Java SE 8 Programming Language

Lab Guides

Document Code	25e-BM/HR/HDCV/FSOFT
Version	1.1
Effective Date	20/11/2012

Hanoi, 06/2019

RECORD OF CHANGES

No	Effective Date	Change Description	Reason	Reviewer	Approver
1	01/Oct/2018	Add the new labs	Create new	DieuNT1	VinhNV
2	01/Jun/2019	Update template	Fsoft template	DieuNT1	VinhNV

Contents

FRESHER ACADEMY	CODE:	JPL.S.L401
	TYPE:	SHORT
	LOC:	
	DURATION:	60 MINUTES

Unit 8: Generics and Collections

Knowledge Summary

ArrayList

- An ArrayList is a re-sizable array, also called a dynamic array. It grows its size to accommodate new elements and shrinks the size when the elements are removed.
- ArrayList internally uses an array to store the elements. Just like arrays, It allows you to retrieve the elements by their index.
- Java ArrayList allows duplicate and null values.
- Java ArrayList is an ordered collection. It maintains the insertion order of the elements.
- You cannot create an ArrayList of primitive types like `int`, `char` etc. You need to use boxed types like `Integer`, `Character`, `Boolean` etc.
- Java ArrayList is not synchronized. If multiple threads try to modify an ArrayList at the same time, then the final outcome will be non-deterministic. You must explicitly synchronize access to an ArrayList if multiple threads are gonna modify it.

ArrayList Methods

Method	Description
void add(int position, element obj)	It inserts specified element at the specified position in the ArrayList.
boolean add(element obj)	It appends specified element to the end of the ArrayList.
boolean addAll(Collection c)	It appends all the elements of the collection to the end of the ArrayList.
element remove(int position)	It removes specified element at the specified position in the ArrayList.
boolean remove(object obj)	It removes first occurrence of specified element obj from the ArrayList.
void clear()	It removes all the elements from the ArrayList.
boolean contains(Object o)	It returns true if ArrayList contains the specified element .
object get(int position)	It returns the element at the specified position in the ArrayList.
int indexOf(Object o)	It returns first occurrence of the specified element in the list or -1 if element not found in the list.
int lastIndexOf(Object o)	It returns the last occurrence of the specified element in the list or -1 if the element is not found in the list.
int size()	It returns the number of elements in the list.

HashSet

- HashSet cannot contain duplicate values.
- HashSet allows null value.
- HashSet is an unordered collection. It does not maintain the order in which the elements are inserted.
- HashSet is not thread-safe. If multiple threads try to modify a HashSet at the same time, then the final outcome is not-deterministic. You must explicitly synchronize concurrent access to a HashSet in a multi-threaded environment.

HashSet Methods

Modifier and type	Method	Description
void	clear()	Removes all elements from the set.
Object	clone()	Returns a shallow copy of the HashSet instance: the elements themselves are not cloned.
boolean	contains(Object o)	Returns true if this set contains the specified element.
boolean	isEmpty()	Returns true if this set is empty.
boolean	add(E e)	Adds the specified element to this set if it is not already present.
boolean	remove(Object o)	Removes the specified element from this set if it is present.
boolean	removeAll(Collection<?> c)	Removes from this set all of its elements that are contained in the specified collection.
Iterator<E>	iterator()	Returns an iterator over the elements in this set.
int	size()	Returns the number of elements in this set.

Lab Guide 1: Use HashSet

Objectives:

This lab guide helps trainees know how to use HashSet in order to perform some operations:

- Create HashSet
- Retrieve elements from HashSet
- Remove elements from HashSet
- Go through HashSet using loop/iterator

Problem Descriptions:

Create a new project named **JPL.S.L401**.

Create package **fa.training.hashsetdemo** that contains:

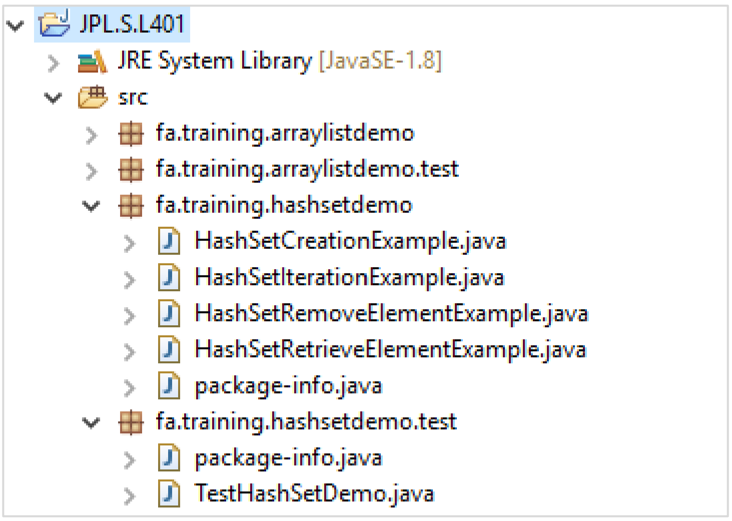
- HashSetCreationExample class
- HashSetRetrieveElementExample class
- HashSetRemoveElementExample class
- HashSetIterationExample class

