在java的多线程中ThreadLocal（线程本地变量）是一个必须掌握的知识点，ThreadLocal在买个线程中都创建了变量的副本，实现了多线程中变量的隔离，在多线程环境下保证了成员变量的安全。

1、熟悉ThreadLocal的使用

2、尝试ThreadLocal源码解析

3、掌握ThreadLocalMap的存储过程

4、解决ThreadLocal内存泄漏

1、ThreadLocal的使用

先看一个简单的例子

@Slf4j  
public class MgDemo20200628 implements Runnable {  
  
 private static ThreadLocal<String> *local*=new ThreadLocal<>();  
 @Override  
 public void run() {  
 try {  
 *local*.set(Thread.*currentThread*().getName() +"---" + System.*currentTimeMillis*());  
 Thread.*sleep*(1000\*5);  
 *log*.info(*local*.get());  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 public static void main(String[] args) {  
 MgDemo20200628 demo =new MgDemo20200628();  
 for(int i=0;i<3;i++)  
 {  
 Thread task = new Thread(demo);  
 task.start();  
 }  
 }  
}

运行一下，控制台的输出结果

22:37:17.293 [Thread-1] INFO com.mg.empty.demo.thread.threadlocal.MgDemo20200628 - Thread-1---1593355032290

22:37:17.293 [Thread-2] INFO com.mg.empty.demo.thread.threadlocal.MgDemo20200628 - Thread-2---1593355032290

22:37:17.293 [Thread-0] INFO com.mg.empty.demo.thread.threadlocal.MgDemo20200628 - Thread-0---1593355032290