**ABSTRACTION**

Abstraction is also one of the fundamental principles of object-oriented programming languages.In Object-oriented programming, abstraction is a process of hiding the implementation details from the user, only the functionality will be provided to the user. In other words, the user will have the information on what the object does instead of how it does it.

### **Real Life Example of Abstraction in Java:**

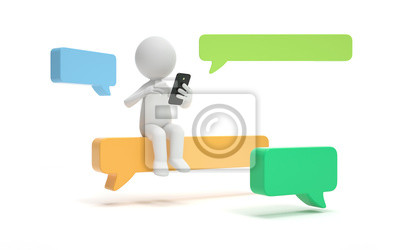
1. when we are driving a car, we are only concerned about driving the car like start/stop the car, accelerate/ brake, etc. We are not concerned about how the actual start/stop mechanism or accelerate/brake process works internally. We are just not interested in those details.



1. Another real life example of Abstraction is ATM Machine; All are performing operations on the ATM machine like cash withdrawal, money transfer, retrieve mini-statement…etc. but we can't know internal details about ATM.



1. When you need to send SMS from your mobile, you only type the text and send the message. But you don’t know the internal processing of the message delivery.



1. Another real world example of abstraction could be your TV remote. The remote has different functions like on/off, change channel, increase/decrease volume etc. You use these functionalities just by pressing the button. The internal mechanism of these functionalities are abstracted from you as those are not essential for you to know.



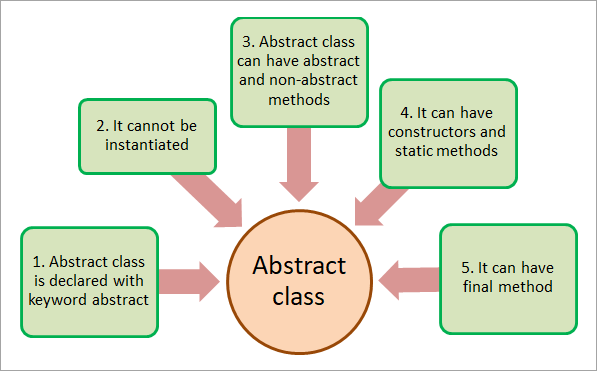
1. when we login to any social networking site like Facebook, Twitter, Linkedin etc, we enter our user id and password and then we get logged in. Here we don't know how they are processing the data or what logic or algorithm they are using for login. Those information are abstracted/hidden from us, since those are not essential for us. This is basically what abstraction is.

## 

## **How to achieve Abstraction in Java?**

There are two ways to achieve or implement abstraction in a Java program. They are as follows:

1. Abstract class (0 to 100%)
2. Interface (100%)
3. **Abstract class:** An abstract class in Java is a class, which is declared with an abstract keyword. It is just like a normal class but has two differences.
4. We cannot create an object of this class. Only objects of its non-abstract (or concrete) sub-classes can be created.
5. It can have zero or more abstract methods which are not allowed in a non-abstract class (concrete class). Classloader class is a good example of an abstract class that does not have any abstract methods.
6. Java Abstract class makes programming more flexible by providing scopes to write abstract methods in subclasses of the abstract class.



**Key points:** A class which contains the abstract keyword in its declaration is known as abstract class.

* Abstract classes may or may not contain *abstract methods*, i.e., methods without body ( public void get(); )
* But, if a class has at least one abstract method, then the class must be declared abstract.
* If a class is declared abstract, it cannot be instantiated.
* To use an abstract class, you have to inherit it from another class, provide implementations to the abstract methods in it.
* If you inherit an abstract class, you have to provide implementations to all the abstract methods in it.An abstract concept is not applicable to variables.
* It can have [constructors](https://www.javatpoint.com/java-constructor) and static methods also.
* It can have final methods which will force the subclass not to change the body of the method.

**Rules of Abstract class in Java:**

1. Class must be declared with an abstract keyword to make an abstract class.

2. We cannot instantiate an abstract class but we can create an object of subclass of the abstract class provided they must implement an abstract method.

