Object Oriented Programming
CS F213
Amit Dua

BITS Pilani
Pilani Campus

Slides Taken from the slides prepared by Dr. Jennifer

innovate achieve lead

BITS Pilani

Pilani Campus

Java Object Model

Questions from prev. class

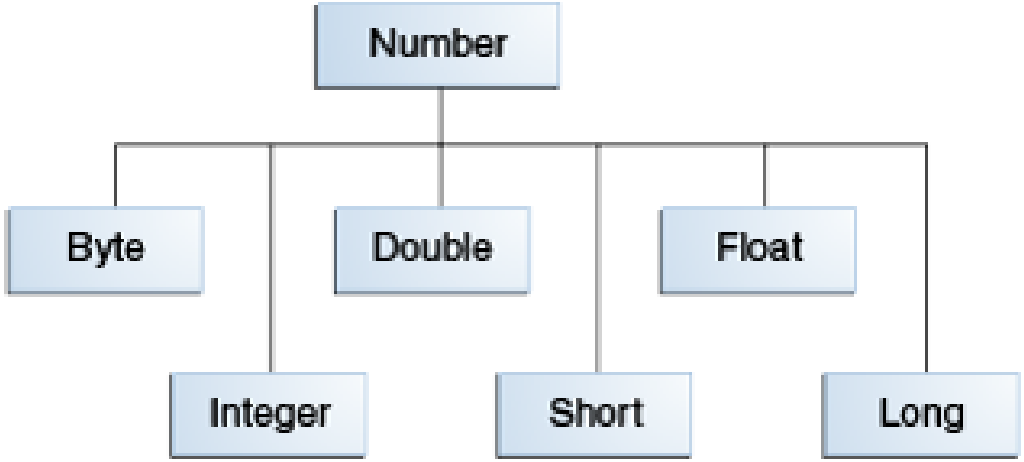
- Relationship between Integer , Double, Number and Object classes?
- Is Number subtype of Object?
- Is Integer and Double subtype of Number?
- Is Integer subtype of Double?

Explanation

https://docs.oracle.com/javase/tutorial/java/data/numberclasses.html

https://docs.oracle.com/javase/8/docs/api/?java/lang/Integer.html

https://docs.oracle.com/javase/9/docs/api/java/lang/Double.html



Array class

Why do we need separate Arrays class when we already have Object class?

Ans

Loops are used to perform tasks on collections

-Fill an array with a particular value.

-Sort an Arrays.

-Search in an Arrays.

Arrays class provides several static methods that can be used to perform these tasks directly without the use of loops.

https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html

String class

How does String class allow to create an object without the use of constructor?

Does it use constructor?

If yes how and if no how can an object be created without a constructor?

String str = “Java”;

String class

Strings are constant;

their values cannot be changed after they are created.

String buffers support mutable strings.

Because String objects are immutable they can be shared.

For example:

String str = "abc";

is equivalent to:

```
char data[] = {'a', 'b', 'c'};
String str = new String(data);
```

https://docs.oracle.com/javase/7/docs/api/java/lang/String.html

equals() in Object

- If equals() is not overidden, how does it actually compare the objects?
- Does equals() method in Object use hashcode to compare?

- Employee e1 = new Employee(“Akhil”);
- Employee e2 = new Employee(“Akhil”);

- e1.equals(e2);

equals()

In equals() definition, only references are compared.

If e1==e2, return true, otherwise false.

To override, guidelines are

1. Reflexive
2. Symmetric
3. Transitive
4. Consistent
5. x.equals(null) should be false on non null objects

hashcode()

Guideline:

Two objects that return true on equals() should return the same hashcode.

Expected:

override hashcode() when overriding equals()

getclass()

What does getClass() return?

We know that getClass().getName() returns the name of the class.

