* Why string is immutable in nature?

# Multithreading in Java

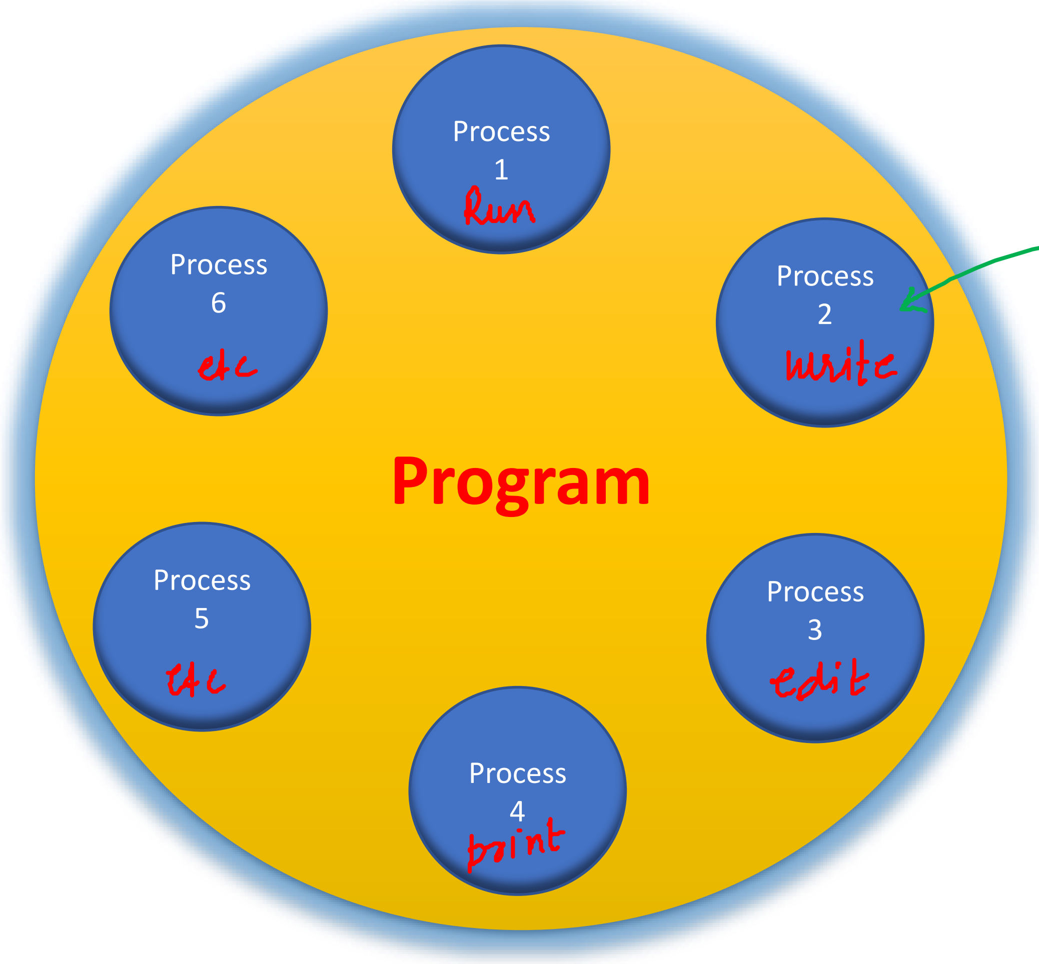
By Praveen Bhosle

# Before We start learning about Multithreading ….Lets start with some basic concepts

1. What is a program?
2. What is a process?
3. What is a thread?

# What is a program?

- Program is an executable file containing the set of instructions written to perform a specific job on your computer. For example, notepad.exe is an executable file containing the set of instructions which help us to edit and print the text files.

