# Temp Object Delete

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IF OBJECT\_ID('tempdb..#tmp') IS NOT NULL** | | | |  |  |  |  |  |  |  |  |  |  |
|  |  |  | **DROP TABLE #tmp** | |  |  |  |  |  |  |  |  |  |
| **IF OBJECT\_ID('tempdb..#tmp1') IS NOT NULL** | | | |  |  |  |  |  |  |  |  |  |  |
|  |  |  | **DROP TABLE #tmp1** | |  |  |  |  |  |  |  |  |  |
|  |  |  | IF OBJECT\_ID('tempdb..#temp\_columns') IS NOT NULL | | | | | |  |  |  |  |  |
|  |  |  | DROP TABLE #temp\_columns | | |  |  |  |  |  |  |  |  |
|  |  |  | IF OBJECT\_ID('tempdb..#temp2') IS NOT NULL | | | | |  |  |  |  |  |  |
|  |  |  | DROP TABLE #temp2 | | |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| insert into AsisErrorLog(Spname,ErrorMessage,createddate) | | | | |  |  |  |  |  |  |  |  |  |
|  |  | values(OBJECT\_NAME(@@PROCID),'Error\_line: '+CONVERT(VARCHAR(10),error\_line())+' - '+error\_message(),getdate()) | | | | | | | | | | | |

# Arithabort

# GET CLIENT IP ADDRESS

|  |
| --- |
| audit.VisitorIP = HttpContext.Current.Request.UserHostAddress==null?" ": HttpContext.Current.Request.UserHostAddress; |
| //audit.VisitorIP = HttpContext.Current.Request.ServerVariables["HTTP\_X\_FORWARDED\_FOR"] |
| // == null ? (HttpContext.Current.Request.ServerVariables["REMOTE\_ADDR"] == null ? " " : HttpContext.Current.Request.ServerVariables["REMOTE\_ADDR"]) : HttpContext.Current.Request.ServerVariables["HTTP\_X\_FORWARDED\_FOR"]; |
| audit.VisitedArea = HttpContext.Current.Request.Url.AbsoluteUri==null?" ": HttpContext.Current.Request.Url.AbsoluteUri; |
|  |
| public static string GetClientIp() |
| { |
| var context = HttpContext.Current; |
| var clientIPAddr= context.Request.ServerVariables["HTTP\_X\_FORWARDED\_FOR"]; |
| if (string.IsNullOrEmpty(clientIPAddr)) |
| { |
| return context.Request.ServerVariables["REMOTE\_ADDR"]; |
| } |
| else |
| { |
| var ipArray = clientIPAddr.Split(new Char[] { ',' }); |
| return ipArray[0]; |
| } |
| } |

# Cursor

Declare @HospitalId int

declare Insertsample cursor for select Hospitalid from gmhospitalmst where hospitalid in (85,167)

open Insertsample

fetch next from Insertsample into @hospitalid

while @@FETCH\_STATUS=0

begin

print @hospitalid

fetch next from insertsample into @hospitalid

end

close Insertsample

Deallocate Insertsample

# DB Notification / Email

EXEC msdb.dbo.sp\_send\_dbmail

@profile\_name = 'ASIS Application'

, @recipients = 'thamaraikannan.r@changepond.com'

, @copy\_recipients = ''

, @subject = 'ASIS Daily Report'

, @body\_format = 'HTML'

, @body = @def

, @importance ='normal'

, @file\_attachments = @FILENAME

# EXCEL in JS

**var blob = new Blob([result.data], { type: "application/vnd.ms-excel"});**

**var objectUrl = URL.createObjectURL(blob);**

**var link = document.createElement("a");**

**document.body.appendChild(link);**

**link.download = "Fems SubParameter BreakUp.xls";**

**link.href = result;**

**link.click();**

**window.open(objectUrl);**

**var s = result.data;**

var a = document.createElement('a');

var data\_type = 'data:application/vnd.ms-excel';

a.href = data\_type + ', ' + encodeURIComponent(response.data.m\_StringValue);

a.download = subParameter +"("+DateFormat()+")"+ '.xls';

a.click();

$scope.ExportLoaderDisplay = false;

