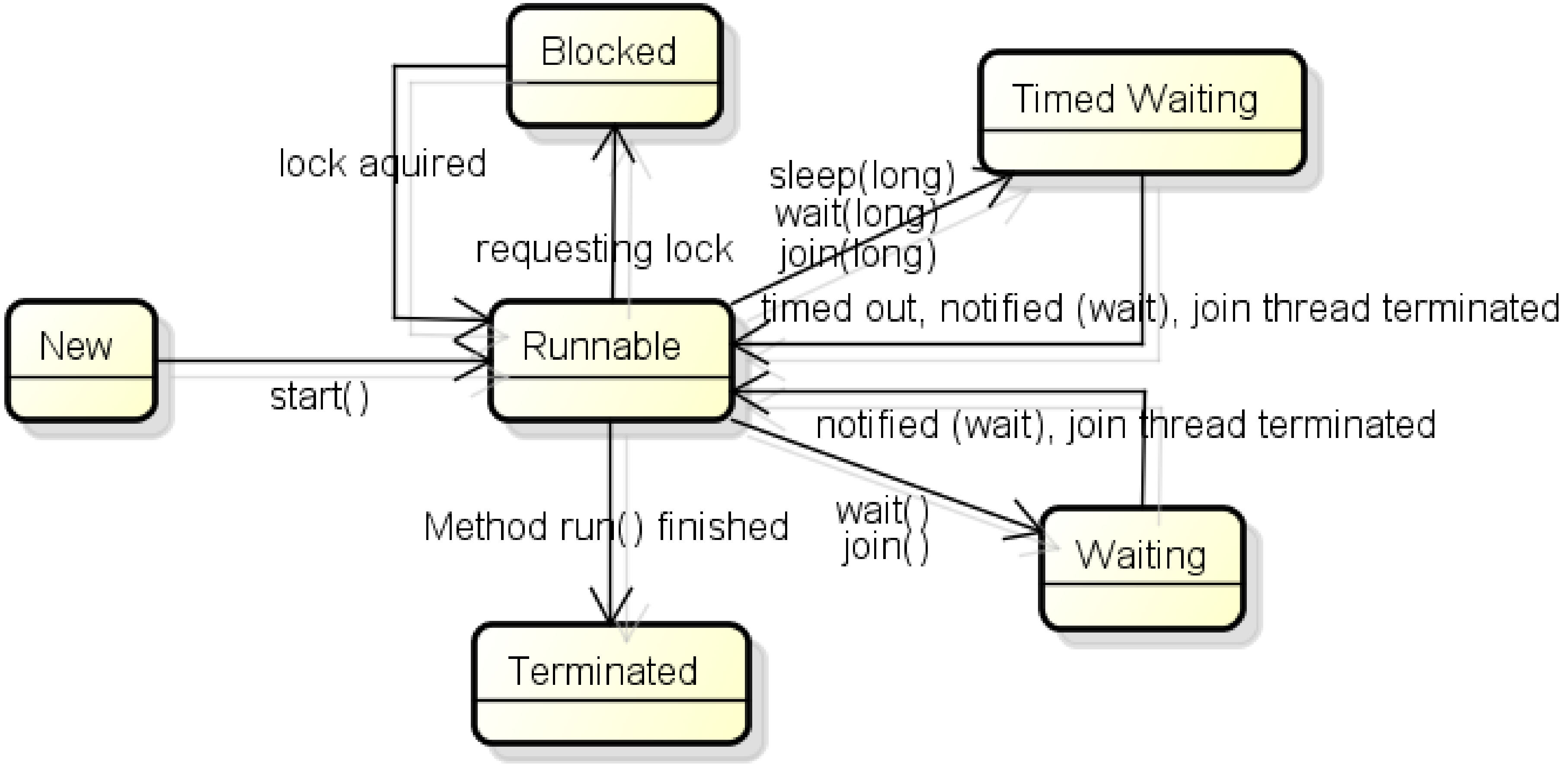
VIA University College

Software Development with UML and Java 2

Learning Objectives

- ❖Understand Thread synchronization
- ❖Write small programs using thread synchronization

