Object Oriented Programming

Kookmin University

Department of Computer Science

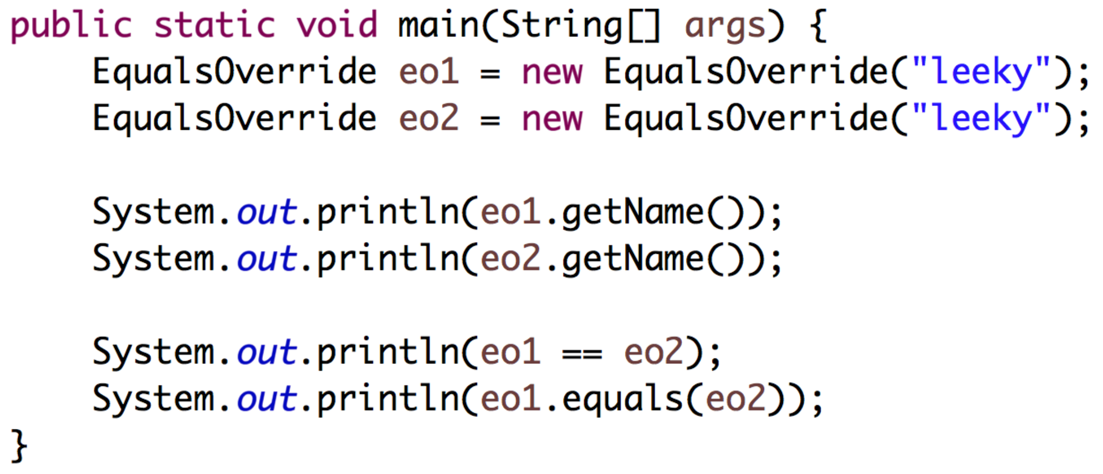
Announcements

- The Final homework
  - https://classroom.github.com/a/53MdECIG
  - Due: Nov. 29th

Complete Car and Tire Exercise

Overriding toString and equals

- By overriding toString, we can replace how an object is converted to a String.
- How about checking if two objects are the same
- Can you guess the outcome of the below example?



```java
public static void main(String[] args) {
    EqualsOverride eo1 = new EqualsOverride("leeky");
    EqualsOverride eo2 = new EqualsOverride("leeky");

    System.out.println(eo1.getName());
    System.out.println(eo2.getName());

    System.out.println(eo1 == eo2);
    System.out.println(eo1.equals(eo2));
}
```

Override equals method

- Method signature
  - public boolean equals(Object obj)
  - After getting an obj to compare from the argument, it needs to check if the type of obj is same
  - instanceof method checks the class hierarchy
  - child instanceof parent : true
  - parent instanceof child : false
- Question
  - How does String.equals works?

Abstraction in OOP

- Abstraction
  - A conceptual process that extract general rules and concepts from real and specific usages
  - In OOP, abstraction refers to hiding certain details and showing the common and essential of features – it deals with the outside view
  - The goal of abstraction in OOP is to generalize an object so that it can be widely reused – note parent and child class
  - EX – in abstracting Person
  - A person can be either a kid, student, boy/girl friend, part time employee, …
  - What is common features of the the above example to represent Person?

Abstraction Class in Java

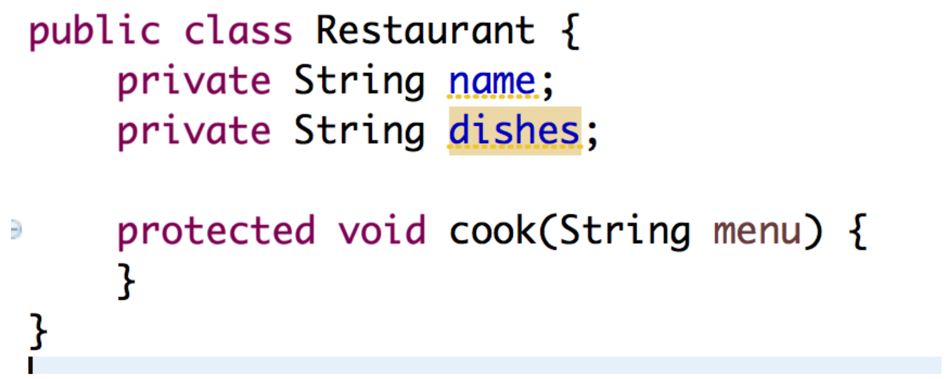
- A class that extracts commonalities among multiple actual classes that can create instances
- The abstraction class can be inherited to child class so that the child class can reuse features (methods/fields) of the abstract parent class

Wait… Inheritance Does the Job

- What abstract class does is actually achieved by the inheritance (parent ← child class)
  - Remember CellPhone and SmartPhone class
  - CellPhone class has all the necessary features for a SmartPhone class
- Let us consider an example
  - In building a Restaurant class, there is a method cook()
  - The body of cook() method is different across all the different restaurant (Chinese, Korean, Japanese, American, …)
  - In which way do we have to set the cook() method?

