



# Abstract Class in Java

[Read](#)[Discuss\(40+\)](#)[Practice](#)

An abstract class in Java is one that is declared with the abstract keyword. It may have both abstract and non-abstract methods(methods with bodies). An abstract is a java modifier applicable for classes and methods in java but *not for Variables*. In this article, we will learn the use of abstract class in java.

## What is Abstract class in Java?

Java abstract class is a class that can not be initiated by itself, it needs to be subclassed by another class to use its properties. An abstract class is declared using the “abstract” keyword in its class definition.

### Illustration of Abstract class

```
abstract class Shape
{
    int color;
    // An abstract function
    abstract void draw();
}
```

In Java, the following some *important observations* about abstract classes are as follows:

1. An instance of an abstract class can not be created.
2. Constructors are allowed.

Sale Ends In 04 : 14 : 18   [Java Arrays](#)   [Java Strings](#)   [Java OOPs](#)   [Java Collection](#)   [Java 8 Tutorial](#)   [Java M](#)

4. There can be a **final method** in abstract class but any abstract method in class(abstract class) can not be declared as final or in simpler terms final method can not be abstract itself as it will yield an error: “Illegal combination of modifiers: abstract and final”
5. We can define static methods in an abstract class
6. We can use the **abstract keyword** for declaring *top-level classes (Outer classes) as well as inner classes* as abstract



We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

**Got It !**

8. If the **Child class** is unable to provide implementation to all abstract methods of the **Parent class** then we should declare that **Child class as abstract** so that the next level Child class should provide implementation to the remaining abstract method

## Example of Abstract Class in Java

---

### Java

```
// Abstract class
abstract class Sunstar {
    abstract void printInfo();
}

// Abstraction performed using extends
class Employee extends Sunstar {
    void printInfo() {
        String name = "avinash";
        int age = 21;
        float salary = 222.2F;

        System.out.println(name);
        System.out.println(age);
        System.out.println(salary);
    }
}

// Base class
class Base {
    public static void main(String args[]) {
        Sunstar s = new Employee();
        s.printInfo();
    }
}
```

### Output

```
avinash
21
222.2
```

Let us elaborate on these observations and do justify them with help of clean java programs as follows.

## Properties of Abstract class

### Observation 1

In Java, just like in C++ an instance of an abstract class cannot be created, we can have references to abstract class type though. It is as shown below via the clean java program.

### Example

---

#### Java

```
// Java Program to Illustrate
// that an instance of Abstract
// Class can not be created

// Class 1
// Abstract class
abstract class Base {
    abstract void fun();
}

// Class 2
class Derived extends Base {
    void fun()
    {
        System.out.println("Derived fun() called");
    }
}

// Class 3
// Main class
class Main {

    // Main driver method
    public static void main(String args[])
    {

        // Uncommenting the following line will cause
        // compiler error as the line tries to create an
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

```
        Base b = new Derived();
        b.fun();
    }
}
```

## Output

Derived fun() called

## Observation 2

Like C++, an **abstract class** can contain **constructors** in Java. And a constructor of an abstract class is called when an instance of an inherited class is created. It is as shown in the program below as follows:

### Example:

---

## Java

```
// Java Program to Illustrate Abstract Class
// Can contain Constructors

// Class 1
// Abstract class
abstract class Base {

    // Constructor of class 1
    Base()
    {
        // Print statement
        System.out.println("Base Constructor Called");
    }

    // Abstract method inside class1
    abstract void fun();
}

// Class 2
class Derived extends Base {

    // Constructor of class2
    Derived()
    {
        System.out.println("Derived Constructor Called");
    }

    // Method of class2
    void fun()
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

```
}

// Class 3
// Main class
class GFG {

    // Main driver method
    public static void main(String args[])
    {
        // Creating object of class 2
        // inside main() method
        Derived d = new Derived();
        d.fun();
    }
}
```

## Output

```
Base Constructor Called
Derived Constructor Called
Derived fun() called
```

## Observation 3

In Java, we can have *an abstract class without any abstract method*. This allows us to *create classes that cannot be instantiated but can only be inherited*. It is as shown below as follows with help of a clean java program.

