой программы или для текущей оболочки

Если заданы только перенаправления, то перенаправления влияют на текущую оболочку без выполнения какой-либо программы.

Опции

Вариант	Описание
`-a NAME`	Передается `NAME` в качестве нулевого аргумента для выполняемой программы
`-c`	Запустите программу с пустой (очищенной) средой
`-l`	Добавляет тире (`-`) к нулевому аргументу выполняемой программы, аналогично тому, что `login` делает программа

Статус выхода

- при ошибках перенаправления он возвращает 1, в противном случае 0
- о сбоях exec:
  - неинтерактивная оболочка завершается; если установлен параметр оболочки `exec` execfail, возвращает сбой
  - в интерактивной оболочке `exec` возвращает сбой

Примеры

Оболочка вокруг программы

```
myprog=/bin/ls
 echo "Это сценарий-оболочка, он будет выполнять $ myprog"# сделайте
 здесь что-нибудь, возможно, измените аргументы и т. Д. # Ну, там ест
ь оболочка для exec




 "$ myprog" "$ @"
```

Откройте файл в качестве входных данных для скрипта

```
<
3  exec# открыть его input.txt

# например: прочитать одну строку из файла (-дескриптор)
читать -u  3 СТРОКИ
# или
 прочитайте СТРОКУ <&3

# наконец, закройте ее
 exec  3<&-
```

Общий файл журнала скрипта

Чтобы перенаправить всю `stdout` и `stderr` оболочки или shellscript в файл, вы можете использовать `exec` встроенную команду:

```
>/ exec var/adm/my.log 2>&1

# сценарий продолжается здесь ...
```

Соображения по переносимости

- POSIX® определяет диапазоны кодов ошибок:
  - если `exec` не удается найти программу для выполнения, код ошибки должен быть 126
  - при ошибке перенаправления код ошибки должен быть в диапазоне от 1 до 125

- `-a NAME` опция появилась в Bash 4.2-alpha
- POSIX® **не** указывает никаких параметров для `exec` (например `-c`, `-l`, `-a NAME`).

См . также

- Перенаправление

Обсуждение

Виталий (http://void.net.ua/wiki/bash:start), 2012/07/25 07:36 ()

Я бы добавил сюда оператор Bash "diamond" для открытия файлов для чтения и записи:

```
exec <>файл
```

Ян Шампера, 08.02.2012 07:02 ()

Согласен, он должен быть перекрестно связан с разделом перенаправлений.

Я никогда не слышал "алмазный оператор" - приятно

Питер Грин, 2014/06/20 13:27 ()

Я попал на эту страницу, пытаясь понять, как перенаправить вывод текущего скрипта как в файл, так и в консоль.

Моя первая попытка была

exec 2>&1 | тройник output.txt

Но это не сработало, спрашивая в irc, мне сказали в irc (от pgas), что это потому, что "exec выполняется в подоболочке" и "каждая сторона a | run в разветвленных подоболочках"

Предложение, данное на irc (pgas), было

exec > >(тройник output.txt) 2>и 1

Это вроде бы сработало, но имело нежелательный побочный эффект, заключающийся в том, что конечный вывод скрипта был записан на терминал после завершения скрипта, а не ожидания скрипта.

Последнее решение, к которому я пришел (также предложенное pgas), состояло в том, чтобы обернуть основную часть моего скрипта в функцию и перенаправить вывод функции.

главный() {

```
#делай что-нибудь здесь.
```

}

основной "$@" 2> & 1 | тройник output.txt

commands/builtin/exec.txt Последнее изменение: 2011/06/20 19:51 клянусь тебонсаем

Bash Хакеры Вики