public static bool ValidateString(string value)

{

bool retValue = true;

if (CheckLength(value))

{

retValue = Regex.IsMatch(value, @"^[\p{L}\p{Zs}\p{Lu}\p{Ll}\';?]{0,1000}$");

}

return retValue;

}

public static bool CheckLength(string value)

{

string check = (value == null) ? string.Empty : value.Trim();

return (check.Length != 0);

}

public static bool CheckLength(string value, int LowerBound)

{

string check = (value == null) ? string.Empty : value.Trim();

return (check.Length <= LowerBound);

}

public static bool CheckLength(string value, int LowerBound, int UpperBound)

{

string check = (value == null) ? string.Empty : value.Trim();

return !(check.Length < LowerBound && check.Length > UpperBound);

}

public static string CheckNull(object value)

{

return (value == null || value == DBNull.Value) ? string.Empty : value.ToString();

}

public static bool IsAlpha(string value)

{

Regex objAlphaPattern = new Regex("[^a-zA-Z]");

return !objAlphaPattern.IsMatch(value);

}

public static bool IsAlphaNumeric(string value)

{

Regex objAlphaNumericPattern = new Regex("[\\<\\>]");

return !objAlphaNumericPattern.IsMatch(value);

}

public static string TryToString(this object value)

{

//return (value == null || value == DBNull.Value) ? string.Empty : value.ToString();

return (value == null || value == DBNull.Value) ? null : value.ToString();

}