join — ожидание завершения потока

Java Core (/quests/QUEST_JAVA_CORE)
6 уровень (/quests/lectures/?quest=QUEST_JAVA_CORE&level=6), 4 лекция (/quests/lectures/questcore.level06.lecture04)

ОТКРЫТА

— Привет, Амиго! Я смотрю, ты делаешь большие успехи в изучении нитей.

— Это оказалось не сложно.

— Это же отлично! Сегодня легкий урок, и тема этого урока – метод join.

Представь себе ситуацию: главная нить создала дочернюю нить для выполнения какого-то задания. Проходит время, и вот главной нити понадобились результаты работы той дочерней нити. А дочерняя нить еще не закончила свою работу. Что делать главной нити?

— Да, что делать главной нити?

— Для этого есть метод join. Смысл его в следующем. Одна нить ждет, пока полностью завершится работа второй нити:

Код

Описание

```
1  class Printer implements Runnable
2  {
3  private String name;
4  public Printer(String name)
5  {
6  this.name = name;
7  }
8  public void run()
9  {
10 System.out.println("I'm " + this.name);
11 }
12 }
```

Класс, который реализует интерфейс Runnable.

```
1  public static void main(String[] args)
2  {
3  Printer printer1 = new Printer("Коля");
4  Thread thread1 = new Thread(printer1);
5  thread1.start();
6
7  thread1.join();
8  }
```

Главная нить создает дочернюю нить – объект thread1.

Затем запускает ее – вызов thread1.start();

И ждет ее завершения – thread1.join();

Одна нить может вызвать метод join у объекта второй нити. В результате первая нить (которая вызвала метод) приостанавливает свою работу до окончания работы второй нити (у объекта которой был вызван метод).

Тут стоит различать две вещи: есть, собственно, нить – отдельный процесс выполнения команд, а есть объект этой нити (объект Thread).

— И это все?

— Да.

— А зачем нужно создавать нить и сразу же ждать ее завершения?

— Сразу же может и не нужно. А вот спустя какое-то время это может и понадобится. Главная нить после запуска первой дочерней нити может *раздать еще много заданий* другим нитям (создав их и вызвав метод start), а потом все – работы ей больше не осталось, нужно обрабатывать результаты работы первой дочерней нити. В таких случаях, когда нужно обязательно дождаться завершения работы другой нити и нужно вызывать метод join.

— Понятно.

‹ (/quests/lectures/questcore.level06.lecture03)

×16 › (/quests/lectures/questcore.level06.lecture05)

0 0 0 0 0 +13

Комментарии (27) популярные новые старые

Никита

Введите текст комментария

Роман 16 уровень, Харьков 1 февраля, 03:51

Наглядно про join();

Начало лекции

Ответить +9

Антон 19 уровень 27 февраля, 11:46

спасибо, действительно понятно объясняет

Ответить 0

theBaldSoprano 16 уровень, Санкт-Петербург 14 марта, 20:58

трустори

Ответить 0

Assanali 19 уровень 17 марта, 16:05

второй раз указали на это видео. посмотрел.
просто гениально мужик рассказывает.
Тебе тоже большое **спасибо**!

Ответить 0

rotarru 20 уровень, Минск 25 января, 04:35

А теперь заново решаем задачу javarush.task.task16.task1604 Вывод стек-трейса ☺

Ответить +1

Дима 20 уровень, Москва 8 января, 22:00

https://www.youtube.com/watch?v=mjHeTCuvyDE вот очень хороший видос на эту тему

Ответить +5

vinsler 28 уровень, Санкт-Петербург 28 декабря 2017, 22:52

```
1   thread1.start();
2
3   thread1.join();
```

т.е. в промежутке я могу накидать еще пару сотен строк?))
А когда мне нужно дождаться результата, я жму джойн?
А это функция или void или что? Возвращает готовность или нет?
Ей можно воткнуть цикл типа на проверку:
if (!thread1,join) я делаю еще кучу, а потом опять спрашиваю?
Или все это дальше будет?)))

Ответить 0

Vladislav Kovyazin 31 уровень 18 июля 2017, 13:34

С английского join значит "присоединиться". Мне кажется, что главная нить не останавливается, а просто присоединяется к выполнению кода нити, у которой и был вызван метод join. Я прав)?

Ответить 0

Karahanid 40 уровень, Алматы 29 июля 2017, 12:05

нет

Ответить 0

Akira Rokudo 20 уровень, Москва 6 августа 2017, 12:44

с английского join имеет много значений) и если бы у object не было метода wait,то можно было бы использовать это слово) ну или block)

Ответить 0

albaslug 36 уровень, Санкт-Петербург 8 августа 2017, 14:05

albasiuy 36 уровень, Санкт-Петербург 8 августа 2017, 14:05

Зачем гадать по названию метода, если есть спецификация (https://docs.oracle.com/javase/tutorial/essential/concurrency/join.html):
"The join method allows one thread to wait for the completion of another. If t is a Thread object whose thread is currently executing, t.join(); causes the current thread to pause execution until t's thread terminates."