### Example:

---

## Java

```
// Java Program to illustrate Abstract class
// Without any abstract method

// Class 1
// An abstract class without any abstract method
abstract class Base {

    // Demo method. This is not an abstract method.
    void fun()
    {
        // Print message if class 1 function is called
        System.out.println("Function of Base class is called");
    }
}
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

```
}

// Class 3
class Main {

    // Main driver method
    public static void main(String args[])
    {
        // Creating object of class 2
        Derived d = new Derived();

        // Calling function defined in class 1 inside main()
        // with object of class 2 inside main() method
        d.fun();
    }
}
```

## Output

Function of Base class is called

## Observation 4

*Abstract classes can also have **final** methods* (methods that cannot be overridden)

### Example:

---

## Java

```
// Java Program to Illustrate Abstract classes
// Can also have Final Methods

// Class 1
// Abstract class
abstract class Base {

    final void fun()
    {
        System.out.println("Base fun() called");
    }
}

// Class 2
class Derived extends Base {

}

// Class 3
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

```
// Main driver method
public static void main(String args[])
{
    {
        // Creating object of abstract class

        Base b = new Derived();
        // Calling method on object created above
        // inside main method

        b.fun();
    }
}
}
```

## Output

Base fun() called

## Observation 5

For any abstract java class we are not allowed to create an object i.e., for an abstract class instantiation is not possible.

---

## Java

```
// Java Program to Illustrate Abstract Class

// Main class
// An abstract class
abstract class GFG {

    // Main driver method
    public static void main(String args[])
    {

        // Trying to create an object
        GFG gfg = new GFG();
    }
}
```

## Output:

```
mayanksolanki@MacBook-Air Desktop % javac GFG.java
GFG.java:11: error: GFG is abstract; cannot be instantiated
    GFG gfg = new GFG();
                  ^
1 error
mayanksolanki@MacBook-Air Desktop %
```

## Observation 6

Similar to the interface *we can define static methods in an abstract class that can be called independently without an object.*

## Java

```
// Java Program to Illustrate
// Static Methods in Abstract
// Class Can be called Independently

// Class 1
// Abstract class
abstract class Helper {

    // Abstract method
    static void demofun()
    {

        // Print statement
        System.out.println("Geeks for Geeks");
    }
}

// Class 2
// Main class extending Helper class
public class GFG extends Helper {

    // Main driver method
    public static void main(String[] args)
    {

        // Calling method inside main()
    }
}
```

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)



```
}
```

## Output

Geeks for Geeks

## Observation 7

We can use the **abstract keyword** for declaring top-level classes (Outer class) as well as inner classes as abstract

---

## Java

```
import java.io.*;

abstract class B {
    // declaring inner class as abstract with abstract
    // method
    abstract class C {
        abstract void myAbstractMethod();
    }
}

class D extends B {
    class E extends C {
        // implementing the abstract method
        void myAbstractMethod()
        {
            System.out.println(
                "Inside abstract method implementation");
        }
    }
}

public class Main {

    public static void main(String args[])
    {
        // Instantiating the outer class
        D outer = new D();

        // Instantiating the inner class
        D.E inner = outer.new E();
        inner.myAbstractMethod();
    }
}
```

## Output

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

## Inside abstract method implementation

### Observation 8

If a **class contains at least one abstract method** then **compulsory that we should declare the class as abstract** otherwise we will get a compile-time error ,If a class contains at least one abstract method then, implementation is not complete for that class, and hence it is not recommended to create an object so in order to restrict object creation for such partial classes we use **abstract keyword**.

---

### Java

```
/*package whatever //do not write package name here */

import java.io.*;

// here if we remove the abstract
// keyword then we will get compile
// time error due to abstract method
abstract class Demo {
    abstract void m1();
}

class Child extends Demo {
    public void m1()
    {
        System.out.print("Hello");
    }
}

class GFG {
    public static void main(String[] args)
    {
        Child c = new Child();
        c.m1();
    }
}
```

### Output

Hello

### Observation 9

If the **Child** class is unable to provide implementation to all abstract methods of the

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

Child class should provide implementation to the remaining abstract method.

