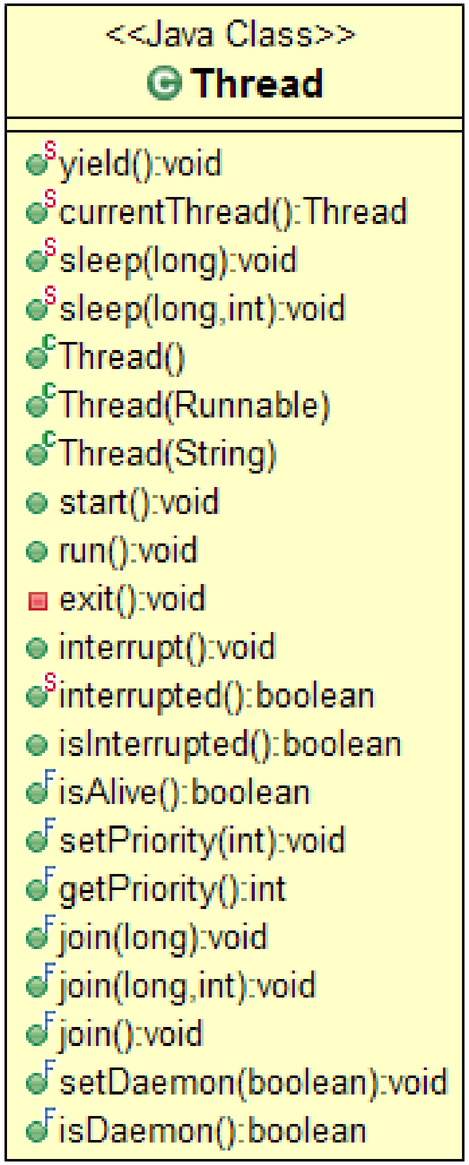
Java Thread: Key Class Methods

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

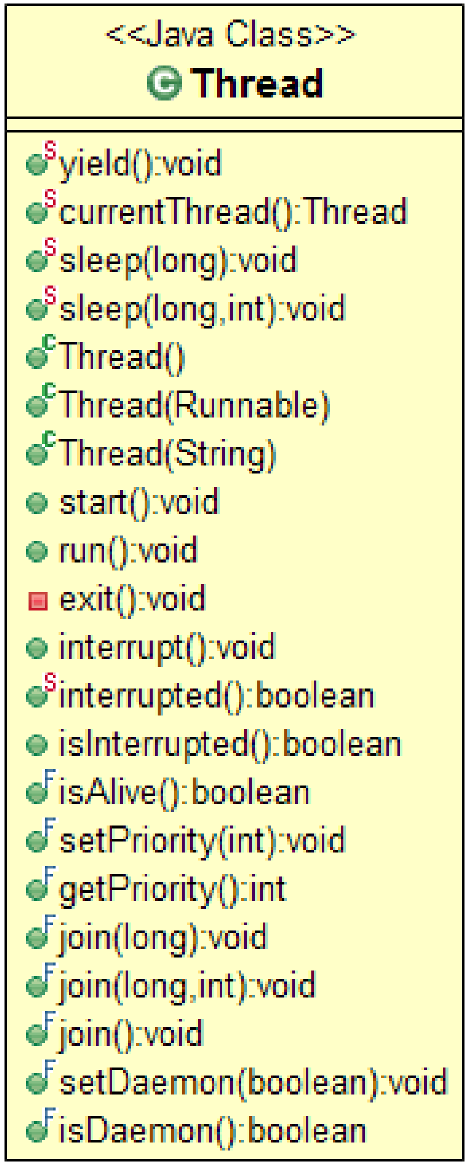
- Understand how Java threads support concurrency
- Learn how our case study app works
- Know alternative ways of giving code to a thread
- Learn how to pass parameters to a Java thread
- Know how to run a Java thread
- Recognize common thread methods



