Topics

- Role of join() Method?
- Suspending and Resuming Threads ?

Joining Threads via join() Method

- final void join() throws interruptedException
- join() method allows the thread to wait until the thread on which it was called terminates.
- Calling thread waits until the specified thread joins.

join() Method Example

```
class JoinTest
{
        public static void main(String args[])
        {
                Thread T1 = new Thread()
                {
                        public void run()
                        {
                                try
                                {
                                        for(int i =0; i<2; i++)
                                        {
                                                System.out.println("Hello");
                                                Thread.sleep(100);
                                        }
                                }
                                catch(InterruptedException e) {}
                        }// End of Method
                }; // End of class and Statement
```

join() Method Example …

```
Thread T2 = new Thread()
{
        public void run()
        {
                try
                {
                        for(int i =0; i<2; i++)
                        {
                                System.out.println("Welcome");
                                Thread.sleep(100);
                        }
                }
                catch(InterruptedException e) {}
        }// End of Methid
}; // End of class and Statement

T1.start();
T2.start();

for(int i =0; i<2;i++)
        System.out.println("Hello Welcome");
System.out.println("Main Method Exits");
    }// End of main() Method
} // End of class JoinTest
```

join() Method Example … Sample Outputs

Hello Welcome
Hello Welcome
Welcome
Hello
Main Method Exits
Welcome
Hello

Hello Welcome
Hello Welcome
Main Method Exits
Welcome
Hello
Welcome
Hello

Hello Welcome
Hello Welcome
Main Method Exits
Welcome
Hello
Hello
Welcome

join() Method Example …

```
class JoinTest
{
        public static void main(String args[])
        {

                Thread T1 = .....;  // Same statement as used Earlier
                Thread T2 = .....;  // Same statement as used Earlier
                T1.start();
                try
                {
                        T1.join();
                }
                catch(InterruptedException e) {        }
                T2.start();
                try
                {
                        T2.join();
                }
                catch(InterruptedException e) {        }
                for(int i =0; i<2;i++)
                        System.out.println("Hello Welcome");
                System.out.println("Main Method Exits");
        } // End of Method
}//End of class
```

<<OUTPUT>>

```
Hello
Hello
Welcome
Welcome
Hello Welcome
Hello Welcome
Main Method Exits
```

Suspending and Resuming Threads

- Earlier Versions of Java supports the following two methods for suspending and resuming the threads
  1. final void suspend(); → Suspends the Currently Running Thread
  2. final void resume(); → Resumes the Currently Suspended Thread
- However, both suspend() and resume() methods have been deprecated
- Modern Approach for suspending and resuming threads is via synchronization [Next slide]

Modern Approach for Suspending and Resuming Threads

```
class SampleThread extends Thread
{
        ……..
        boolean suspendFlag=false;          // Introduce a boolean type flag variable

        // Introduce a suspend method which sets the value of suspendFlag = true
        void mySuspend() { suspendFlag = true; }

        // Introduce a synchronized resume method which sets the value of suspendFlag = false
        synchronized void myResume() { suspendFlag = false; }

        public void run()
        {
                // Check the value of flag at the entry
                try
                {
                        synchronized(this) { while(suspendFlag) wait(); }
                        ……..
                }
                catch(InterruptedException e) { }
        } // End of run Method
} // end of thread class
```

Thank You