OOP using Java

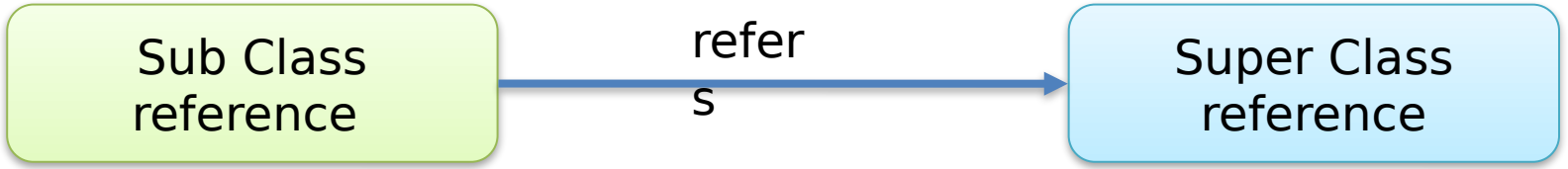
Trainer: Mr. Rohan Paramane

Agenda

- DownCasting
- Instanceof
- Object class
  - toString()
- Final
- Abstract

Downcasting

- Assigning parent class reference (which is pointing to child class object) to child class reference .
- Syntax for down casting : Child c = (Child)p;
- Here p is pointing to the object of child class as we saw earlier and now we cast this parent reference p to child class reference c.
- Now this child class reference c can access all the methods and variables of child class as well as parent class.



Subclass sub = (Subclass) s ;

Instanceof

- Downcasting is not always safe, as we explicitly write the class names before doing downcasting.
- It won’t give an error at compile time but it may throw ClassCastExcpetion at run time, if the parent class reference is not pointing to the appropriate child class.
- To remove ClassCastException we can use instanceof operator to check right type of class reference in case of down casting .

```
If (s instance of Subclass)  //(<Superclass_Ref> instanceof <subclass_className>)
{
Subclass sub = (Subclass) s;
}
```

Final

- In java we do not get const keyword. But we can use final keyword.
- After storing value, if we don’t want to modify it then we should declare variable final.
- We can provide value to the final variable either at compile time or run time.
- once initialized, if we don't want to modify state of any field inside any method of the class (including constructor body) then we should declare final field.
- If we want to declare any field final then we should declare it static also.
- In Java, we can declare reference final.
- But we can not declare instance final.
- We can declare method final. It is not allowed to override final method in sub class.
- We can declare class final. It is not allowed to extend final class.

Object class

- It is a non final and concrete class declared in java.lang package.
- In java all the classes(not interfaces)are directly or indirectly extended from java.lang.Object class.
- In other words, java.lang.Object class is ultimate base class/super
- cosmic base class/root of Java class hierarchy
- Object class do not extend any class or implement any interface.
- It doesn’t contain nested type as well as field.
- It contains default constructor.
  - Object o = new Object(“Hello”); // NOT OK
  - Object o = new Object( ); // OK
- Object class contains 11 methods.
- In the given code, java.lang.object is direct super class of class Person.
- For class Employee, class Person is direct super class and class object is indirect super class.