# Assembly Stored Procedure

**USE [ASISPATWeb]**

**GO**

**/\*\*\*\*\* Object: StoredProcedure [dbo].[InsertResultSetsToTables] Script Date: 3/10/2020 8:12:40 PM \*\*\*\*\*/**

**SET ANSI\_NULLS OFF**

**GO**

**SET QUOTED\_IDENTIFIER OFF**

**GO**

**CREATE PROCEDURE [dbo].[InsertResultSetsToTables]**

**@sourceQuery [nvarchar](max),**

**@targetTableList [nvarchar](max)**

**WITH EXECUTE AS CALLER**

**AS**

**EXTERNAL NAME [ASIS1].[StoredProcedures].[InsertResultSetsToTables]**

**GO**

# Date to DAY for Month

**var WcEntity = WcDetEntity == null ? WcDetEntity : WcDetEntity;**

**List<FMSWorkingCalendarDetEntity> ObjWCEntity = null;**

**var dayList = Enumerable.Range(Month, 1).Select(m =>**

**Enumerable.Range(1, DateTime.DaysInMonth(year, m)) // Days: 1, 2 ... 31 etc.**

**.Select(day => new FMSWorkingCalendarDetEntity**

**{**

**Day = new DateTime(year, m, day).Day,**

**DayName = new DateTime(year, m, day).DayOfWeek.ToString().Substring(0, 3),**

**WeekNo = CultureInfo.InvariantCulture.Calendar.GetWeekOfYear(new DateTime(year, m, day), System.Globalization.CalendarWeekRule.FirstDay, DayOfWeek.Monday),**

**}).ToList()).ToList(); // Map each day to a date**

**return dayList;**

(<https://www.c-sharpcorner.com/UploadFile/f0b2ed/temporary-table-vs-temporary-variable-in-sql-server/> )-Refer Site./Temp\_variable.

# COPY the values from one Table to Another Table

select \*into #tbl\_Destination\_table from #tbl\_Source\_table

# TEMPORARY TABLES AND VALUES

**# Temporary Variables**

Declare @number2 int;

SET @number2=5;

print @number2;

**# Temporary Tables**

Declare @MyData TABLE

(

IID int Unique ,

Name Nvarchar(50) primary Key,

Salary Int Check (Salary<50000),

City\_Name Nvarchar(50) Not Null Default('Alwar')

)

insert into @MyData values (02,'AAA',5000,'KTG'),(03,'BBB',25000,'CNR')

select\*from @MyData

**#Rollback**

**(Temporary variable will not support Rollback Option)**

Declare @MyData TABLE

(

ID int Unique ,

Name Nvarchar(50) primary Key,

Salary Int Check (Salary<50000),

City\_Name Nvarchar(50) Not Null Default('Alwar')

)

insert into @MyData values (02,'AAA',5000,'KTG'),(03,'BBB',25000,'CNR')

begin transaction MY\_Trans

delete from @MyData where id=02

rollback transaction MY\_Trans

select\*from @MyData

**(Temporary Tables will support for Rollback Option)**

create table #tbl\_Temp\_tbl

(

Id int constraint Cont\_PK\_TempP primary key,

Name varchar (40),

Salary int check(Salary<50000),

City Varchar(52) not null default ('MAS')

)

insert into #tbl\_Temp\_tbl values (01,'AAA',25000,'CBE'),(05,'AAB',28000,CNR'),(06,'ACA',2000,'Mas')

begin transaction My\_trans

delete from #tbl\_Temp\_tbl where #tbl\_Temp\_tbl.id=05;

begin transaction My\_trans

Rollback transaction My\_trans

select \*from #tbl\_Temp\_tbl

# Trigger

alter trigger Tri\_Log on tbl\_new\_table

after insert ,delete,update

as begin

insert into tbl\_New\_Log\_Table select\*,CURRENT\_TIMESTAMP from inserted

insert into tbl\_New\_Log\_Table select\*,CURRENT\_TIMESTAMP from deleted

end

# CRUD\_STORED\_PROCEDURE

CREATE PROCEDURE Usp\_InsertUpdateDelete\_Customer

@CustomerID BIGINT = 0

,@Name NVARCHAR(100) = NULL

,@Mobileno NVARCHAR(15) = NULL

,@Address NVARCHAR(300) = 0

,@Birthdate DATETIME = NULL

,@EmailID NVARCHAR(15) = NULL

,@Query INT

AS

BEGIN

IF (@Query = 1)

BEGIN

INSERT INTO Customer(

NAME

,Address

,Mobileno

,Birthdate

,EmailID

)

VALUES (

@Name

,@Address

,@Mobileno

,@Birthdate

,@EmailID

)

IF (@@ROWCOUNT > 0)

BEGIN

SELECT 'Insert'

END

END

IF (@Query = 2)

BEGIN

UPDATE Customer

SET NAME = @Name

,Address = @Address

,Mobileno = @Mobileno

,Birthdate = @Birthdate

,EmailID = @EmailID

WHERE Customer.CustomerID = @CustomerID

SELECT 'Update'

END

IF (@Query = 3)

BEGIN

DELETE

FROM Customer

WHERE Customer.CustomerID = @CustomerID

SELECT 'Deleted'

END

IF (@Query = 4)

BEGIN

SELECT \*

FROM Customer

END

END

IF (@Query = 5)

BEGIN

SELECT \*

FROM Customer

WHERE Customer.CustomerID = @CustomerID

END

# String interpolation

<https://www.microsoft.com/net/learn/in-browser-tutorial/6>

String interpolation lets you piece together strings in a more concise and readable way.

If you add a $ before the opening quotes of the string, you can then include string values, like the name variable, inside the string in curly brackets. Try it out and select **Run Code**.

