Add JavaScript and VBScript to your application

1 Introduction

E-XD++ Visualization Kit includes an MFC-compatible ActiveScript engine that gives MFC applications complete access to Microsoft's ActiveScript technology so they can host scripts. With Microsoft's ActiveX scripting technology (Internet Client SDK), you can host VBScript or any other compliant scripting language in your own application.

Internet Explorer version 6.0 includes DLLs that implement a VBScript and JavaScript scripting engine. By implementing several required COM interfaces, your C++ code can expose automation objects that you can control with a VBScript script.

E-XD++ Visualization Kit includes MFC extension classes that provide a default implementation of these COM interfaces and wrap these interfaces as an MFC extension. This makes it easier to incorporate scripting into your own MFC-based applications.

E-XD++ Visualization Kit's ActiveScript framework supports two scripting languages- JavaScript and VBScript.

2 Overview of JavaScript

JavaScript is a compact, cross-platform, object-oriented scripting language developed by Netscape that enables the creation of interactive web pages. A JavaScript enabled browser, such as Internet Explorer or Netscape, interprets JavaScript statements embedded in an HTML page. This enables you to create server-based applications similar to Common Gateway Interface (CGI) programs.

The JavaScript language resembles Java in that it supports most of Java's expression and basic control-flow constructs. However, there are some fundamental differences between Java and JavaScript. JavaScript lacks Java's static typing and strong type checking and supports a smaller number of data types. For more information about JavaScript, we recommend the following Web page:

http://www.javascript.com/

3 VBScript

Visual Basic Script is a scripting language that is upwardly compatible with Visual Basic for Applications (VBA). VBA currently ships with Microsoft Office and Visual Basic. Microsoft Internet Explorer 5.0 includes a COM server that implements a VBScript engine. Unlike VBA, you can add VBScript to your existing applications without having to pay licensing or royalty charges.

Aside from language syntax and semantics, a key difference between VBScript and JScript is that VBScript has a complete server-side role and JScript does not. For example, VBScript can be integrated with the ISAPI component of Microsoft's Internet Information Server (IIS) to run a variety of back-end applications.

For a complete description of VBScript, we suggest the article, *Visual Basic Script*, by Keith Pleas in the Spring 1996 issue of *Microsoft Interactive Developer* magazine (MIND). In addition, Microsoft has a wealth of information about VBScript at the following URL:

http://msdn.microsoft.com/en-us/library/t0aew7h6(VS.85).aspx

4 Hosting an Active Script

Before you can make your application scriptable, you need to determine the application's ***object model***. The object model is a specification of the objects that are available to the scripting engine through OLE automation. Internet Explorer, for example, exposes objects that are appropriate to HTML authoring, such as the document, form, window, and browser. In your application, expose objects that are specific to your problem domain.

After you determine the object model for your application, you can begin to implement the application. To host a script, you need to implement the necessary ActiveX scripting COM interfaces. By implementing these interfaces, you expose the properties and methods for each object in your object model and enable manipulation of these objects from VBScript or JavaScript.

If you need to make your application supports Java Script or VB Script, please take the following steps:

1). Create your application with E-XD++ Diagrammer Enterprise Edition AppWizard. For example, your project name is ScriptDemo.

2). Open ExtDataModel.h and ExtDataModel.cpp, change it's base class from CFODataModel to CFOScriptDataModel.

3). Open ScriptDemoView.h and ScriptDemoView.cpp, change it's base class from CFODrawView to CFOScriptView.

4). To use the script of default, please add a memeber to ScriptDemoView.h, CScriptObject *m_ScriptObj; CFOMethodHost m_host;

5). To use this script object, please add following codes within OnInitialUpdate method:

```
OleInitialize(NULL);

CoInitialize(NULL);

m_ScriptObj = new CScriptObject;

m_host.pModel = GetCurrentModel()->m_pDataModel;

m_host.pView = this;

BOOL bOK = m_ScriptObj->AddObject("app",
m_host.GetIDispatch(FALSE), TRUE);

m_ScriptObj->SetTimeout(-1);
```

6). To execute code at any time, please call: RunCode method of CFOScriptView.

7). Please remember to call the following codes within ~~CScriptDemoView:

```
delete m_ScriptObj;

CoUninitialize();
```

5 ActiveScript Classes

5.1 CFOScriptDataModel

The ***CFOScriptDataModel*** class defines a data model object that can be used for Script Support application, within this class, it defined a few script codes saving text, these codes will help you builing your script application, once your application is generated with E-XD++ Application Wizard, you must change it's base from from CFODataModel to CFOScriptDataModel.

