

Joining threads

 studytonight.com/java/joining-a-thread.php

Sometimes one thread needs to know when other thread is terminating. In java, **isAlive()** and **join()** are two different methods that are used to check whether a thread has finished its execution or not.

The **isAlive()** method returns **true** if the thread upon which it is called is still running otherwise it returns **false**.

```
final boolean isAlive()
```

But, **join()** method is used more commonly than **isAlive()**. This method waits until the thread on which it is called terminates.

```
final void join() throws InterruptedException
```

Using **join()** method, we tell our thread to wait until the specified thread completes its execution. There are overloaded versions of **join()** method, which allows us to specify time for which you want to wait for the specified thread to terminate.

```
final void join(long milliseconds) throws InterruptedException
```

, the main thread must always be the last thread to finish its execution. Therefore, we can use Thread join() method to ensure that all the threads created by the program has been terminated before the execution of the main thread.

Example of **isAlive** method

```

public class MyThread extends Thread
{
    public void run()
    {
        System.out.println("r1 ");
        try {
            Thread.sleep(500);
        }
        catch(InterruptedException ie) { }
        System.out.println("r2 ");
    }
    public static void main(String[] args)
    {
        MyThread t1=new MyThread();
        MyThread t2=new MyThread();
        t1.start();
        t2.start();
        System.out.println(t1.isAlive());
        System.out.println(t2.isAlive());
    }
}

```

r1 true true r1 r2 r2

Example of thread without `join()` method

```

public class MyThread extends Thread
{
    public void run()
    {
        System.out.println("r1 ");
        try {
            Thread.sleep(500);
        }
        catch(InterruptedException ie){ }
        System.out.println("r2 ");
    }
    public static void main(String[] args)
    {
        MyThread t1=new MyThread();
        MyThread t2=new MyThread();
        t1.start();
        t2.start();
    }
}

```

r1 r1 r2 r2

In this above program two thread t1 and t2 are created. t1 starts first and after printing "r1" on console thread t1 goes to sleep for 500 ms. At the same time Thread t2 will start its process and print "r1" on console and then go into sleep for 500 ms. Thread t1 will wake up