Limnor Studio – References

Part – I

Integrated Development Environment and Components

Contents

1 Limnor Studio IDE 15
1.1 Introduction to Limnor Studio IDE 15
1.2 Solution Explorer/Project Explorer 16
1.3 Properties Windows 17
1.4 Toolbox 17
1.5 Messages Windows 17
1.6 User Interface Designer 17
1.7 Object Explorer 17
1.8 Event Path 17
1.9 Menus 18
1.10 Tools 20
2 Mouse Pointer 20
2.1 Methods 20
2.1.1 ClipCursor `(Rectangle rc)` 20
2.1.2 ClearCursorClip 20
2.1.3 int SetCursorPosition(int x, int y) 20
3 Keyboard 21
3.1 Properties 21
3.1.1 bool EnableHotKeys 21
3.1.2 HotKeyList HotKeys 21
3.1.3 RemoveHotKey(Key key) 21
4 Windows Manager 21
4.1 Properties 21
4.1.1 Controls - WindowControl[] 21
4.1.2 EventsEnabled - Boolean 21

4.1.3 FoundWindow - IntPtr 21
4.1.4 LastErrorCode - Int32 22
4.1.5 LastErrorMessage - String 22
4.1.6 WindowBounds - Rectangle 22
4.1.7 WindowLocation - Point 22
4.1.8 WindowSize - Size 22
4.2 Methods 22
4.2.1 static Image CropImage(Image image, Rectangle cropRectangle) 22
4.2.2 static bool SaveBitmapToFile(Bitmap bitmap, string filename, EnumImageFormat format, bool cropImage) 22
4.2.3 static Bitmap CaptureWindowImage(IntPtr hWnd) 22
4.2.4 static Bitmap CaptureControlImage(Control control) 23
4.2.5 IntPtr GetWindowByTitle(string windowTitle) 23
4.2.6 IntPtr GetWindowTitleStartWith(string titleStart) 23
4.2.7 IntPtr GetWindowTitleEndWith(string titleEnd) 23
4.2.8 IntPtr GetWindowTitleContains(string titlePart) 23
4.2.9 string GetControlText(int controlId) 23
4.2.10 Bitmap CaptureWindowImage(string windowTitle) 24
4.2.11 bool SaveWindowImageToFile(string windowTitle, string filename, EnumImageFormat format, bool cropImage) 24
4.2.12 PrintControl(Control control, string documentName, bool preview) 24
4.2.13 IntPtr GetWindowUnderMouse() 24
4.2.14 Bitmap CaptureWindowUnderMouse() 24
4.2.15 bool SaveWindowUnderMouseImageToFile(string filename, EnumImageFormat format, bool cropImage) 24
4.2.16 Bitmap CaptureScreen() 25
4.2.17 bool SaveScreenImageToFile(string filename, EnumImageFormat format, bool cropImage) 25
4.3 Events 25
4.3.1 ApplicationActivate - EventHandlerApplicationActivate 25
4.3.2 EnterSizeMove - EventHandler 25
4.3.3 ExitSizeMove - EventHandler 25
4.3.4 SystemCommand - EventHandlerSystemCommand 26

4.3.5 WindowActivate - EventHandlerWindowActivate 26
4.3.6 WindowGotFocus - EventHandlerWindowFocus 26
4.3.7 WindowKillFocus - EventHandlerWindowFocus 26
4.3.8 WindowPositionChanged - EventHandlerWindowPositionChange 26
4.3.9 WindowPositionChanging - EventHandlerWindowPositionChange 26
4.3.10 WindowResized - EventHandlerWindowResized 26
4.3.11 WindowResizeOrMove - EventHandler 26
4.3.12 WindowSizing - EventHandlerWindowSizing 26
4.4 Classes for Event Arguments 27
4.4.1 EventArgsApplicationActivate 27
4.4.2 EventArgsSystemCommand 27
4.4.3 EventArgsWindowActivate 27
4.4.4 EventArgsWindowFocus 28
4.4.5 EventArgsWindowPositionChange 28
4.4.6 EventArgsWindowSized 30
4.4.7 EventArgsWindowSizing 30
4.5 Enumerators used in events 30
4.5.1 EnumSystemCommand 30
4.5.2 EnumSizeEdge 31
4.5.3 EnumWindowSizeType 32
4.5.4 EnumWindowRelativePostion 33
5 Application Configuration 33
5.1 Properties 33
5.1.1 Guid ApplicationGuid 33
5.1.2 string ConfigurationProfile 33
5.1.3 string Filename 33
5.1.4 CategoryList Configurations 33
5.2 5.2 Methods 34
5.2.1 string ChangePassword(string name, string password, string newPassword) 34
5.2.2 bool CreateNamedProfile(string name, string password) 34
5.2.3 ExecuteLoadingConfigurations 34
5.2.4 ExecuteSavingConfigurations 34

5.2.5 CreateProfile(Form caller) 34
5.2.6 UseFactoryProfile 34
5.2.7 UseUserProfile 35
5.2.8 string LogOnProfile(string name, string password) 35
5.2.9 bool LogOnProfileWithUI(Form caller) 35
5.2.10 ChangePasswordByUI(Form caller) 35
5.2.11 string GetSetting(string name) 35
5.2.12 SetSetting(string name, object value) 35
5.2.13 Save 35
5.3 Events 35
5.3.1 static event EventHandler LoadingConfigurations 36
5.3.2 static event EventHandler SavingConfigurations 36
6 Mathematic Expression 36
6.1 Properties 36
6.1.1 object Result 36
6.1.2 string XmlString 36
6.1.3 bool Compiled 36
6.1.4 FormulaProperty Formula 36
6.1.5 int VariableCount 36
6.1.6 string[] VariableNames 36
6.1.7 string LastError 37
6.2 6.2 Methods 37
6.2.1 bool EditFormula 37
6.2.2 SaveToXml(string xmlFile) 37
6.2.3 LoadXml(string xmlFile) 37
6.2.4 AddDrawingSurface(Control surface, Point location) 37
6.2.5 RemoveDrawingSurface(Control surface) 37
6.2.6 Image CreateMathExpressionImage(Graphics g) 37
6.2.7 IMathEditor CreateEditor(Rectangle rc) 37
6.2.8 Compile 37
6.2.9 ShowSourceCode 38
6.2.10 object Execute 38

7 Mail Sender 38
7.1 Properties 38
7.1.1 SmtpClient Smtp 38
7.1.2 string MailServer 38
7.1.3 int Port 38
7.1.4 bool EnableSsl 38
7.1.5 int Timeout 38
7.1.6 bool UseDefaultCredentials 38
7.1.7 ServicePoint ServicePoint 38
7.1.8 string TargetName 39
7.1.9 X509CertificateCollection ClientCertificates 39
7.1.10 SmtpDeliveryMethod DeliveryMethod 39
7.1.11 string PickupDirectoryLocation 39
7.1.12 string SenderAddress 39
7.1.13 string SenderDisplay 39
7.1.14 EnumCharEncode SenderDisplayEncode 39
7.1.15 string FromAddress 39
7.1.16 string FromDisplay 39
7.1.17 EnumCharEncode FromDisplayEncode 40
7.1.18 string ReplyToAddress 40
7.1.19 string ReplyToDisplay 40
7.1.20 EnumCharEncode ReplyToDisplayEncode 40
7.1.21 string Recipients 40
7.1.22 string CCRecipients 40
7.1.23 string BccRecipients 40
7.1.24 MailPriority Priority 40
7.1.25 bool IsBodyHtml 40
7.1.26 DeliveryNotificationOptions DeliveryNotificationOptions 40
7.1.27 string Subject 41
7.1.28 EnumCharEncode SubjectEncoding 41
7.1.29 string Body 41
7.1.30 EnumCharEncode BodyEncoding 41

7.1.31 string UserDomain 41
7.1.32 string UserAccount 41
7.1.33 string UserPassword 41
7.1.34 string[] Attachments 41
7.2 Methods 41
7.2.1 ClearEmbededImages 41
7.2.2 RemoveEmbededImage(string id) 41
7.2.3 AddEmbededImage(string id, string filePath) 42
7.2.4 ClearAttachments 42
7.2.5 RemoveAttachment(string filename) 42
7.2.6 AddAttachment(string filename) 42
7.2.7 AddRecipient(string address, string display, EnumCharEncode displayEncoding) 42
7.2.8 AddCCRecipient(string address, string display, EnumCharEncode displayEncoding) 42
7.2.9 AddBCCRecipient(string address, string display, EnumCharEncode displayEncoding) 42
7.2.10 AddHeaderItem(string itemName, string itemContent) 42
7.2.11 Send 42
7.2.12 SendAsync 42
7.2.13 SendAsyncCancel 42
7.3 Events 42
7.3.1 event SendCompletedEventHandler SendCompleted 43
7.3.2 event OperationFailHandler OperationFailed 43
7.3.3 event EventHandler Disposed 43
8 FTP Client 43
8.1 Properties 43
8.1.1 string ErrorMessage 43
8.1.2 string OperationResponse 43
8.1.3 FtpFileInfo[] FileList 43
8.1.4 string[] FileNames 43
8.1.5 string[] FolderNames 43
8.1.6 string FtpServer 43
8.1.7 string Usename 44
8.1.8 string Password 44

8.1.9 string CurrentServerFolder 44
8.1.10 bool UseBinary 44
8.1.11 bool EnableSsl 44
8.1.12 AuthenticationLevel AuthenticationLevel 44
8.1.13 bool KeepAlive 44
8.1.14 RequestCacheLevel CachePolicy 44
8.1.15 string ConnectionGroupName 44
8.1.16 TokenImpersonationLevel ImpersonationLevel 45
8.1.17 string ProxyName 45
8.1.18 int ProxyPort 45
8.1.19 int ReadWriteTimeout 45
8.1.20 int Timeout 45
8.1.21 bool UsePassive 45
8.2 Methods 45
8.2.1 bool UploadFile(string sourceFilePath, string targetFileName) 45
8.2.2 bool UploadFiles(string sourceFolder, string filePattern, SearchOption searchOption, string targetFolder) 45
8.2.3 bool DeleteFile(string filename) 46
8.2.4 bool Download(string localFolder, string filename) 46
8.2.5 bool CreateFtpFolder(string folderName) 46
8.2.6 bool DeleteFtpFolder(string folderName) 46
8.2.7 bool Rename(string filename, string targetName) 46
8.2.8 bool GetContents(string folderName) 46
8.3 Events 46
8.3.1 event FtpOperationEvent OperationFailed 46
8.3.2 event FtpOperationEvent FileDownloading 47
8.3.3 event FtpOperationEvent FileDownloaded 47
8.3.4 event FtpOperationEvent FileUploading 47
8.3.5 event FtpOperationEvent FileUploaded 47
9 TextBoxNumber 47
9.1 Properties 47
9.1.1 bool DisableValidation 47