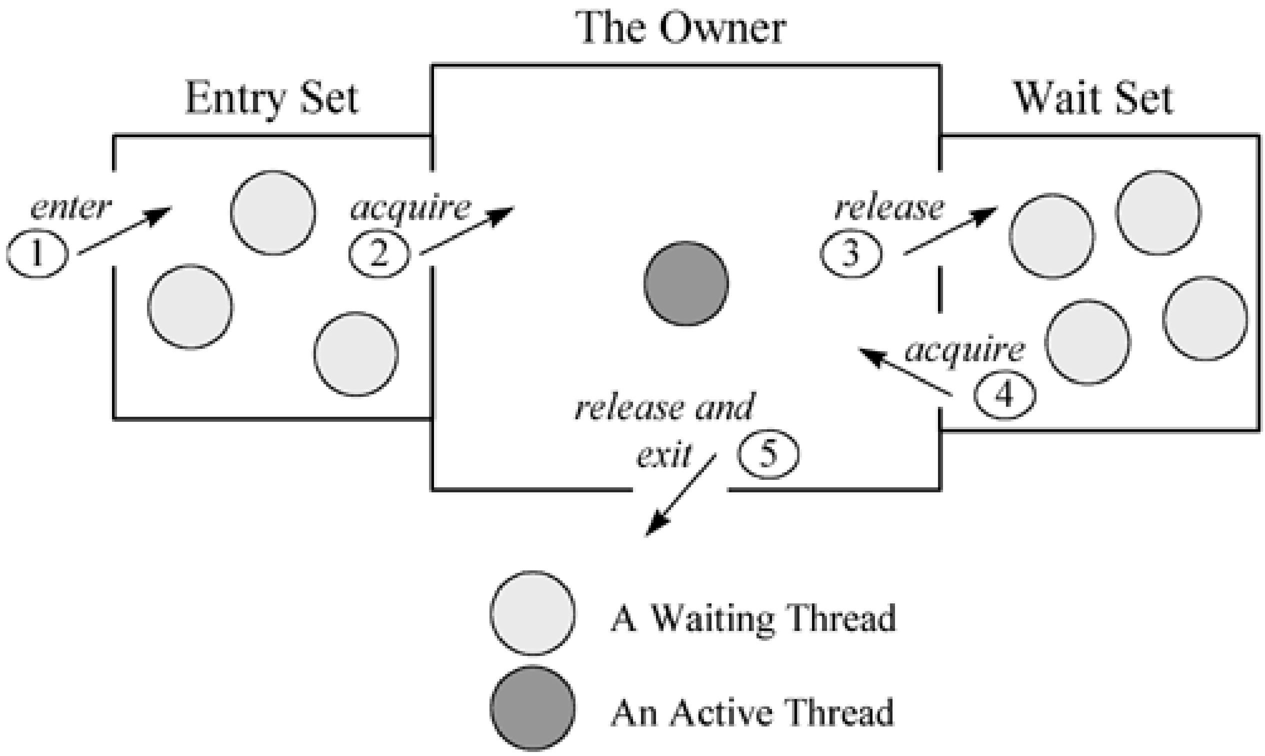
Thread states



Java Monitor

- A Monitor is a mechanism that ensures that at most one thread at a time can execute a given critical section or method.
- Every object in Java is potentially a Monitor
  - Keyword `synchronized` is used to define a critical section
  - Methods `wait()` and `wait(long)` are used to temporarily leave the Monitor and go to Wait State
  - Methods `notify()` and `notifyAll(long)` are used to wake up one or more threads from Wait State (making the waked-up thread go to Runnable and then directly to Blocked State until the Monitor is available)

Java Monitor (“The Owner”)

