**Entrega y documentación de “Aplicación de Consola para registro (Enrollment) y validación (Validate), mediante el sistema biométrico Digital Persona“**

La aplicación consiste de una interfaz simple, dado que solo serviría como utilidad para una aplicación en Adobe AIR.

**Requisitos**

Debido a la poca demanda de recursos de parte de la aplicación, se consideraran como requisitos mínimos necesarios, los del .NET Framework bajo el cual se desarrolló.

En este caso, sería la versión 4.0 del framework. Los requisitos son los siguientes:

Sistema operativo compatible

Windows 7; Windows 7 Service Pack 1; Windows Server 2003 Service Pack 2; Windows Server 2008; Windows Server 2008 R2; Windows Server 2008 R2 SP1; Windows Vista Service Pack 1; Windows XP Service Pack 3

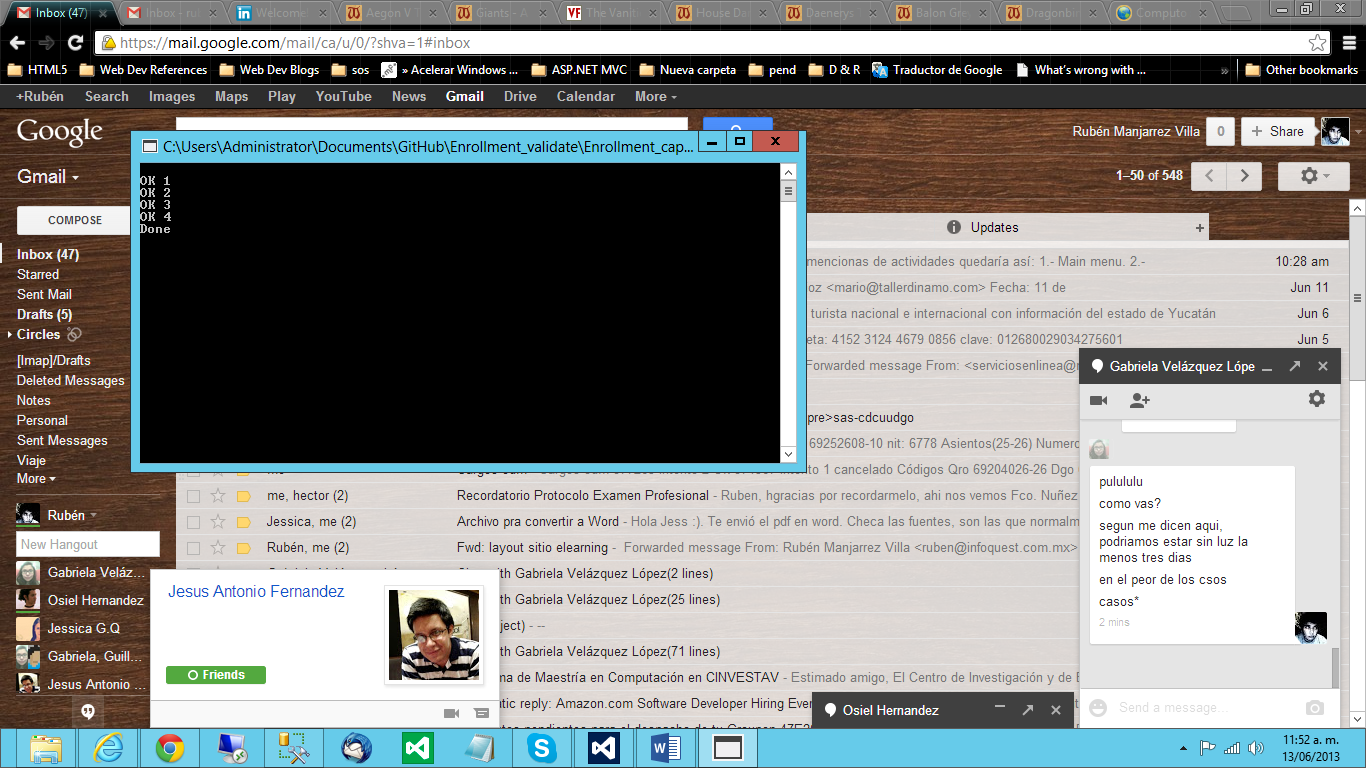
* + - Windows XP SP3
    - Windows Server 2003 SP2
    - Windows Vista SP1 o posterior
    - Windows Server 2008 (no admitido en el rol Server Core)
    - Windows 7
    - Windows Server 2008 R2 (no admitido en el rol Server Core)
    - Windows 7 SP1
    - Windows Server 2008 R2 SP1
  + Arquitecturas admitidas:
    - x86
    - x64
    - ia64
  + Requisitos de hardware:
    - Mínimo recomendado: Pentium 1 GHz o superior con 512 MB de RAM o más
    - Espacio en disco mínimo:
      * x86: 850 MB
      * x64: 2 GB
  + Requisitos previos:
    - [Windows Installer 3.1](http://www.microsoft.com/downloads/details.aspx?familyid=889482fc-5f56-4a38-b838-de776fd4138c&displaylang=es) o posterior
    - [Internet Explorer 5.01](http://www.microsoft.com/windows/downloads/ie/getitnow.mspx) o posterior