Var name=’’artr”

Console.WriteLine($"Hello {name}!");

# DATATABLE

**//if (datatable!=null)**

**//{**

**// for (int i =1; i < datatable.Rows.Count; i++)**

**// {**

**// List.Add**

**// (**

**// new UmRoleScreenPermissionAccessLevel**

**// {**

**// UserType = Convert.ToString(datatable.Rows[i]["Name"]),**

**// Role = Convert.ToString(datatable.Rows[i]["Rolename"]),**

**// Module = Convert.ToString(datatable.Rows[i]["ServiceName"]),**

**// Screen = Convert.ToString(datatable.Rows[i]["MenuName"]),**

**// Add = Convert.ToString(datatable.Rows[i]["Add"]),**

**// Edit = Convert.ToString(datatable.Rows[i]["Edit"]),**

**// View = Convert.ToString(datatable.Rows[i]["View"]),**

**// Delete = Convert.ToString(datatable.Rows[i]["Delete"]),**

**// Print = Convert.ToString(datatable.Rows[i]["Print"]),**

**// Export = Convert.ToString(datatable.Rows[i]["Export"]),**

**// Approve = Convert.ToString(datatable.Rows[i]["Approve"]),**

**// Reject = Convert.ToString(datatable.Rows[i]["Reject"]),**

**// Verify = Convert.ToString(datatable.Rows[i]["Verify"]),**

**// Clarify = Convert.ToString(datatable.Rows[i]["Clarify"]),**

**// Renew = Convert.ToString(datatable.Rows[i]["Renew"]),**

**// Acknowledge = Convert.ToString(datatable.Rows[i]["Acknowledge"]),**

**// Recommend = Convert.ToString(datatable.Rows[i]["Recommend"]),**

**// }**

**// );**

**// }**

**//foreach (DataRow row in datatable.Rows)**

**//{**

**// List.Add**

**// (**

**// new UmRoleScreenPermissionAccessLevel**

**// {**

**// UserType = Convert.ToString(row["Name"]),**

**// Role = Convert.ToString(row["Rolename"]),**

**// Module = Convert.ToString(row["ServiceName"]),**

**// Screen = Convert.ToString(row["MenuName"]),**

**// Add = Convert.ToString(row["Add"]),**

**// Edit = Convert.ToString(row["Edit"]),**

**// View = Convert.ToString(row["View"]),**

**// Delete = Convert.ToString(row["Delete"]),**

**// Print = Convert.ToString(row["Print"]),**

**// Export = Convert.ToString(row["Export"]),**

**// Approve = Convert.ToString(row["Approve"]),**

**// Reject = Convert.ToString(row["Reject"]),**

**// Verify = Convert.ToString(row["Verify"]),**

**// Clarify = Convert.ToString(row["Clarify"]),**

**// Renew = Convert.ToString(row["Renew"]),**

**// Acknowledge = Convert.ToString(row["Acknowledge"]),**

**// Recommend = Convert.ToString(row["Recommend"]),**

**// }**

**// );**

**//}**

**//var AsEnumerableTOList = datatable.AsEnumerable();**

**//var List = new List<UmRoleScreenPermissionAccessLevel>();**

**// List = (from item in AsEnumerableTOList**

**// select new UmRoleScreenPermissionAccessLevel**

**// {**

**// UserType = item.Field<string>("Name"),**

**// Role = item.Field<string>("Rolename"),**

**// Module = item.Field<string>("ServiceName"),**

**// Screen = item.Field<string>("MenuName"),**

**// Add = item.Field<string>("Add"),**

**// Edit = item.Field<string>("Edit"),**

**// View = item.Field<string>("View"),**

**// Delete = item.Field<string>("Delete"),**

**// Print = item.Field<string>("Print"),**

**// Export = item.Field<string>("Export"),**

**// Approve = item.Field<string>("Approve"),**

**// Reject = item.Field<string>("Reject"),**

**// Verify = item.Field<string>("Verify"),**

**// Clarify = item.Field<string>("Clarify"),**

**// Renew = item.Field<string>("Renew"),**

**// Acknowledge = item.Field<string>("Acknowledge"),**

**// Recommend = item.Field<string>("Recommend"),**

**// }).ToList();**

**//return List;**

**public DataTable Download()**

**{**

**var Conn = new SqlConnection();**

**DataTable datatable=null;**

**try**

**{**

**var context = new ASISWebDatabaseEntities();**

**//var ASISWebDatabaseEntitiesconString = context.Database.Connection.ConnectionString;**

**var connString =**

**(string.IsNullOrEmpty(Convert.ToString((ConfigurationManager.ConnectionStrings["ASISBIDatabaseEntities"]))) ?