9.1.2 string DisplayFormat 47
9.1.3 EnumCulture ValidateCulture 47
9.1.4 EnumNumber TargetType 48
9.1.5 object NumericValue 48
9.1.6 CultureInfo ValidateCultureInfo 48
10 LabelNumber 48
10.1 Properties 48
10.1.1 bool DisableValidation 48
10.1.2 string DisplayFormat 48
10.1.3 EnumCulture ValidateCulture 48
10.1.4 EnumNumber TargetType 48
10.1.5 object NumericValue 49
10.1.6 CultureInfo ValidateCultureInfo 49
11 ButtonKey 49
11.1 Properties 49
11.1.1 string KeyMaps 49
11.1.2 string KeysToSend 49
12 StringTool 49
12.1 Properties 49
12.1.1 string String 49
12.1.2 string MD5Hash 49
12.1.3 EnumCommonCulture FormatCulture 50
12.1.4 string FormattedString 50
12.1.5 string[] Fields 50
12.1.6 string DirectoryName 50
12.1.7 string FileName 50
12.1.8 FileNameWithoutExtension 50
12.1.9 string FullPath 51
12.2 Methods 51
12.2.1 string FormatStringWithValues(params object[] values) 51
12.2.2 SetFieldValue(string fieldName, object value) 51
12.2.3 string Encrypt(string input, string key) 51

12.2.4 static string Decrypt(string input, string key) 51
12.2.5 static string GetMD5Hash(string input) 51
12.2.6 static bool VerifyMd5Hash(string input, string hash) 51
13 Capturer 52
13.1 Properties 52
13.1.1 bool ShowMessagesOnError 52
13.1.2 string VideoDeviceName 52
13.1.3 string AudioDeviceName 52
13.1.4 string VideoCompressor 52
13.1.5 string AudioCompressor 52
13.1.6 string VideoSourceName 52
13.1.7 string AudioSourceName 52
13.1.8 int FrameRate 52
13.1.9 Size FrameSize 53
13.1.10 EnumAudioChannel AudioChannel 53
13.1.11 int AudioSamplingRate 53
13.1.12 short AudioSampleSize 53
13.1.13 int TvTunerChannel 53
13.1.14 TunerInputType TunerInputType 53
13.1.15 string Filename 53
13.1.16 Control PreviewWindow 53
13.1.17 IList<string> VideoDeviceList 53
13.1.18 IList<string> AudioDeviceList 53
13.1.19 IList<Filter> VideoCompressorList 54
13.1.20 IList<Filter> AudioCompressorList 54
13.1.21 IList<Source> VideoSourceList 54
13.1.22 IList<Source> AudioSourceList 54
13.1.23 IList<string> PropertyPages 54
13.1.24 bool Stopped 54
13.1.25 bool Started 54
13.1.26 bool Cued 54
13.2 Methods 54

13.2.1 Cue 54
13.2.2 StartCapture 55
13.2.3 StopCapture 55
13.2.4 DisplayErrors 55
13.2.5 ClearErrorLog 55
13.2.6 SetInputDevices(string videoDevice, string audioDevice) 55
13.2.7 bool SelectInputDevices 55
13.2.8 bool SelectVideoCompression 55
13.2.9 bool SelectAudioCompression 55
13.2.10 bool SelectVideoSource 55
13.2.11 bool SelectAudioSource 55
13.2.12 ShowPropertyPageByIndex(int index) 56
13.2.13 ShowPropertyPageByName(string pageName) 56
13.2.14 bool ShowPropertiesDialogue 56
13.3 Events 56
13.3.1 event EventHandler CaptureCompleted 56
13.3.2 event EventHandler Error 56
14 TreeViewX 56
14.1 Properties 56
14.1.1 Boolean IsCategoryNodeSelected 56
14.1.2 Boolean IsShortcutNodeSelected 57
14.1.3 Boolean IsPropertyNodeSelected 57
14.1.4 TreeNodeShortcut SelectedShortcut 57
14.1.5 TreeNodeX SelectedCategryNode 57
14.1.6 TreeNodeValue SelectedPropertyNode 57
14.1.7 Boolean ShowPropertyNodes 57
14.1.8 Guid TreeViewGuid 57
14.1.9 EnumTreeViewMenu EnabledMenuItems 57
14.1.10 Boolean ReadOnly 57
14.1.11 TreeNodeCollection Nodes 57
14.1.12 String ErrorMessage 58
14.2 Methods 58

14.2.1 AddRootNode(String text) TreeNodeX 58
14.2.2 AddSubNode(String text) TreeNodeX 58
14.2.3 DeleteSelectedCategoryNode(Boolean confirm) 58
14.2.4 DeleteSelectedShortcut(Boolean confirm) 58
14.2.5 DeleteSelectedValue(Boolean confirm) 58
14.2.6 MoveSelectedNodeToRoot 58
14.2.7 CreateNewProperty() TreeNodeValue 59
14.2.8 MoveNodeUp(TreeNode node) 59
14.2.9 MoveNodeDown(TreeNode node) 59
14.2.10 MoveSelectedNodeUp 59
14.2.11 MoveSelectedNodeDown 59
14.2.12 EditNodes() Boolean 59
14.2.13 RemoveAllNodes 59
14.2.14 RemoveAllShortcuts(Guid id) 59
14.2.15 ShowSelectedObjectProperties 59
14.2.16 SaveToXmlDocument() XmlDocument 60
14.2.17 SaveToFile(String filename) Boolean 60
14.2.18 LoadFromFile(String filename) Boolean 60
14.2.19 LoadFromXmlDocument(XmlDocument doc) Boolean 60
14.2.20 SyncShortcuts(TreeNodeX node) 60
14.2.21 GetCategoryNodeById(Guid id) TreeNodeX 60
14.3 Events 60
14.3.1 CategoryNodeSelected - EventHandlerTreeNodeX 60
14.3.2 ShortcutNodeSelected - EventHandlerTreeNodeShortcut 60
14.3.3 PropertyNodeSelected - EventHandlerTreeNodeValue 60
14.3.4 ShortcutMouseClick - MouseEventHandlerTreeNodeShortcut 61
14.3.5 ShortcutMouseDoubleClick - MouseEventHandlerTreeNodeShortcut 61
14.3.6 CategoryNodeMouseClick - MouseEventHandlerTreeNodeX 61
14.3.7 CategoryNodeMouseDoubleClick - MouseEventHandlerTreeNodeX 61
14.3.8 PropertyNodeMouseClick - MouseEventHandlerTreeNodeValue 61
14.3.9 PropertyNodeMouseDoubleClick - MouseEventHandlerTreeNodeValue 61
14.3.10 ShortcutMouseHover - EventHandlerTreeNodeShortcut 61

14.3.11 CategoryNodeMouseHover - EventHandlerTreeNodeX 61
14.3.12 PropertyNodeMouseHover - EventHandlerTreeNodeValue 61
15 TreeNodeX 61
15.1 Properties 62
15.1.1 Guid TreeNodeGuid 62
15.2 List<TypedNamedValue> ValueList 62
15.3 Boolean ShowPropertyNodes 62
15.4 Boolean NextLevelLoaded 62
15.5 TreeNodeCollection Nodes 62
15.5.1 Boolean IsByShortcut 62
15.6 Methods 62
15.6.1 RemoveAllShortcuts(Guid id) 62
15.6.2 GetCategoryNodeById(Guid id) TreeNodeX 62
15.6.3 GetValue(String name) Object 63
15.6.4 SetValue(String name,Object value) 63
15.6.5 ClearValues 63
15.6.6 CreateValue(Type t) TreeNodeValue 63
15.6.7 AddValue(String name,Type type,Object value) 63
16 TypedValue 63
16.1 Properties 63
16.1.1 Type ValueType 63
16.1.2 object Value 63
16.2 Methods 63
16.2.1 ResetValue 63
17 TypedNamedValue 63
17.1 Properties 64
17.1.1 string Name 64
17.1.2 TypedValue Value 64
18 WebBrowserControl 64
18.1 Properties 64
18.1.1 PopupLevel - EnumPopupLevel 64
18.2 Methods 64

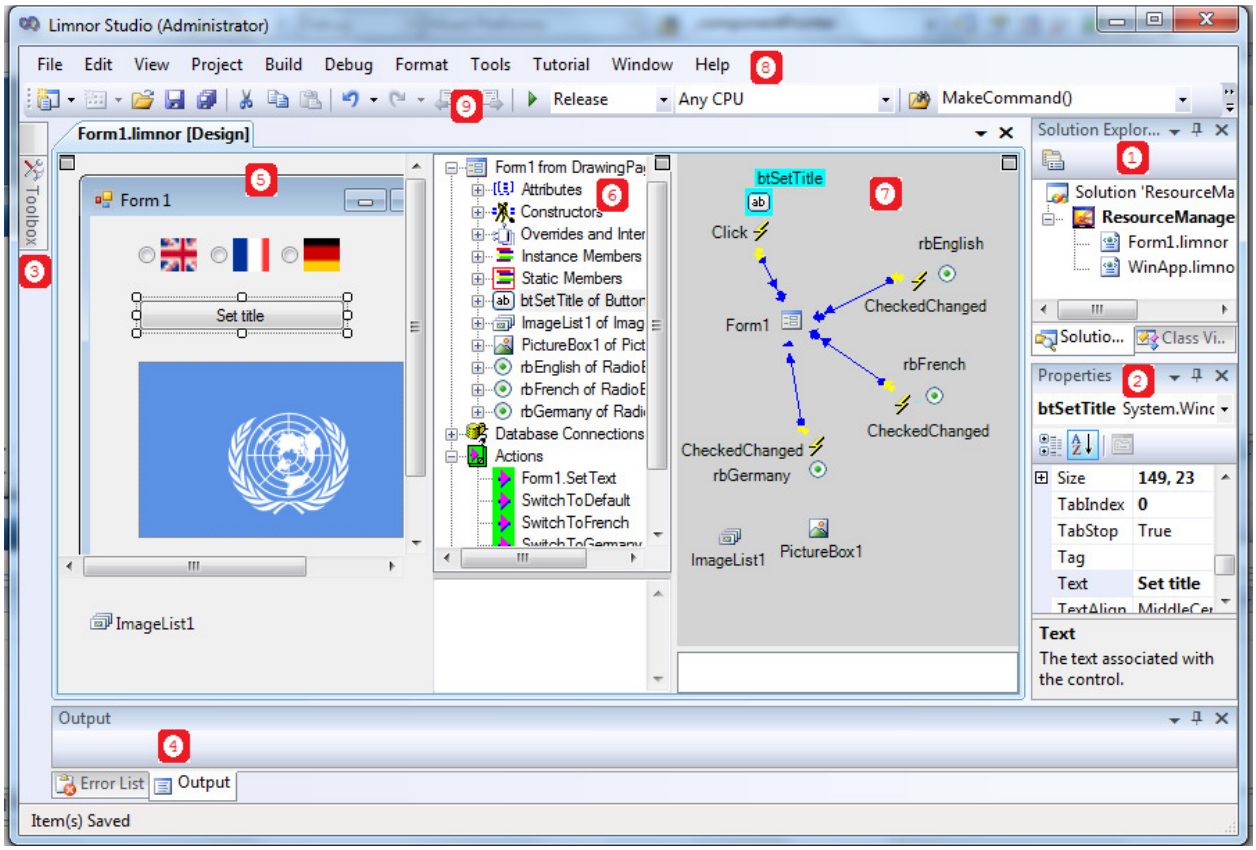
18.3 Events 64
18.3.1 Downloading - EventHandler 64
18.3.2 DownloadComplete - EventHandler 64
18.3.3 StartNavigate - EventHandler<BrowserNavigationEventArgs> 64
18.3.4 StartNewWindow – EventHandler<BrowserNavigationEventArgs> 65
18.3.5 ApplicationQuit - EventHandler 65
19 On-Screen-Keyboard 65
19.1 Properties 65
19.1.1 OskLoaded - Boolean 65
19.1.2 ErrorMessage - String 65
19.1.3 Location - Point 65
19.1.4 DisableCloseButton - Boolean 65
19.1.5 Visible - Boolean 65
19.1.6 Movable - Boolean 65
19.2 Methods 66
19.2.1 StartOsk 66
19.2.2 CloseOsk 66
19.3 Events 66
20 Copy Protector 66
20.1 Properties 66
20.1.1 ApplicationId - Guid 66
20.1.2 SupportMessage - String 66
20.2 Methods 66
20.2.1 VerifyLicense 66
21 Printer Manager 67
21.1 Properties 67
21.1.1 Printers - StringCollection 67
21.2 Methods 67
21.2.1 SetDefaultPrinter(String printerName) 67
21.2.2 SetDefaultPrinterByUI(Form caller) 67
21.3 Events 67
21.3.1 Disposed - EventHandler 67

