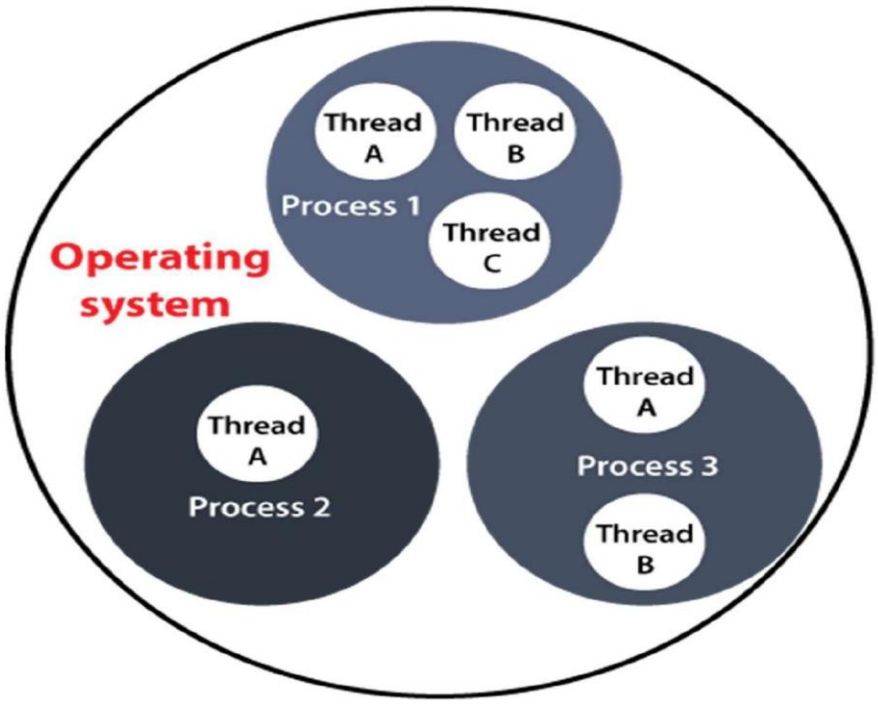
LIFE CYCLE OF A THREAD IN JAVA

A **Thread** is a very light-weighted process, or we can say the smallest part of the process that allows a program to operate more efficiently by running multiple tasks simultaneously.

In order to perform complicated tasks in the background, we used the **Thread concept in Java**. All the tasks are executed without affecting the main program. In a program or process, all the threads have their own separate path for execution, so each thread of a process is independent.



Another benefit of using **thread** is that if a thread gets an exception or an error at the time of its execution, it doesn't affect the execution of the other threads. All the threads share a common memory and have their own stack, local variables and program counter. When multiple threads are executed in parallel at the same time, this process is known as **Multithreading.**

In a simple way, a Thread is a:

- Feature through which we can perform multiple activities within a single process.
- Lightweight process.
- Series of executed statements.
- Nested sequence of method call

