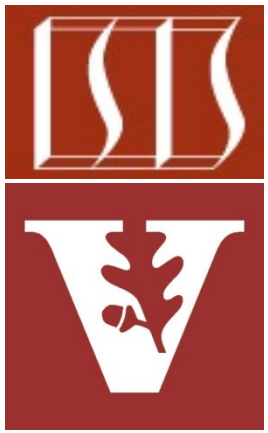
# Types of Java Threads (Part 2)

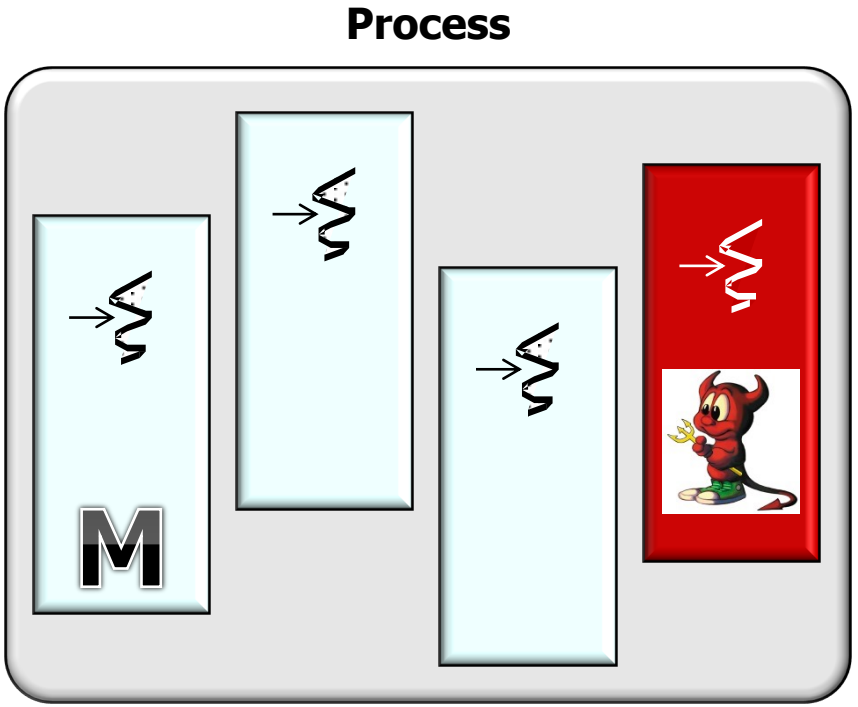


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

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

# Learning Objectives in this Part of the Lesson

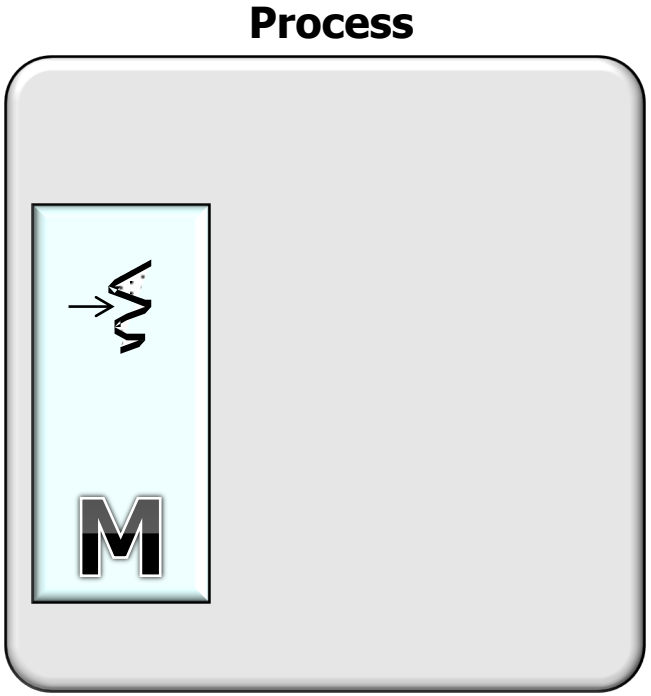
- Understand how Java threads support concurrency
- Learn how our case study app works
- Know alternative ways of giving code to a thread
- Learn how to pass parameters to a Java thread
- Know the differences between Java platform & virtual threads
- Be aware of how a Java thread starts & runs
- Recognize common thread methods
- Be aware of the different types of Java threads
  - Know how to program user & daemon threads



