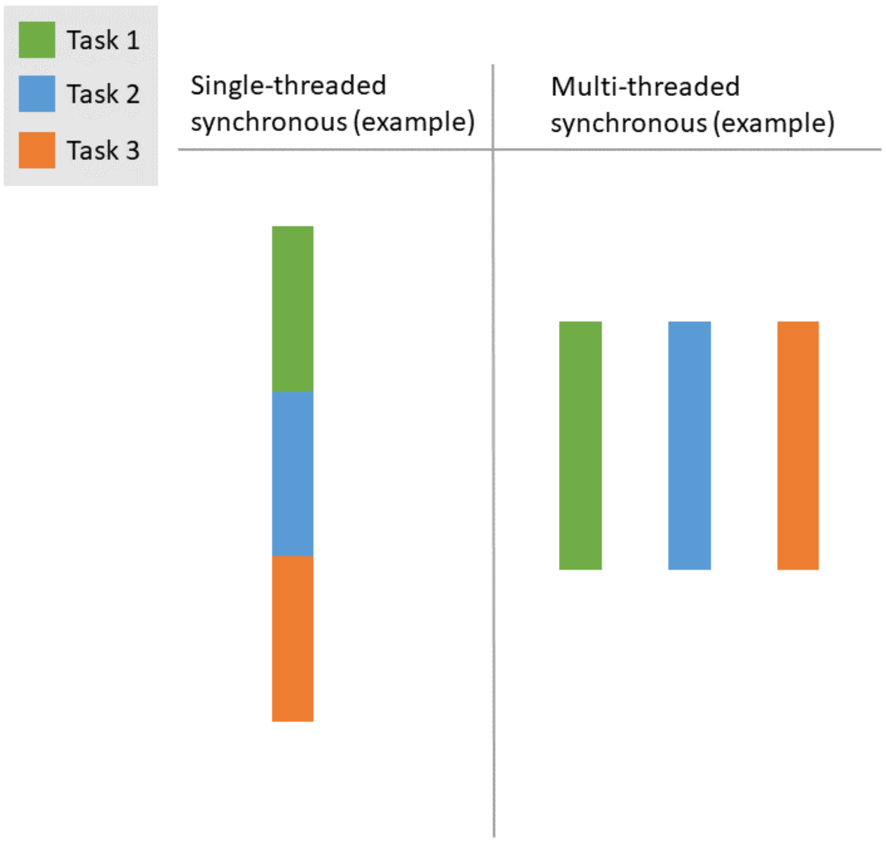
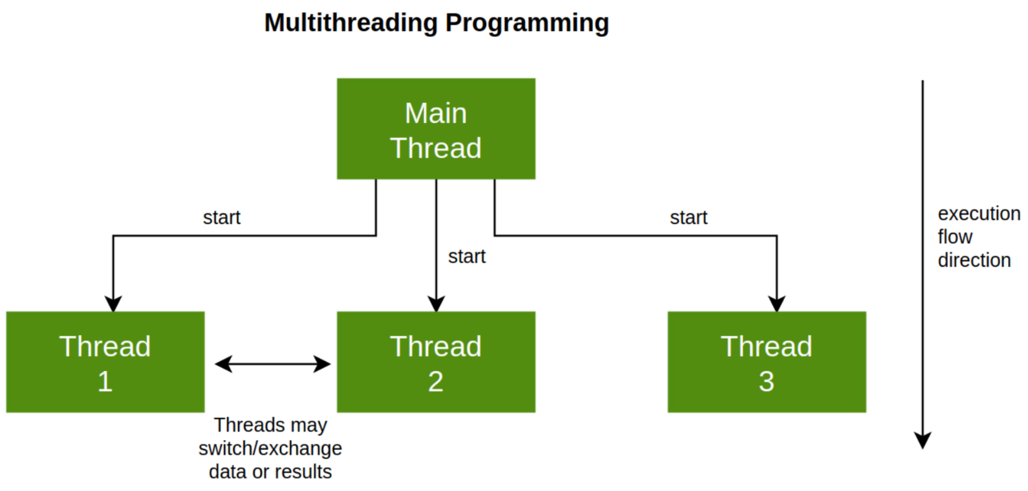
# Lecture 23

## Multithreading

- Multithreading in Java is a process of executing multiple threads simultaneously.
- A thread is a lightweight sub-process, the smallest unit of processing.
- A multi-threaded program contains two or more parts that can run concurrently and each part can handle a different task at the same time to maximum utilization of CPU.

