链滴

ThreadLocal

作者：614756773

原文链接：https://ld246.com/article/1595750141933

来源网站：链滴

功能

- 让每个线程使用一个独立的副本数据

源码

-

```
class ThreadLocalMap {
    class Entry extends WeakReference<ThreadLocal<?>> {
        Object value;

        Entry(ThreadLocal<?> k, Object v) {
            super(k);// 关键是这行代码，调用父类的构造方法，才将k这个对象设置为弱引用对象
            value = v;
        }

        private void set(ThreadLocal<?> key, Object value) {
            Entry[] tab = table;
            int len = tab.length;
            int i = key.threadLocalHashCode & (len-1);

            for (Entry e = tab[i];
                 e != null;
                 e = tab[i = nextIndex(i, len)]) {
                ThreadLocal<?> k = e.get();

                if (k == key) {
                    e.value = value;
                    return;
                }

                if (k == null) {
                    replaceStaleEntry(key, value, i);
                    return;
                }
            }
            ......
        }

        private static int nextIndex(int i, int len) {
            return ((i + 1 < len) ? i + 1 : 0);
        }
    }
}
```

-

```
private int expungeStaleEntry(int staleSlot) {
    Entry[] tab = table;
    int len = tab.length;

    tab[staleSlot].value = null;
```

```
    tab[staleSlot] = null;
    size--;
    ......
}
```

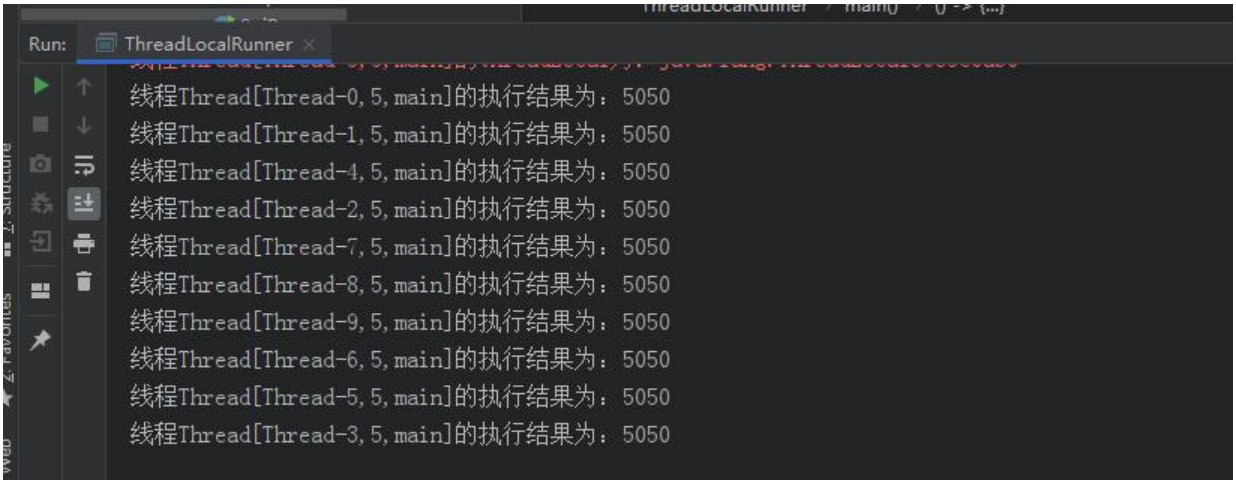
使用流程

先看下面这段代码

```
public class ThreadLocalRunner {
    private static ThreadLocal<Integer> threadLocal = new ThreadLocal<>();

    public static void main(String[] args) {
        for (int i = 0; i < 10; i++) {
            new Thread(() -> {
                threadLocal.set(0);
                for (int j = 1; j <= 100; j++) {
                    threadLocal.set(threadLocal.get() + j);
                }
                System.out.println("线程" + Thread.currentThread() + "的执行结果为：" + threadLocal
get());
            }).start();
        }
    }
}
```

这段代码的运行结果会是下面这样的



10个线程都是用的同一个 threadLocal对象，但是在使用 threadLocal.set(xx)时会创建各自的 ThreaLocalMap对象并且绑定在自己的线程对象里，代码如下

```
    // java.lang.ThreadLocal.java
    public void set(T value) {
        Thread t = Thread.currentThread();
        ThreadLocalMap map = getMap(t);
        if (map != null)
            map.set(this, value);
        else
            createMap(t, value);
    }
```

```
void createMap(Thread t, T firstValue) {
    t.threadLocals = new ThreadLocalMap(this, firstValue);
}
```

当调用 threadLocal.get()方法时也是使用自己线程的 ThreadLocalMap对象获取值，代码：

```
public T get() {
    Thread t = Thread.currentThread();
    ThreadLocalMap map = getMap(t);
    if (map != null) {
        ThreadLocalMap.Entry e = map.getEntry(this);
        if (e != null) {
            @SuppressWarnings("unchecked")
            T result = (T)e.value;
            return result;
        }
    }
    return setInitialValue();
}

ThreadLocalMap getMap(Thread t) {
    return t.threadLocals;
}
```