

Apr 07, 04 12:43

Example1.java

Page 1/1

//Listing A: Stopping a thread with Thread.interrupt()

```
class Example1 extends Thread {  
    public static void main( String args[] ) throws Exception {  
        Example1 thread = new Example1();  
        System.out.println( "Starting thread..." );  
        thread.start();  
        Thread.sleep( 3000 );  
        System.out.println( "Interrupting thread..." );  
        thread.interrupt();  
        Thread.sleep( 3000 );  
        System.out.println( "Stopping application..." );  
        System.exit( 0 );  
    }  
  
    public void run() {  
        while ( true ) {  
            System.out.println( "Thread is running..." );  
            long time = System.currentTimeMillis();  
            while ( System.currentTimeMillis()-time < 1000 ) {  
                }  
            }  
        }  
    }  
}
```

Apr 07, 04 12:44

Example2.java

Page 1/1

//Listing B: Signaling a thread to stop

```
class Example2 extends Thread {  
    volatile boolean stop = false;  
  
    public static void main( String args[] ) throws Exception {  
        Example2 thread = new Example2();  
        System.out.println( "Starting thread..." );  
        thread.start();  
        Thread.sleep( 3000 );  
        System.out.println( "Asking thread to stop..." );  
        thread.stop = true;  
        Thread.sleep( 3000 );  
        System.out.println( "Stopping application..." );  
        System.exit( 0 );  
    }  
  
    public void run() {  
        while ( !stop ) {  
            System.out.println( "Thread is running..." );  
            long time = System.currentTimeMillis();  
            while ( (System.currentTimeMillis()-time < 1000) && (!stop) ) {  
                }  
        }  
        System.out.println( "Thread exiting under request..." );  
    }  
}
```

Apr 07, 04 12:44

Example3.java

Page 1/1

// Listing C: Exiting blocked states with Thread.interrupt()

```
class Example3 extends Thread {
    volatile boolean stop = false;

    public static void main( String args[] ) throws Exception {
        Example3 thread = new Example3();
        System.out.println( "Starting thread..." );
        thread.start();
        Thread.sleep( 3000 );
        System.out.println( "Asking thread to stop..." );
        thread.stop = true;
        thread.interrupt();
        Thread.sleep( 3000 );
        System.out.println( "Stopping application..." );
        System.exit( 0 );
    }

    public void run() {
        while ( !stop ) {
            System.out.println( "Thread running..." );
            try {
                Thread.sleep( 1000 );
            } catch ( InterruptedException e ) {
                System.out.println( "Thread interrupted..." );
            }
        }
        System.out.println( "Thread exiting under request..." );
    }
}
```

Apr 07, 04 12:45

Example4.java

Page 1/2

// Listing D: Interrupting I/O operations with Thread.interrupt()

```
import java.io.*;
import java.net.*;
```

```
class Example4 extends Thread {
```

```
    public static void main( String args[] ) throws Exception {
```

```
        Example4 thread = new Example4();
        System.out.println( "Starting thread..." );
        thread.start();
        Thread.sleep( 3000 );
        System.out.println( "Interrupting thread..." );
        thread.interrupt();
        Thread.sleep( 3000 );
        System.out.println( "Stopping application..." );
        System.exit( 0 );
    }
```

```
    public void run() {
```

```
        ServerSocket socket;
        try {
            socket = new ServerSocket(7856);
        } catch ( IOException e ) {
            System.out.println( "Could not create the socket..." );
            return;
        }
        while ( true ) {
            System.out.println( "Waiting for connection..." );
            try {
                Socket sock = socket.accept();
            } catch ( IOException e ) {
                System.out.println( "accept() failed or interrupted..." );
            }
        }
    }
}
```

Apr 07, 04 12:45

Example4.java

Page 2/2

```
}
```

Apr 07, 04 12:46

Example5.java

Page 1/2

// Listing E: Causing I/O operations to fail

```

import java.net.*;
import java.io.*;

class Example5 extends Thread {
    volatile boolean stop = false;
    volatile ServerSocket socket;

    public static void main( String args[] ) throws Exception {
        Example5 thread = new Example5();
        System.out.println( "Starting thread..." );
        thread.start();
        Thread.sleep( 3000 );
        System.out.println( "Asking thread to stop..." );
        thread.stop = true;
        thread.socket.close();
        Thread.sleep( 3000 );
        System.out.println( "Stopping application..." );
        System.exit( 0 );
    }

    public void run() {
        try {
            socket = new ServerSocket(7856);
        } catch ( IOException e ) {
            System.out.println( "Could not create the socket..." );
            return;
        }
        while ( !stop ) {
            System.out.println( "Waiting for connection..." );
            try {
                Socket sock = socket.accept();
            } catch ( IOException e ) {
                System.out.println( "accept() failed or interrupted..." );
            }
        }
    }
}

```

Apr 07, 04 12:46

Example5.java

Page 2/2

```

    }
    System.out.println( "Thread exiting under request..." );
}
}

```