

FACEPORT ACTIVEX DATASHEET

About Ex-Sight.Com

Ex-Sight.Com is an Israeli software company that specializes in accurate Recognition Systems. Our technology allows high-end facial recognition detection based on the latest technology in the market.


We provide worldwide OEM Development Services, High-End Biometric Engineering, Interactive systems and Image processing applications for both Security and Retail. We provide customized solutions for Banking, Security and Retail systems.

We are protecting Corporate Security and Personal Identity using smart Biometric Authentication Algorithms with the latest technology.

Introduction

The FACEPORT.DLL ACTIVEX allows fast use, development and implementation of facial recognition capabilities in existing or new designed applications. The component provides an easy to use Methods, Properties and Events that simplifies the facial recognition development process.

The FACEPORT ActiveX was specially designed for embedded environment like Intel ATOM CPU, and runs very well on Low performance CPU as well as high performance CPUs. The component supports Web Cameras, IP or Analog cameras through external injection mechanism. The component includes internal web camera management. The component contains internal Database that provides fast yet reliable management of the matching subjects.



Applications:

Security:

- Access Control
- Suspect Detection
- Time Attendance
- Web Login
- PC Login

Commercial

- Repeated clients
- Interactive Content
- Customer counting
- Graphical Applications

Platforms Supported

Faceport ActiveX supports platforms based on x86, x86_64 and PowerPC processors architectures. Libraries for Windows operating systems are provided.

System Requirements

- PC with Pentium-compatible 1GHz processor or better.
- Microsoft Windows 2000/XP/Vista.

LICENSING

PC BASED.

Single computer license

A single computer license allows installing and running a Faceport component installation on one computer processor core.

The following license management options are available:


- License activation online by communicating with Ex-Sight server
 - License activation by email
- 

Image Support

Image support in the Faceport ActiveX can be divided into the following three parts:

- **Image**. The base of all image support. Developers should start using this part and take advantage of other parts if it is required.
- **Image Format**. Declares the supported image formats. Shows how to load and save images in a format-neutral way.
- **Low-Level Image Input-Output**. Should be used to have more control on how images are loaded and saved in particular format.

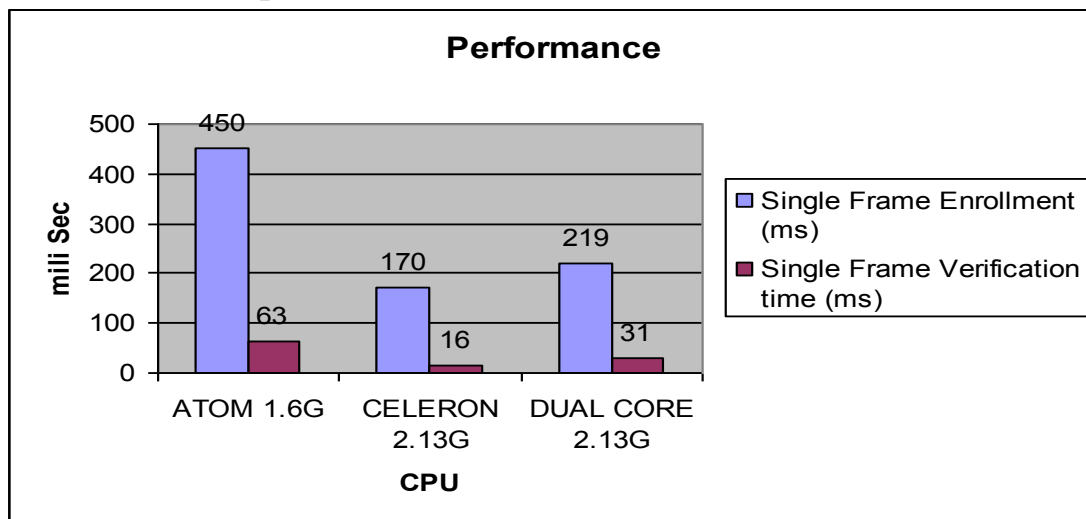
Image Format

Image format is a specification of **image** storage in a file. The specification may require compress/decompress image during writing/reading it to/from a file.