**

**(Convert.ToString((ConfigurationManager.ConnectionStrings["AsisReportDataBase"]))) :**

**(Convert.ToString((ConfigurationManager.ConnectionStrings["ASISBIDatabaseEntities"])))**

**);**

**connString = !string.IsNullOrEmpty(connString) ? Helper.DbHelper.SetADODBConnectionString(connString) : connString;**

**Conn.ConnectionString = connString;**

**Conn.Open();**

**SqlCommand command = new SqlCommand("UmRoleScreenPermissionExport", Conn);**

**command.CommandTimeout = 300;**

**command.CommandType = System.Data.CommandType.StoredProcedure;**

**using (SqlDataReader dr = command.ExecuteReader())**

**{**

**while (!dr.IsClosed)**

**{**

**datatable = new DataTable();**

**datatable.Load(dr);**

**}**

**}**

**return datatable;**

**}**

**catch (Exception ex)**

**{**

**throw ex;**

**}**

**finally**

**{**

**Conn.Close();**

**}**

**}**

# ConvertToExcelTable

public StringBuilder ConvertToExcelTable (DataTable sqlTable ,List<string>HeaderColumns)

{

try

{

var sb = new StringBuilder();

var table = new DataTable();

table = sqlTable;

sb.Append("<table border='1px' cellpadding='1' cellspacing='1'>");

sb.Append("<tr>");

foreach (string \_title in HeaderColumns)

{

var \_data = \_title.TrimStart('"');

\_data = \_data.TrimEnd('"');

sb.Append("<td align=\"center\">");

sb.Append(\_data);

sb.Append("</td>");

}

sb.Append("</tr>");

foreach (DataRow myRow in table.Rows)

{

var i = 1;

sb.Append("<tr>");

foreach (string ColumnData in myRow.ItemArray.Select(o => QuoteValue(o.ToString())))

{

var \_data = ColumnData.TrimStart('"');

\_data = \_data.TrimEnd('"');

sb.Append("<td style='mso-number-format:\\@'>");

sb.Append(\_data);

sb.Append("</td>");

i++;

if (i > HeaderColumns.Count) break;

}

sb.Append("</tr>");

}

sb.Append("</table>");

table.Dispose();

return sb;

}

catch (Exception ex)

{

throw ex;

}

}

static string QuoteValue(string value)

{

return String.Concat("\"",

value.Replace("\"", ""), "\"");

//return String.Concat("\"", value, "\"");

}

# ConvertIntoDataTable

public DataTable ConvertIntoDataTable<T>(List<T> List)

{

try

{

DataTable dataTable = new DataTable(typeof(T).Name);

//Get all the properties by using reflection

PropertyInfo[] Props = typeof(T).GetProperties(BindingFlags.Public | BindingFlags.Instance);

foreach (PropertyInfo prop in Props)

{

//Setting column names as Property names

dataTable.Columns.Add(prop.Name);

}

foreach (T item in List)

{

var values = new object[Props.Length];

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

{

values[i] = Props[i].GetValue(item, null);

}

dataTable.Rows.Add(values);

}

return dataTable;

}

catch (Exception ex)

{

throw ex;

}

}

# BuildHtmlTable

public string BuildHtmlTable(DataTable table)

{

try

{

var stringBuilder = new StringBuilder();

stringBuilder.Append("<table style='border: solid 1px black; border-collapse:collapse;' cellpadding='10'>");

stringBuilder.Append("<tr>");

foreach (DataColumn Column in table.Columns)

{

var \_data = Column.ColumnName.TrimStart('"');

\_data = \_data.TrimEnd('"');

stringBuilder.Append("<th style='border: solid 1px black;' align='center'>");

stringBuilder.Append(\_data);

stringBuilder.Append("</th>");

}

stringBuilder.Append("</tr>");

var columnCount = table.Columns.Count ;

foreach (DataRow myRow in table.Rows)

{

var i = 1;

stringBuilder.Append("<tr>");

foreach (string ColumnData in myRow.ItemArray.Select(o => QuoteValue(o.ToString())))

{

var \_data = ColumnData.TrimStart('"');

\_data = \_data.TrimEnd('"');

stringBuilder.Append("<td style='border: solid 1px black;'>");

stringBuilder.Append(\_data);

stringBuilder.Append("</td>");

i++;

if (i > columnCount) break;

}

stringBuilder.Append("</tr>");

}

stringBuilder.Append("</table>");

table.Dispose();

return stringBuilder.ToString();

}

catch (Exception ex)

{

throw ex;

}

}

$("#BindSummaryData").append(jsonpart.CopyEngPlannerTxnDets);