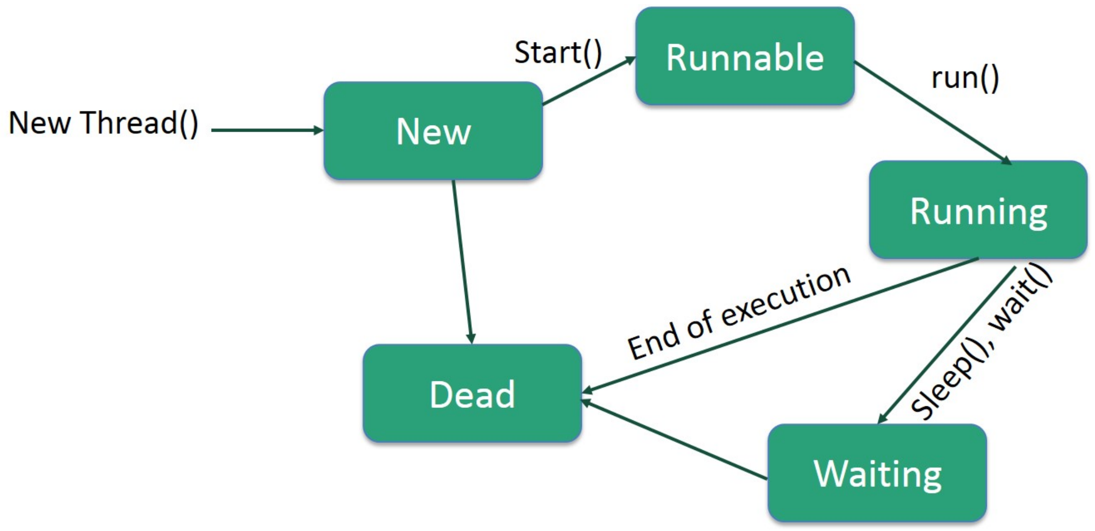

🟩 Task 1
🟦 Task 2
🟧 Task 3
Single-threaded synchronous (example)
Multi-threaded synchronous (example)



## Advantages of Java Multithreading

- It **doesn't block the user** because threads are independent and you can perform multiple operations at the same time.
- You **can perform many operations together, so it saves time**.
- Threads are **independent**, so it doesn't affect other threads if an exception occurs in a single thread.

## Life Cycle of Thread



1. **New:** In this phase, the thread is created using class "Thread class".It remains in this state till the program **starts** the thread. It is also known as born thread.
2. **Runnable:** In this phase, the instance of the thread is invoked with a start method.
3. **Running:** When the thread starts executing, then the state is changed to "running" state.
4. **Waiting:** This is the state when a thread has to wait.
5. **Dead:** This is the state when the thread is terminated.

**<u>Threads can be created by using two mechanisms:</u>**

- Extending the Thread class
- Implementing the Runnable Interface

## Extending Thread

```
public class MultithreadingDemo extends Thread {
    public void run() {
        System.out.println("Running " + Thread.currentThread().getId());
        try {
            for (int i = 0; i < 2; i++) {
                System.out.println(Thread.currentThread().getId() + ": " + i);
                // Let the thread sleep for a while.
                Thread.sleep(50);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread " + Thread.currentThread().getId() + " interrupted.");
        }
        System.out.println("Thread " + Thread.currentThread().getId()+ " exiting.");
    }
}
```

```
public class MultithreadTest {

    public static void main(String[] args) {
        for (int i = 0; i < 3; i++) {
            MultithreadingDemo object = new MultithreadingDemo();
            object.start();
        }
    }
}
```

## Output

```
Running 10
Running 12
Running 11
12: 0
10: 0
11: 0
10: 1
11: 1
12: 1
Thread 10 exiting.
Thread 12 exiting.
Thread 11 exiting.
```

## Implementing Runnable

```java
public class RunnableDemo implements Runnable {
    private Thread thread;
    private String threadName;

    public RunnableDemo(String threadName) {
        this.threadName = threadName;
        System.out.println("Creating " + this.threadName);
    }

    public void run() {
        System.out.println("Running " + this.threadName);
        try {
            for (int i = 0; i < 5; i++) {
                System.out.println(this.threadName + ": " + i);
                // Let the thread sleep for a while.
                Thread.sleep(50);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread " + this.threadName + " interrupted.");
        }
        System.out.println("Thread " + this.threadName + " exiting.");
    }

    public void start() {
        System.out.println("Starting " + this.threadName);
        if (this.thread == null) {
            this.thread = new Thread(this, this.threadName);
            this.thread.start();
        }
    }
}
```

```java
public class ThreadTest {

    public static void main(String args[]) {
        RunnableDemo demo1 = new RunnableDemo( "Thread-1");
        demo1.start();

        RunnableDemo demo2 = new RunnableDemo( "Thread-2");
        demo2.start();
    }
}
```

## Output

```
Creating Thread-1
Starting Thread-1
Running Thread-1
Thread-1: 0
Creating Thread-2
Starting Thread-2
Running Thread-2
Thread-2: 0
Thread-1: 1
Thread-2: 1
Thread-1: 2
Thread-2: 2
Thread-1: 3
Thread-2: 3
Thread-1: 4
Thread-2: 4
Thread Thread-2 exiting.
Thread Thread-1 exiting.
```