Lec 18

Threads & Inheritance
Multi-threaded
public class Sum extends Thread {
    int start, stop, sum;
    public Sum(int start, int stop) {
        //initialize vars.
    }
    public void run() {
        for(int i = start; i <= stop; i++) {
            sum += i;
        }
    }
}

public class Example {
    private static final int SUM_NUMBERS = 5;
    public static void main(String[] args) {
        Thread t1 = new Sum(0, SUM_NUMBERS);
        t1.start();
        System.out.println(((Sum) t1).sum); //note: this is ok!
    }
}
How would you modify this to compute the answer faster

```java
public class Sum extends Thread {
    int start, stop, sum;
    public Sum(int start, int stop) {
        //initialize vars.
    }
    public void run() {
        for(int i = start; i <= stop; i++) {
            sum += i;
        }
    }
}
```

```java
public class Example {
    private static final int SUM_NUMBERS = 5000000;
    public static void main(String[] args) {
        Thread t1 = new Sum(0, SUM_NUMBERS/2);
        Thread t2 = new Sum(SUM_NUMBERS/2+1, SUM_NUMBERS);
        t1.start();
        try {
            t1.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        System.out.println(((Sum) t1).sum); //note: this is ok!
    }
}
```
Can/should we parallelize or thread everything?

• Suppose you are playing a board game. Why make everyone take their turn in sequence? Can’t everyone go at the same time?
Threading

• You get up for the morning and here are your tasks before you can leave the house:
  – Brew coffee (takes 5 minutes)
  – Pour coffee in to-go mug (1 minute)
  – Get dressed (4 minutes)
  – Make cereal (2 minutes)
  – Eat cereal (5 minutes)
  – Brush teeth (2 minutes)
  – Turn on tablet to download podcast (3 minutes)

How long did it take to get ready?
Reasons to use threads

- HW2 - turtles
- HW3 – triforce
- HW4 – poker
- HW5 – picture recovery
- HW6 - WordCloud

Avoid blocking on background actions

- Should your GUI stop updating when waiting for, say, the hard drive to return a value?
- Two terrible examples:
  - Your GPS not updating the screen while doing something else
  - Windows Print manager locking down the whole system
Threading vs. Parallel Programming

• Threading
  – Commonly refers to the scenario where you launch a thread to take care of something which may take a while
  – Think coffee maker example (happens in the background)

• Parallel Programming
  – Commonly refers to the scenario where you have more than one computing resource and are trying to divide up your task
  – Think summing example
Back to GTA
Inheritance
FastCar

- topSpeed: int
- bodyType: String
- color: String
- damage: int

+ Car()
+ Car(topSpeed: int, bodyType: String, color: String, damage: int)
+ accelerate()
+ reverse()
+ brake()
### Tank

- `topSpeed: int`
- `bodyType: String`
- `color: String`
- `damage: int`
- `ammunition: int`

```java
public class Tank {
    private int topSpeed;
    private String bodyType;
    private String color;
    private int damage;
    private int ammunition;

    public Tank() {
    }

    public Tank(int topSpeed, String bodyType, String color, int damage) {
    }

    public void accelerate() {
    }

    public void reverse() {
    }

    public void brake() {
    }

    public void fire() {
    }
}
```
Zamboni

- topSpeed: int
- bodyType: String
- color: String
- damage: int

+ Car()
+ Car(topSpeed: int, bodyType: String, color: String, damage: int)
+ accelerate()
+ reverse()
+ brake()
public class XXX extends Car

Car
- damage: int
- color: String
+ Car()
+ Car(damage: int, color: String)
+ accelerate(): void
+ break(): void

FastCar
+ FastCar(damage: int, color: String)

Zamboni
+ Zamboni(damage: int, color: String)

Tank
- ammo: int
+ Tank(damage: int, color: String, ammo: int)
+ fire(): void
class RX7 extends Car {
    private int NOS;
    public RX7() {
        NOS = 10;
    }

    public void useNOS() {
        System.out.println("hold on");
        NOS--;
        accelerate()
    }
}

class Car {
    private String color;
    public Car() {
        color = "RED";
    }
    public void accelerate() {
        System.out.println("let's GO");
        System.out.println("let's GO");
        System.out.println("let's GO");
    }
}

class Start {
    public static void main(...) {
        RX7 dom = new RX7();
        dom.useNOS();
        dom.accelerate();
    }
}
Inheritance

https://www.youtube.com/watch?v=KgJs95dzFE0
class RX7 extends Car {
    private int NOS;
    public RX7() {
        NOS = 10;
    }

    public void useNOS() {
        System.out.println("hold on");
        NOS--;
        accelerate()
    }
}

class Car {
    private String color;
    public Car() {
        color = "RED";
    }
    private void accelerate() {
        System.out.println("let’s GO");
    }
}

class Start {
    public static void main(...) {
        RX7 dom = new RX7();
        dom.useNOS();
        dom.accelerate();
    }
}
Inheritance – Key Points

Superclass

class Superclass

Subclass

class Subclass extends Superclass

Subclass **is-a** Superclass

Not the other way around

Superclass members become members of the Subclass
(inheritance)

- except private methods
- private instance variables accessible through public superclass methods

Implicit **extends** Object if no explicit extends
Details

• Can only extend one class
  – But that class could extend its own class, e.g.:

    ```java
    public class USTank extends Tank {}
    public class Tank extends Vehicle {}
    public class Vehicle {}
    ```

• Eventually, every class extends Object
  – if not explicitly, then implicitly