Beyond Generics

Hsuan-Tien Lin

Department of CSIE, NTU

OOP Class, June 7-8, 2010

More on Generic Array

We had this code last time.....

```
1  class SimpleArray<E>{
2      E[] arr; int count;
3
4      @SuppressWarnings("unchecked")
5      SimpleArray(int init_size){
6          arr = (E[])(new Object[init_size]);
7      }
8  }
```

- it works when we do

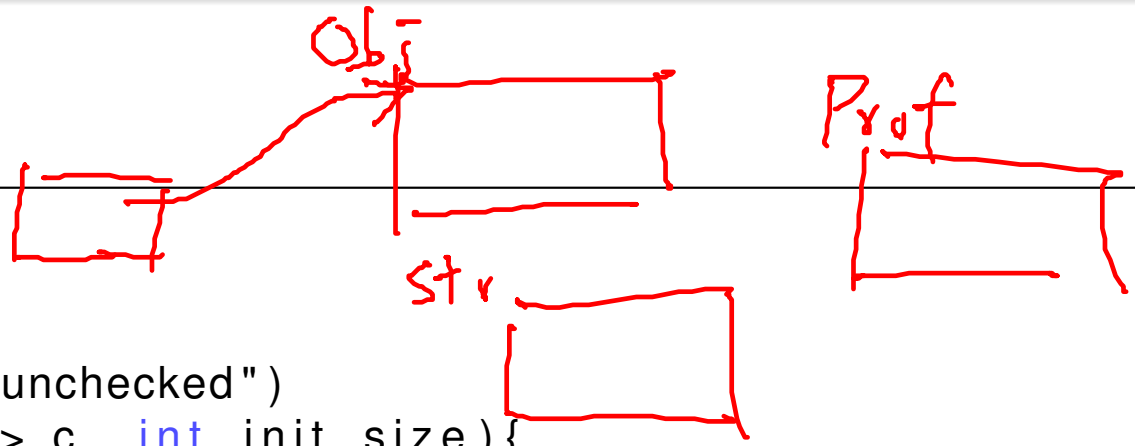
```
1  SimpleArray<Object> sao = new SimpleArray<Object>(10);
```

- it doesn't work when we do

```
1  SimpleArray<String> sas = new SimpleArray<String>(10);
```

More on Generic Array

the code that works.....




```
1  class SimpleArray<E>{
2      E[] arr; int count;
3
4      @SuppressWarnings("unchecked")
5      SimpleArray(Class<E> c, int init_size){
6          arr = (E[])java.lang.reflect.Array.newInstance(c,
               init_size);
7      }
8  }
```

```
1    SimpleArray<Object> sao =
2      new SimpleArray<Object>(Object.class, 10);
3    SimpleArray<String> sas =
4      new SimpleArray<String>(String.class, 10);
5    SimpleArray<String> sas_wrong =
6      new SimpleArray<String>(Integer.class, 10); //compile error
```

(new Object()).getClass()

More on for-each

```
1  class Util{
2    public static double cool_avg(double[] arr){
3      double res = 0.0;
4      for(double element: arr) res += element;
5      return res / arr.length;
6    }
7  }
```

- code translation for native (primitive or extended) array:

```
1    for(int i=0;i<arr.length;i++){
2      double element = arr[i];
3      res += element;
4    }
```

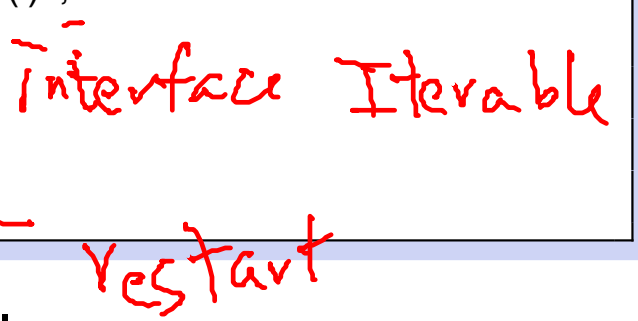
- implicitly **read-only**

More on for-each

```
1  class Util{
2    public static double cool_avg(ArrayList<Double> arr){
3      double res = 0.0;
4      for(Double element: arr) res += element;
5      return res / arr.length;
6    }
7  }
```

- code translation for `Collection<E>` (actually, any `Iterable<E>`):

```
1  for(Iterator<Double> iter=arr.iterator();
2      iter.hasNext(); ){
3    Double element = iter.next();
4    res += element;
5  }
```



- assume elements won't change during the loop