Create package **fa.training.hashsetdemo.test** that contains:

- TestHashSetDemo class

Guidelines:

Step 1: Project structure:



```
JPL.S.L401
  JRE System Library [JavaSE-1.8]
  src
    fa.training.arraylistdemo
    fa.training.arraylistdemo.test
    fa.training.hashsetdemo
      HashSetCreationExample.java
      HashSetIterationExample.java
      HashSetRemoveElementExample.java
      HashSetRetrieveElementExample.java
      package-info.java
    fa.training.hashsetdemo.test
      package-info.java
      TestHashSetDemo.java
```

Step 2: Create **HashSetCreationExample** class

```
package fa.training.hashsetdemo;

import java.util.HashSet;
import java.util.Set;

/**
 * Examples of creating a HashSet
 *
 * @author hoabt2
 *
```

```java
 */
public class HashSetCreationExample {

/**
 * Create a HashSet
 *
 */
public void createHashSet() {

       System.out.println("createHashSet() !!!");

       Set<String> brands = new HashSet<>();

       brands.add("Wilson");
       brands.add("Nike");
       brands.add("Volvo");
       brands.add("IBM");
       brands.add("IBM");

       int nOfElements = brands.size();

       System.out.format("The set contains %d elements%n", nOfElements);
       System.out.println(brands);
}
}
```

Step 3: Create **HashSetRetrieveElementExample** class

```java
package fa.training.hashsetdemo;

import java.util.HashSet;
import java.util.Set;

/**
 * Examples of retrieving data from HashSet
 *
 * @author hoabt2
 *
 */
public class HashSetRetrieveElementExample {

/**
 * Get data from HashSet
 *
 */
public void retrieveElements() {

       System.out.println("retrieveElements() !!!");

       Set<String> brands = new HashSet<>();

       brands.add("Wilson");
       brands.add("Nike");
       brands.add("Volvo");
       brands.add("Kia");
       brands.add("Lenovo");

       if (brands.contains("Wilson")) {
              System.out.println("The set contains the Wilson element");
       } else {
              System.out.println("The set does not contain
                                            the Wilson element");
    }

    if (brands.contains("Apple")) {
        System.out.println("The set contains the Apple element");
    } else {
        System.out.println("The set does not contain the Apple element");
    }

    brands.clear();
    if (brands.isEmpty()) {
        System.out.println("The set does not contain any elements.");
    }
}
}

```

Step 4: Create **HashSetRemoveElementExample** class:

```
package fa.training.hashsetdemo;

import java.util.HashSet;
import java.util.Set;

/**
 * @author hoabt2
 *
 */
public class HashSetRemoveElementExample {

/**
 * Remove elements from HashSet
 *
 */
public void removeElements() {
    System.out.println("removeElements() !!!");

    Set<String> brands = new HashSet<>();

    brands.add("Wilson");
    brands.add("Nike");
    brands.add("Volvo");
    brands.add("Kia");
    brands.add("Lenovo");

    Set<String> brands2 = new HashSet<>();

    brands2.add("Wilson");
    brands2.add("Nike");
    brands2.add("Volvo");

    System.out.println(brands);

    brands.remove("Kia");
    brands.remove("Lenovo");

    System.out.println(brands);

    brands.removeAll(brands2);

    System.out.println(brands);
```

```java

        if (brands.isEmpty()) {
             System.out.println("The brands set is empty");
        }
}
}
```

Step 5: Create **HashSetIterationExample** class

```java
package fa.training.hashsetdemo;

import java.util.HashSet;
import java.util.Iterator;
import java.util.Set;

/**
 * Examples of how to iterate a HashSet
 *
 * @author hoabt2
 *
 */
public class HashSetIterationExample {

/**
 * Go through a HashSet using Iterator
 *
 */
public void hashSetIterator() {

      System.out.println("hashSetIterator() !!!");

      Set<String> brands = new HashSet<>();

      brands.add("Wilson");
      brands.add("Nike");
      brands.add("Volvo");
      brands.add("Kia");
      brands.add("Lenovo");

      Iterator<String> it = brands.iterator();

      while (it.hasNext()) {

            String element = it.next();

            System.out.println(element);
      }

}

}

