edu.mit.csail.aeolus.api
Class AeolusLock

```
java.lang.Object
  └ edu.mit.csail.aeolus.api.AeolusShared
      └ edu.mit.csail.aeolus.api.AeolusLock
```

All Implemented Interfaces:
AeolusSafe

```
public final class AeolusLockextends AeolusShared
```

Implementation of a shared lock.

Field Summary

Fields inherited from class edu.mit.csail.aeolus.api.AeolusShared
sharedLastEid

Constructor Summary

AeolusLock() Construct an AeolusLock with the current thread's labels.	
AeolusLock(AeolusLabel sLabel, AeolusLabel iLabel) Construct an AeolusLock with the specified labels.	

Method Summary

void	**lock**() Blocks until the lock is acquired.
boolean	**tryLock**() Acquires the lock if it is not currently held and returns true, otherwise returns false.
void	**unlock**() Releases the lock

Methods inherited from class edu.mit.csail.aeolus.api.AeolusShared

`getIntegrityLabel, getSecrecyLabel`

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

AeolusLock

```
public AeolusLock()
```

Construct an AeolusLock with the current thread's labels.

AeolusLock

```
public AeolusLock(AeolusLabel sLabel,
                  AeolusLabel iLabel)
           throws InfoFlowControlException
```

Construct an AeolusLock with the specified labels.

Throws:

`InfoFlowControlException` - if thread's labels are more constrained than given labels

Method Detail

lock

```
public void lock()
          throws InfoFlowControlException
```

Blocks until the lock is acquired.

Throws:

`InfoFlowControlException` - if the thread can't write the lock

---

tryLock

```
public boolean tryLock()
                throws InfoFlowControlException
```

Acquires the lock if it is not currently held and returns true, otherwise returns false.

Returns:

true if lock acquired, else false

Throws:

`InfoFlowControlException` - if the thread can't read and write the lock

---

unlock

```
public void unlock()
            throws InfoFlowControlException
```

Releases the lock

Throws:

`InfoFlowControlException` - if the thread can't write the lock