Java 核心技术(高阶)

第二章 Java泛型

第一节 泛型入门

华东师范大学 陈良育

问题

- 数据结构：存放多个(不同类型)对象
  - 容易出现转型错误ClassCastException

```
ArrayList list = new ArrayList();
list.add(123);
list.add("456");
list.add(789);

for(Object o:list)
{
    //编译可以通过  运行报错
    System.out.println((String) o));
}
```

```
Exception in thread "main" java.lang.ClassCastException: java.lang.Integer cannot be cast to java.lang.String
        at ClassCast.main(ClassCast.java:13)
```

解决方法

- 数据结构：存放多个对象(限定一种类型)
  - 不需要转型，没有转型错误

```
//<String> 限定了list只能存放字符串
ArrayList<String> list = new ArrayList<String>();
list.add("123");
list.add("456");
list.add("789");

for(String o:list)
{
    System.out.println(o);
}
```

泛型(1)

- 泛型：Generic Programming，JDK1.5推出的特性
- 编写的代码可以被很多不同类型的对象所重用

```
//<String> 限定了list只能存放字符串
ArrayList<String> list = new ArrayList<String>();
list.add("123");
list.add("456");
list.add("789");
String s = list.get(1);
```

```
//Java 7 菱形语法
//限定存储Integer
ArrayList<Integer> list2 = new ArrayList<>();
list2.add(123);
list2.add(456);
list2.add(789);
int i = list2.get(1);
```

泛型(2)

- 泛型： Generic Programming
  - 泛型类： ArrayList，HashSet，HashMap等
  - 泛型方法： Collections.binarySearch，Arrays.sort 等
  - 泛型接口： List, Iterator 等

```
//Collections.binarySearch方法支持泛型
int pos1 = Collections.binarySearch(list, "456");
int pos2 = Collections.binarySearch(list2, 456);
```

```
//Iterator接口支持泛型
Iterator<String> iter = list.iterator();
while(iter.hasNext()){
    System.out.println(iter.next());
}

Iterator<Double> iter2 = set1.iterator();
while(iter.hasNext()){
    System.out.println(iter.next());
}
```

泛型(3)

- 泛型的本质：参数化类型，避免类型转换，代码可复用
- 同类
  - C++的模板(Template)
  - C#的泛型

泛型(4)

- 进一步探索ArrayList的源码

```
public class ArrayList<E> extends AbstractList<E>
        implements List<E>, RandomAccess, Cloneable, java.io.Serializable
{
```

```
public E get(int index) {
    rangeCheck(index);

    return elementData(index);
}
```

总结

- 了解泛型的基本含义和用途
- 了解JDK中自带的泛型类和用法

谢 谢！