Class class

https://docs.oracle.com/javase/7/docs/api/java/lang/Class.html

getClass() returns a Class object that represents the runtime class of the object

Example: twoString.java

Outline

- Methods of object class
  - toString()
  - equals()
  - hashCode()
  - clone()
- Serialization
- Reflection

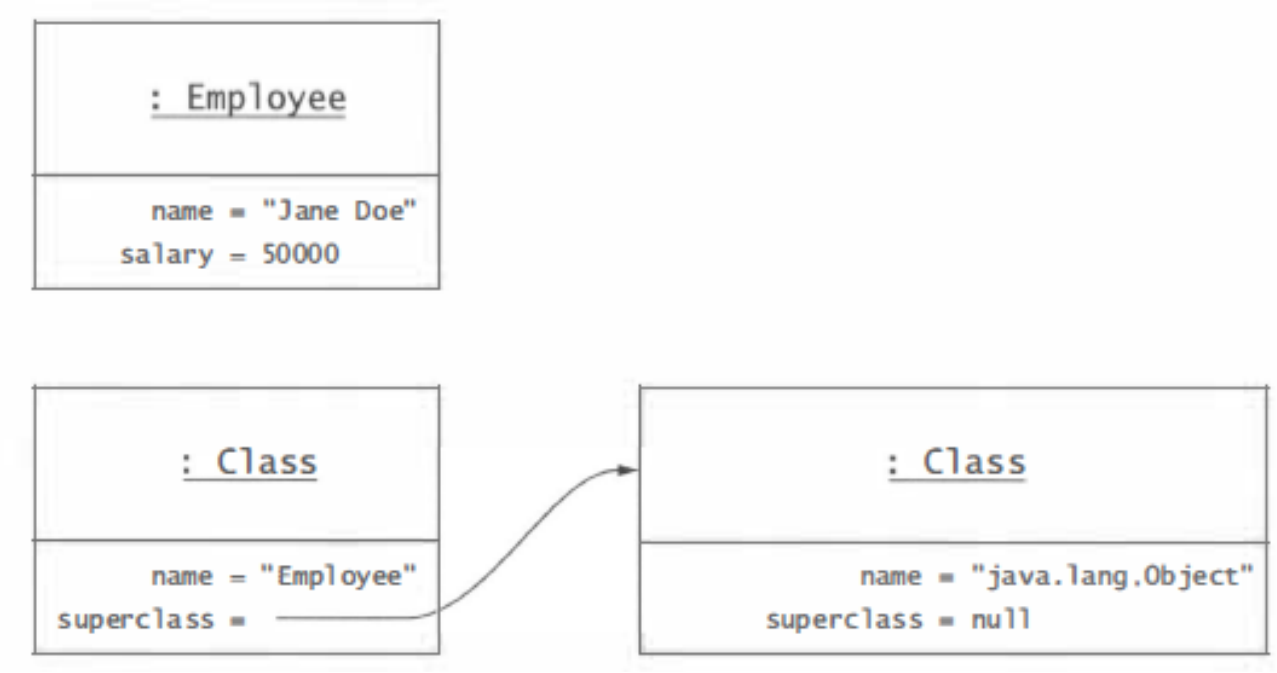
Methods in Object Class

- toString()
- hashCode()
  - For every object, JVM generates a unique number which is hashcode.
  - It is not the internal address of the object, but the hash based on the address
  - Advantage of saving objects based on hash code is searching becomes easy
- equals()
  - When hashCode() method is overridden a general contract is to be maintained which generates equal hash codes for equal objects.
- getClass()
  - Returns the actual runtime class of the object.

a.equals(b)

a==b

Class


: Employee
name = "Jane Doe"
salary = 50000
: Class
name = "Employee"
superclass =
: Class
name = "java.lang.Object"
superclass = null