5.2 CFOMethodHost

The script host implemented by the ***CFOMethodHost*** class is a COM object that implements the JavaScript and VBScript interfaces. Microsoft defines the JavaScript and VBScript interfaces as part of their ActiveScript technology. A script host can create and destroy a script engine (either VBScript or JScript), create a new top level OLE automation object as part of the script, set a reference to the window hosting the script, and parse and execute a script. An application that requires scripting must instantiate a script host and use its pointer to create the scripting engine and execute a script.

5.3 CFOScriptView

The CFOScriptView class is the main view interface for script engine, you must make sure your CFODrawView class is replaced by this view class.

6 ActiveScript Sample

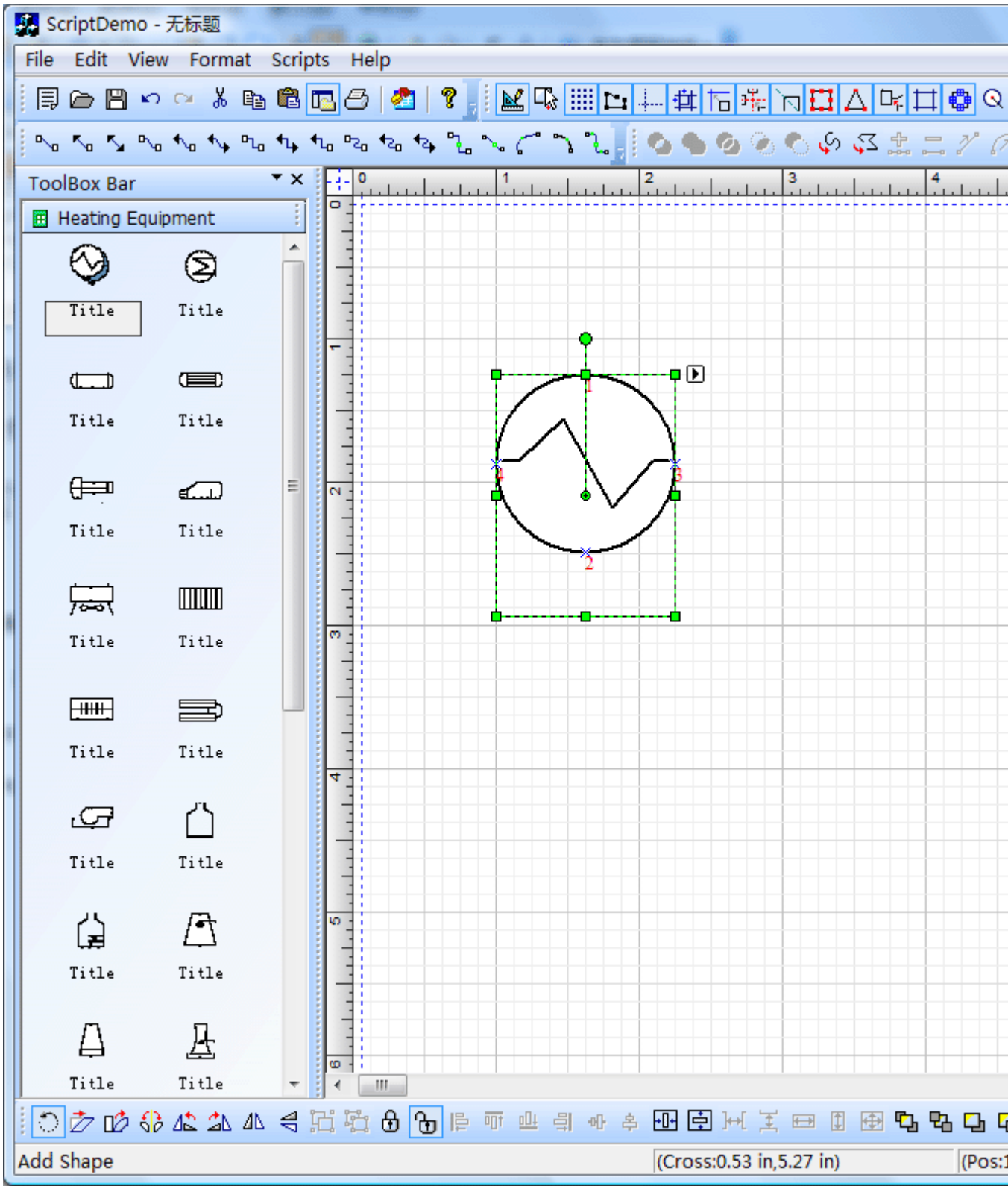
The ScriptDemo sample includes a dialog that demonstrates the canonical form of an ActiveScript host. Refer to the ***CScriptDemoDlg*** class in the sample for an example of a minimal ActiveScript host.

