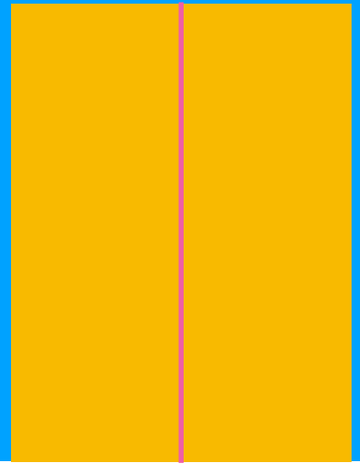
Thread

ThreadLocalMap threadLocals

ThreadLocalMap

Entry - WeakReference
remove(threadLocal)
Key: ThreadLocal
Value: Object



ThreadLocal

get()
set(..)
remove()

```
ThreadLocal threadLocal = new ThreadLocal();
threadLocal.set("A");
threadLocal.get(); // A
```

set 方法先得到 current thread, 然后调用 thread.threadLocals.set(this, value); // this —> ThreadLocal
get 方法先得到 current thread, 然后调用 thread.threadLocals.get(this); // this —> ThreadLocal

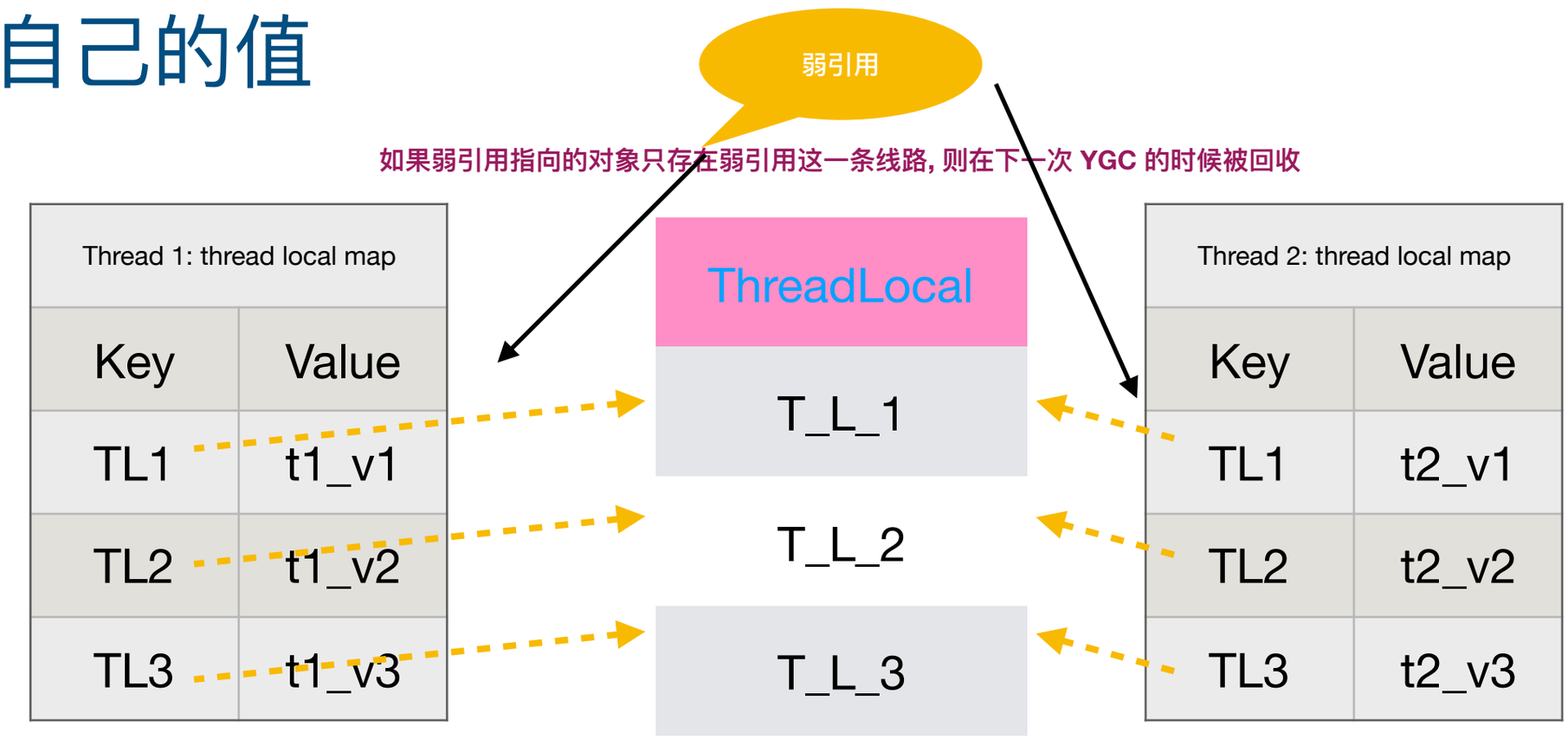
一个 thread local 对象可以被多个线程使用
每个线程内部都有一个 ThreadLocalMap
当调用 threadLocal.get / set 方法时都会先找到当前的线程对象的引用
然后分别在各自线程所对应的 thread local map 上进行操作

- **Thread 和 ThreadLocalMap 的关系**
- 每个 Thread 都有一个 ThreadLocalMap
- thread 对象不会直接使用这个 thread local map, 而是通过 thread local 对象

- **Thread 和 ThreadLocal 的关系**
- Thread 不会关心 thread local, 而 thread local 会关心 thread
- 当调用 threadLocal.set() / get() 方法的时候, threadLocal 会先得到当前线程对象的引用, 然后得到这个 thread 的 thread local map, 最终用 thread local map保存起来 thread local 和 value 的映射关系

- **ThreadLocalMap 保存了什么?**
- Key: ThreadLocal 对象
- Value: 任意对象

- 不同的线程都有不同的 thread local map
- 对于同一个 thread local 对象他们可以分别设定自己的值



- **ThreadLocal的弱引用**
- thread local map 中的 Entry extends WeakReference, 默认弱引用到 thread local 对象上
- 弱引用初衷是: 当只有 thread local map 上的 entry 弱引用到 threadLocal 对象时, 如果再次调用了 threadLocal 的 set 或 get 方法, thread local map 发现 threadLocal 对象为 null, 则会自动清除对这个 entry 的引用从而清除对 value 的引用.
- 但实际情况下, 一旦没有对 thread local 对象再次调用 set get 方法就无法清除对 value 的引用,从而导致内存泄露
- 实际情况下,需要显示调用 threadLocal 的 remove方法, 这个方法会移除 thread local map 中与该 thread local 对象相关的 entry, 从而解除对 value 的引用, 则不会因为持有对 value 对象的引用而造成内存泄露