Otro requisito previo a considerar, es la creación del siguiente directorio: “C:\temp”.

**Aplicación**

La utilidad consiste en dos aplicaciones (ejecutables). La primera se encarga de realizar el registro de la huella digital, un proceso conocido como “Enrollment”, en el cual se capturan 4 muestras de huellas, para después procesarlas y generar como resultado una huella muestra promedio.

**Enrollment**



Se captura 4 veces la huella digital hasta que la consola responde “Done”. Para este momento, ya existe el archivo “cap\_fmd\_enrolled.txt”.

El código de este apartado de la utilería es el siguiente:

Imports DPUruNet.Fingerbase

Imports DPUruNet

Imports System.Threading

Imports System.Collections.Generic

Imports DPUruNet.Constants

Imports System.IO

Module Module1

'Declaración de variables globales

Private \_readers As ReaderCollection

Private count As Integer

Dim reader As Reader

Dim fp As String

Dim preenrollmentFmds As New List(Of Fmd)

'Metodo que genera las colecciones de huellas

Public Property Fmds() As Dictionary(Of Int16, Fmd)

Get

Return \_fmds

End Get

Set(ByVal value As Dictionary(Of Int16, Fmd))

\_fmds = value

End Set

End Property

Private \_fmds As Dictionary(Of Int16, Fmd) = New Dictionary(Of Int16, Fmd)

'Metodo que reinicia el lector

Public Property Reset() As Boolean

Get

Return \_reset

End Get

Set(ByVal value As Boolean)

\_reset = value

End Set

End Property

Private \_reset As Boolean

'Metodo que abre un conexión con el lector

Public Function OpenReader() As Boolean

Reset = False

Dim result As Constants.ResultCode = Constants.ResultCode.DP\_DEVICE\_FAILURE

result = reader.Open(Constants.CapturePriority.DP\_PRIORITY\_COOPERATIVE)

If result <> Constants.ResultCode.DP\_SUCCESS Then

Console.WriteLine("Error: " & result.ToString())

Reset = True

Return False

End If

Return True

End Function

'Metodo que obtiene el estatus de determinado lector de huella

Public Sub GetStatus()

Dim result = reader.GetStatus()

If (result <> ResultCode.DP\_SUCCESS) Then

If reader IsNot Nothing Then

Reset = True

Throw New Exception("" & result.ToString())

End If

End If

If (reader.Status.Status = ReaderStatuses.DP\_STATUS\_BUSY) Then

Thread.Sleep(50)

ElseIf (reader.Status.Status = ReaderStatuses.DP\_STATUS\_NEED\_CALIBRATION) Then

reader.Calibrate()

ElseIf (reader.Status.Status <> ReaderStatuses.DP\_STATUS\_READY) Then

Throw New Exception("Reader Status - " & reader.Status.Status.ToString())

End If

End Sub

'Metodo que captura la huella asincronamente

Public Function CaptureFingerAsync() As Boolean

Try

GetStatus()