# Java User Threads vs. Daemon Threads (Example 1)

# User Threads vs. Daemon Threads (Example 1)

- Demonstrates the difference between a Java user thread & a daemon thread

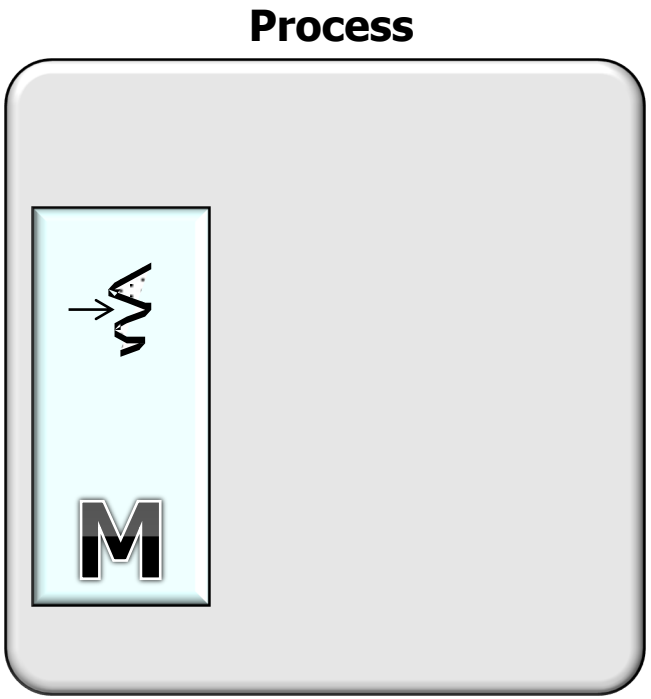



```
public class UserOrDaemonThread
             extends Thread  {
  ...
  private int computeGCD
    (int number1, int number2)
  { ... }

  public void run() {
    ...
    computeGCD(number1, number2);
    ...
  }

  public UserOrDaemonThread(Boolean daemonThread) {
  if (daemonThread) {
    setDaemon(true);
   ...
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/UserOrDaemonThread

# User Threads vs. Daemon Threads (Example 1)

- Demonstrates the difference between a Java user thread & a daemon thread

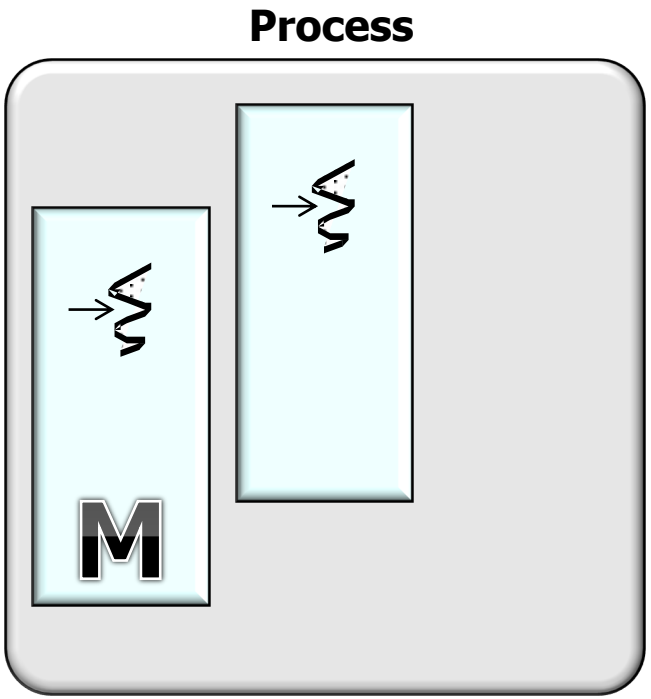



```
public class UserOrDaemonThread
             extends Thread  {
  ...
  private int computeGCD
    (int number1, int number2)
  { ... }