# DB Connection for ADO.net

public class DBConnection  
    {  
        public void DBConncected(Dictionary<string,string> parameters,string SPname, CommandType Type)  
        {  
            SqlParameter[] paramete= DictionaryToSqlParameterArray(parameters);  
            string connectionsring = System.Configuration.ConfigurationManager.ConnectionStrings["AsisReportDataBase"].ConnectionString;  
            using (var con = new SqlConnection(connectionsring))  
            {  
                con.Open();  
                using (var comm = this.CreateSQLCommand(con, Type, SPname, paramete))  
                {  
                    var ds = CreateDataSet(comm);  
                }  
            }  
        }  
        public SqlParameter[] DictionaryToSqlParameterArray(Dictionary<string, string> parameters)  
        {  
            var sqlParameterCollection = new List<SqlParameter>();  
            foreach (var parameter in parameters)  
            {  
                sqlParameterCollection.Add(new SqlParameter(parameter.Key, parameter.Value));  
            }  
            return sqlParameterCollection.ToArray();  
        }  
        public DataSet CreateDataSet(SqlCommand command)  
        {  
            DataSet ds = new DataSet();  
            try  
            {  
                using (var adp=new SqlDataAdapter(command))  
                {  
                    command.ExecuteNonQuery();  
                    adp.Fill(ds);  
                    command.Connection.Close();  
                    return ds;  
                }  
                      
            }  
            catch (Exception ex)  
            {  
                throw new Exception(ex.ToString());  
            }  
        }  
  
  
        public SqlCommand CreateSQLCommand(SqlConnection connection, CommandType commandType, string spName, params object[] values)  
        {  
            var command = new SqlCommand();  
            command.Connection = connection;  
            command.CommandText = spName;  
            command.CommandType = commandType;  
            if(commandType== CommandType.StoredProcedure)  
            {  
                if (values.Length > 0)  
                {  
                    foreach (var val in values)  
                    {  
                        command.Parameters.Add(val);  
                    }  
                }  
            }  
              
  
  
            return command;  
        }

    }

  Edit  Delete

# Linq for Dataset

DataSet ds = new DataSet();  
            DataTable dt1 = new DataTable("Table1");  
            DataTable dt2 = new DataTable("Table2");  
  
            dt1.Columns.Add("Id");  
            dt1.Columns.Add("Name");  
  
            DataRow ro = dt1.NewRow();  
            ro["Id"] = 1;  
            ro["Name"] = "Jana";  
            dt1.Rows.Add(ro);  
  
            dt2.Columns.Add("Id");  
            dt2.Columns.Add("Name");  
  
            DataRow ro1 = dt2.NewRow();  
            ro1["Id"] = 1;  
            ro1["Name"] = "kumar";  
            dt2.Rows.Add(ro1);  
            ds.Tables.Add(dt1);  
            ds.Tables.Add(dt2);  
            var quertable = ds.Tables["Table1"].AsEnumerable();  
            var query = (from c in ds.Tables["Table1"].AsEnumerable()  
                         join c1 in ds.Tables["Table2"].AsEnumerable()  
                         on c.Field<string>("Id") equals c1.Field<string>("Id")  
                         where c.Field<string>("Name").ToLower().Contains("a")  
                         select new  
                         {  
                             Name=c.Field<string>("Name")

                         }).ToList();

  Edit  Delete

# Decimal Handling

**--------------------------------------------------------------------------------------------------------------------**

**Not Helpful -**Math.Round(num,2)

**Helpful-** C# - Rounding off  2 number after the decimal point

**(Convert.ToDecimal(String.Format("{0:0.##}",(num))));**

**Not Helpful-** num.toFixed(2)

**Helpful-**JQuery -Taking 2 numbers without rounding off after the decimal point

**num.toString().match(/^-?\d+(?:\.\d{0,2})?/)[0];alert(with2Decimals);**

**--------------------------------------------------------------------------------------------------------------------**

  Edit  Delete

# Data Table to List 3 Ways (For,ForEach,Linq)

**--------------------------------------------------------------------------------------------------------------------**

                  for (int i =1; i < datatable.rows.count; i++)

                   {

                       list.add

                          (

                          new ExampleList

                          {

                              usertype = convert.tostring(datatable.rows[i]["name"]),

                              role = convert.tostring(datatable.rows[i]["rolename"]),

                              module = convert.tostring(datatable.rows[i]["servicename"]),

                              screen = convert.tostring(datatable.rows[i]["menuname"]),

                              add = convert.tostring(datatable.rows[i]["add"]),

                              edit = convert.tostring(datatable.rows[i]["edit"]),

                              view = convert.tostring(datatable.rows[i]["view"]),

                              delete = convert.tostring(datatable.rows[i]["delete"]),

                              print = convert.tostring(datatable.rows[i]["print"]),

                              export = convert.tostring(datatable.rows[i]["export"]),

                              approve = convert.tostring(datatable.rows[i]["approve"]),

                              reject = convert.tostring(datatable.rows[i]["reject"]),

                              verify = convert.tostring(datatable.rows[i]["verify"]),

                              clarify = convert.tostring(datatable.rows[i]["clarify"]),

                              renew = convert.tostring(datatable.rows[i]["renew"]),

                              acknowledge = convert.tostring(datatable.rows[i]["acknowledge"]),

                              recommend = convert.tostring(datatable.rows[i]["recommend"]),

                          }

                          );

                foreach (datarow row in datatable.rows)