1 Limnor Studio IDE

1.1 Introduction to Limnor Studio IDE

Limnor Studio is an Integrated Development Environment (IDE) for developing computer software visually without using computer languages. For visual and codeless programming concepts, see Users' Guide from http://www.limnor.com/index.html?Doc=studio_userGuide.html.

Limnor Studio IDE user interface is based on Microsoft Visual Studio. By default the UI looks like the screenshot below.

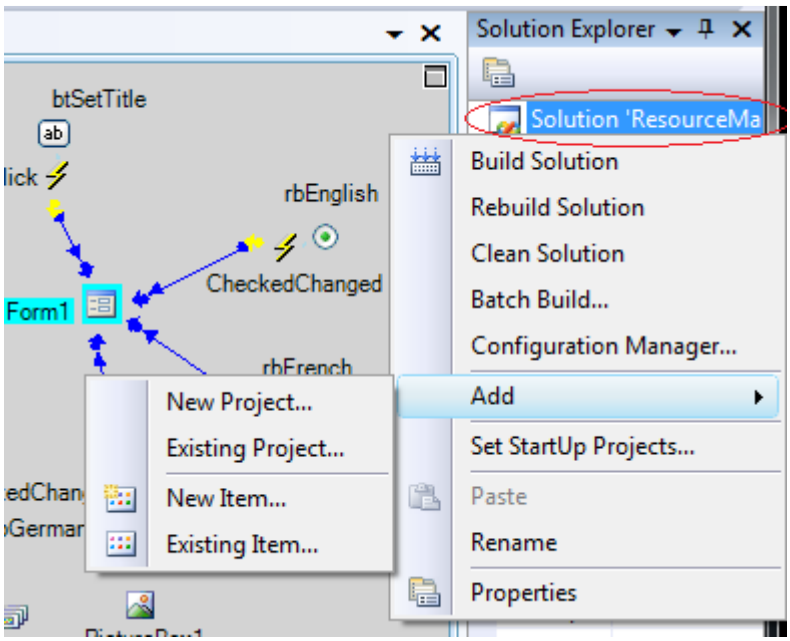


1. Solution Explore/Project Explorer
2. Properties Window
3. Toolbox
4. Messages Windows
5. User Interface Designer
6. Object Explorer
7. Event Path
8. Menus
9. Tools

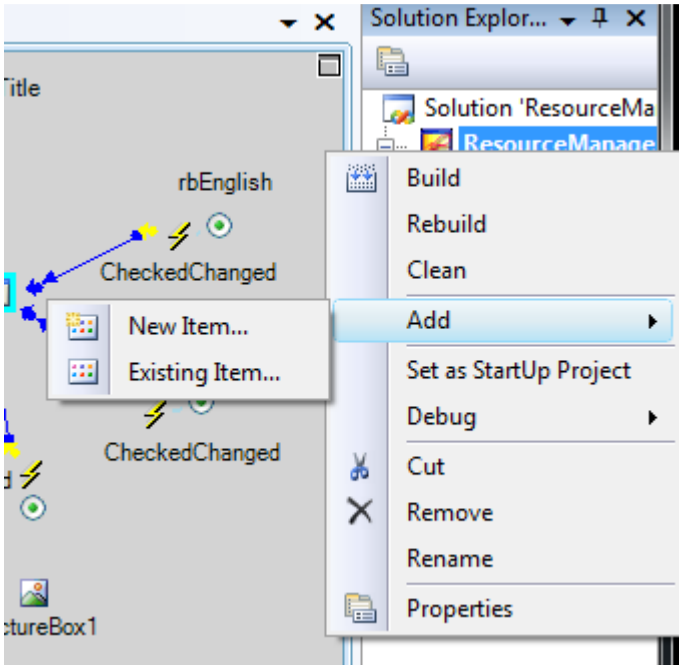
1.2 Solution Explorer/Project Explorer

A Solution may contain one or more Projects. A project is for generating an executable program or a dynamic link library.

Projects are listed under the Solution Explorer. To add projects into the Solution, right-click the Solution and choose “Add”:



Files belonging to a project are listed under the Project Explorer. To add files into the project, right-click the project, choose “Add”:

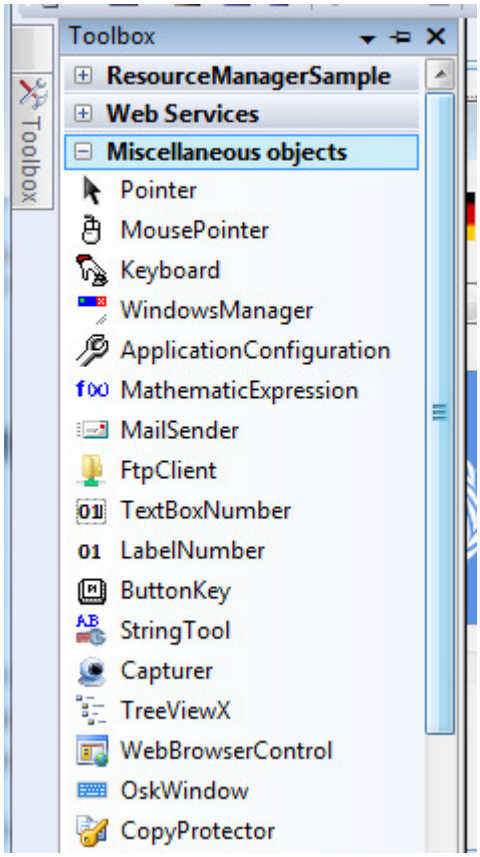


1.3 Properties Windows

The Properties Window shows the properties of the selected object. Properties may also be changed in the Properties Window.

1.4 Toolbox

Toolbox contains objects which can be inserted into the class under development.



1.5 Messages Windows

Compiling messages, error messages, status, etc., can be found in these windows.

1.6 User Interface Designer

User Interface Designer is for designing user interface.

1.7 Object Explorer

An object may contain other objects. An object may have properties, methods, and events. Each property is an object and thus may have its own properties, methods and events. Object Explorer shows such hierarchical member relationships.

1.8 Event Path

When an action is assigned to an event, at runtime, the action will be executed when the event occurs. Event Path shows the event-action relationships.

1.9 Menus

File

New

Project -- Create a new project.

Open

Project – Opens an existing project

Add

New project – Add a new project to the solution

Existing project – Add an existing project to the solution

Close – Closes current file

Close Solution – Closes the solution

Save all – Save all modifications to files

Recent projects – opens a recently used project

Exit – Shut down the Limnor Studio IDE

Edit

Undo – Undo UI modification

Redo – Redo the undo

Cut – Cut the selected object on UI Designer

Copy – Copy the selected object on UI Designer

Paste – Paste to the UI designer

Delete – Delete the selected object on UI Designer

View

Solution Explorer – Show the Solution Explorer

Tab Order – Show the tab order on UI Designer. At runtime, when the user types Tab key, the control on the UI gets the focus based on the tab order

View Error Log File – Display the contents of the error log file

View Log File – Display the contents of the compiler log file

Properties Window – Show the Properties Window

Toolbox – Show the Toolbox

Full Screen – Show Limnor Studio in full screen.

Project

Manage MySQL Database – manages MySQL databases, editing tables, fields, indexes

Manage Microsoft Access Database -- manages Access databases, editing tables, fields, indexes

Add Toolbox Item – Add classes from software libraries to the Toolbox. ActiveX and .Net classes can be added.

Add Web Service Proxy – Add a web service proxy to the project so that the project may call the web service

Remove Web Service Proxy – Remove a web service proxy from the project.

Update Web Service Proxy – When a web service's interface is changed its proxy should be updated

Manage Visual Programming Systems – Limnor Studio is a visual programming platform for hosting visual programming systems, such as UI Designer, Object Explorer, and Event Path. Choose this menu to add new visual programming systems to Limnor Studio, delete visual programming systems, enable/disable visual programming systems.

Database Connections – Manages database connections for the project

License Manager – Issue licenses for the application for copy-protection

Resource Manager – Managing resources for the project. Resource Manager allows embedding resources (text, image, icon and other files) in software. Since the resources are embedded in the software (*.EXE/*.DLL), it is easy to distribute the software to your customers. You do not have to distribute image files, text files, icon files, etc. Another important usage of the Resource Manager is to help localizing your software. It helps you managing multiple language resources. Multi-language resources are embedded in your software and correct resources are used based on the culture at runtime. Your software may instantly switch language at runtime. For example, a kiosk at an international airport should allow the user to choose language for the user interface.

Set as StartUp Project – Set the current select project as the project to be debugged.

{Project name} Properties – Shows/edits project properties. See http://www.limnor.com/studio_x86.html for setting compiling target to generate 32-bit programs. If a project uses 32-bit libraries then it must be compiled into 32-bit.

Build

Build Solution – Compile modified projects

Rebuild Solution – Compile all projects

Build {project name} – Compile modified files in the project

Rebuild {project name} – Compile all files in the project

Debug

Windows

Breakpoints – Show all breakpoints

Output – Show output window

Immediate – Show Immediate window

Start Debugging – Run the startup project for debugging

Start Without Debugging -- Run the startup project without debugging

Attach to Process – Attach debugger to a process

Exceptions – Show exceptions

Step Into – Step into a method call

Step Over – Step over a call.

Toggle Breakpoint – When a C# source code is opened, set or remove breakpoint

Delete All Breakpoints – Delete all breakpoints.

