

Forum:[Hack Forums](#)

Thread: [INI Class](#)

KingDeath360 - February 1, 2015, 8:11 am

This is a lightweight INI class for reading and writing INI data in VB .NET that can also support reading and writing the INI data in memory so there no need to load from disk.

Version 1.0 Created By Ben Jones.

INI Members

Spoiler ([Click to View](#))

EraseKey (Erase a key from a selection)

EraseSection (Erase a selection and it's key values)

LoadFromFile (Loads INI data from a filename)

LoadFromString (Loads INI data from a string)

KeyExists (Check if a key exists in a selection)

ReadSelections (Returns a list of selection names)

ReadSectionValues (Returns a list of key names from a selection)

ReadString (Returns a value from a key name in selection)

RenameKey (Renames a key in a selection)

RenameSelection (Renames a selection)

SelectionExsits (Checks that a selection exists)

SaveToFile (Saves INI contents to a file)

ToString (Returns the INI Raw data)

Update (Updates the INI data)

ValueExsits (Checks that a value exists in a selection)

WriteString (Writes a string to the INI data)

Code:

Code 'Memory ini file class for VB.NET 2010

'Note keys and group names are case sensitive. Comments are also not supported as yet.

Option Explicit On

Imports System.Text

Imports System.IO

Public Class memIni

Private IniData As List(Of String)

Private groups As List(Of String)

'INI data consts

Private Const SplitChar As Char = "•"

Private Const AssignChar As Char = "="

Private Const groupStart As Char = "["

Private Const groupEnd As Char = "]"

Private Const IDX_ERROR As Integer = -1

```

Public Function EraseKey(ByVal group As String, ByVal Key As String) As Boolean
    Dim SelName As String = groupStart & group & groupEnd
    Dim Found As Boolean = False
    Dim TempData As New List(Of String)

    'Return true or false if group found.
    For Each Line In IniData
        'get index of split char,
        Dim Idx As Integer = Line.IndexOf(SplitChar)
        'Get group name.
        Dim sName As String = Line.Substring(0, Idx)

        'Check we found group.
        If sName.Equals(SelName) Then
            'Remove group name from string.
            Line = Line.Remove(0, Idx + 1)

            If Not Line.Split(AssignChar)(0).Equals(Key) Then
                'Build the line.
                Line = SelName & SplitChar & Line
                'Add to list
                Call TempData.Add(Line)
            Else
                Found = True
            End If
        Else
            'Add to list.
            Call TempData.Add(Line)
        End If
    Next

    'Set new list data.
    IniData = TempData

    'Clear up
    TempData = Nothing

    Return Found
End Function

```

```

Public Function EraseSection(ByVal group As String) As Boolean
    Dim SelName As String = groupStart & group & groupEnd
    'Remove a group and all it's keys.
    Dim TempINI As New List(Of String)

    If Not groupExsits(group) Then
        Return False
    Else
        'If found remove group.
        For Each Line In IniData
            If Not Line.StartsWith(SelName) Then
                'Add data to new list.
            End If
        Next
    End If

```

```
        Call TempINI.Add(Line)
    End If
Next
End If
```

```
'Set new list data.
IniData = TempINI
Return True
```

```
'Clear up
TempINI = Nothing
```

```
End Function
```

```
Public Sub LoadFromFile(ByVal Filename As String)
    Dim sr As StreamReader
    'Load ini data from a filename.
    Try
        sr = New StreamReader(Filename)
        'Read the whole file.
        Dim Buffer As String = sr.ReadToEnd
        'Load ini data.
        Call LoadFromString(Buffer)
        'Close file.
        Call sr.Close()
        Catch ex As Exception
            Throw New FileNotFoundException
        End Try
End Sub
```

```
Public Sub LoadFromString(ByVal Source As String)
    Dim Lines() As String = Source.Split(vbLf)
    Dim Temp As String = vbNullString
    Dim sSel As String = vbNullString
    Dim Idx As Integer = IDX_ERROR

    'Create collections
    IniData = New List(Of String)

    For Each Line In Lines
        Temp = Line.Trim()

        If (Temp.Length > 0) Then
            'Check for selectin start and end brackets.
            If Temp.StartsWith(groupStart) And Temp.EndsWith(groupEnd) Then
                sSel = Temp
            Else
                Call IniData.Add(sSel & SplitChar & Temp)
            End If
        End If
    Next
End Sub
```

```

Public Function KeyExists(ByVal group As String, ByVal Key As String) As Boolean
    Dim SelName As String = groupStart & group & groupEnd
    Dim Found As Boolean = False

    'Return true if a key is found in a group.
    For Each Line In IniData
        'Get position of split char.
        Dim idx As Integer = Line.IndexOf(SplitChar)
        If (idx <> IDX_ERROR) Then
            'Get group name.
            Dim sName As String = Line.Substring(0, idx)
            'Remove group name from string.
            Line = Line.Remove(0, idx + 1)

            'Check if we have a group match.
            If sName.Equals(SelName) Then
                'Check if we got a match on the keyname.
                If Line.Split(AssignChar)(0).TrimEnd().Equals(Key) Then
                    Found = True
                    Exit For
                End If
            End If
        End If
    Next

    Return Found
End Function

