**Custom ThreadPool in Java – 2022**

**public class** CustomThreadPool {  
 **private** BlockingQueue<Runnable> **bq** = **new** LinkedBlockingQueue<>();  
 **private boolean isShutDown**;  
  
 **public** CustomThreadPool(**int** noOfThreads) {  
**for** (**int** i = 1; i <= noOfThreads; i++) { 🡸 create worker threads  
 **WorkerThread thread = new WorkerThread(bq, this);**  
 thread.setName(**"Worker Thread - "** + i);  
 thread.start();  
 }  
 }  
  
 **public void** execute(Runnable runnable) **throws** InterruptedException {  
 **if** (!**isShutDown**)  
 **bq**.put(runnable);  
 }  
  
 **public void** shutdown() {  
 **isShutDown** = **true**;  
 }  
  
 **private class** WorkerThread **extends** Thread {

**private** BlockingQueue<Runnable> **bq**;  
 **private** CustomThreadPool **threadPool**;  
  
 **public** WorkerThread(BlockingQueue<Runnable> taskQueue, CustomThreadPool threadPool) {  
 **this**.**bq** = taskQueue;  
 **this**.**threadPool** = threadPool;  
 }  
  
 @Override  
 **public void** run() {  
 **try** {  
 *// continue until all tasks finished processing* **while** (!**threadPool**.**isShutDown** || !**bq**.isEmpty()) {  
 Runnable runnable;  
 *// Poll a runnable from the queue and execute it  
// if ((runnable = runnableQueue.poll()) != null) { 🡸 also correct* **if** ((runnable = **bq**.take()) != **null**) {  
 runnable.run();  
 }  
 Thread.*sleep*(1);  
 } // 🡸 End of while loop  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

**To Test**

**public void** check() **throws** InterruptedException {  
 CustomThreadPool threadPool = **new** CustomThreadPool(2);  
 Runnable r1 = () -> m1();  
 Runnable r2 = () -> m1();  
 threadPool.execute(r1);  
 threadPool.execute(r2);  
 threadPool.shutdown();  
}

**Better Custom ThreadPool in Java 17-2024**

**import** java.util.concurrent.BlockingQueue;  
**import** java.util.concurrent.LinkedBlockingQueue;  
  
**public class** CustomThreadPool {  
 **private** BlockingQueue<Runnable> bq = **new** LinkedBlockingQueue<>();  
 **private boolean** isShutDown;  
  
 **public** CustomThreadPool(**int** noOfThreads) {  
 **for** (**int** i = 1; i <= noOfThreads; i++) { // 🡸 create worker threads  
 **Thread th = new Thread(() -> perform());**  
 **th.start();** }  
 }  
  
 **public void** shutdown() {  
 isShutDown = **true**;  
 }  
  
 **public void** perform() {  
 **while** (!isShutDown || !bq.isEmpty()) {  
 **try** {  
 Runnable runnable = bq.take();  
 **if**(runnable != **null**)  
 runnable.run();  
 } **catch** (InterruptedException e) {  
 **throw new** RuntimeException(e);  
 }  
 } // 🡸 End of while loop  
 }  
  
 **public void** execute(Runnable runnable) **throws** InterruptedException {  
 **if** (!isShutDown)  
 bq.put(runnable);  
 }  
}

To Test

**public void** check() **throws** InterruptedException {  
 CustomThreadPool threadPool = **new** CustomThreadPool(2);  
 Runnable r1 = () -> m1();  
 Runnable r2 = () -> m1();  
 threadPool.execute(r1);  
 threadPool.execute(r2);  
 threadPool.shutdown();  
}