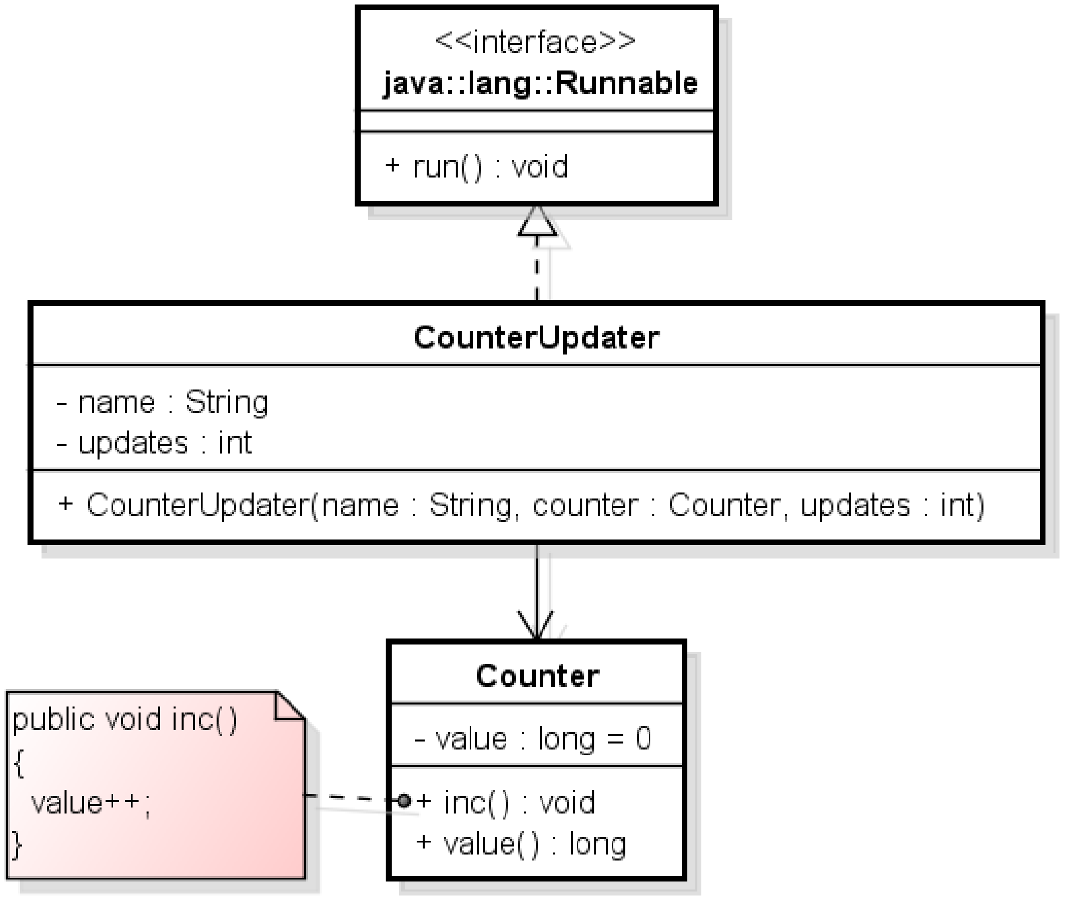


Bill Venners, “Thread Synchronization, Chapter 20 of Inside the Java Virtual Machine”, http://www.artima.com/insidejvm/ed2/threadsynch.html

Synchronized methods

- A thread can call the methods (if it is in the monitor of that object – in a block or method synchronized on this object)
  - wait(), wait(long) // Going from Runnable to Wait state
  - // releasing the monitor's lock
  - notify(), notifyAll() // Wake up one or all threads waiting to // acquire the monitor's lock

Updating shared variables



```
public void inc()
{
  value++;
}
```

Thread safe Counter (Monitor)

```
class Counter
{
  private long value;

  public void inc()
  {
    synchronized(this)  // syncronized on the Counter object
    {
      value++;
    }
  }

  public long value()
  {
    synchronized(this)
    {
      return value;
    }
  }
}
```

Thread safe Counter (Monitor)

```
class Counter
{
  private long value;

  public synchronized void inc()
  {
    value++;
  }

  public synchronized long value()
  {
    return value;
  }
}
```

Monitor:

1) All instance variables are private
2) All methods are synchronized

Updating shared variables

```
Main Thread Ended
Updater1 finished: Counter.value = 40000
Updater2 finished: Counter.value = 40000

Main Thread Ended
Updater2 finished: Counter.value = 40000
Updater1 finished: Counter.value = 40000

Main Thread Ended
Updater1 finished: Counter.value = 39842
Updater2 finished: Counter.value = 40000

Main Thread Ended
Updater2 finished: Counter.value = 38880
Updater1 finished: Counter.value = 40000
```

Waiting for a shared object

```
~~public synchronized void method() throws InterrupterException~~
~~{~~
~~  if (! conditionToEnterIsOK)~~
~~    wait();~~

~~  // modify monitor data attributes~~
~~  notifyAll();~~
~~}~~
```

```
public synchronized void method() throws InterrupterException
{
  while (! conditionToEnterIsOK)
    wait();

  // modify monitor data attributes
  notifyAll();
}
```

Thread safe Counter (waiting)

```
class Counter
{
  private long value;

  public synchronized void inc()
  {
     while (value > 10)
     {
       try
       {
          wait();
       }
       catch (InterruptedException e)
       {
          //...
       }
     }
     value++;

     notifyAll();
   }

  //...
}
```