```

```

Public Sub Readgroups(ByVal groups As List(Of String))
    'Return a list of group names.
    For Each Line In IniData
        'Get index of split char.
        Dim idx As Integer = Line.IndexOf(SplitChar)

        If (idx <> IDX_ERROR) Then
            Dim sName As String = Line.Substring(1, idx - 2)

            'Check if group is found.
            If Not groups.Contains(sName) Then
                'Add the groups
                Call groups.Add(sName)
            End If
        End If
    Next
End Sub

```

```

Public Sub ReadSectionValues(ByVal group As String, ByVal Values As List(Of String))
    Dim SelName As String = groupStart & group & groupEnd

    'Return a list of key names from group.
    For Each Line In IniData
        'Get index of split char.

```

```
Dim idx As Integer = Line.IndexOf(SplitChar)
```

```
If (idx <> IDX_ERROR) Then
```

```
    Dim sName As String = Line.Substring(0, idx)
```

```
    'Check for group name.
```

```
    If sName.Equals(SelName) Then
```

```
        'Remove group name from string.
```

```
        Line = Line.Remove(0, idx + 1)
```

```
        'Get key names
```

```
        Call Values.Add(Line.Split(AssignChar)(0).TrimStart())
```

```
    End If
```

```
End If
```

```
Next
```

```
End Sub
```

```
Public Function ReadString(ByVal group As String, ByVal Key As String, Optional ByVal iDefault  
As String = vbNullString) As String
```

```
    Dim SelName As String = groupStart & group & groupEnd
```

```
    Dim sValue As String = vbNullString
```

```
For Each Line In IniData
```

```
    'Get position of split char.
```

```
    Dim idx As Integer = Line.IndexOf(SplitChar)
```

```
    If (idx <> IDX_ERROR) Then
```

```
        'Get group name.
```

```
        Dim sName As String = Line.Substring(0, idx)
```

```
        'Remove group name from string.
```

```
        Line = Line.Remove(0, idx + 1)
```

```
        'Check if we have a group match.
```

```
        If sName.Equals(SelName) Then
```

```
            'Check if we got a match on the keyname.
```

```
            If Line.Split(AssignChar)(0).Trim().Equals(Key) Then
```

```
                'Get value
```

```
                sValue = Line.Split(AssignChar)(1).TrimStart()
```

```
            Exit For
```

```
        End If
```

```
    End If
```

```
End If
```

```
Next
```

```
If String.IsNullOrEmpty(sValue) Then
```

```
    'Return default if no string is found.
```

```
    Return iDefault
```

```
Else
```

```
    'Return found value.
```

```
    Return sValue
```

```
End If
```

```
End Function
```

```
Public Function RenameKey(ByVal group As String, ByVal OldKey As String, ByVal NewKey As  
String) As Boolean
```

```
Dim SelName As String = groupStart & group & groupEnd
Dim LinIdx As Integer = 0
Dim Found As Boolean = False
Dim Line As String = vbNullString
```

```
'Check if selecion is found.
If Not groupExsits(group) Then
Return Found
End If
```

```
For Each Line In IniData
'get index of split char,
Dim Idx As Integer = Line.IndexOf(SplitChar)
'Get group name.
Dim sName As String = Line.Substring(0, Idx)
If (Idx <> IDX_ERROR) Then
'Check we found group.
If sName.Equals(SelName) Then
'Remove group name from string.
Line = Line.Remove(0, Idx + 1)
'Check for old key
If (Line.Split(AssignChar)(0)).Trim().Equals(OldKey) Then
Exit For
End If
End If
End If
'INC line counter.
LinIdx += 1
Next
```

```
Try
Found = True
'Set new ini data.
IniData(LinIdx) = SelName & SplitChar & NewKey & AssignChar & Line.Split(AssignChar)(1)
Catch ex As Exception
Found = False
End Try
```

```
Return Found
End Function
```

```
Public Function Renamegroup(ByVal OldName As String, ByVal NewName As String) As
Boolean
```

```
'Rename a group.
Dim SelName As String = groupStart & OldName & groupEnd
Dim TempData As New List(Of String)
```

```
'Check for group.
If Not groupExsits(OldName) Then
Return False
Else
For Each Line In IniData
Dim idx As Integer = Line.IndexOf(SplitChar)
```

```

'Check for split char.
If (idx <> IDX_ERROR) Then
'Check for old selecion.
If Line.StartsWith(SelName) Then
'Remove group name.
Line = Line.Remove(0, idx)
'Rename group.
Line = groupStart & NewName & groupEnd & Line
End If
End If
'Store new data
Call TempData.Add(Line)
Next
End If

'Set new ini data.
IniData = TempData
Return True

'Clear up
TempData = Nothing

End Function

Public Function groupExsits(ByVal group As String) As Boolean
Dim SelName As String = groupStart & group & groupEnd
Dim Found As Boolean = False

If IsNothing(IniData) Then
Return False
End If

'Return true or false if group found.
For Each Line In IniData
If Line.StartsWith(SelName) Then
'group found exit for.
Found = True
Exit For
End If
Next

Return Found

End Function

Public Function SaveToFile(ByVal Filename As String)
Dim Done As Boolean = True
'Save ini data to filename.
Dim sw As StreamWriter

Try
'Create new stream writer.
sw = New StreamWriter(Filename)

