# 公共类方法

## 取得本机串口

/// </summary>

/// <returns></returns>

public static string[] GetHostCOM()

{

string[] sAllPort = null;

try

{

sAllPort = SerialPort.GetPortNames();

}

catch (Exception ex)

{

throw new Exception(ex.Message);

}

return sAllPort;

}

## 获取本地IP地址

/// </summary>

/// <returns></returns>

public static string GetHostIPAddress()

{

string strHostIPAddr;

strHostIPAddr = "";

System.Net.IPAddress[] addressList = Dns.GetHostEntry(Dns.GetHostName()).AddressList;

strHostIPAddr += addressList[addressList.Length - 1].ToString();

return strHostIPAddr;

}

## 判断对象是否是整型数

/// </summary>

/// <param name="obj"></param>

/// <returns></returns>

public static bool CheckObjectIsInteger(object obj)

{

obj = obj.ToString().Trim();

if (string.IsNullOrEmpty(obj.ToString()))

{

return false;

}

else

{

return Regex.IsMatch(obj.ToString(), @"^\d{1,}$");

}

}

## 验证对象是否是字节数组

/// </summary>

/// <param name="objValue"></param>

/// <returns></returns>

public static bool CheckObjIsByteArray(object objValue)

{

if (objValue is byte[])

{

return true;

}

else

{

return false;

}

}

## 验证身份证号码的格式

/// </summary>

/// <param name="IDNo"></param>

/// <returns></returns>

public static bool IsValidIdentityNo(string IdentityNo)

{

IdentityNo = IdentityNo.Trim();

if (string.IsNullOrEmpty(IdentityNo))

{

return false;

}

else

{

return Regex.IsMatch(IdentityNo, @"^(\d{14}(\d{1}|\d{4}|\d{3}[xy]))$");

}

}

## 防注入方法

/// </summary>

/// <param name="text"></param>

/// <returns></returns>

public static string InputText(string text)

{

text = text.Trim();

if (string.IsNullOrEmpty(text))

return string.Empty;

// 替换两个或两个以上的空格为一个空格

text = Regex.Replace(text, "[\\s]{2,}", " ");

// 替换<br>为转义字符“\n”

text = Regex.Replace(text, "(<[b|B][r|R]/\*>)+|(<[p|P](.|\\n)\*?>)", "\n");

// 替换"&nbsp;"为空格

text = Regex.Replace(text, "(\\s\*&[n|N][b|B][s|S][p|P];\\s\*)+", " ");

// 替换特殊字符

text = Regex.Replace(text, "<(.|\\n)\*?>", string.Empty);

// 替换单引号

text = text.Replace("'", "''");

return text;

}

## 生成指定个数的随机数

/// </summary>

/// <param name="codeCount"></param>

/// <returns></returns>

private string CreateRandomCode(int codeCount)

{

string allChar = "0,1,2,3,4,5,6,7,8,9";

string[] allCharArray = allChar.Split(',');

string randomCode = "";

int temp = -1;

Random rand = new Random();

for (int i = 0; i < codeCount; i++)

{

if (temp != -1)

{

rand = new Random(i \* temp \* ((int)DateTime.Now.Ticks));

}

int t = rand.Next(10);

if (temp == t)

{

return CreateRandomCode(codeCount);

}

temp = t;

randomCode += allCharArray[t];

}

return randomCode;

}

## 生成20位的主键数

/// </summary>如AB200807021438250001

/// <param name="str">主键字符串的前两位</param>

/// <param name="str">例如调用接口：PKey.CreateKeyStr("RO");返回RO打头的20位字符</param>

/// <returns></returns>

public static string CreateKeyStr(string str)

{

StringBuilder strKey = new StringBuilder();

Random rd = new Random(System.DateTime.Now.Millisecond);

string strDate;

string strRd;

str = str.ToUpper().Replace(" ", "") + "AA";

str = str.Substring(0, 2);

strKey.Append(str);

strDate = DateTime.Now.ToString("yyyyMMdd") + DateTime.Now.ToString("HH:mm:ss").Replace(":", "");

strKey.Append(strDate);

strRd = "0000" + rd.Next(1, 10000);

strRd = strRd.Substring(strRd.Length - 4, 4);

strKey.Append(strRd);

return strKey.ToString();

}

## 系统重启

/// </summary>

public static void Reset()

{

Application.ExitThread();

System.Threading.Thread thtmp = new System.Threading.Thread(new System.Threading.ParameterizedThreadStart(run));

object appName = Application.ExecutablePath;

