Article on Java **Generics**

**Generics** are one of the most controversial Java language features. A class method or interface that operators on a parameterized type is called generics ,as in generic class. Generics work only with reference type. Generics class can be part of a class hierarchy in just the same way as a non-generic class. Thus, a generic class can act as a super class or be a subclass .A method in generic class can be overridden just like any other method. A generic class cannot extend Throwable .This means that you cannot create generic exception classes

**Generic provides:**

Using generics, it is possible to create a single class .For example ,that automatically works with different types of data. Generics added the type safety that was lacking. With generics ,all cast are automatic and implicit.Thus,generics expanded you the ability to reuse code and let you do safely and easily. Generics allows a type or method operate on objects of various type while providing compile time safety. Java generics is a powerful addition on the Java language as it makes the programmers job easier and less error prone.

**5 important things on Generics:**

1. Generics are implemented using type Erasure

2. Generics does not support sub-typing

3.you cannot create Generics Arrays

4.Use of Multiple Bounds

5.Use of wildcards with extends or super to increase API flexibility.

**Some Generic restrictions:**

1.Type parameters can’t be installed

2.Restrictions on static members

3.generic array restriction

4. Generic Exception restriction

**Example on Generics:**

package oopbgen;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Vector;

public class OOPBGen {

Vector<String> data = new Vector();

ArrayList<Integer> list = new ArrayList();

public OOPBGen() {

data.add("hi");

data.add("Okay");

System.out.println("total=" + data.size());

System.out.println("second=" + data.get(1));

list.add(2);

list.add(0);

list.add(545);

for (int i = 0; i < list.size(); i++) {

System.out.println(list.get(i));

}

Collections.sort(list);

System.out.println("after sorted");

for (int i = 0; i < list.size(); i++) {

System.out.println(list.get(i));

}

}

public static void main(String[] args) {

new OOPBGen();

}

}