See the following table for details.

Image Format	Can read	Can write
BMP	Yes	Yes
GIF	In .NET only	In .NET only
JPEG	Yes	Yes
PNG	Yes	Yes
TIFF	Yes	In .NET only

Performance Capabilities



Registration

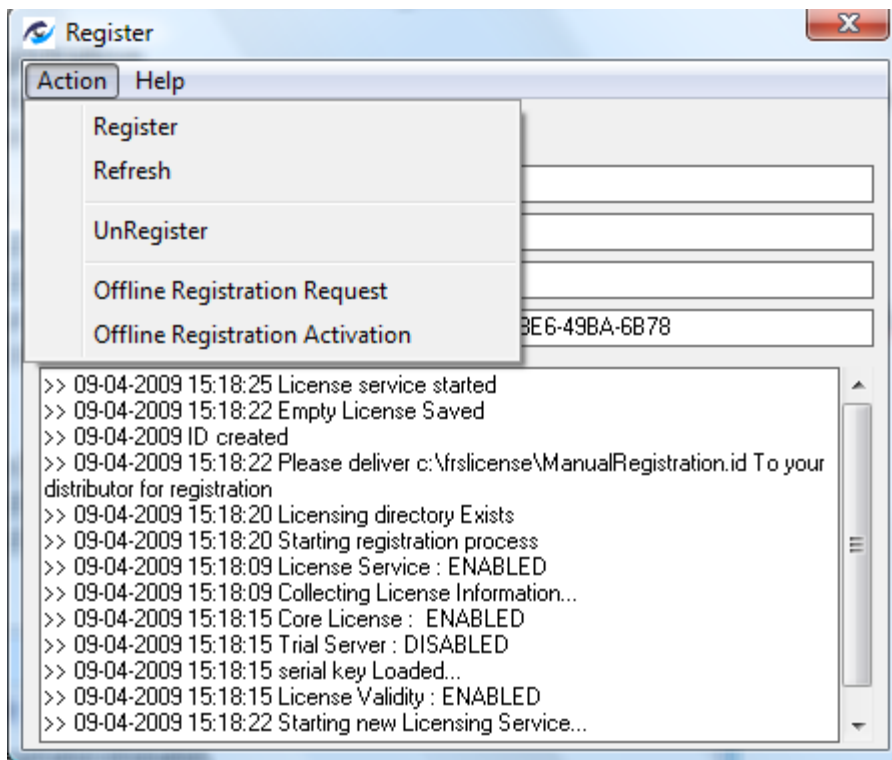
Launch the registration application from:

C:\Program Files\FacePort\FacePortRegistration.exe

Manual Registration

Manual registration allows users that do not have direct internet connection on their PCs to use offline registration mechanism.

1. Enter all fields in a valid way to the registration application
2. Select "Action" Menu
3. Select "Offline Registration Request"
4. Under c:\FRSLicense\ directory you can find a file named "ManualRegistration.id"
5. Send the file to your distributor and wait for 2 files:
 - a. Serial.id
 - b. FACEPORT.LIC
6. Copy the files to "C:\FRSLicense\"
7. Select "Action" menu
8. Select "Offline Registration Activation"



Online Registration (Requires internet connection)

1. Enter all fields in a valid manner to the registration application
2. Select "Action" Menu
3. Select "Register" option
4. Wait for registration confirmation
5. If registration process fails please contact your distributor

FRS Database

Jet 4.0 MS Access database (Ex-Sight will consider using ODBC in order not to be depend on DB type, but must test the robustness of using ODBC. In case of lack of robustness, it will keep MS Access database)

Log Database

JET 4.0 MS Access database

FRS Component

ActiveX Entrance / Reception DLL

FILE LIST

Faceport.Dll
FacePort.Lib
FacePort.exp
FRS.mdb

Faceport Properties:

1. NoOfFramesForValidation

Min. number of frames for validation. Number of frames in which the matching attempts must succeed, a successful match is if $(\text{Matching Attempts} / \text{NoOfFramesForValidation})$ of the frames match the same person.

