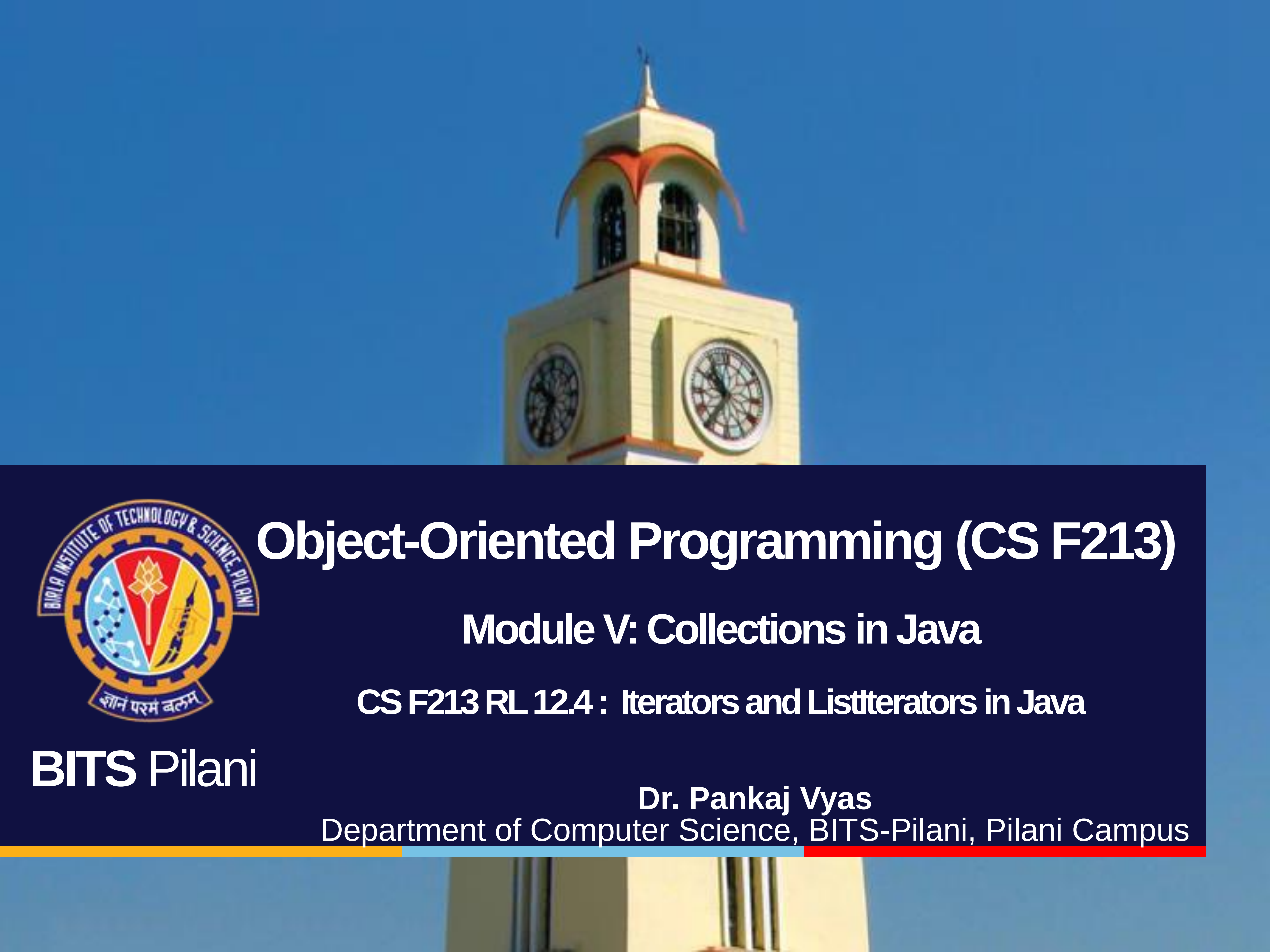


BITS Pilani

Object-Oriented Programming (CS F213)

Module V: Collections in Java

CS F213 RL 12.4 : Iterators and ListIterators in Java

Dr. Pankaj Vyas
Department of Computer Science, BITS-Pilani, Pilani Campus

CS F213 RL 12.4 : Topics

- Iterators and ListIterators

Iterator Interface

- Allows the traversing/iterating the elements of collections only in forward direction
- All Collections use iterator interface and provides method for attaching iterator for any collection.