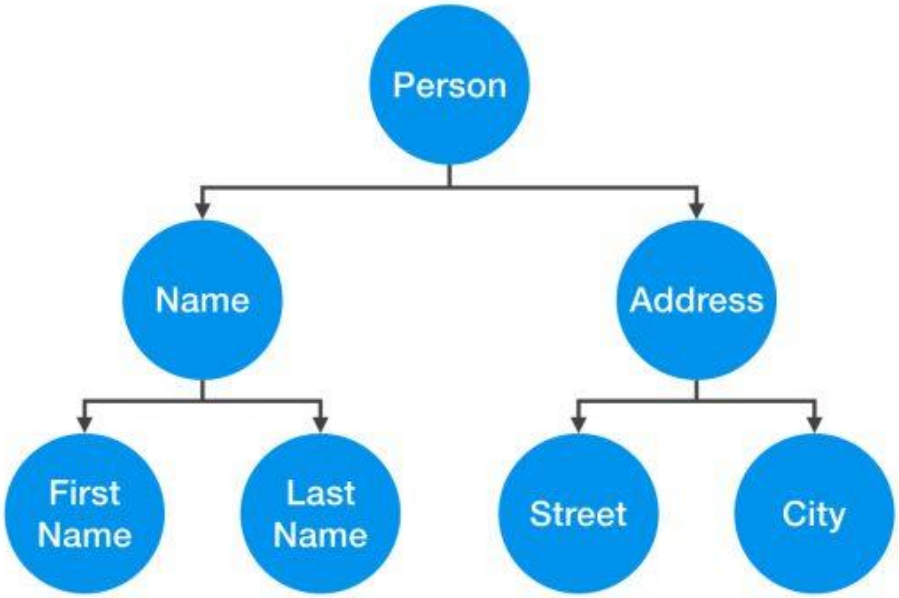
Example

- Equals.java

flaws?

- Equals2string.java

Objects – an Example



Person is made up of Name and Address objects which in turn is made up of objects like FirstName, LastName, Street and City respectively

Copying Objects

- When we use assignment operator it will create a copy of reference variable and not the object.

- Cloning refers to creation of exact copy of an object

- It creates a new instance of the class of current object and initializes all its fields with exactly same contents.

Cloning Condition

innovate achieve lead

x.clone() != x

x.clone().equals(x) return true

x.clone().getClass() == x.getClass()

“ clone should be a new object but it should be equals to its original”

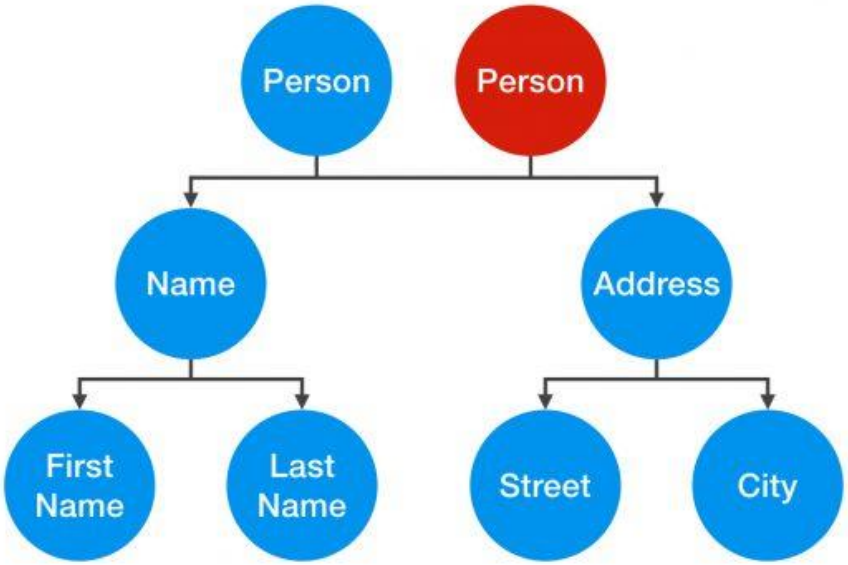
Clone requirements

Any class willing to be cloned must

1. **Declare the clone() method to be public**
2. **Implement Cloneable interface**

```
class Account implements Cloneable
{
public Object clone()
{
 try { super.clone() }
catch(CloneNotSupportedException e){ .. }
}
```

