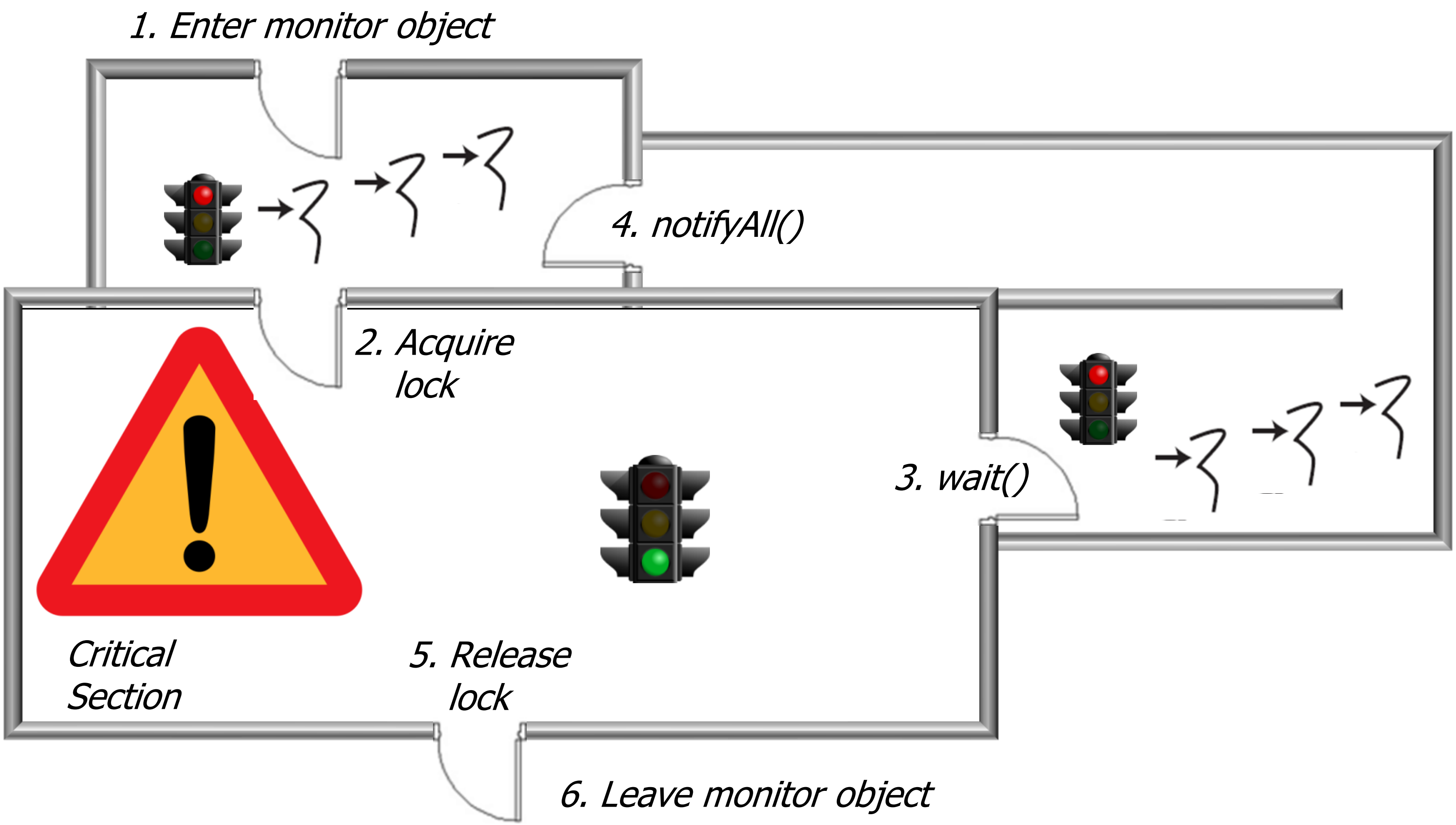
Java Monitor Objects: Coordination Methods

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand how Java built-in monitor objects provide waiting & notification mechanisms that coordinate threads running in a concurrent program



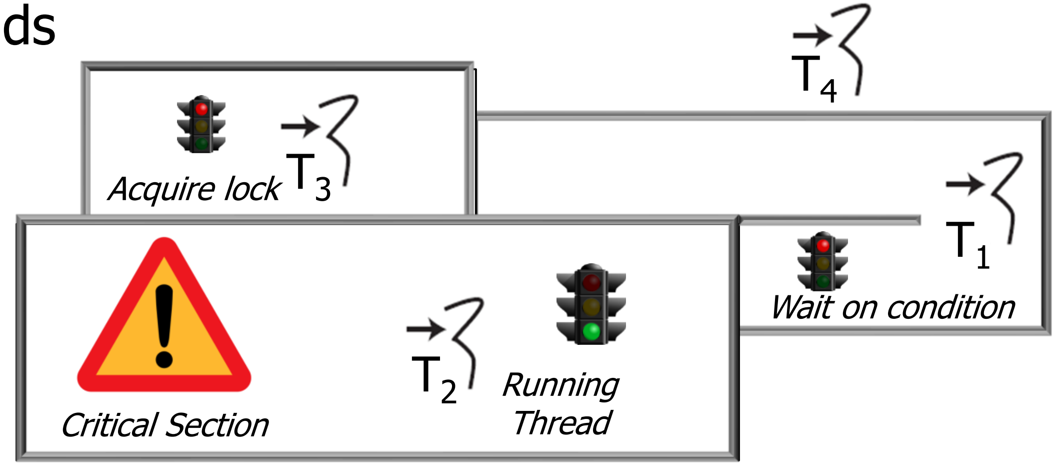
Java Built-in Waiting & Notification Mechanisms