                {

                   list.add

                       (

                       new ExampleList

                       {

                           usertype = convert.tostring(row["name"]),

                           role = convert.tostring(row["rolename"]),

                           module = convert.tostring(row["servicename"]),

                           screen = convert.tostring(row["menuname"]),

                           add = convert.tostring(row["add"]),

                           edit = convert.tostring(row["edit"]),

                           view = convert.tostring(row["view"]),

                           delete = convert.tostring(row["delete"]),

                           print = convert.tostring(row["print"]),

                           export = convert.tostring(row["export"]),

                           approve = convert.tostring(row["approve"]),

                           reject = convert.tostring(row["reject"]),

                           verify = convert.tostring(row["verify"]),

                           clarify = convert.tostring(row["clarify"]),

                           renew = convert.tostring(row["renew"]),

                           acknowledge = convert.tostring(row["acknowledge"]),

                           recommend = convert.tostring(row["recommend"]),

                       }

                       );

                }

var asenumerabletolist = datatable.asenumerable();

                var list = new list<ExampleList>();

                    list = (from item in asenumerabletolist

                           select new ExampleList

                                   {

                                       usertype = item.field<string>("name"),

                                       role = item.field<string>("rolename"),

                                       module = item.field<string>("servicename"),

                                       screen = item.field<string>("menuname"),

                                       add = item.field<string>("add"),

                                       edit = item.field<string>("edit"),

                                       view = item.field<string>("view"),

                                       delete = item.field<string>("delete"),

                                       print = item.field<string>("print"),

                                       export = item.field<string>("export"),

                                       approve = item.field<string>("approve"),

                                       reject = item.field<string>("reject"),

                                       verify = item.field<string>("verify"),

                                       clarify = item.field<string>("clarify"),

                                       renew = item.field<string>("renew"),

                                       acknowledge = item.field<string>("acknowledge"),

                                       recommend = item.field<string>("recommend"),

                                   }).tolist();

**-------------------------------------------------------------------------------------------------------------**

  Edit  Delete

# Dynamic List from DataTable

        private List<dynamic> ToDynamicList(DataTable dt)

        {

            try

            {

                var dns = new List<dynamic>();

                foreach (var item in dt.AsEnumerable())

                {

                    // Expando objects are IDictionary<string, object>

                    IDictionary<string, object> dn = new ExpandoObject();

                    foreach (var column in dt.Columns.Cast<DataColumn>())

                    {

                        dn[column.ColumnName] = item[column];

                    }

                    dns.Add(dn);

                }

                return dns;

            }

            catch (Exception ex)

            {

                throw;

            }

        }

  Edit  Delete

# Event Key Codes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event Key Codes** | |  | |  | |
| *Key* | *Code* | *Key* | *Code* | *Key* | *Code* |
| *backspace* | *8* | *e* | *69* | *numpad 8* | *104* |
| *tab* | *9* | *f* | *70* | *numpad 9* | *105* |
| *enter* | *13* | *g* | *71* | *multiply* | *106* |
| *shift* | *16* | *h* | *72* | *add* | *107* |
| *ctrl* | *17* | *i* | *73* | *subtract* | *109* |
| *alt* | *18* | *j* | *74* | *decimal point* | *110* |
| *pause/break* | *19* | *k* | *75* | *divide* | *111* |
| *caps lock* | *20* | *l* | *76* | *f1* | *112* |
| *escape* | *27* | *m* | *77* | *f2* | *113* |
| *(space)* | *32* | *n* | *78* | *f3* | *114* |
| *page up* | *33* | *o* | *79* | *f4* | *115* |
| *page down* | *34* | *p* | *80* | *f5* | *116* |
| *end* | *35* | *q* | *81* | *f6* | *117* |
| *home* | *36* | *r* | *82* | *f7* | *118* |
| *left arrow* | *37* | *s* | *83* | *f8* | *119* |
| *up arrow* | *38* | *t* | *84* | *f9* | *120* |
| *right arrow* | *39* | *u* | *85* | *f10* | *121* |
| *down arrow* | *40* | *v* | *86* | *f11* | *122* |
| *insert* | *45* | *w* | *87* | *f12* | *123* |
| *delete* | *46* | *x* | *88* | *num lock* | *144* |
| *0* | *48* | *y* | *89* | *scroll lock* | *145* |
| *1* | *49* | *z* | *90* | *semi-colon* | *186* |
| *2* | *50* | *left window key* | *91* | *equal sign* | *187* |
| *3* | *51* | *right window key* | *92* | *comma* | *188* |
| *4* | *52* | *select key* | *93* | *dash* | *189* |
| *5* | *53* | *numpad 0* | *96* | *period* | *190* |
| *6* | *54* | *numpad 1* | *97* | *forward slash* | *191* |
| *7* | *55* | *numpad 2* | *98* | *grave accent* | *192* |
| *8* | *56* | *numpad 3* | *99* | *open bracket* | *219* |
| *9* | *57* | *numpad 4* | *100* | *back slash* | *220* |
| *a* | *65* | *numpad 5* | *101* | *close braket* | *221* |
| *b* | *66* | *numpad 6* | *102* | *single quote* | *222* |
| *c* | *67* | *numpad 7* | *103* |  | |
| *d* | *68* |  | |  | |