See http://www.limnor.com/support/debug.swf.html and http://www.limnor.com/support/Debug%20Windows%20Service.pdf for more information on debugging.

Tutorial – Show tutorial lessons.

1.10 Tools

 -- Save current file

 -- Save all files

 -- Cut the selected UI objects

 -- Copy the selected UI objects

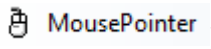
 -- Paste to the UI Designer

 -- Undo

 -- Redo

 -- Run debug

2 Mouse Pointer



This object provides mouse operations.

2.1 Methods

2.1.1 ClipCursor (Rectangle rc)

It confines the cursor to a rectangular area on the window. If a subsequent cursor position (set by the SetCursorPos function or the mouse) lies outside the rectangle, the system automatically adjusts the position to keep the cursor inside the rectangular area.

Parameters

rc – the rectangle for confining the cursor.

2.1.2 ClearCursorClip

Remove cursor clipping so that the cursor is free to move anywhere on the screen

2.1.3 int SetCursorPosition(int x, int y)

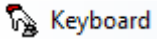
Moves the cursor to the specified window coordinates. If the new coordinates are not within the screen rectangle set by the most recent ClipCursor function call, the system automatically adjusts the coordinates so that the cursor stays within the rectangle. Returns 0 if successful or error code otherwise.

Parameters

x – the x coordinate of the cursor position

y – the y coordinate of the cursor position

3 Keyboard



This object provides keyboard operations.

3.1 Properties

3.1.1 bool EnableHotKeys

Gets and sets a Boolean value indicating whether the hotkey events will occur. If this property is false then the actions assigned to hotkeys will not be executed.

3.1.2 HotKeyList HotKeys

Gets and sets Hotkeys for assigning actions

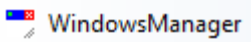
3.1.3 RemoveHotKey(Key key)

Remove the specified key from the hot key list

Parameter

key – key to be removed

4 Windows Manager



This object provides Windows Management operations. If this component is used by a 32-bit application then the events of this component work for windows of 32-bit applications. If this component is used by a 64-bit application then the events work for windows from 64-bit applications.

4.1 Properties

4.1.1 Controls - WindowControl[]

The child windows contained in the window represented by FoundWindow

4.1.2 EventsEnabled - Boolean

Gets and sets a Boolean indicating whether the events are enabled. Disabling events makes the program run faster.

4.1.3 FoundWindow - IntPtr

The window handle found by calling GetWindowByTitle, GetWindowUnderMouse, CaptureWindowImage, CaptureWindowUnderMouse, and other methods

4.1.4 LastErrorCode - Int32

Windows API error code for the last failed operation

4.1.5 LastErrorMessage - String

Windows API error message for the last failed operation

4.1.6 WindowBounds - Rectangle

Gets and sets the location and size of the FoundWindow

4.1.7 WindowLocation - Point

Gets and sets the location of the FoundWindow

4.1.8 WindowSize - Size

Gets and sets the size of the FoundWindow

4.2 Methods

4.2.1 static Image CropImage(Image image, Rectangle cropRectangle)

Crop image according the specified rectangle.

Parameters:

Image – the image to be cropped.

cropRectangle – the rectangle for cropping.

4.2.2 static bool SaveBitmapToFile(Bitmap bitmap, string filename, EnumImageFormat format, bool cropImage)

Save a bitmap image to a file

Parameters:
bitmap – bitmap image to be saved
filename – file name for saving the image
format – image format. It can be Bmp, Emf, Exif, Gif, Icon, Jpeg, MemoryBmp, Png, Tiff, Wmf
cropImage – If it is true then a dialogue box will appear for doing image cropping visually.

4.2.3 static Bitmap CaptureWindowImage(IntPtr hWnd)

Capture window image to a bitmap

Parameter

hWnd – the handle of the window to be captured

4.2.4 static Bitmap CaptureControlImage(Control control)

Capture control image to a bitmap

Parameter

control – the control to be captured

4.2.5 IntPtr GetWindowByTitle(string windowTitle)

Find the Window by its title

Parameter

windowTitle – title of the window to be found

4.2.6 IntPtr GetWindowTitleStartWith(string titleStart)

Find a Window with its title starting with a specified string

Parameter

titleStart – the starting text of the title of the window to be found

4.2.7 IntPtr GetWindowTitleEndWith(string titleEnd)

Find a Window with its title ending with a specified string

Parameter

titleEnd – the ending text of the title of the window to be found

4.2.8 IntPtr GetWindowTitleContains(string titlePart)

Find a Window with its title containing a specified string

Parameter

titlePart – the part of text of the title of the window to be found

4.2.9 string GetControlText(int controlId)

Get the text of the control identified by control id. The parent window must be found first by calling FoundWindow

Parameter

controlId – the id of the control

4.2.10 Bitmap CaptureWindowImage(string windowTitle)

Find the Window by its title and capture the Window image into a bitmap

Parameter

windowTitle – the title of the window

4.2.11 bool SaveWindowImageToFile(string windowTitle, string filename, EnumImageFormat format, bool cropImage)

Find the Window by its title, capture the Window image, and save the image to a file

Parameters

windowTitle – the title of the window
filename – file name for saving the image
format – image format. It can be Bmp, Emf, Exif, Gif, Icon, Jpeg, MemoryBmp, Png, Tiff, Wmf
cropImage – If it is true then a dialogue box will appear for doing image cropping visually.

4.2.12 PrintControl(Control control, string documentName, bool preview)

Print the specified control as it appears on the screen

Parameters

control – the control being printed
documentName – print job name
preview – indicates whether to preview the printing

4.2.13 IntPtr GetWindowUnderMouse()

Find the Window under the mouse pointer

4.2.14 Bitmap CaptureWindowUnderMouse()

Find the Window under the mouse pointer and capture the window image

4.2.15 bool SaveWindowUnderMouseImageToFile(string filename, EnumImageFormat format, bool cropImage)

Find the Window under the mouse pointer, capture the window image, and save the image to a file

Parameter

filename – file name for saving the image

format – image format. It can be Bmp, Emf, Exif, Gif, Icon, Jpeg, MemoryBmp, Png, Tiff, Wmf
cropImage – If it is true then a dialogue box will appear for doing image cropping visually.

4.2.16 Bitmap CaptureScreen()

Capture the whole screen image

4.2.17 bool SaveScreenImageToFile(string filename, EnumImageFormat format, bool cropImage)

Capture the whole screen image, and save the image to a file

Parameter

filename – file name for saving the image
format – image format. It can be Bmp, Emf, Exif, Gif, Icon, Jpeg, MemoryBmp, Png, Tiff, Wmf
cropImage – If it is true then a dialogue box will appear for doing image cropping visually.

4.3 Events

4.3.1 ApplicationActivate - EventHandlerApplicationActivate

Occurs when a window belonging to a different application than the active window is about to be activated. The message is sent to the application whose window is being activated and to the application whose window is being deactivated.

4.3.2 EnterSizeMove - EventHandler

Occurs one time to a window after it enters the moving or sizing modal loop. The window enters the moving or sizing modal loop when the user clicks the window's title bar or sizing border, or when the window passes the WM_SYSCOMMAND message to the DefWindowProc function and the wParam parameter of the message specifies the SC_MOVE or SC_SIZE value. The operation is complete when DefWindowProc returns.

4.3.3 ExitSizeMove - EventHandler

Occurs one time to a window, after it has exited the moving or sizing modal loop. The window enters the moving or sizing modal loop when the user clicks the window's title bar or sizing border, or when the window passes the WM_SYSCOMMAND message to the DefWindowProc function and the wParam parameter of the message specifies the SC_MOVE or SC_SIZE value. The operation is complete when DefWindowProc returns.

4.3.4 SystemCommand - EventHandlerSystemCommand

Occurs when the user chooses a command from the Window menu (formerly known as the system or control menu) or when the user chooses the maximize button, minimize button, restore button, or close button.

4.3.5 WindowActivate - EventHandlerWindowActivate

Occurs to both the window being activated and the window being deactivated. If the windows use the same input queue, the message is sent synchronously, first to the window procedure of the top-level window being deactivated, then to the window procedure of the top-level window being activated. If the windows use different input queues, the message is sent asynchronously, so the window is activated immediately.

4.3.6 WindowGotFocus - EventHandlerWindowFocus

Occurs when a window has gained the keyboard focus.

4.3.7 WindowKillFocus - EventHandlerWindowFocus

Occurs to a window immediately before it loses the keyboard focus.

4.3.8 WindowPositionChanged - EventHandlerWindowPositionChange

Occurs to a window whose size, position, or place in the Z order has changed as a result of a call to the SetWindowPos function or another window-management function.

4.3.9 WindowPositionChanging - EventHandlerWindowPositionChange

Occurs to a window whose size, position, or place in the Z order is about to change as a result of a call to the SetWindowPos function or another window-management function.

4.3.10 WindowResized - EventHandlerWindowResized

Occurs to a window after its size has changed.

4.3.11 WindowResizeOrMove - EventHandler

Occurs when the window is moved or resized

4.3.12 WindowSizing - EventHandlerWindowSizing

Occurs to a window that the user is resizing. By processing this message, an application can monitor the size and position of the drag rectangle and, if needed, change its size or position.

4.4 Classes for Event Arguments

4.4.1 EventArgsApplicationActivate

It contains event arguments for event ApplicationActivate

It is derived from EventArgs

Properties

IsActivated - Boolean

Gets a Boolean indicating whether the window is being activated or deactivated. This value is TRUE if the window is being activated; it is FALSE if the window is being deactivated.

TheOtherThreadId - Int32

Gets the thread identifier of the other application. If IsActivated is TRUE, it is the identifier of the thread that owns the window being deactivated. If IsActivated is FALSE, it is the identifier of the thread that owns the window being activated. It can be 0.

4.4.2 EventArgsSystemCommand

It contains event arguments for event SystemCommand

It is derived from EventArgs

Properties

SystemCommand - EnumSystemCommand

Gets the type of system command requested

4.4.3 EventArgsWindowActivate

It contains event arguments for event WindowActivate

It is derived from EventArgs

Properties

IsActivated - Boolean

Gets a Boolean indicating whether the window is being activated or deactivated

IsActivatedByMouse - Boolean

Gets a Boolean indicating whether the window is being activated by mouse click

IsMinimized - Boolean

Gets a Boolean indicating the minimized state of the window being activated or deactivated. A True value indicates the window is minimized.

TheOtherWindowHandle - IntPtr

Gets a handle to the window being activated or deactivated, depending on the value of the IsActivated. If IsActivated is False then it is the handle to the window being activated. If IsActivated is True then it is the handle to the window being deactivated. This handle can be NULL

4.4.4 EventArgsWindowFocus

It contains event arguments for events WindowGotFocus and WindowKillFocus

It is derived from EventArgs

Properties

GotFocus - Boolean

Gets a Boolean indicating whether it is getting or losing keyboard input focus.

TheOtherWindowHandle - IntPtr

Gets a handle to the window that has lost/gained the keyboard focus. If the current window gets the focus then TheOtherWindowHandle loses focus. If the current window loses focus then TheOtherWindowHandle gains the focus. It can be NULL.

