

Computing Tutorial: Week 3

Introduction and Subclasses

Tony Chung a.chung@lancaster.ac.uk

Objectives Today

- Get to know each other!
- Introduction
- Subclasses
- Today's slides were based on those from Simon Lock.
 - It went horribly wrong.
 - So I've simplified it.
- Every week slides may cover different modules.
- **You can suggest topics to cover at any time over email or at the end of the class.**

Tutorials

- Your class is every week until week 25.
- The location is Fylde C32.
- Attendance at all Tutorials and Practicals is **mandatory**.
 - Things have been tightened up significantly this year.
- Support is available from many sources:
 - <http://info.comp.lancs.ac.uk/> - IntraNet
 - <http://javacafe.v6testbed.net/> - Forum
 - Signup required. No code allowed.
 - a.chung@lancaster.ac.uk - Me
- General or Admin staff: Cath Ewan or Simon Lock

What Is a Tutorial?

- A tutorial is meant to:
 - Re-enforce material from the practicals and lectures.
 - A different perspective.
 - Give you a chance to ask questions.
 - Give you practise dealing with problems.
- It's interactive.
 - Not a lecture.
 - You will be expected to talk.
 - You will be working in groups.

Types

- All data items in Java must be of a specific Type.
- In Java, there are two forms of Type.
 - Primitives are the simplest form of Type.
 - int, char, boolean, float
 - Classes are complex types.
 - Made up of Primitives and other Classes.
 - String (an array of chars), Date, etc
- Classes also define functions.
 - Some languages have Structures. Structures are Classes without functions.

Variables

- All data must be stored somewhere.
- In Java we Declare a placeholder called a variable.
- Giving that placeholder a Type.
- If we use Classes, we must create an Instance of that class called an Object.
 - In today's class, we are going to create a custom Class, and demonstrate how to create an Object of that Class.
 - We will then create a Subclass, and then demonstrate how to deal with inheritance.

Methods

- All instructions live in methods.
- All methods have a name. (Starting with a lower case letter.)
- All methods have a return type. (Even if it is void.)
- All methods can take parameters. (But it is not mandatory.)
- All methods belong in Classes.
 - **In processing, you are actually working in a default class. You just can't see it. It already has 500 or so methods.**

```
type methodName( type argName1, type argName2 ){
  ...
  return value;
}
```

Classes

Classes, Methods, Properties, Accessors, Constructor, Shadowing

```
public class Tower{
  private int height;
  public Tower( int height ){
    this.height = height;
  }
  public int getHeight(){
    return height;
  }
}
```

Subclasses

- Subclasses Inherit from a Superclass. They inherit all the Properties and Methods.
- They can add additional properties and methods.
- They can override existing properties and methods.
- You can force Java to use something in the Superclass by using the keyword super.

```
public class Structure extends
  Tower{
  private int width;
  public Tower( int width, int
    height ){
    super( height );
    this.width = width;
  }
  public int getWidth(){
    return width;
  }
  public int getArea(){
    return width * height;
  }
}
```

Exercise 1

- Create a TribbleEatingBear.
- Provide a simple constructor
- Write an eatTribble method which takes one parameter:
 - The Tribble that is to be eaten!
- Increase the size of the bear by using the size of the Tribble.
- Write a method to test the new Class.

```
public class Bear{
  protected int size;
  public Bear( int size ){
    this.size = size;
  }
  public int getSize(){
    return size;
  }
}

public class Tribble{
  private int size;
  public int getSize(){
    return size;
  }
}
```

Exercise 2: Handling Key Presses

- Write a keyPressed method so that:
 - If user presses 'a' Tribble moves up 20 pixels
 - If user presses 'z' Tribble moves down 20 pixels
 - If user presses 'o' Tribble moves left 20 pixels
 - If user presses 'p' Tribble moves right 20 pixels
 - If user presses space bar:
 - The Bear goes and eats a Tribble
- Bear extends Creature.
 - Creature has the following methods:
 - void setLocation(int x, int y)
 - void kill()
 - void setSize(int size)
 - int getX()
 - int getY()
 - void setX(int x)
 - void setY(int y)

That's All Folks...

- Any questions?
- Slides will go online to <http://www.tonychung.net/>
- Questions and tutorial suggestions to a.chung@lancaster.ac.uk