Scala Seq

Seq is a trait which represents indexed sequences that are guaranteed immutable. You can access elements by using their indexes. It maintains insertion order of elements.

Sequences support a number of methods to find occurrences of elements or subsequences. It returns a list.

Scala Seq Example

In the following example, we are creating Seq and accessing elements from Seq.

```
import scala.collection.immutable._

object MainObject{

   def main(args:Array[String]){

      var seq:Seq[Int] = Seq(52,85,1,8,3,2,7)

      seq.foreach((element:Int) => print(element+" "))

      println("\nAccessing element by using index")

      println(seq(2))

   }

}
```

Output:

```
52 85 1 8 3 2 7

Accessing element by using index

1
```

You can also access elements in reverse order by using reverse method. Below we have listed some commonly used method and their description.

Commonly used Methods of Seq

Method	Description
def contains[A1 >: A](elem: A1): Boolean	Check whether the given element present in

this sequence.

def copyToArray(xs: Array[A], start: Int, len: Int): Unit It copies the seq elements to an array.

def endsWith[B](that: GenSeq[B]): Boolean It tests whether this sequence ends with the given sequence or not.

def head: A It selects the first element of this seq collection.

def indexOf(elem: A): Int It finds index of first occurrence of a value in this immutable sequence.

def isEmpty: Boolean It tests whether this sequence is empty or not.

def lastIndexOf(elem: A): Int It finds index of last occurrence of a value in this immutable sequence.

def reverse: Seq[A] It returns new sequence with elements in reversed order.

Scala Seq Example

In this example, we have applied some predefined methods of Seq trait.

```
import scala.collection.immutable._
object MainObject{
   def main(args:Array[String]){
      var seq:Seq[Int] = Seq(52,85,1,8,3,2,7)
      seq.foreach((element:Int) => print(element+" "))
      println("\nis Empty: "+seq.isEmpty)
      println("Ends with (2,7): "+ seq.endsWith(Seq(2,7)))
      println("contains 8: "+ seq.contains(8))
      println("last index of 3 : "+seq.lastIndexOf(3))
      println("Reverse order of sequence: "+seq.reverse)
   }
}
```

Output:

52 85 1 8 3 2 7

is Empty: false

Ends with (2,7): true

contains 8: true

last index of 3 : 4

Reverse order of sequence: List(7, 2, 3, 8, 1, 85, 52)