4.4.5 EventArgsWindowPositionChange

It contains event arguments for events WindowPositionChanging and WindowPositionChanged

It is derived from EventArgs

Properties

ChangeStyle - EnumWindowPositionChangeStyle

Gets the style for the window postion change

Handle - IntPtr

Gets a handle to the window.

InsertAfter - IntPtr

Gets a handle to the window presenting the position of the window in Z order (front-to-back position). If this member is not 0 then it is a handle to the window behind which this window is placed. If this member is 0 then WindowRelativePosition property represents the window position change.

IsMoved - Boolean

Gets a Boolean indicating whether the window is moved

IsResized - Boolean

Gets a Boolean indicating whether the window is resized

IsResizedOrMoved - Boolean

Gets a Boolean indicating whether the window is resized or moved

Location - Point

Gets the location of the window

Size - Size

Gets the size of the window

WindowRelativePosition - EnumWindowRelativePostion

Gets the window relative position if InsertAfter is 0

4.4.6 EventArgsWindowSized

It contains event arguments for event WindowResized

It is derived from EventArgs

Properties

NewSize - Size

Gets the new size

ResizeType - EnumWindowSizeType

Gets the type of resizing requested

4.4.7 EventArgsWindowSizing

It contains event arguments for event WindowSizing

It is derived from EventArgs

Properties

Rectangle - Rectangle

Gets the drag rectangle

SizeEdge - EnumSizeEdge

Gets the edge of the window that is being sized.

4.5 Enumerators used in events

4.5.1 EnumSystemCommand

It is a list of system commands.

```
public enum EnumSystemCommand : int
{
    SC_CLOSE = 0xF060, // Closes the window.
```

```
        SC_CONTEXTHELP = 0xF180, // Changes the cursor to a question mark
with a pointer. If the user then clicks a control in the dialog box, the
control receives a WM_HELP message.
        SC_DEFAULT = 0xF160, // Selects the default item; the user double-
clicked the window menu.
        SC_HOTKEY = 0xF150, // Activates the window associated with the
application-specified hot key. The lParam parameter identifies the window to
activate.
        SC_HSCROLL = 0xF080, // Scrolls horizontally.
        SCF_ISSECURE = 0x00000001, // Indicates whether the screen saver is
secure.
        SC_KEYMENU = 0xF100, // Retrieves the window menu as a result of a
keystroke. For more information, see the Remarks section.
        SC_MAXIMIZE = 0xF030, // Maximizes the window.
        SC_MINIMIZE = 0xF020, // Minimizes the window.
        SC_MONITORPOWER = 0xF170, // Sets the state of the display. This
command supports devices that have power-saving features, such as a battery-
powered personal computer.
        //The lParam parameter can have the following values:
        //-1 (the display is powering on)
        //1 (the display is going to low power)
        //2 (the display is being shut off)
        SC_MOUSEMENU = 0xF090, // Retrieves the window menu as a result of a
mouse click.
        SC_MOVE = 0xF010, // Moves the window.
        SC_MOVE_HTCAPTION = 0xF012, //
        SC_NEXTWINDOW = 0xF040, // Moves to the next window.
        SC_PREVWINDOW = 0xF050, // Moves to the previous window.

        SC_RESTORE = 0xF120, // Restores the window to its normal position
and size.
        SC_SCREENSAVE = 0xF140, // Executes the screen saver application
specified in the [boot] section of the System.ini file.
        SC_SIZE = 0xF000, // Sizes the window.
        SC_SIZE_WEST = 0xF001, //Resize from left
        SC_SIZE_EAST = 0xF002, // (Resize from right)
        SC_SIZE_NORTH = 0xF003,// (Resize from up)
        SC_SIZE_NORTH_WEST = 0xF004, // (Lock the bottom right corner of the
form, the up left corner move for resize)
        SC_SIZE_NORTH_EAST = 0xF005, // (Same from bottom left corner)
        SC_SIZE_NORTH_SOUTH = 0xF006, // (Lock up right and left border,
resize other)
        SC_SIZE_SOUTH_WEST = 0xF007, // (Lock up and right border, resize
other border)
        SC_SIZE_SOUTH_EAST = 0xF008, //Lock left and up border and resize
other
        SC_TASKLIST = 0xF130, // Activates the Start menu.
        SC_VSCROLL = 0xF070, // Scrolls vertically.
    }
```

4.5.2 EnumSizeEdge

It is a list of window resize edges.

```
    public enum EnumSizeEdge : int
    {
        WMSZ_BOTTOM = 6, // Bottom edge
```

```
        WMSZ_BOTTOMLEFT = 7, // Bottom-left corner
        WMSZ_BOTTOMRIGHT = 8, // Bottom-right corner
        WMSZ_LEFT = 1,// Left edge
        WMSZ_RIGHT = 2,// Right edge
        WMSZ_TOP = 3,// Top edge
        WMSZ_TOPLEFT = 4,// Top-left corner
        WMSZ_TOPRIGHT = 5,// Top-right corner
    }
```

4.5.3 EnumWindowSizeType

```
    It is a list of window resize types.
    public enum EnumWindowSizeType : int
    {
        SIZE_MAXHIDE = 4,// Message is sent to all pop-up windows when some
other window is maximized.
        SIZE_MAXIMIZED = 2, // The window has been maximized.
        SIZE_MAXSHOW = 3,// Message is sent to all pop-up windows when some
other window has been restored to its former size.
        SIZE_MINIMIZED = 1, // The window has been minimized.
        SIZE_RESTORED = 0, // The window has been resized, but neither the
SIZE_MINIMIZED nor SIZE_MAXIMIZED value applies.
    }

    [Flags]
    public enum EnumWindowPositionChangeStyle : int
    {
        SWP_DRAWFRAME = 0x0020,// Draws a frame (defined in the window's
class description) around the window. Same as the SWP_FRAMECHANGED flag.
        SWP_FRAMECHANGED = 0x0020,// Sends a WM_NCCALCSIZE message to the
window, even if the window's size is not being changed. If this flag is not
specified, WM_NCCALCSIZE is sent only when the window's size is being changed.
        SWP_HIDEWINDOW = 0x0080, // Hides the window.
        SWP_NOACTIVATE = 0x0010, // Does not activate the window. If this
flag is not set, the window is activated and moved to the top of either the
topmost or non-topmost group (depending on the setting of the hwndInsertAfter
member).
        SWP_NOCOPYBITS = 0x0100,// Discards the entire contents of the client
area. If this flag is not specified, the valid contents of the client area
are saved and copied back into the client area after the window is sized or
repositioned.
        SWP_NOMOVE = 0x0002, // Retains the current position (ignores the x
and y members).
        SWP_NOOWNERZORDER = 0x0200,// Does not change the owner window's
position in the Z order.
        SWP_NOREDRAW = 0x0008,// Does not redraw changes. If this flag is set,
no repainting of any kind occurs. This applies to the client area, the
nonclient area (including the title bar and scroll bars), and any part of the
parent window uncovered as a result of the window being moved. When this flag
is set, the application must explicitly invalidate or redraw any parts of the
window and parent window that need redrawing.
        SWP_NOREPOSITION = 0x0200,// Does not change the owner window's
position in the Z order. Same as the SWP_NOOWNERZORDER flag.
        SWP_NOSENDCHANGING = 0x0400,// Prevents the window from receiving the
WM_WINDOWPOSCHANGING message.
        SWP_NOSIZE = 0x0001,// Retains the current size (ignores the cx and
cy members).
```



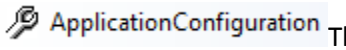
```
        SWP_NOZORDER = 0x0004,// Retains the current Z order (ignores the
hwndInsertAfter member).
        SWP_SHOWWINDOW = 0x0040,// Displays the window.
    }
```

4.5.4 EnumWindowRelativePostion

It is a list of window relative position types.

```
    public enum EnumWindowRelativePostion : int
    {
        HWND_BOTTOM = 1,// Places the window at the bottom of the Z order. If
the hWnd parameter identifies a topmost window, the window loses its topmost
status and is placed at the bottom of all other windows.
        HWND_NOTOPMOST = -2,// Places the window above all non-topmost
windows (that is, behind all topmost windows). This flag has no effect if the
window is already a non-topmost window.
        HWND_TOP = 0,// Places the window at the top of the Z order.
        HWND_TOPMOST = -1,// Places the window above all non-topmost windows.
The window maintains its topmost position even when it is deactivated.
    }
```

5 Application Configuration



This object lets applications save and load runtime configurations.

5.1 Properties

5.1.1 Guid ApplicationGuid

This is a Guid uniquely identifies the application

5.1.2 string ConfigurationProfile

The application configuration profile currently used

5.1.3 string Filename

Configuration file path

5.1.4 CategoryList Configurations

Definitions for the application configurations. Each configuration data has a name and belongs to a category.

5.2 5.2 Methods

5.2.1 string ChangePassword(string name, string password, string newPassword)

Changes the password for the named profile.

Returns: error message

Parameters

name – the name for the profile

password – the current password

newPassword – the new password

5.2.2 bool CreateNamedProfile(string name, string password)

Create a named application configuration profile. Password is optional. If the application configuration profile exists then this method returns false.

5.2.3 ExecuteLoadingConfigurations

It fires event LoadingConfigurations and thus executes all the actions assigned to this event. Usually the actions assigned to this event are for loading configuration values from the configuration file into the application.

5.2.4 ExecuteSavingConfigurations

It fires event SavingConfigurations and thus execute all actions assigned to this event. Usually the actions assigned to this event are for saving configuration values to the configuration file. The purpose of executing those actions is to saves all configuration values from the application into the configuration file.

5.2.5 CreateProfile(Form caller)

Create a named application configuration profile or an application configuration profile for the current user.

Give options for creating user-specific profile if it does not exist, or creating a named-profile.

Parameters

caller – the form from which this method is called. It is to be used as the owner of message boxes

5.2.6 UseFactoryProfile

Switch to use the default application configurations

5.2.7 UseUserProfile

Switch to use the application configuration profile for the current user. If it does not exist then it will be created.

5.2.8 string LogOnProfile(string name, string password)

Switch to a named application configuration profile. The profile must exist. It returns an error message if log on fails.

5.2.9 bool LogOnProfileWithUI(Form caller)

Use a dialogue to select desired application configuration profile. It returns false if the dialogue is canceled.

Parameters

caller – the form from which this method is called. It is to be used as the owner of the dialogue box

5.2.10 ChangePasswordByUI(Form caller)

Use a dialogue box to change password.

caller – the form from which this method is called. It is to be used as the owner of the dialogue box

5.2.11 string GetSetting(string name)

This method is obsolete. Gets application setting by name from the default section.

5.2.12 SetSetting(string name, object value)

This method is obsolete. Set application setting within the default section

5.2.13 Save

Save the settings

5.3 Events

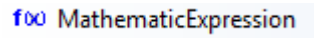
5.3.1 static event EventHandler LoadingConfigurations

It occurs when a configuration file is loaded. It also occurs when ExecuteLoadingConfigurations method is called. Usually all actions for loading configurations into the application are assigned to this event.

