Generics

The **Java Generics** programming is introduced in J2SE 5 to deal with type-safe objects. Before generics, we can store any type of objects in collection i.e. non-generic. Now generics, forces the java programmer to store specific type of objects.

Advantage of Java Generics

There are mainly 3 advantages of generics. They are as follows:

1) Type-safety: We can hold only a single type of objects in generics. It doesn't allow to store other objects.

2) Type casting is not required: There is no need to typecast the object.

Before Generics, we need to type cast:

```
List list = new ArrayList();
list.add("hello");
String s = (String) list.get(0);        //typecasting
```

After Generics, we don't need to typecast the object.

```
List<String> list = new ArrayList<String>();
list.add("hello");
String s = list.get(0);
```

3) Compile-Time Checking: It is checked at compile time so problem will not occur at runtime. The good programming strategy says it is far better to handle the problem at compile time than runtime.

```
List<String> list = new ArrayList<String>();
list.add("hello");
list.add(32);       //Compile Time Error
```

How create a generic class?

First, understand what a generic class is. A class that can refer to any type is known as generic class.

Let's see an example:

```
class MyClass{
    int x = 10;
    in y = 20;

    public static void main(String...args){
        new MyClass();
        new MyClass();
        new MyClass();
        new MyClass();
    }
}
```

In the above program every time I create an object, a memory box will create with two Integer type data member. x =10 and y = 20. The data member type won't change for every object. In the above program we can only store integer value inside the memory.

With help of generic concept, we can make generic class that will have generic type data member that can hold different type value for each object we create. It means for one object we can have integer data and for other object we can have string and so on.

We can change the data type of data member of each object of same class. Every object will hold different data type value.