**Part 80 - Some useful methods of List collection class**

This is continuation to [Part 79](http://csharp-video-tutorials.blogspot.com/2013/09/part-79-sort-list-of-complex-types.html). Please watch [Part 79](http://csharp-video-tutorials.blogspot.com/2013/09/part-79-sort-list-of-complex-types.html), before proceeding.  
  
**In this video, we will discuss the following methods**  
**1. TrueForAll() -** Returns true or false depending on whether if every element in the list matches the conditions defined by the specified predicate.  
  
**2. AsReadOnly() -**Returns a read-only wrapper for the current collection. Use this method, if you don't want the client code to modify the collection i.e add or remove any elements from the collection. The ReadOnlyCollection will not have methods to add or remove items from the collection. You can only read items from this collection.  
  
**3. TrimExcess() -**Sets the capacity to the actual number of elements in the List, if that number is less than a threshold value.

**According to MSDN:**  
This method can be used to minimize a collection's memory overhead if no new elements will be added to the collection. The cost of reallocating and copying a large List<T> can be considerable, however, so the TrimExcess method does nothing if the list is at more than 90 percent of capacity. This avoids incurring a large reallocation cost for a relatively small gain. The current threshold is 90 percent, but this could change in the future.

public class Program  
{  
    public static void Main()  
    {  
        Customer customer1 = new Customer()  
        {  
            ID = 101,  
            Name = "Mark",  
            Salary = 5200  
        };  
  
        Customer customer2 = new Customer()  
        {  
            ID = 103,  
            Name = "John",  
            Salary = 7000  
        };  
  
        Customer customer3 = new Customer()  
        {  
            ID = 102,  
            Name = "Ken",  
            Salary = 5500  
        };  
  
        List<Customer> listCutomers = new List<Customer>(100);  
        listCutomers.Add(customer1);  
        listCutomers.Add(customer2);  
        listCutomers.Add(customer3);  
  
        Console.WriteLine("Are all salaries greater than 5000: "   
            + listCutomers.TrueForAll(x => x.Salary > 5000));  
  
        // ReadOnlyCollection will not have Add() or Remove() methods  
        System.Collections.ObjectModel.ReadOnlyCollection<Customer>   
            readOnlyCustomers = listCutomers.AsReadOnly();  
  
        Console.WriteLine("Total Items in ReadOnlyCollection = " +  
            readOnlyCustomers.Count);  
  
        // listCutomers list is created with an initial capacity of 100  
        // but only 3 items are in the list. The filled percentage is   
        // less than 90 percent threshold.  
        Console.WriteLine("List capacity before invoking TrimExcess = " +  
                listCutomers.Capacity);  
        // Invoke TrimExcess() to set the capacity to the actual   
        // number of elements in the List  
        listCutomers.TrimExcess();  
        Console.WriteLine(listCutomers.Capacity);  
    }  
}  
  
public class Customer  
{  
    public int ID { get; set; }  
    public string Name { get; set; }  
    public int Salary { get; set; }  
}