5.3.2 static event EventHandler SavingConfigurations

It occurs when ExecuteSavingConfigurations method is executed. Usually all actions for saving configuration values are assigned to this event.

6 Mathematic Expression



This control shows/edits an expression graphically. It can be a math expression for numeric value, a logic expression for Boolean, or a text expression for string. Evaluation of the expression (calculations) can be done at runtime to get results from variable inputs. Inputs can be linked to properties from objects.

6.1 Properties

6.1.1 object Result

The evaluation result of the math expression by executing Execute() method

6.1.2 string XmlString

Gets an Xml string representing this math expression

6.1.3 bool Compiled

Gets a Boolean indicating whether the math expression is compiled and ready to do calculation

6.1.4 FormulaProperty Formula

The formula for this math expression

6.1.5 int VariableCount

The number of the input parameters for this math expression

6.1.6 string[] VariableNames

The names of the input parameters for this math expression

6.1.7 string LastError

Error message of the last operation error

6.2 6.2 Methods

6.2.1 bool EditFormula

Launch math expression editor for modify the math expression. It returns False if the user cancels the editing.

6.2.2 SaveToXml(string xmlFile)

Save math expression to an xml file

6.2.3 LoadXml(string xmlFile)

Load math expression from xml file

6.2.4 AddDrawingSurface(Control surface, Point location)

Calling this function will cause the math expression being displayed on the specified control starting at the location.

6.2.5 RemoveDrawingSurface(Control surface)

Stop displaying the math expression on the specified control

6.2.6 Image CreateMathExpressionImage(Graphics g)

Create math expression image

6.2.7 IMathEditor CreateEditor(Rectangle rc)

Create math expression editor

6.2.8 Compile

Compile the math expression

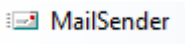
6.2.9 ShowSourceCode

Display source code for the math expression

6.2.10 object Execute

Execute the math expression and return the calculation result

7 Mail Sender



This component can be used to send emails with attachments and embedded images. It supports Secure Socket Layer (SSL) connections. It supports authentication request of SMTP servers.

7.1 Properties

7.1.1 SmtpClient Smtp

Gets the object for sending the email

7.1.2 string MailServer

Gets or sets the name or IP address of the host used for SMTP transactions

7.1.3 int Port

Gets or sets the port used for SMTP transactions

7.1.4 bool EnableSsl

Specify whether Secure Socket Layer (SSL) is used to encrypt the connection

7.1.5 int Timeout

Gets or sets a value that specifies the amount of time after which a synchronous Send action times out

7.1.6 bool UseDefaultCredentials

Gets or sets a value that controls whether DefaultCredentials are sent with requests

7.1.7 ServicePoint ServicePoint

Gets the network connection used to transmit the e-mail message

7.1.8 string TargetName

Gets or sets a string that represents the SPN to use for authentication when using extended protection to connect to an SMTP mail server. This is only supported by Microsoft .Net 4 and later

7.1.9 X509CertificateCollection ClientCertificates

Gets a value specifies which certificates should be used to establish the Secure Socket Layer (SLL) connection

7.1.10 SmtpDeliveryMethod DeliveryMethod

Gets or sets an object specifying how the outgoing email messages will be handles

7.1.11 string PickupDirectoryLocation

Gets or sets the folder where applications save mail messages to be processed by the local SMTP server

7.1.12 string SenderAddress

Gets or sets sender email address

7.1.13 string SenderDisplay

Gets or sets sender display name

7.1.14 EnumCharEncode SenderDisplayEncode

Gets or sets sender display name encoding

7.1.15 string FromAddress

Gets or sets "From" email address

7.1.16 string FromDisplay

Gets or sets "From" display name

7.1.17 EnumCharEncode FromDisplayEncode
Gets or sets "From" display name encoding

7.1.18 string ReplyToAddress
Gets or sets "Reply to" email address

7.1.19 string ReplyToDisplay
Gets or sets "Reply to" display name

7.1.20 EnumCharEncode ReplyToDisplayEncode
Gets or sets "Reply to" display name encoding

7.1.21 string Recipients
Gets mail recipients separated by ";"

7.1.22 string CCRecipients
Gets mail CC recipients separated by ";"

7.1.23 string BccRecipients
Gets mail BCC recipients separated by ";"

7.1.24 MailPriority Priority
Gets or sets the priority of this email message

7.1.25 bool IsBodyHtml
Gets or sets a value indicating whether the mail message body is in HTML

7.1.26 DeliveryNotificationOptions DeliveryNotificationOptions
Gets or sets the delivery notifications for this email message.

7.1.27 string Subject
Gets or sets the subject line for this e-mail message

7.1.28 EnumCharEncode SubjectEncoding
Gets or sets the encoding used for the subject content of this email message

7.1.29 string Body
Gets or sets the message body

7.1.30 EnumCharEncode BodyEncoding
Gets or sets the encoding used to encode the message body

7.1.31 string UserDomain
Gets or set user domain. UserDomain, UserAccount and UserPassword are used as the credential for using the SMTP server when UseDefaultCredentials is false

7.1.32 string UserAccount
Gets or sets User account. UserDomain, UserAccount and UserPassword are used as the credential for using the SMTP server when UseDefaultCredentials is false")]

7.1.33 string UserPassword
Gets or sets user password. UserDomain, UserAccount and UserPassword are used as the credential for using the SMTP server when UseDefaultCredentials is false")]

7.1.34 string[] Attachments
Gets or sets names for the files to be attached to the email

7.2 Methods

7.2.1 ClearEmbededImages
Remove all embedded images.

7.2.2 RemoveEmbededImage(string id)
Remove the embedded image identified by id.

7.2.3 AddEmbededImage(string id, string filePath)

Add an image file identified by filePath to the embedded image list, giving an id to identify the embedded image.

7.2.4 ClearAttachments

Remove all attached files.

7.2.5 RemoveAttachment(string filename)

Remove the attached file identified by filename.

7.2.6 AddAttachment(string filename)

Add a file identified by filename as to the attachments.

7.2.7 AddRecipient(string address, string display, EnumCharEncode displayEncoding)

Add a recipient to the recipient list.

7.2.8 AddCCRecipient(string address, string display, EnumCharEncode displayEncoding)

Add a cc recipient

7.2.9 AddBCCRecipient(string address, string display, EnumCharEncode displayEncoding)

Add a bcc recipient

7.2.10 AddHeaderItem(string itemName, string itemContent)

Add a header item

7.2.11 Send

Send the email to an SMTP server for delivery

7.2.12 SendAsync

Send the email to an SMTP server for delivery. This method does not block the calling thread. When the send mail operation completes the event SendCompleted occurs.

7.2.13 SendAsyncCancel

Cancel an asynchronous operation to send an email message

7.3 Events

7.3.1 event SendCompletedEventHandler SendCompleted

Occurs when an asynchronous email send operation completes

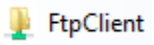
7.3.2 event OperationFailHandler OperationFailed

Occurs when an error occurs calling Send or SendAsync

7.3.3 event EventHandler Disposed

Occurs when this object is disposed

8 FTP Client



FTP client component can be used to perform FTP operations such as uploading/downloading files, deleting files from FTP servers, etc.

8.1 Properties

8.1.1 string ErrorMessage

Gets the error message for the last failed operation.

8.1.2 string OperationResponse

Gets response from the FTP server.

8.1.3 FtpFileInfo[] FileList

Gets an array containing the file information returned by calling GetContents.

8.1.4 string[] FileNames

Gets an array containing the file names returned by calling GetContents.

8.1.5 string[] FolderNames

Gets an array containing the folder names returned by calling GetContents.

8.1.6 string FtpServer

Gets and sets FTP Server name or IP address.

8.1.7 string Usename

Gets and sets the user name used for connecting to the FTP server.

8.1.8 string Password

Gets and sets the password used for connecting to the FTP server.

8.1.9 string CurrentServerFolder

Gets and sets the folder under the root of the FTP server to be used for uploading and downloading operations.

8.1.10 bool UseBinary

Gets or sets a value that specifies the data type for file transfers.

8.1.11 bool EnableSsl

Gets or sets a value that specifies whether an SSL connection should be used.

8.1.12 AuthenticationLevel AuthenticationLevel

Gets or sets a value indicating the level of authentication and impersonation used for this request.

8.1.13 bool KeepAlive

Gets or sets a value that specifies whether the control connection to the FTP server is closed after the request completes.

8.1.14 RequestCacheLevel CachePolicy

Gets or sets the cache policy for this request.

8.1.15 string ConnectionGroupName

Gets or sets the name of the connection group that contains the service point used to send the current request.

8.1.16 TokenImpersonationLevel ImpersonationLevel

Gets or sets the impersonation level for the current request.

8.1.17 string ProxyName

Gets or sets the name for the proxy used to communicate with the FTP server.

8.1.18 int ProxyPort

Gets or sets the port for the proxy used to communicate with the FTP server.

8.1.19 int ReadWriteTimeout

Gets or sets a time-out when reading from or writing to a stream.

8.1.20 int Timeout

Gets or sets the number of milliseconds to wait for a request.

8.1.21 bool UsePassive

Gets or sets the behaviour of a client application's data transfer process.

8.2 Methods

8.2.1 bool UploadFile(string sourceFilePath, string targetFileName)

Upload a local file to the FTP server. sourceFilePath is a full path to a local file to be uploaded. targetFileName can be empty. If targetFileName is empty then the file name of the sourceFilePath is used. If targetFileName contains '/' then it is considered a full path on the FTP server else the value of property CurrentServerFolder is used as the folder on the server.

8.2.2 bool UploadFiles(string sourceFolder, string filePattern, SearchOption searchOption, string targetFolder)

Upload all files matching filePattern in the sourceFolder to targetFolder on the FTP server. If filePattern is empty then all files in the folder are uploaded. If targetFolder is empty then CurrentServerFolder is used.

8.2.3 bool DeleteFile(string filename)

Delete a file from the FTP server. If filename contains '/' then it is considered a full path else it is considered a file under the CurrentServerFolder.

8.2.4 bool Download(string localFolder, string filename)

Download a file specified by filename from the FTP server and save it to localFolder. If filename contains '/' then it is considered a full path else it is considered a file under the CurrentServerFolder.

8.2.5 bool CreateFtpFolder(string folderName)

Create a folder on the FTP server. If folderName contains '/' then it is considered a full path else it is considered a folder under the CurrentServerFolder.

8.2.6 bool DeleteFtpFolder(string folderName)

Delete a folder from the FTP server. If folderName contains '/' then it is considered a full path else it is considered a folder under the CurrentServerFolder.

8.2.7 bool Rename(string filename, string targetName)

Rename a file on the FTP server. If filename contains '/' then it is considered a full path else it is considered a file under the CurrentServerFolder.

