

Главная (<https://wm-help.net/lib/>) / Библиотека (<https://wm-help.net/lib/b/>) / Linux программирование в примерах (<https://wm-help.net/lib/b/book/2075737573/>)
/ Часть 2 Процессы, IPC и интернационализация (<https://wm-help.net/lib/b/book/2075737573/166/>) / Глава 9 Управление процессами и каналы (<https://wm-help.net/lib/b/book/2075737573/167/>)
/ 9.1. Создание и управление процессами (<https://wm-help.net/lib/b/book/2075737573/168/>) / 9.1.2. Идентификация процесса: getpid() и getppid()

Обложка

Аннотация



(<https://wm-help.net/lib/b/book/2075737573/>)

 (<https://wm-help.net/lib/b/book/2075737573/b-map>)  (<https://wm-help.net/lib/b/book/2075737573/top>)  (http://wm.seo-doks.info/download_book.php?id=2075737573) 

(http://ucl.mixmarket.biz/uni/clk.php?id=1294931646&zid=1294933476&prid=1294933344&redir=http%3A%2F%2Fmy-shop.ru%2Fshop%2Fsearch%2Fa%2Fpage%2F1.html%3Fsm%3Dadv%26extended%3D3%26search_speller%3Don%26mode%3D0%26cmode%3D2%26f12_0%3Don%26f11%3D0%26f24%3D0%26f51%3D0%26f52%3D0)

Арнольд Роббинс (<https://wm-help.net/lib/b/a-list/#robbins>)i

Книги автора: Linux программирование в примерахUnix Programming By Example (<https://wm-help.net/lib/b/book/2075737573/>)

/ Р. Галеев (<https://wm-help.net/lib/b/a-list/#galeev>)i

Книги автора: Linux программирование в примерахUnix Programming By Example (<https://wm-help.net/lib/b/book/2075737573/>)

/ Arnold Robbins (<https://wm-help.net/lib/b/a-list/#robbins>)i

Книги автора: Linux программирование в примерахUnix Programming By Example (<https://wm-help.net/lib/b/book/2075737573/>)

Книга: Linux программирование в примерах (<https://wm-help.net/lib/b/book/2075737573/>)

9.1.2. Идентификация процесса: getpid() и getppid()

9.1.2. Идентификация процесса: getpid() и getppid()

У каждого процесса есть уникальный ID номер процесса (PID). Два системных вызова предоставляют текущий PID и PID родительского процесса:

```
#include <sys/types.h> /* POSIX */
#include <unistd.h>
pid_t getpid(void);
pid_t getppid(void);
```

Функции так просты, как выглядят:

pid_t getpid(void) Возвращает PID текущего процесса

pid_t getppid(void) Возвращает PID родителя.

Значения PID уникальны; по определению, не может быть двух запущенных процессов с одним и тем же PID. PID обычно возрастают в значении, так что порожденный процесс имеет обычно больший PID, чем его родитель. Однако, на многих системах значения PID *переполняются*; когда достигается значение системного максимума для PID, следующий процесс создается с наименьшим не используемым номером PID. (Ничто в POSIX не требует такого поведения, и некоторые системы назначают неиспользуемые номера PID случайным образом.)

