

Difference between abstract class and interface

Abstract class and interface both are used to achieve abstraction where we can declare the abstract methods. Abstract class and interface both can't be instantiated.

But there are many differences between abstract class and interface that are given below.

Abstract class	Interface
1) Abstract class can have abstract and non-abstract methods.	Interface can have only abstract methods. Since Java 8, it can have default and static methods also.
2) Abstract class doesn't support multiple inheritance.	Interface supports multiple inheritance.
3) Abstract class can have final, non-final, static and non-static variables.	Interface has only static and final variables.
4) Abstract class can provide the implementation of interface.	Interface can't provide the implementation of abstract class.
5) The abstract keyword is used to declare abstract class.	The interface keyword is used to declare interface.
6) An abstract class can extend another Java class and implement multiple Java interfaces.	An interface can extend another Java interface only.
7) An abstract class can be extended using keyword "extends".	An interface can be implemented using keyword "implements".
8) A Java abstract class can have class members like private, protected, etc.	Members of a Java interface are public by default.
9) Example: public abstract class Shape{ public abstract void draw(); }	Example: public interface Drawable{ void draw(); }

Simply, abstract class achieves partial abstraction (0 to 100%) whereas interface achieves fully abstraction (100%).

Example of abstract class and interface in Java

Let's see a simple example where we are using interface and abstract class both.

```
//Creating interface that has 4 methods
```

```
interface A{  
void a();//bydefault, public and abstract  
void b();  
void c();  
void d();  
}
```

```
//Creating abstract class that provides the implementation of one method of A interface
```

```
abstract class B implements A{  
public void c(){System.out.println("I am C");}  
}
```

```
//Creating subclass of abstract class, now we need to provide the implementation of rest
```

```
class M extends B{  
public void a(){System.out.println("I am a");}  
public void b(){System.out.println("I am b");}  
public void d(){System.out.println("I am d");}  
}
```

```
//Creating a test class that calls the methods of A interface
```

```
class Test5{  
public static void main(String args[]){  
    A a=new M();  
    a.a();  
    a.b();  
    a.c();  
    a.d();  
}
```

}}

 [Test it Now](#)

Output:

```
I am a
I am b
I am c
I am d
```

[← Prev](#)

[Next →](#)

 [For Videos Join Our Youtube Channel: Join Now](#)


Feedback


- Send your Feedback to feedback@javatpoint.com

Help Others, Please Share





Learn Latest Tutorials


 [Splunk tutorial](#)
Splunk


 [SPSS tutorial](#)
SPSS


 [Swagger tutorial](#)
Swagger

 [T-SQL tutorial](#)
Transact-SQL

 [Tumblr tutorial](#)
Tumblr

 [React tutorial](#)
ReactJS

 [Regex tutorial](#)
Regex

 [Reinforcement learning tutorial](#)