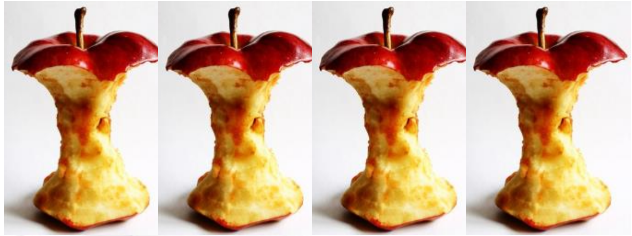
Java Built-in Waiting & Notification Mechanisms

Java synchronized methods & statements only provide a partial solution to concurrent programs

Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions





Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions
  - via the wait(), notify(), & notifyAll() methods

void	wait() – Causes the current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object
void	notify() – Wakes up a single thread that is waiting on this object's monitor
void	notifyAll() – Wakes up all threads that are waiting on this object's monitor

See docs.oracle.com/javase/8/docs/api/java/lang/Object.html

Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions
  - via the wait(), notify(), & notifyAll() methods

void	wait() – Causes the current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object
void	notify() – Wakes up a single thread that is waiting on this object's monitor
void	notifyAll() – Wakes up all threads that are waiting on this object's monitor

Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions
  - via the wait(), notify(), & notifyAll() methods

void	wait() – Causes the current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object
void	notify() – Wakes up a single thread that is waiting on this object's monitor
void	notifyAll() – Wakes up all threads that are waiting on this object's monitor

Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions
  - via the wait(), notify(), & notifyAll() methods

void	wait() – Causes the current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object
void	notify() – Wakes up a single thread that is waiting on this object's monitor
void	notifyAll() – Wakes up all threads that are waiting on this object's monitor

Java Built-in Waiting & Notification Mechanisms

- Java monitor objects allow threads to coordinate their interactions
  - via the wait(), notify(), & notifyAll() methods

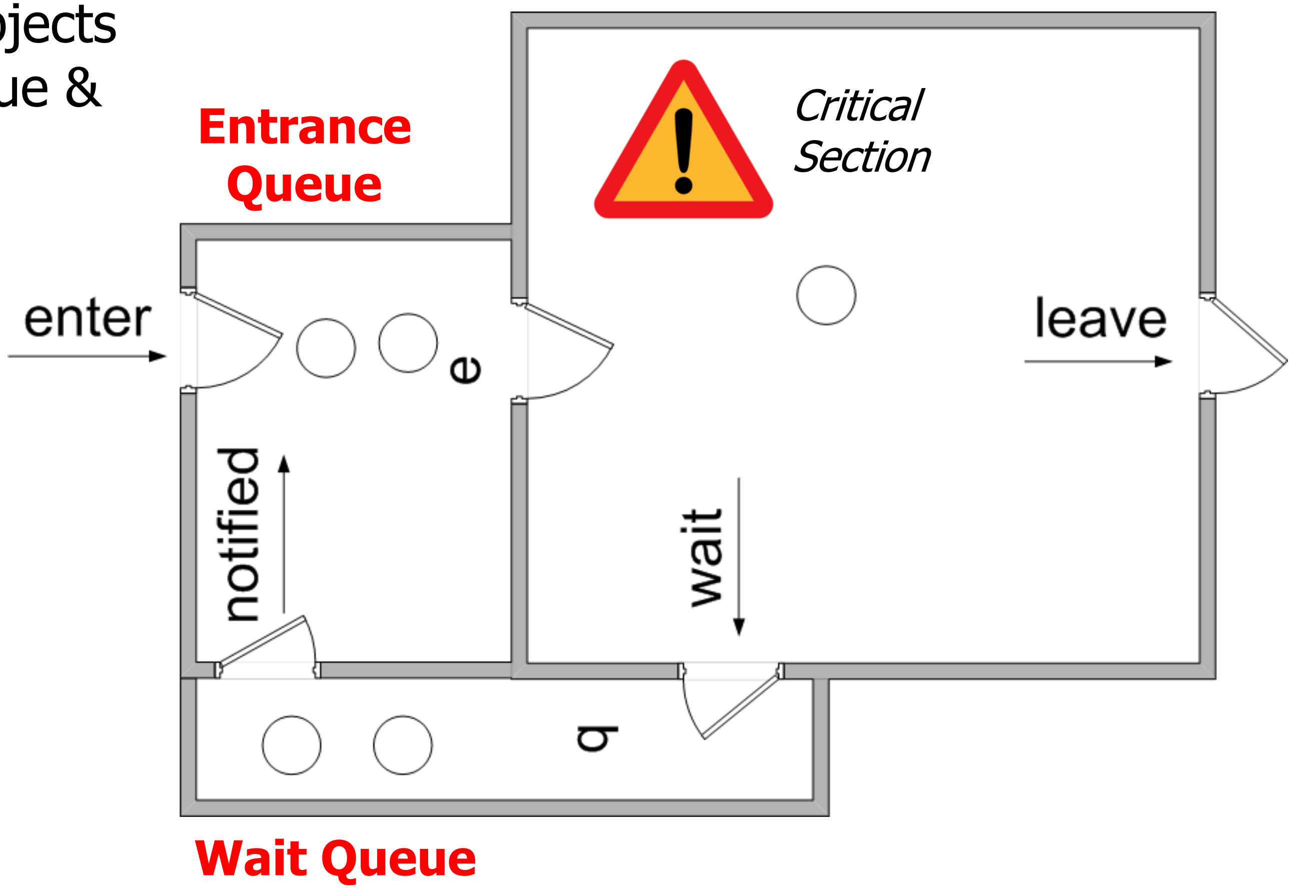
void	wait() – Causes the current thread to wait until another thread invokes the notify() method or the notifyAll() method for this object
void	notify() – Wakes up a single thread that is waiting on this object's monitor
void	notifyAll() – Wakes up all threads that are waiting on this object's monitor