```

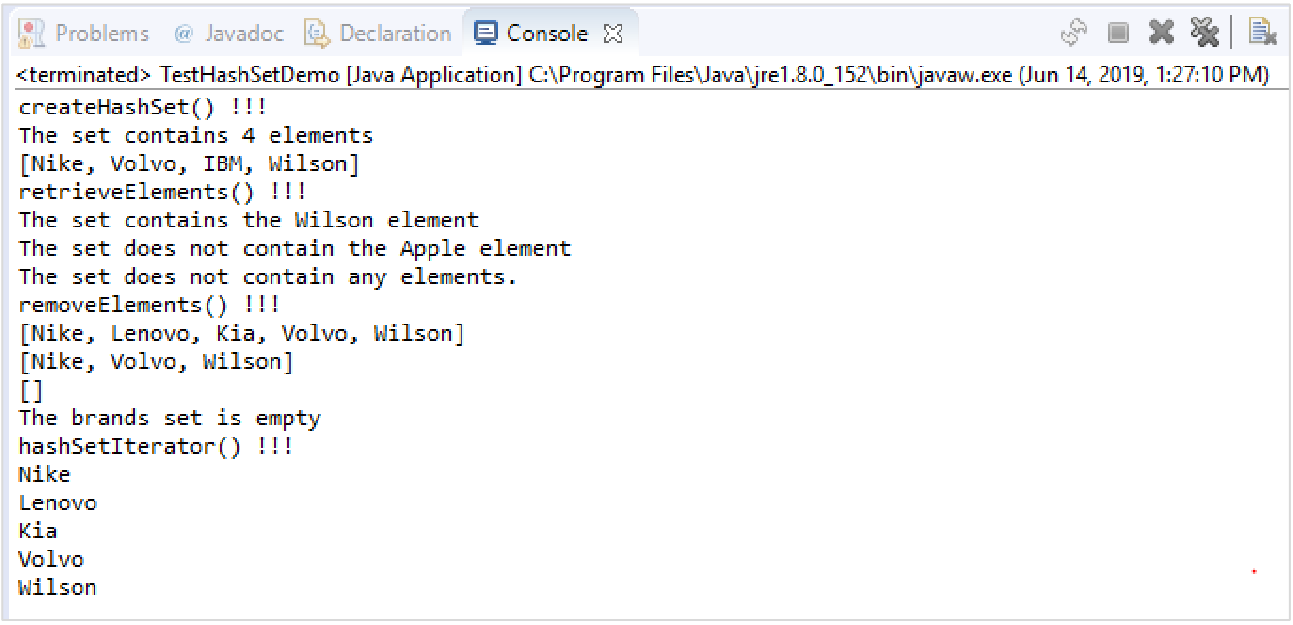
Step 6: Create **TestHashSetDemo** class

```java
package fa.training.hashsetdemo.test;

import fa.training.hashsetdemo.HashSetCreationExample;
import fa.training.hashsetdemo.HashSetIterationExample;
import fa.training.hashsetdemo.HashSetRemoveElementExample;
import fa.training.hashsetdemo.HashSetRetrieveElementExample;

/**
 * @author hoabt2
 *
 */
public class TestHashSetDemo {

/**
 * @param args
 */
public static void main(String[] args) {
      HashSetCreationExample hashSetCreation = new HashSetCreationExample();
      HashSetRetrieveElementExample hashSetElements =
                          new HashSetRetrieveElementExample();
      HashSetRemoveElementExample hashSetRemove =
                          new HashSetRemoveElementExample();
      HashSetIterationExample hashSetIterator =
                          new HashSetIterationExample();
      hashSetCreation.createHashSet();
      hashSetElements.retrieveElements();
      hashSetRemove.removeElements();
      hashSetIterator.hashSetIterator();
}
}
```

Step 7: Run **TestHashSetDemo** to see the result

You can call corresponding methods separatedly in order to test the result clearly.

Result:



```
<terminated> TestHashSetDemo [Java Application] C:\Program Files\Java\jre1.8.0_152\bin\javaw.exe (Jun 14, 2019, 1:27:10 PM)
createHashSet() !!!
The set contains 4 elements
[Nike, Volvo, IBM, Wilson]
retrieveElements() !!!
The set contains the Wilson element
The set does not contain the Apple element
The set does not contain any elements.
removeElements() !!!
[Nike, Lenovo, Kia, Volvo, Wilson]
[Nike, Volvo, Wilson]
[]
The brands set is empty
hashSetIterator() !!!
Nike
Lenovo
Kia
Volvo
Wilson
```

-- THE END --