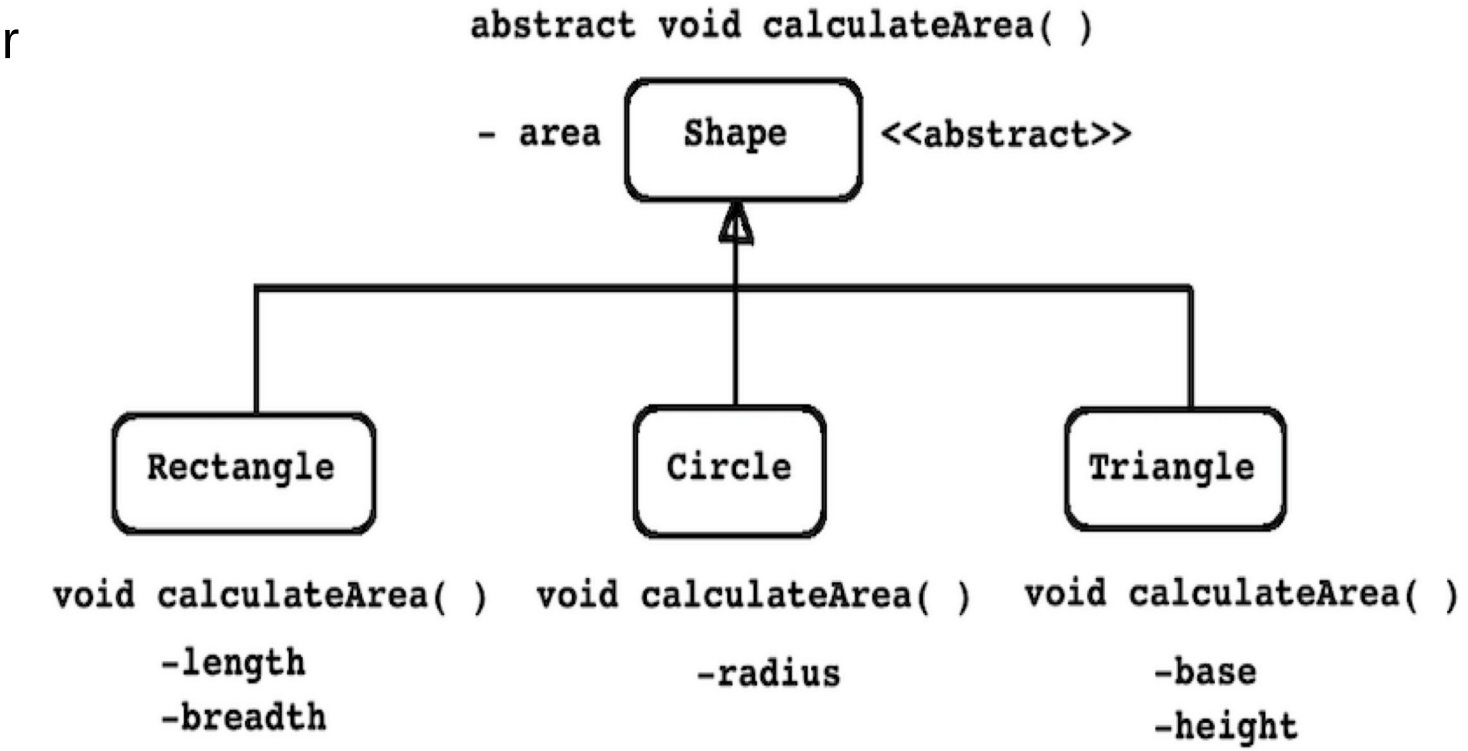
```
class Person{

}

class Employee extends Person{

}
```

toString( ) method

- It is a non final method of java.lang.Object class.
- Syntax:
  - public String toString( );
- If we want to return state of Java instance in String form then we should use toString() method.
- If we do not define toString() method inside class then super class's
- toString() method gets call.
- If we do not define toString() method inside any class then object class's
- toString() method gets call.
- It return String in the form: F.Q.ClassName@HashCode
  - Example : test.Employee@6d06d69c
- If we want state of instance then we should override toString() method inside class.
- The result in toString method should be a concise but informative that is easy for a person to read.
- It is recommended that all subclasses override this method.

Abstract Class

- If "is-a" relationship is exist between super type and sub type and if we want same method design in all the sub types then super type must be abstract.
- Using abstract class, we can group instances of related type together
- Abstract class can extend only one abstract/concrete class.
- We can define constructor inside abstract class.
- Abstract class may or may not contain abstract method.
- **Hint :** In case of inheritance if state is involved in super type then it should be abstract.




```
Shape[] arr = new Shape[ 3 ];
arr[ 0 ] = new Rectangle( ); //Upcasting
arr[ 1 ] = new Circle( ); //Upcasting
arr[ 2 ] = new Triangle( ); //Upcasting
```

Thank you!

Rohan Paramane

rohan.paramane@sunbeaminfo.com