  public void run() {
    ...
    computeGCD(number1, number2);
    ...
  }

  public UserOrDaemonThread(Boolean daemonThread) {
  if (daemonThread) {
    setDaemon(true);
   ...
```

Extends Thread, generates random numbers, & computes their "Greatest Common Divisor" (GCD)

# User Threads vs. Daemon Threads (Example 1)

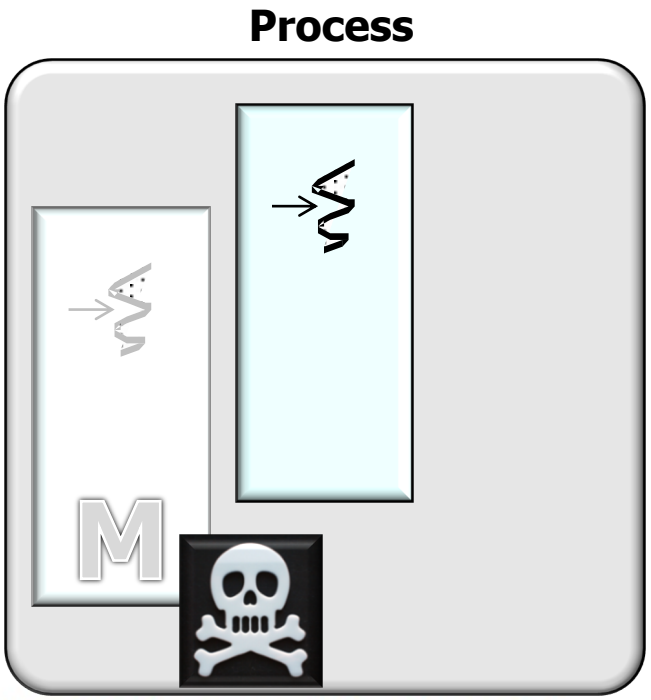
- Demonstrates the difference between a Java user thread & a daemon thread
  - If launched with no command-line parameters the main thread creates a user thread




```
public static void
  main(String[] args) {
  final boolean daemonThread =
    args.length > 0;

  // Create thread type
  UserOrDaemonThread thr =
    new UserOrDaemonThread(daemonThread);

  thr.start();
...
```

# User Threads vs. Daemon Threads (Example 1)

- Demonstrates the difference between a Java user thread & a daemon thread
  - If launched with no command-line parameters the main thread creates a user thread

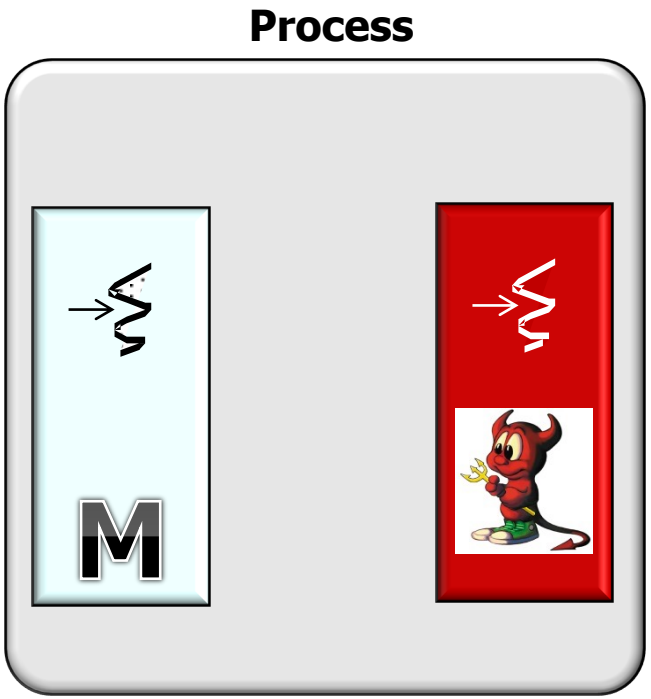



*The user thread can outlive the main thread*

