Mukunda Paudel
paudelmuku@gmail.com

ArrayList in C#

- **ArrayList** is a powerful feature of C# language.
- It is the non-generic type of collection which is defined in *System.Collections* namespace.
- It is used to create a dynamic array means the size of the array is increase or decrease automatically according to the requirement of your program, there is no need to specify the size of the ArrayList.
- Or in other words, ArrayList represents an ordered collection of an object that can be indexed individually.
- In ArrayList, you can store elements of the same type and of the different types.
- It belongs to the non-generic collection.

Important Points:

- ✓ The ArrayList class inherits the Object class.
- ✓ The ArrayList is defined under *System.Collections* namespace. So, when you use Arraylist in your program you must add *System.Collections* namespace.
- ✓ You cannot implement a multi-dimensional array using ArrayList.
- ✓ The capacity of an ArrayList is the number of elements the ArrayList can hold.
- ✓ You are allowed to use duplicate elements in your ArrayList.
- ✓ You can apply searching and sorting on the elements present in the ArrayList.
- ✓ Arraylist can accept null as a valid value.

How to create the ArrayList?

ArrayList class has ***three constructors*** which are used to create the ArrayList.

1. **ArrayList():** This constructor is used to create an instance of ArrayList class which is empty and having no initial capacity.

2. **ArrayList(Int32):** This constructor is used to create an instance of ArrayList class which is empty and having the specified initial capacity.

3. **ArrayList(ICollection):** This constructor is used to create an array list initialized with the elements from the specified collection and having the same initial capacity which is copied from collection.

<u>Let's see how to create an ArrayList using ArrayList() constructor:</u>

- **Step 1:** Include *System.Collections* namespace in your program with the help of *using* keyword.
  Syntax: using System.Collections;

  Step 2: Create an ArrayList using ArrayList class as shown below:

  ArrayList list_name = new ArrayList();

  Step 3: If you want to add elements in your ArrayList then use **Add()** method to add elements in your ArrayList. As shown in the below example.

  Step 4: The elements of the ArrayList is accessed by using a foreach loop, or by for loop, or by indexer.

Example: *Below program show how to create an ArrayList, adding elements to the ArrayList, and how to access the elements of the ArrayList.*

```
using System;
using System.Collections;

class EEC
   {

   // Main Method
   static public void Main()
   {

       // Creating ArrayList
       ArrayList My_array = new ArrayList();

     // adding elements in the My_array ArrayList This ArrayList contains
     elements of different types
       My_array.Add(123);
       My_array.Add("Everest Engineering College!");
       My_array.Add(null);
       My_array.Add('M');
       My_array.Add(12.34);

       // Accessing the elements of My_array ArrayList Using foreach loop
       foreach (var elements in My_array)
       {
           Console.WriteLine(elements);
       }
   }
}
```

O/P:

```
123

Everest Engineering College!


M

12.34
```

How to find the Capacity and Count of elements of the ArrayList?

- To find this we can use Count and Capacity property of an ArrayList class as follows**:**

```
// C# program to find the number of elements and capacity of ArrayList
using System;
using System.Collections;

class EEC
 {
     // Driver code
     public static void Main()
     {
         // Creating an ArrayList
         ArrayList myList = new ArrayList();

         // Adding elements to ArrayList
         myList.Add(1);
         myList.Add(2);
         myList.Add(3);
         myList.Add(4);
         myList.Add(5);

         // Displaying count of elements of ArrayList
         Console.WriteLine("Number of elements: " + myList.Count);

         // Displaying Current capacity of ArrayList
         Console.WriteLine("Current capacity: " + myList.Capacity);
     }
 }
```

O/P:

```
Number of elements: 5

Current capacity: 8
```

Mukunda Paudel
paudelmuku@gmail.com

How to remove elements from the ArrayList?

- ❖ In ArrayList, you are allowed to remove elements from the ArrayList.
- ❖ ArrayList provides four different methods to remove elements and the methods are:

1. **Remove ():**
   - ✓ This method is used to remove the first occurrence of a specific object from the ArrayList.
2. **RemoveAt ():**
   - ✓ This method is used to remove the element at the specified index of the ArrayList.
3. **RemoveRange ():**
   - ✓ This method is used to remove a range of elements from the ArrayList.
4. **Clear ():**
   - ✓ This method is used to remove all the elements from the ArrayList.

```
// C# program to illustrate how to remove elements from the ArrayList
            using System;
            using System.Collections;

       class EEC
         {
            static public void Main()
            {
                // Creating ArrayList
                ArrayList My_array = new ArrayList();

              // Adding elements in the My_array ArrayList This ArrayList contains
              elements of the same types
                My_array.Add('E');
                My_array.Add('V');
                My_array.Add('E');
                My_array.Add('R');
                My_array.Add('E');
                My_array.Add('S');
                My_array.Add('T');
                My_array.Add('C');
                My_array.Add('O');
                My_array.Add('L');
                My_array.Add('L');
                My_array.Add('E');
                My_array.Add('G');

                Console.WriteLine("Initial number of elements : "+ My_array.Count);

                // Remove the 'E' element from the ArrayList Using Remove() method
                My_array.Remove('E');
                Console.WriteLine("After Remove() method the " +
                                  "number of elements: " + My_array.Count);

                // Remove the element present at index 8 Using RemoveAt() method
                My_array.RemoveAt(8);
                Console.WriteLine("After RemoveAt() method the " +
                                  "number of elements:" + My_array.Count);

                // Remove 3 elements starting from index 1 using RemoveRange() method
                My_array.RemoveRange(1, 3);
                Console.WriteLine("After RemoveRange() method the" +
                                  " number of elements: " + My_array.Count);

                // Remove the all element present in ArrayList Using Clear() method
                My_array.Clear();
                Console.WriteLine("After Clear() method the " +
                                   "number of elements:" + My_array.Count);
            }
        }
```

O/P:

```
Initial number of elements : 13
After Remove() method the number of elements: 12
After RemoveAt() method the number of elements: 11
After RemoveRange() method the number of elements: 8
After Clear() method the number of elements: 0
```

How to sort the elements of the ArrayList?

- In ArrayList, you can perform sorting on the elements present in the given ArrayList using the Sort() method. This method uses the QuickSort algorithm to perform sorting on the ArrayList and the elements are arranged in ascending order.

```
public class EEC
      {
     static public void Main()
     {
         // Creating ArrayList
         ArrayList My_array = new ArrayList();
         // Adding elements in the My_array ArrayList This ArrayList contains
         elements of the same types
         My_array.Add(1);
         My_array.Add(6);
         My_array.Add(40);
         My_array.Add(10);
         My_array.Add(5);
         My_array.Add(3);
         My_array.Add(2);
         My_array.Add(4);
         // ArrayList before sorting
         Console.WriteLine("ArrayList before using Sort() method:");

         foreach (var elements in My_array)
         {
             Console.WriteLine(elements);
         }

         // Sort the elements of the ArrayList Using sort() method
         My_array.Sort();
         // ArrayList after sorting
         Console.WriteLine("ArrayList after using Sort() method:");
         foreach (var elements in My_array)
         {
             Console.WriteLine(elements);
         }
     }
 }
```

O/P:

```
ArrayList before using Sort() method:
1
6
40
10
5
3
2
4
ArrayList after using Sort() method:
1
2
3
4
5
6
10
40
```

Difference between Array and ArrayList

Array (Simple Array)	**Array List**
1. Array is strongly typed. i.e. an array can store only specific type of items\elements 2. Arrays belong to **System.Array** namespace 3. In Arrays, we can store only one datatype either int, string, char etc. 4. Insertion and deletion operation is fast. 5. Arrays are strongly typed which means it can store only specific type of items or elements. 6. Array cannot accept null.	1. ArrayList can store any type of items\elements. 2. ArrayList belongs to **System.Collection** namespace. 3. In ArrayList we can store different datatype variables. 4. Insertion and deletion operation in ArrayList is slower than an Array. 5. Arraylist are not strongly typed. 6. ArrayList can accepts null.

Example- Array and Array List

```
       // Main Method
    public static void Main(string[] args)
        {
            // creating array
            int[] arr = new int[4];
            // initializing array
            arr[0] = 10;
            arr[1] = 15;
            arr[2] = 20;
            arr[3] = 25;

            // traversing array
            for (int i = 0; i < arr.Length; i++)
            {
                Console.WriteLine(arr[i]);
            }
        }
```

Example-ArrayList

```
class EEC
       {
            public static void Main(string[] args)
            {
                // Create a list of strings
                ArrayList al = new ArrayList();
                al.Add("BECMP");
                al.Add("IVSem");
                al.Add(41);
                al.Add(12.34);
                // Iterate list element using foreach loop
                foreach (var names in al)
                {
                    Console.WriteLine(names);
                }
            }
        }
```