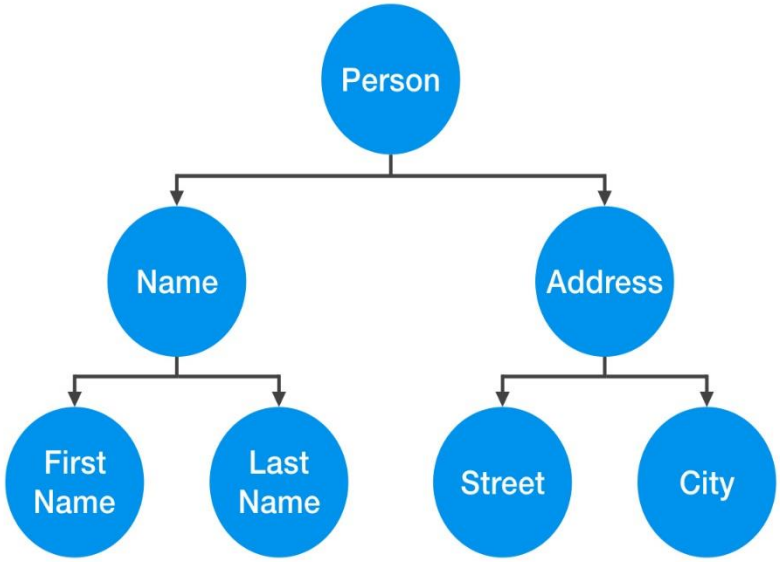
Shallow Copy

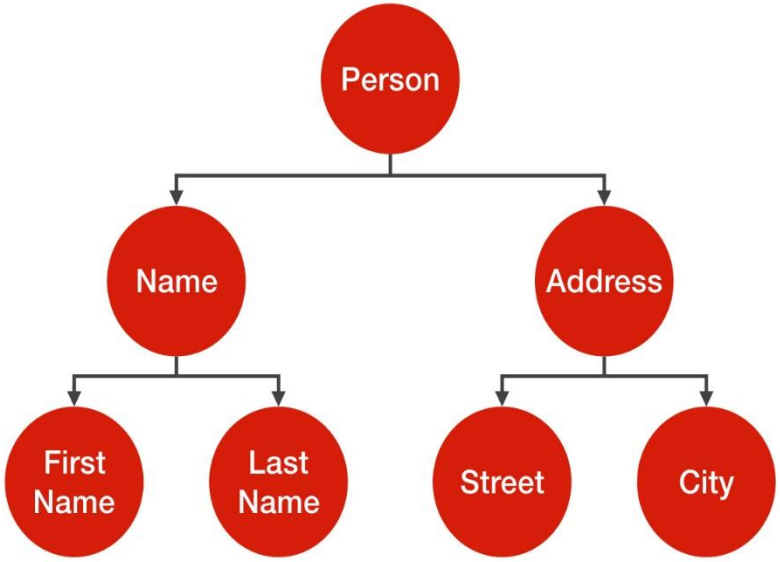


- It copies the main object but doesn't copy the inner objects.
- Inner objects are still shared between the original and its copy

