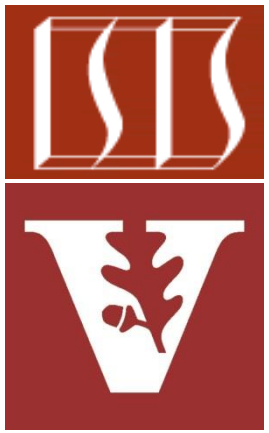
Java ConditionObject: Introduction

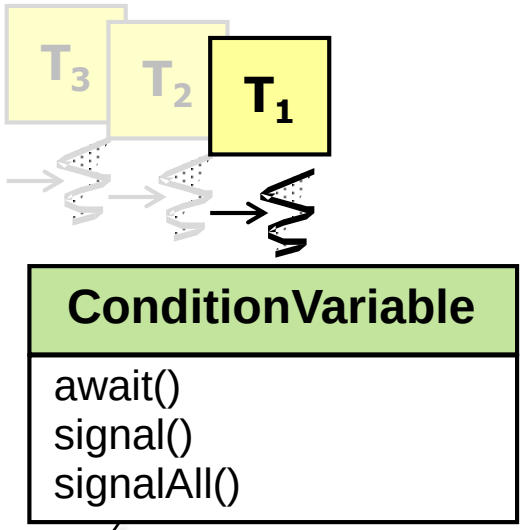


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

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

Learning Objectives in this Part of the Lesson

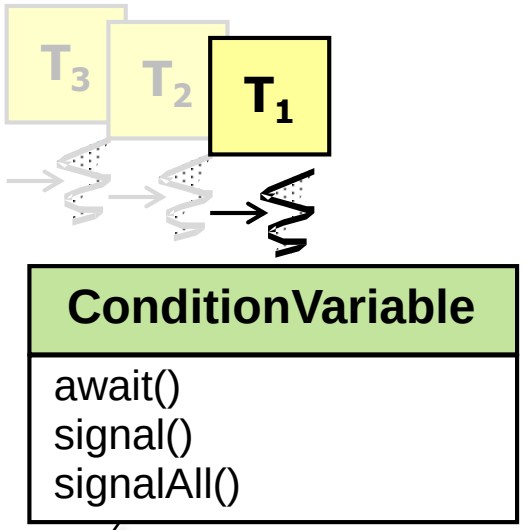
- Understand what condition variables are




```
Lock l = new Lock()
Condition cond = l.newCondition()
...
l.lock()
while (conditionNotSatisfied())
  cond.await()
doOperationProcessing()
```

Learning Objectives in this Part of the Lesson

- Understand what condition variables are




```
Lock l = new Lock()
Condition cond = l.newCondition()
...
l.lock()
while (conditionNotSatisfied())
  cond.await()
doOperationProcessing()
```

Condition variables can be tricky, so I recommend you rewatch this lesson & read the links carefully

Learning Objectives in this Part of the Lesson

- Understand what condition variables are
- Note a human known use of condition variables

Overview of Condition Variables

Overview of Condition Variables

- A CV is a synchronizer that allows a thread to (repeatedly) suspend its execution until a condition is satisfied

