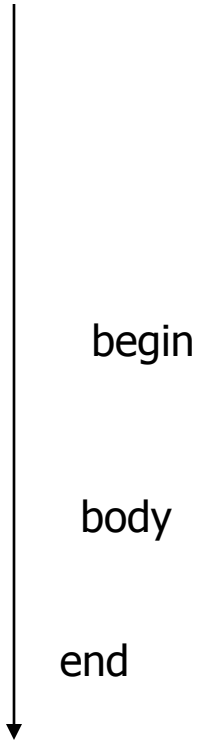
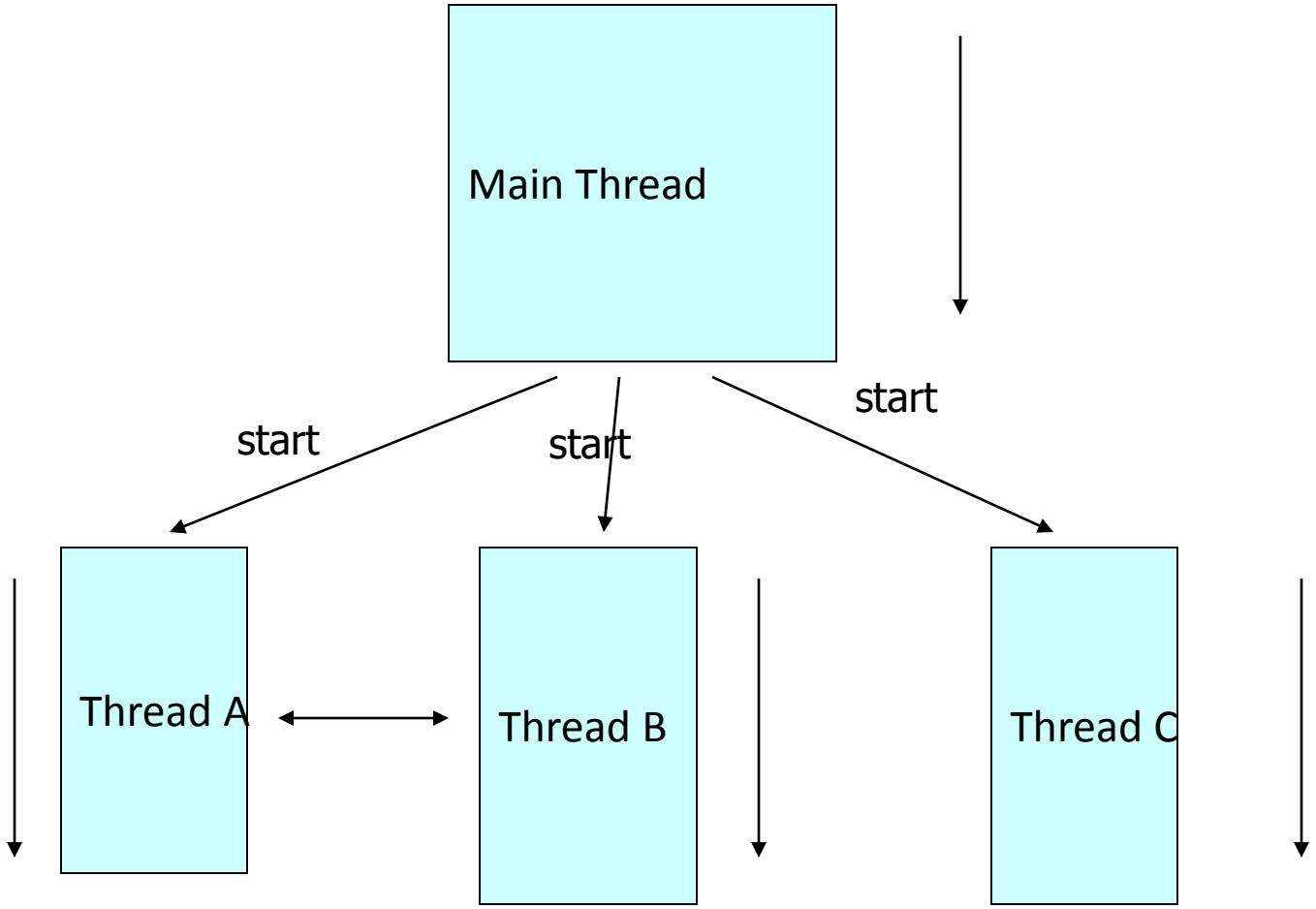
Java Threads