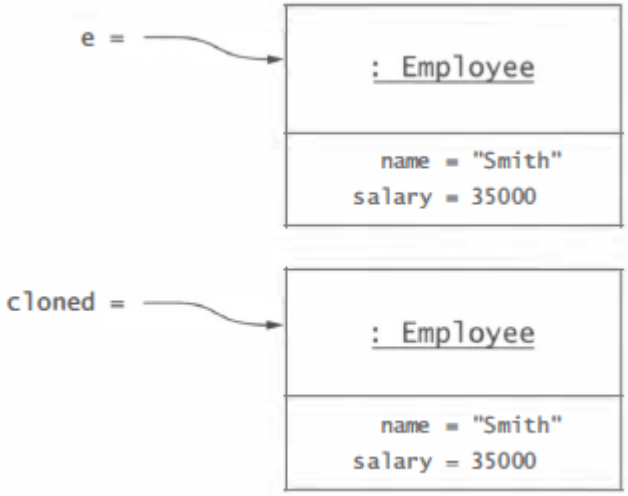
Deep Copy

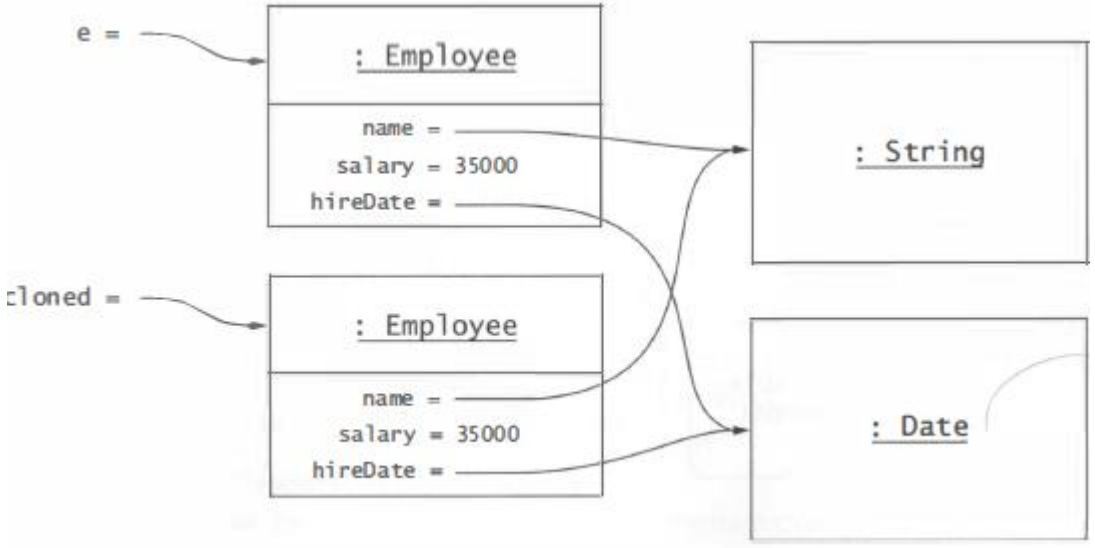
- It is a fully independent copy of an object and it copies the entire object structure