# Inspect Element Restrction

----------------------------------------------------------------------------------------------------------------------------

    window.onload = function ()

    {

        var promiseGet = LayoutService.ASISDevRestriction();

        promiseGet.then(function (response)

        {

           var AllowDevTool = response.data;

            if (AllowDevTool == 1)

            {

                document.addEventListener("contextmenu", function (e) {

                    e.preventDefault();

                }, false);

                document.addEventListener("keydown", function (e) {

                    // "I" key

                    if (e.ctrlKey && e.shiftKey && e.keyCode == 73) {

                        RestrictActions(e);

                    }

                    // "J" key

                    if (e.ctrlKey && e.shiftKey && e.keyCode == 74) {

                        RestrictActions(e);

                    }

                    // "S" key + macOS

                    if (e.keyCode == 83 && (navigator.platform.match("Mac") ? e.metaKey : e.ctrlKey)) {

                        RestrictActions(e);

                    }

                    // "U" key

                    if (e.ctrlKey && e.keyCode == 85) {

                        RestrictActions(e);

                    }

                    if (

                        event.keyCode == 112 ||  // "F1" key

                        event.keyCode == 113 ||  // "F2" key

                        event.keyCode == 114 ||  // "F3" key

                        event.keyCode == 115 ||  // "F4" key

                        //event.keyCode == 116 ||  // "F5" key

                        event.keyCode == 117 ||  // "F6" key

                        event.keyCode == 118 ||  // "F7" key

                        event.keyCode == 119 ||  // "F8" key

                        event.keyCode == 120 ||  // "F9" key

                        event.keyCode == 121 ||  // "F10" key

                        event.keyCode == 122 ||  // "F11" key

                        event.keyCode == 123     // "F12" key

                        ) {

                        RestrictActions(e);

                    }

                       //FireFox

        // "C" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 67)

        {

            RestrictActions(e);

        }

        // "K" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 75) {

            RestrictActions(e);

        }

        // "E" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 69) {

            RestrictActions(e);

        }

        // "J" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 74) {

            RestrictActions(e);

        }

        // "M" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 77) {

            RestrictActions(e);

        }

        // "Sh+F7" key

        if (e.shiftKey && e.keyCode == 118) {

            RestrictActions(e);

        }

        // "Sh+F5" key

        if ( e.shiftKey && e.keyCode == 73) {

            RestrictActions(e);

        }

        // "Sh+F9" key

        if ( e.shiftKey && e.keyCode == 120) {

            RestrictActions(e);

        }

        // "Sh+F12" key

        if ( e.shiftKey && e.keyCode == 123) {

            RestrictActions(e);

        }

        // "Sh+F8" key

        if ( e.shiftKey && e.keyCode == 119) {

            RestrictActions(e);

        }

        // "Sh+F4" key

        if (e.shiftKey && e.keyCode == 115) {

            RestrictActions(e);

        }

        // "Ct+U" key

        if (e.ctrlKey && e.keyCode == 85) {

            RestrictActions(e);

        }

                }, false);

            }

                  },

             function (errorP2) {

                 console.log(errorP2);

             });

    };

    document.addEventListener("keydown", function (e) {

        // "I" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 73) {

            RestrictActions(e);

        }

        // "J" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 74) {

            RestrictActions(e);

        }

        // "S" key + macOS

        if (e.keyCode == 83 && (navigator.platform.match("Mac") ? e.metaKey : e.ctrlKey)) {

            RestrictActions(e);

        }

        // "U" key

        if (e.ctrlKey && e.keyCode == 85) {

            RestrictActions(e);

        }

        if (

            //Chrome

            event.keyCode == 112 ||     // "F1" key

            event.keyCode == 113 ||     // "F2" key

            event.keyCode == 114 ||     // "F3" key

            event.keyCode == 115 ||     // "F4" key

            //event.keyCode == 116 ||   // "F5" key

            event.keyCode == 117 ||     // "F6" key

            event.keyCode == 118 ||     // "F7" key

            event.keyCode == 119 ||     // "F8" key

            event.keyCode == 120 ||     // "F9" key

            event.keyCode == 121 ||     // "F10" key

            event.keyCode == 122 ||     // "F11" key

            event.keyCode == 123        // "F12" key

            ) {

            RestrictActions(e);

        }

        //FireFox

        // "C" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 67)

        {

            RestrictActions(e);

        }

        // "K" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 75) {

            RestrictActions(e);

        }

        // "E" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 69) {

            RestrictActions(e);

        }

        // "J" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 74) {

            RestrictActions(e);

