SKBL – Serial Key Builder Library

~~~~~~

Build on Visual Studio 2008 with Visual Basic
Based on .NET Framework 3.5

OVERVIEW:
~~~~~~

What is SKBL.

SKBL is a library (dll) that works similar to the Regex in .NET Framework (included by using: System.Text.RegularExpressions). In SKBL you have two words that are import in order to understand how the whole library works. First of all, there is a TEMPLATE, which's the pattern all keys should follow. With other words, this is the similarity to Regex. The second word is a KEY, which's the output of a template.
A template might be defined as a rule that keys should follow in order to be valid. More information about this on page 2-3.

SKBL is so simple that it only requires two functions, creating a key, and validating a key. Both this functions require a template.

The SKBL itself might be used to protect a .NET application. It works both in Visual Basic, C#, and almost all other .NET based applications. It might also be used to generate random strings, numbers, et cetera. If a text should follow some rule than SKBL can be used.

HISTORY:
SKBL was developed by Artem Los 2009. It should be an add-on for Serial Key Builder. The design of previous version of Serial Key Builder was similar to anther application, so the only place it can be found is in Artem's computer. Then, another design was build, but there is only Alpha version available. SKBL is actually end-of-support, no support, or no changes will be made in this library. It is as it was 2009. The current perspective of CliZ Ware is to release SKGL, Serial Key Generating Library, where you can save data directly into a key. This is the first, and the last tutorial written for SKBL. The library itself will be available on internet, though. If there is some usage of this library, if you find it useful, please send a message to me at artem.los@nilssons.ws. I will try to help as much as I can, for free!

Good Luck,
Artem (www.clizware.net)

Simple Template

In SKBL there are symbols that often define something. The table below defines each symbol included in SKBL.

Symbol	Function
#	A random number
*	A random uppercase letter
@	A random lowercase letter
%	A random lowercase or uppercase letter
?	A random number or uppercase letter
!	A random number or lowercase letter

And here are the outputs of few examples

Template	Output
#####	86475
*****-@@@@@	MXDNJ-nyeky
%%%%%-?????	EHDtw-JQ64B
!!!!!	t2p6n
REG-#####-@@@@@-*****	REG-84210-wfwpd-OTHNB

In the table above, all letters/digits are randomly generated. There is a way how the frequency of them might be changed. The function is called Random To. With this function you might specify the frequency of either letters or numbers. [15] is a simple function. It starts with "[", which indicates a function for the SKBL. The output of this function will be either 1,2,3,4 or 5. Number 1 is where to start and number 5 is where to end. This function also works for letters, both uppercase and lowercase. [DF] will output either D,E, or F, uppercase. [df] will output the same letters, but, in lowercase, which means, d,e,f.
Below, there is an example.

Template: #####-[13][59][46][79][03]-[DF][QS][MP][AF][TZ]-
[dt][er][os][no][eg]
Key: 08124-29692-DSPCW-ngsof

If a char is not assigned to do something, it will output as it is, without some changes. "[" and "]" are an exception.

Advance Modular protection

In SKBL there are two functions that can be used to make a strong template. Its name is Rem. The main principle of both of them is that there are taking one or two letters/digits in the key, and divides them with another letter/digit. It takes the reminder of them after the division and writes it down. The result will only be one number.

The first version of the function takes char X and divides it with Y. Note, the X stands for the coordinates, position, of the char that will be divided, and Y stays for the denominator.
If there is a template with five random numbers, #####, the output will be similar to 32042. If the next part of the key contains [1/7], the template might be #####[1/7]. The output is therefore 602885, or 748346, etc. The function takes the first char (1) and divides it by 7, and the rest is then displayed. To sum up, the formula of this function is [X/Y]

The second version of this function takes two chars and divides them by the third one. The formula is [+X,Y/Z]. For example, if the template is #####[+1,2/9], the output will be similar to 568732.

NOTE: In both of this functions the X (first function) or X and Y (second function) have to be less than their own position. With other words, the function cannot use a char that is after it. The SKBL reads the template from left to right; therefore it should be less than its own position.