Dim captureResult = reader.CaptureAsync(Formats.Fid.ANSI, \_

CaptureProcessing.DP\_IMG\_PROC\_DEFAULT, \_

reader.Capabilities.Resolutions(0))

If captureResult <> ResultCode.DP\_SUCCESS Then

Reset = True

Throw New Exception("" + captureResult.ToString())

End If

Return True

Catch ex As Exception

Console.WriteLine("Error: " & ex.Message)

Return False

End Try

End Function

'Funcion que inicia la captura de huella de manera asincrona

Public Function StartCaptureAsync(ByVal OnCaptured As Reader.CaptureCallback) As Boolean

AddHandler reader.On\_Captured, OnCaptured

If Not CaptureFingerAsync() Then

Return False

End If

Return True

End Function

'Metodo que cancela la captura de huella

Public Sub CancelCaptureAndCloseReader(ByVal OnCaptured As Reader.CaptureCallback)

If reader IsNot Nothing Then

If (Reset) Then

reader = Nothing

End If

End If

End Sub

'Metodo principal

Sub Main()

'Obtengo la coleccion de lectores conectadosa

\_readers = ReaderCollection.GetReaders

Dim serial\_reader As String

'Elijo el primer lector, asumiendo que solo existe uno conectado

serial\_reader = \_readers(0).Description.SerialNumber

reader = \_readers(0)

count = 0

'Abro el lector elegido

OpenReader()

'Inicio la captura asincrona

StartCaptureAsync(AddressOf OnCaptured)

Console.WriteLine("")

Console.ReadLine()

End Sub

'Metodo que se ejecuta al detectar una captura.

Public Sub OnCaptured(ByVal captureResult As CaptureResult)

'Contador para llevar el control de las capturas de las 4 muestras de huella

count += 1

'Convierto la huella cpaturada a un objeto para extraer sus propiedades.

Dim resultConversion As DataResult(Of Fmd) = FeatureExtraction.CreateFmdFromFid(captureResult.Data, Formats.Fmd.DP\_PRE\_REGISTRATION)

'Si la huella se capturo correctamente, la agrego a la coleccion de huellas, si no, la descarto

If resultConversion.ResultCode <> Constants.ResultCode.DP\_SUCCESS Then

Console.WriteLine("no funciona")

Else

preenrollmentFmds.Add(resultConversion.Data)

Console.WriteLine("OK " + count.ToString)

End If

'Al recolectar las 4 huellas, en base a la coleccion, genero la huella muestra, la serializo y la escribo en un archivo de texto en la ruta indicada

If count >= 4 Then

Dim resultEnrollment As DataResult(Of Fmd) = DPUruNet.Enrollment.CreateEnrollmentFmd(Formats.Fmd.DP\_REGISTRATION, preenrollmentFmds)

fp = Fmd.SerializeXml(resultEnrollment.Data)

File.WriteAllText("c:\temp\cap\_fmd\_enrolled.txt", fp)

Console.WriteLine("Done")

End If

End Sub

End Module

Para este momento la aplicación AIR abrirá el archivo generado, guardara su contenido en la base de datos, y eliminara el archivo del directorio.

**Validate**

En el caso d la validación, la aplicación valida una huella capturada contra la huella previamente guardada, respondiendo “Usuario valido” o “Usuario Invalido”. Según sea el caso.

El código de este apartado de la utilería es el siguiente:

Imports DPUruNet.Fingerbase

Imports DPUruNet

Imports System.Threading

Imports System.Collections.Generic

Imports DPUruNet.Constants

Imports System.IO

Module Module1

'Declaro variables globales

Private \_readers As ReaderCollection

Private count As Integer

Dim reader As Reader

Dim fp As String

Dim Fmds\_ver As New List(Of Fmd)

'Metodo que almacena la coleccion de huellas

Public Property Fmds() As Dictionary(Of Int16, Fmd)

Get

Return \_fmds

End Get

Set(ByVal value As Dictionary(Of Int16, Fmd))

\_fmds = value

End Set

End Property

Private \_fmds As Dictionary(Of Int16, Fmd) = New Dictionary(Of Int16, Fmd)

