Abstract class

Introduction

An *abstract class* is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

Example

Consider a class Figure to represent all 2d shapes like rectangle, triangle, circle etc.

So Figure can be superclass for all these and can have a method called area().

But it's not possible for Figure class to give general definition for calculating area for different shapes as area is dependent on particular shape.

Example

That's why we can not implement area() method in figure class. But still we want all figure classes should have area() method. So as to achieve this we can write area() method as abstract.

An abstract method is a method without body. That is you don't have to write the implementation.

Abstract method

- Abstract method only contain method header
- An abstract method is one to which a signature has been provided, but no implementation for that method is given.
- An Abstract method is a placeholder. It means that we declare that a method must exist, but there is no meaningful implementation for that methods within this class

Abstract method

An abstract method is declared by using a keyword called abstract.

Example

public abstract double area(double d1, double d2);

And class containing abstract method must be declared as abstract.

Abstract methods can be present in abstract class and interface.

Abstract class

- Abstract class is incomplete class and cannot be instantiated.
- Abstract class act as super class and all the subclasses must implement all abstract methods or must be declared as abstract.
- Abstract classes can contain both concrete and abstract methods.
- Abstract class may also not contain any abstract methods