weakCompareAndSet与compareAndSet的区别

# 以AtomicInteger为例

## compareAndSet

public final boolean **compareAndSet**(int expect,int update)

**Atomically sets the value to the given updated value if the current value == the expected value.**

Parameters:

expect - the expected value

update - the new value

Returns:

true if successful. False return indicates that the actual value was not equal to the expected value.

## weakCompareAndSet

public final boolean weakCompareAndSet(int expect,

int update)

Atomically sets the value to the given updated value if the current value == the expected value.

**May fail spuriously and does not provide ordering guarantees, so is only rarely an appropriate alternative to compareAndSet.**

Parameters:

expect - the expected value

update - the new value

Returns: true if successful

## 区别

从名字上可以看出：weak表示差的，显然**weakCompareAndSet有不足之处。**

**weakCompareAndSet可以会伪造地失败，并且不提供顺序的保证，因此一般不能替代compareAndSet方法。**

**May fail spuriously and does not provide ordering guarantees, so is only rarely an appropriate alternative to compareAndSet.**

**spuriously英['spjʊərɪəslɪ]美['spjʊərɪrslɪ]adv. 伪造地;**

# AtomicStampedReference<V>类似

**AtomicStampedReference<V>的提出是为了解决CAS的ABA问题。**

**除了对应的值之外，还有一个stamp戳，用来标识版本号。**

**compareAndSet与weakCompareAndSet的区别相同。**

public boolean **compareAndSet**(V expectedReference,V newReference,

**int expectedStamp,int newStamp**)

Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp.

Parameters:

expectedReference - the expected value of the reference

newReference - the new value for the reference

expectedStamp - the expected value of the stamp

newStamp - the new value for the stamp

Returns:true if successful

public boolean **weakCompareAndSet**(V expectedReference,V newReference,

int expectedStamp,int newStamp)

Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp.

May fail spuriously and does not provide ordering guarantees, so is only rarely an appropriate alternative to **compareAndSet**.

Parameters:

expectedReference - the expected value of the reference

newReference - the new value for the reference

expectedStamp - the expected value of the stamp

newStamp - the new value for the stamp

Returns:true if successful