Java Generics

In previous post we saw [Introduction to Java Generics](http://data-structure-learning.blogspot.com/2015/06/java-generics-introduction.html). In this post we will see what cast-iron guarantee is.

For instance let us take the following code (Java 1.5 or greater)

List<String> names = **new** ArrayList<String>();

names.add("String1 ");

names.add("String2");

String result = names.get(0) + names.get(1);

System.***out***.println(result);

ArrayList<E> class implements List<E> interface. So we wrote List<String> names = **new** ArrayList<String>();

Then we add two strings into the list names.add("String1 "); names.add("String2");

After that we concat both the strings using “+” operator.

Now, before Java Generics (before Java 1.5) were introduced same code was written as

List names = **new** ArrayList();

names.add("String1 ");

names.add("String2");

String result = ((String) names.get(0)) + ((String) names.get(1));

System.***out***.println(result);

The difference between both the piece of code is that before Java 5 we need to explicitly cast the elements and type parameters “**<E>**” were eliminated.

But the identical part is that they byte code compiled from both the piece of code is same. Reason is type *erasure*. If it is List<String>, List<Integer> or List<List<Integer>> all of them are represented as List in run-time. The process erases the type parameters hence the name type erasure.

As we saw that we need to cast explicitly before Java 1.5. Now, Generics does the same cast implicitly in compile time. Hence there won’t be ClassCastException at run-time. The implicit cast added by generics will never fail. This is a cast-iron guarantee provided to us.

Now the questions comes why compiler take List<String> as List? The reason is backward compatibility. Developers who developed code prior to Java 5 need not change their code according to the new syntax provided in Java 5. So at the byte code level the code that uses generics and the code that doesn’t looks same. So the developers need not shift their syntax overnight.

[Index Page](http://data-structure-learning.blogspot.com/)