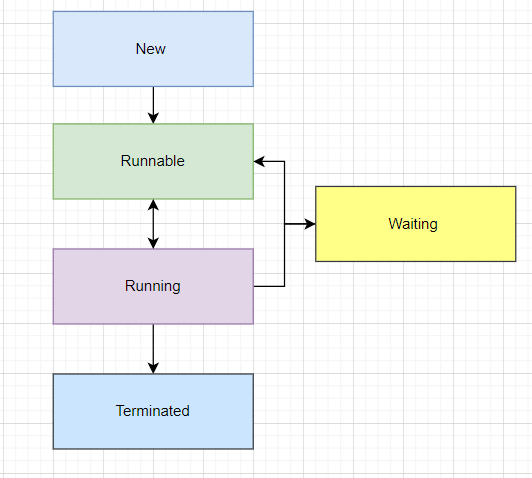
**What is a java thread ?**

Java thread is a lightweight sub process . In an application we have one main thread . When application is running operating system is going to create one process. Similarly , JVM will take care of the java application and a process will be created and each and every process will have a main thread.

Smallest Independent unit of a program and contains a separate path of the execution . Main thread is nothing just an execution context where the certain jobs will be executed one after another . Its is created and controlled by the java.lang.Thread class . No need to explicitly import it .

**Life Cycle of a Thread**



**New** : A new thread begins its life cycle in this state and remains here until the program starts the thread and is also known as born thread.

**Runnable** : Once a newly born thread starts , the thread comes under the runnable state. A thread stays in this state is until it is executing its task.

**Running** : it is executing a run method which is being overridden by the thread class. The yield () method can send them to go back to the runnable state .

**Waiting** : When a threads enters a state of inactivity temporarily . It is still alive but not eligible to run . It can be waiting , sleeping or blocked state . Maybe waiting for some thread to give back data . May be due to some network issues .

**Terminated** : A runnable thread enters the terminated state when it completes its task or otherwise terminates.

Analogy of playing songs in maybe a Spotify .

**Creating a thread :**

Code in the main method is executed as the main thread itself . All the instruction in main method are executed sequentially .

When we have long running operation in main method and it takes time and other operations are blocked . It is at that time when operating system starts giving messages to the users that this application is not responding would you like to wait or kill the applications.

We need to understand when we have a long running application then we might not put a load on the main method or main thread , instead create a separate thread to perform the operation .

A thread can be created in two ways. If a class is inheriting a class it cannot extends another class as java doesn