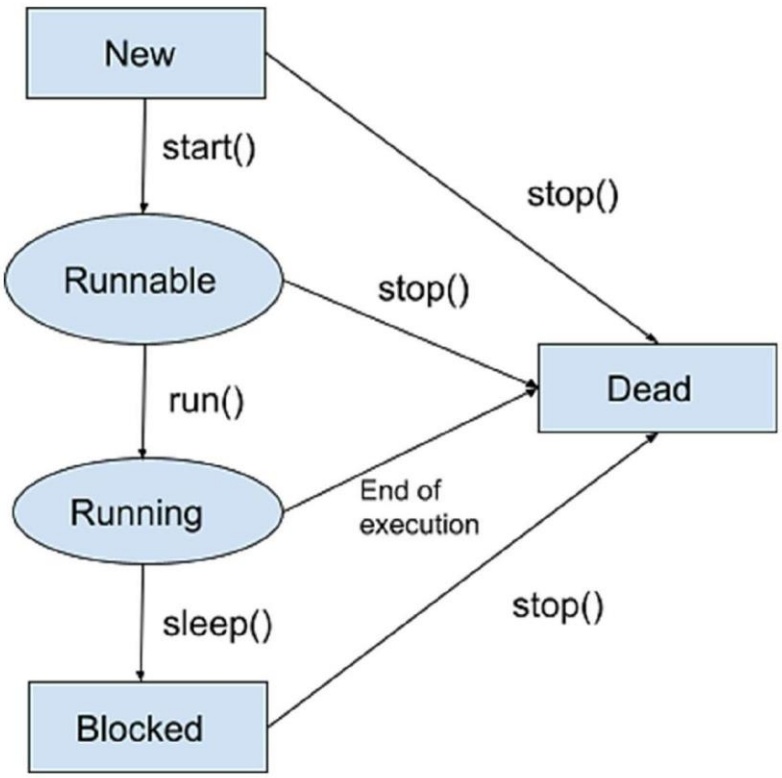
LIFE CYCLE OF A THREAD IN JAVA

The **Life Cycle of a Thread** in Java refers to the state transformations of a thread that begins with its birth and ends with its death. When a thread instance is generated and executed by calling the start() method of the Thread class, the thread enters the runnable state. When the sleep() or wait() methods of the Thread class are called, the thread enters a non-runnable mode.

Thread returns from non-runnable state to runnable state and starts statement execution. The thread dies when it exits the run() process. In Java, these thread state transformations are referred to as the Thread life cycle.

There are basically 4 stages in the lifecycle of a thread, as given below:

1. New
2. Runnable
3. Running
4. Blocked (Non-runnable state)
5. Dead



LIFE CYCLE OF A THREAD IN JAVA

New State:

As we use the Thread class to construct a thread entity, the thread is born and is defined as being in the New state. That is, when a thread is created, it enters a new state, but the start() method on the instance has not yet been invoked.

Runnable State:

A thread in the runnable state is prepared to execute the code. When a new thread's start() function is called, it enters a runnable state.

In the runnable environment, the thread is ready for execution and is awaiting the processor's availability (CPU time). That is, the thread has entered the queue (line) of threads waiting for execution.

Running State:

Running implies that the processor (CPU) has assigned a time slot to the thread for execution. When a thread from the runnable state is chosen for execution by the thread scheduler, it joins the running state.

In the running state, the processor allots time to the thread for execution and runs its run procedure. This is the state in which the thread directly executes its operations. Only from the runnable state will a thread enter the running state.

Blocked State:

When the thread is alive, i.e., the thread class object persists, but it cannot be selected for execution by the scheduler. It is now inactive.

Dead State:

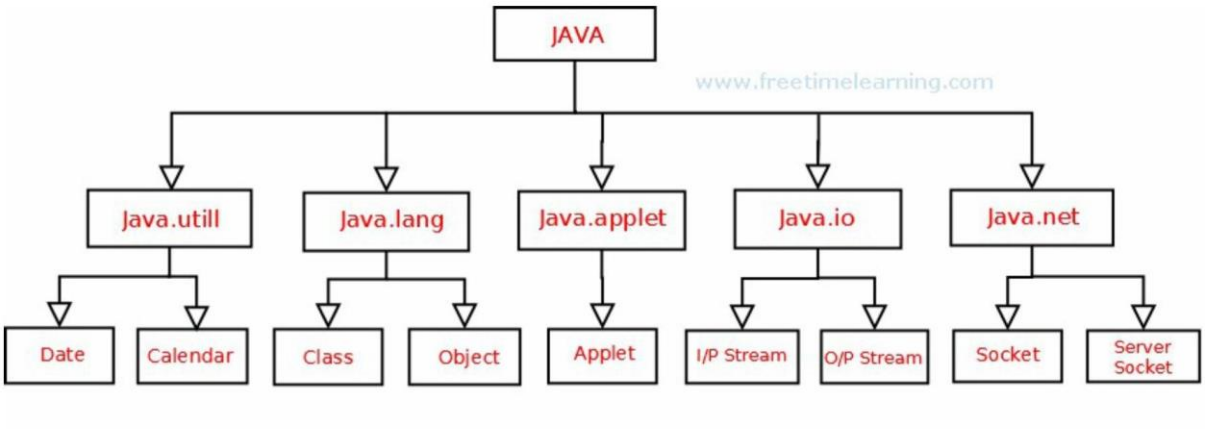
When a thread's run() function ends the execution of sentences, it automatically dies or enters the dead state. That is, when a thread exits the run() process, it is terminated or killed. When the stop() function is invoked, a thread will also go dead.