ADOLPH ZUKOR and JESSE L. LASKY PRESENT
ZANE GREY'S
THE THUNDERING HERD
with JACK HOLT, LOIS WILSON
NOAH BEERY, RAYMOND HATTON
SCREEN PLAY BY LUCIEN HUBBARD • • DIRECTED BY WILLIAM HOWARD

See en.wikipedia.org/wiki/Thundering_herd_problem

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue



Entrance Queue

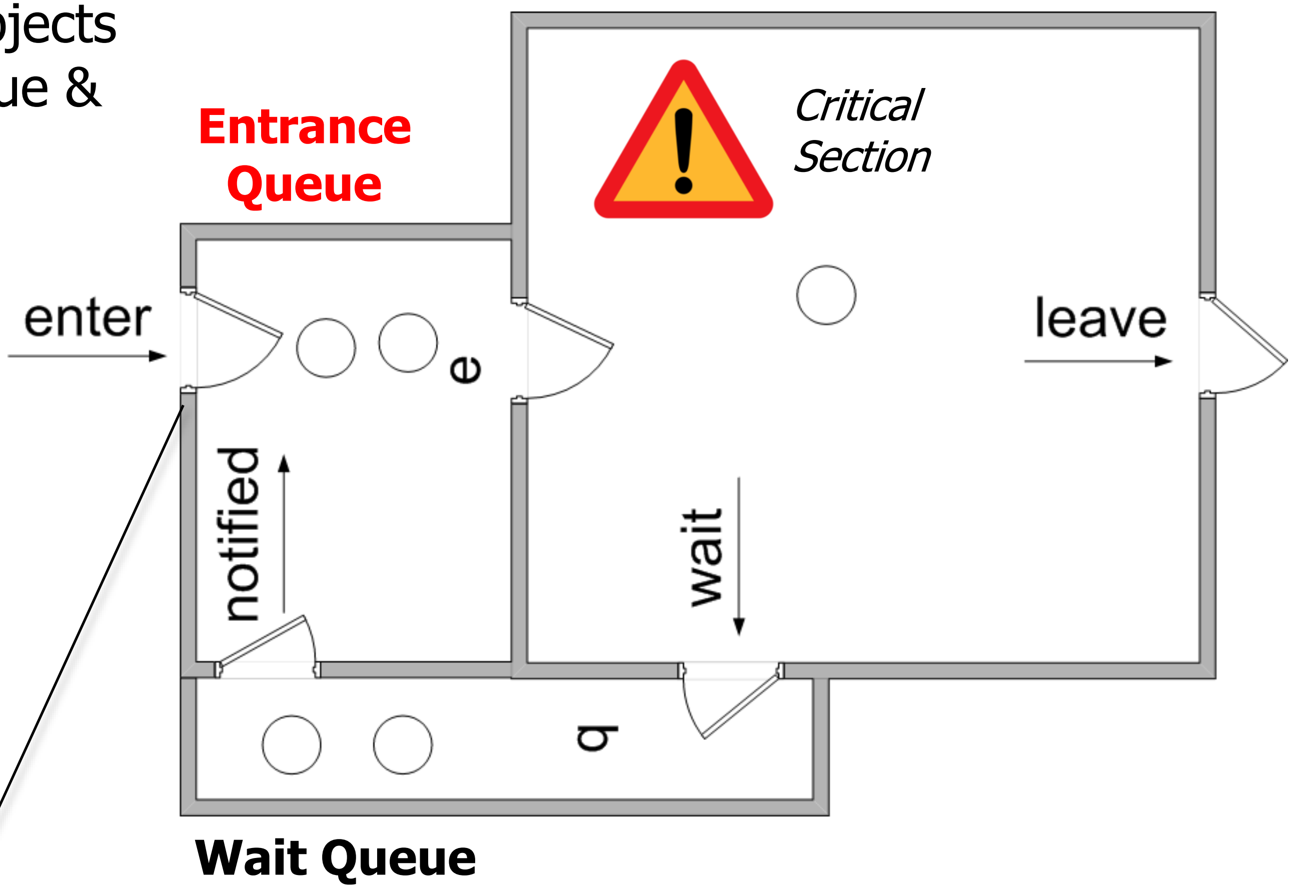
Critical Section

Wait Queue

See en.wikipedia.org/wiki/Monitor_(synchronization)#Implicit_condition_variable_monitors