Iterator<E> iterator();

- Methods :
  1. ***boolean hasNext()*** → ***Returns true/false if there exists next element or not***
  2. ***E next() / Object next()*** → ***Returns the next element. Used in conjunction with hasNext()***
  3. ***void remove()*** → ***Removes the element from location pointed to by iterator***

Iterator : Example 1

```
import java.util.*;
class IteratorTest
{
        public static void main(String args[])
        {
                ArrayList<String> arrStr = new ArrayList<String>(20);

                arrStr.add("A");    //          A
                arrStr.add("B");    //          A B
                arrStr.add("X");    //          A B X
                arrStr.add("Y");    //          A B X Y
                arrStr.add("Z");    //          A B X Y Z

                Iteraor<String> itr = arrStr.iterator();
                while( itr.hasNext()
                System.out.println(itr.next());
        }// End of Method
}// End of class
```

Iterator Location in the Beginning

| A B X Y Z

Next() Returns the Next Element and forwards the Cursor Location

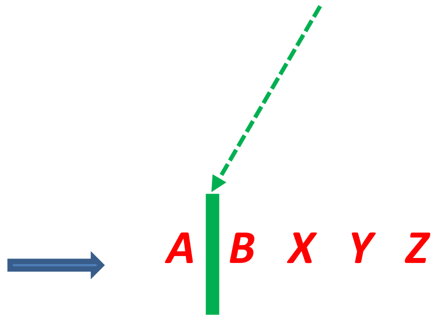
Iterator : Example 1 …

```
import java.util.*;
class IteratorTest
{
            public static void main(String args[])
            {
                        ArrayList<String> arrStr = new ArrayList<String>(20);

                        arrStr.add("A");    //          A
                        arrStr.add("B");    //          A B
                        arrStr.add("X");    //          A B X
                        arrStr.add("Y");    //          A B X Y
                        arrStr.add("Z");    //          A B X Y Z

                        Iteraor<String> itr = arrStr.iterator();
                        while( itr.hasNext()
                        System.out.println(          itr.next());
            }// End of Method
}// End of class
```

Iterator Location Changes when next() Executes and the previous Element is Returned



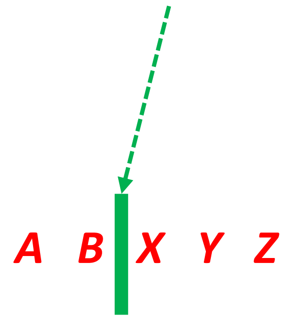
Iterator : Example 1 …

```
import java.util.*;
class IteratorTest
{
        public static void main(String args[])
        {
                ArrayList<String> arrStr = new ArrayList<String>(20);

                arrStr.add("A");    //        A
                arrStr.add("B");    //        A B
                arrStr.add("X");    //        A B X
                arrStr.add("Y");    //        A B X Y
                arrStr.add("Z");    //        A B X Y Z

                Iteraor<String> itr = arrStr.iterator();
                while( itr.hasNext()
                System.out.println(        itr.next());
        }// End of Method
}// End of class
```

Iterator Location Changes when next() Executes and the previous Element is Returned



Iterator : Example 1 …

```
import java.util.*;
class IteratorTest
{
            public static void main(String args[])
            {
                        ArrayList<String> arrStr = new ArrayList<String>(20);

                        arrStr.add("A");     //          A
                        arrStr.add("B");     //          A B
                        arrStr.add("X");     //          A B X
                        arrStr.add("Y");     //          A B X Y
                        arrStr.add("Z");     //          A B X Y Z

                        Iteraor<String> itr = arrStr.iterator();
                        while( itr.hasNext()
                        System.out.println(          itr.next()); ⟶    A  B  X | Y  Z
            }// End of Method
}// End of class
```

Removing Elements Through remove() Method of Iterator

- **remove()** → This method can be used to remove the element from a Collection
- Rules

1. remove() method via Iterator interface instance can only be invoked only once per call to next() or previous()
2. add() method should not have been executed in between after the last call to next()/previous() and the remove() Method

remove() Method of Iterator() : Example 1

```
// File Name : RemoveTest.java
import java.util.*;
class RemoveTest
{
     public static void main(String args[])
     {
          ArrayList<String> nameList = new ArrayList<String>();
          nameList.add("Java");
          nameList.add("Object");
          nameList.add("Fortran");
          nameList.add("Pascal");
          nameList.add("Python");
          Iterator<String> itrnameList = nameList.iterator();
          itrnameList.remove();
     } // End of Method
}// End of class
```

remove() Method Can not be used Without a previous next() Method Call

```
Exception in thread "main" java.lang.IllegalStateException
        at java.util.ArrayList$Itr.remove(Unknown Source)
        at RemoveTest.main(TTR-1.java:15)
```

remove() Method of Iterator() : Example 2

```
// File Name : RemoveTest.java
import java.util.*;
class RemoveTest
{
    public static void main(String args[])
    {
        ArrayList<String> nameList = new ArrayList<String>();
        nameList.add("Java");
        nameList.add("Object");
        nameList.add("Fortran");
        Iterator<String> itrnameList = nameList.iterator();
        System.out.println(nameList);
        itrnameList.next();
        itrnameList.remove();
        System.out.println(nameList);
    } // End of Method
}// End of class
```