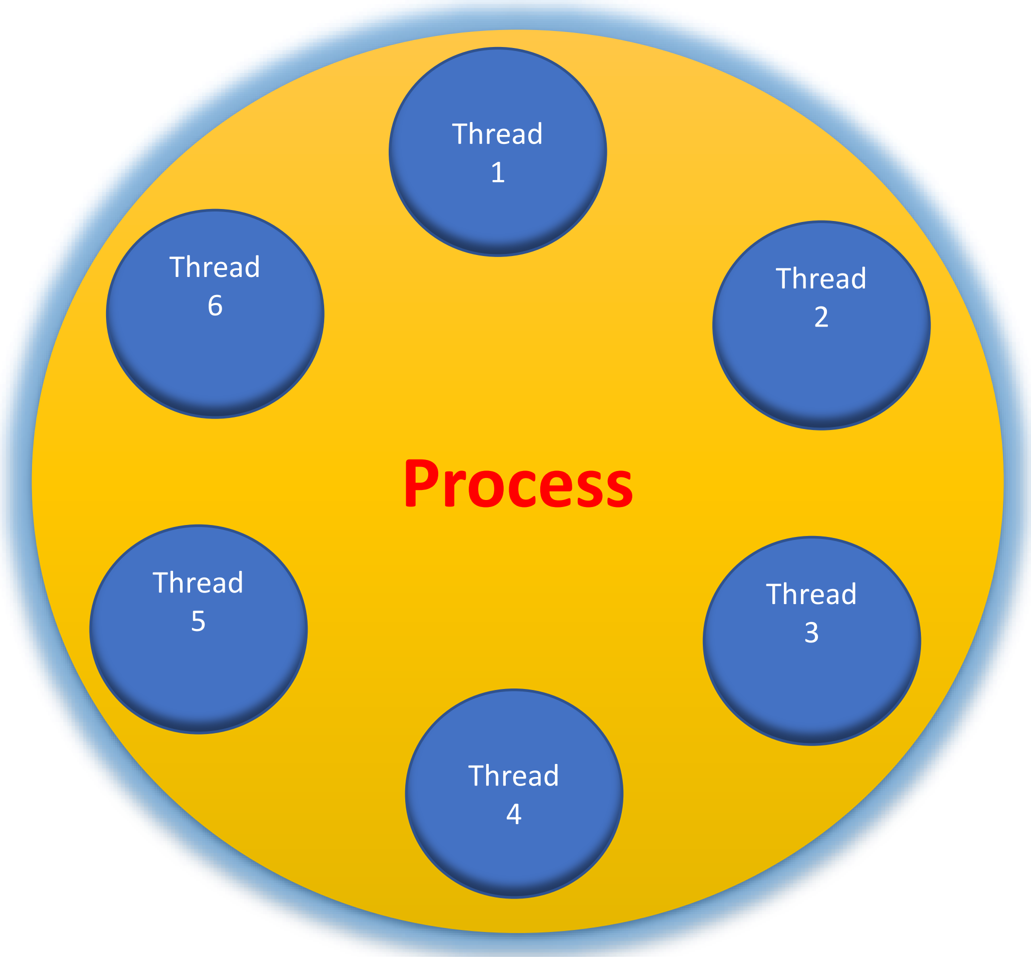


* They are set of Instructions

# What is a process?

- Process is an executing instance of a program. For example, when you double click on a notepad icon on your computer, a process is started that will run the notepad program.

Program
↓
process
↓
thread



Process

Thread 1

Thread 2

Thread 3

Thread 4

Thread 5

Thread 6

Note pad
↓
execute
↓
process.

# * What is a thread?

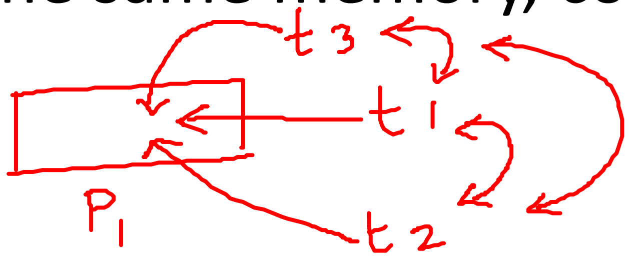
Thread → smallest unit of a program

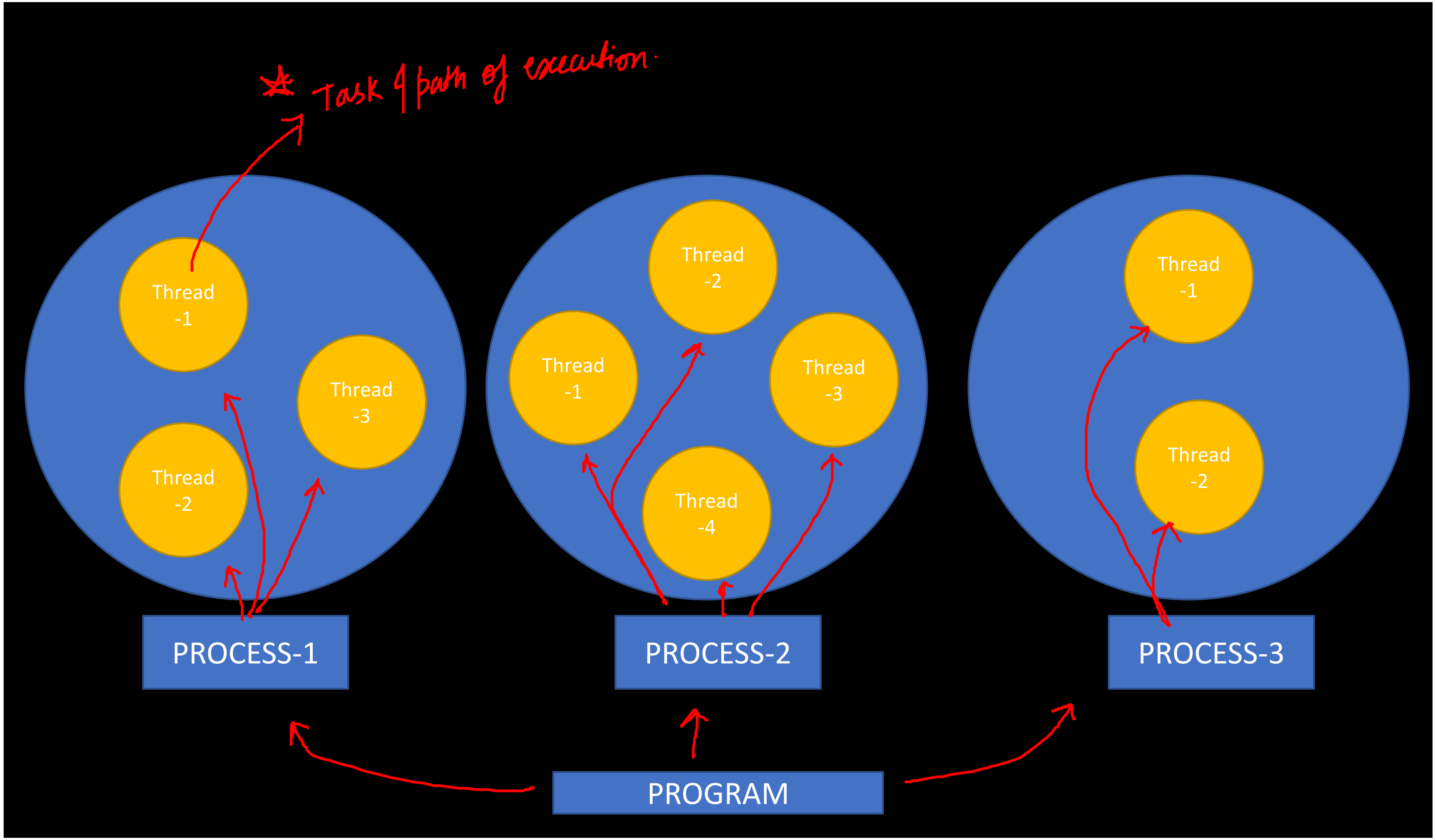
Keyboard-input → console-output.

program

process
↓
· has its main thread.

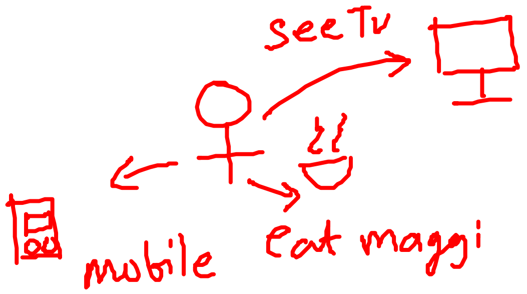
- Thread is the smallest executable unit of a process. For example, when you run a notepad program, operating system creates a process and starts the execution of main thread of that process. ← code.
- A process can have multiple threads. Each thread will have their own task and own path of execution in a process. For example, in a notepad program, one thread will be taking user inputs and another thread will be printing a document.
- All threads of the same process share memory of that process. As threads of the same process share the same memory, communication between the threads is fast.