Ответить +7

Сергей Зимовец 31 уровень 19 июня 2017, 17:13

join такой себе брейпоинт процесса, мейн не заканчивается пока нить thread1 не закончит свои процессы , вот только не понятно почему метод join вызывается на thread1 как в этом увидеть связь с мейн????
я так понимаю нужно в принципе отталкиваться то того что все нити это составные части мейн

Ответить +1

Сергей 19 уровень 26 июня 2017, 04:07

Связб самая прямая. Метод мейн работает в главной нитке(потоке), в этом же главном потоке вызывается работа дочернего (Thread1.start). Это два разных потока и они не являются составляющей мейна, как вы это написали. Выполняют разную работу и делают это параллельно, при этом может возникнуть ситуация что в главном потоке могут потребоваться данные из дочернего, а дочерний в свою очередь эти данные еще не отдал. И вот тут вызывается метод .join в главном потоке, который вызвал дочернюю нить, заставляющий главный поток приостановить работу до окончания работы дочернего потока. Как-то так)
Короче в мейне вызываешь поток и чтоб не ускакать дальше него отрабатываешь на этот поток методом join. Все.

Ответить +15

Akira Rokudo 20 уровень, Москва 6 августа 2017, 12:42

```
else {
            x = x *5;
            System.out.println("Работа в 1 дочерней нити завершена");
        }
    }
  }
}
```

Т.е. отталкиваться нужно от методов run. Если дело происходит не в результате вызова через какой-то из методов run-значит это происходит в главной нити)(т.е. в приведенном примере мб также еще какой-то метод
JoinerThread(Thread t) {
t.join;
}
который вызываться может как и в методе main(т.е. главной нити) так и в методе run какой-то дочерней нити.

Ответить 0

fromCPPtoJava 19 уровень 19 августа 2017, 10:06

Четко отмечено!

Ответить 0

Карина С. 27 уровень, Москва 11 мая 2017, 14:39

Чего-то я не поняла где тут первая а где вторая нить?

Ответить 0

Alexander 25 уровень 21 мая 2017, 22:18

1 я нить main, 2 я thread1.

Ответить +1

Denis Beck 37 уровень, Екатеринбург 7 июня 2017, 16:57

первая нить - в методе main, вторая, которую запускает thread1, - в методе run объекта printer1

Ответить +1

Тимур Шамилов 35 уровень 3 апреля 2017, 17:17

Седьмой всегда крутой!

Ответить 0

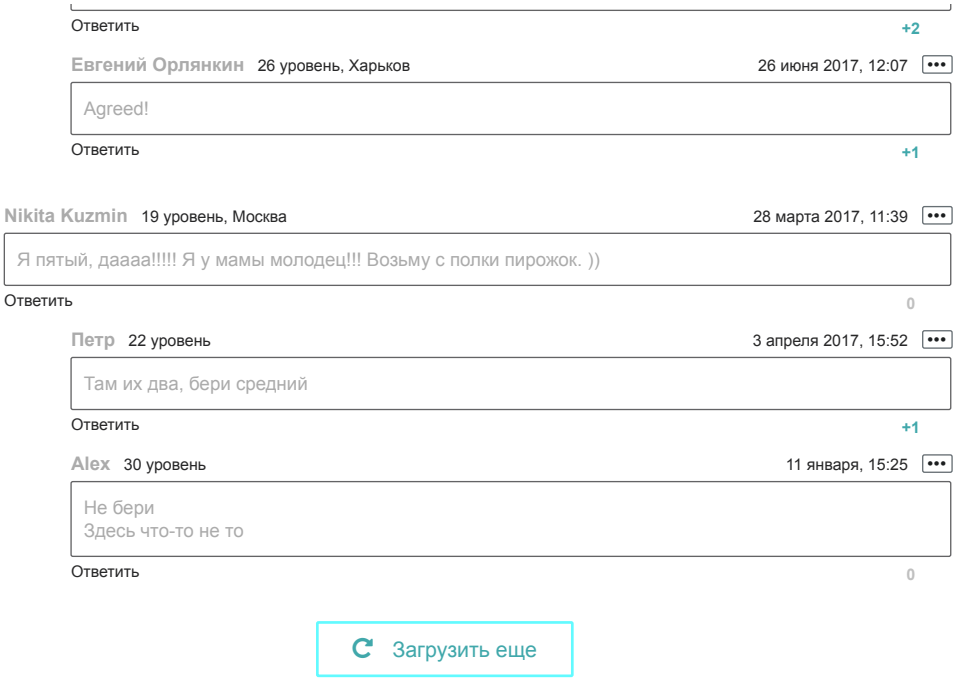
Евгений Гутько 22 уровень, Минск 29 марта 2017, 22:05

Господи, сколько наркоманов! Я ожидал увидеть в комментах дополнение по теме.

Ответить +3

Alexey Smirnov 29 уровень, Москва 2 мая 2017, 15:35

Типично для № 6, не так ли, коллеги?



ush.ru/) (https://plus.google.com/114772402300089087607) (https://twitter.com/javarush_ru)

JAVARUSH
Программистами не рождаются
© 2018