

Java join() method

The `join()` method in Java is provided by the `java.lang.Thread` class that permits one thread to wait until the other thread to finish its execution. Suppose *th* be the object the class `Thread` whose thread is doing its execution currently, then the `th.join();` statement ensures that *th* is finished before the program does the execution of the next statement. When there are more than one thread invoking the `join()` method, then it leads to overloading on the `join()` method that permits the developer or programmer to mention the waiting period. However, similar to the `sleep()` method in Java, the `join()` method is also dependent on the operating system for the timing, so we should not assume that the `join()` method waits equal to the time we mention in the parameters. The following are the three overloaded `join()` methods.

Description of The Overloaded `join()` Method

join(): When the `join()` method is invoked, the current thread stops its execution and the thread goes into the wait state. The current thread remains in the wait state until the thread on which the `join()` method is invoked has achieved its dead state. If interruption of the thread occurs, then it throws the `InterruptedException`.

Syntax:

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

join(long mls): When the `join()` method is invoked, the current thread stops its execution and the thread goes into the wait state. The current thread remains in the wait state until the thread on which the `join()` method is invoked called is dead or the wait for the specified time frame(in milliseconds) is over.

Syntax:

```
public final synchronized void join(long mls) throws InterruptedException, where mls is in milliseconds
```

join(long mls, int nanos): When the join() method is invoked, the current thread stops its execution and go into the wait state. The current thread remains in the wait state until the thread on which the join() method is invoked called is dead or the wait for the specified time frame(in milliseconds + nanos) is over.

Syntax:

```
public final synchronized void join(long mls, int nanos) throws InterruptedException, where mls is in
```

Example of join() Method in Java

The following program shows the usage of the join() method.

FileName: ThreadJoinExample.java

```
// A Java program for understanding
// the joining of threads

// import statement
import java.io.*;

// The ThreadJoin class is the child class of the class Thread
class ThreadJoin extends Thread
{
    // overriding the run method
    public void run()
    {
        for (int j = 0; j < 2; j++)
        {
            try
            {
                // sleeping the thread for 300 milli seconds
                Thread.sleep(300);
                System.out.println("The current thread name is: " + Thread.currentThread().getName());
            }
        }
    }
}
```

```
// catch block for catching the raised exception
catch(Exception e)
{
    System.out.println("The exception has been caught: " + e);
}
System.out.println(j);
}
}
}

public class ThreadJoinExample
{
    // main method
    public static void main (String args[])
    {

        // creating 3 threads
        ThreadJoin th1 = new ThreadJoin();
        ThreadJoin th2 = new ThreadJoin();
        ThreadJoin th3 = new ThreadJoin();

        // thread th1 starts
        th1.start();

        // starting the second thread after when
        // the first thread th1 has ended or died.
        try
        {
            System.out.println("The current thread name is: " + Thread.currentThread().getName());

            // invoking the join() method
            th1.join();
        }

        // catch block for catching the raised exception
        catch(Exception e)
        {
            System.out.println("The exception has been caught " + e);
        }
    }
}
```

```
// thread th2 starts
th2.start();

// starting the th3 thread after when the thread th2 has ended or died.
try
{
    System.out.println("The current thread name is: " + Thread.currentThread().getName());
    th2.join();
}

// catch block for catching the raised exception
catch(Exception e)
{
    System.out.println("The exception has been caught " + e);
}

// thread th3 starts
th3.start();
}
}
```

Output:

```
The current thread name is: main
The current thread name is: Thread - 0
0
The current thread name is: Thread - 0
1
```

```
The current thread name is: main
The current thread name is: Thread - 1
0
The current thread name is: Thread - 1
1
The current thread name is: Thread - 2
0
The current thread name is: Thread - 2
1
```

Explanation: The above program shows that the second thread th2 begins after the first thread th1 has ended, and the thread th3 starts its work after the second thread th2 has ended or died.

The Join() Method: InterruptedException

We have learnt in the description of the join() method that whenever the interruption of the thread occurs, it leads to the throwing of InterruptedException. The following example shows the same.

FileName: ThreadJoinExample1.java

```
class ABC extends Thread
{
    Thread threadToInterrupt;
    // overriding the run() method
    public void run()
    {
        // invoking the method interrupt
        threadToInterrupt.interrupt();
    }
}

public class ThreadJoinExample1
{
    // main method
    public static void main(String[] args)
    {
        try
        {
            // creating an object of the class ABC
            ABC th1 = new ABC();
```

```
th1.threadToInterrupt = Thread.currentThread();
th1.start();

// invoking the join() method leads
// to the generation of InterruptedException
th1.join();
}
catch (InterruptedException ex)
{
    System.out.println("The exception has been caught. " + ex);
}
}
```

Output:

```
The exception has been caught. java.lang.InterruptedException
```

Some More Examples of the join() Method

Let' see some other examples.