Must be higher than the **MatchingAttempts** value.
(Default 3, Min = 1, Max = 10)

(Default 10, Min = 1, Max = 25, if AllowLiveness is true Min = 10)

2. NoOfFramesForEnrollment

Min. number of frames for enrollment (Default 10, Min = 1, Max = 25)

3. FaceConfidenceThreshold

- parameter defines a threshold for a factor "how an image is similar to a human face". Sometimes the Component can detect non-face images. To avoid this you can increase the value of threshold. Usually FACECONFIDENCETHRESHOLD is not used by end-user. Recommendation is to leave default value, if no problems with non-face image recognition is received.

(Default 52, Min = 1, Max = 100)

4. MinimalIOD

Min. Inner distance between 2 eyes (Default 40, Range- 0 to Maximal IOD)

5. MaximalIOD

Max. Inner distance between 2 eyes (Default 4000) (Range- MinimalIOD to 4000)

(Minimal Inter Ocular Distance (IOD) and Maximal IOD defines the distance between detected eyes range. Distances outside this range are ignored and faces are not extracted. Distance between the eyes must be at least 40 px to be able to enroll successfully. For successful matching 50 px is recommended.)

6. FaceQualityThreshold

- Specifies the threshold which is considered when extracting facial features from the image. With higher threshold better quality of face image is required to successfully extract facial features. The value of this parameter can be in range [0..255]. The default value is 128.

7. MinMatchingThreshold

Min. value for matching a face (Default 48, Min = 1, Max = 100)

8. **MaxMatchingThreshold**

Max. value for matching a face (Over this value it is considered a sure match).
The algorithm will stop looking for better matches when it reaches this value.
(Default 100, Min = MinMatchingThreshold, Max = 100)

9. **FAR**

Will be calculated from matching threshold (theoretical value)

10. MatchingAttempts - Attempts of matching when matching from camera are performed.
Must be lower than the **NoOfFramesForValidation** value.
(Default 3, Min = 1, Max = 10)

11. MaxRecordsPerTemplate – Maximum number of records to be stored in the Face Template
(Default 5, Min = 1, Max = 20)

12. **AllowLiveness**

Boolean, will define if we need to use liveness Checking. The mode isn't available when allowing multiple faces detection.

13. **LivenessThreshold**

How strict to check for Live Faces in an image Stream.(default 50, Min = 1 , Max = 100)

14. **AllowAgeing**

Boolean, will define if we need to use aging. Aging will cause the algorithm to save a new template

15. **AgeingMinimalThreshold**

Minimum value criteria of Matching Threshold to update the Same Face Feature
(Default = MatchingThreshold, Min= MatchingThreshold,, Max = 100)

16. **AgeingMaximalThreshold**

Maximal value criteria of Matching Threshold to update the same Face Feature

17. EnrolledFaceImagePath

Path to store all the enrolled Face Images

18. RejectedFaceImagePath

Path to store all the rejected Face Images

19. ValidatedFaceImagePath

Path to store all the validated Face Images

20. LogDBPath

Path of the Access MDB File where the Log's should be stored.

21. FRSDBPath

Path of the Access MDB File where the Face Data should be stored.

22. IsRegistered

Checking for the Licensing is registered. (Returns true if registered)

23. FRR

Will be calculated from matching threshold (theoretical value)

24. Camera

Gets/Sets the currently selected Camera

25. VideoFormat

Gets/Sets the Currently Selected Camera's Video Format.

26. FlipImagesHorizontal

Gets/Sets value whether to Flip the Images Horizontally

27. SaveEnrolledFaceImage

Gets/Sets value Whether to Save the Learned/Enrolled Face Image to harddisk

28. SaveRejectedFaceImage

Gets/Sets value whether to save the Rejected Face Image to harddisk

29. SaveValidatedFaceImage

Gets/Sets value Whether to Save the Validated/Matched Face Image to harddisk

30. EnableMultipleFacesExtraction

Gets/Sets value whether extract face templates from all the faces in the picture or only from the first face found. (Liveness check isn't available in this mode)

31. AlwaysExtractTemplate

Gets/Sets value whether to constantly extract face templates from each frame

32. PrimaryEnable

Enable the use of the primary algorithm to extract facial features and templates

33. SecondaryEnable

Enable the use of the primary algorithm to extract facial features and templates

34. SecondaryFaceConfidenceThreshold

Set the confidence threshold for the secondary algorithm (Between 1-10)

35. SecondaryIncreasedAngle

Set the secondary algorithm to support increased angle range (between -30 to 30 degrees)

36. SecondaryInternalResize

Set the secondary algorithm's internal resizing size. Smaller values mean higher performance but lower accuracy

37. SecondaryMaxFaces

Sets the maximum number of faces the secondary algorithm may extract

38. DrawGUI

Enables or Disables the circles around the detected faces.

