CMSC 131
Object-Oriented Programming I

Static Variables, Instance Variables

Initialization

Dept of Computer Science
University of Maryland College Park

This material is based on material provided by Ben Bederson, Bonnie Dorr, Fawzi Emad, David Mount, Jan Plane
Overview

- Instance Variable Initialization
- Static Variables
- Quick intro to inheritance
Some Sites

- First bug
  - http://www.jamesshuggins.com/h/tek1/first_computer_bug.htm

- Funny Quotes
  - http://www.comedy-zone.net/quotes/Science_and_Technology/programming.htm

- To keep informed
  - http://slashdot.org/
Initialization of Variables

- **Note: Review**
  - An object is an instance of a class
  - The class is a blueprint (only one)
- Remember that instance variables have default values if they are not initialized
  - boolean → false
  - numeric → zero
  - references → null
- Instance variables can be initialized when they are declared
- Instance variables can be initialized in the constructor
- The constructor will override
- Let’s see an example with our SuperHero class
  - Making one of our initialization an object initialization
Static Variables

- We have seen static methods
  - Methods that do not require an object in order to be called
  - They don’t refer to any instance variables
  - We called them using the name of the class
  - They can also be called using an object of the class but that is not recommended (e.g., you get warning in Eclipse)
Static Variables

- We can have static variables
  - Variable that is shared by all instances of the class
  - There is only one copy
  - You can access it by using the name of the class
    - ClassName.staticVariableName
  - In methods of the class you can access the variable directly (no need for class names)
- Let’s see a diagram that shows how static variables are shared
Static Variables

- **Uses**
  - If you have a constant that applies to all instances of the class, then define it as static constant
  - If you need to keep track of instances of a class

- Let’s see how to define them
  - Adding superhero count to our SuperHero class

- When is a static variable initialized?
  - At class load time
  - In the constructor
Scope

- What is the scope (visibility) of:
  - Instance variables
  - Static variables
  - Local variables
  - Parameters
**Review Of Variables**

- **Instance variables**
  - Belong to the class and created when an object is created
  - Space for them exists in the heap
  - We need an object to access them
  - In a non-static method we can refer to them directly (no need for object reference)
  - They have default values

- **Static variables**
  - Belong to the class
  - They are shared by all instances of the class
  - We can refer to them by using the class name (no need for class name if reference by methods of the class)

- **Local Variables**
  - Defined in a method
  - Created when method is called and destroy on exit
  - Don’t have default values (Eclipse will know if you are using uninitialized variables and force you to initialized them)

- **Parameters**
  - Like local variables but initialized when a method is called
Quick Introduction To Inheritance

- We can use one class to create another by using inheritance
- Let’s define a MegaSuperHero class based on our SuperHero class by using extends