        }

        // "M" key

        if (e.ctrlKey && e.shiftKey && e.keyCode == 77) {

            RestrictActions(e);

        }

        // "Sh+F7" key

        if (e.shiftKey && e.keyCode == 118) {

            RestrictActions(e);

        }

        // "Sh+F5" key

        if ( e.shiftKey && e.keyCode == 73) {

            RestrictActions(e);

        }

        // "Sh+F9" key

        if ( e.shiftKey && e.keyCode == 120) {

            RestrictActions(e);

        }

        // "Sh+F12" key

        if ( e.shiftKey && e.keyCode == 123) {

            RestrictActions(e);

        }

        // "Sh+F8" key

        if ( e.shiftKey && e.keyCode == 119) {

            RestrictActions(e);

        }

        // "Sh+F4" key

        if (e.shiftKey && e.keyCode == 115) {

            RestrictActions(e);

        }

        // "Ct+U" key

        if (e.ctrlKey && e.keyCode == 85) {

            RestrictActions(e);

        }

    }, false);

    function RestrictActions(e) {

        if (e.stopPropagation) {

            e.stopPropagation();

        } else if (window.event) {

            window.event.cancelBubble = true;

        }

        e.preventDefault();

        return false;

    }

----------------------------------------------------------------------------------------------------------------------------

  Edit

# JQUERY-Tab Selection

$('.nav-tabs a[href="#' + 'DivCorrection' + '"]').tab('show');

# JQUERY- setInterval

setInterval(function () {

$scope.ReminderandAlertPopUp()

//code goes here that will be run every 5 seconds.

}, 1);

Dynmaic ng-model

var SelectedDetIds = $scope.Details.TestingandCommissioningDetIds.split(",");

$scope.AssetPraNo = 'Selected Count :' + ' ' + SelectedDetIds.length;

$scope.CommonCollection = $scope.Details.EngTestingandCommissioningTxnDets;

// $scope.ProjectAssetPreRegistrationNos = $scope.PopUpPagination($scope.CommonCollection);

$scope.ProjectAssetPreRegistrationNos =$scope.CommonCollection;

for (var i = 0; i < SelectedDetIds.length; i++) {

var TestingandCommisioningDetid = parseInt(SelectedDetIds[i]);

var ngMOdel = "IsChecked" + TestingandCommisioningDetid;

$scope[ngMOdel] = true;

# Date to DAY for Month

var WcEntity = WcDetEntity == null ? WcDetEntity : WcDetEntity;

List<FMSWorkingCalendarDetEntity> ObjWCEntity = null;

var dayList = Enumerable.Range(Month, 1).Select(m =>

Enumerable.Range(1, DateTime.DaysInMonth(year, m)) // Days: 1, 2 ... 31 etc.

.Select(day => new FMSWorkingCalendarDetEntity

{

Day = new DateTime(year, m, day).Day,

DayName = new DateTime(year, m, day).DayOfWeek.ToString().Substring(0, 3),

WeekNo = CultureInfo.InvariantCulture.Calendar.GetWeekOfYear(new DateTime(year, m, day), System.Globalization.CalendarWeekRule.FirstDay, DayOfWeek.Monday),

}).ToList()).ToList(); // Map each day to a date

return dayList;

# SQL Data Types vs. C# Data Types

This article is just a reference of SQL Data Types to C# Data Types.

|  |  |  |
| --- | --- | --- |
| **SQL Server data type** | **CLR data type (SQL Server)** | **CLR data type (.NET Framework)** |
| varbinary | SqlBytes, SqlBinary | Byte[] |
| binary | SqlBytes, SqlBinary | Byte[] |
| varbinary(1), binary(1) | SqlBytes, SqlBinary | byte, Byte[] |
| image | None | None |
| varchar | None | None |
| char | None | None |
| nvarchar(1), nchar(1) | SqlChars, SqlString | Char, String, Char[] |
| nvarchar | SqlChars, SqlString | String, Char[] |
| nchar | SqlChars, SqlString | String, Char[] |
| text | None | None |
| ntext | None | None |
| uniqueidentifier | SqlGuid | Guid |
| rowversion | None | Byte[] |
| bit | SqlBoolean | Boolean |
| tinyint | SqlByte | Byte |
| smallint | SqlInt16 | Int16 |
| int | SqlInt32 | Int32 |
| bigint | SqlInt64 | Int64 |
| smallmoney | SqlMoney | Decimal |
| money | SqlMoney | Decimal |
| numeric | SqlDecimal | Decimal |
| decimal | SqlDecimal | Decimal |
| real | SqlSingle | Single |
| float | SqlDouble | Double |
| smalldatetime | SqlDateTime | DateTime |
| datetime | SqlDateTime | DateTime |
| sql\_variant | None | Object |
| User-defined type(UDT) | user-defined type | None |
| table | None | None |
| cursor | None | None |
| timestamp | None | None |
| xml | SqlXml | None |