Java 集合框架

- List（有序不唯一）
- Set
- Map

List Set：存储的是单个数据，List 可以存储重复的数据，Set 数据不能重复

Map：存储的是一组数据

list.add(1)

list.add(2)

set.add(1)

map.put("name","张三"); key/value

map.put("张三")

map：map 的解释

Set

跟 List 一样，Set 是 Collection 的子接口，Set 集合是以散列的形式存储数据，所以元素是没有顺序的，可以存储一组无序且唯一的数据。

```
public interface Set<E> extends Collection<E>
    // Query Operations
```

Set 常用实现类：

- HashSet
- LinkedHashSet
- TreeSet

HashSet 是开发中经常使用的一个实现类，存储一组无序且唯一的对象。

无序：元素的存储顺序和遍历顺序不一致。

```
public class Test {
    public static void main(String[] args) {
        HashSet set = new HashSet();
        set.add("Hello");
        set.add("World");
        set.add("Java");
        set.add("Hello");
        Iterator iterator = set.iterator();
        while(iterator.hasNext()){
```

```
            System.out.println(iterator.next());
        }
        set.remove("World");
        System.out.println("****************");
        iterator = set.iterator();
        while(iterator.hasNext()){
            System.out.println(iterator.next());
        }
    }
}
```

LinkedHasSet 是 Set 的另外一个实现类，可以存储一组有序且唯一的元素.

有序：元素的存储顺序和遍历顺序一致。

```
package com.southwind.demo2;

import java.util.Iterator;
import java.util.LinkedHashSet;

public class Test {
    public static void main(String[] args) {
        LinkedHashSet linkedHashSet = new LinkedHashSet();
        linkedHashSet.add("Hello");
        linkedHashSet.add("World");
        linkedHashSet.add("Java");
        linkedHashSet.add("Hello");
        System.out.println("LinkedHashSet的长度是"+linkedHashSet.size());
        System.out.println("遍历LinkedHashSet");
        Iterator iterator = linkedHashSet.iterator();
        while(iterator.hasNext()){
            System.out.println(iterator.next());
        }
        linkedHashSet.remove("Java");
        System.out.println(linkedHashSet.contains("Java"));
    }
}
```

equals 和 == 的区别?

所有类中的 equals 都是继承自 Object 类，Object 类中原生的 eqauls 方法就是在通过 == 进行判断

```
public boolean equals(Object obj) {
    return (this == obj);
}
```

但是每个类都可以对 equals 方法进行重写，覆盖掉之前使用 == 进行判断的逻辑，改用新的逻辑进行判断是否相等。

LinkedHashSet 如何判断两个对象是否相等?

首先会判断两个对象的 hashCode 是否相等

什么是 hashCode?

将对象的内部信息（内存地址、属性值等），通过某种特定规则转换成一个散列值，就是该对象的 hashCode。

- 两个不同对象的 hashCode 值可能相等。
- hashCode 不相等的两个对象一定不是同一个对象。

集合在判断两个对象是否相等的时候，会先比较他们的 hashCode，如果 hashCode 不相等，则认为不是同一个对象，可以添加。

如果 hashCode 值相等，还不能认为两个对象是相等的，需要通过 equals 进行进一步的判断，equals 相等，则两个对象相等，否则两个对象不相等。

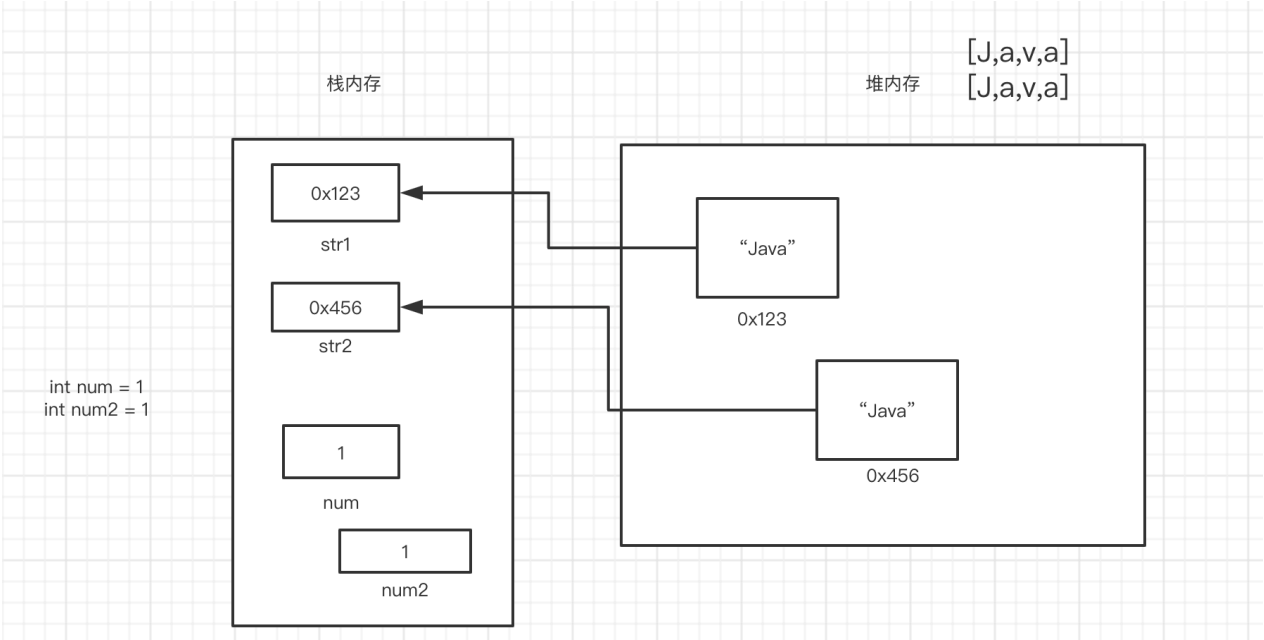
```
package com.southwind.demo2;

import java.util.Iterator;
import java.util.LinkedHashSet;

public class Test {
    public static void main(String[] args) {

        LinkedHashSet set = new LinkedHashSet();
        Data data1 = new Data(1);
        set.add(data1);
        Data data2 = new Data(1);
        set.add(data2);
        //是一个对象
        System.out.println(data1.equals(data2));
        //不是一个对象
        System.out.println(set);

    }
}

class Data{
    private int num;

    public Data(int num) {
        this.num = num;
    }

    @Override
    public String toString() {
        return "Data{" +
                "num=" + num +
                '}';
```

```
    }

    //hashcode
    @Override
    public boolean equals(Object obj) {
        if (this == obj) {
            return true;
        }
        //instanceof 判断对象是否属于某个类
        if(obj instanceof Data){
            Data data = (Data) obj;
            if(this.num == data.num){
                return true;
            }
        }
        return false;
    }

    @Override
    public int hashCode() {
        return 1;
    }
}
```

==：判断的是栈内存中的值。

引用类型的数据，栈内存中存储的是地址，所以此时 == 判断的是引用地址。

基本数据类型，栈内存中存储的是具体的数值。



栈中存储的是变量

Data data;

int num;

引用类型具体的对象（属性）存储在堆中的，再将堆中对象的内存地址赋值给栈中的变量 data，data 中存储的就是地址。

基本数据类型不需要用到堆内存，变量在栈中，变量的值直接存储在变量中。