Если родительский процесс завершается, порожденный получает нового родителя, `init`. В этом случае PID родителя будет 1, что является PID `init`. Такой порожденный процесс называется *висячим* (*orphan*). Следующая программа, `ch09-reparent.c`, демонстрирует это. Это также первый пример `fork()` в действии:

```
1  /* ch09-reparent.c --- показывает, что getppid() может менять значения */
2
3  #include <stdio.h>
4  #include <errno.h>
5  #include <sys/types.h>
6  #include <unistd.h>
7
8  /* main --- осуществляет работу */
9
10 int main(int argc, char **argv)
11 {
12     pid_t pid, old_ppid, new_ppid;
13     pid_t child, parent;
14
15     parent = getpid(); /* перед fork() */
16
17     if ((child = fork()) < 0) {
18         fprintf(stderr, "%s: fork of child failed: %sn",
19             argv[0], strerror(errno));
20         exit(1);
21     } else if (child == 0) {
22         old_ppid = getppid();
23         sleep(2); /* см. главу 10 */
24         new_ppid = getppid();
25     } else {
26         sleep(1);
27         exit(0); /* родитель завершается после fork() */
28     }
29
30     /* это выполняет только порожденный процесс */
31     printf("Original parent: %dn", parent);
32     printf("Child: %dn", getpid());
33     printf("Child's old ppid: %dn", old_ppid);
34     printf("Child's new ppid: %dn", new_ppid);
```

```
35
36 exit(0);
37 }
```

Строка 15 получает PID начального процесса, используя `getpid()` . Строки 17–20 создают порожденный процесс, проверяя по возвращении ошибки.

Строки 21–24 выполняются порожденным процессом: строка 22 получает PPID. Строка 23 приостанавливает процесс на две секунды (сведения о `sleep()` см в разделе 10.8.1 «Аварийные часы: `sleep()` , `alarm()` и `SIGALRM`»), а строка 24 снова получает PPID.

Строки 25–27 исполняются в родительском процессе. Строка 26 задерживает родителя на одну секунду, давая порожденному процессу достаточно времени для осуществления первого вызова `getppid()` . Строка 27 завершает родителя.

Строки 31–34 выводят значения. Обратите внимание, что переменная `parent` , которая была установлена до разветвления, сохраняет свое значение в порожденном процессе. После порождения у двух процессов идентичные, но независимые копии адресного пространства. Вот что происходит при запуске программы:

РЕКЛАМА

Денежные переводы из РФ за рубеж

Подробнее на koronapay.com

УСЛУГА ПРЕДОСТАВЛЯЕТСЯ РИКО «КОММЕРСИАЛЬНЫЙ ЦЕНТР» ЮЮЮ, Г. НЕВΟΣКОБИНСК, УЛ. КИРОВА, Б/Д, ОГРН 1025400012968. ЛИЦЕНЗИЯ ЦЕ РО № 3165-К ОТ 14.04.2014 С ПЕРЕЧЕНЬ НАПРАВЛЕНИЙ, ОТРАВИНИИИИИ, ДОГОВОРЫ О КОМПЛЕКСНОМ ОБСЛУЖИВАНИИ КЛИЕНТОВ, А ТАКЖЕ УСЛОВИЯ ОБЯЗАНИЯ УСЛУГИ СМОТРИТЕ НА ВВКО.ВВ В РАЗДЕЛЕ «ДЕНЕЖНЫЕ ПЕРЕВОДЫ». РИКО «КОММЕРСИАЛЬНЫЙ ЦЕНТР» ЮЮЮ ВПРАВЕ ОТКАЗАТЬ В ОКАЗАНИИ УСЛУГИ.

Золотая Корона

KoronaPay

```
$ ch09-reparent /* Запуск программы */
$ Original parent: 6582 /* Программа завершается: приглашение оболочки
                        и вывод порожденного процесса */

Child: 6583
Child's old ppid: 6582
Child's new ppid: 1
```

Помните, что обе программы выполняются *параллельно*. Графически это изображено на рис. 9.2.

Время	PID 6582	PID 6583	Внимание лишь один процесс
0	<code>child = fork();</code>		Порождение процесса
1	<code>sleep(1);</code>	<code>old_ppid = getppid();</code>	Родитель приостановлен, порожденный вызывает <code>getppid()</code>
2	<code>exit(0);</code>	<code>sleep(2);</code>	Родитель завершается, порожденный приостановлен
3	6583 получает нового родителя	Приостановка прерывается	Смена родителя порожденного процесса, пока он приостановлен
4		<code>new_ppid = getppid();</code>	Высший порожденный процесс вызывает <code>getppid()</code>

Рис. 9.2. Два параллельно исполняющихся процесса после разветвления

ЗАМЕЧАНИЕ. Использование `sleep()` , чтобы заставить один процесс пережить другой, работает в большинстве случаев. Однако, иногда случаются ошибки, которые трудно воспроизвести и трудно обнаружить. Единственным способом гарантировать правильное поведение является явная синхронизация с помощью `wait()` или `waitpid()` , которые описываются далее в главе (см. раздел 9.1.6.1 «Использование функций POSIX: `wait()` и `waitpid()` »).