Key Java Thread Methods

Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs



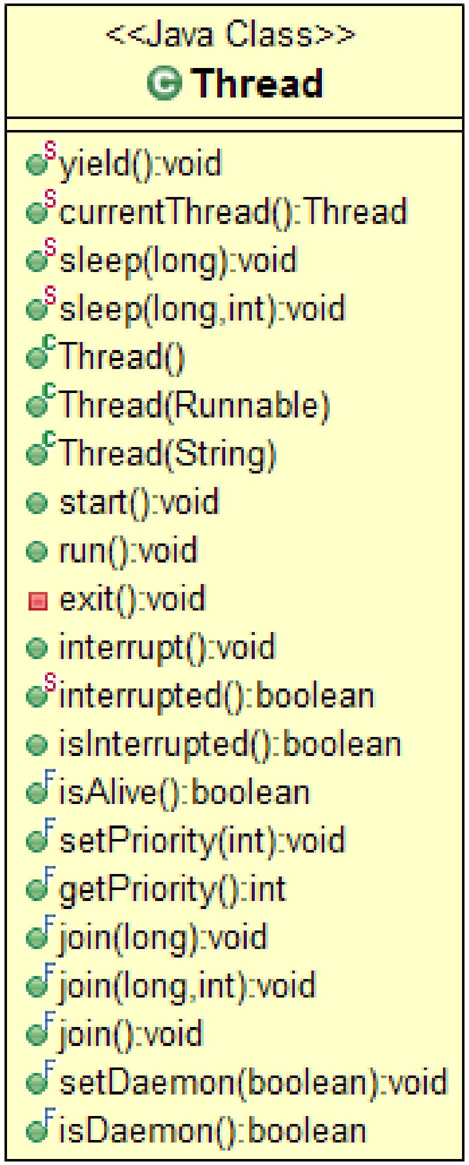
<<Java Class>>
Thread

```
yield():void
currentThread():Thread
sleep(long):void
sleep(long,int):void
Thread()
Thread(Runnable)
Thread(String)
start():void
run():void
exit():void
interrupt():void
interrupted():boolean
isInterrupted():boolean
isAlive():boolean
setPriority(int):void
getPriority():int
join(long):void
join(long,int):void
join():void
setDaemon(boolean):void
isDaemon():boolean
```

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

Key Java Thread Methods

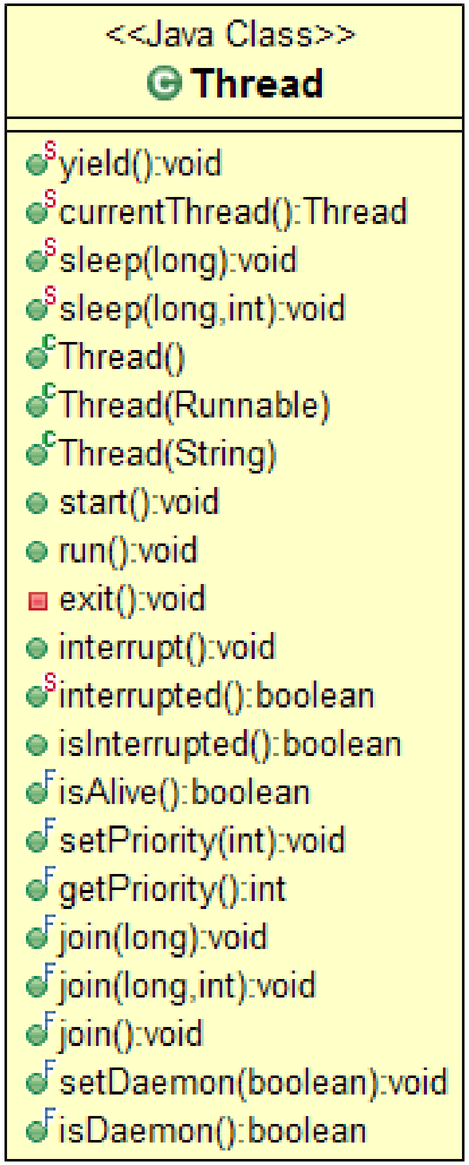
- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - **`void setDaemon()`**
    - Marks thread as a "daemon"



See javarevisited.blogspot.com/2012/03/what-is-daemon-thread-in-java-and.html

Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - **`void start()`**
    - Allocates thread resources & initiates thread execution by calling the run() hook method



