```
class ArrayList{
 boolean add(Object o); //Argument is Object so no typesafety.
 Object get(int index); //return type is Object so type casting is required.
}
```

In 1.5 Version Generic version class is defined as follows

```
class ArrayList{
 boolean add(Object o); //Argument is Object so no typesafety.
 Object get(int index); //return type is Object so type casting is required.
}
```

In 1.5 Version Generic version class is defined as follows

```
                ⟹ TypeParameter
class ArrayList<T>{
 boolean add(T t);
 T get(int index);
}
```

T => Based on our runtime requirement, T will be replaced with our provided type.

class ArrayList

```
<String>{
 boolean add(String t); //We can add only String type of Object it provides TypeSafety
 String get(int index); //Retrieval Object is always of type String, so TypeCasting not  required.
   }
```

To hold only String type of Object

ArrayList

```
<String> al =new ArrayList<String>
 al.add("sachin");
 al.add(new Integer(10)); //CE: can't find symbol method: add(java.lang.Integer)
```

location: class ArrayList<String>
String name=al.get(0);
System.out.println(name);

Note:
In Generics we are associating a type-parameter to the class, such type of parameterised classes are nothing but Generic classes.
Generic class : class with type-parameter.