39. Version

Returns the SDK's version number.

40. IsBusy

Returns "True" when the algorithm is busy.

41. SecondaryWeight

Sets the weight of the secondary algorithm's similarity value. Default is 50% meaning Faceport will take the average of the similarity rates.

42. DBPassword

Sets the password Faceport will use to access the DataBase

44. DBString

Allows you to set your own connection string to the database. You should use OleDb Connection syntax for the string. [OLEDB connection strings](#)

45. OnlyEnrollNewFace

When this property is enabled, the algorithm will check if the face is recognized as a different user. If the face matches a different user then it will not be enrolled under a different name.

Events :

1. Camera

- **CameraOpened()** - Occurs While the Camera Start's Capturing.
- **CameraClosed ()** - Occurs while the camera Stop's Capturing.
- **NewCameraFrame(bitmap image)** – Occurs for every new frame

2. **ExceptionHappened**(string Description, int ErrorNumber)

Occurs when an exception has been raised.

3. **FaceLearnt**(int FaceQuality, float Similarity, float Face Confidence, int IOD)

Occurs after a Successful Learning of Face.

4. **FaceValidated**(int FaceQuality, float Similarity, float FaceConfidence, int IOD)

Occurs after a Successful Validation of Face.

5. **FaceLearningFailed**(int ErrorTypeID, String Description, int FaceQuality, float Similarity, float FaceConfidence, int IOD)

Occurs While a Learning of Face Failed.

6. **FaceValidationFailed**(int ErrorTypeID, String Description, int FaceQuality, float Similarity, float FaceConfidence, int IOD)

Occurs after a failure of validating a Face.

7. **FaceDeleted()** – Occurs after deletion of face

8. **TemplateReady**(Array Templates, int numTemplates)

Occurs when a template is extracted.

9. **LogOverflow()** – when log record number is above 100K record

10. **FrameProcessStarted()**- Triggers when Faceport starts processing.

11. **FrameProcessFinished()** - Triggers when Faceport finishes processing

Methods

1. **InitializeFRS()** – Re-Initializes the FRS Component.
2. **OpenFRSSettingsDialog ()** – Open's the face recognition component settings Dialogbox.
3. Bool **LearnFaceFromCamera** (string FaceID) – Starts the process of learning a new face from Camera Input.
4. Bool **LearnFaceFromImage**(string FaceID, string ImagePath) - Starts the process of learning a new face from File Input.
5. Bool **ValidateFace** (string FaceId) - Starts the process of Validating the face
6. String[] **GetRejectedImagesByFaceID**(string FaceID) – Return's a comma separated string of rejected Images File Path.
7. String[] **GetRejectedFaceIDsFromDate**(string FromDate, string ToDate) - Return's a comma separated string of rejected FaceId's.
8. **DeleteRejecedImagesFromDate**(string FaceID, string UpToDate) – Delete's the Rejected Images between the date.
9. **DeleteAllRejecedImagesFromDate** (string UpToDate) – Delete All the rejected images up to the given date.
10. String[] **GetImagesByFaceID**(string FaceID) – Returns a comma separated Sting of FaceID's that are learntk
11. **DeleteFaceID**(string FaceID) – Deletes the given FaceID.
12. Bool **DeleteFaceIDImages**(string FaceID, string UpToDate) –Delete's Only the Image if the given FaceID.
13. Float **MatchTwoFaces**(string FaceId1, string FaceId2) – returns the best Match between the two faces
14. Float **MatchImageWithFace**(string filename, string FaceID) – Matches the given image to given FaceID
15. **AddToLog**(string timestamp, string type, string data) – Add's a log message.