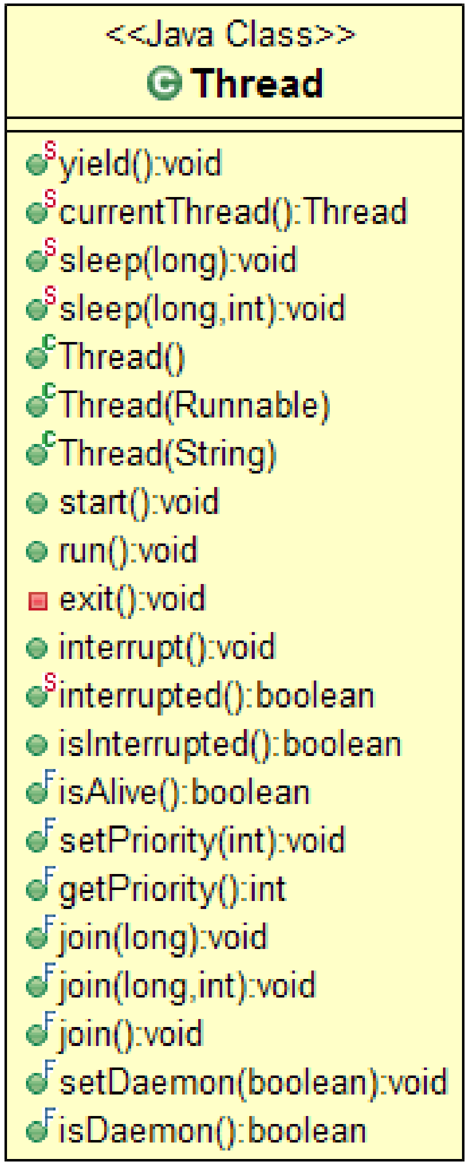
<<Java Class>>
Thread

```
yield():void
currentThread():Thread
sleep(long):void
sleep(long,int):void
Thread()
Thread(Runnable)
Thread(String)
start():void
run():void
exit():void
interrupt():void
interrupted():boolean
isInterrupted():boolean
isAlive():boolean
setPriority(int):void
getPriority():int
join(long):void
join(long,int):void
join():void
setDaemon(boolean):void
isDaemon():boolean
```

The start() method can only be called once per thread object

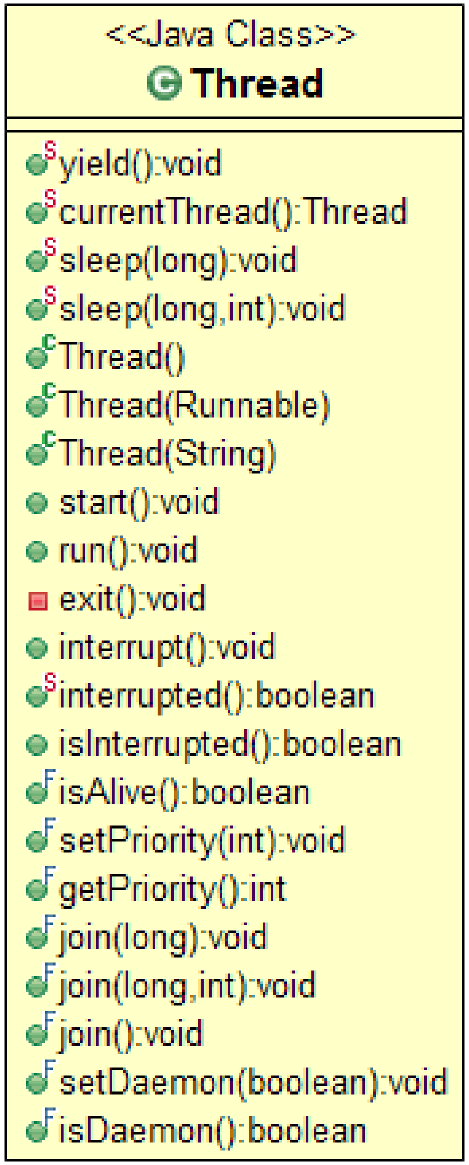
Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - **`void run()`**
    - Hook method where user code is supplied