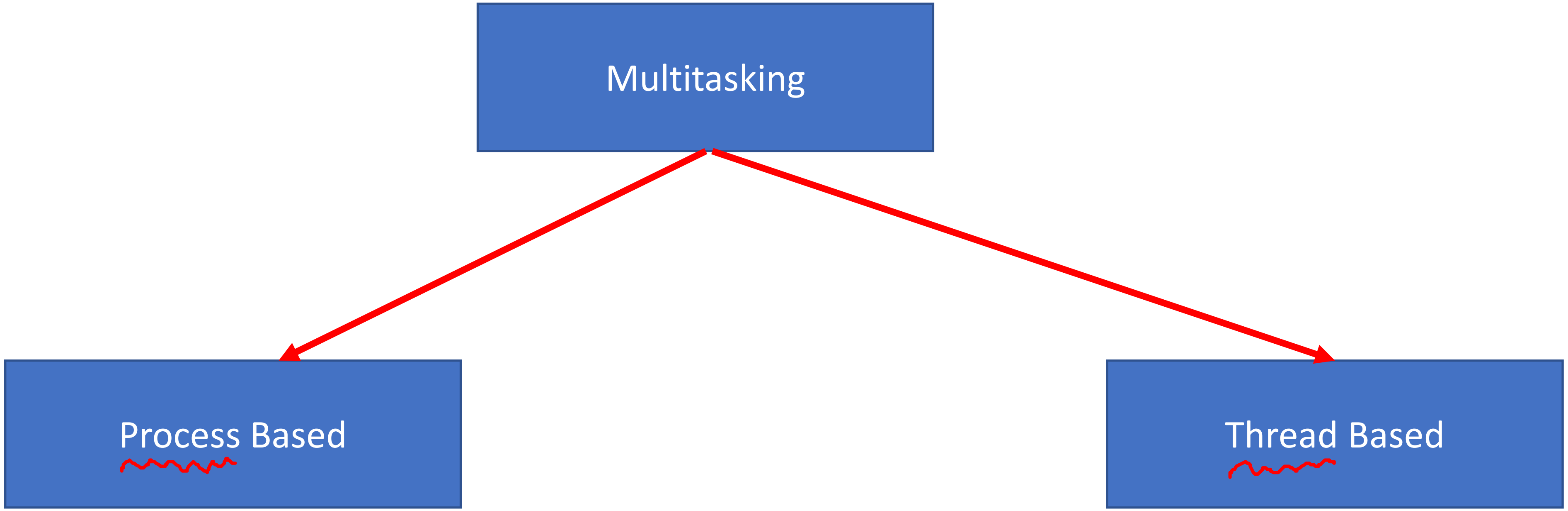

* Task / path of execution.
Thread -1
Thread -3
Thread -2
PROCESS-1
Thread -2
Thread -1
Thread -3
Thread -4
PROCESS-2
Thread -1
Thread -2
PROCESS-3
PROGRAM

# What is multitasking in java ?

- Multitasking means executing several tasks simultaneously.
- Advantage: It improves the performance of the system.
-



# Multitasking is achieved by using two ways.



Multitasking

Process Based

Thread Based

Cc → $P_1$

Music → $P_2$

chrome - $P_3$

# Process Based Multitasking (Multiprocessing)

1.Executing several tasks simultaneously where each task is separate & independent process such as multitasking is called as process based.

called as

2.**Example 1-** Typing java program into ① eclipse, also ② listening the audio songs, download a file from internet. ③

3.In this every activity is independent process here.

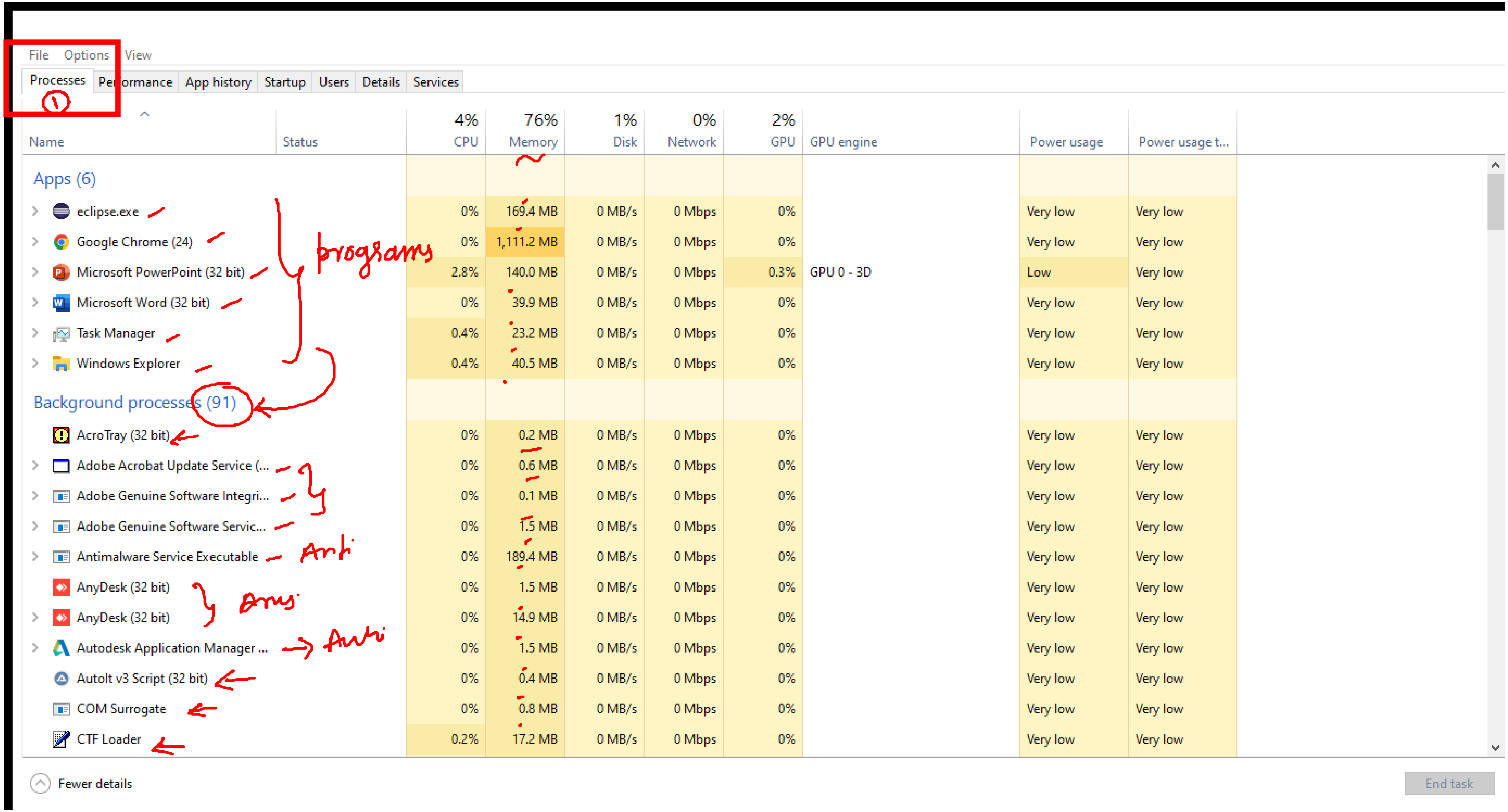
4.**Example-2** Task manager, see the multiple process list.