```

```
'Write contents to file.  
Call sw.Write(ToString)  
'Close filename.  
Call sw.Close()  
Catch ex As Exception  
Done = False  
End Try
```

```
Return Done
```

```
End Function
```

```
Public Overrides Function ToString() As String  
Dim sb As New StringBuilder()
```

```
'If list not created create it.  
If IsNothing(groups) Then  
Call Update()  
End If
```

```
For Each Item In groups  
Call sb.AppendLine(Item)  
Next
```

```
Return sb.ToString()
```

```
End Function
```

```
Public Sub Update()  
'Rebuild the ini data.  
Dim sName As String = vbNullString  
Dim idx As Integer = IDX_ERROR
```

```
'Create groups list.  
groups = New List(Of String)
```

```
If IsNothing(IniData) Then  
Exit Sub  
End If
```

```
For Each Item In IniData  
Dim sPos As Integer = Item.IndexOf(SplitChar)
```

```
If (sPos > 0) Then  
'Get group.  
sName = Item.Substring(0, sPos)  
  
'Check if group is found.  
If Not groups.Contains(sName) Then  
'Add the groups  
Call groups.Add(sName)  
End If
```

```

'Fix item line.
Item = Item.Remove(0, sPos + 1)
'Check if group in groups
idx = groups.IndexOf(sName)

If (idx <> IDX_ERROR) Then
    Call groups.Insert(idx + 1, Item)
End If
End If
Next
End Sub

```

```

Public Function ValueExsits(ByVal group As String, ByVal Value As String) As Boolean
    Dim SelName As String = groupStart & group & groupEnd
    Dim Found As Boolean = False

```

```

'Check if group exsits.
If Not groupExsits(group) Then
    Return Found
End If

