



Types of References in Java

Difficulty Level : Medium • Last Updated : 06 Dec, 2019

[Read](#)[Discuss](#)[Practice](#)[Video](#)[Courses](#)

In Java there are four types of references differentiated on the way by which they are garbage collected.

1. Strong References
2. Weak References
3. Soft References
4. Phantom References

Prerequisite: [Garbage Collection](#)

- **Strong References:** This is the default type/class of Reference Object. Any object which has an active strong reference are not eligible for garbage collection. The object is garbage collected only when the variable which was strongly referenced points to null.

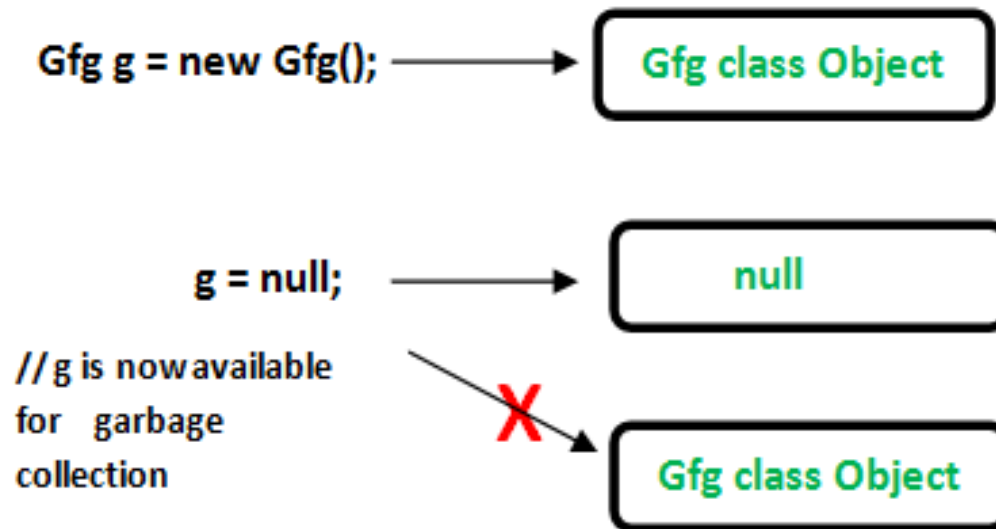
```
MyClass obj = new MyClass ();
```



Start Your Coding Journey Now!

[Login](#)[Register](#)

```
obj = null;  
// 'obj' object is no longer referencing to the instance.  
So the 'MyClass' type object is now available for garbage collection.
```



AD



```
// Java program to illustrate Strong reference  
class Gfg
```

Start Your Coding Journey Now!

[Login](#)[Register](#)

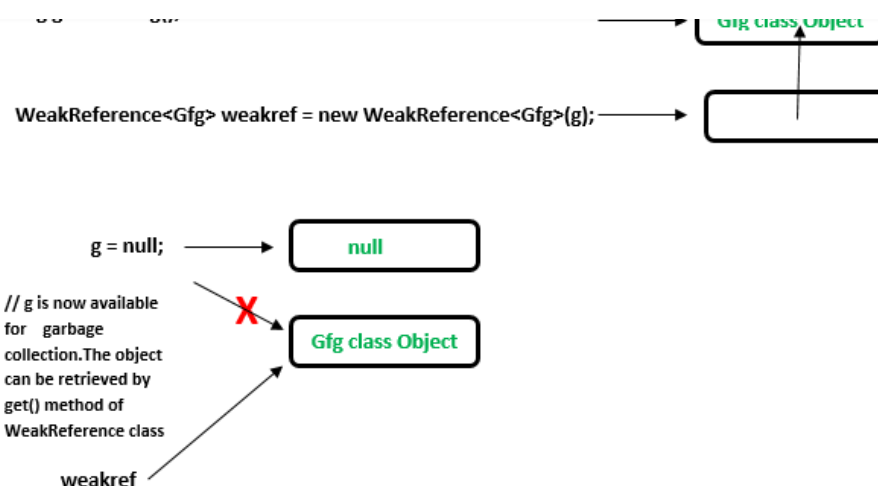
```
public class Example
{
    public static void main(String[] args)
    {
        //Strong Reference - by default
        Gfg g = new Gfg();

        //Now, object to which 'g' was pointing earlier is
        //eligible for garbage collection.
        g = null;
    }
}
```

- **Weak References:** Weak Reference Objects are not the default type/class of Reference Object and they should be explicitly specified while using them.
 - This type of reference is used in WeakHashMap to reference the entry objects .
 - If JVM detects an object with only weak references (i.e. no strong or soft references linked to any object), this object will be marked for garbage collection.
 - To create such references [java.lang.ref.WeakReference](#) class is used.
 - These references are used in real time applications while establishing a DBConnection which might be cleaned up by Garbage Collector when the application using the database gets closed.



Start Your Coding Journey Now!

[Login](#)
[Register](#)


```
//Java Code to illustrate Weak reference
import java.lang.ref.WeakReference;
class Gfg
{
    //code
    public void x()
    {
        System.out.println("GeeksforGeeks");
    }
}

public class Example
{
    public static void main(String[] args)
    {
        // Strong Reference
        Gfg g = new Gfg();
        g.x();
    }
}
```

Start Your Coding Journey Now!