//System.Threading.Thread.Sleep(2000);

thtmp.Start(appName);

}

private static void run(Object obj)

{

System.Diagnostics.Process ps = new System.Diagnostics.Process();

ps.StartInfo.FileName = obj.ToString();

ps.Start();

}

## 导出DataGridView数据 长城修改

/// <param name="dgv">DataGridView控件</param>

public void outPutDataGridViewData(DataGridView dgv, string txtTitle)

{

SaveFileDialog save = new SaveFileDialog();

//save.Filter = "user files(" + filter + ")|" + filter";

save.Filter = "文本格式.txt|.txt|表格.xls|.xls|文档.doc|.doc|RLC格式.rlc|.rlc";

save.Title = "导出文件到";

if (save.ShowDialog() == DialogResult.OK)

{

string padStr = "";//用于填充空格

Stream myStream = save.OpenFile();

StreamWriter sw = new StreamWriter(myStream, System.Text.Encoding.GetEncoding("GB2312"));

try

{

int[] temp = new int[dgv.Columns.Count];

//for (int index = 0; index < dgv.Columns.Count; index++)

for (int index = 1; index < dgv.Columns.Count; index++)

{

temp[index] = GetMaxWidth(dgv, index);

}

//写标题

sw.WriteLine(txtTitle);

//写数据字段

string tempTitle = "";

//for (int i = 0; i < dgv.Columns.Count; i++)

for (int i = 1; i < dgv.Columns.Count; i++)

{

if (save.FilterIndex == 4)//如果是表格xls格式

{

//if (i > 0)

if (i > 1)

{

tempTitle += "\t";

}

tempTitle += dgv.Columns[i].Name;

}

else//如果是doc、txt等格式

{

//if (i == 0)

if (i == 1)

{

tempTitle += dgv.Columns[i].Name;

}

else

{

if (temp[i - 1] - Encoding.Default.GetBytes(dgv.Columns[i - 1].Name).Length < 0)

{

tempTitle += padStr.PadRight(2, ' ');

}

else

{

tempTitle += padStr.PadRight(temp[i - 1] - Encoding.Default.GetBytes(dgv.Columns[i - 1].Name).Length + 2, ' ');

}

tempTitle += dgv.Columns[i].Name;

}

}

}

sw.WriteLine(tempTitle);

//循环写内容

for (int j = 0; j < dgv.Rows.Count; j++)

{

string tempStr = "";

//for (int k = 0; k < dgv.Columns.Count; k++)

for (int k = 1; k < dgv.Columns.Count; k++)

{

if (save.FilterIndex == 4)//如果是表格xls格式

{

//if (k > 0)

if (k > 1)

{

tempStr += "\t";

}

tempStr += "'" + dgv.Rows[j].Cells[k].Value.ToString().Trim();

}

else //如果是doc、txt等格式

{

//if (k == 0)

if (k == 1)

{

tempStr += dgv.Rows[j].Cells[k].Value.ToString();

}

else

{

if (temp[k - 1] - Encoding.Default.GetBytes(dgv.Columns[k - 1].Name).Length < 0)

{

tempStr += padStr.PadRight(Encoding.Default.GetBytes(dgv.Columns[k - 1].Name).Length - Encoding.Default.GetBytes(dgv.Rows[j].Cells[k - 1].Value.ToString().Trim()).Length + 2, ' ');

}

else

{

tempStr += padStr.PadRight(temp[k - 1] - Encoding.Default.GetBytes(dgv.Rows[j].Cells[k - 1].Value.ToString().Trim()).Length + 2, ' ');

}

tempStr += dgv.Rows[j].Cells[k].Value.ToString().Trim();

}

}

}

sw.WriteLine(tempStr);

}

MessageBox.Show("导出数据成功！", "提示", MessageBoxButtons.OK, MessageBoxIcon.Information);

sw.Close();

myStream.Close();

}

catch (Exception ee)

{

MessageBox.Show(ee.Message);

}

finally

{

sw.Close();

myStream.Close();

}

}

}

/// 配合上面方法得到DataGridView每列最大的宽度

/// </summary>

/// <returns></returns>

private int GetMaxWidth(DataGridView dgv, int columnIndex)

{

int max = 0;

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

{

max = max > Encoding.Default.GetBytes(dgv.Rows[i].Cells[columnIndex].Value.ToString().Trim()).Length ? max : Encoding.Default.GetBytes(dgv.Rows[i].Cells[columnIndex].Value.ToString().Trim()).Length;

}

return max;

}

}