16. ClearLog() – Clear Log db

17. Camera

1. **String[] GetConnectedCameras()** – Returns a comma separated string of the name of the Connected Camera's.
2. **Bool OpenCamera(string camera)** – Start's the camera to capture Images.
3. **CloseCamera()** – Stops the Running Camera
4. **ShowCameraSettingsDialog(int x,y) - -1,-1 center of screen** – Displays the Camera settings Dialog.
5. **ShowCameraPreviewDialog(int x,y)- -1,-1 center of screen** – Display's the Camera Preview Dialog.
6. **bool SaveSnapshot(string Filename)** – Save's a snapshot image to the disk with the given file name.
7. **PlaceImageAnnotation(string TargetFilename, string Data)** – Places an annotation to the currently selected Image with the given data.

18. InjectImage (int HImage) – Injects an Image using it HBITMAP or Image Handle

19. InjectImageFromFile (String fileName) – Injects an Image from a File

20. ExtractTemplatesFromHBitmap(int hbitmap, ref Array TemplateArray) – Extracts face templates from Hbitmap

21. ExtractTemplatesFromFile(String ImagePath, ref Array TemplateArray) – Extracts face templates from a file.

22. SaveSettingsToFile(String Path) – Save the current settings to the specified file.

23. LoadSettingsFromFile(String Path) – Load settings from the specified file.

24. **Activate**(ref a FacePortLicense license) - Activate the instance using a FacePortLicense object.
25. Int **MatchTemplates**(ref FaceTemplate template1, ref FaceTemplate template2) – Compares 2 templates and returns the matching join score.
26. Int **PrimaryMatchTemplates**(ref Array template1, ref Array template2) – Compares 2 Primary templates and returns the matching score.
27. Int **SecondaryMatchTemplates**(ref Array template1, ref Array template2) – Compares 2 Secondary templates and returns the matching score.
28. **ValidateFaceFromTemplate**(ref FaceTemplate, ref string userID, ref int score) – Tries to validate a template and return the matching userID and the join score.
29. **ExtractResizedTemplatesFromFile**(string ImagePath, ref Array FaceTemplateArray, ref int numTemplates) – Extracts a template from the file, and then resizes it to get an IOD of 40 , 60 , 80 , 100. Then extracts templates from those images as well
30. **ExtractResizedTemplatesFromHBitmap**(ref int hbitmap, ref Array FaceTemplateArray, ref int numTemplates) – Extracts a template from the bitmap, and then resizes it to get an IOD of 40 , 60 , 80 , 100. Then extracts templates from those images as well.
31. **LearnFaceFromTemplate**(string faceid, ref FaceTemplate face, int hbitmap) – Learns the template and adds it to the database.
32. **Array CropFacesFromFile**(string fileName) – Crops all the faces from a file and returns an array of the bitmap handlers.
33. **Array CropFacesFromHBitmap**(int bmpPic) – Crops all the faces from a bitmap and returns an array of the bitmap handlers.
34. **Array GetTemplatesByID**(string userId) – Returns an array of all the templates stored for that ID in the database.

STRUCTURES

FaceTemplate:

Contains information about the faces that were extracted.

Members:

PrimaryTemplate – A binary representation of the template extracted by the primary algorithm

SecondaryTemplate. A binary representation of the template extracted by the secondary algorithm.

X : The x value of the face's top-left corner

Y: The y value of the face's top-left corner

W: The face's width value

H: The Face's height value

CHANGE LIST

- **Offline registration feature**
- **ContinuousValidating Property:**

Allows continues face recognition for access control systems
- **EnableImageInjection Property – (True/False)**

Allows grabbing an image from external video source and performing facial recognition on it. This allows external connectivity to different video sources.
- **InjectImage**

Injects a new image to the Faceport Object by the HBITMAP Image Handle
- **InjectImageFromFile**

Injects a new Image to the Faceport Object from the given filename.

- **CloseCameraPreviewDialog()**

Closes the Camera Preview Dialog and releases the Form from the memory.

- **FrameRate Property**

Helps in adjusting the Frame Rate of the Camera.

