Programming with Concurrent Threads (Part 3)

Robert M. Dondero, Ph.D.
Princeton University

Objectives

- We will cover:
  - Thread conditions

Agenda

- **Thread conditions**

Thread Conditions

- Problem in bank example:

```
$ python lockinresource.py
1
2
3
4
5
3
1
-1
-3
-5
-4
-3
-2
-1
0
Final balance: 0
$
```

Unrealistic for bank accounts to have negative balances

Thread Conditions

- Solution:
  - Before withdrawing, withdraw thread should **wait** for the bank account balance to be sufficiently large
  - After depositing, deposit thread should **notify** waiting threads that they can try again

Thread Conditions

- Solution in general:
  - **Consumer** thread **waits** for some condition on shared object to become true
  - **Producer** thread changes condition, and **notifies** waiting threads that they can try again
- Implementation: ***Thread conditions***

Thread Conditions: Example

- See **conditions.py**
  - The job:
    - Same as lockinresource.py, except...
    - Disallows negative bank balances
      - `withdraw()` calls `condition.wait()`
        - » `withdraw()` must wait until balance is sufficiently large
      - `deposit()` calls `condition.notifyAll()`
        - » `deposit()` informs waiting threads to try again

7

Conditions: Example

[see slide]

Code notes:

Study code statically

Condition wraps around lock

`condition.wait()`

Releases the lock

Moves current thread from **runnable** state to **waiting** state

Upon return, reacquires lock

`condition.notifyAll()`

Moves all threads waiting for lock on this object from **waiting** state to **runnable** state

Study code dynamically

`withdraw()` calls `condition.wait()`

`withdraw()` must wait until balance is sufficiently large

`deposit()` calls `condition.notifyAll()`

`deposit()` informs waiting threads to try again

Thread Conditions: Example

- See **conditions.py** (cont.)

```
$ python conditions.py
1
2
3
4
5
6
7
8
9
10
8
6
4
2
0
Final balance: 0
$
```

```
$ python conditions.py
1
2
3
4
5
3
1
2
3
4
5
6
4
2
0
Final balance: 0
$
```

8

Conditions: Example

[see slide]

Note that balances are never negative

Thread Conditions: Example

- See **conditionsw.py**
  - The job:
    - Same as conditions.py
  - The difference:
    - Uses `with` statement

Thread Conditions: Pattern

Thread conditions pattern:

```
consumer thread
   while (! condition)
      wait();
   // Do what should be done when
   // condition is true.

producer thread
   // Change condition.
   notifyAll();
```

10

Conditions: Pattern

`condition.wait()`

Releases the lock

Moves current thread from **runnable** state to **waiting** state

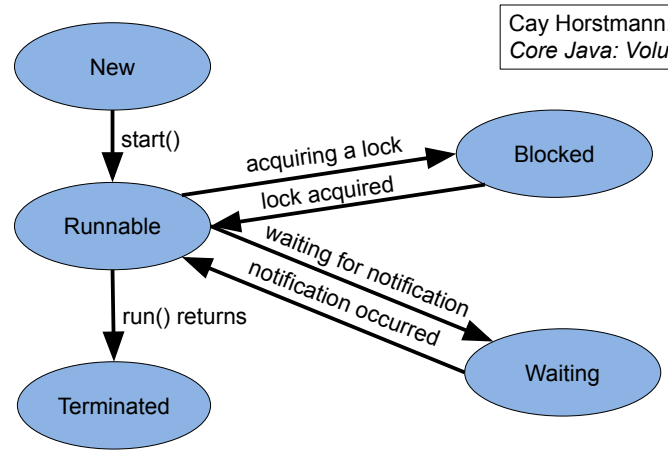
Upon return, reacquires lock

`condition.notifyAll()`

Moves all threads waiting on this object from **waiting** state to **runnable** state

Aside: Thread States

Cay Horstmann.
Core Java: Volume 1



At any time OS gives CPU(s) to Runnable thread(s)

Summary

- We have covered:
  - Thread conditions