UNIVERSITY OF TWENTE.

Threads in Java

Topic of Software Systems (TCS module 2)

Lecturer: Marieke Huisman

CREATING A THREAD

- As a subclass of Thread
  - Define a subclass of class Thread
  - Override the inherited method run
  - Construct an object of your subclass
  - Call the method start to start the thread!

- Better: implement a Runnable
  - Define class that implements the interface Runnable
  - Implement the method run
  - Construct an object of your class
  - Construct a new Thread(yourRunnableObject) and start() it!

EXAMPLE

What will happen if you run this?

```
public class SaySomething implements Runnable {
    private String text;
    public SaySomething(String text) {
        this.text = text;
    }
    public void run() {
        for (int i=0; i<1000; i++) System.out.println(text);
    }
}

Thread threadOne = new Thread(new SaySomething(“Hello from thread 1!”));
Thread threadTwo = new Thread(new SaySomething(“Hello from thread 2!”));
threadOne.start();
threadTwo.start();
```

Random sequence of:
Hello from thread 1!
Hello from thread 2!

EXAMPLE

What will happen if you run this?

```
public class SaySomething implements Runnable {
    private String text;
    public SaySomething(String text) {
        this.text = text;
    }
    public void run() {
        for (int i=0; i<1000; i++) System.out.println(text);
    }
}

for (int i=0; i<1000; i++) {
    new Thread(new SaySomething(“Hello from thread “ + i + !”)).start();
}
```

Random sequence of:
Hello from thread ...!
With values from 0 to 999

THREADING FUNCTIONALITY

After creating a Thread t, you can do things with the thread

- Call `t.start()` to start the thread
- Call `t.join()` to wait until the thread is terminated
- Call `t.join(n)` to wait n milliseconds or until the thread is terminated
- Call `t.interrupt()` to cause an `InterruptedException` to be thrown in the running thread
- Use `setName(name)` and `getName()` to name your thread and retrieve its name
- Call `t.setDaemon(true)` before starting the thread to make it a daemon thread
  - If all non-daemon threads are terminated, the program terminates
  - Use daemon threads for background supporting tasks

- Methods `suspend`, `resume`, `stop` are deprecated
  - They often lead to deadlocks, so it is not a good practice to use them