Change List (30/10/2009 Ver : 1.2.0.3)

- **CameraType Property** – Added new property Camera type(ctCameraMan, ctGeneric) defines the method of accessing the camera.
- **Camera Settings Added For Generic (ctGeneric) Camera Type.**
- **Error Codes Added Below.**

DeleteFaceIDImages Function Problem Corrected.

ERROR CODES:

N_OK	9000	No error.
N_E_FAILED	8999	Unspecified error has occurred.
N_E_CORE	8998	Standard error has occurred (for internal use).
N_E_NULL_REFERENCE	8997	Null reference has occurred (for internal use).
N_E_OUT_OF_MEMORY	8996	There were not enough memory.
N_E_NOT_IMPLEMENTED	8995	Functionality is not implemented.
N_E_NOT_SUPPORTED	8994	Functionality is not supported.
N_E_INVALID_OPERATION	8993	Attempted to perform invalid operation.
N_E_OVERFLOW	8992	Arithmetic overflow has occurred."
N_E_INDEX_OUT_OF_RANGE	8991	Index is out of range (for internal use).
N_E_ARGUMENT	8990	Argument is invalid.
N_E_ARGUMENT_NULL	8989	Argument value is NULL where non-NULL value was expected.
N_E_ARGUMENT_OUT_OF_RANGE	8988	Argument value is out of range.
N_E_FORMAT	8987	Format of argument value is invalid.
N_E_IO	8986	Input/output error has occurred.
N_E_END_OF_STREAM	8985	Attempted to read file or buffer after its end.
N_E_EXTERNAL	8984	Error in external code has occurred (for internal use).
N_E_WIN32	8983	Win32 error has occurred.
N_E_COM	8982	COM error has occurred.

N_E_CLR	8981	CLR exception has occurred.
N_E_PARAMETER	8980	Parameter ID is invalid.
N_E_PARAMETER_READ_ONLY	8979	Attempted to set read only parameter.
N_E_NOT_REGISTERED	8978	Module is not registered.

PRODUCT_IS_NOT_REGISTERED	1010	Product is not registered
ENROLLEMENT_FAILED	1013	Face Learning Failed
CAMERA_NOT_SELECTED	1015	Camera is not selected
CAMERA_IS_NOT_RUNNING	1016	Camera is not Running
IMAGEINJECTIONENABLED_CANT_ACCESS_CAMERA	1017	ImageInjection Enabled Cannot Access Camera
LICENSE_IS_NOT_VALID	1018	License is not valid
VALIDATION_FAILED	1100	Validation Failed
VALIDATION_FAILED_NO_FACE_ENROLLED	1101	Validation Failed : No Face Enrolled
VALIDATION_FAILED_NO_FACE_FOUND	1102	Validation Failed : No Face Found
VALIDATION_FAILED_LIVENESS_CHECK_FAILED	1103	Validation Failed : Liveness Check Failed
VALIDATION_FAILED_TIMED_OUT	1104	Vaildation Failed: Timed out
VIDEO_FORMAT_DIALOG_NOT_SUPPORTED	1019	Video Fromat Dialog Not Supported

VIDEO_SOURCE_DIALOG _NOT_SUPPORTED	1020	Video source Dialog Not Supported
ENROLLEMENT_FAILED _NO_FACE_FOUND	1200	Enrollement Failed:No Face Found
ENROLLEMENT_FAILED _TIMED_OUT	1201	Enrollement Failed:Timed Out
ENROLLEMENT_FAILED _FACE_EXISTS	1202	Enrollement Failed: Face already exists


Change List (Ver:1.2.0.4 05/11/2009)

1. Registration of Hasp Key and Evaluation Included.
2. FaceAppears(int facecount) – Event Added
3. ValidateWhenFaceAppears - Property Added(true/False)

Change List (Ver:1.2.0.5 , Dt:21/12/2009)

1. ImageRotation - A New property Added to rotate Image in desired Direction.(0, 90, 180, 270 Degrees)
2. SaveSettings - New Method added for saving the FRS Settings Externally.
3. LoadSettings - New Method to Load the FRS Settings Externally.
4. Internal COM Exception Bugs Fixed.
5. All Events Integer Parameters changed to Long.
6. FaceLearnt – Event signature changed (a new argument added – Byval mDBReocrdID as Long).