A single threaded program

```
class ABC
{
....
      public void main(..)
      {
      …
      ..
      }
}
```



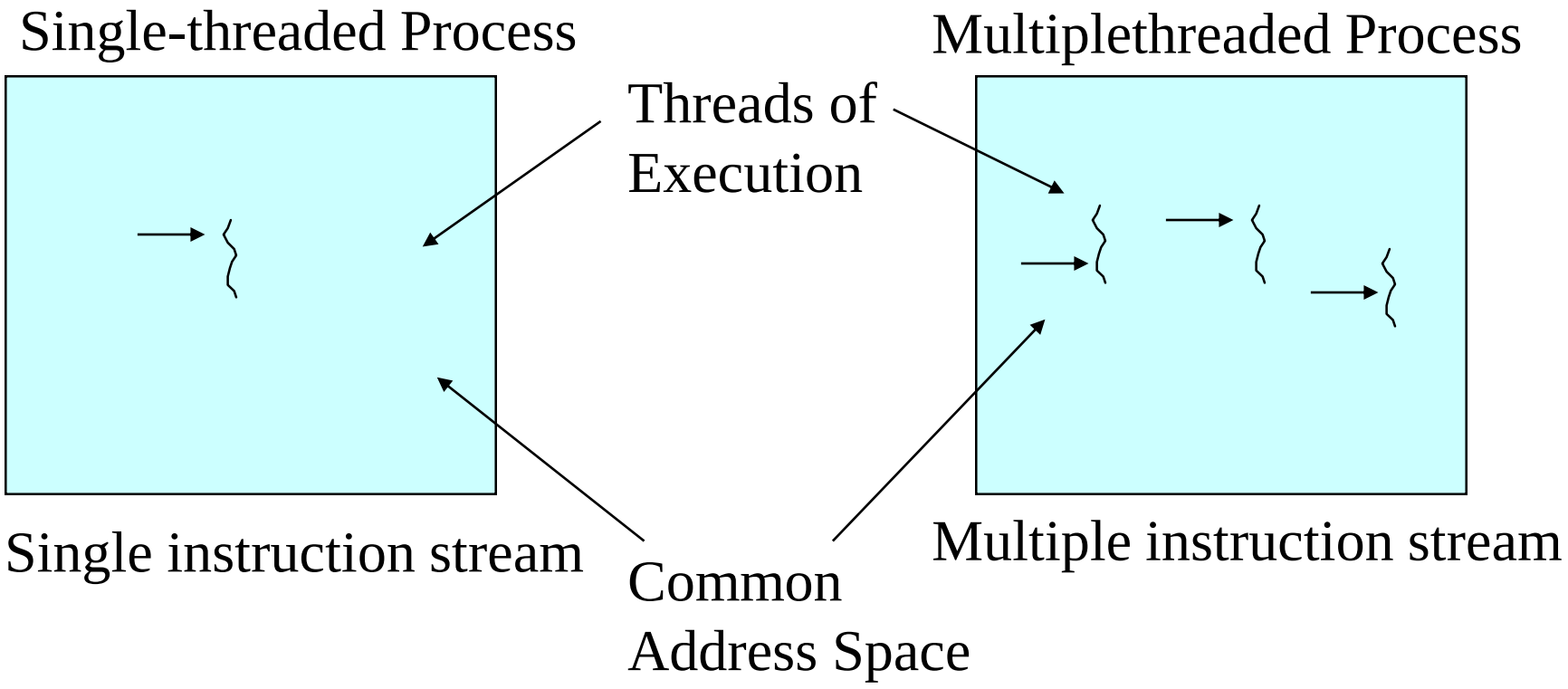
A Multithreaded Program



Threads may switch or exchange data/results

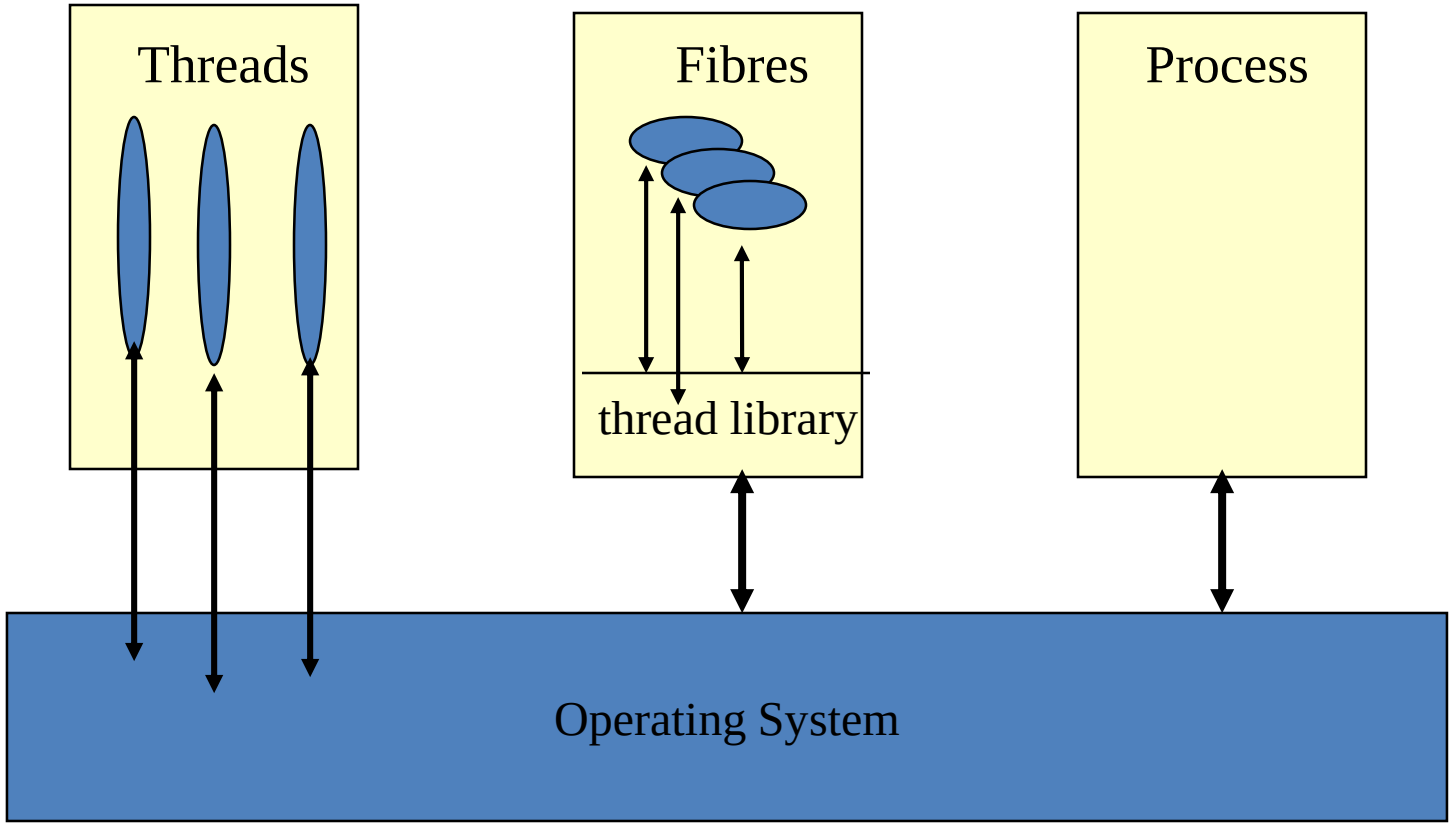
Single and Multithreaded Processes

threads are light-weight processes within a process



Java Concurrency Models

