Private Interface Method

Java 7 and Java 8

Default methods

Static method

Interface Myinterface{

void method1();

default void method2(){

sysout(“common code”)

Sysout(“test code”)

}

default void method3(){

sysout(“common code”)

Sysout(“test code”)

}

}

10 classes implemets MyInterface

Interface MyPerCombi

Permutation –n!/(n-r)!

Combinamtion ----- n!/r!\*(n-r)!

Try -with resources in Java 9

Autoclosable ----- java7

PrintWriter pr=newPrintWriter(“kdhjkh”);

try(pr)

{

}

Rule : pr has to be final or effectively final

Effectively final ----- the variable is not declared as final, but once it is initiallised the it should not be modified.

}

Collection factory methods

Mutable and Imutable objects

Modifiable collection and nonmodifiable

List<String> list=new ArrayList<>();

List.add(“aaa”);

List.add(“bbb”);

List list1=Collections.unmodifiableList(list);

list.add(“cccc”)

other way

Myclass[] e=new MyClass[]{new MyClass(1),new MyClass(2) };

List<Myclass> list=Arrays.asList(e);

Drawbacks

Changes in array will be reflected in the list

List.add(4) ///// generate UnsupportedOperationException

Set<String> set = Collections.unmodifiableSet(new HashSet<String>() {{

add(“v1"); add(“v2"); add(“v3");

}});

Stream.of(“v1", “v2", “v3")

.collect(collectingAndThen(toSet(), Collections::unmodifiableSet));

Factory methods:

12 factory methods

List, Set and Map interface

List.of()

List.of(Ele);

List.of(ele1,ele2);

List.of(ele1,ele2,………….,ele10)

List.of(Elements….ele) ///not performance efficient

Values based object

List<String> lst1=List.of(“aaa”,”bbb”);

List<String> lst2=List.of(“aaa”,”bbb”);

MyClass ob=lst1.get(0);

Set<String> s=Set.of(“aaa”,”bbb”);

Set<String> s=Set.of(“aaa”,”bbb”,”aaa”); //duplicate object IllegalArgumentExpection

You cannot add null;

Try using Set.of

Try adding duplicate values.