8.2.8 bool GetContents(string folderName)

Get file names (FileNames) and folder names (FolderNames) under folder specified by folderName from the FTP server. Detailed file information including size and datetime is represented by FileList. If folderName contains '/' then it is considered a full path else it is considered a folder under the CurrentServerFolder.

8.3 Events

8.3.1 event FtpOperationEvent OperationFailed

Occurs when an operation fails. The ErrorMessage property contains the error message for the failure.

8.3.2 event FtpOperationEvent FileDownloading

Occurs before a file downloading starts.

8.3.3 event FtpOperationEvent FileDownloaded

Occurs after a file is downloaded from an FTP server.

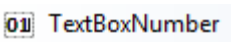
8.3.4 event FtpOperationEvent FileUploading

Occurs before a file uploading starts.

8.3.5 event FtpOperationEvent FileUploaded

Occurs after a file is uploaded to an FTP server.

9 TextBoxNumber



TextBox for showing/entering numbers. It is derived from the TextBox class defined in Microsoft .Net Framework. For reference of TextBox, see http://msdn.microsoft.com/en-us/library/system.windows.forms.textbox(VS.71).aspx

9.1 Properties

9.1.1 bool DisableValidation

Gets or sets a value indicating whether the data validation is disabled. When the data validation is disabled this component acts as a normal TextBox.

9.1.2 string DisplayFormat

Gets or sets a value indicating the display format. The value can be C, c, D, d, E, e, F, f, G, g, N, n, P, p, R, r, X, x, or other format string, for example: quanty:#.00. See http://msdn.microsoft.com/en-us/library/kfsatb94.aspx and other Microsoft .Net references for details. If the value is not empty then this text box is read-only.

9.1.3 EnumCulture ValidateCulture

Gets or sets a value indicating the culture used for data validation.

9.1.4 EnumNumber TargetType

Gets or sets a value indicating the data type allowed for the Text.

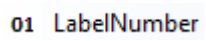
9.1.5 object NumericValue

Gets the numeric value represented by the Text property.

9.1.6 CultureInfo ValidateCultureInfo

Gets a CultureInfo object corresponding to the ValidateCulture value.

10 LabelNumber



Label for showing numbers. This class is derived from the Label class in Microsoft .Net Framework. For reference of label, see http://msdn.microsoft.com/en-us/library/system.windows.forms.label.aspx

10.1 Properties

10.1.1 bool DisableValidation

Gets or sets a value indicating whether the data validation is disabled. When the data validation is disabled this component acts as a normal Label.

10.1.2 string DisplayFormat

Gets or sets a value indicating the display format. The value can be C, c, D, d, E, e, F, f, G, g, N, n, P, p, R, r, X, x, or other format string, for example: quanty:#.00. See http://msdn.microsoft.com/en-us/library/kfsatb94.aspx and other Microsoft .Net references for details. If the value is not empty then this text box is read-only.

10.1.3 EnumCulture ValidateCulture

Gets or sets a value indicating the culture used for data validation.

10.1.4 EnumNumber TargetType

Gets or sets a value indicating the data type allowed for the Text.

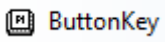
10.1.5 object NumericValue

Gets the numeric value represented by the Text property.

10.1.6 CultureInfo ValidateCultureInfo

Gets a CultureInfo object corresponding to the ValidateCulture value.

11 ButtonKey



Button as a key for an onscreen keyboard. When it is clicked/touched the contents of KeysToSend property are sent to the window having input focus. It is derived from the Button class in Microsoft .Net Framework. For reference of the Button class, see http://msdn.microsoft.com/en-us/library/system.windows.forms.button.aspx

11.1 Properties

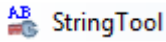
11.1.1 string KeyMaps

Gets and sets a string representing a list of keys to watch for. If the previous key sent is among the key list, a coressponding keys are sent to replace the previous key. This function is enabled only when SendKeys property is not empty.

11.1.2 string KeysToSend

Gets and sets a string indicating the string of keystrokes to be sent to the active application when this button is clicked by the user. To specify characters that aren't displayed when you press a key, such as ENTER or TAB, and keys that represent actions rather than characters, or specify key combinations, see the codes and descriptions in the Limnor Studio tutorial, and the sample applications.

12 StringTool



This component provides functionality for handling strings .

12.1 Properties

12.1.1 string String

Gets or sets a string for processing/handling.

12.1.2 string MD5Hash

Gets an MD5 hash for String property.

12.1.3 EnumCommonCulture FormatCulture

Gets or sets the culture to be used to get FormattedString.

12.1.4 string FormattedString

Gets a value formatted by using String as the formatting text, using FormatCulture as the formatting culture and substituting {field name} in the String with the corresponding property values. The value is the same as the String property if the String does not have fields.

Example:

Suppose String property is

Date:{date} Dear {name}, How are you?

Execute two actions:

SetFieldValue("date", 2010-01-02) and SetFieldValue("name", "John")

After the executions of above actions, this property becomes

Date:2010-01-02 Dear John, How are you?

12.1.5 string[] Fields

Gets a string array. It lists all the fields contained in String property.

Example:

Suppose String property is

Date:{date} Dear {name}, How are you?

Then this property is a string array with 2 items. The first item is "date" and the second item is "name".

12.1.6 string DirectoryName

Gets a value as a directory name using FormattedString as a file full path.

12.1.7 string FileName

Gets a value as a file name using FormattedString as a file full path.

12.1.8 FileNameWithoutExtension

Gets a value as a file name without extension using FormattedString as a file full path.

12.1.9 string FullPath

Gets a value as an absolute path using FormattedString as a path string.

12.2 Methods

12.2.1 string FormatStringWithValues(params object[] values)

Return a string value formatted by using String as the formatting text, using FormatCulture as the formatting culture, and substituting {field name} in the String with the corresponding parameter values.

Example:

Suppose String property is

Date:{date} Dear {name}, How are you?

Calling this method with parameter 2010-01-02 and John gives following result:

Date:2010-01-02 Dear John, How are you?

12.2.2 SetFieldValue(string fieldName, object value)

Set the value for the field named by fieldname parameter.

12.2.3 string Encrypt(string input, string key)

Encrypt the input string with the specified key.

12.2.4 static string Decrypt(string input, string key)

Decrypt the input string with the specified key.

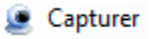
12.2.5 static string GetMD5Hash(string input)

Calculate MD5 hash for the input string. For saving a password to a database input can be a password and the return value of this method can be saved to the database.

12.2.6 static bool VerifyMd5Hash(string input, string hash)

Verify that the string input has the same MD5-Hash given by the parameter hash. This method can be used to verify log in where input can be a password given by the user and hash can be the password-hash saved in a database.

13 Capturer



This component can be used to capture video and audio from sources such as web cameras, TV, etc.

13.1 Properties

13.1.1 bool ShowMessagesOnError

Gets and sets a value indicating whether error messages are displayed when an error occurs. Error messages may also be displayed by executing a DisplayErrors action.

13.1.2 string VideoDeviceName

Gets and sets video device name.

13.1.3 string AudioDeviceName

Gets and sets audio device name.

13.1.4 string VideoCompressor

Gets and sets video compressor name.

13.1.5 string AudioCompressor

Gets and sets audio compressor name.

13.1.6 string VideoSourceName

Gets and sets video source name.

13.1.7 string AudioSourceName

Gets and sets audio source name.

13.1.8 int FrameRate

Gets and sets the frame rate for capturing video.

13.1.9 Size FrameSize

Gets and sets the frame size for capturing video.

13.1.10 EnumAudioChannel AudioChannel

Gets and sets the number of channels in the waveform-audio data.

13.1.11 int AudioSamplingRate

Gets and sets the number of audio samples taken per second.

13.1.12 short AudioSampleSize

Gets and sets the number of bits recorded per sample.

13.1.13 int TvTunerChannel

Gets and sets the TV Tuner channel.

13.1.14 TunerInputType TunerInputType

Gets and sets the tuner frequency (cable or antenna).

13.1.15 string Filename

Gets and sets the name of the file for saving the capturing.

13.1.16 Control PreviewWindow

Gest and sets the window for showing the video preview.

13.1.17 IList<string> VideoDeviceList

Gets a list of available video input devices.

13.1.18 IList<string> AudioDeviceList

Gets a list of available audio input devices.

13.1.19 IList<Filter> VideoCompressorList

Gets available video compressors.

13.1.20 IList<Filter> AudioCompressorList

Gets available audio compressors.

13.1.21 IList<Source> VideoSourceList

Gets a collection of available video sources/physical connectors on the current video device.

13.1.22 IList<Source> AudioSourceList

Gets a Collection of available audio sources/physical connectors on the current audio device.

13.1.23 IList<string> PropertyPages

Gets a list of property pages.

13.1.24 bool Stopped

Gets a Boolean indicating whether the capturing is stopped.

13.1.25 bool Started

Gets a Boolean indicating whether the capturing is started.

13.1.26 bool Cued

Gets a Boolean indicating whether it is cued for starting capturing.

13.2 Methods

13.2.1 Cue

Prepare for capturing. Use this method when capturing must begin as quickly as possible.

13.2.2 StartCapture
Start capturing.

13.2.3 StopCapture
Stop capturing.

13.2.4 DisplayErrors
Show all errors occured since the starting of the application or since the last ClearErrorLog was executed.

13.2.5 ClearErrorLog
Remove all errors logged in memory.

13.2.6 SetInputDevices(string videoDevice, string audioDevice)
Use the video input device and audio input device specified by the parameters.

13.2.7 bool SelectInputDevices
Use a dialogue box to select video and audio input devices.

13.2.8 bool SelectVideoCompression
Use a dialogue box to select video compressor.

13.2.9 bool SelectAudioCompression
Use a dialogue box to select audio compressor.

13.2.10 bool SelectVideoSource
Use a dialogue box to select video source.

13.2.11 bool SelectAudioSource
Use a dialogue box to select audio source.

13.2.12 ShowPropertyPageByIndex(int index)

Show property page identified by index. Some property pages cannot be displayed while previewing and/or capturing. This method will block until the property page is closed by the user. If the input device drivers are 32-bit then your application must be compiled to x86 for this method to work.

13.2.13 ShowPropertyPageByName(string pageName)

Show property page identified by page name. Some property pages cannot be displayed while previewing and/or capturing. This method will block until the property page is closed by the user. If the input device drivers are 32-bit then your application must be compiled to x86 for this method to work.

13.2.14 bool ShowPropertiesDialogue

Display a dialogue box showing all properties.

13.3 Events

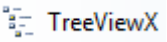
13.3.1 event EventHandler CaptureCompleted

It occurs when the capture is completed.

13.3.2 event EventHandler Error

It occurs when an operation failed.

14 TreeViewX



Displays a hierarchical collection of labeled items to the user that optionally contain an image and data. It can be saved to/load from XML file. It supports creating shortcuts to nodes. This class is derived from the TreeView class in Microsoft .Net Framework. For reference of TreeView, see http://msdn.microsoft.com/en-us/library/system.windows.forms.treeview.aspx

This component uses class TreeNodeX, TreeNodeValue, and TreeNodeShortcut. These classes are documented after this chapter.

