Access Modification

CS 1324

Deborah A. Trytten

Instance Data

- Instance data describes the state of an object
- When we declare instance data in a class, each object gets its own unique copy
- Example: Contact
  - Show memory diagram of two contact objects

Data Encapsulation

- A class should be a contract
  - If you follow the rules of the class, it should behave as described
- Instance data cannot be accessible to the outside world for this to be successful
  - How can you keep your contractual obligations if anyone and everyone can mess with your data?
  - What if we could access the characters in a String object?
- This is why classes like Point and Rectangle are so rare
  - They have public instance data

Example: Change Data

- What happens when you try to change data values in an object from the Contact class in another class?
- Consider the design of the Contact class. Is this a problem?
- What about the Shape class from Project 13?

Access Modifiers

- public
  - Can be accessed by any method in any class
  - UML: prepend “+” to public variables and methods
- private
  - Can be accessed only inside the class
  - UML: prepend “-” to private variables and methods
- If there isn’t one, package level access
  - Package means same directory (folder)
  - We’ve been using the default package
    - You may have noticed that eclipse (rightly) doesn’t approve

Data Access Guidelines

- Data should be private
  - Exception: constants
- Methods should be public
  - Exception: helper methods that are only used in implementing other methods
- Update Contact UML to reflect improved design
- Return to Contact code and encapsulate data
- What happens when we try to access data in the Contact class with these modifications?

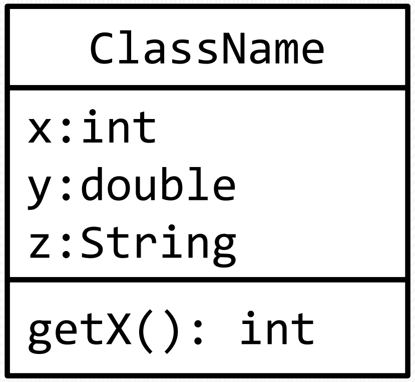
Example: ZooAnimal

- Design and implement a class to store zoo animal data:
  - Name
  - Species
  - Food type
  - Pounds of food per day
  - On display or not
- Encapsulate the data
- Should the class be mutable or immutable?

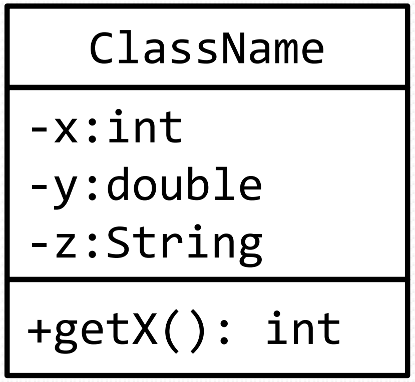
iClicker Question

Which of the following designs is typical for a class?

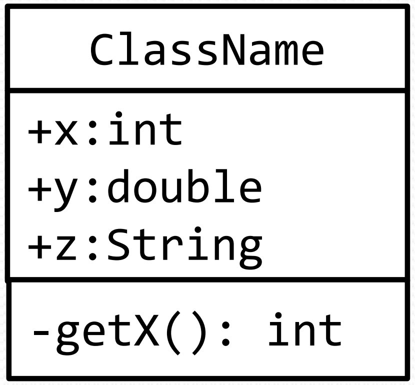
a)



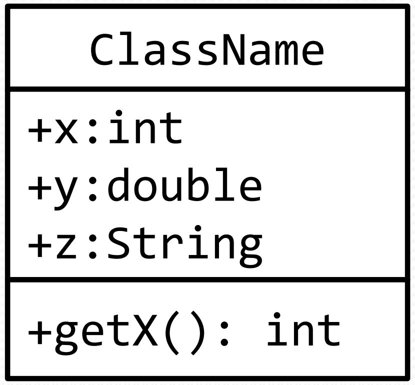
b)



c)



d)



Constants

- Constants should not be instance data
  - Why? Remember: every object has a copy of instance data
- Constants should be class data
  - Keyword "static" indicates class data
  - Keyword "final" indicates cannot change
  - Not created by a constructor
  - Accessed by prepending class name
  - UML: underline
- public or private?

iClicker Question

Suppose we are creating a class to store information about a homework assignment. The class contains the following information: title, due date, and priority (which can be low, regular, or high).