3. If any method is abstract in a class, the class must be declared as abstract.

4. To use methods declared in an abstract class, the abstract class must be extended by an ordinary class and must implement (override) all abstract methods in that ordinary class.

5. If a new abstract method is added in the abstract class, all non-abstract subclasses which extend that abstract class, must implement the newly added abstract method. If it does not implement all the abstract methods, the class must be declared as abstract.

6. If a new instance method is added in the abstract class, all non-abstract subclasses which extend that abstract class are not necessary to implement the newly added instance method.

7. Inside the abstract class, we can create any number of constructors. If you do not create a constructor, the compiler will create a default constructor.

**When to use Abstract class in Java?**

An abstract class can be used when we need to share the same method to all non-abstract subclasses with their own specific implementations.Moreover, the common member of the abstract class can also be shared by the subclasses. Thus, abstract class is useful to make the program more flexible and understandable.

**Features of Abstract class in Java:**

1. Abstract class is not a pure abstraction in java.

2. In Java, object creation is not possible for an abstract class because it is a partially implemented class, not fully implemented class.

3. It can be abstract even with no abstract method.

4. It can have one or more abstract methods or non-abstract methods (or concrete methods) or a combination of both methods.

5. Abstract class allows defining private, final, static and concrete methods. Everything is possible to define in an abstract class as per application requirements.

6. It can have [constructors](https://www.scientecheasy.com/2020/06/constructor-in-java.html/).

7. Abstract class does not support multiple [inheritance](https://www.scientecheasy.com/2020/07/inheritance-in-java.html/) in java but allows in interfaces.

8. It can implement one or more interfaces in java.

## **Abstract Method in Java:**

A method that is declared with an abstract modifier in an abstract class and has no implementation (means nobody) is called an abstract **method in java**. It does not contain anybody.

Abstract method has simply a signature declaration followed by a semicolon. It has the following general form as given below: **public abstract void draw();**

If you want a class to contain a particular method but you want the actual implementation of that method to be determined by child classes, you can declare the method in the parent class as an abstract.

**When to use the Abstract method in Java?**

There are the following uses of abstract methods in Java. They are as follows:

1. An abstract method can be used when the same method has to perform different tasks depending on the object calling it.

2. A method can be used as abstract when you need to be overridden in its non-abstract subclasses.

Thus it becomes compulsory to override the abstract method in the subclass. If the abstract method is not implemented in the subclass as well, then we have to declare the subclass also as “abstract”.

## **Rules of Abstract method in Java**

1. Abstract methods can only be declared in an abstract class.

2. A non-abstract class cannot have an abstract method, whether it is inherited or declared in Java.

3. It must not provide a method body/implementation in the abstract class for which it is defined.

4. Method name and signature must be the same as in the abstract class.

5. The visibility of the method in the subclass cannot be reduced while overriding the abstract method.

6. Abstract method cannot be static or final.

7. It cannot be private because the abstract method must be implemented in the subclass. If we declare it private, we cannot implement it from outside the class.

## **Why does abstract class have constructor even though we cannot create objects?**

We cannot create an object of abstract class but we can create an object of subclass of abstract class. When we create an object of a subclass of an abstract class, it calls the constructor of the subclass.

This subclass constructor has super in the first line that calls the constructor of an abstract class. Thus, the constructors of an abstract class are used from the constructor of its subclass.

If the abstract class doesn’t have a constructor, a class that extends that abstract class will not get compiled.

### **Why should we create a reference to superclass (abstract class reference)?**

We should create a reference of the superclass to access subclass features because superclass reference allows only to access those features of subclass which have already been declared in superclass.

If you create an individual method in a subclass, the superclass reference cannot access that method. Thus, any programmer cannot add their own additional features in subclasses other than whatever is given in superclass.

**Advantage of Abstraction in java :**

* It **reduces the complexity** of viewing the things as it shows only essential information.
* It helps to **decouple** the behavior(method) and its implementation in a software/application.
* It allows the **flexibility** to modify/change implementation logic if needed, as long as the outcome of behavior is not affected by the modification.
* It helps to increase **security** of an application or program as only important details are provided to the user.
* It also helps to avoid **code duplication** and increases **reusability**.