Filename: TestJoinMethod1.java

```
class TestJoinMethod1 extends Thread{
    public void run(){
        for(int i=1;i<=5;i++){
            try{
                Thread.sleep(500);
            }catch(Exception e){System.out.println(e);}
            System.out.println(i);
        }
    }
    public static void main(String args[]){
        TestJoinMethod1 t1=new TestJoinMethod1();
        TestJoinMethod1 t2=new TestJoinMethod1();
        TestJoinMethod1 t3=new TestJoinMethod1();
        t1.start();
```

```
try{  
    t1.join();  
}catch(Exception e){System.out.println(e);}  
  
t2.start();  
t3.start();  
}  
}
```

Output:

```
1  
2  
3  
4  
5  
1  
1  
2  
2  
3  
3  
4  
4  
5  
5
```

We can see in the above example, when t1 completes its task then t2 and t3 starts executing.

join(long milliseconds) Method Example

Filename: TestJoinMethod2.jav

```
class TestJoinMethod2 extends Thread{  
    public void run(){  
        for(int i=1;i<=5;i++){  
            try{  
                Thread.sleep(500);  
            }catch(Exception e){System.out.println(e);}  
            System.out.println(i);  
        }  
    }  
    public static void main(String args[]){  
        TestJoinMethod2 t1=new TestJoinMethod2();  
        TestJoinMethod2 t2=new TestJoinMethod2();  
        TestJoinMethod2 t3=new TestJoinMethod2();  
        t1.start();  
        try{  
            t1.join(1500);  
        }catch(Exception e){System.out.println(e);}  
  
        t2.start();  
        t3.start();  
    }  
}
```

Output:

```
1  
2  
3  
1  
4  
1  
2  
5  
2  
3  
3  
4  
4  
5  
5
```

In the above example, when t1 completes its task for 1500 milliseconds(3 times), then t2 and t3 start executing.

[← Prev](#)[Next →](#)

 **For Videos Join Our Youtube Channel: [Join Now](#)**


Feedback


- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share





Learn Latest Tutorials


 Splunk tutorial
Splunk


 SPSS tutorial
SPSS


 Swagger
tutorial
Swagger


 T-SQL tutorial
Transact-SQL


 Tumblr tutorial
Tumblr

 React tutorial
ReactJS

 Regex tutorial
Regex

 Reinforcement
learning tutorial
Reinforcement
Learning


 R Programming
tutorial
R Programming


 RxJS tutorial
RxJS

 React Native
tutorial
React Native

 Python Design
Patterns
Python Design
Patterns


 Python Pillow
tutorial
Python Pillow


 Python Turtle
tutorial
Python Turtle

 Keras tutorial
Keras

Preparation

 Aptitude
Aptitude

 Logical
Reasoning
Reasoning

 Verbal Ability
Verbal Ability

 Interview
Questions
Interview Questions



Company
Interview
Questions

Company Questions

Trending Technologies



Artificial
Intelligence

Artificial
Intelligence



AWS Tutorial
AWS



Selenium
tutorial

Selenium



Cloud
Computing

Cloud Computing



Hadoop tutorial
Hadoop



ReactJS
Tutorial

ReactJS



Data Science
Tutorial

Data Science



Angular 7
Tutorial

Angular 7



Blockchain
Tutorial

Blockchain



Git Tutorial
Git



Machine
Learning Tutorial

Machine Learning



DevOps
Tutorial

DevOps

B.Tech / MCA



DBMS tutorial
DBMS



Data Structures
tutorial

Data Structures



DAA tutorial
DAA



Operating
System

Operating System

| | | | |
|--|--|--|--|
| | | | |
|  Computer Network tutorial Computer Network |  Compiler Design tutorial Compiler Design |  Computer Organization and Architecture Computer Organization |  Discrete Mathematics Tutorial Discrete Mathematics |
|  Ethical Hacking Ethical Hacking |  Computer Graphics Tutorial Computer Graphics |  Software Engineering Software Engineering |  html tutorial Web Technology |
|  Cyber Security tutorial Cyber Security |  Automata Tutorial Automata |  C Language tutorial C Programming |  C++ tutorial C++ |
|  Java tutorial Java |  .Net Framework tutorial .Net |  Python tutorial Python |  List of Programs Programs |
|  Control Systems tutorial Control System |  Data Mining Tutorial Data Mining |  Data Warehouse Tutorial Data Warehouse | |

