



Templates in C++ vs Generics in Java

Trie Data
Structure
using smart
pointer and
OOP in C++

C++ program
to compare
two Strings
using Operator
Overloading

Derived Data
Types in C++

Go vs C++

Pointers and
References in
C++



fill in C++ STL

Deque vs
Vector in C++
STL

Types of
Literals in
C/C++ with
Examples

Strings in C++
and How to
Create them?

Conditional or
Ternary
Operator (?:) in
C/C++

C++
Programming
Basics

not1 and not2
function
templates in
C++ STL with
Examples

C++ | asm
declaration

Enum Classes
in C++ and
Their
Advantage
over Enum
DataType

How to Insert
an element at
a specific
position in an
Array in C++

C++ Program
to print an
Array using
Recursion

Count of
distinct
remainders
when N is
divided by all
the numbers
from the range
[1, N]

Difference
Between
Constructor
and Destructor
in C++

How to erase
an element
from a vector
using erase()
and
reverse_iterator?

How to
implement our
own Vector
Class in C++?

Difference
between
Inheritance
and
Polymorphism

Introduction to
C++
Programming
Language

→ OpenCV |
Hands on
Image
Contrast

Difference
between
Abstraction
and
Encapsulation
in C++

ios eof()
function in
C++ with
Examples

return
statement in
C/C++ with
Examples

OpenCV |
Saving an
Image

ios rdstate()
function in
C++ with
Examples

fill() function
in C++ STL
with examples

Hmm. We
having
trouble
finding the
site.



We can't connect to
server at
googleads.g.doubleclick.net

**If that address is correct,
here are three other things
you can try:**

- Try again later.
- Check your network
connection.
- If you are connected but
behind a firewall, please
check that Firewall has
permission to access
the Web.

Templates in C++ vs Generics in Java


While building large-scale projects, we need the code to be compatible with any kind of data which is provided to it. That is the place where your written code stands above than that of others. Here what we meant is to make the code you write be generic to any kind of data provided to the program regardless of its data type. This is where Generics in Java and the similar in C++ named Template comes in handy. While both have similar functionalities, but they differ in a few places.

Template in C++

Writing Generic programs in C++ is called Templates.

1. One of the major features of the template in C++ is the usage of metaprogramming. It Let the template signature of the provided code be different, were C++ provides the ability to implement them.
2. Template arguments can be both classes and in functions.
3. C++ requires template sources to be added in their headers.
4. Template specialization could be achieved i.e, Specific type and method of template can be implemented.





```
// CPP program to illustrate Templates
#include <iostream>
#include <string.h>

using namespace std;

template <class T>
class TempClass {
    T value;




public:
    TempClass(T item)
    {
        value = item;
    }

    T getValue()
    {
        return value;
    }
};

int main()
{
    class TempClass<string>* String =
        new TempClass<string>("Generics vs Templates");

    cout << "Output Values: " << String->getValue()
        << "\n";





    class TempClass<int>* integer = new TempClass<int>(9);
    cout << "Output Values: " << integer->getValue();
}
```

**Output:**

```
Output Values: Generics vs Templates
Output Values: 9
```

Generics in Java

1. One of the major features of Java Generics is that It handles type checking during instantiation and generates byte-code equivalent to non-generic code.
The compiler of Java checks type before instantiation, that in turn makes the implementation of Generic type-safe. Meanwhile, in C++, templates know nothing about types.
2. If Generics is applied in a class, then it gets Applied to classes and methods within classes.
3. Another major factor that leads to the use of generics in Java is because it allows you to eliminate downcasts.
4. Instantiating a generic class has no runtime overhead over using an equivalent class that uses as specific *object* rather than a generic type of *T*.



```
// Java program to illustrate
// Generics
public class GenericClass<T> {
    private T value;

    public GenericClass(T value)
    {
        this.value = value;
    }

    public void showType()
    {
        System.out.println("Type: " +
            value.getClass().getSimpleName());
        System.out.println("Value: " + value);
    }

    public static void main(String[] args)
    {
        GenericClass<String> Str =
            new GenericClass<String>("Generics vs Templates");

        GenericClass<Integer> integer =
            new GenericClass<Integer>(9);

        Str.showType();
        integer.showType();
    }
}
```

Output:

```
Type:String
Value: Generics vs Templates
Type:Integer
Value: 9
```

C++ Templates vs Generics in Java

Though both of the methods to create a generic type is similar, but they vary at some places, while the implementation property that they possess is the same.

1. **Type erasure** : Type erasure ensures tighter type check during compile time. Java generics simply offer compile-time safety and eliminate the need for casts. This is directly implemented in the Java compiler front-end and make sure type erasure is done.
2. In C++ when you use a template the compiler will emit the template code again after replacing the generic parameter in it with the type you used. This is more powerful in several ways but can lead to **bloated executables**.
3. **Wrapper class** : In Java, Even if we have to specifically specify the datatype within which the function call using any object, we don't need to cast it similar to that of C++ with actual data types, rather we use wrapper classes to do the required.

4. **Type Checking** : Java Generics handles type checking during instantiation and generates byte-code equivalent to non-generic code C++ has “latent typing” and template metaprogramming and generates a new class for each instantiation

Java encourages software reuse and adds some basic support for generic programming. For Java, it is a serious step forward in the area of commercial software development.

This article is contributed by **Aniketh Girish**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or email 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.

Hmm. We're having trouble finding that site.

Recommended Posts:

[Generics in Java](#)

[Bounded types with generics in Java](#)

[Program to convert a Set to Stream in Java using Generics](#)

[Generics in C++](#)

 [Templates in C++](#)

[Templates and Static variables in C++](#)

[Variadic function templates in C++](#)

[Templates and Default Arguments](#)

[not1 and not2 function templates in C++ STL with Examples](#)

[Java.util.LinkedList.poll\(\), pollFirst\(\), pollLast\(\) with examples in Java](#)

[Java.util.LinkedList.peek\(\) , peekfirst\(\), peeklast\(\) in Java](#)

[Java.util.LinkedList.offer\(\), offerFirst\(\), offerLast\(\) in Java](#)

[Java.util.Collections.rotate\(\) Method in Java with Examples](#)

[Java lang.Long.numberOfLeadingZeros\(\) method in Java with Examples](#)

[Java lang.Long.reverse\(\) method in Java with Examples](#)

Improved By : thundercode, Akanksha_Rai

Article Tags : [C++](#) [Java](#)

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



Be the First to upvote.

2.7

☐ To-do ☐ Done

Based on 4 vote(s)

Feedback/ Suggest Improvement

Add Notes

Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
A computer science portal for geeks



5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY

About Us
Careers
Privacy Policy
Contact Us

LEARN

Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE

Courses
Company-wise
Topic-wise
How to begin?

CONTRIBUTE

Write an Article
Write Interview Experience
Internships
Videos

@geeksforgeeks, Some rights reserved