*5.Process is heavy weight components.

*6.Each process has address into memory.

process
↓
has memory
↓
shared with
threads of
same process

# Task Manager → ctr + Alt + delte



| Name | Status | 4%<br>CPU | 76%<br>Memory | 1%<br>Disk | 0%<br>Network | 2%<br>GPU | GPU engine | Power usage | Power usage t... |
|---|---|---|---|---|---|---|---|---|---|
| Apps (6) | | | | | | | | | |
| eclipse.exe | | 0% | 169.4 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Google Chrome (24) | | 0% | 1,111.2 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Microsoft PowerPoint (32 bit) | | 2.8% | 140.0 MB | 0 MB/s | 0 Mbps | 0.3% | GPU 0 - 3D | Low | Very low |
| Microsoft Word (32 bit) | | 0% | 39.9 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Task Manager | | 0.4% | 23.2 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Windows Explorer | | 0.4% | 40.5 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Background processes (91) | | | | | | | | | |
| AcroTray (32 bit) | | 0% | 0.2 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Adobe Acrobat Update Service (... | | 0% | 0.6 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Adobe Genuine Software Integri... | | 0% | 0.1 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Adobe Genuine Software Servic... | | 0% | 1.5 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Antimalware Service Executable | | 0% | 189.4 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| AnyDesk (32 bit) | | 0% | 1.5 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| AnyDesk (32 bit) | | 0% | 14.9 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| Autodesk Application Manager ... | | 0% | 1.5 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| AutoIt v3 Script (32 bit) | | 0% | 0.4 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| COM Surrogate | | 0% | 0.8 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |
| CTF Loader | | 0.2% | 17.2 MB | 0 MB/s | 0 Mbps | 0% | | Very low | Very low |

# Thread based Multitasking (Multithreading)

1.Executing several tasks simultaneously where each task is separate { is a part of same program called as thread based.

1000

500 500

$t_1$ $t_2$

2.Example- suppose I have 1000 lines of code into java program and it will takes 8 hours to execute it where first 500 line is executed after that remaining 500 lines is executed but there is no any dependency between them so I can run that tasks simultaneously to minimize the execution time.

* 3.Thread is light weight components.

4.Thread shares the same address space

process memory.

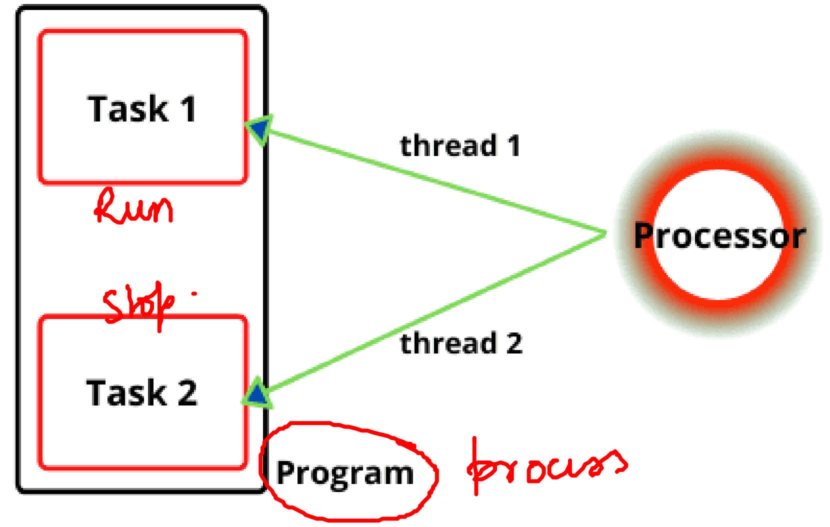


Fig: Thread-based Multitasking

Wrod

word $t_1$ $t_2$

# Realtime Example of Multithreading in Java

- 1. A very good example of thread-based multithreading is a word processing program that checks the spelling of words in a document while writing the document. This is possible only if each action is performed by a separate thread.
- 2. Another familiar example is a browser that starts rendering a web page while it is still downloading the rest of page.

