Advanced Object Orientation - Part II

CSC02A2

Outline

Outline

Outline

Abstract Classes and Interfaces

Concept of an Abstract class

Abstract Classes in Java

Interfaces

Abstract classes vs Interfaces

Abstract classes and Interfaces Example

Abstract Classes and Interfaces

Concept of an Abstract class

Inheritance specialises classes further down the class hierarchy. Abstraction is the converse: the higher up the class hierarchy the more generalised classes become.

Once a superclass becomes generalised to a point where creating instances of the class is not meaningful such classes can be marked as abstract classes.

In the same line once a specific behaviours in an abstract class cannot be provided a meaningful definition and can also be marked as abstract methods.

Abstract Classes in Java

- Abstract methods cannot be static.
- Abstract methods cannot be contained in concrete classes.
- A class can be declared abstract without having any abstract methods.
- Abstract classes can have constructors but cannot be instantiated.
- A subclass may be abstract even if the parent class is concrete.
- References to abstract classes can be created.

Interface

Interfaces are a class-like construct that contains abstract methods. Implementation is not allowed[1].

Interfaces are used to define a set of shared behaviours which a group of otherwise unrelated class will share.

Unlike inheritance, a class can realise many interfaces in Java.

The `instanceOf` keyword can be used to test if an instance of an object realises a specific interface.

Note

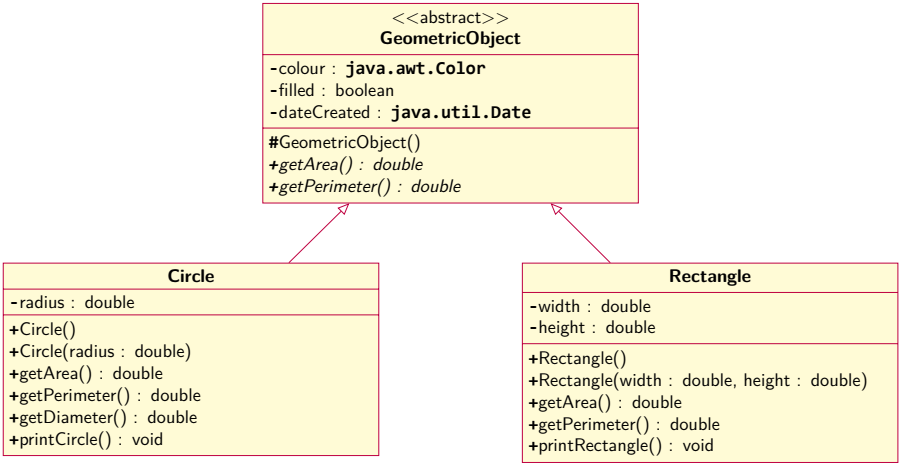
Since `Java SE 1.8` `interfaces` have received a few changes

[1] `default` methods are available but we will not be considering them.

Abstract classes vs Interfaces

Criteria	**Abstract class**	**Interface**
Instance methods	Allowed	Only `abstract` methods
Instance variables	Allowed	Not allowed
Usage	Extend from one class	Realise multiple interfaces
Relationship	***is-a***	***is-kind-of***

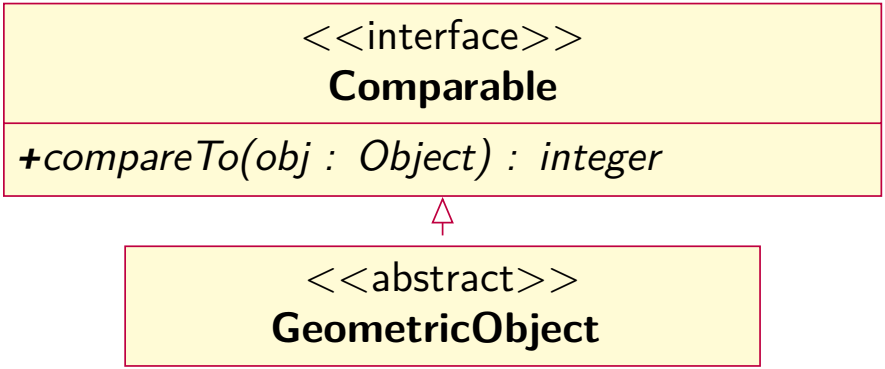
Abstract classes and Interfaces Example I



Abstract classes and Interfaces Example II

```java
import java.awt.Color;
import java.util.Date;
public abstract class GeometricObject
{
    private Color colour = Color.WHITE;
    private boolean filled = false;
    private Date dateCreated = new Date();
    // Accessor methods omitted for brevity
    // Mutator methods omitted for brevity
    public String toString()
    {
        return "Created on " + dateCreated +
               "\nColour " + colour +
               "\nFilled: " + filled;
    }
    public abstract double getArea();
    public abstract double getPerimeter();
}
```


Abstract classes and Interfaces Example III



Abstract classes and Interfaces Example IV

```java
public class GeometricObject implements Comparable
{
    // Old code omitted
    public int compareTo(Object other)
    {
      if(!(o instanceof GeometricObject))
      {
        return 1;
      }
      GeometricObject otherGO = ((GeometricObject)other);
      if(getArea() > otherGO.getArea())
      {
        return 1;
      }
        else if (getArea() < otherGO.getArea())
        {
          return -1;
        }
        return 0;
    }
}
```