'Metodo que reinicia el lector de huella

Public Property Reset() As Boolean

Get

Return \_reset

End Get

Set(ByVal value As Boolean)

\_reset = value

End Set

End Property

Private \_reset As Boolean

'Metodo que abre determinado lector de huella

Public Function OpenReader() As Boolean

Reset = False

Dim result As Constants.ResultCode = Constants.ResultCode.DP\_DEVICE\_FAILURE

result = reader.Open(Constants.CapturePriority.DP\_PRIORITY\_COOPERATIVE)

If result <> Constants.ResultCode.DP\_SUCCESS Then

Console.WriteLine("Error: " & result.ToString())

Reset = True

Return False

End If

Return True

End Function

'Metodo que regresa el status del lector de huella

Public Sub GetStatus()

Dim result = reader.GetStatus()

If (result <> ResultCode.DP\_SUCCESS) Then

If reader IsNot Nothing Then

Reset = True

Throw New Exception("" & result.ToString())

End If

End If

If (reader.Status.Status = ReaderStatuses.DP\_STATUS\_BUSY) Then

Thread.Sleep(50)

ElseIf (reader.Status.Status = ReaderStatuses.DP\_STATUS\_NEED\_CALIBRATION) Then

reader.Calibrate()

ElseIf (reader.Status.Status <> ReaderStatuses.DP\_STATUS\_READY) Then

Throw New Exception("Reader Status - " & reader.Status.Status.ToString())

End If

End Sub

'Metodo que captura la huella asincronamente

Public Function CaptureFingerAsync() As Boolean

Try

GetStatus()

Dim captureResult = reader.CaptureAsync(Formats.Fid.ANSI, \_

CaptureProcessing.DP\_IMG\_PROC\_DEFAULT, 500)

If captureResult <> ResultCode.DP\_SUCCESS Then

Reset = True

Throw New Exception("" + captureResult.ToString())

End If

Return True

Catch ex As Exception

Console.WriteLine("Error: " & ex.Message)

Return False

End Try

End Function

'metodo que inicia la captura asincrona

Public Function StartCaptureAsync(ByVal OnCaptured As Reader.CaptureCallback) As Boolean

AddHandler reader.On\_Captured, OnCaptured

If Not CaptureFingerAsync() Then

Return False

End If

Return True

End Function

Public Sub CancelCaptureAndCloseReader(ByVal OnCaptured As Reader.CaptureCallback)

If reader IsNot Nothing Then

'Metodo para cerrar el lector de huellas

If (Reset) Then

reader = Nothing

End If

End If

End Sub

Sub Main()

'Obtengo la coleccion de lectores de huella conectados al equipo, asumo que solo hay uno conectado y lo elijo.

\_readers = ReaderCollection.GetReaders

Dim serial\_reader As String

serial\_reader = \_readers(0).Description.SerialNumber

reader = \_readers(0)

count = 0

'Abro el lector elejido e inicio el proceso de captura asincrona

OpenReader()

StartCaptureAsync(AddressOf OnCaptured)

Console.WriteLine("")

Console.ReadLine()

End Sub

'Metodo que se ejcuta en cuanto se captura una huella

Private Sub OnCaptured(ByVal captureResult As CaptureResult)

'Genero el objecto con la huella en base a la huella capturada

Dim resultConversion As DataResult(Of Fmd) = FeatureExtraction.CreateFmdFromFid(captureResult.Data, Formats.Fmd.DP\_VERIFICATION)

'Genero un objeto para la clase de comnparacion de huellas

Dim compareResult As DPUruNet.CompareResult

'Comparo la huella capturada con la huella guardada en la ruta especificada, de-serializandola previamente

compareResult = Comparison.Compare(resultConversion.Data, 0, Fmd.DeserializeXml(File.ReadAllText("c:\temp\cap\_fmd\_enrolled.txt")), 0)

'Salida indicando si el usuario es valido o no

Console.WriteLine(IIf(compareResult.Score < (&H7FFFFFFF / 100000), "Usuario Valido", "Usuario Invalido"))

End Sub

End Module

Cabe recalcar que es necesario tener instalado el RTE de Digital Persona para que la utilería funcione.