Amzon

↓

✓

Ords $t_1$ tray.

custm

# What is thread in java ?

- A thread is the smallest unit of a program.

program
↓
process
↓
thread

Am ← AC
← I

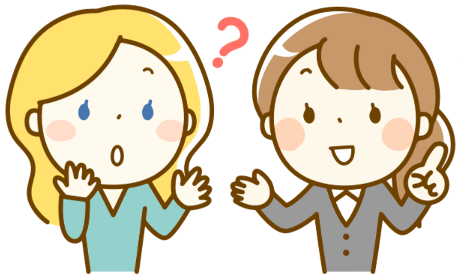
E ↓ ↓ I

override
↓
public

How can we create our own thread in java?

We can Create it in two ways
1.By extending Thread class ← ①
2.Implementing Runnable interface ← ②

JAVA



functional Interface
↓
It has only one abstract method.
→
nobody / noImplementation

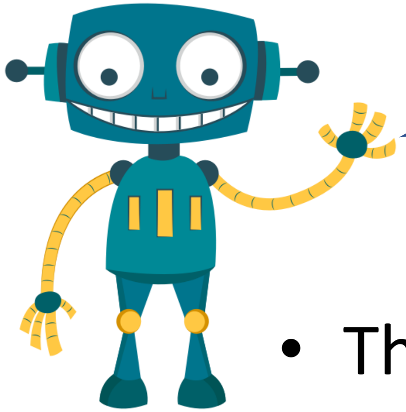
# 1.By extending Thread class ← parent class → object class

object class.

↑

* Thread class ←→ Runnable interface

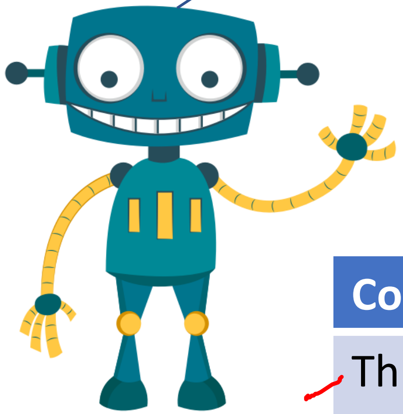
Hi



Hi I am Thread class

You can extend me with current class to create your own thread.

- Thread class provide constructors and methods to create and perform Operations on a thread. Thread class extends Object class and implements Runnable interface.

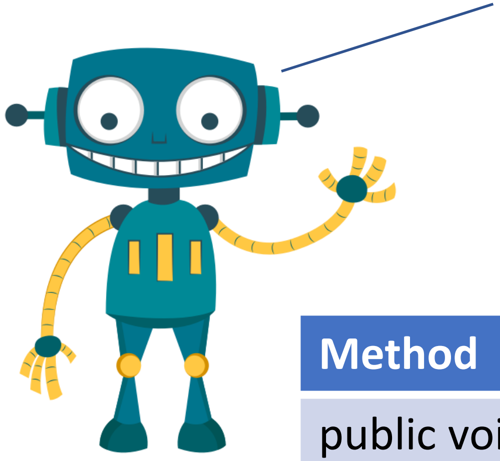


The various constructors of thread class are defined in java.lang package that can be used to create an object of thread are as follows:

