**Остановить бушующий поток: неофициальная версия**

[Java Core](https://javarush.ru/quests/QUEST_JAVA_CORE)

[Уровень 6](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_CORE&level=6), Лекция 8

— Привет, Амиго! Вот интересный вопрос, с которым ты уже столкнулся или столкнёшься в ближайшее время. А **как остановить запущенную нить**?

Допустим, пользователь отдал программе команду «загрузить файл из интернета». Главная нить создала для этого задания отдельную дочернюю нить, и передала ей объект, метод run которого содержит все необходимые действия для скачивания файла.

А тут пользователь – раз и передумал. Не хочет он качать этот файл. Как отменить задание и остановить нить?

— Да, как?

— **Никак**. Это и есть самый общий и самый правильный ответ. Нить остановить нельзя, она может остановиться только сама.

Но можно дать нити сигнал, сообщить ей каким-нибудь образом, что работу больше выполнять не нужно, и ей нужно завершиться. Так же, как главная нить завершается вместе с выходом из метода main, чтобы завершить дочернюю нить, она должна закончить выполнение метода run.

— И как это лучше всего сделать?

— Можно завести какую-нибудь переменную, например типа **boolean**. Если она **true** – нить работает. Если же она стала **false** – нить должна завершиться. Например, так:

|  |  |
| --- | --- |
| **Код** | **Описание** |
| class Clock implements Runnable  {  public void run()  {  while (true)  {  Thread.sleep(1000);  System.out.println("Tik");  if (!ClockManager.isClockRun)  return;  }  }  } | Класс Clock (часы) будет вечно писать в консоль раз в секунду слово «Tik»  Если переменная ClockManager.isClockRun равна false – метод run завершится. |
| class ClockManager  {  public static boolean isClockRun = true;  public static void main(String[] args)  {  Clock clock = new Clock();  Thread clockThread = new Thread(clock);  clockThread.start();  Thread.sleep(10000);  isClockRun = false;  }  } | Главная нить, запускает дочернюю нить – часы, которая должна работать вечно.  Ждет 10 секунд и подает часам сигнал на завершение.  Главная нить завершает свою работу.  Нить часов завершает свою работу. |

— А если у нас несколько нитей, что тогда?

— Тогда лучше завести такую переменную для каждой нити. Удобнее всего будет добавить ее прямо в класс. Можно добавить туда переменную **boolean** **isRun**. Хотя лучше добавить переменную boolean **isCancel**, которая будет принимать значение **true**, если задание отменено.

|  |  |
| --- | --- |
| **Код** | **Описание** |
| class Clock implements Runnable  {  private boolean isCancel = false;  public void cancel()  {  this.isCancel = true;  }  public void run()  {  while (!isCancel)  {  Thread.sleep(1000);  System.out.println("Tik");  }  }  } | Класс Clock (часы) будет писать в консоль раз в секунду слово «Тик», пока переменная isCancel равна false.  Когда переменная isCancel станет равной true, метод run завершится. |
| public static void main(String[] args)  {  Clock clock = new Clock();  Thread clockThread = new Thread(clock);  clockThread.start();  Thread.sleep(10000);  clock.cancel();  } | Главная нить, запускает дочернюю нить – часы, которая должна работать вечно.  Ждет 10 секунд и отменяет задание, вызовом метода cancel.  Главная нить завершает свою работу.  Нить часов завершает свою работу. |

— Буду знать, спасибо, Элли.