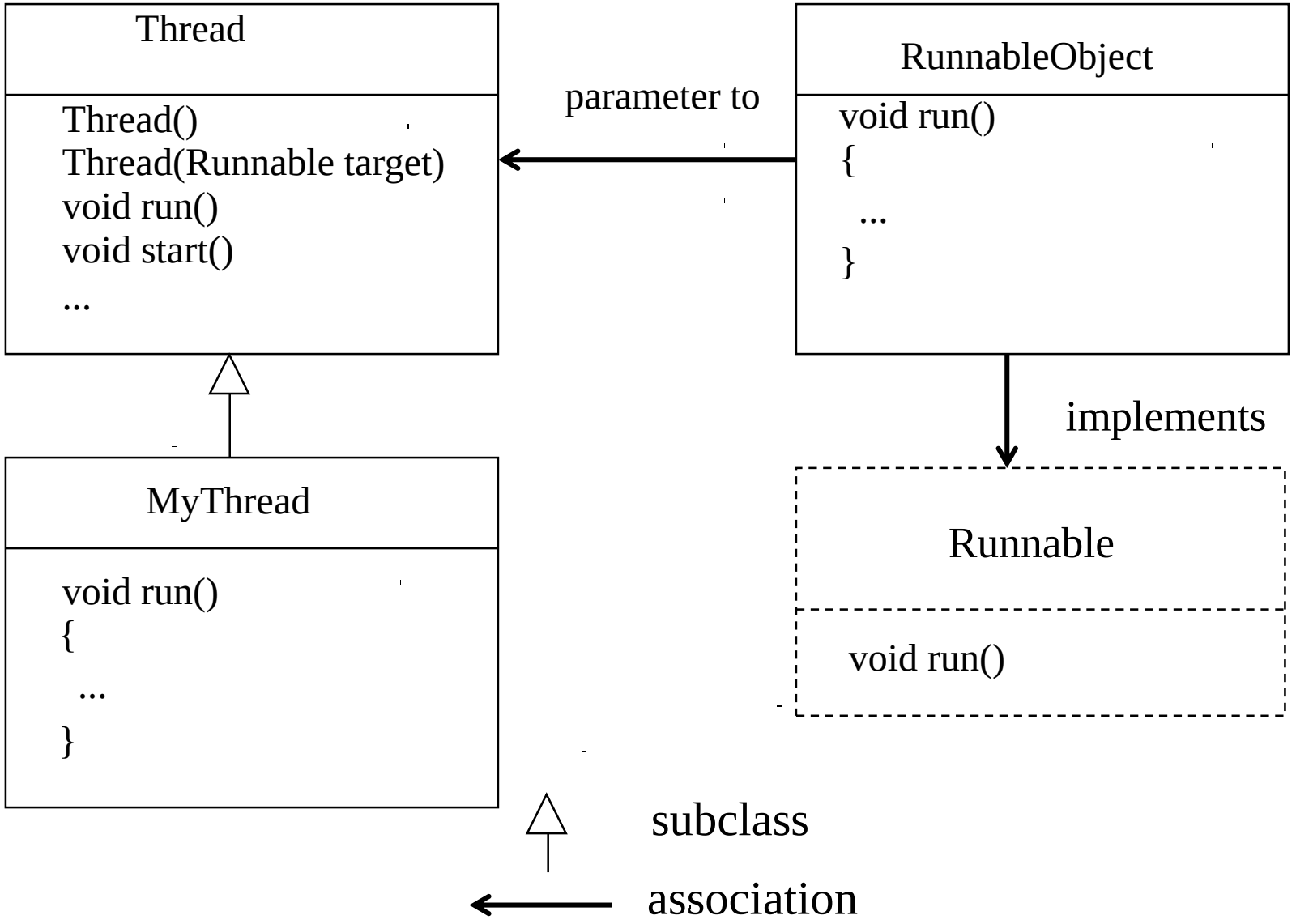
- Processes versus Threads



Java Concurrency Models

- Java supports threads
  - Threads execute within a single JVM
  - Native threads map a single Java thread to an OS thread
  - Green threads adopt the thread library approach (threads are invisible to the OS)
  - On a multiprocessor system, native threads are required to get true parallelism (but this is still implementation dependent)

Threads in Java



The Thread Class

```
public class Thread extends Object
                implements Runnable {
  public Thread();
  public Thread(String name);
  public Thread(Runnable target);
  public Thread(Runnable target,
                String name);
  public Thread(Runnable target,
          String name, long stackSize);

  public void run();
  public void start();
  ...
}
```