Thread class

overloading
↳ no. of para
↳ types
↳ seque

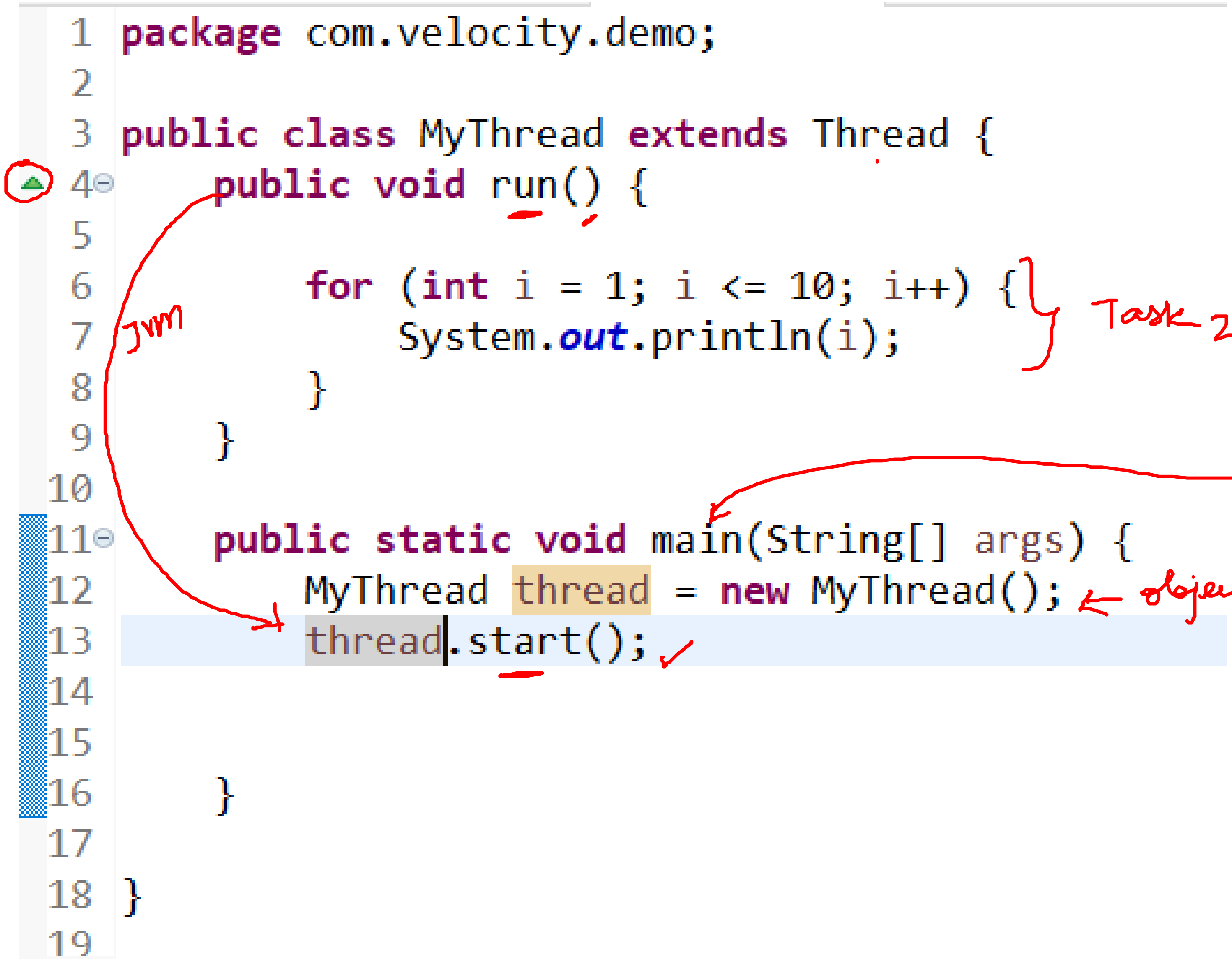
| Constructors | Description |
|---|---|
| Thread(): | This is a basic and default constructor without parameters. It simply creates an object of Thread class. |
| Thread(String name): | It creates a new thread object with specified name to a thread. |
| Thread(Runnable r): | It creates a thread object by passing a parameter r as a reference to an object of the class that implements Runnable interface. |
| Thread(Runnable r, String name): | This constructor creates a thread object by passing two arguments r and name. Here, variable r is a reference of an object of class that implements Runnable interface. |



## Methods of Thread Class in Java

| Method | Decsription |
|---|---|
| public void run(): ✓ ✱ | is used to perform action for a thread. |
| public void start(): ✓ ✱ | starts the execution of the thread.JVM calls the run() method on the thread. |
| public void sleep(long miliseconds): | Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds. |
| public void join(): | waits for a thread to die. |
| public void join(long miliseconds): | waits for a thread to die for the specified miliseconds. |
| public int getPriority(): | returns the priority of the thread. |
| public int setPriority(int priority): | changes the priority of the thread. |
| public String getName(): | returns the name of the thread. |
| public void setName(String name): | changes the name of the thread. |

# Coding….



```java
package com.velocity.demo;

public class MyThread extends Thread {
    public void run() {

        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        MyThread thread = new MyThread();
        thread.start();


    }

}
```

Jvm

Task 2

object

task 1

Jvm Thread main thread.

