**Part 73 - What is dictionary in c# continued**

In this video, we will discuss the following methods of Dictionary class.  
**1.** TryGetValue()  
**2.** Count()  
**3.** Remove()  
**4.** Clear()  
**5.** Using LINQ extension methods with Dictionary  
**6.** Different ways to convert an array into a dictionary

**Code used in the demo:**  
public class Program  
{  
    public static void Main()  
    {  
        // Create Customer Objects  
        Customer customr1 = new Customer()  
        {  
            ID = 101,  
            Name = "Mark",  
            Salary = 5000  
        };  
  
        Customer customr2 = new Customer()  
        {  
            ID = 102,  
            Name = "Pam",  
            Salary = 7000  
        };  
  
        Customer customr3 = new Customer()  
        {  
            ID = 104,  
            Name = "Rob",  
            Salary = 5500  
        };  
  
        // Create a Dictionary, CustomerID is the key. Type is int  
        // Customer object is the value. Type is Customer  
        Dictionary<int, Customer> dictionaryCustomers = new Dictionary<int, Customer>();  
  
        // Add customer objects to the dictionary  
        dictionaryCustomers.Add(customr1.ID, customr1);  
        dictionaryCustomers.Add(customr2.ID, customr2);  
        dictionaryCustomers.Add(customr3.ID, customr3);  
  
        // If you are not sure if a key is present or not, you can use   
        // TryGetValue() method to get the value from a dictionary.  
        Customer customer999;  
        if (dictionaryCustomers.TryGetValue(999, out customer999))  
        {  
            Console.WriteLine("ID = {0}, Name = {1}, Salary = {2}", customer999.ID, customer999.Name, customer999.Salary);  
        }  
        else  
        {  
            Console.WriteLine("Customer with Key = 999 is not found in the dictionary");  
            Console.WriteLine("-------------------------------------------------------------------");  
        }  
  
        // To find the total number of items in a dictionary use Count() method  
        Console.WriteLine("Total items in Dictionary = {0}", dictionaryCustomers.Count());  
        Console.WriteLine("-------------------------------------------------------------------");  
  
        // LINQ extension methods can be used with Dictionary. For example, to find the   
        // total employees whose salary is greater than 5000.  
        Console.WriteLine("Items in dictionary where Salary is greater than 5000 = {0}",  
            dictionaryCustomers.Count(x => x.Value.Salary > 5000));  
        Console.WriteLine("-------------------------------------------------------------------");  
  
        // To remove an item from the dictionary, use Remove() method  
        dictionaryCustomers.Remove(101);  
  
        // To remove all items from the dictionary, use Clear() method  
        dictionaryCustomers.Clear();  
  
        // Create an array of customers  
        Customer[] arrayCustomers = new Customer[3];  
        arrayCustomers[0] = customr1;  
        arrayCustomers[1] = customr2;  
        arrayCustomers[2] = customr3;  
  
        // Convert customer array to a dictionary using ToDictionary() method.  
        // In this example, key is Customer ID and value is the customer object  
        Dictionary<int, Customer> dict = arrayCustomers.ToDictionary(customer => customer.ID, customer => customer);  
        // OR          
        // Dictionary<int, Customr> dict = arrayCustomers.ToDictionary(customer => customer.ID);  
        // OR use a foreach loop  
        // Dictionary<int, Customer> dict = new Dictionary<int, Customer>();  
        // foreach (Customer cust in arrayCustomers)  
        // {  
        //     dict.Add(cust.ID, cust);  
        // }  
  
        // Loop thru the dictionary and print the key/value pairs  
        foreach (KeyValuePair<int, Customer> kvp in dict)  
        {  
            Console.WriteLine("Key = {0}", kvp.Key);  
            Customer customr = kvp.Value;  
            Console WriteLine("ID = {0}, Name = {1}, Salary {2}", customr.ID, customr.Name, customr.Salary);  
        }  
        Console.WriteLine("-------------------------------------------------------------------");  
    }  
}  
  
public class Customer  
{  
    public int ID { get; set; }  
    public string Name { get; set; }  
    public int Salary { get; set; }  
}s