

Drawing Board ActiveX Control v2.0

Drawing Board ActiveX Control acts as a drawing board for creating online drawing programs rapidly. The OCX is lightweight and flexible, and gives developers new possibilities for drawing flowcharts, organizational charts, workflow, network diagrams and other business charts in a form or web page. You can custom the toolbars and template library according your requirement.

Home: <http://www.anydraw.com>

Online Demo: http://www.anydraw.com/Drawing_Board_Online_Demo.htm

Support Mail: support@anydraw.com

Sales online: <https://www.regnow.com/softsell/nph-softsell.cgi?item=13274-3> Sales only \$299.95

Installation

You can find the installation package at our web site or on some other web sites from authorized Any Draw distributors. You have just to launch the setup program and follow the instructions given in the installation program. The list of installed components is the following: The setup program will register the ActiveX file. Anyway you can do it yourself with the regsvr32 tool program. For instance, to register EDImage.ocx, the command is the following: REGSVR32 FilePath\EDImage.ocx. The Drawing Board ActiveX Control has a library file catalog. All the files in the catalog must be copy into the windows system catalog. The installation program may install the Microsoft shared DLLs used by ActiveX controls, if not already installed on your machine. Those DLLs are: mfc42.dll, msvcr7.dll and gdiplus.dll.

How to add Drawing Board ActiveX Control to your Visual Basic 6.0 project

1. From the Project Menu select **Components...**
2. Select control **"EDImage Control"**
3. Click the OK Button
4. The control will now appear in your toolbox
5. Drag and drop the control on your form
6. You can set all properties such as show/hide toolbar, show/hide template, show/hide grid, template background, view background, etc.
7. There are some methods to operator the ActiveX control. You can custom your own template shapes and toolbars.

How to Custom the shape templates

1. The shape templates were saved in the "EDLibrary" catalog. The template file format is "*.fcl" which is a binary serialize file. FCL file can be created with Flow Diagrams Software.
2. In the "EDLibrary" catalog there is a "Config.xml" file which is the configuration files of shape templates.

In the "EDLibrary" catalog there is a "Config.xml" file which is the configuration files of shape templates. `<?xml version="1.0"?>`

```

<EDConfig>
<EDLibrary>
  <Template id="1001" name="Base Shapes" path="BaseShape.fcl" bShow="true"
bFocus="false"></Template>
  <Template id="1002" name="Line and Arrow" path="BaseLine.fcl" bShow="true"
bFocus="false"></Template>
  <Template id="1003" name="Base Connector" path="BaseJoint.fcl" bShow="true"
bFocus="true"></Template>
  <Template id="1004" name="Work Flow" path="FlowWork.fcl" bShow="false"
bFocus="false"></Template>
</EDLibrary>
</EDConfig>

```

- You can define a template with the <Template></Template>. The "id" is an exclusive tag. The "name" is the title shown in the template top. The "path" is the template file which can be created by Flow Diagrams Software. Only when the bShow="true", the template will display in the template library pane. Only when the bFocus="true", the template will act as current active one. You should also notice there only is one bFocus="true" in the <EDLibrary></EDLibrary>. The above codes show three shape templates in the left template pane.
- You can set the relevancy between control and library. If you haven't the source code, you should copy all files in the EDLibrary catalog into windows system32 catalog. The installation program has done this thing.
- How to create the FCL file, please visit <http://www.anydraw.com/CreateTemplate.htm>.

Code a solution using the control

The control is very customizable. You can change the color scheme of any of the control elements, as well as determine the toolbars and template shapes. These can be set at run time or design time as needed.

1. Create new documents

```

HRESULT NewDocument ();
Description: Creates a new document.

```

The method allows you to create a new document.

2. Open documents

```

Function: boolean OpenDocument(BSTR filename);
Description: Opens a document from a file in local drive.

Function: boolean OpenWebFile(BSTR strPath);
Description: Opens a document from an URL.

```

You can also open and edit edraw documents (*.edd or *.xml file format) that exist on a local drive, Universal Naming Convention (UNC) share, or Web folder. You can call the method directly and give the control a specific file to open.

Open takes either a qualified file path or a URL to a file on a remote Web server. For example, the

following code opens a local file and keeps a lock on it for editing:

```
EDImage.OpenDocument "C:\Draw.edd"
Or EDImage.OpenDocument "C:\Draw.xml"
```

If you combine this ability with a URL, you can use code that resembles the following to open the resulting.

```
EDImage.OpenWebFile "https://www.anydraw.com/demo/test.xml"
```

The user can then edit the results and save the file as a local file on disk, or save the file to the server as a new file in a Web folder.