① thread → main thread

start default

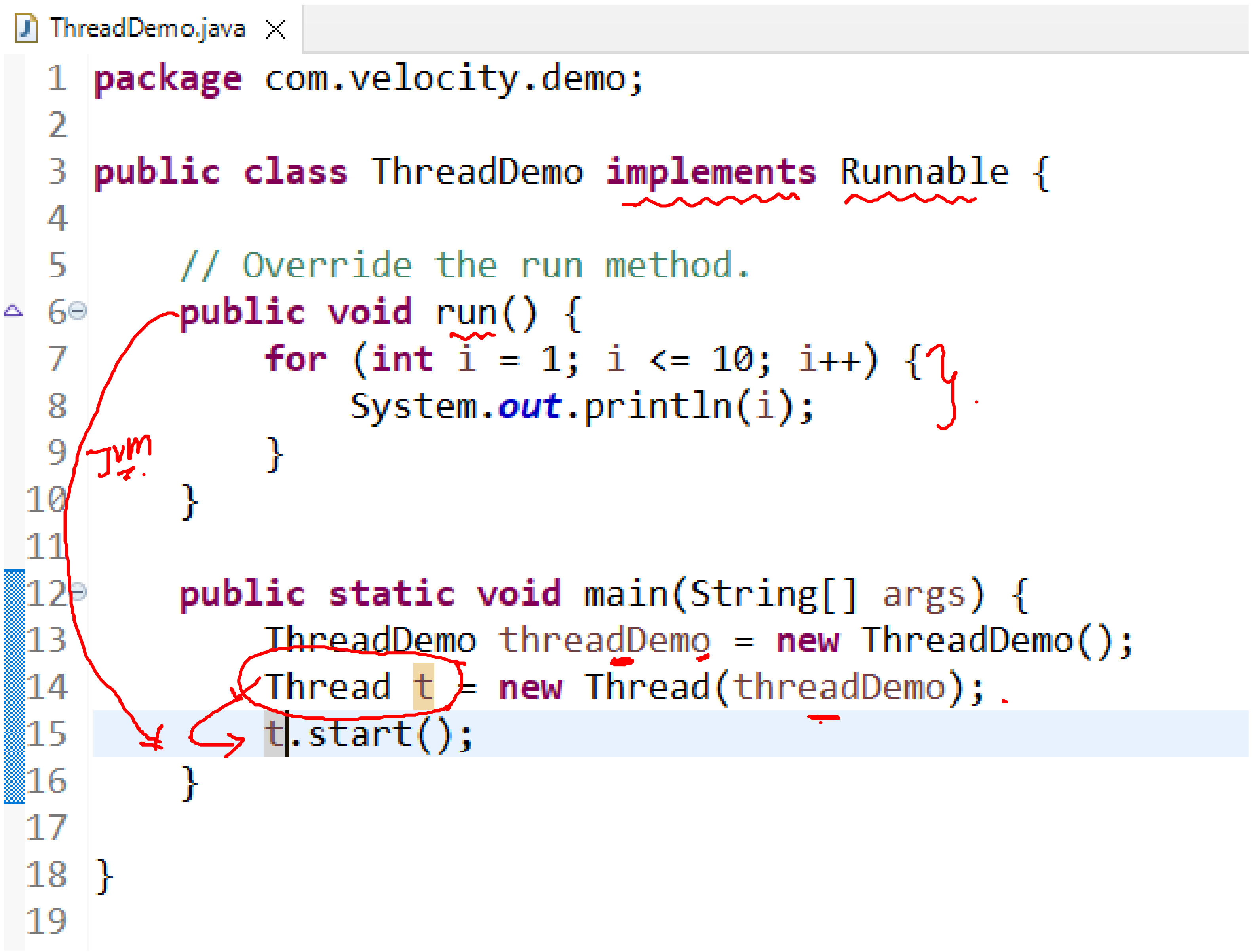
new thread

# 2.By implementing Runnable interface.

- The Runnable interface should be implemented by any class whose instances are intended to be executed by a thread. Runnable interface have only one method named run().

obj

# Coding...



```java
package com.velocity.demo;

public class ThreadDemo implements Runnable {

    // Override the run method.
    public void run() {
        for (int i = 1; i <= 10; i++) {
            System.out.println(i);
        }
    }

    public static void main(String[] args) {
        ThreadDemo threadDemo = new ThreadDemo();
        Thread t = new Thread(threadDemo);
        t.start();
    }

}
```

# Note-

- If you are not extending the Thread class, your class object would not be treated as a thread object. So you need to explicitly create Thread class object. We are passing the object of your class that implements Runnable so that your class run() method may execute.

# When to use ?

- **Extending thread class-** if the class is not extending another class then we should go for thread class. <Because of multiple inheritance>
- **Implementing runnable interface-** if our class is already extending another class then we could not use extend keyword due to multiple inheritance. So best way to go for runnable interface.

# Difference between Thread class and Runnable interface

| S.NO | KEY | THREAD CLASS | RUNNABLE INTERFACE |
|---|---|---|---|
| 1 | Basic | Thread is a class. It is used to create a thread | Runnable is a functional interface which is used to create a thread |
| 2 | Methods | It has multiple methods including start() and run() | It has only abstract method run() |
| 3 | | Each thread creates a unique object and gets associated with it | Multiple threads share the same objects. |
| 4 | Memory | More memory required | Less memory required |
| 5 | Limitation | Multiple Inheritance is not allowed in java hence after a class extends Thread class, it can not extend any other class | If a class is implementing the runnable interface then your class can extend another class. |

# Coding…..

# Pros:

- Better use of system resources.
- Parallel execution of tasks and thus less execution time
- Enhanced performance on multi-processor machines
- Improved GUI responsiveness
- Independent threads (don’t impact other threads of the same process if an exception occurs)
- Multithreading doesn’t always yield benefits. It comes with its disadvantages too.

# Cons:

- It results in the increased complexity of the code.
- Synchronization of shared resources (objects, data) is CPU/memory intensive.
- Debugging is hard because sometimes you can’t predict the results.
- Increased potential for deadlock occurrence.
- “Starvation” some of the threads may not be served with due to poor design.