Wheel of Pain – Conan the Barbarian

See blog.dcoles.net/2012/02/understanding-how-to-use-condition.html

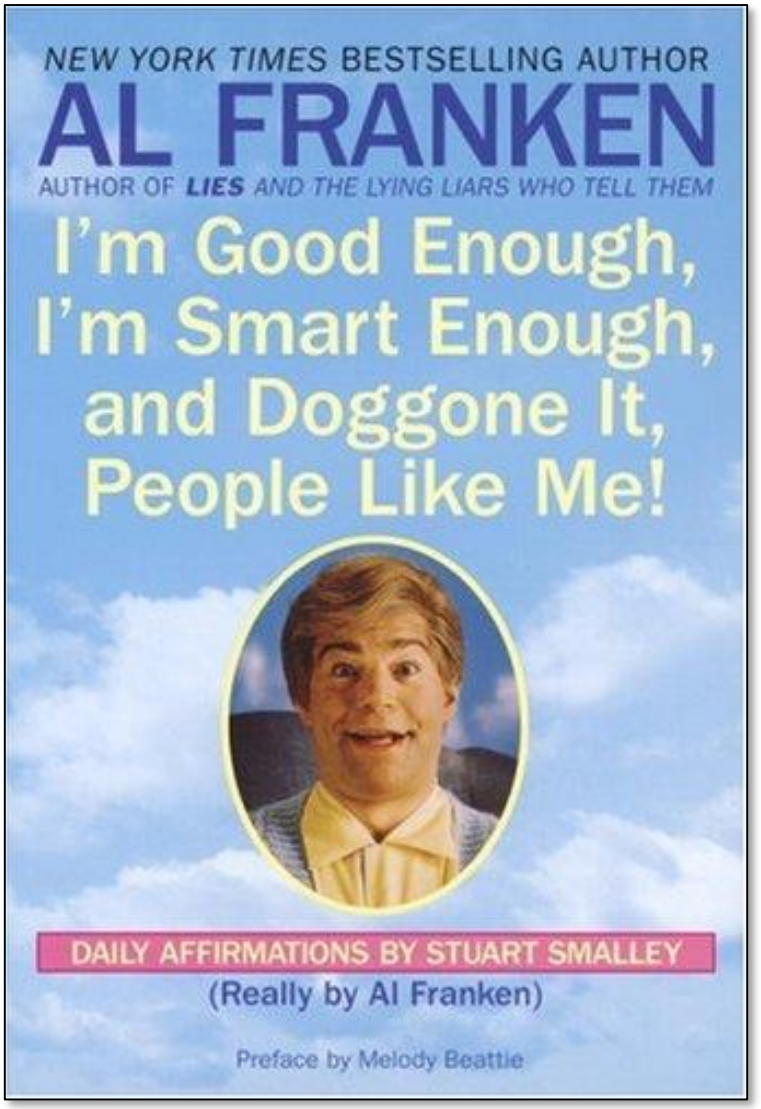
Overview of Condition Variables

- A CV is a synchronizer that allows a thread to (repeatedly) suspend its execution until a condition is satisfied
  - A thread whose execution is suspended on a CV is said to be "blocked" on the CV

Tree of Woe – Conan the Barbarian

Overview of Condition Variables

- CVs are often used when *mutual exclusion* alone is inadequate



Overview of Condition Variables

- CVs are often used when *mutual exclusion* alone is inadequate, e.g.
  - Inefficient use of resources
    - e.g., due to excessive "busy waiting" incurred by spin locks

See en.wikipedia.org/wiki/Busy_waiting & en.wikipedia.org/wiki/Spinlock

Overview of Condition Variables

- CVs are often used when *mutual exclusion* alone is inadequate, e.g.
  - Inefficient use of resources
  - Insufficient to ensure *coordination*
    - e.g., what to do when a thread encounters shared state that it can't do any work upon (yet)

Overview of Condition Variables

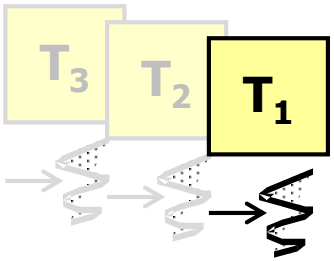
- CVs are often used when *mutual exclusion* alone is inadequate, e.g.
  - Inefficient use of resources
  - Insufficient to ensure *coordination*
    - e.g., what to do when a thread encounters shared state that it can't do any work upon (yet)

Waiting on an empty list

See en.wikipedia.org/wiki/Waiting_for_Godot

Overview of Condition Variables

- To alleviate the inadequacies of mutual exclusion, a CV is implemented as a queue of threads

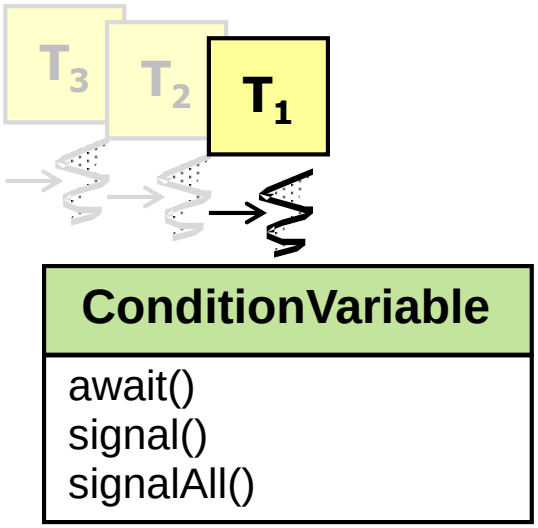


ConditionVariable
await() signal() signalAll()

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

Overview of Condition Variables

- To alleviate the inadequacies of mutual exclusion, a CV is implemented as a queue of threads

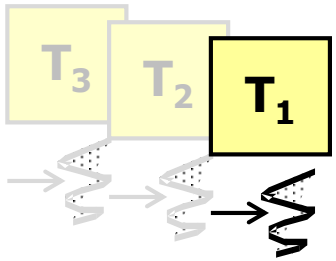


- This queue of threads is known as the "wait set"

See docs.oracle.com/javase/specs/jls/se7/html/jls-17.html#jls-17.2

Overview of Condition Variables

- To alleviate the inadequacies of mutual exclusion, a CV is implemented as a queue of threads



ConditionVariable

await()
signal()
signalAll()

- This queue of threads is known as the "wait set"
- Rather than "spinning," threads can choose to wait for certain condition(s) to be satisfied

```
Lock l = new Lock()
Condition cond = l.newCondition()
...
l.lock()
while (conditionNotSatisfied())
  cond.await()
doOperationProcessing()
```

Human Known Use of Condition Variables

Human Known Uses of Condition Variables

- A human known use is a pizza delivery protocol
  - Must acquire both the pizza & the keys to deliver the pizza

End of Java ConditionObject: Introduction