CSE 3302  
Programming Languages  
Smalltalk (cont.)  
Chengkai Li  
Fall 2007

Object(Instance), Class, Inheritance

Examples

"print it" and "inspect it"
- 3
- 3 class
- 3 class class
- 3 class class class
- 3 class class class class
- SmallInteger superclass

Instance variables and methods

- instance variables and methods of an object, which is an instance of the corresponding class:
  - defined in the class
  - click "instance" in system browser

- instance variables and methods of a class (thus the name class instance variables and class methods), which is an instance of the corresponding metaclass:
  - defined in the metaclass
  - click "class" in system browser

Class Variables

- Class Variables: Shared by all the instances of the class and the class itself (i.e., the instance of the metaclass)
Access

• All data members are private
  – Instance variables: only directly accessible to the instance itself (inside methods defined in the class)
  – Different instances have different copies of the instance variables
  – Class instance variables: only directly accessible to the class itself (inside class methods)
  – Each subclass has its own copies of the class instance variables
  – Class variables: directly accessible to all the instances of the class and the class itself.
  • The same copy shared by all instances and subclasses

• All methods are public
  – The private instance variables are accessible to outside through the methods.

For Accessing Private Data: Setter and Getter

Example:

```
class Complex
  instance variable real, imaginary
  
  getter
  Complex>>real
    ^real

  setter
  Complex>>real: aNumber
    real _ aNumber
```

Class Methods for Constructing New Instances

• Example:
  class Complex
class methods:

  Complex class>>real: aNumber1 imaginary: aNumber2
  | newComplex |
  newComplex _ super new.
  newComplex
  real: aNumber1;
  imaginary: aNumber2.
  ^ newComplex

  Complex class>>new
  ^ self real: 0 imaginary: 0
```

Class Methods for Accessing Class Variables

• Example:
  class Float
class variable: Pi E Epsilon …
class method:

  Float class>>pi
  ^Pi

  Compare:
  – method real is defined in Complex, so an instance of Complex(e.g., 3+2i) can receive message real
  – method pi is defined in Float class (instead of Float), so an instance of Float class (i.e., Float) can receive message pi

Example of Class Instance Variables

• superclass, subclasses
  – Number superclass
  – Number subclasses

Inheritance and Handling Messages

• Inheritance:
  – Smalltalk allows only single inheritance
  – trait is used for sharing methods among classes that don’t have inheritance relationship

• Method Lookup along the inheritance chains
  When an object receives a message:
  – If the class of the object has the method, use it;
  – Otherwise check the superclass, and the superclass of the superclass, and so on.

• Return value of a method
  – Message receiver if no explicit return (i.e., no ^)
Inheritance and Handling Messages

- **Overriding**
  - Multiple classes on inheritance chain may define the same method
  - Only the lowest one (starting from the receiver object) is used
  - Need to say "super methodName" if want to extend the method defined in some superclass (and this is a good practice)
    - E.g., initialize, new, ...

- **self and super**
  - Both `self` and `super` refer to the message receiver itself!
  - "self methodName" will start method looking-up from the class of the message receiver.
  - "super methodName" will start method looking-up from the class that defines the method which sends this message "super methodName".

Example

```
A>>m2
...
B subclass: #A ...
B>m2
super m2
C subclass: #B ...
C>m1
self m2
aC := C new.
aC m1
```

What will happen if the lookup of `m2` starts from the superclass of message receiver?

Abstract Method, Abstract Class

- **self subclassResponsibility**
- **Example**: Number>>+...