Key Java Thread Methods

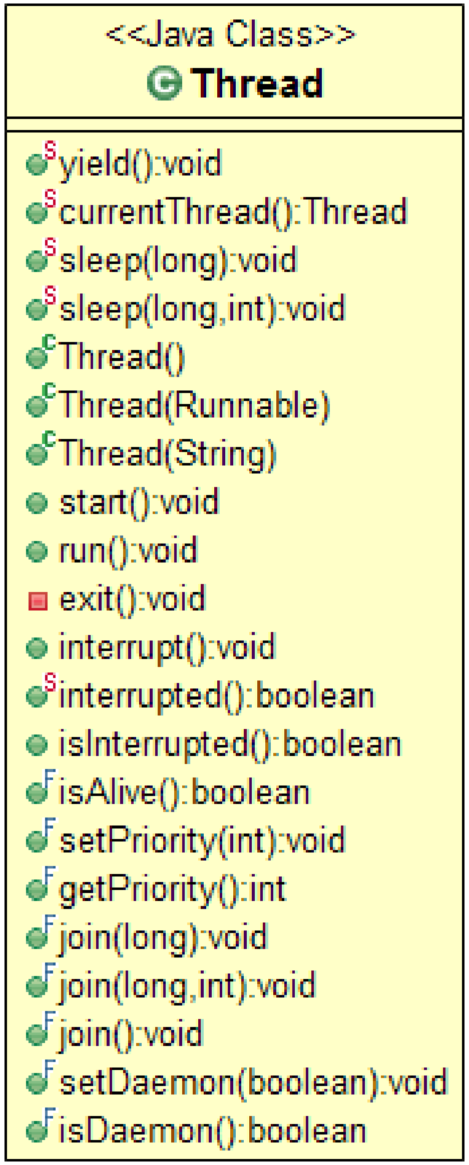
- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - **`void join()`**
    - Waits for a thread to finish



A simple form of "barrier synchronization"

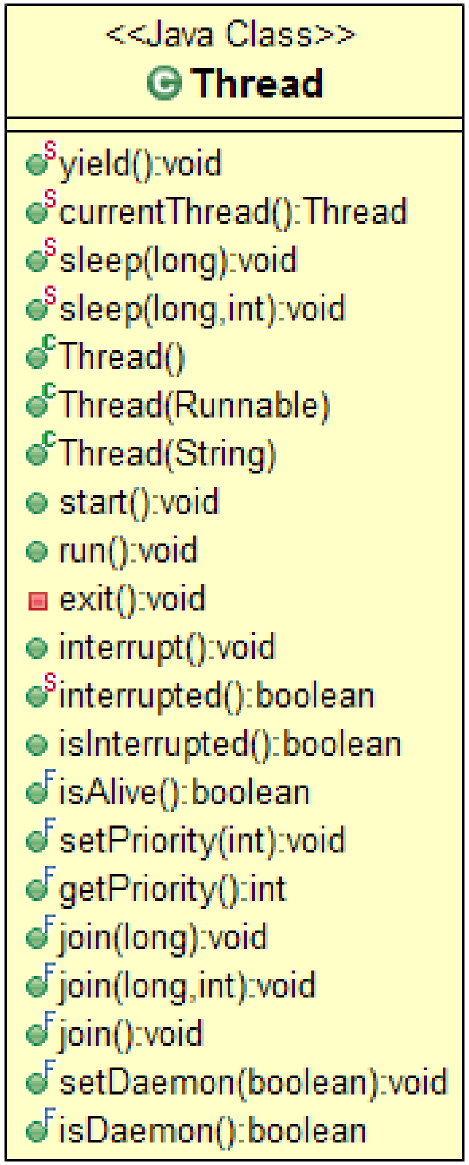
Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - **`void sleep(long time)`**
    - Sleeps for given time in ms



Key Java Thread Methods

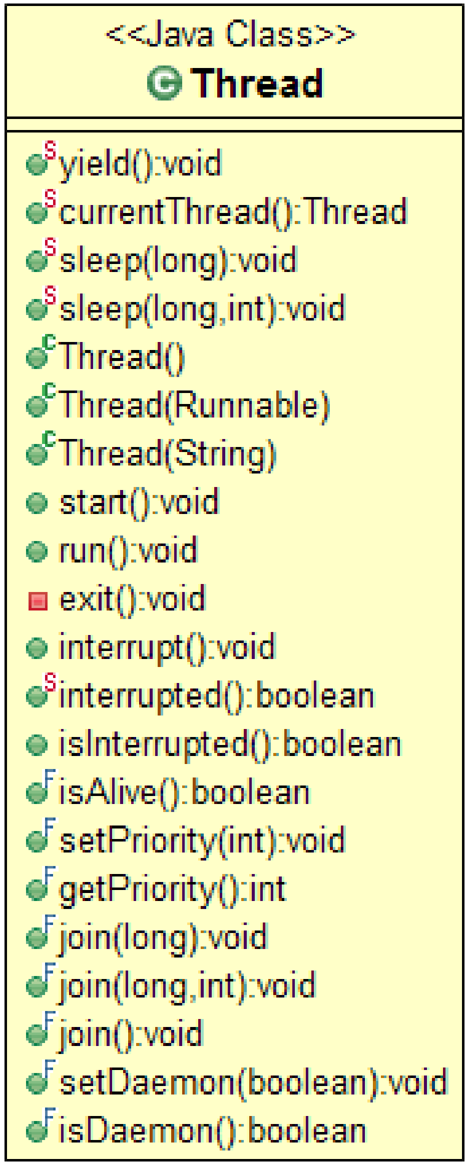
- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - `void sleep(long time)`
  - **`Thread currentThread()`**
    - Object for current Thread