**Thread Class**

* public class Thread extends Object implements Runnable

**Runnable Interface**

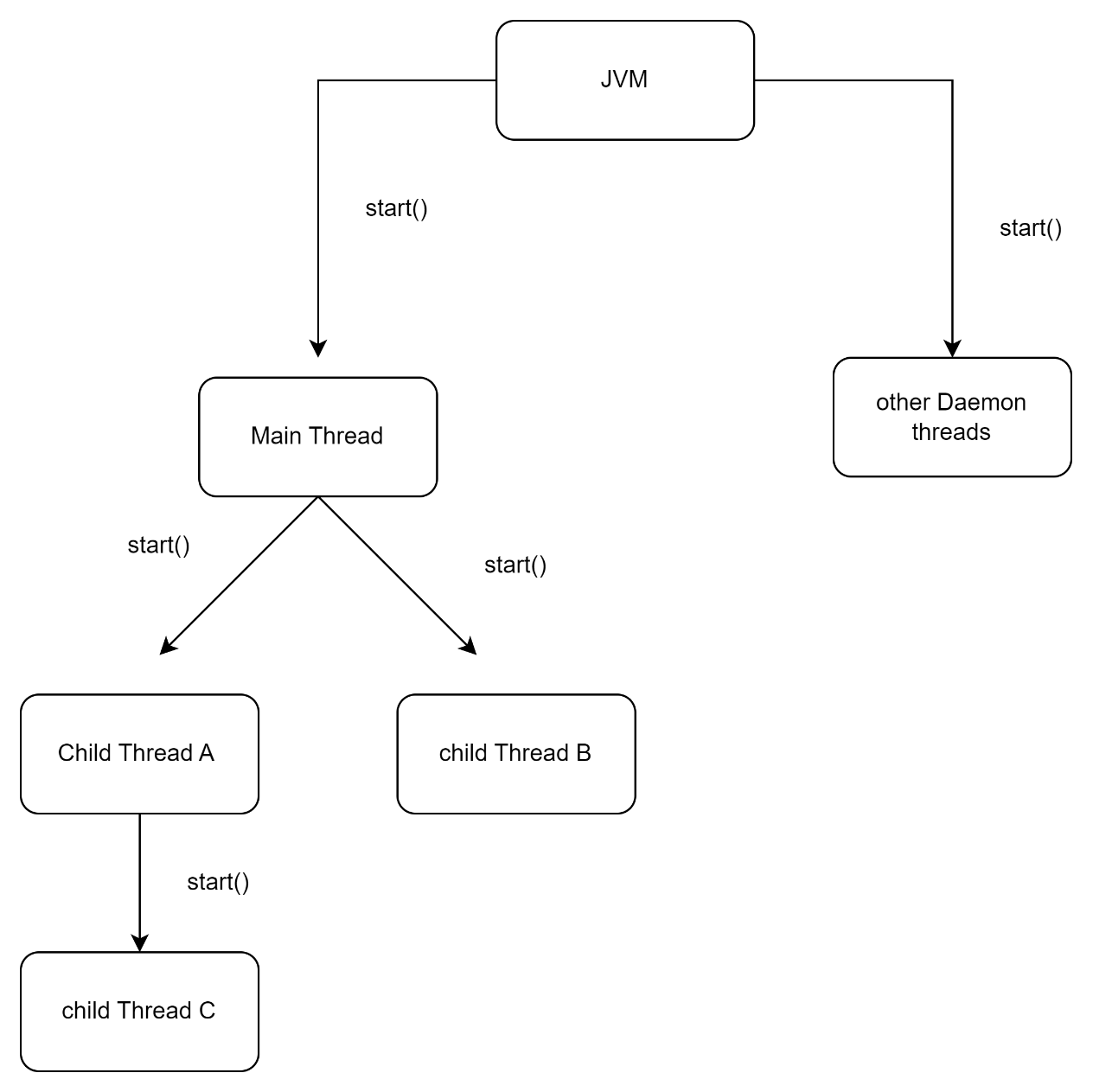
* public interface Runnable

Execution of threads in JVM's hand and it performs time slicing.

|  |  |
| --- | --- |
| **Thread Class** | **Runnable Interface** |
| Each thread create its unique object | Each thread create its unique object |
| More memory consumption | More memory consumption |
| A class extending Thread class cant extend any other class | A class implementing runnable can implement other interfaces . |
| Thread class is extended only if there is a need of overriding other methods to it | Runnable is implemented only if there is a need of special run methods |
| Enables tight coupling | Enable loose coupling |
|  |  |

**Java Main Thread -** from this other child threads are spawned . And it must be the last thread to finish execution i.e when the main thread stops program terminates.

Any thread can be marked as a daemon thread.- Daemon thread is executed by the JVM along with the main thread . We can set any thread as a Daemon thread.



**Multithreading :** is the ability of a program to run two or more threads concurrently where each thread can handle a different task at the same time making optimal use of the available resources .

**Join** : mechanism of inter thread synchronization ; what it means if when we are running multiple thread , output will be async and controlled by JVM . In order to have a synchronization between the output of multiple thread we use different mechanism and join is one of them.

* public final void join() throws InterruptedException - *Waits for this thread to die*
* public final void join(long millis) throws InterruptedException -

*Waits at most*millis*milliseconds for this thread to die. A timeout of 0 means to wait forever.*

* public final void join(long millis,intnanos) throws InterruptedException - *Waits at most*millis*milliseconds plus*nanos*nanoseconds for this thread to die*

**Timed*join()* is dependent on the OS for timing. So, we cannot assume that *join()* will wait exactly as long as specified.**

**\*\*\*\*\*\*\*Cannot put join on each and every thread . We use Synchronized keyword - It acquires the lock on the object , until its being released no other thread can access the method which have this lock**

**Synchronized block and keyword**

**Use case for synchronization - When multiple threads are working on the same object**