[Login](#)[Register](#)

```
//Now, Gfg-type object to which 'g' was pointing earlier
//is available for garbage collection.
//But, it will be garbage collected only when JVM needs memory.
g = null;

// You can retrieve back the object which
// has been weakly referenced.
// It successfully calls the method.
g = weakref.get();

g.x();
}
}
```

Two different levels of weakness can be enlisted: Soft and Phantom

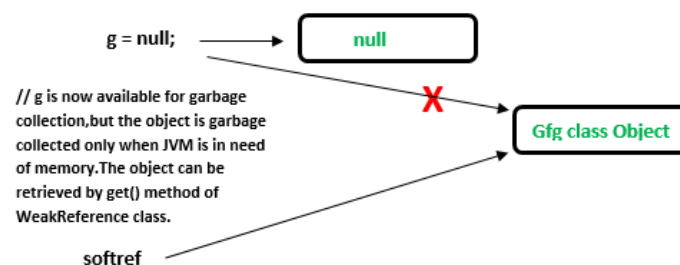
- **Soft References:** In Soft reference, even if the object is free for garbage collection then also its not garbage collected, until JVM is in need of memory badly. The objects gets cleared from the memory when JVM runs out of memory. To create such references [java.lang.ref.SoftReference](#) class is used.



Start Your Coding Journey Now!

[Login](#)
[Register](#)

```
SoftReference<Gfg> softref = new SoftReference<Gfg>(g);
```



```
//Code to illustrate Soft reference
```

```
import java.lang.ref.SoftReference;
```

```
class Gfg
```

```
{
```

```
    //code..
```

```
    public void x()
```

```
{
```

```
        System.out.println("GeeksforGeeks");
```

```
}
```

```
}
```

```
public class Example
```

```
{
```

```
    public static void main(String[] args)
```

```
{
```

```
        // Strong Reference
```

```
        Gfg g = new Gfg();
```

```
        g.x();
```

```
        // Creating Soft Reference to Gfg-type object to which 'g'
```

Start Your Coding Journey Now!

[Login](#)[Register](#)

```
// Now, Gfg-type object to which 'g' was pointing
// earlier is available for garbage collection.
g = null;

// You can retrieve back the object which
// has been weakly referenced.
// It successfully calls the method.
g = softref.get();

g.x();
}
}
```

Output:

GeeksforGeeks
GeeksforGeeks

- **Phantom References:** The objects which are being referenced by phantom references are eligible for garbage collection. But, before removing them from the memory, JVM puts them in a queue called 'reference queue'. They are put in a reference queue after calling `finalize()` method on them. To create such references [java.lang.ref.PhantomReference](#) class is used.

```
//Code to illustrate Phantom reference
import java.lang.ref.*;
class Gfg
{
    //code
    public void x()
```

Start Your Coding Journey Now!

[Login](#)[Register](#)

```
}
```

```
public class Example
```

```
{
```

```
    public static void main(String[] args)
```

```
    {
```

```
        //Strong Reference
```

```
        Gfg g = new Gfg();
```

```
        g.x();
```

```
        //Creating reference queue
```

```
        ReferenceQueue<Gfg> refQueue = new ReferenceQueue<Gfg>();
```

```
        //Creating Phantom Reference to Gfg-type object to which 'g'
```

```
        //is also pointing.
```

```
        PhantomReference<Gfg> phantomRef = null;
```

```
        phantomRef = new PhantomReference<Gfg>(g, refQueue);
```

```
        //Now, Gfg-type object to which 'g' was pointing
```

```
        //earlier is available for garbage collection.
```

```
        //But, this object is kept in 'refQueue' before
```

```
        //removing it from the memory.
```

```
        g = null;
```

```
        //It always returns null.
```

```
        g = phantomRef.get();
```

```
        //It shows NullPointerException.
```

```
        g.x();
```

```
    }
```

```
}
```



Start Your Coding Journey Now!

[Login](#)[Register](#)

```
Exception in thread "main" java.lang.NullPointerException  
    at Example.main(Example.java:31)
```

Output:

GeeksforGeeks

This article is contributed by [Pratik Agarwal](#). If you like GeeksforGeeks and would like to contribute, you can also write an article using [contribute.geeksforgeeks.org](https://www.geeksforgeeks.org/contribute) or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Like 20



Previous

Next

Start Your Coding Journey Now!

[Login](#)[Register](#)

Related Articles

1. C/C++ Pointers vs Java References
2. Method References in Java with examples
3. Can an ArrayList Contain Multiple References to the Same Object in Java?
4. Difference Between Equality of Objects and Equality of References in Java
5. Bounded Types with Generics in Java
6. Comparison of double and float primitive types in Java
7. Types of JVM Garbage Collectors in Java with implementation details
8. How to Create Different Types of Cells in a Spreadsheet using Java?



Start Your Coding Journey Now!

[Login](#)[Register](#)

10. Why Java Collections Cannot Directly Store Primitives Types?

Article Contributed By :



GeeksforGeeks

Vote for difficulty

Current difficulty : [Medium](#)

[Easy](#)[Normal](#)[Medium](#)[Hard](#)[Expert](#)

Improved By : [nidhi_biet](#)

Article Tags : [Java](#)

Practice Tags : [Java](#)

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

Start Your Coding Journey Now!

[Login](#)[Register](#)feedback@geeksforgeeks.org

Company

About Us
Careers
In Media
Contact Us
Privacy Policy
Copyright Policy
Advertise with us

Learn

DSA
Algorithms
Data Structures
SDE Cheat Sheet
Machine learning
CS Subjects
Video Tutorials
Courses

News

Top News
Technology
Work & Career
Business
Finance
Lifestyle
Knowledge

Languages

Python
Java
CPP
Golang
C#
SQL
Kotlin

Web Development

Web Tutorials
Django Tutorial
HTML
JavaScript
Bootstrap
ReactJS
NodeJS

Contribute

Write an Article
Improve an Article
Pick Topics to Write
Write Interview Experience
Internships
Video Internship

@geeksforgeeks , Some rights reserved