```
Entering main()
Entering run() with user thread id Thread[Thread-0,5,main]
In run() with user thread id Thread[Thread-0,5,main] the GCD of 143699154 and 222547454 is 2
Leaving main()
In run() with user thread id Thread[Thread-0,5,main] the GCD of 490663306 and 1105718378 is 2
In run() with user thread id Thread[Thread-0,5,main] the GCD of -1689926891 and -227942117 is -1
In run() with user thread id Thread[Thread-0,5,main] the GCD of 899726708 and 390462480 is 4
In run() with user thread id Thread[Thread-0,5,main] the GCD of -1567920985 and -1959228087 is -1
In run() with user thread id Thread[Thread-0,5,main] the GCD of -1686019921 and 188605637 is -1
In run() with user thread id Thread[Thread-0,5,main] the GCD of -583128694 and 915559046 is 2
In run() with user thread id Thread[Thread-0,5,main] the GCD of 666720057 and -1900927349 is -1
In run() with user thread id Thread[Thread-0,5,main] the GCD of 1044019644 and 2002366675 is 1
In run() with user thread id Thread[Thread-0,5,main] the GCD of -416210668 and 914702688 is -116
Leaving run()  with user thread id Thread[Thread-0,5,main]
```

# User Threads vs. Daemon Threads (Example 1)

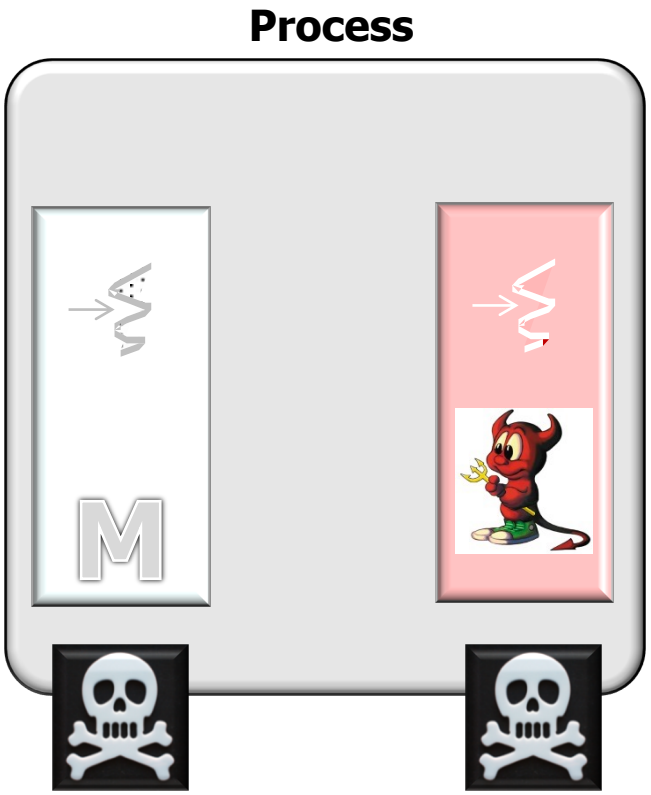
- Demonstrates the difference between a Java user thread & a daemon thread
  - If launched with no command-line parameters the main thread creates a user thread
  - If launched with a command-line parameter it creates a daemon thread




```
public static void
  main(String[] args) {
  final boolean daemonThread =
    args.length > 0;

  // Create thread type
  UserOrDaemonThread thr =
    new UserOrDaemonThread(daemonThread);

  thr.start();
...
```

# User Threads vs. Daemon Threads (Example 1)

- Demonstrates the difference between a Java user thread & a daemon thread
  - If launched with no command-line parameters the main thread creates a user thread
  - If launched with a command-line parameter it creates a daemon thread




*The daemon thread exits when the main thread exits*

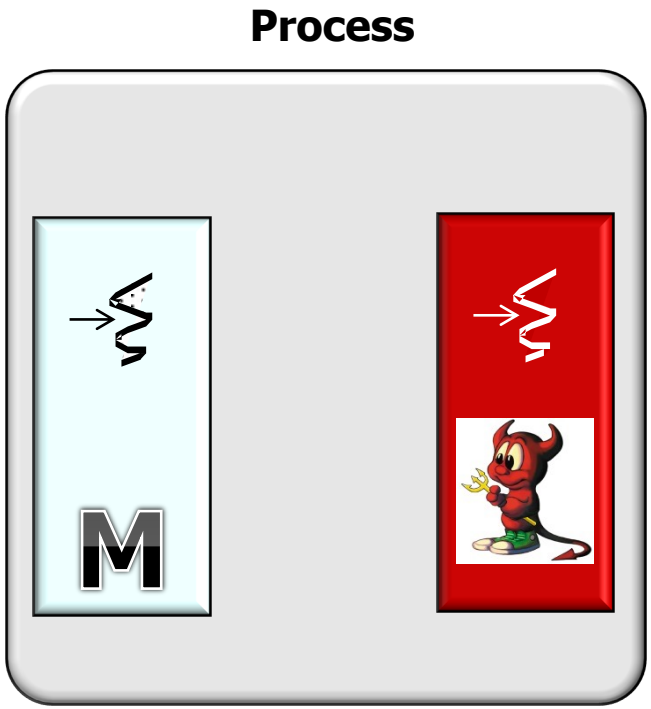
```
Entering main()
Entering run() with daemon thread id Thread[Thread-0,5,main]
In run() with daemon thread id Thread[Thread-0,5,main] the GCD of 808096814 and 1606093510 is 14
Leaving main()
```

# Java User Threads vs. Daemon Threads (Example 2)

# User Threads vs. Daemon Threads (Example 2)

- Demonstrates the difference between a Java user thread & a daemon thread