---

## Java

```
// Java Program to demonstrate
// Observation
import java.io.*;

abstract class Demo {
    abstract void m1();
    abstract void m2();
    abstract void m3();
}

abstract class FirstChild extends Demo {
    public void m1() {
        System.out.println("Inside m1");
    }
}

class SecondChild extends FirstChild {
    public void m2() {
        System.out.println("Inside m2");
    }
    public void m3() {
        System.out.println("Inside m3");
    }
}

class GFG {
    public static void main(String[] args)
    {
        // if we remove the abstract keyword from FirstChild
        // Class and uncommented below obj creation for
        // FirstChild then it will throw
        // compile time error as didn't override all the
        // abstract methods

        // FirstChild f=new FirstChild();
        // f.m1();

        SecondChild s = new SecondChild();
        s.m1();
        s.m2();
        s.m3();
    }
}
```

## Output

Inside m1

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

Inside m3

In C++, if a class has at least one [pure virtual function](#), then the class becomes abstract. Unlike C++, in Java, a separate keyword `abstract` is used to make a class abstract.

## Conclusion

Points to remember from this article are mentioned below:

- An abstract class is a class that can not be initiated by itself, it needs to be subclassed by another class to use its properties.
- An abstract class can be created using “`abstract`” keywords.
- We can have an abstract class without any abstract method.

## FAQs of Abstract class

### 1. What is an abstract class in Java?

An abstract class in Java is a class that can not be initiated on its own but can be used as a subclass by another class.

### 2. What is the abstract class purpose?

The main purpose of the abstract class is to create a base class from which many other classes can be derived.

### 3. What is the main advantage of abstract class?

An abstract class provides the provides of data hiding in Java.

### 4. Why abstract class is faster than interface?

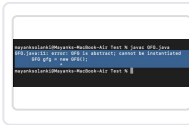
An abstract class is faster than an interface because the interface involves a search before calling any overridden method in Java whereas abstract class can be directly used.

## Must-Read Topics:


- [Difference between Abstract class and Interface in Java](#)
- [Difference between Abstract class and Abstract Methods](#)
- [Constructors in Java Abstract Class](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

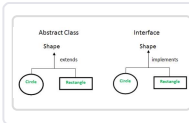
## Similar Reads




Difference Between Abstract Class and Abstract Method in Java




Difference between Abstract Class and Concrete Class in Java




Difference between Abstract Class and Interface in Java




Java | Abstract Class and Interface | Question 1




Java | Abstract Class and Interface | Question 2




Java | Abstract Class and Interface | Question 3




Can We Instantiate an Abstract Class in Java?



Constructor in Java Abstract Class

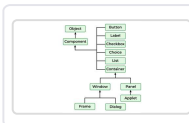


Implement Interface using Abstract Class in Java

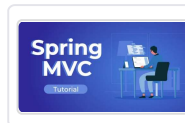


abstract keyword in java

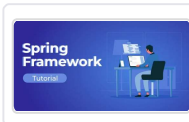
## Related Tutorials



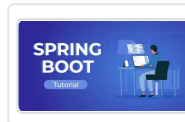
Java AWT Tutorial



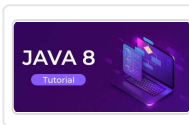
Spring MVC Tutorial



Spring Tutorial



Spring Boot Tutorial



Java 8 Features - Complete Tutorial

[Previous](#)

[Next](#)

## Article Contributed By :



GeeksforGeeks

## Vote for difficulty

Current difficulty : [Easy](#)



We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

**Improved By :** [RAJAT BHATTA 1](#), [gauravmoney26](#), [rocknaga81](#), [solankimayank](#), [adityakumar129](#), [sagartomar9927](#), [imkunal0306](#), [kashishsoda](#), [paritoshchaudhari](#), [ankitshivaneas](#), [worldofsimpleprogrammers](#), [kamleshjoshi18](#), [ankur5oz5](#), [laxmishinde5t82](#), [dhanu7207](#)