<<Java Class>>
Thread

```
yield():void
currentThread():Thread
sleep(long):void
sleep(long,int):void
Thread()
Thread(Runnable)
Thread(String)
start():void
run():void
exit():void
interrupt():void
interrupted():boolean
isInterrupted():boolean
isAlive():boolean
setPriority(int):void
getPriority():int
join(long):void
join(long,int):void
join():void
setDaemon(boolean):void
isDaemon():boolean
```

Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - `void sleep(long time)`
  - `Thread currentThread()`
  - **`void interrupt()`**
    - Post an interrupt request to a Thread



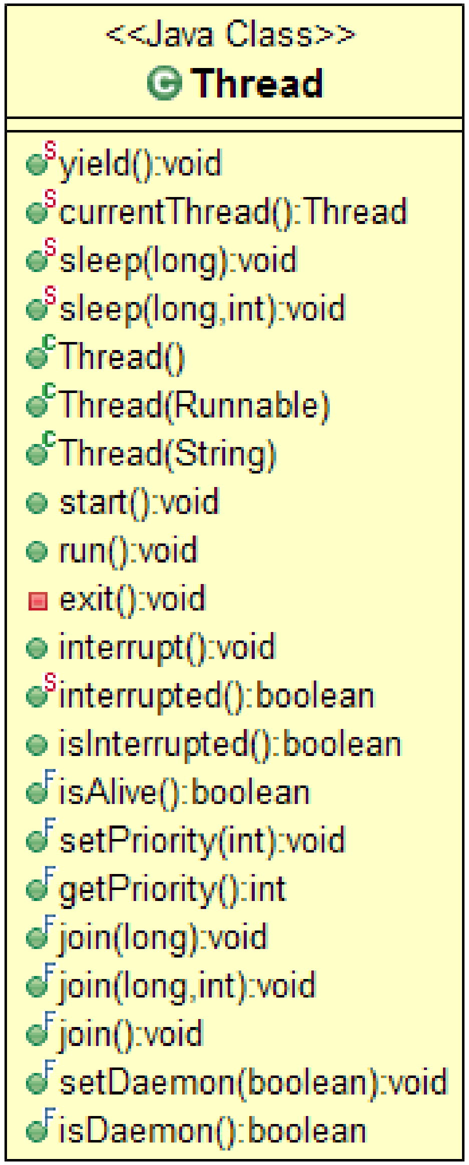
```
<<Java Class>>
Thread

yield():void
currentThread():Thread
sleep(long):void
sleep(long,int):void
Thread()
Thread(Runnable)
Thread(String)
start():void
run():void
exit():void
interrupt():void
interrupted():boolean
isInterrupted():boolean
isAlive():boolean
setPriority(int):void
getPriority():int
join(long):void
join(long,int):void
join():void
setDaemon(boolean):void
isDaemon():boolean
```

See upcoming lesson on "*Managing the Java Thread Lifecycle*"

