**How to create thread**

**There are two ways to create a thread:**

1. By extending Thread class
2. By implementing Runnable interface.

**Thread class:**

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

**Commonly used Constructors of Thread class:**

|  |
| --- |
| * Thread() * Thread(String name) * Thread(Runnable r) * Thread(Runnable r,String name) |

**Commonly used methods of Thread class:**

|  |
| --- |
| 1. **public void run():**is used to perform action for a thread. 2. **public void start():**starts the execution of the thread.JVM calls the run() method on the thread. 3. **public void sleep(long miliseconds):**Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds. 4. **public void join():**waits for a thread to die. 5. **public void join(long miliseconds):**waits for a thread to die for the specified miliseconds. 6. **public int getPriority():**returns the priority of the thread. 7. **public int setPriority(int priority):**changes the priority of the thread. 8. **public String getName():**returns the name of the thread. 9. **public void setName(String name):**changes the name of the thread. 10. **public Thread currentThread():**returns the reference of currently executing thread. 11. **public int getId():**returns the id of the thread. 12. **public Thread.State getState():**returns the state of the thread. 13. **public boolean isAlive():**tests if the thread is alive. 14. **public void yield():**causes the currently executing thread object to temporarily pause and allow other threads to execute. 15. **public void suspend():**is used to suspend the thread(depricated). 16. **public void resume():**is used to resume the suspended thread(depricated). 17. **public void stop():**is used to stop the thread(depricated). 18. **public boolean isDaemon():**tests if the thread is a daemon thread. 19. **public void setDaemon(boolean b):**marks the thread as daemon or user thread. 20. **public void interrupt():**interrupts the thread. 21. **public boolean isInterrupted():**tests if the thread has been interrupted. 22. **public static boolean interrupted():**tests if the current thread has been interrupted. |

**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(). |

|  |
| --- |
| 1. **public void run():**is used to perform action for a thread. |

**Starting a thread:**

|  |
| --- |
| **start() method** of Thread class is used to start a newly created thread. It performs following tasks:   * A new thread starts(with new callstack). * The thread moves from New state to the Runnable state. * When the thread gets a chance to execute, its target run() method will run |

### Java Thread Example by extending Thread class

|  |  |  |
| --- | --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi mm = multis.**new** Multi();  mm.start();  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread is running"**);  }  } } | thread is running |

### 2) Java Thread Example by implementing Runnable interface

|  |  |  |
| --- | --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi mm = multis.**new** Multi();   Thread t1 = **new** Thread(mm);  t1.start();  } } | **package** com.company;  **public class** Multis {   **public class** Multi **implements** Runnable{  **public void** run(){  System.***out***.println(**"thread is running"**);  }  } } | thread is running |

**public void sleep(long miliseconds)**

|  |  |  |
| --- | --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi mm = multis.**new** Multi();   System.***out***.println(**"Thread is starting"**);  mm.*sleep*(15000);  System.***out***.println(**"Thread is done"**);  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread is running"**);  }  } } | Thread is starting  Thread is done |

**public void join() & public void join(long miliseconds)**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi mm = multis.**new** Multi();   mm.start();  System.***out***.println(**"Thread is starting"**);  System.***out***.println(**"Alive @ first : "** + mm.isAlive());  mm.join(10000);  System.***out***.println(**"Thread is done"**);  System.***out***.println(**"name : "** + mm.getName());  System.***out***.println(**"Alive @ last : "** + mm.isAlive());  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread is running"**);  }  } } |
| **Thread is starting**  **thread is running**  **Alive @ first : true**  **Thread is done**  **name : Thread-0**  **Alive @ last : false** | |

**public int setPriority(int priority):**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi t1 = multis.**new** Multi();   Multies2 multies2 = **new** Multies2();  Multies2.Multi1 t2 = multies2.**new** Multi1();   t1.setPriority(1);  t2.setPriority(2);   t2.start();  System.***out***.println(**"isAlive 2 : "** + t2.isAlive());  t1.start();  System.***out***.println(**"isAlive 1 : "** + t1.isAlive());  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 1 is running"**);  }  } }  **package** com.company;  **public class** Multies2 {   **public class** Multi1 **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 2 is running"**);  }  } } |
| **isAlive 2 : true – t2 has started** (priority : 2)  ----(NOT EXECUTING THE t2.start() method)----  **isAlive 1 : true** (priority : 1)  **thread 1 is running** ----( NOW EXECUTING THE t1.start() method)  **thread 2 is running** ----( Finally EXECUTING THE t2.start() method) – bcz priority 2, doesn’t matter where it initialized. | |