7 How to use the JavaScript?

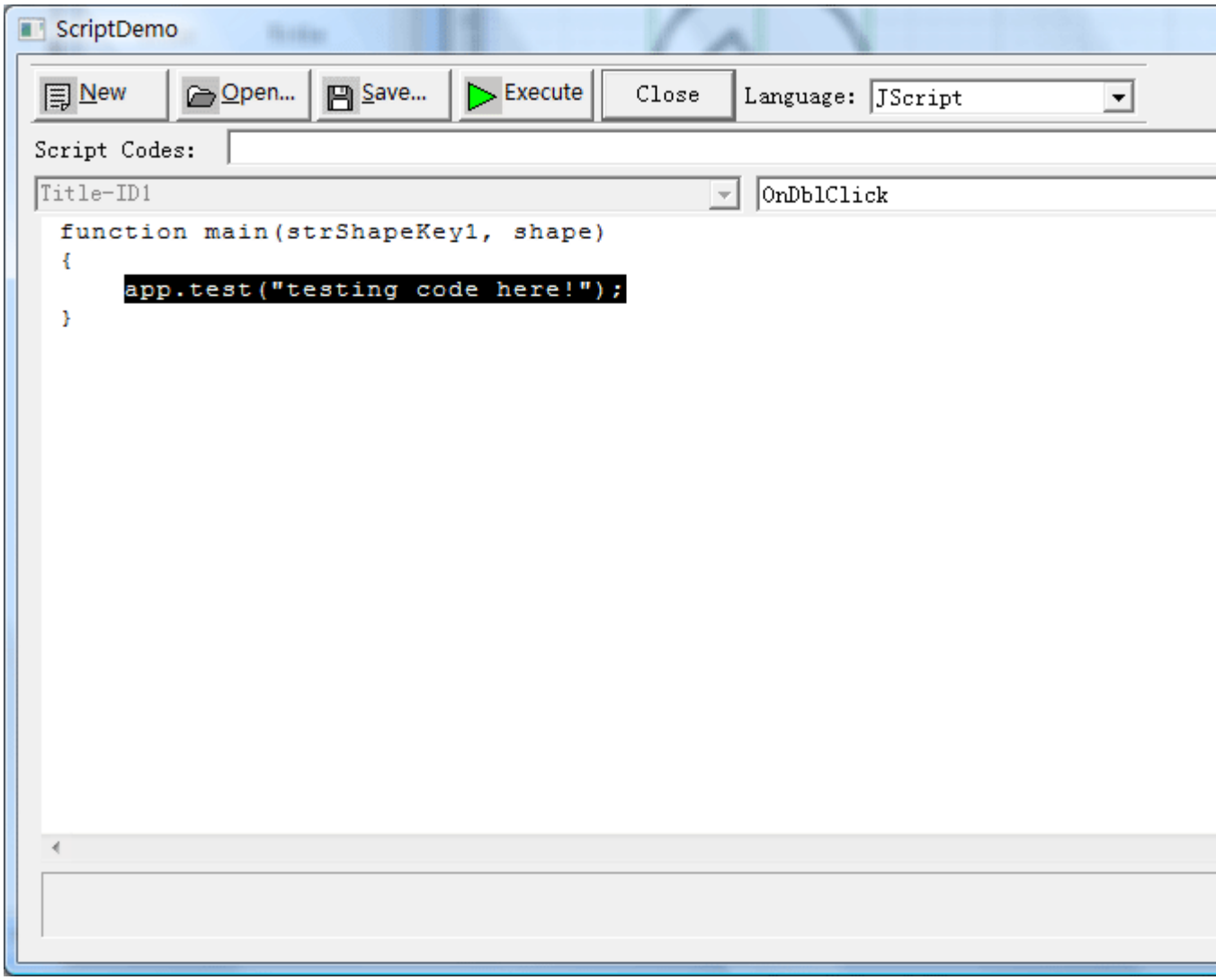
Do the following steps:

1). Running ScriptDemo.exe.

2). Drag and drop a shape on the canvas, as below:



3). Choose "Script | Write Script Code" menu item:



4). From the combo box list of Script Codes, you will find there are over 400 methods of E-XD++ are predefined. And you can call any of them with the following like:

app.

For example, if we want to create a rectangle shape on the canvas, please call:

app.AddRectShape(0,0,100,100,"test");

If we want to change the shape's line width, please call:

var xshape = app.AddRectShape(0,0,100,100,"rect");

app.ChangeSingleShapeIntProp(xshape, 20, 1701); // 20 is the new line width, 1701 is the property ID of line width.

If we want to find a shape with it's key 1 text, please call:

```
var xshape = app.FindShapeWithKey1("Key1");
```

5). How to execute the script codes?

To execute the script codes that inputted, jus click the "Execute" button of toolbar.

7 How to Create your own host?

Please add to MSN Messager: ucancode@hotmail.com, you can talk to our professional engineer for this question.