14.1 Properties

14.1.1 Boolean IsCategoryNodeSelected

Gets a value indicating whether a category node is selected

14.1.2 Boolean IsShortcutNodeSelected

Gets a value indicating whether a shortcut node is selected

14.1.3 Boolean IsPropertyNodeSelected

Gets a value indicating whether a property node is selected

14.1.4 TreeNodeShortcut SelectedShortcut

Gets the selected shortcut node. Returns null if no shortcut is selected

14.1.5 TreeNodeX SelectedCategryNode

Gets the selected category node. Returns null if no category node is selected

14.1.6 TreeNodeValue SelectedPropertyNode

Gets the selected property node. Returns null if no property node is selected

14.1.7 Boolean ShowPropertyNodes

Gets and sets a value indicating whether the property nodes should be displayed

14.1.8 Guid TreeViewGuid

Gets a Guid identifying this control

14.1.9 EnumTreeViewMenu EnabledMenuItems

Gets and sets a value indicating which menu items should be enabled

14.1.10 Boolean ReadOnly

Gets or sets a Boolean value indicating whether it allows the user to modify the TreeView

14.1.11 TreeNodeCollection Nodes

Gets the root nodes in the TreeView control.

14.1.12 String ErrorMessage

Gets error message from SaveToFile/ReadFromFile actions

14.2 Methods

14.2.1 AddRootNode(String text) TreeNodeX

Add a root node. The new node is selected. Returns the new node.

14.2.2 AddSubNode(String text) TreeNodeX

Add a sub node to the selected node. The new node is selected. Returns the new node. If there is not a selected node then this method does nothing and returns null.

14.2.3 DeleteSelectedCategoryNode(Boolean confirm)

Delete selected node. If there is not a selected node or the selected node is for value then this method does nothing. If parameter 'confirm' is true then a message box appears to ask the user to confirm the deletion.

14.2.4 DeleteSelectedShortcut(Boolean confirm)

Delete selected shortcut. If there is not a selected node or the selected node is not a shortcut then this method does nothing. If parameter 'confirm' is true then a message box appears to ask the user to confirm the deletion.

14.2.5 DeleteSelectedValue(Boolean confirm)

Delete selected value node. If there is not a selected node or the selected node is not for value then this method does nothing. If parameter 'confirm' is true then a message box appears to ask the user to confirm the deletion.

14.2.6 MoveSelectedNodeToRoot

Make selected category node to be a root node

14.2.7 CreateNewProperty() TreeNodeValue

Attach a new property to the selected category node and return the new node representing the new property. A dialogue appears asking for the type of the new data. The default value for the data type is used as the new value. If there is not a category node selected then this method does nothing.

14.2.8 MoveNodeUp(TreeNode node)

Move the tree node up

14.2.9 MoveNodeDown(TreeNode node)

Move the tree node down

14.2.10 MoveSelectedNodeUp

Move selected node up

14.2.11 MoveSelectedNodeDown

Move selected node down

14.2.12 EditNodes() Boolean

Launch Tree Nodes Editor to edit the nodes and values. It returns false if the editing is canceled.

14.2.13 RemoveAllNodes

Remove all nodes

14.2.14 RemoveAllShortcuts(Guid id)

Remove all shortcuts specified by id

14.2.15 ShowSelectedObjectProperties

Show properties of the selected node in a dialogue box

14.2.16 SaveToXmlDocument() XmlDocument

Save the contents of this control to an XmlDocument

14.2.17 SaveToFile(String filename) Boolean

Save the contents of this control to an XML file

14.2.18 LoadFromFile(String filename) Boolean

load contents from an XML file into this control

14.2.19 LoadFromXmlDocument(XmlDocument doc) Boolean

Load contents from an XmlDocument into this control

14.2.20 SyncShortcuts(TreeNodeX node)

Reload all the shortcuts to the specified node. When the node is changed, call this method to synchronize all its shortcuts.

14.2.21 GetCategoryNodeById(Guid id) TreeNodeX

Get a category node identified by guid

14.3 Events

14.3.1 CategoryNodeSelected - EventHandlerTreeNodeX

It occurs when a category node is selected

14.3.2 ShortcutNodeSelected - EventHandlerTreeNodeShortcut

It occurs when a shortcut node is selected

14.3.3 PropertyNodeSelected - EventHandlerTreeNodeValue

It occurs when a value node is selected

14.3.4 ShortcutMouseClick - MouseEventHandlerTreeNodeShortcut

Occurs when the user clicks a shortcut TreeNode with the mouse.

14.3.5 ShortcutMouseDoubleClick - MouseEventHandlerTreeNodeShortcut

Occurs when the user double-clicks a shortcut TreeNode with the mouse.

14.3.6 CategoryNodeMouseClick - MouseEventHandlerTreeNodeX

Occurs when the user clicks a category TreeNode with the mouse.

14.3.7 CategoryNodeMouseDoubleClick - MouseEventHandlerTreeNodeX

Occurs when the user double-clicks a category TreeNode with the mouse.

14.3.8 PropertyNodeMouseClick - MouseEventHandlerTreeNodeValue

Occurs when the user clicks a property TreeNode with the mouse.

14.3.9 PropertyNodeMouseDoubleClick - MouseEventHandlerTreeNodeValue

Occurs when the user double-clicks a property TreeNode with the mouse.

14.3.10 ShortcutMouseHover - EventHandlerTreeNodeShortcut

Occurs when the mouse hovers over a shortcut TreeNode.

14.3.11 CategoryNodeMouseHover - EventHandlerTreeNodeX

Occurs when the mouse hovers over a category TreeNode.

14.3.12 PropertyNodeMouseHover - EventHandlerTreeNodeValue

Occurs when the mouse hovers over a property TreeNode.

15 TreeNodeX

This class is derived from class TreeNode in Microsoft .Net Framework. See
http://msdn.microsoft.com/en-us/library/system.windows.forms.treenode.aspx

TreeNodeShortcut is derived from this class.

15.1 Properties

15.1.1 Guid TreeNodeGuid

Gets a Guid identifying this node.

15.2 List<TypedNamedValue> ValueList

Gets a list of values associated with this node

15.3 Boolean ShowPropertyNodes

Gets and sets a value indicating whether the property nodes should be displayed

15.4 Boolean NextLevelLoaded

Gets a Boolean value indicates whether its child nodes have been loaded

15.5 TreeNodeCollection Nodes

Gets the child nodes in this node.

15.5.1 Boolean IsByShortcut

Gets a Boolean value indicating whether this node is generated from a shortcut.

15.6 Methods

15.6.1 RemoveAllShortcuts(Guid id)

Remove all shortcuts specified by id

15.6.2 GetCategoryNodeById(Guid id) TreeNodeX

Gets category node by guid

15.6.3 GetValue(String name) Object
Gets associated value by value name

15.6.4 SetValue(String name,Object value)
Associate a value to this node. Use a name to identify the value.

15.6.5 ClearValues
Remove all associated values

15.6.6 CreateValue(Type t) TreeNodeValue
Create a Value-Node by specifying the type for the value

15.6.7 AddValue(String name,Type type,Object value)
Associate a value to this node. The value is identified by name. Value type is specified by type.

16 TypedValue
This class is for defining a data with its type and value.

16.1 Properties

16.1.1 Type ValueType
Gets and sets the type of the data.

16.1.2 object Value
Gets and sets the value of the data.

16.2 Methods

16.2.1 ResetValue
Sets the value of the data to the default value according to the type of the data.

17 TypedNamedValue
This class defines a data with name, type and value.

17.1 Properties

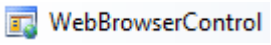
17.1.1 string Name

Gets and sets the name of the data.

17.1.2 TypedValue Value

Gets and sets the type and value of the data.

18 WebBrowserControl



This control derives from .Net WebBrowser and adds new features such as allowing blocking of pop-up windows.

For reference of WebBrowser, see http://msdn.microsoft.com/en-us/library/system.windows.forms.webbrowser.aspx

18.1 Properties

18.1.1 PopupLevel - EnumPopupLevel

Block of pop-up windows. AllowAllPopups: do not block pop-ups; AllowSecureSites: do not block pop-ups from secure web sites; BlockMost: block most pop-ups; BlockAll: block all pop-ups

18.2 Methods

18.3 Events

18.3.1 Downloading - EventHandler

Occurs when downloading of a document starts

18.3.2 DownloadComplete - EventHandler

Occurs when downloading of a document is completed

18.3.3 StartNavigate - EventHandler<BrowserNavigationEventArgs>

Occurs before navigation occurs

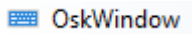
18.3.4 StartNewWindow – EventHandler<BrowserNavigationEventArgs>

Occurs when a new window is to be created

18.3.5 ApplicationQuit - EventHandler

Occurs when the browser application quits

19 On-Screen-Keyboard



This object manages on screen keyboard.

19.1 Properties

19.1.1 OskLoaded - Boolean

Gets a Boolean value indicating whether the on screen keyboard is loaded successfully

19.1.2 ErrorMessage - String

Gets a string value representing the error message for the last action

19.1.3 Location - Point

The coordinates of the upper-left corner of the on-screen-keyboard

19.1.4 DisableCloseButton - Boolean

Gets a value indicating whether the close button should be disabled.

19.1.5 Visible - Boolean

Gets a value indicating whether the on-screen-keyboard is visible.

19.1.6 Movable - Boolean

Gets a value indicating whether the on-screen-keyboard has a title bar so that the user may move it

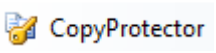
19.2 Methods

19.2.1 StartOsk
Start on-screen-keyboard

19.2.2 CloseOsk
Close on-screen-keyboard

19.3 Events

20 Copy Protector



This component can be used to do copy-protection for your application. Your user creates a register file and sends it to you. Limnor Studio processes the register file and creates a license file. You send the license file to your user. Your user may start using your application. One license file allows one copy of your application to run.

See Users' Guide (http://www.limnor.com/index.html?Doc=studio_userGuide.html) for how to use this component.

20.1 Properties

20.1.1 ApplicationId - Guid
Gets a Guid identifying the application to be protected

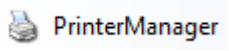
20.1.2 SupportMessage - String
Gets or sets a string describing the ways to send the register files, for example, email address or FTP server name.

20.2 Methods

20.2.1 VerifyLicense
Verify software license for the application protected by this component. It asks for a license file if a license is not installed. If the user does not have a license file then a register file can be created. The

user of this application should send the register file to you. You should use Limnor Studio to issue a license based on the register file and send the license file to the user.

21 Printer Manager



This object provides printer management.

21.1 Properties

21.1.1 Printers - StringCollection

Gets a StringCollection containing all printer names

21.2 Methods

21.2.1 SetDefaultPrinter(String printerName)

Set the default printer to the one specified by printerName. To get all printer names, use property Printers

21.2.2 SetDefaultPrinterByUI(Form caller)

Display a dialogue box showing all printers. Select a printer from the list as the default printer

21.3 Events

21.3.1 Disposed - EventHandler

Occurs when this object is disposed