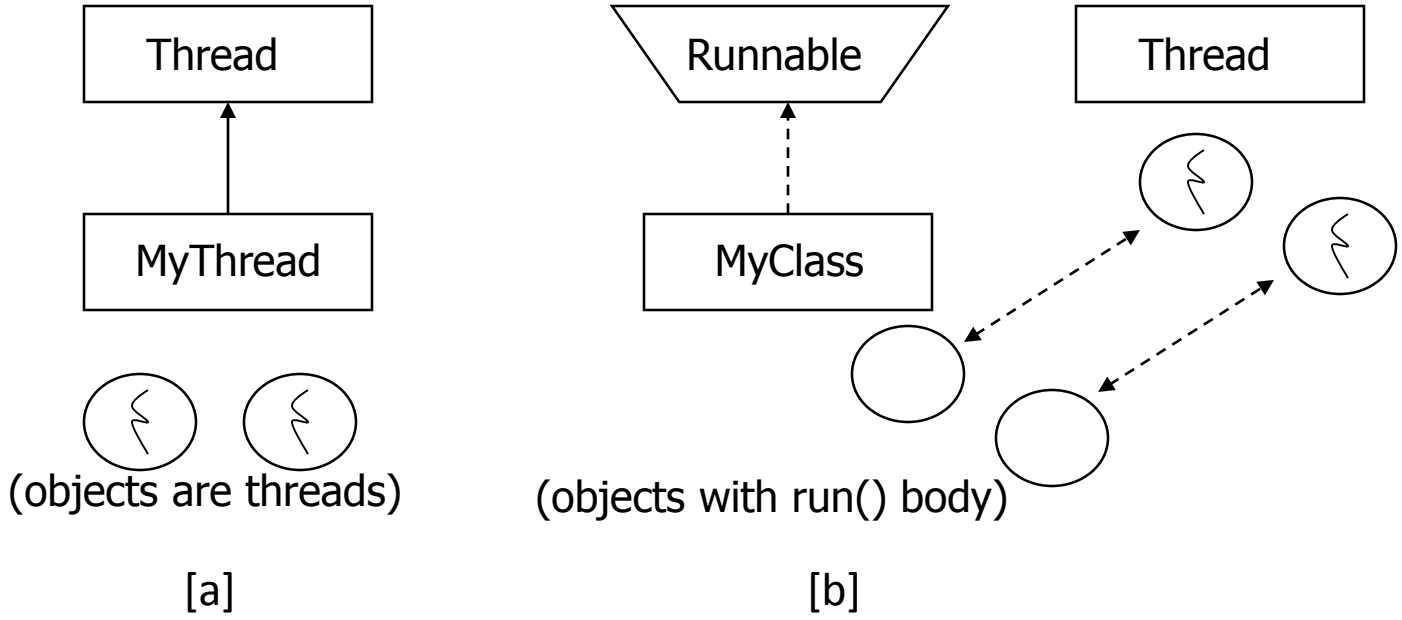
Thread Creation

1. Extend `Thread` class and override the **`run`** method, or

2. Create an object which implements the `Runnable` interface and pass it to a `Thread` object via the `Thread` constructor

Thread Creation

- Create a class that extends the Thread class
- Create a class that implements the Runnable interface



1st method: Extending Thread class

- Create a class by extending Thread class and override run() method:

```
class MyThread extends Thread
{
     public void run()
    {
       // thread body of execution
    }
}
```

- Create a thread:

```
MyThread thr1 = new MyThread();
```

- Start Execution of threads:

```
thr1.start();
```

- Create and Execute:

```
new MyThread().start();
```

An example

```
class MyThread extends Thread {
        public void run() {
                System.out.println(" this thread is running ... ");
        }
}

class ThreadEx1 {
        public static void main(String [] args  ) {
                MyThread t = new MyThread();
                 t.start();
       }
}
```

2nd method: Threads by implementing Runnable interface

- Create a class that implements the interface Runnable and override run() method:

```
class MyThread implements Runnable
{
   .....
   public void run()
   {
      // thread body of execution
   }
}
```

- Creating Object:

```
MyThread myObject = new MyThread();
```

- Creating Thread Object:

```
Thread thr1 = new Thread( myObject );
```

- Start Execution:

```
thr1.start();
```

An example

```
class MyThread implements Runnable  {
        public void run() {
                System.out.println(" this thread is running ... ");
        }
}

class ThreadEx2 {
        public static void main(String [] args  ) {
                Thread t = new Thread(new MyThread());
                 t.start();
        }
}
```

Warning

The run method should not be called directly by the application. The system calls it.

If the run method is called explicitly by the application then the code is executed sequentially not concurrently

A Program with Three Java Threads

- Write a program that creates 3 threads

Three threads example

```
class A extends Thread
{
      public void run()
       {
           for(int i=1;i<=5;i++)
            {
                System.out.println("\t From ThreadA:
i= "+i);
            }
             System.out.println("Exit from A");
       }
}
```

```
class B extends Thread

{
public void run()
       {for(int j=1;j<=5;j++)
            {
                System.out.println("\t From
ThreadB: j= "+j);
            }
             System.out.println("Exit from
B");
       }
}
```

```
class C extends Thread
{      public void run()
       {       for(int k=1;k<=5;k++)
            {
                System.out.println("\t From
               ThreadC: k= "+k);
            }
             System.out.println("Exit from C");
       }
}
```

```
class ThreadTest
{
        public static void main(String args[])
         {
A Aobj = new A();
B Bobj= new B();
C Cobj = new C();
       Aobj.start();
       Bobj.start();
       Cobj..start();
         }
}
```