#####-[3/7] or #####-[+2,5/9] are right; #####-[7/5] is wrong. So, the char position has to be less than the actual position of function.

NOTE: Either Y (first function) or Z (second function) cannot be greater than 10, i.e. function 1: Y<10; function 2: Z<10; It can only be divided with {1,2,3,4,5,6,7,8,9}

To understand the process is knowledge of ASCII numbers required.
If the template is #[1/5], the key will look like 31. SKBL takes 3 (ascii:51) and divides by 5. The rest is therefore 1. An ASCII table might, for instance, be found at http://www.asciitable.com/.

It differs from function two. If the template is ##[+1,2/6], the key might be 633. SKBL takes the value as it is i.e. 6 and 3, adds them up, and divides by 6, and outputs the rest 3. It also works with letters as well. The method is based on the previous one, but, contains few changes.

Summary

This is the summary of p3-4.

DEFINITION:

Symbol	Function
#	A random number
*	A random uppercase letter
@	A random lowercase letter
%	A random lowercase or uppercase letter
?	A random number or uppercase letter
!	A random number or lowercase letter

FUNCTIONS:

ID	Definition:	Description:	Example:	Note:
1	[XY]	Generates a random letter/char from X-Y	[AC] = A,B,C [ac] = a,b,c [35] = 3,4,5	The Y ASCII value cannot be less than X ASCII value
2	[XY]	Takes char X mod number Y	#[1/7] = 21 1 is reminder of ascii 50(2)	X cannot be a char after this function
3	[+X,Y/Z]	Adds chars X and Y Mo number Y.	##[+1,2/7]=156 6 is reminder of 1+5=6	X,Y cannot be a char after this function

Adding DLL to project

To use this Dynamic link library in a project, it has to be included as Reference.
In Visual Studio, go to Project>Add reference...>Browse. Enter the path of the library and press OK.

There are several ways to generate keys in SKBL (based on one). If only one key is to be generated, type:

```
Visual Basic  SKBL_Library.SKBL.Create_Key("tempate...")
C Sharp(C#)   SKBL_Library.SKBL.Create_Key("tempate...");
```

It is strongly recommended that you use a Try Catch in both languages.
VISUAL BASIC:

```
Try
   TextBox1.Text = SKBL_Library.SKBL.Create_Key("template...")
Catch ex As Exception
   MessageBox.Show(ex.Message)
'Showing a message if template or anything fails.
End Try
```

C SHARP (C#):

```
try
{
   TextBox1.Text = SKBL_Library.SKBL.Create_Key("template...");
}
catch (Exception e2)
{
   MessageBox.Show(e2.Message);
   //Showing a message if template or anything fails.
}
```

Another way shows how to generate more than one key
VISUAL BASIC:

```
Try
   Dim SplitChar As String = vbCrLf 'Change this variable if you want to
change the split char between keys
   Dim Keys As String = ""
   Dim i As Integer = 0
   While i < NumericUpDown1.Value
   i += 1 'Add 1 to "i"
   Keys = Keys & SKBL_Library.SKBL.Create_Key(TextBox1.Text) 'Here we add
key to variabel "Keys"

      If i < NumericUpDown1.Value Then
         Keys = Keys & SplitChar
      End If
   End While
   MessageBox.Show(Keys) 'Show all keys in a message
Catch ex As Exception
   MessageBox.Show(ex.Message) 'Show a message if template or anything is
corrupt
End Try
```

C SHARP (C#):

```
try
{
   String SplitChar = Environment.NewLine; //Change this variable if you
want to change the split char between keys
   String Keys="";
   int i=0;
   while (i < NumericUpDown1.Value)
   {
      i++; //Add 1 to "i"
      Keys = Keys + SKBL_Library.SKBL.Create_Key(TextBox1.Text); //Here we
add key to variabel "Keys"

      if (i < NumericUpDown1.Value)
      {
         Keys = Keys + SplitChar;
      }
   }
   MessageBox.Show(Keys); //Show all keys in a message
}
catch (Exception e2)
{
   MessageBox.Show(e2.Message); //Show a message if template or anything is
corrupt
}
```