Java Built-in Waiting & Notification Mechanisms

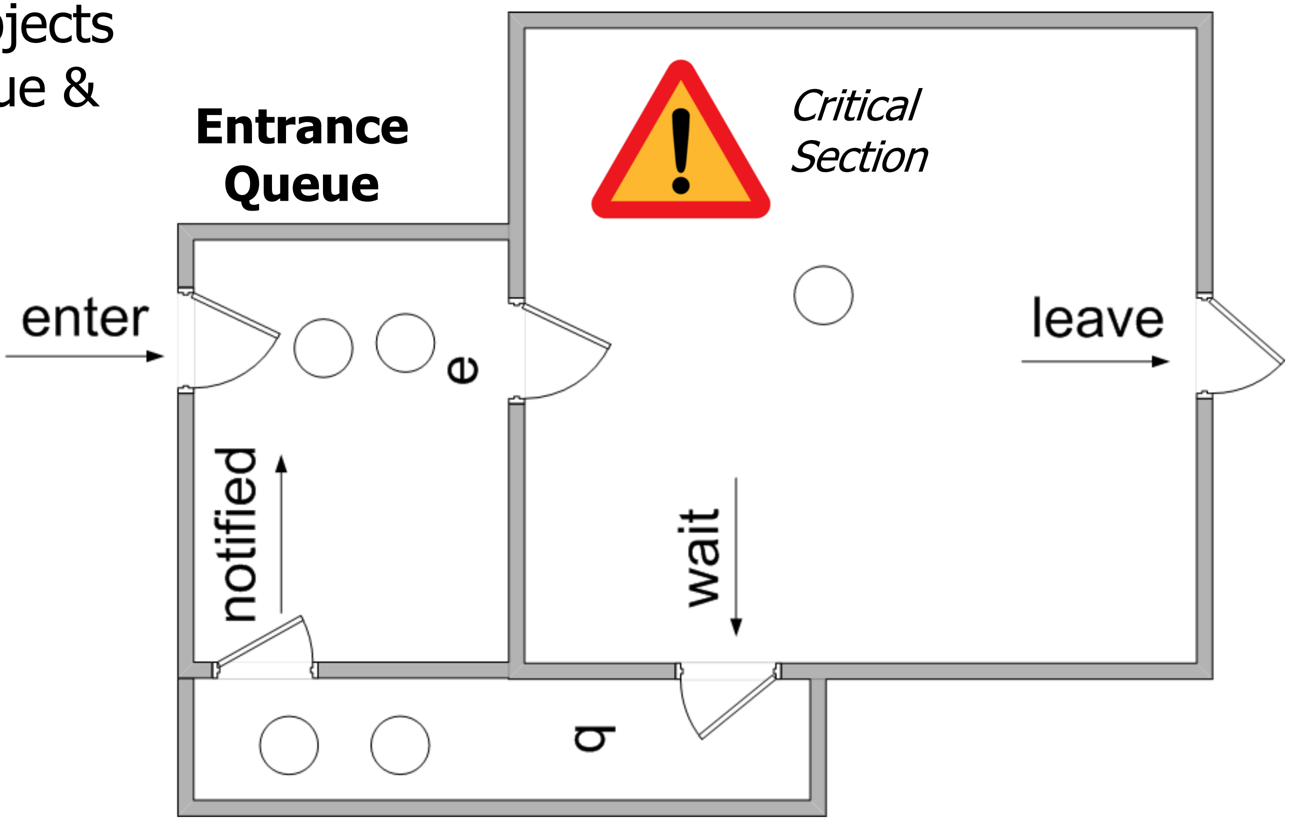
- Java built-in monitor objects have one entrance queue & one wait queue



