**join — ожидание завершения нити**

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

[Уровень 6](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_CORE&level=6), Лекция 4

— 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** |
|  |  |
| class Printer implements Runnable  {  private String name;  public Printer(String name)  {  this.name = name;  }  public void run()  {  System.out.println("I’m " + this.name);  }  } | | | Класс, который реализует интерфейс Runnable. |
|  | | |  |

|  |  |
| --- | --- |
| public static void main(String[] args)  {  Printer printer1 = new Printer("Коля");  Thread thread1 = new Thread(printer1);  thread1.start();  **thread1**.join();  } | 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.