**public int getPriority():**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi t1 = multis.**new** Multi();   Multies2 multies2 = **new** Multies2();  Multies2.Multi1 t2 = multies2.**new** Multi1();   t1.setPriority(1);  t2.setPriority(2);   t2.start();  System.***out***.println(**"isAlive 2 : "** + t2.isAlive());  System.***out***.println(**"getpriority t2 : "** + t2.getPriority());  t1.start();  System.***out***.println(**"isAlive 1 : "** + t1.isAlive());  System.***out***.println(**"getpriority t1 : "** + t1.getPriority());  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 1 is running"**);  }  } }  **package** com.company;  **public class** Multies2 {   **public class** Multi1 **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 2 is running"**);  }  } } |
| isAlive 2 : true  getpriority t2 : 2  isAlive 1 : true  getpriority t1 : 1  thread 2 is running  thread 1 is running | |

**public String getName()** & **public void setName(String name)**

|  |  |
| --- | --- |
| **package** com.company; **import** java.io.\*; **import** java.util.Scanner; **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi t1 = multis.**new** Multi();   Multies2 multies2 = **new** Multies2();  Multies2.Multi1 t2 = multies2.**new** Multi1();   System.***out***.println(**"t1's name : "** + t1.getName());  System.***out***.println(**"t2's name : "** + t2.getName());   t1.setName(**"soham"**);  t2.setName(**"Ivy"**);   System.***out***.println(**"t1's name after .getName : "** + t1.getName());  System.***out***.println(**"t2's name after .getName : "** + t2.getName());  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 1 is running"**);  }  } }  **package** com.company;  **public class** Multies2 {   **public class** Multi1 **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 2 is running"**);  }  } } |
| t1's name : Thread-0  t2's name : Thread-1  t1's name after .getName : soham  t2's name after .getName : Ivy | |

**public Thread currentThread()**

|  |  |
| --- | --- |
| **package** com.company;   **import** java.io.\*;  **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args) **throws** Exception{  Multis multis = **new** Multis();  Multis.Multi t1 = multis.**new** Multi();   Multies2 multies2 = **new** Multies2();  Multies2.Multi1 t2 = multies2.**new** Multi1();   t1.setPriority(1);  t2.setPriority(2);   t2.start();  System.***out***.println(**"isAlive 2 : "** + t2.isAlive());  System.***out***.println(**"Current thread : "** + Thread.*currentThread*().getName());  System.***out***.println(**"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"**);  t1.start();  System.***out***.println(**"isAlive 1 : "** + t1.isAlive());  System.***out***.println(**"Current thread : "** + Thread.*currentThread*().getName());  } } | **package** com.company;  **public class** Multis {   **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 1 is running"**);  }  } }  **package** com.company;  **public class** Multies2 {   **public class** Multi1 **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 2 is running"**);  }  } } |
| isAlive 2 : true  Current thread : main  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  isAlive 1 : true  Current thread : main  thread 2 is running  thread 1 is running | |

**public void yield():**

|  |  |
| --- | --- |
| **package** com.company;   **import** java.io.\*;  **import** java.util.Scanner;  **public class** Main **implements** Runnable {  Thread **t**;   Main(String str) {   **t** = **new** Thread(**this**, str);  *// this will call run() function* **t**.start();  }   **public void** run() {  **for** (**int** i = 0; i < 5; i++) {  *// yields control to another thread every 5 iterations* **if** ((i % 5) == 0) {  System.***out***.println(Thread.*currentThread*().getName() + **" yielding control..."**);   */\* causes the currently executing thread object to temporarily  pause and allow other threads to execute \*/* Thread.*yield*();  }  }  System.***out***.println(Thread.*currentThread*().getName() + **" has finished executing."**);  }   **public static void** main(String[] args) **throws** Exception{  **new** Main(**"Thread 1"**);  **new** Main(**"Thread 2"**);  **new** Main(**"Thread 3"**);  } } | Thread 1 yielding control...  Thread 2 yielding control...  Thread 1 has finished executing.  Thread 3 yielding control...  Thread 2 has finished executing.  Thread 3 has finished executing. |

**public void interrupt() & public boolean isInterrupted():**

|  |  |
| --- | --- |
| **package** com.company;  **import** java.io.\*; **import** java.util.Scanner;  **public class** Main {   **public static void** main(String[] args){  Multis multis = **new** Multis();  Multis.Multi t1 = multis.**new** Multi();   Multies2 multies2 = **new** Multies2();  Multies2.Multi1 t2 = multies2.**new** Multi1();   t1.start();  t1.interrupt();  System.***out***.println(**"if t1 is interrupted : "** + t1.isInterrupted());   t2.start();  System.***out***.println(**"if t2 is interrupted : "** + t2.isInterrupted());  } } | **package** com.company;  **import** java.util.Date;  **public class** Multis **extends** Thread{  **public class** Multi **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 1 is running"**);  }  }  }  **package** com.company;  **public class** Multies2 {   **public class** Multi1 **extends** Thread{  **public void** run(){  System.***out***.println(**"thread 2 is running"**);  }  } } |
| thread 1 is running  if t1 is interrupted : true  if t2 is interrupted : false  thread 2 is running | |