Multithreading in Java:

Multithreading in Java allows you to execute multiple threads concurrently, which can help improve the performance of your applications, especially on multi-core processors. Here are two code examples to demonstrate multithreading in Java:

Example 1: Creating Threads by Extending Thread Class

In this example, we'll create two threads by extending the `Thread` class and overriding the `run()` method.

```java
class MyThread extends Thread {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(Thread.currentThread().getId() + " Value " + i);
        }
    }
}

public class ThreadExample1 {
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        MyThread t2 = new MyThread();

        t1.start();
        t2.start();
    }
}
```

In this code, we create two threads `t1` and `t2` by extending the `Thread` class. We override the `run()` method to specify the code that each thread should execute. We then start both threads using the `start()` method.

Example 2: Creating Threads by Implementing Runnable Interface

In this example, we'll create two threads by implementing the `Runnable` interface.

```java
class MyRunnable implements Runnable {
    public void run() {
        for (int i = 1; i <= 5; i++) {
            System.out.println(Thread.currentThread().getId() + " Value " + i);
        }
    }
}

public class ThreadExample2 {
    public static void main(String[] args) {
        MyRunnable myRunnable = new MyRunnable();

        Thread t1 = new Thread(myRunnable);
        Thread t2 = new Thread(myRunnable);

        t1.start();
        t2.start();
    }
}
```

In this code, we create a `MyRunnable` class that implements the `Runnable` interface and overrides the `run()` method. We then create two `Thread` instances, `t1` and `t2`, passing the `MyRunnable` object to their constructors. Finally, we start both threads using the `start()` method.

Both examples demonstrate the use of multithreading in Java, allowing you to run multiple threads concurrently. Depending on the system and execution order, you may see interleaved output from both threads, indicating concurrent execution.