Change List (Ver:2.0.0.0 , Dt:26/05/2010)

1. **MatchTemplates** – Matches 2 templates and returns the matching score.
 2. **ExtractTemplatesFromHBitmap** – Extracts face templates from Hbitmap
 3. **ExtractTemplatesFromFile**– Extracts face templates from a file.
 4. **TemplateReady** – Event that occurs when a template is extracted.
 5. **EnableMultipleFacesExtraction**- Gets/Sets value whether extract face templates from all the faces in the picture or only from the first face found.
 6. **AlwaysExtractTemplate** -Gets/Sets value whether to constantly extract face templates from each frame
 7. **FacePortLicense** – An object that controls the license activation.
- 

8. **AgingMaximalThreshold** - Maximal value criteria of Matching Threshold to update the same Face Feature
9. **PrimaryEnable** – Enable the use of the primary algorithm to extract facial features and templates
10. **SecondaryEnable** – Enable the use of the primary algorithm to extract facial features and templates
11. **SecondaryFaceConfidenceThreshold** – Set the confidence threshold for the secondary algorithm (Between 1-10)
12. **SecondaryIncreasedAngle** – Set the secondary algorithm to support increased angle range (between -30 to 30 degrees)
13. **SecondaryIntrenalResize** – Set the secondary algorithm's internal resizing size. Smaller values mean higher performance but lower accuracy.
14. **SecondaryMaxFaces** – Sets the maximum faces the secondary algorithm may extract.
15. **DrawGUI** –Enables/Disables the drawing of the circles around detected faces.
16. **SaveSettingsToFile** – Saves the settings to a file
17. **LoadSettingsFromFile** – Loads the settings from a file
18. **IsBusy** – A property that checks if the algorithm is busy.

Change List (Ver:2.0.0.0 , Dt:27/07/2010)

1. **SetDBPassword** – Sets the password Faceport will use to access the DataBase
2. **SecondaryWeight** – Sets the weight of the secondary algorithm's similarity value. Default is 50% meaning Faceport will take the average of the similarity rates.
3. **FrameProcessStarted** – An event that triggers when Faceport starts processing.
4. **FrameProcessFinished** – An event that triggers when Faceport finishes processing

Change List (Ver:2.1.3.0 , Dt:10/08/2010)

1. **Added more property exception information**
2. **MinMatchingThreshold** – Instead of MatchingThreshold. (minimum threshold to accept)
3. **MaxMatchingThreshol** - Any score over this value will be treated as a sure match.
4. **DBString** – Set a Database connection string (OleDbConnection Syntax)

Change List (Ver:2.1.4.0 , Dt:18/08/2010)

1. **MatchTemplates** – Compares 2 FaceTemplates and gives you the combined scored according to your settings.
2. **ValidateFaceFromTemplate** - Validates a FaceTemplate, returns the matching faceID and triggers FaceValidated or FaceValidationFailed . Can add an image to be saved in the events.
3. **Removed the DBRecordID from events.**
4. **OnlyEnrollNewFace** – When enabled, a frame will only enroll if the face in it isn't already recognized as another userID.
5. **ExtractResizedTemplatesFromHBitmap** – Extracts a template from the bitmap, and then resizes it to get an IOD of 40 , 60 , 80 , 100. Then extracts templates from those images as well.
6. **ExtractResizedTemplatesFromFile** – Extracts a template from the file, and then resizes it to get an IOD of 40 , 60 , 80 , 100. Then extracts templates from those images as well.
7. **LearnFaceFromTemplate** – Learns a face from a given template. Can add an image to be saved in the events.

8. **CropFacesFromFile** – Crops the faces from a file and returns a list of bitmaps containing the faces. You need to free the bitmaps yourself when finished.
9. **CropFacesFromHBitmap** - Crops the faces from an HBitmap and returns a list of bitmaps containing the faces. You need to free the bitmaps yourself when finished
10. **GetTemplatesByID** – Returns an array of all the user's templates.