3. Save documents

```
boolean SaveDocument(BSTR filename);
Description: Saves the document to specified location in local drive. Support .edd, xml, bmp,
jpg, gif, tif and png file format.

boolean SaveWebFile(BSTR strPath, FileType nType);
Description: Saves the document to specified web folder.
typedef enum FileType
{
    edd = 0,
    xml = 1,
    bmp = 2,
    jpg = 3,
    gif = 4,
    tif = 5,
    png = 6
} EDrawFileType;
```

To save a document, you can use the menu or call the **Save** method. The **Save** method acts both as a simple **Save** command and as a **SaveAs** command, depending on whether you pass a file location for the first parameter.

You can also save to a Web folder on a remote server if that server supports either Microsoft FrontPage Server Extensions (FPSE) or the Web Distributing Authoring and Versioning (WebDAV) protocol extension for HTTP. The following code shows a new xml file that is saved to a remote file server:

```
EDImage.SaveWebFile "http://www.anydraw.com/demo/SaveFile.php", 1
```

The following code shows a new jpg file that is saved to a remote file server:

```
EDImage.SaveWebFile "http://www.anydraw.com/demo/SaveFile.php", 3
```

The ActiveX Control posts a stream to the web page. Then the web page gets the stream. If you write the receive code with php, you can see the follow sample.

```
<?php
/* PUT data comes in on the stdin stream */
$putdata = fopen("php://stdin", "r");
/* Open a file for writing */
$fwp = fopen("test.edd", "w");
/* Read the data 1 KB at a time and write to the file */
```

```

while ($data = fread($putdata, 1024))
    fwrite($fp, $data);
/* Close the streams */
fclose($fp);
fclose($putdata);
?>

```

4. Close Document

You can close the currently open document by call the "NewDocument" again.

5. Upload/Download File

```

FTPUploadFile(LPCTSTR strURL, LPCTSTR strFilePath, LPCTSTR UserName, LPCTSTR
Password);
Description: Upload file to specified web folder with FTP method.
FTPDownloadFile(LPCTSTR strURL, LPCTSTR strFilePath, LPCTSTR UserName, LPCTSTR
Password);
Description: Download file from specified web folder with FTP method.

HttpUploadFile(LPCTSTR strURL, LPCTSTR strFilePath);
Description: Upload file to specified URL with HTTP method.
HttpDownloadFile(LPCTSTR strURL, LPCTSTR strFilePath);
Description: Download file from specified URL with HTTP method.

```

You can upload or download any file with these methods. The control provides FTP and HTTP method. The following code shows how to upload file to a remote file server or download file from remote file server.

```

EDImage.FTPUploadFile "ftp://www.anydraw.com:21/draw.edd", "C:\EDImage\draw.edd",
"superadmin", "key"
EDImage.FTPDownloadFile "ftp://www.anydraw.com:21/draw.edd", "C:\EDImage\draw.edd",
"superadmin", "key"
EDImage.HttpUploadFile "http://www.anydraw.com/SaveFile.php", "C:\EDImage\draw1.xml"
EDImage.HttpUploadFile "http://www.andraw.com/draw1.xml", "C:\draw1.xml"

```

6. Show Dialog

```

void ShowDialog(ShowDialogType DlgType);
Description: Displays a modal dialog of the given type for user action.
typedef enum ShowDialogType
{
    DialogOpen = 0,
    DialogSaveAs = 1,
    DialogFilePrint= 2,

```

```
DialogPrintSetup = 3,  
DialogPageSetup = 4,  
DialogFormatLine = 5,  
DialogFormatShape = 6,  
DialogFormatText = 7,  
DialogFormatProtect = 8  
} ShowDialogType;
```

You can call the common dialogs such as Open File dialog, Save As Dialog, Print Setup Dialog, Page Setup Dialog, Format Line Dialog, Format Shape Dialog, Format Text Dialog and Format Protect Dialog. The following code shows some sample examples.

```
EDImage.ShowDialog 4
```

7. Show or hide the template pane

```
void ShowTemplates(VARIANT_BOOL bShow);  
Description: Show or hide the template pane.
```

8. Show or hide the toolbars

```
void ShowToolBar(VARIANT_BOOL bShow);  
Description: Show or hide the toolbars.
```

9. Show or hide the rulers

```
void ShowRuler(VARIANT_BOOL bShow);  
Description: Show or hide the rulers.
```

10. Show or hide the grid

```
void ShowGrid(VARIANT_BOOL bShow);  
Description: Show or hide the grid.
```

11. Show or hide the Connect Points

```
void ShowConnectDot(BOOL bShow);  
Description: Show or hide the connect Points.
```

You can programmatically show or hide the template pane, toolbars, or ruler by setting these parameters to True or False. This may be useful when you try to restrict user actions or control the appearance of the board while it is embedded.