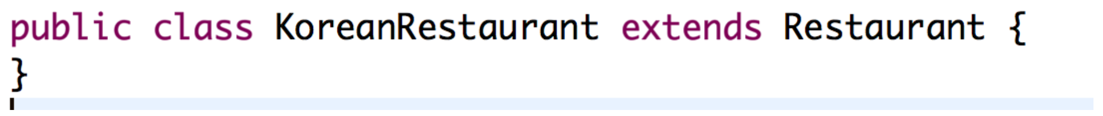
Why Using Abstract Class?

- In the right example, if cook() method should be different from all other restaurant, why don't we remove it from the Restaurant class?

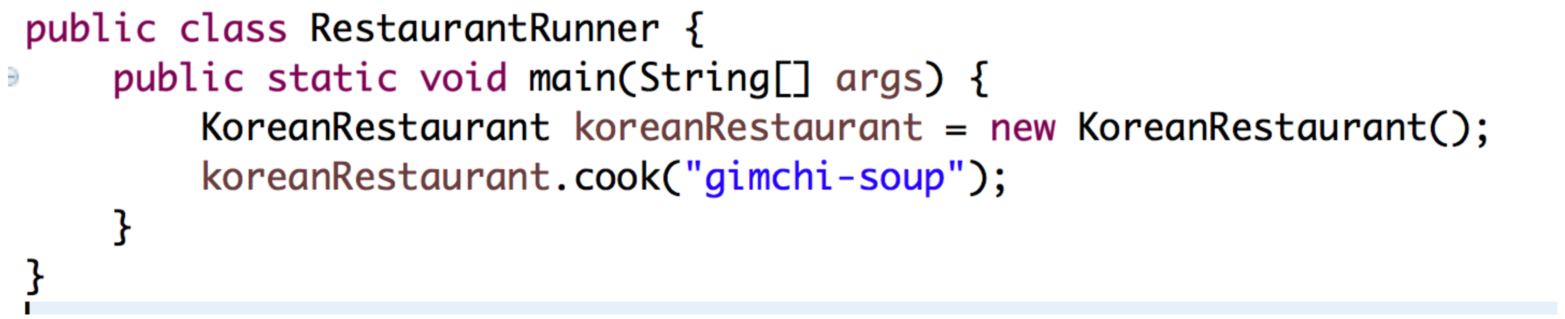


```java
public class Restaurant {
    private String name;
    private String dishes;

    protected void cook(String menu) {
    }
}
```



```java
public class KoreanRestaurant extends Restaurant {
}
```



```java
public class RestaurantRunner {
    public static void main(String[] args) {
        KoreanRestaurant koreanRestaurant = new KoreanRestaurant();
        koreanRestaurant.cook("gimchi-soup");
    }
}
```

Importance of Abstract Class

- We can define an empty body for a shared method (e.g., cook()), but it might cause a child class to reuse it without overriding the method. Abstract class enforces overriding a method in the compile time
- Why not removing a method that is not common?
  - Polymorphism allows to use parent class type to access common elements
  - In the Restaurant example, what if a class declares cook() method with a name makeDish()?
  - It guarantees the advantages of polymorphism and inheritance

Declaring Abstract Class

- Syntax
- ACCESS_MODIFIER abstract class CLASS_NAME
    - EX: public abstract class Restaurant

Abstract Method

- A method that has declaration without implementation
  - A child class should Override it (@Override annotation)
- Benefit of abstract method
  - Unifying the method name and arguments
  - Enforces implementation from a child class in the compile time
  - Restaurant cook() method
  - Syntax: ACCESS_MODIFIER abstract METHOD_DEFINITION;
  - EX: public abstract void cook();
    - No curly bracket for body declaration
  - Animal.java, Mammal.java, Dog.java, Cat.java

References

- 이것이자바다 – 한빛미디어 2015