```
public class GCDRunnable
             extends Random
             implements Runnable {
  ...
  private int computeGCD
      (int number1,
       int number2) {
    ...
  }

  public void run() {
    ...
  }
  ...
}
```

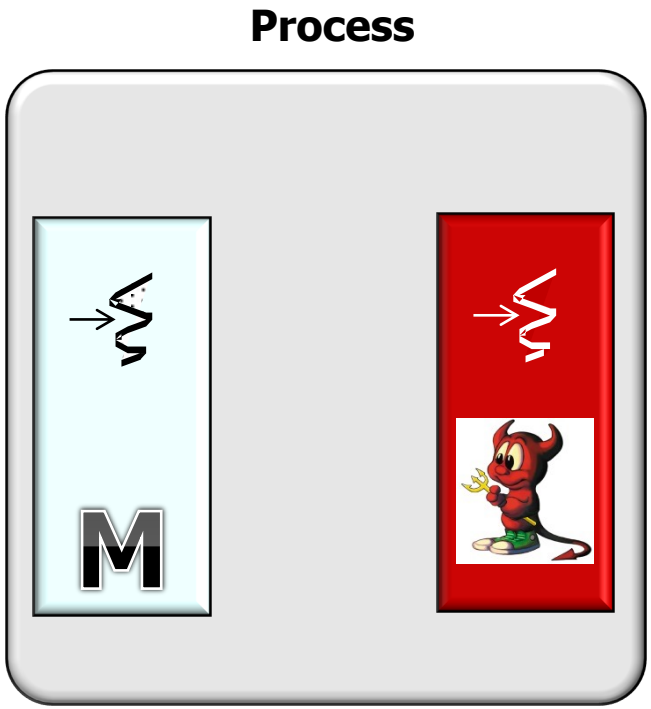



See github.com/douglascraigschmidt/LiveLessons/tree/master/UserOrDaemonRunnable

# User Threads vs. Daemon Threads (Example 2)

- Demonstrates the difference between a Java user thread & a daemon thread

```
public class GCDRunnable
             extends Random
             implements Runnable {
  ...
  private int computeGCD
      (int number1,
       int number2) {
    ...
  }

  public void run() {
    ...
  }
  ...
}
```

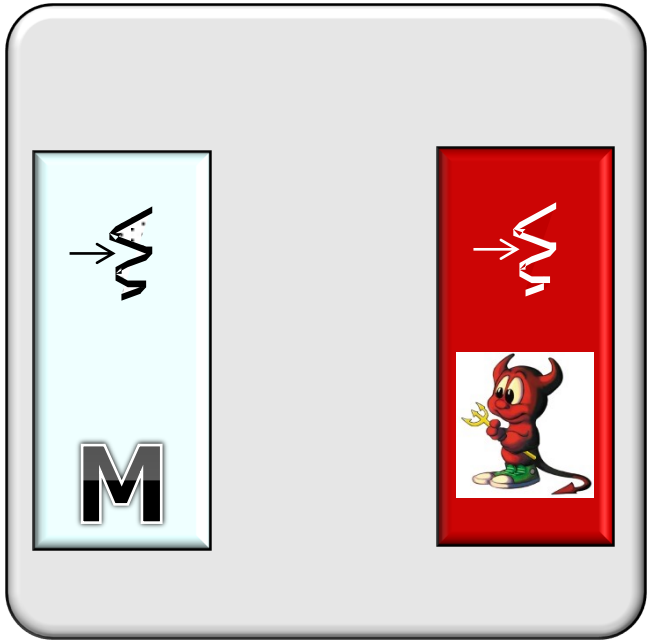



Java doesn't allow multiple inheritance of classes, so implement Runnable

# User Threads vs. Daemon Threads (Example 2)

- Demonstrates the difference between a Java user thread & a daemon thread

```
public static void main(String[] args){
  final boolean daemonThread =
    args.length > 0;

  GCDRunnable runnableCommand =
    new GCDRunnable(daemonThread ?
                    "daemon" : "user");

  Thread thr =
    new Thread(runnableCommand);

  if (daemonThread)
    thr.setDaemon(true);

  thr.start();
```




Create a new thread to execute the GCDRunnable command concurrently

# End of Types of Java Threads (Part 2)