Key Java Thread Methods

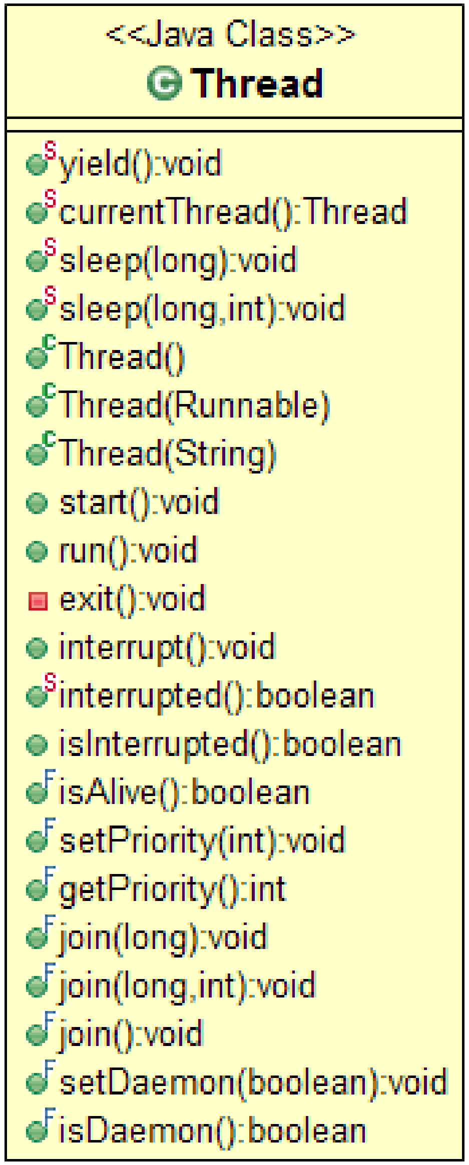
- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - `void sleep(long time)`
  - `Thread currentThread()`
  - `void interrupt()`
  - **`boolean isInterrupted()`**
    - Tests whether a thread has been interrupted



isInterrupted() can be called multiple times w/out affecting *interrupted status*

Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - `void sleep(long time)`
  - `Thread currentThread()`
  - `void interrupt()`
  - `boolean isInterrupted()`
  - **`boolean interrupted()`**
    - Tests whether current thread has been interrupted



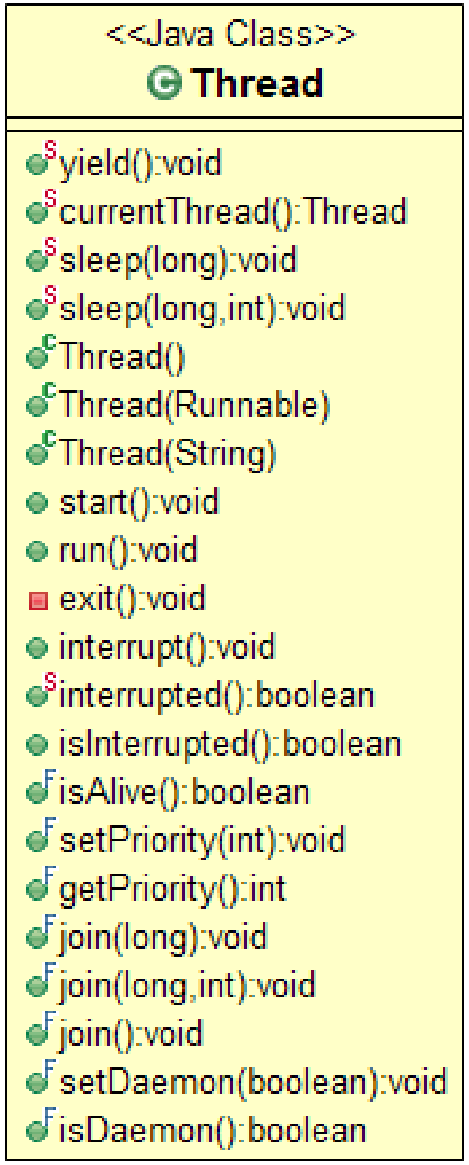
```
<<Java Class>>
Thread

yield():void
currentThread():Thread
sleep(long):void
sleep(long,int):void
Thread()
Thread(Runnable)
Thread(String)
start():void
run():void
exit():void
interrupt():void
interrupted():boolean
isInterrupted():boolean
isAlive():boolean
setPriority(int):void
getPriority():int
join(long):void
join(long,int):void
join():void
setDaemon(boolean):void
isDaemon():boolean
```

interrupted() clears the *interrupted status* the first time it's called

Key Java Thread Methods

- Certain Java Thread class methods are used in many concurrent Java programs, e.g.
  - `void setDaemon()`
  - `void start()`
  - `void run()`
  - `void join()`
  - `void sleep(long time)`
  - `Thread currentThread()`
  - `void interrupt()`
  - `boolean isInterrupted()`
  - `boolean interrupted()`
  - **`void setPriority(int newPriority)` & `int getPriority()`**
    - Set & get the priority of a Thread



High values of **`newPriority`** result in higher priority threads

End of Java Thread: Key Class Methods