OUTPUT

```
[Java, Object, Fortran]
[Object, Fortran]
```

remove() Method of Iterator() : Example 3

```
// File Name : RemoveTest.java
import java.util.*;
class RemoveTest
{
    public static void main(String args[])
    {
        ArrayList<String> nameList = new ArrayList<String>();
        nameList.add("Java");
        nameList.add("Object");
        nameList.add("Fortran");
        Iterator<String> itrnameList = nameList.iterator();
        System.out.println(nameList);
        itrnameList.next();
        nameList.add(“Pascal”);
        itrnameList.remove();
        System.out.println(nameList);
    } // End of Method
}// End of class
```

add() Method should not have been used in between the last calls to next() and the remove() Methods

```
[Java, Object, Fortran]
Exception in thread "main"
java.util.ConcurrentModificationException
        at
java.util.ArrayList$Itr.checkForComodification(
Unknown Source)
        at
java.util.ArrayList$Itr.remove(Unknown
Source)
        at RemoveTest.main(TTR-1.java:15)
```

ListIterator Interface

- Allows Traversal in Both Ways [First to Last and Last to First] (Sub Interface of Iterator)
- Every Collection of type List provides a method(s) for getting a suitable list iterators
- Method(s) Provided by List interface for ListIterators
  1. ListIterator listIterator() → Returns a ListIterator instance pointing at the beginning of the list
  2. ListIterator listIterator(int index) → Returns a ListIterator instance pointing at ‘index’ location . The value of ‘index’ should be in range ‘0<=index<=S’ where ‘S’ is the size of the list.

ListIterator Interface : Syntax and Important Methods

- Syntax

  interface ListIterator<E> extends Iterator<E>

- Important Methods

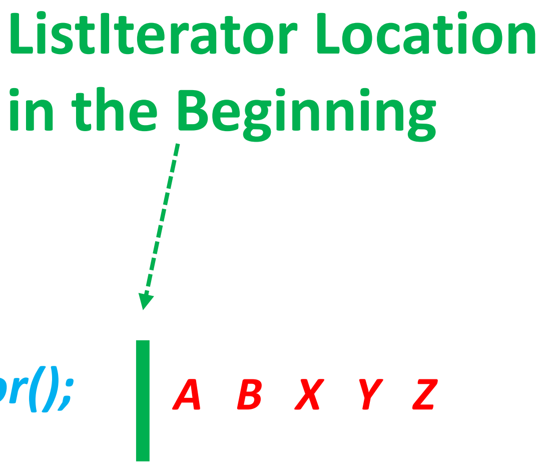
1. public void add(E e) → Inserts element ‘E’ in the list. The location is decided by the location where list iterator is pointing
2. public boolean hasNext() / hasPrevious() → Returns true if there exists the next or previous element otherwise false
3. public E next() / previous() → Returns the next or previous element and advances the cursor position forward/backward
4. public int nextIndex() / previousIndex() → Returns the index of the element returned by next() / previous()

ListIterator : Example 1 (Forward Traversal)

```
import java.util.*;
class IteratorTest
{
        public static void main(String args[])
        {
                ArrayList<String> arrStr = new ArrayList<String>(20);

                arrStr.add("A");    //          A
                arrStr.add("B");    //          A B
                arrStr.add("X");    //          A B X
                arrStr.add("Y");    //          A B X Y
                arrStr.add("Z");    //          A B X Y Z

                ListIteraor<String> itr = arrStr.listIterator();
                while( itr.hasNext()
                System.out.println(         itr.next());
        }// End of Method
}// End of class
```



ListIterator : Example 2 (Backward Traversal)

```
import java.util.*;
class IteratorTest
{
        public static void main(String args[])
        {
                ArrayList<String> arrStr = new ArrayList<String>(20);

                arrStr.add("A");    //        A
                arrStr.add("B");    //        A B
                arrStr.add("X");    //        A B X
                arrStr.add("Y");    //        A B X Y
                arrStr.add("Z");    //        A B X Y Z

                ListIteraor<String> itr = arrStr.listIterator(arrStr.size());
                while( itr.hasPrevious()
                System.out.println(itr.previous());
        }// End of Method
}// End of class
```

ListIterator Location in the End

A B X Y Z |

innovate achieve lead

Thank You