12. Sets the Basic properties

boolean bShowToolBar;

boolean bShowRuler;

boolean bShowTemplate;

boolean bShowGrid;

Description: Show/Hide toolbars, ruler, template pane and grid.

OLE_COLOR clrPaper;

OLE_COLOR clrMargin;

OLE_COLOR clrGrid;

OLE_COLOR clrTemplateBk;

Description: Sets the color of paper, paper margin, grid and template background.

short nLeftMargin;

short nRightMargin;

short nTopMargin;

short nBottomMargin;

Description: Sets the page margin. The unit is millimeter. The default is 50 mm.

13. Add Basic Shapes

void SetStateNone(); Description: Restore the none state.

void SetAddLineState(); Description: Draw line.

void SetAddRectangleState(); Description: Draw rectangle.

void SetAddEllipseState(); Description: Draw ellipse.

void SetAddArcState(); Description: Draw arc.

void SetAddBezierState(); Description: Add bezier line.

void SetAddTextState(); Description: Draw text.

void SetAddImageState(); Description: Insert picture from a graphic file.

void SetAddConnectLineState(); Description: Draw smart connector.

void SetAddConnectDotState(); Description: Select the Shape and adjust the position of connection point.

void SetAddOleObjectState(); Description: Insert OLE.

14. Edit Functions – Undo/Redo, Cut/Copy/Paste, Select All, Delete

void Undo(short nTimes); Description: Undo the last action. You can undo multiply steps.

void Redo(short nTimes); Description: Redo the previously undone action. You can undo multiply steps.

void Cut(); Description: Cut the selection and put it on the Clipboard.

void Copy(); Description: Copy the selection and put it on the Clipboard.

void Paste(); Description: Insert clipboard contents.

void SelectAll(); Description: Select all the shape.

void Delete(); Description: Delete selected object(s).

15. Zoom Function

void ZoomView(float percent); Description: Zoom the view into adopt scale. The percent is from 0 to 1.

void ZoomViewFitWidth(); Description: Zoom to adopt Width.

void ZoomViewFitWholePage(); Description: Zoom to adopt size.

void ZoomViewUserDef(); Description: Zoom to the user defined scale.

16. Nudge Selected Shapes

void NudgeUp(); Description: Nudge up the selected objects.

void NudgeDown(); Description: Nudge down the selected objects.

void NudgeLeft(); Description: Nudge left the selected objects.

void NudgeRight(); Description: Nudge right the selected objects.

17. Adjust the Order of Shapes

void SendToTopLayer(); Description: Bring shape to front.

void SendToBottomLayer(); Description: Send shape backward.

void SendToHigherLayer(); Description: Bring shape forward.

void SendToLowerLayer(); Description: Send shape to backward.

18. Align and Attribute

void AlignLeft(); Description: Align shapes by their left edges.

void AlignHCenter(); Description: Align shapes by their horizontal centers.

void AlignRight(); Description: Align shapes by their right edges.

void AlignTop(); Description: Align shapes by their top edges.

void AlignVCenter(); Description: Center the shapes horizontally within the page.

void AlignBottom(); Description: Align shapes by their bottom edges.

void AdjustPosVCenter(); Description: Center the shapes horizontally within the page.

void AdjustPosHCenter(); Description: Center the shapes vertically within the page.

void EvenlyHSpace(); Description: Create equal horizontal spacing between shapes.

void EvenlyVSpace(); Description: Create equal vertical spacing between shapes.

void AdjustSameWidth(); Description: Make selected shapes same width.

void AdjustSameHeight(); Description: Make selected shapes same height.

19. Group and UnGroup

Void Group(); Description: Group selected shapes.

void Ungroup(); Description: Ungroup selected shapes.

20. Action

void AddAction(LPCTSTR name); Description: Add a undo action.

void ResetAction(); Description: Clear all action. Include undo and redo.

21. Invalidate Client Region

void InvalidateClient(); Description: Invalidate the client region.

Please visit http://www.anydraw.com/Drawing_Board_ActiveX.htm to get more information. If you have any question, please feel free to contact us via support@anydraw.com.

We try to process every support request we receive as fast as possible.