PACKAGES IN JAVA

Package:

Package in Java is a mechanism to encapsulate a group of classes, sub packages and interfaces.

Package in java can be categorized in two form, built-in package and user-defined package.



Built-in Packages in Java

USER DEFINED PACKAGE EXAMPLE

//Filename:Circle.java

```
package shapes;

import java.lang.Math;

public class Circle
{
   double radius;

   public Circle(double radius)
         {
      this.radius = radius;
   }

   public double area()
         {
      return Math.PI * radius * radius;
   }
}
```

PACKAGES IN JAVA

//Filename: Rectangle.java

```
package shapes;

public class Rectangle
{
   double width;
   double height;

   public Rectangle(double width, double height)
         {
      this.width = width;
      this.height = height;
   }

   public double area()
         {
      return width * height;
   }
}
```

//Filename: UserDefinedPackage.java

```
import shapes.Circle;
import shapes.Rectangle;

public class UserDefinedPackage
{
   public static void main(String args[])
   {
      Circle circle = new Circle(5.0);
      Rectangle rectangle = new Rectangle(4.0, 6.0);

      System.out.printf("Area of the circle: %.2f%n", circle.area());
      System.out.printf("Area of the rectangle: %.2f%n", rectangle.area());
   }
}
```

Output:

```
Area of the circle: 78.54
Area of the rectangle: 24.00
```

CALCULATOR GUI PROGRAM

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

class Calculator extends JFrame
{
   private JButton add, subtract, multiply, divide;
   private JTextField num1, num2;
   private JLabel result;

   Calculator()
{
      setLayout(new GridBagLayout());
      GridBagConstraints c = new GridBagConstraints();

      // Adding components
      addComponent(new JLabel("1st: "), 0, 0, 1, c);
      num1 = new JTextField(10);
      addComponent(num1, 1, 0, 3, c);

      addComponent(new JLabel("2nd: "), 0, 1, 1, c);
      num2 = new JTextField(10);
      addComponent(num2, 1, 1, 3, c);

      add = new JButton("+");
      addComponent(add, 0, 2, 1, c);

      subtract = new JButton("-");
      addComponent(subtract, 1, 2, 1, c);

      multiply = new JButton("*");
      addComponent(multiply, 2, 2, 1, c);

      divide = new JButton("/");
      addComponent(divide, 3, 2, 1, c);

      result = new JLabel("");
      addComponent(result, 0, 4, 4, c);

      // Adding action listeners
      CalculatorAction a = new CalculatorAction();
      add.addActionListener(a);
      subtract.addActionListener(a);
      multiply.addActionListener(a);
      divide.addActionListener(a);
   }

   private void addComponent(Component comp, int x, int y, int width, GridBagConstraints c) {
      c.fill = GridBagConstraints.HORIZONTAL;
      c.gridx = x;
      c.gridy = y;
```

```
        c.gridwidth = width;
        add(comp, c);
    }

    class CalculatorAction implements ActionListener
{
        public void actionPerformed(ActionEvent e)
{
            try {
                double number1 = Double.parseDouble(num1.getText());
                double number2 = Double.parseDouble(num2.getText());
                String op = e.getActionCommand();
                double res = 0;
                boolean valid = true;

                switch (op)
{
                    case "+":
                        res = number1 + number2;
                        break;
                    case "-":
                        res = number1 - number2;
                        break;
                    case "*":
                        res = number1 * number2;
                        break;
                    case "/":
                        if (number2 == 0) {
                            result.setText("Cannot divide by zero");
                            result.setForeground(Color.RED);
                            valid = false;
                        } else {
                            res = number1 / number2;
                        }
                        break;
                }
                if (valid)
{
                    result.setText(number1 + " " + op + " " + number2 + " = " + res);
                    result.setForeground(Color.BLACK);
                }
            }
catch (NumberFormatException ex)
{
                result.setText("Invalid input");
                result.setForeground(Color.RED);
            }
        }
    }

    public static void main(String[] args)
{
```

CALCULATOR GUI PROGRAM

```
        Calculator calc = new Calculator();
        calc.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        calc.setTitle("Calculator");
        calc.setVisible(true);
        calc.setSize(300, 200);
    }
}
```

COMPILATION AND EXECUTION:

javac Calculator.java
java Calculator

OUTPUT: