Остановить бушующий поток: официальная версия

Java Core (/quests/QUEST_JAVA_CORE)
6 уровень (/quests/lectures/?quest=QUEST_JAVA_CORE&level=6), 9 лекция (/quests/lectures/questcore.level06.lecture09)

ОТКРЫТА

— Привет, Амиго! Согласись, Элли хорошо придумала с этим Cancel?

— Ага.

— На самом деле нечто подобное существует в классе Thread. Только переменная называется не isCancel, а isInterrupt, и метод остановки, соответственно, не cancel(), а interrupt().

— Да?

— Ага. Вот смотри:

Код

Описание

```
1  class Clock implements Runnable
2  {
3  public void run()
4  {
5  Thread current = Thread.currentThread();
6
7  while (!current.isInterrupted())
8  {
9  Thread.sleep(1000);
10 System.out.println("Tik");
11 }
12 }
13 }
```

Т.к. много нитей могут вызвать метод run одного объекта, то объект Clock в своем методе runполучает объект вызвавшей его нити («текущей нити»).

Класс Clock (часы) будет писать в консоль раз в секунду слово «Tik», пока переменная isInterrupt текущей нити равна false.

Когда переменная isInterrupt станет равной true, метод run завершится.

```
1 public static void main(String[] args)
2 {
3 Clock clock = new Clock();
4 Thread clockThread = new Thread(clock);
5 clockThread.start();
6
7 Thread.sleep(10000);
8 clockThread.interrupt();
9 }
```

Главная нить, запускает дочернюю нить – часы, которая должна работать вечно.

Ждет 10 секунд и отменяет задание, вызовом метода interrupt.

Главная нить завершает свою работу.

Нить часов завершает свою работу.

Более того, в методе sleep, который так любят использовать для организации вечного цикла в методе run, есть автоматическая проверка переменной isInterrupt. Если нить вызовет метод sleep, то этот метод сначала проверит, а не установлена ли для текущей (вызвавшей его нити) переменная isInterrupt в true. И если установлена, то метод не будет спать, а выкинет исключение InterruptedException.

— А зачем выкидывать исключение? Не лучше ли тоже просто в цикле вместо isCancel подставить isInterrupted()?

— Во-первых, не всегда в методе run есть цикл. Метод может состоять просто из двух десятков вызовов других методов. Тогда перед вызовом каждого придется добавлять проверку isInterrupted.

Во-вторых, вдруг какой-то метод очень долго исполняется, т.к. делает много разных действий.

В-третьих, выкидывание исключения – это не замена проверке isInterrupted, а скорее удобное дополнение. Выкинутое исключение позволяет быстро раскрутить стек вызовов до самого run.

В-четвертых, метод sleep часто используют, и, получается, к такому полезному методу неявно добавили не менее полезную проверку. Вроде бы никто специально проверку не добавлял, а она есть. Это очень ценно, когда ты используешь много чужого кода и не можешь сам добавить в него проверку.

В-пятых, дополнительная проверка не приводит к снижению производительности. Вызов метода sleep значит, что нить должна ничего не делать (спать), поэтому дополнительная работа никому не мешает.

— Серьёзные аргументы.

— И, наконец, последнее: ты можешь в своем методе run вызывать чужой код, к которому у тебя нет доступа (исходников и/или прав их менять). Он может не иметь проверок на isInterrupted, а также перехватывать с помощью try…catch(Exception e) все возникшие исключения.

Никто не гарантирует, что нить можно остановить. Она может остановиться только сама.

‹ (/quests/lectures/questcore.level06.lecture08)

×16 › (/quests/lectures/questcore.level06.lecture10)

Комментарии (30) популярные новые старые

Никита

Введите текст комментария

Max 19 уровень, Киев 13 марта, 22:21

Кто-нибудь собирается править ошибки в тексте?
Мало того, что полу чушь, так еще и безграмотно написано...

Ответить 0

Redas Shuliakas 21 уровень 9 февраля, 10:04

"то объект Clock в своем методе runполучает" - Я думаю многие после этой строчки перестали читать, из уважения к себе.

Ответить 0

Alex 30 уровень 10 января, 17:22

Ээ, а кто-то понял эту фразу?

— А зачем выкидывать исключение? Не лучше ли тоже просто в цикле вместо isCancel подставить isInterrupted()?

В примере же нет isCancel и уже используется isInterrupted()...
Или я уже ебобо немного?

Ответить +1

RayFinkle-s-Pilles 19 уровень 11 января, 12:51

— А зачем выкидывать исключение? Не лучше ли просто использовать пример из предыдущего разъяснения, только написать вместо isCancel isInterrupted?

ну и да, ты походу устал тогда, лол

Ответить 0

Alex 30 уровень 11 января, 13:44

та да
весь день пилил
уже закипал

сэнкс)

Ответить 0

Джонни 23 уровень 5 декабря 2017, 00:53

Из этой лекции неясно как работать с этими методами.
Вот небольшое пояснение.


Ответить +12

Джонни 23 уровень 4 декабря 2017, 22:50

Ребят, если этот пример написать в идее, то цикл вечный! Он не отменяется. Просто выкидывает исключение и идёт дальше.

Ответить +2

Andrei Rusu 22 уровень 4 декабря 2017, 23:50

The same in Eclipse.

Ответить 0

Andrei Rusu 22 уровень 4 декабря 2017, 23:52

Смотри ниже:

Хочу добавить от себя, если в методе run потока есть метод sleep то компилятор предупреждает что sleep может выбросить InterruptedException и просит его перехватить блоком try/catch. Если мы это исключение просто поглощаем, т.е не чего не делаем в блоке catch, то поток при вызове метода interrupt не остановится. Для остановки потока нужно повторно вызвать метод interrupt (Thread.currentThread().interrupt();) в блоке catch либо return из метода run.

Ответить +1

Ильяс 23 уровень, Москва 5 декабря 2017, 00:06

Подводя итоги, скажу следующее:
Вызывая у треада interrupt() в момент, когда он спит, выбрасывается InterruptedException

Флаг interrupt сбрасывается на false. Поэтому цикл будет вечен. И чтобы избежать этого, можно (например) в блоке catch повторно вызывать interrupt()

И да, то что написано в статье
"*Класс Clock (часы) будет писать в консоль раз в секунду слово «Tik», пока переменная isInterrupt текущей нити равна false.*

Когда переменная isInterrupt станет равной true, метод run завершится."

Полная чушь получается...

Поэтому надо бдить.

Ответить +2

Джонни 23 уровень 5 декабря 2017, 01:03

А вот после того, как я вызвал в блоке catch метод interrupt(); - всё заработало!

Ответить 0

Сергей 27 уровень 28 ноября 2017, 17:09

Отличная статья на Хабре в тему Многопоточность Java

Ответить +11

Джонни 23 уровень 3 декабря 2017, 14:49

Спасибо, очень познавательно!

Ответить 0

Эльдар Ахметов 20 уровень, Уфа 2 ноября 2017, 10:16

Хочу добавить от себя, если в методе run потока есть метод sleep то компилятор предупреждает что sleep может выбросить InterruptedException и просит его перехватить блоком try/catch. Если мы это исключение просто поглощаем, т.е не чего не делаем в блоке catch, то поток при вызове метода interrupt не остановится. Для остановки потока нужно повторно вызвать метод interrupt (Thread.currentThread().interrupt();) в блоке catch либо return из метода run.

Ответить +2

Roman Kopyl 39 уровень 24 сентября 2017, 16:21

так isInterrupt или isInterrupted?.

Ответить 0

Hamster726 22 уровень, Киев 6 октября 2017, 08:22

interrupt() - прервать нить.
isInterrupted() - Проверяет прервана нить или нет. Возвращает значение true или false

Ответить 0

Дмитрий 31 уровень 1 ноября 2017, 06:33

Не путайте людей! isIterrupted - только переменная (которая по умолчанию = true), а interrupt - метод, который меняет значение переменной isIterrupted на - false.
Соответственно - interrupt - не в любом случае прервет нить, а только если есть условие:

```
1   while (!current.isInterrupted())
```

И isIterrupted - ничего не проверяет, а уж тем более, ничего не возвращает - это всего лишь переменная, с помощью которой уже, можно составить условие - прервать нить или нет!

Ответить +5

Дмитрий 31 уровень 1 ноября 2017, 06:35

Да и вопрос стоял совсем другой, в лекции ошибка, в одном месте переменная называется isIterrupt, а в другом isIterrupted

Ответить +1

Дмитрий 31 уровень 1 ноября 2017, 06:37

Я всё же думаю, что isIterrupted, т.к. в коде, именно это название, а там, ошибки быть не может

Ответить 0

Дмитрий Николаев 21 уровень, Нижний Новгород 7 ноября 2017, 19:54

Судя по документации, метод isInterrupted() также существует:
https://docs.oracle.com/javase/8/docs/api/java/lang/Thread.html#isInterrupted--
Он проверяет, был ли данный поток прерван.

Более того в классе Thread есть ещё и статический метод interrupted():
https://docs.oracle.com/javase/8/docs/api/java/lang/Thread.html#interrupted--
Он проверяет, был ли текущий поток прерван.

Ответить 0

Mikhail 20 уровень 1 декабря 2017, 08:33

Наверное имел ввиду по-умолчанию = false

Ответить 0

Dmitry Kaltovich 35 уровень, Минск 21 июня 2017, 19:30

Очепятки в тескте

Ответить 0

gerd 18 уровень, Москва 14 июня 2017, 16:57

runполучает > run получает

Ответить 0

Загрузить еще