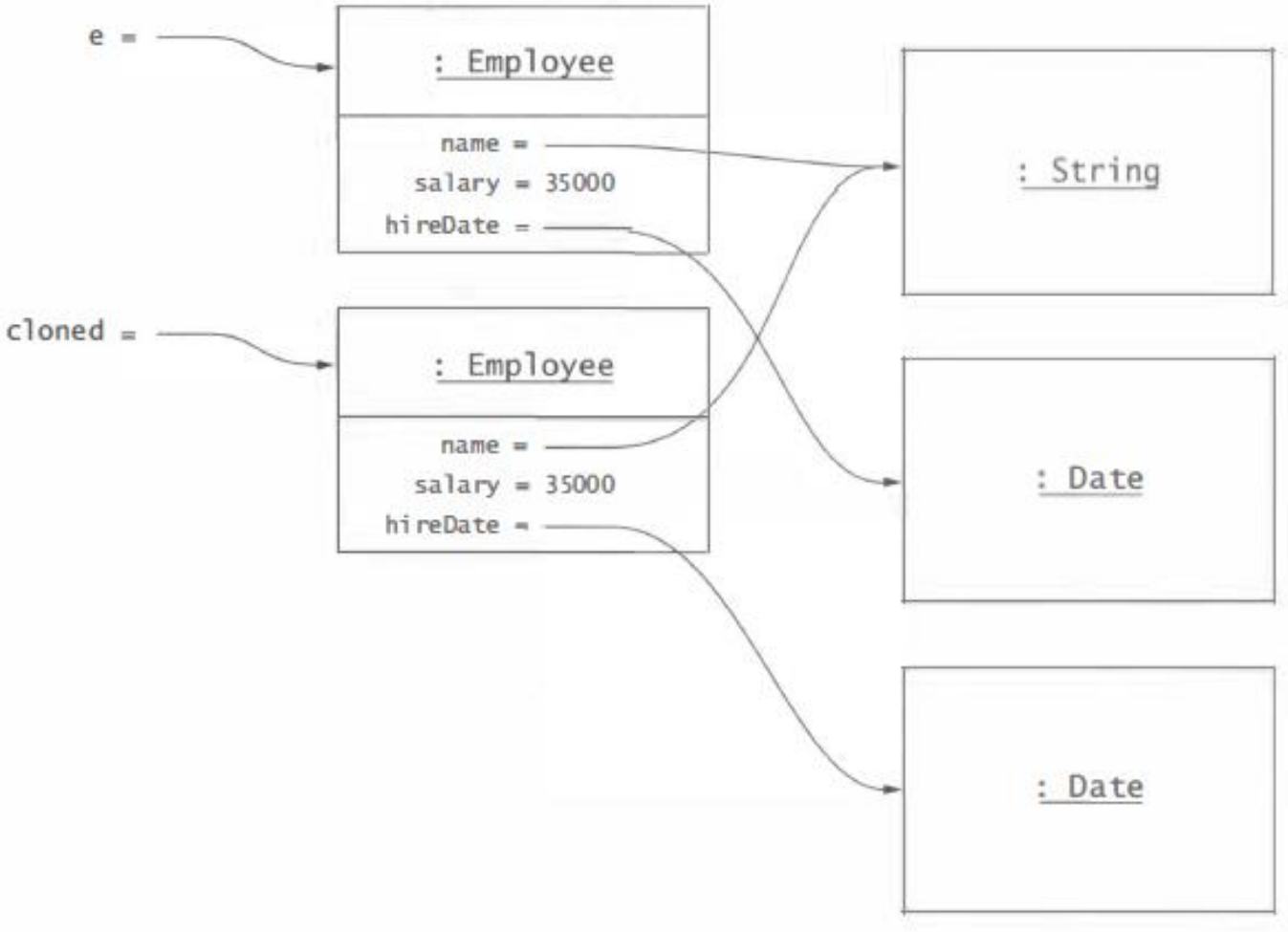


Shallow copy





Deep copy


e =
: Employee
name =
salary = 35000
hireDate =
cloned =
: Employee
name =
salary = 35000
hireDate =
: String
: Date
: Date

Example

- Clone.java

- By default clone() is a shallow copy in Object class
- Use super.clone() in overridden method.

Enum Type

- Used for representing a group of named constants in programming
- Enum in java is more powerful than C/C++
- In Java, we can add variables, methods and constructors to it.
- Enum can be declared outside the class or inside the class but not inside the method.

```
class Test{
enum Color{
 RED, GREEN, BLUE;
 }
 public static void main(String[] args) {
    }
}
```

Features of enum

- Enum is internally implemented using class

```
/* internally above enum Color is converted to
class Color {
public static final Color RED = new Color();
public static final Color BLUE = new Color();
public static final Color GREEN = new Color(); }*/
```

- Constants represents an object of type enum
- Constants are always implicitly public static final
  - It can be accessed using enum name
  - Child enums can not be created.
- It can be passed as an argument to switch statements

Features of enum

- All enums implicitly extend java.lang.Enum class
- toString() returns the enum constant name
- values() method can be used to return all values present inside enum
- ordinal() method is used to retrieve the constant index
- Enum can contain constructor and it is executed separately for each enum constant at the time of class loading.
- We cant create enum objects explicitly and hence we cannot invoke the enum constructor directly
- Enum can contain concrete method and not abstract methods.

Enum Example

Enumex.java

- finalize()
  - This method is called before garbage collection when an object has no more references.
  - It could be overridden to dispose system resources, perform clean up and minimize memory leaks.
  - finalize() method is called just once on an object
  - protected void finalize()
- gc()
  - It is used to invoke the garbage collector to perform clean up
  - It is found in System and Runtime classes.
  - public static void gc()

Java Runtime class

- It is used to interact with the Java runtime environment
- It provides methods to execute a process, invoke GC, get total and free memory etc.
- Only one instance of the java.lang.Runtime class is available for one Java application

Garbage Collector : gc()

GarbageCollector.java

Finalize()

- The *finalize()* method called by Garbage Collector not JVM. Although Garbage Collector is one of the module of JVM.
- Object class *finalize()* method has empty implementation, thus it is recommended to override *finalize()* method to dispose of system resources or to perform other cleanup.
- The *finalize()* method is never invoked more than once for any given object.
- If an uncaught exception is thrown by the *finalize()* method, the exception is ignored and finalization of that object terminates.