Inheritance Revisited

- A subclass
  - Can add new members to those it inherits
  - Can override an inherited method of its superclass
    - A method in a subclass overrides a method in the superclass if the two methods have the same declarations

Inheritance Revisited

- Inheritance
  - Allows a class to derive the behavior and structure of an existing class

- Superclass or base class
  - A class from which another class is derived
- Subclass, derived class, or descendant class
  - A class that inherits the members of another class
- Benefits of inheritance
  - It enables the reuse of existing classes
  - It reduces the effort necessary to add features to an existing object

Figure 9-2
The subclass Ball inherits members of the superclass Sphere and overrides and adds methods
Inheritance Revisited

- A subclass inherits private members from the superclass, but cannot access them directly
- Methods of a subclass can call the superclass’s public methods
- Clients of a subclass can invoke the superclass’s public methods
- An overridden method
  - Instances of the subclass will use the new method
  - Instances of the superclass will use the original method

![Figure 9-3](image1.png)

An object invokes the correct version of a method

Java Access Modifiers

- Membership categories of a class
  - Public members can be used by anyone
  - Members declared without an access modifier (the default) are available to
    - Methods of the class
    - Methods of other classes in the same package
  - Private members can be used only by methods of the class
  - Protected members can be used only by
    - Methods of the class
    - Methods of other classes in the same package
    - Methods of the subclass

Is-a and Has-a Relationships

- Two basic kinds of relationships
  - Is-a relationship
  - Has-a relationship

![Figure 9-5](image2.png)

A ball is a sphere
Is-a Relationship

• Object type compatibility
  – An instance of a subclass can be used instead of an instance of the superclass, but not the other way around

Has-a Relationship

• Has-a relationship
  – Also called containment
  – Cannot be implemented using inheritance
    • Example: To implement the has-a relationship between a pen and a ball
      – Define a data field `point` whose type is `Ball` within the class `Pen`

Dynamic Binding and Abstract Classes

• A polymorphic method
  – A method that has multiple meanings
  – Created when a subclass overrides a method of the superclass
• Late binding or dynamic binding
  – The appropriate version of a polymorphic method is decided at execution time

Figure 9-6
A pen ‘has a’ or ‘contains a’ ball

Figure 9-7a
area is overridden: a) `mySphere.DisplayStatistics()` calls area in Sphere

Figure 9-7b
area is overridden: b) `myBall.displayStatistics()` calls area in Ball
Dynamic Binding and Abstract Classes

- Controlling whether a subclass can override a superclass method
  - Field modifier `final`
    - Prevents a method from being overridden by a subclass
  - Field modifier `abstract`
    - Requires the subclass to override the method
- Early binding or static binding
  - The appropriate version of a method is decided at compilation time
  - Used by methods that are `final` or `static`

Overloading methods
- To overload a method is to define another method with the same name but with a different set of parameters
- The arguments in each version of an overloaded method determine which version of the method will be used

Please open file `carrano_ppt09_B.ppt` to continue viewing chapter 9.