var dataTable = new System.Data.DataTable(Guid.NewGuid().ToString());

var columnCode = new DataColumn("Code");

var columnProduct = new DataColumn("Product");

dataTable.Columns.AddRange(new DataColumn[]

{

columnCode,

columnProduct

});

var item = new List<SomeClass>();

item.Select(data => new

{

data.Id,

data.SomeValue

}).AddToDataTable(dataTable);

}

public static DataTable ToDataTable<T>(IList<T> data)

{

PropertyDescriptorCollection props =

TypeDescriptor.GetProperties(typeof(T));

DataTable table = new DataTable();

Type Propiedad = null;

for (int i = 0; i < props.Count; i++)

{

PropertyDescriptor prop = props[i];

Propiedad = prop.PropertyType;

if (Propiedad.IsGenericType && Propiedad.GetGenericTypeDefinition() == typeof(Nullable<>))

{

Propiedad = Nullable.GetUnderlyingType(Propiedad);

}

table.Columns.Add(prop.Name, Propiedad);

}

object[] values = new object[props.Count];

foreach (T item in data)

{

for (int i = 0; i < values.Length; i++)

{

values[i] = props[i].GetValue(item);

}

table.Rows.Add(values);

}

return table;

}

**= = = = = =**

**ANOTHER WAY, SIMPLER**

Convert enum to DataTable

using System.Reflection;

private DataTable ConvertEnumToDataTable(IEnumerable ien)  
{  
 DataTable dt = new DataTable();  
 foreach (object obj in ien)  
 {  
 Type t = obj.GetType();  
 PropertyInfo[] pis = t.GetProperties();

if (dt.Columns.Count == 0)  
 {  
 foreach (PropertyInfo pi in pis)  
 {  
 dt.Columns.Add(pi.Name, pi.PropertyType);  
 }  
 }  
 DataRow dr = dt.NewRow();

foreach (PropertyInfo pi in pis)  
 {  
 object value = pi.GetValue(obj, null);  
 dr[pi.Name] = value;  
 }  
 dt.Rows.Add(dr);  
 }  
 return dt;  
}