Serializes thread access to monitor object's critical section

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue

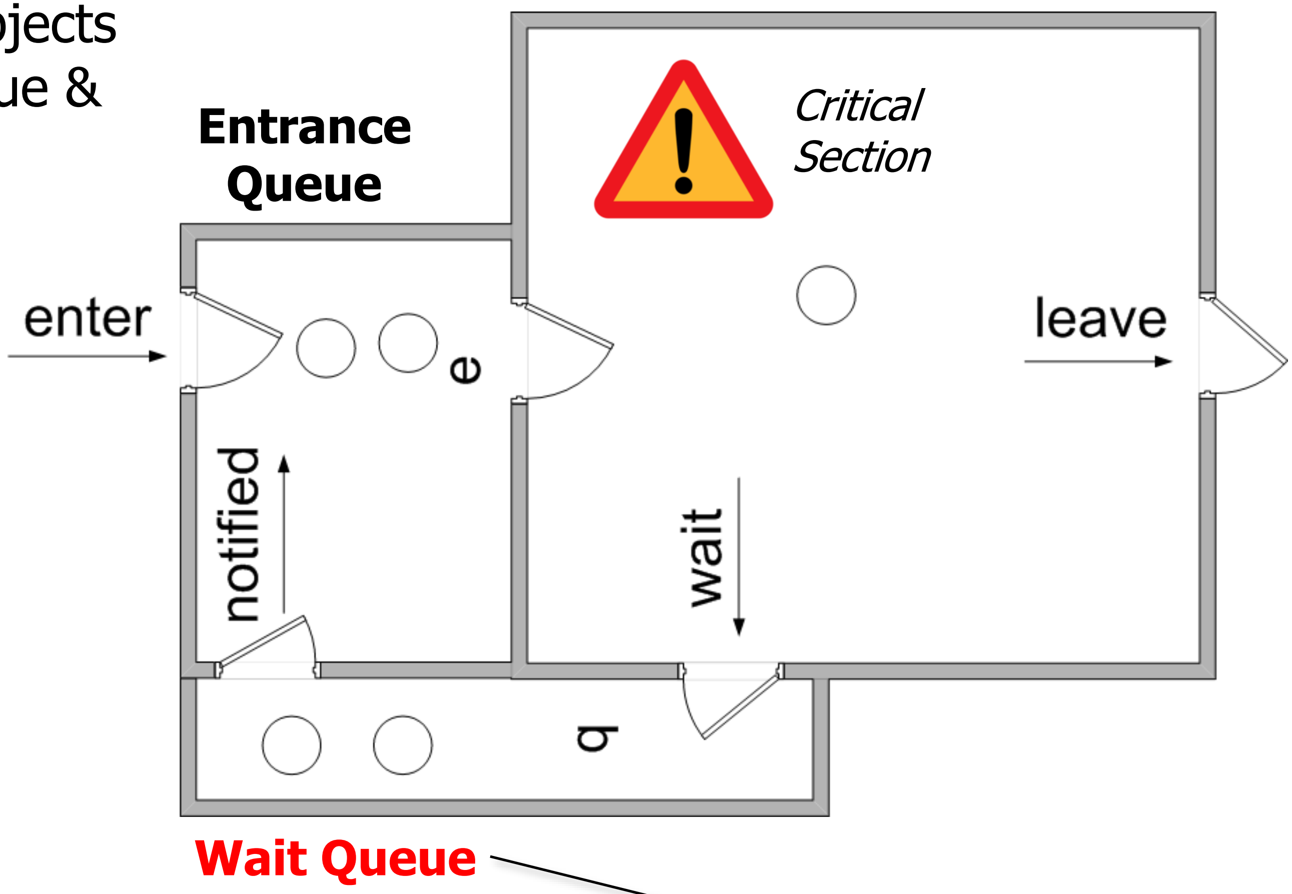


Wait Queue

All threads that call wait() are parked on the wait queue

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue



All notify() & notifyAll() calls also apply to the wait queue

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

This class fixes the "busy waiting" problem with BusySynchronizedQueue

See github.com/douglascraigschmidt/POSA/tree/master/ex/M3/BoundedBuffers/SimpleBoundedBlockingQueue

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue, e.g.
  - put() calls wait() when the queue is full

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

Atomically releases the intrinsic lock & sleeps on the wait queue

See en.wikipedia.org/wiki/Guarded_suspension

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue, e.g.
  - put() calls wait() when the queue is full
    - It also calls notifyAll() after adding an item

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

Must wake up all the threads blocked on the wait queue since waiters are non-uniform

See upcoming lesson on "*Java Monitor Objects: Usage Considerations*"

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue, e.g.
  - put() calls wait() when the queue is full
    - It also calls notifyAll() after adding an item

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

notifyAll() is required due to a Java monitor object only having one wait queue

LIMITED

See stackoverflow.com/questions/37026/java-notify-vs-notifyall-all-over-again/3186336#3186336

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue, e.g.
  - put() calls wait() when the queue is full
  - take() calls wait() when the queue is empty

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

Atomically releases the intrinsic lock & sleeps on the wait queue

See en.wikipedia.org/wiki/Guarded_suspension

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue, e.g.
  - put() calls wait() when the queue is full
  - take() calls wait() when the queue is empty
    - It also calls notifyAll() after removing an item

```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

Must wake up all the threads blocked on the wait queue since waiters are non-uniform

Again, notifyAll() is required here due to the limitations of Java monitor objects, which only have one wait queue

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor objects have one entrance queue & one wait queue

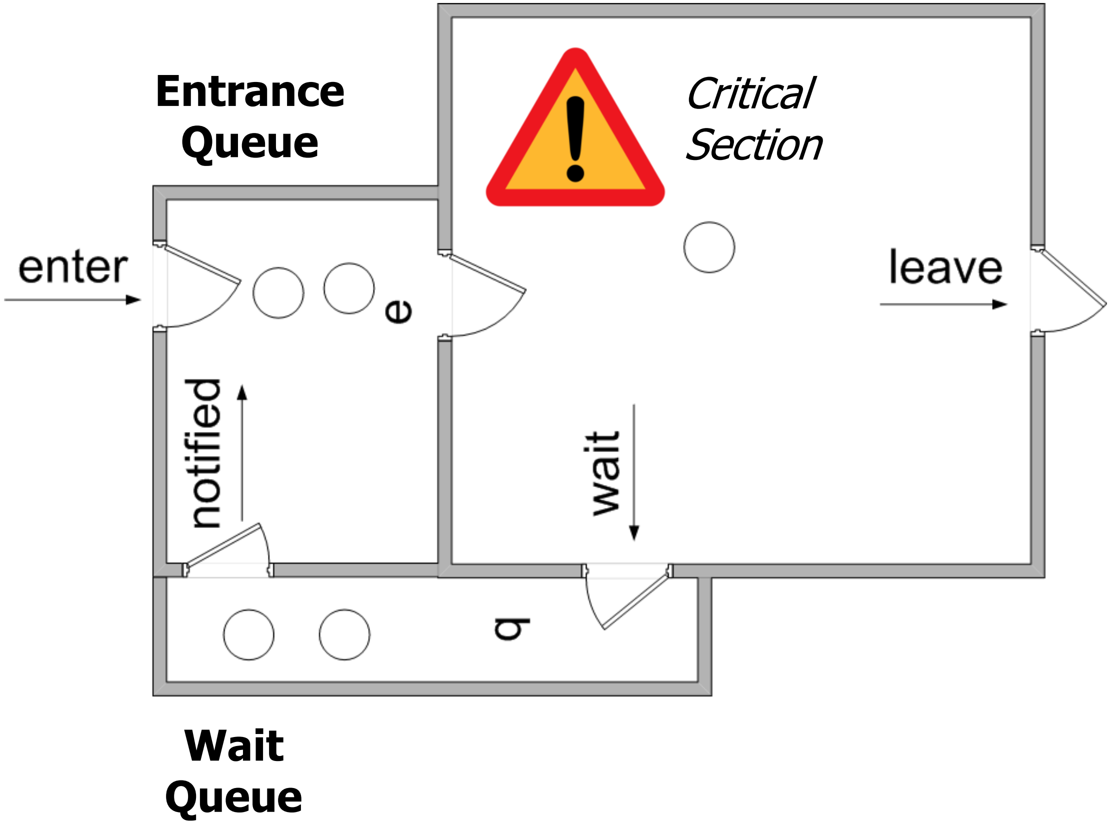
```
class SimpleBoundedBlockingQueue<E>
      implements BlockingQueue<E> {
  ...
  public void put(E msg){
    synchronized(this) {
      while (isFull()) wait();
      mList.add(msg);
      notifyAll();
    }
  }