**Article Tags :** [Java-Abstract Class and Interface](#), [Java-Object Oriented](#), [Java](#)

**Practice Tags :** [Java](#)

[Improve Article](#)[Report Issue](#)**GeeksforGeeks**

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305

[feedback@geeksforgeeks.org](mailto:feedback@geeksforgeeks.org)



## Company

[About Us](#)[Legal](#)[Careers](#)[In Media](#)[Contact Us](#)

## Explore

[Job-A-Thon Hiring Challenge](#)[Hack-A-Thon](#)[GfG Weekly Contest](#)[Offline Classes \(Delhi/NCR\)](#)[DSA in JAVA/C++](#)

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

## Languages

Python  
Java  
C++  
PHP  
GoLang  
SQL  
R Language  
Android Tutorial

## DSA Concepts

Data Structures  
Arrays  
Strings  
Linked List  
Algorithms  
Searching  
Sorting  
Mathematical  
Dynamic Programming

## DSA Roadmaps

DSA for Beginners  
Basic DSA Coding Problems  
Complete Roadmap To Learn DSA  
DSA for FrontEnd Developers  
DSA with JavaScript  
Top 100 DSA Interview Problems  
All Cheat Sheets  
DSA Roadmap by Sandeep Jain

## Web Development

HTML  
CSS  
JavaScript  
Bootstrap  
ReactJS  
AngularJS  
NodeJS  
Express.js  
Lodash

## Computer Science

GATE CS Notes  
Operating Systems  
Computer Network  
Database Management System  
Software Engineering  
Digital Logic Design  
Engineering Maths

## Python

Python Programming Examples  
Django Tutorial  
Python Projects  
Python Tkinter  
OpenCV Python Tutorial  
Python Interview Question

## Data Science & ML

Data Science With Python  
Data Science For Beginner  
Machine Learning Tutorial

## DevOps

Git  
AWS  
Docker

We use cookies to ensure you have the best browsing experience on our website. By using our site, you acknowledge that you have read and understood our [Cookie Policy](#) & [Privacy Policy](#)

[Pandas Tutorial](#)[Azure](#)[NumPy Tutorial](#)[GCP](#)[NLP Tutorial](#)[Deep Learning Tutorial](#)

## Competitive Programming

[Top DSA for CP](#)[Top 50 Tree Problems](#)[Top 50 Graph Problems](#)[Top 50 Array Problems](#)[Top 50 String Problems](#)[Top 50 DP Problems](#)[Top 15 Websites for CP](#)

## Interview Corner

[Company Wise Preparation](#)[Preparation for SDE](#)[Experienced Interviews](#)[Internship Interviews](#)[Competitive Programming](#)[Aptitude Preparation](#)

## Commerce

[Accountancy](#)[Business Studies](#)[Economics](#)[Management](#)[Income Tax](#)[Finance](#)[Statistics for Economics](#)

## SSC/ BANKING

[SSC CGL Syllabus](#)[SBI PO Syllabus](#)[SBI Clerk Syllabus](#)[IBPS PO Syllabus](#)

## System Design

[What is System Design](#)[Monolithic and Distributed SD](#)[Scalability in SD](#)[Databases in SD](#)[High Level Design or HLD](#)[Low Level Design or LLD](#)[Top SD Interview Questions](#)

## GfG School

[CBSE Notes for Class 8](#)[CBSE Notes for Class 9](#)[CBSE Notes for Class 10](#)[CBSE Notes for Class 11](#)[CBSE Notes for Class 12](#)[English Grammar](#)

## UPSC

[Polity Notes](#)[Geography Notes](#)[History Notes](#)[Science and Technology Notes](#)[Economics Notes](#)[Important Topics in Ethics](#)[UPSC Previous Year Papers](#)

## Write & Earn

[Write an Article](#)[Improve an Article](#)[Pick Topics to Write](#)[Write Interview Experience](#)



Aptitude Questions

SSC CGL Practice Papers

@geeksforgeeks , Some rights reserved