(<https://wm-help.net/lib/b/book/2075737573/172>) (<https://wm-help.net/lib/b/book/2075737573/174>)

Оглавление книги

Оглавление статьи/книги

- 9.1.1. Создание процесса: `fork()` (<https://wm-help.net/lib/b/book/2075737573/169>)
- 9.1.2. Идентификация процесса: `getpid()` и `getppid()` (<https://wm-help.net/lib/b/book/2075737573/173>)
- 9.1.3. Установка приоритетов процесса: `nice()` (<https://wm-help.net/lib/b/book/2075737573/174>)
- 9.1.4. Запуск новой программы: семейство `exec()` (<https://wm-help.net/lib/b/book/2075737573/176>)
- 9.1.5. Завершение процесса (<https://wm-help.net/lib/b/book/2075737573/181>)
- 9.1.6. Использование статуса завершения порожденного процесса (<https://wm-help.net/lib/b/book/2075737573/185>)

Похожие страницы

- 9.1. Создание и управление процессами (<http://wm-help.net/lib/b/book/2075737573/168>)
- 9.1.1. Создание процесса: `fork()` (<http://wm-help.net/lib/b/book/2075737573/169>)
- Сущность процесса миграции (<http://wm-help.net/lib/b/book/1940220047/250>)
- V Совершенствование процесса (<http://wm-help.net/lib/b/book/1972705335/31>)
- Использование сервера Yaffil внутри процесса (<http://wm-help.net/lib/b/book/1940220047/400>)
- Глава 28 Идентификация и аутентификация пользователей (<http://wm-help.net/lib/b/book/3372038912/390>)
- 4. Стадии бизнес-процесса взаимодействия с клиентами (<http://wm-help.net/lib/b/book/3496167054/75>)

<https://wm-help.net/lib/b/book/2075737573/173>

3/4

- 2.2.2.2 Состояния процесса (<http://wm-help.net/lib/b/book/276807463/21>)
- Идентификация и аутентификация (<http://wm-help.net/lib/b/book/2989009035/70>)
- 1.2 Процесс, контекст процесса и потоки (<http://wm-help.net/lib/b/book/2770685295/7>)
- Вытеснение процесса (<http://wm-help.net/lib/b/book/1662500978/57>)
- При выключении не дождался конца процесса и отключил питание. Теперь при запуске компьютер начинает бесконечно обращаться... (<http://wm-help.net/lib/b/book/2179913312/197>)



(<http://www.liveinternet.ru/click>)

Генерация: 0.046. Запросов К БД/Cache: 0 / 0

(<https://vk.com/share.php?url=https%3A%2F%2Fwm-help.net%2Flib%2Fb%2Fbook%2F2075737573%2F173&title=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

(<https://www.facebook.com/sharer.php?src=sp&u=https%3A%2F%2Fwm-help.net%2Flib%2Fb%2Fbook%2F2075737573%2F173&title=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

(<https://connect.ok.ru/offer?url=https%3A%2F%2Fwm-help.net%2Flib%2Fb%2Fbook%2F2075737573%2F173&title=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

(<https://twitter.com/intent/tweet?text=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

(<https://www.livejournal.com/update.bml?subject=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

(<https://api.whatsapp.com/send?text=9.1.2.%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getpid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getppid%20%D0%98%D0%B4%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%B0%3A%20getp>)

поделиться

Вверх Вниз