```

```

'Return true if a key is found in a group.
For Each Line In IniData
    'Get position of split char.
    Dim idx As Integer = Line.IndexOf(SplitChar)
    If (idx <> IDX_ERROR) Then
        'Get group name.
        Dim sName As String = Line.Substring(0, idx)
        'Remove group name from string.
        Line = Line.Remove(0, idx + 1)

        'Check if we have a group match.
        If sName.Equals(SelName) Then
            'Check if we got a match on the keyname.
            If Line.Split(AssignChar)(1).TrimEnd().Equals(Value) Then
                Found = True
                Exit For
            End If
        End If
    End If
End For
Next

```

```

    Return Found
End Function

```

```

Public Sub WriteString(ByVal group As String, ByVal Key As String, ByVal Value As String)
    Dim SelName As String = groupStart & group & groupEnd
    Dim ItemPos As Integer = 0

    'If item list is not created create it.
    If (IsNothing(IniData)) Then
        IniData = New List(Of String)
    End If

```

```

For Each Line In IniData
'Get position of split char.
Dim idx As String = Line.IndexOf(SplitChar)
If (idx <> IDX_ERROR) Then
'Get group name.
Dim sName As String = Line.Substring(0, idx)

'Remove group name from string.
Line = Line.Remove(0, idx + 1)

'Check if we have a group match.
If sName.Equals(SelName) Then
'Check if we got a match on the keyname.
If Line.Split(AssignChar)(0).Equals(Key) Then
Exit For
End If
End If
End If
'Inc counter
ItemPos += 1
Next

Try
'Try updateing the item.
IniData(ItemPos) = SelName & SplitChar & Key & AssignChar & Value
Catch ex As Exception
'Add new item.
Call IniData.Add(SelName & SplitChar & Key & AssignChar & Value)
End Try

End Sub

```

End Class

Reading Usage:

```
Code Dim i As New INIcreator.memIni
```

```
    i.LoadFromFile("testing.txt")
```

```
    MsgBox(i.ReadString("grouphere", "keyhere"))Writing Usage:
```

```
Code Dim i As New INIcreator.memIni
```

```
    i.LoadFromFile("testing.txt")
```

```
    i.WriteString("grouphere", "keyhere", "valuehere")
```

sh@rp - February 1, 2015, 8:38 am

Why reinvent the wheel o_O....

You do prefer using P/Invoke functions : <http://pastebin.com/91jwdLbL>

Nikey646 - February 1, 2015, 9:04 am

(Feb 1, 2015 09:38 AM)Ch@rLi3 (3D?tD3v) Wrote: [\[CITE\]](#) Why reinvent the wheel o_O....

You do prefer using P/Invoke functions : <http://pastebin.com/91jwdLbL>

Cross Platform Support > P/Invoke.

Why cut off support for any OS besides those w/ access to the Kernal32.dll for Windows.

SIM-1456245 - February 1, 2015, 12:50 pm

(Feb 1, 2015 10:04 AM)Nikey646 Wrote: [\[CITE\]](#) Cross Platform Support > P/Invoke.

Why cut off support for any OS besides those w/ access to the Kernal32.dll for Windows.
Wait, what? Cross-Platform with VB.Net? O_o

Nikey646 - February 1, 2015, 1:16 pm

(Feb 1, 2015 01:50 PM)Electroalek Wrote: [\[CITE\]](#) Wait, what? Cross-Platform with VB.Net? O_o
Mono has (limited) support for VB.Net. Not sure about the Compilers for Xamarin's .Net Emulator's on Android/iOS.

Still cross-platform, and last i checked, you didn't need to use Mono's Compiler to make a binary that worked on Mono, as long as it didn't use feature's Mono itself doesn't support.

SIM-1456245 - February 1, 2015, 1:20 pm

(Feb 1, 2015 02:16 PM)Nikey646 Wrote: [\[CITE\]](#) Mono has (limited) support for VB.Net. Not sure about the Compilers for Xamarin's .Net Emulator's on Android/iOS.

Still cross-platform, and last i checked, you didn't need to use Mono's Compiler to make a binary that worked on Mono, as long as it didn't use feature's Mono itself doesn't support.

Yeah yeah yeah. But there is soo many other reasons why this class is good. One of them being the fact of practice. I coded a JSON serializer/deserializer just for fun.