```
class thread implements Runnable {
  @Override
  public void run() {
    try {
      Thread.sleep(1500);
    } catch (InterruptedException e) {
    }
    System.out.println("State of thread 1-" + Test.thread1.getState());
    try {
      Thread.sleep(200);
    } catch (InterruptedException e) {
    }
  }
}


class Test implements Runnable {
  static Thread thread1;
  static Test obj;

  public static void main(String[] args) throws InterruptedException {
    obj = new Test();
    thread1 = new Thread(obj);
    System.out.println("thread 1 after create-" + thread1.getState());
    thread1.start();
    System.out.println("thread 1 after start" + thread1.getState());
  }

  @Override
  public void run() {
    Thread thread2 = new Thread(new thread());
    System.out.println("state of thread 2" + thread2.getState());
    thread2.start();
```

```
  }
}
=======================
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

class workerThread implements Runnable{
  private String message;
  public  workerThread(String s){
     this.message=s;
  }

  @Override
  public void run() {

System.out.println(Thread.currentThread().getName()+"---"+message);
// Thread-0
      processMessage();
      System.out.println(Thread.currentThread().getName());
  }

  private void processMessage() {
     try {
        Thread.sleep(2000);
     } catch (InterruptedException e) {
        e.printStackTrace();
     }
  }
  public static void main(String[] args) {
     ExecutorService executorService=
Executors.newFixedThreadPool(5);
     for (int i = 0; i <10 ; i++) {
        Runnable worker= new workerThread("");
        executorService.execute(worker);
```

```
        }
        executorService.shutdown();
        while (!executorService.isTerminated()){}
        System.out.println("Finish");
    }
}

======================
class t1 extends Thread{
    @Override
    public void run() {

        for (int i = 0; i < 10; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 1");
        }
    }
}
class t2 extends Thread{
    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 2");
        }
    }
```

```
}

class Demo{
  public static void main(String[] args) {
     // 0.-5-.10
     System.out.println("Main start");
     t1 myThread1=new t1();
     t2 myThread2=new t2();

     myThread1.setPriority(10); // Linux
     System.out.println(myThread1.getPriority());
     System.out.println(myThread2.getPriority());


     myThread1.start();
     myThread2.start();

     System.out.println("Main End");
  }
}
================================
class t1 extends Thread{
  @Override
  public void run() {

     for (int i = 0; i < 10; i++) {
        try {
           Thread.sleep(500);
        } catch (InterruptedException e) {
           e.printStackTrace();
        }
        System.out.println("Thread 1");
     }
  }
```

```
}
class t2 extends Thread{
  @Override
  public void run() {
    for (int i = 0; i < 10; i++) {
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      System.out.println("Thread 2");
    }
  }
}

class Demo{
  public static void main(String[] args) {
    // 0.-5-.10
    System.out.println("Main start");
    t1 myThread1=new t1();
    t2 myThread2=new t2();

    System.out.println(myThread1.getName());
    System.out.println(myThread2.getName());


    myThread1.start();
    myThread2.start();

    System.out.println("Main End");
  }
}
```

```java
// 10%
============================
class t1 extends Thread{
  @Override
  public void run() {

    for (int i = 0; i < 10; i++) {
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      System.out.println("Thread 1");
    }
  }
}
class t2 extends Thread{
  @Override
  public void run() {
    for (int i = 0; i < 10; i++) {
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      System.out.println("Thread 2");
    }
  }
}

class Demo{
  public static void main(String[] args) {
    // 0.-5-.10
    System.out.println("Main start");
```

```
        t1 myThread1=new t1();
        t2 myThread2=new t2();

        myThread1.setName("55");
        myThread2.setName("56");

        System.out.println(myThread1.getName());
        System.out.println(myThread2.getName());


        myThread1.start();
        myThread2.start();

        System.out.println("Main End");
    }
}

// 10%
================================================
class t1 extends Thread {
    @Override
    public void run() {

        for (int i = 0; i < 10; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 1");
        }
    }
}
```

```java
class t2 extends Thread {
    @Override
    public void run() {
        for (int i = 0; i < 10; i++) {
            try {
                Thread.sleep(500);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            System.out.println("Thread 2");
        }
    }
}

class Demo {
    public static void main(String[] args) throws InterruptedException {
        System.out.println("Main start");
        t1 myThread1 = new t1();
        t2 myThread2 = new t2();

        myThread1.start();
        myThread2.start();
        /*ddgdfgdfgfg*/
        myThread1.join();
        myThread2.join();
        System.out.println(Thread.currentThread().isAlive());
        System.out.println(myThread1.isAlive());


        System.out.println("Main End");
    }
}
```

```java
// 10%
==========================================
class t1 extends Thread {
  @Override
  public void run() {

    for (int i = 0; i < 10; i++) {
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      System.out.println("Thread 1");
    }
  }
}

class t2 extends Thread {
  @Override
  public void run() {
    for (int i = 0; i < 10; i++) {
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
      }
      System.out.println("Thread 2");
    }
  }
}

class Demo {
  public static void main(String[] args) throws InterruptedException {
    System.out.println("Main start");
```

```
        t1 myThread1 = new t1();
        t2 myThread2 = new t2();

        myThread1.start();// user Thread
        myThread2.start();// user Thread
        myThread1.join();
        myThread2.join();

        myThread1.setDaemon(true);

        System.out.println(myThread1.isDaemon());
        System.out.println(myThread2.isDaemon());


        System.out.println("Main End");
    }
}

// 10%
////////////////////////////////////////////
What is a Thread
------
Thread is a lightweight sub process
it is a smallest independent unit of a program
every java program contains at least one thread
----------------------
To crate a Thread---> Option 2
1---> Extend the Thread Class
2---> Implements Runnable Interface(lamdas)

=================================================
=========================================
Thread Class                                    Runnable Interface
```

*Each Thread creates its unique Object *Each Thread creates its unique Object
*A class extending thread class cant any other class *Along with runnable a class can implements any other interface
* Enable Tight Couple * Enables loose Couple

//==== Java Main Thread

*it is executed whenever a program starts

-====================================

```
class LambdaThread{
  public static void main(String[] args) {
    // child thread
    new Thread(()->{
      for (int i = 0; i < 10; i++) {
        System.out.println("Thread Child "+i);
        try {
          Thread.sleep(500);
        } catch (InterruptedException e) {
          e.printStackTrace();
        }
      }
    }).start();

    // main thread
    for (int i = 0; i < 10; i++) {
      System.out.println("Thread Main"+i);
      try {
        Thread.sleep(500);
      } catch (InterruptedException e) {
        e.printStackTrace();
```

```
            }
        }

    }
}
```