CS2136: Paradigms of Computation

Class 13: Constructors

More on Classes and Inheritance

Final

Static

Copyright 2001, 2002, 2003 Michael J. Ciaraldi and David Finkel
Constructors

- Unless you say otherwise, the default constructor of each class’s parent class is invoked at the start of the constructor for that class.
  - Default constructor has no parameters.
- Can invoke non-default constructor explicitly.
  - Invocation must be the first statement in the constructor.
More on Classes and Inheritance

- Visibility
- Which Method is Invoked?
- Abstract Methods and Classes
- Final
- Static
Visibility (Access Specifiers) for Variables and Methods

- **“friendly”** (default)
  - Visible to other classes in same package.
  - If not in a package, visible to other classes in same directory.

- **public**
  - Visible to all.

- **private**
  - Only visible within same class (same object, too).

- **protected**
  - Visible to same class and its subclasses.
Simple Inheritance Example

Art

String owner
private String Address

Drawing

double length
double width
Simple Inheritance Example
(too big to fit)

class Art {
    String owner; private String address;
}
class Drawing extends Art {
    double length, width;
}

public static void main(String[] args) {
    Drawing b = new Drawing("George W. Bush");
    Drawing c = new Drawing("Al Gore", 20.5, 31.768);
    // System.out.println("Owner = " + owner);
    System.out.println("Owner = " + c.owner);
    // System.out.println("Address = " + c.address);
}
More on Visibility

- If a class extends a class from another package, it can only access the public methods.

- A class can only be “friendly” (default) or public, not private or protected.

- Can only have one public class per compilation unit (file); must have same name as file.
Enhanced Inheritance Example

Art
- Person owner
- Date creation

Drawing
- Size dimensions

Cartoon
- Person penciller
- Person inker
- Person colorist

Sketch
- Person artist
Enhanced Inheritance Example

- Too big to fit on the screen.
- See Handout.
Enhanced Inheritance
Example Output

Art constructor
Drawing constructor
Cartoon constructor

No toString() defined for Cartoon!
Cartoon@72e62e3f
Owner of x = Person@71ce2e3f
Owner of x = Bill Gates
Art constructor
Drawing constructor
Cartoon constructor
Owner of y = Ed Parrish
Which Method Is Invoked?

Art
- Person owner
- Date creationDate
- getAge() : int
- getVolume() : int
- addValue()
- addValue(amount : int)

Sculpture
- Person sculptor
- dimensions : 3DSize
- getVolume()

Cartoon
- Person penciller
- Person inker
- Person colorist
- addValue()
- addValue(amount : int)

Drawing
- dimensions : Size
- addValue()

Sketch
- Person artist

getVolume() returns 0.
Which Method Is Invoked?

1) For an object of a particular type, Java looks in the class for a method which matches the signature (order and type of parameters).
2) If there is a match, Java is done.
3) If no match, try the parent class.
4) Repeat until you run out of classes.
Which Method Is Invoked?

start

set current class = class of object

does signature match method in current class?

Yes

success

No

set current class = parent class of current class

does current class have a parent class?

Yes

No

failure
Where to Put Methods

z Arrange classes to minimize your work.
  y If an operation is appropriate for all objects defined by a class, and all its subclasses, put the method in that class.
  y If an operation is appropriate for all objects defined by a class, and most of its subclasses, put the method in that class, and override in the subclasses as needed.
Where to Put Methods

- Arrange classes so there is always an “isA” relationship.
- If there should never be an object of this type, make this an abstract class.
The Abstract Keyword

You can declare a method with the keyword abstract and no body.

- Intention: Method will be overridden in a subclass.
  - No default method.

Like “pure virtual function” in C++.
A class with at least one abstract method must be declared as an abstract class. Compiler enforces this.

- Abstract classes cannot be instantiated.
- Any subclass of an abstract class must be an abstract class, until the method is overridden.
- You can still have a variable of that type.
Abstract Class Example

//: Abstract.java
public abstract class Abstract {
   Abstract() { System.out.println("Abstract constructor"); }  
   abstract void abMethod(); // No body allowed
   void regMethod() { System.out.println("regMethod"); }  
// Main program
public static void main(String[] args) {
   // Abstract x = new Abstract();
   Abstract x;
   Concrete y = new Concrete(); x = y;
   x.abMethod(); x.regMethod(); y.regMethod();
}
}
Abstract Class Example (cont.)

abstract class SubAbstract extends Abstract {
    SubAbstract() { System.out.println("SubAbstract constructor"); }
}
class Concrete extends SubAbstract {
    Concrete() { System.out.println("Concrete constructor"); }
    void abMethod() { System.out.println("abMethod in Concrete"); }
    void regMethod() { System.out.println("regMethod in Concrete"); }
}
Abstract Class
Example Output

Abstract constructor
SubAbstract constructor
Concrete constructor
abMethod in Concrete
regMethod in Concrete
regMethod in Concrete
Next Time

- Working with Classes and Methods:
  - Overloading
  - Overriding
  - Polymorphism
- Strings