

DrlvmLocking

DRLVM has a complex system of locks used to synchronize multiple threads operation.

Thread primitives in DRLVM

Low-level primitives for use in C++ code

- Mutex: `hymutex_create`, `hymutex_lock`, `hymutex_trylock`, `hymutex_unlock`, `hymutex_destroy`
Most of the internal DRLVM locks are mutexes
- Conditional variable: `hycond_create`, `hycond_wait`, `hycond_wait_timed`, `hycond_wait_interruptable`, `hycond_notify`, `hycond_notify_all`, `hycond_destroy`
Conditional variables are used in pair with mutex. Most of the internal DRLVM events are implemented using conditional variables.

Higher-level synchronization objects

- Java monitors are accessible from C++ code using functions

```
jthread_monitor_init  thread\src\thread_java_monitors.c
jthread_monitor_enter thread\src\thread_java_monitors.c
jthread_monitor_try_enter thread\src\thread_java_monitors.c
jthread_monitor_exit  thread\src\thread_java_monitors.c
jthread_monitor_notify thread\src\thread_java_monitors.c
jthread_monitor_notify_all thread\src\thread_java_monitors.c
jthread_monitor_wait  thread\src\thread_java_monitors.c
jthread_monitor_timed_wait thread\src\thread_java_monitors.c
jthread_monitor_init  thread/src/thread_java_monitors.c
jthread_monitor_enter thread/src/thread_java_monitors.c
jthread_monitor_try_enter thread/src/thread_java_monitors.c
jthread_monitor_exit  thread/src/thread_java_monitors.c
jthread_monitor_notify thread/src/thread_java_monitors.c
jthread_monitor_notify_all thread/src/thread_java_monitors.c
jthread_monitor_wait  thread/src/thread_java_monitors.c
jthread_monitor_timed_wait thread/src/thread_java_monitors.c
```

Topics to be covered by this page

- What synchronization objects should be used and when during DRLVM development
- The general locking rules ("locking protocol") in DRLVM
 - the list of global locks
 - the order of obtaining locks
 - interaction with suspension

Some of the topics are already covered by <http://harmony.apache.org/subcomponents/drlvm/TM.html>