  public E take() ... {
    synchronized(this) {
      while (isEmpty()) wait();
      notifyAll();
      return mList.poll();
    }
  }
  ...
```

The put() & take() methods are examined later in this lesson

See upcoming lesson on "*Java Monitor Objects: Coordination Example Implementation*"

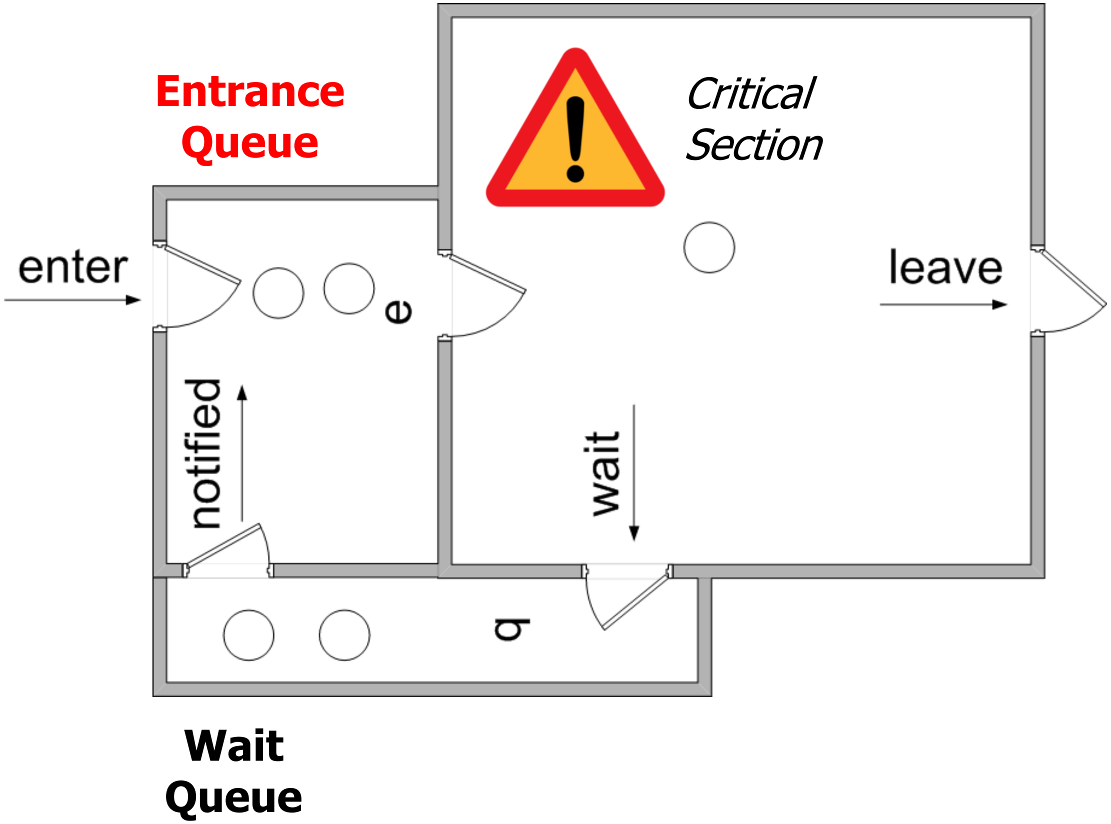
Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor object synchronizers can be implemented w/POSIX-like synchronizers



Java Built-in Waiting & Notification Mechanisms

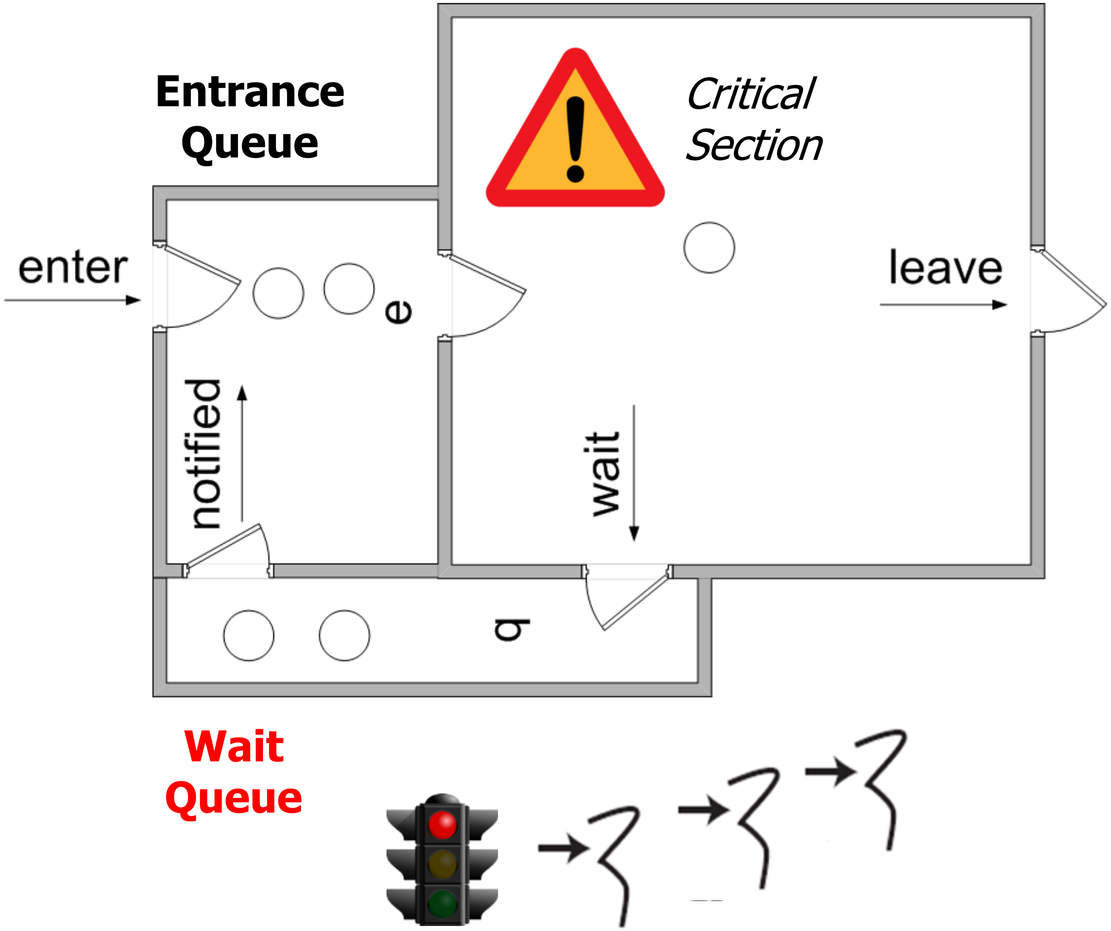
- Java built-in monitor object synchronizers can be implemented w/POSIX-like synchronizers, e.g.
  - Entrance queue is akin to a POSIX recursive mutex



See computing.llnl.gov/tutorials/pthreads/#Mutexes

Java Built-in Waiting & Notification Mechanisms

