Object Oriented Analysis and Design with Java

UE19CS353

Prof. Sindhu R Pai
Department of Computer Science and Engineering

Acknowledgements: Significant portions of the information in the slide sets presented through the course in the class, are extracted from the prescribed text books, information from the Internet. Since these are only intended for presentation for teaching within PESU, there was no explicit permission solicited. We would like to sincerely thank and acknowledge that the credit/rights remain with the original authors/creators only

UE19CS353: Object Oriented Analysis and Design with Java

Recap: Interface coding examples, Overloading Vs Overriding, Covariant return type – Example on clone()

Prof. Sindhu R Pai

Department of Computer Science and Engineering

Interface

- Coding examples on

  Interface can have default and static methods

  Interface achieves loose coupling

Overloading vs Overriding

Method Overloading	Method Overriding
The argument type needs to be different (at least the order).	The argument type needs to be the same (including the order).
Return type can be different or the same. But it is a must for a user to change the parameter.	Return type must be the very same until the 1.4 version of Java only. After that, it only allows the **Covariant return type** from Java 1.5 onwards.
Can use any access modifier. It can be same or different	The access modifier for a subclass method must be the same or higher than the access modifier of the superclass method.
Every method signature must be different (with the same name)	Every method signature must be the same (with the same name)
A user can generally perform method overloading within the same class.	A user can usually perform the method overriding in two of the classes through the Inheritance (considered an Is-A relationship).
A user can easily overload final/static/private methods	Not possible
A user can always take care of **method resolution with a Java compiler based on the reference type.**	A user can always take care **of method resolution with the JVM based on the runtime object.**
It is also known as the early binding, static polymorphism, or compile-time polymorphism.	It is also known as late binding, dynamic polymorphism/dispatch, or runtime polymorphism.
It may or may not be requiring inheritance.	It is always in need of inheritance.
The parameter needs to be different in the case of method overloading.	The parameter needs to be the same in the case of method overriding.

Covariant return type

- If the method of the superclass returns a superclass object, the method of the subclass overriding the method of the superclass can return a subclass object. This is called **co-variant return type.**

- clone() method

  Available in Object class which is protected.

  The client cannot use unless it is made public

  The class must implement a marker interface named cloneable

Points to think!!

- Can you extend abstract class from an interface?
- Can abstract class implement an interface?
- Can you write user defined covariant return type method?
- Is it possible to override co-variant return types?
- User defined Operator overloading is there in java?

References

- Difference Between Method Overloading and Method Overriding in Java (byjus.com)
- java - What is a covariant return type? - Stack Overflow
- Static method in Interface in Java - GeeksforGeeks

THANK YOU

Prof. Sindhu R Pai

Department of Computer Science and Engineering

sindhurpai@pes.edu

+91 8277606459