Which information should be class data?

a) The title
b) The due date
c) The priority
d) The values low, regular, and high

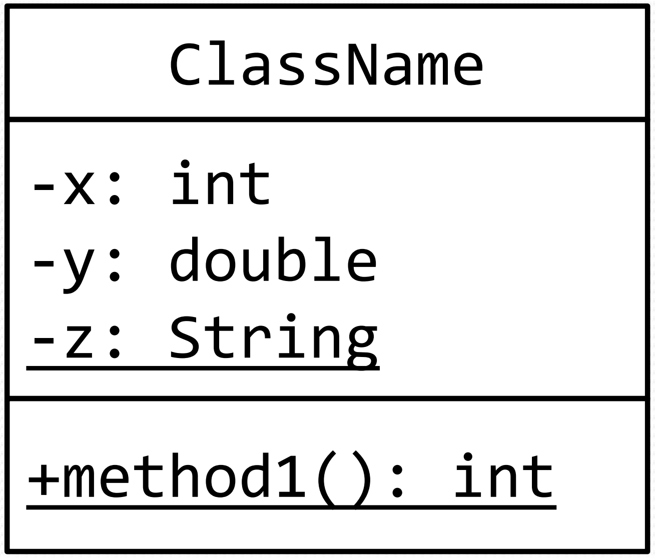
Class (Static) Methods

- Most of the methods we’ve written are class methods
  - Marked with static keyword
  - UML: underline (just like static fields)
- Class methods can access
  - Arguments (by using parameters)
  - Local variables
  - Class data (static—usually constants)
  - Other class methods
- Class methods cannot access instance data or instance methods from the class
  - Which instance?

iClicker Question

Consider the UML diagram below. Which data elements can method1 use?

a) x, y, and z
b) x and y
c) z
d) none



Instance (Non-Static) Methods

- Methods without static are instance methods
  - Instance is another name for object methods
- Instance methods can access both instance and class methods
- Instance methods can access both instance and class data
  - Every instance knows what class it is in
- Instance methods have an implicit (hidden) parameter
  - this
  - Used in constructor of ZooAnimal and Contact classes

Instance Methods: Implicit Parameter

```
Contact me = new Contact("James", "555-1234");
me.getName();
```

- The object reference me is an implicit argument to the method getName
- Argument is assigned to a parameter named “this”
- Tells Java which of the many objects from this class should be used
- Static methods do not have **this**
  - Without this they cannot find instance data

iClicker Question

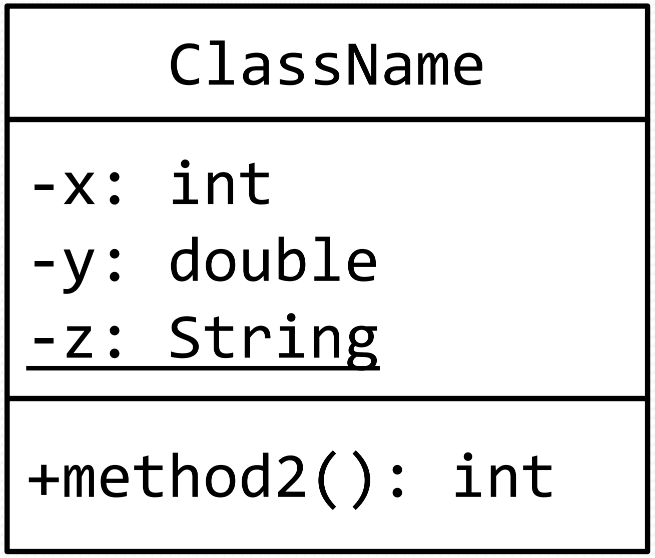
Consider the UML diagram below. Which data elements can method2 use?

a) x, y, and z

b) x and y

c) z

d) none



Guidelines: Class vs. Instance

- When should we use the static keyword?
- Guideline: Constants should be static
- Guideline: Methods that do not use instance data should be static
  - Observation: You can call static methods from objects but you shouldn’t
- Note: It’s common to have both static and non-static versions of methods
  - The static version usually has one more parameter
  - Example: Integer class toString()

Packages

- Groups classes together (another layer of abstraction!)
  - java.util is a package
  - Implemented as a directory
- We've been using the default package
  - Not a best practice
- To make a new package
  - `package packageName; // before class declaration`
- One purpose of packages: to prevent name conflicts
  - Check out Date class in API