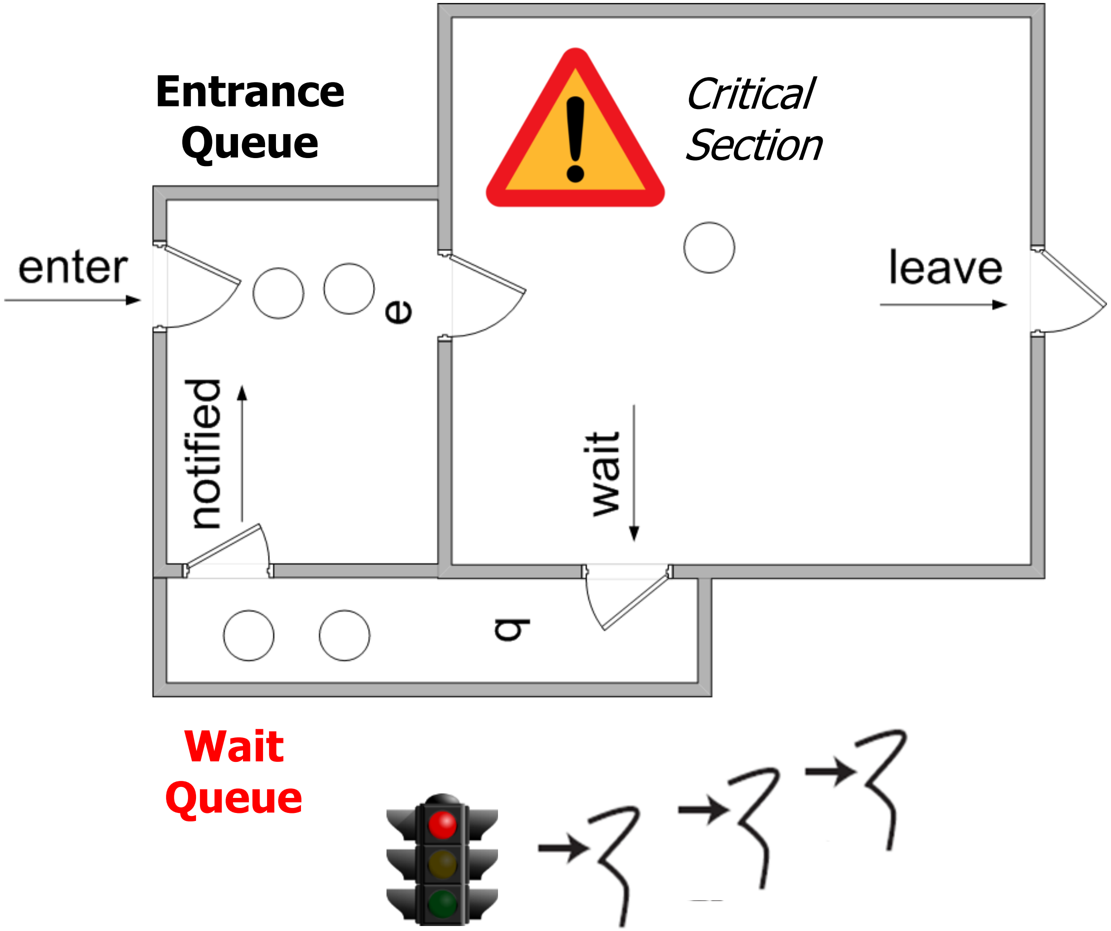
- Java built-in monitor object synchronizers can be implemented w/POSIX-like synchronizers, e.g.
  - Entrance queue is akin to a POSIX recursive mutex
  - Wait queue is akin to a POSIX condition variable



See computing.llnl.gov/tutorials/pthreads/#ConditionVariables

Java Built-in Waiting & Notification Mechanisms

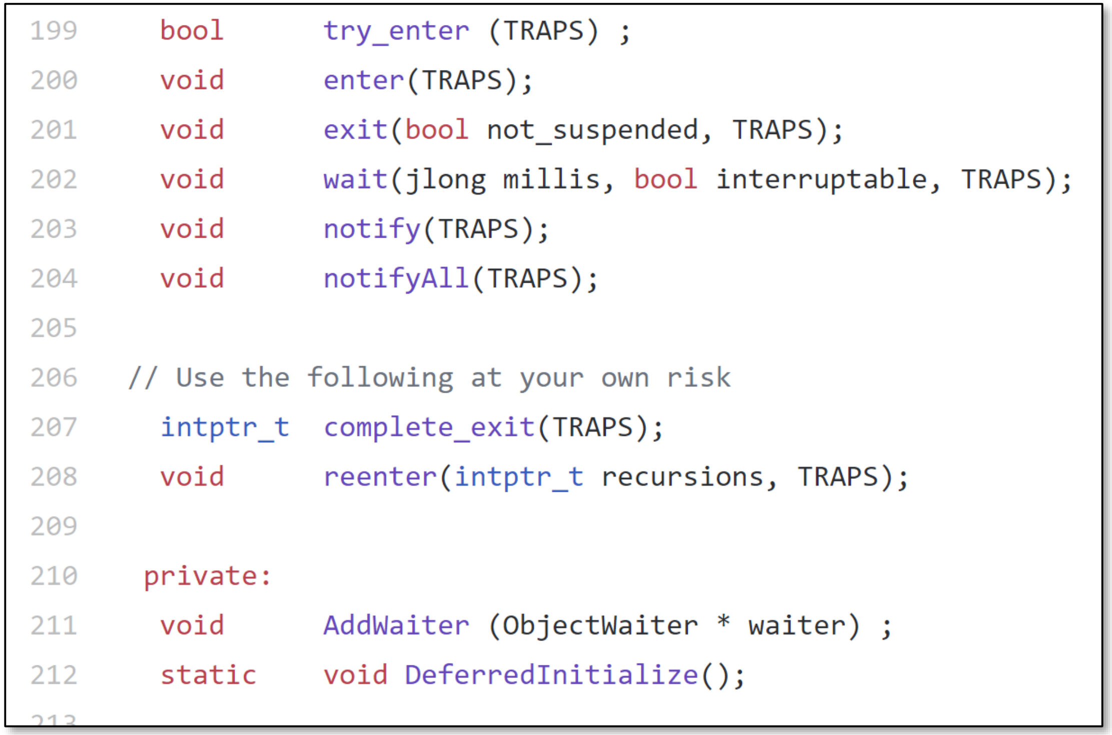
- Java built-in monitor object synchronizers can be implemented w/POSIX-like synchronizers, e.g.
  - Entrance queue is akin to a POSIX recursive mutex
  - Wait queue is akin to a POSIX condition variable
    - Similar to Java ConditionObjects



See earlier lessons on "*Java ConditionObjects*"

Java Built-in Waiting & Notification Mechanisms

- Java built-in monitor object synchronizers can be implemented w/POSIX-like synchronizers, e.g.
  - Entrance queue is akin to a POSIX recursive mutex
  - Wait queue is akin to a POSIX condition variable
  - The implementation in the Oracle JDK uses lower-level locking primitives



```
  bool        try_enter (TRAPS) ;
  void        enter(TRAPS);
  void        exit(bool not_suspended, TRAPS);
  void        wait(jlong millis, bool interruptable, TRAPS);
  void        notify(TRAPS);
  void        notifyAll(TRAPS);

// Use the following at your own risk
  intptr_t    complete_exit(TRAPS);
  void        reenter(intptr_t recursions, TRAPS);

 private:
  void        AddWaiter (ObjectWaiter * waiter) ;
  static      void DeferredInitialize();
```

See github.com/JetBrains/jdk8u_hotspot/blob/master/src/share/vm/runtime/objectMonitor.cpp

End of Java Monitor Object: Coordination Methods