**Part 74 - List collection class in c#**

List is one of the generic collection classes present in **System.Collections.Generic** namespcae. There are several generic collection classes in System.Collections.Generic namespace as listed below.  
**1.** Dictionary - Discussed in [Parts 72](http://csharp-video-tutorials.blogspot.com/2013/08/part-72-what-is-dictionary-in-c.html) & [73](http://csharp-video-tutorials.blogspot.com/2013/08/part-73-what-is-dictionary-in-c.html)  
**2.** List  
**3.** Stack  
**4.** Queue etc

A List class can be used to create a collection of any type. For example, we can create a list of Integers, Strings and even complex types. The objects stored in the list can be accessed by index. Unlike arrays, lists can grow in size automatically. This class also provides methods to search, sort, and manipulate lists.

public class Program  
{  
    public static void Main()  
    {  
        // Create Customer Objects  
        Customer customer1 = new Customer()  
        {  
            ID = 101,  
            Name = "Mark",  
            Salary = 5000  
        };  
  
        Customer customer2 = new Customer()  
        {  
            ID = 102,  
            Name = "Pam",  
            Salary = 7000  
        };  
  
        Customer customer3 = new Customer()  
        {  
            ID = 104,  
            Name = "Rob",  
            Salary = 5500  
        };  
  
        Customer[] arrayCustomers = new Customer[2];  
        arrayCustomers[0] = customer1;  
        arrayCustomers[1] = customer2;  
        // The following line will throw an exception, Index was outside the bounds of the array.   
        // This is because, arrays does not grow in size automatically.  
        // arrayCustomers[2] = customer3;  
          
        // Create a List of Customers. Here, we have set the size to 2. But when I add a third   
        // element the list size will automatically grow and we will not get an exception.

        List<Customer> listCustomers = new List<Customer>(2);

  // To add an element to the list, use Add() method.  
        listCustomers.Add(customer1);  
        listCustomers.Add(customer2);  
        // Adding an element beyond the initial capacity of the list will not throw an exception.  
        listCustomers.Add(customer3);  
  
        // Items can be retrieved from the list by index. The following code will   
        // retrieve the first item from the list. List index is ZERO based.  
        Customer cust = listCustomers[0];  
        Console.WriteLine("ID = {0}, Name = {1}, Salary = {2}",  
                 cust.ID, cust.Name, cust.Salary);  
        Console.WriteLine("------------------------------------------------");  
  
        // foreach or for loop can be used to iterate thru all the items in the list  
        // Using for loop  
        for (int i = 0; i < listCustomers.Count; i++)  
        {  
            Customer customer = listCustomers[i];  
            Console.WriteLine("ID = {0}, Name = {1}, Salary = {2}",  
                     customer.ID, customer.Name, customer.Salary);  
        }  
        Console.WriteLine("------------------------------------------------");  
  
        // Using foreach loop  
        foreach (Customer c in listCustomers)  
        {  
            Console.WriteLine("ID = {0}, Name = {1}, Salary = {2}", c.ID, c.Name, c.Salary);  
        }  
        Console.WriteLine("------------------------------------------------");  
  
        // All generic collection classes including List are strongly typed. This means

    // if you have created a List of type Customer, only objects of type Customer   
        // can be added to the list. If you try to add an object of different type you would   
        // get a compiler error. The following line will raise a compiler error.  
        // listCustomers.Add("This will not compile");  
  
        // If you want to insert an item at a specific index location of the list, use Insert() method.   
        // The following line will insert customer3 object at index location 1.  
        listCustomers.Insert(1, customer3);  
        Console.WriteLine("ID = {0}, Name = {1}, Salary = {2}",   
               listCustomers[1].ID, listCustomers[1].Name, listCustomers[1].Salary);  
        Console.WriteLine("------------------------------------------------");  
  
        // To get the index of specific item in the list use Indexof() method  
        Console.WriteLine("Index of Customer3 object in the List = " +        
                listCustomers.IndexOf(customer3));  
        Console.WriteLine("------------------------------------------------");  
    }  
}  
  
public class Customer  
{  
    public int ID { get; set; }  
    public string Name { get; set; }  
    public int Salary { get; set; }  
}