How to publish a Fairmat plug-in

Revision #2



Date: 2012/07/20

Fairmat plug-ins development

How to publish a plug-in

Index

2
2
3
6
10
11
11
12
13
14

1 Introduction

This tutorial shows how to build from scratch and how to publish a plug-in for Fairmat Academic.

2 Setup a development environment

For this tutorial we used Windows and the Visual C# 2010 Express Edition IDE, but note that the same things can be done using other systems or other IDEs such as Mono Develop (<u>http://www.monodevelop.com</u>) or Sharp Develp (<u>http://www.icsharpcode.net/OpenSource/SD/</u>).

To download *Visual C# 2010 Express Edition*, launch your web browser and go to this address: <u>http://www.microsoft.com/Express/Download/</u> and choose "Visual C# 2010 Express", download and install it. **You should register your copy** of *Visual C# 2008 Express Edition* within 30 days. The product is free, and registration is free, but Microsoft does require you to register it in order to use it for more than 30 days.

3 Project setup

Start Visual C# 2010 Express Edition create new Project and select a Class Library type project, insert the name (eg. "FairmatPluginTest") in the field "Name" and press OK Button

New Project	×
Templates: Visual Studio installed templates Windows Class Library VUPF WPF Browser Console Forms Ap Class Library My Templates Image: Class Library Search Online Te	
A project for creating a C# class library (.dll) (.NET Framework 3.5)	
Name: FairmatPluginTest	
OK Canc	el

Note:

Visual C# 2010 Express Edition uses the .NET Framework 4.0 as default, which is fine, while Visual C# 2012 RC Express Edition uses the .NET Framework 4.5. In this latter case you need to change the target framework to .NET 4.0.

Now, in "Solution Explorer" tab

· · · · · · · · · · · · · · · · · · ·					
ClassLibrary1 - Microsoft Visual C# 2008 Express Edit	ion (Administrator)				
<u>Eile Edit View Refactor Project Build Del</u>	oug D <u>a</u> ta <u>T</u> ools <u>V</u>	<u>V</u> indow <u>H</u> elp			
🖥 🖽 • 😂 🖬 🖓 🗴 🛍 🛍 🔊 • 🗠 🤤	1 - II,)	*		- 2	- 💀 🕾 😨 谋
) 🔒 🤿 📱				
Class1.cs Start Page				+ ×	Solution Explorer - Solution 'ClassLibrar 👻 🗜 🗙
ClassLibrary1.Class1	-			-	
g using System;				_	Solution 'ClassLibrary1' (1 project)
using System.Collections.Gener	ic;			<u>^</u>	Classibility
using System.Ling;					References
- using System.lext;					🛄 Class1.cs
namespace ClassLibrary1					
(E	
public class Class1					
- 3					
L ₃ '					J
					Properties _ I X
2 0 Errors A 0 Warnings (i) 0 Messages				↓ # ×	C .
Description	File	Line	Column	Project	
					-
Ready				Ln 1	Col1 Ch1 INS

righ-clicking on "Class1.cs", and selecting "Rename" from the menu, Rename the file (eg. "TestPlugin.cs"), then press Enter key and, when system asks, confirm to change the class name:

S	ystem	.Xml.Linq											
····· 省 Clas	ß	Open	Micro	osoft	t Visual	l Studio							- 23
portion		Open With	-										
s1 cs File Pro	E	View Code		2	You	are renam	ing a file. \	Would you	i also like	to perform	n a rena	me in this	;
		Exclude From Project		•	proj	ect of all re	erences t	o the code	elemen	t Classi (
uild Action	₩	Cut											_
Copy to Outp		Сору								Kes		<u>N</u> o	
Custom Tool	X	Delete							e e	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~			
Custom Tool		Renarce											
ile Name		Properties	ŀ										

Click OK and you can view the changes in Code tab:

TestPlugin.cs Start Page
🕫 ClassLibrary1.TestPlugin
using System;
using System.Collections.Generic;
using System.Ling;
using System.Text;
<pre>namespace ClassLibrary1 { public class TestPlugin { } } </pre>

Now you must reference Fairmat Libraries, to do this right-click on "References" in Solution Explorer and select "Add Reference" and then select "Browse" tab and in "Look in" field select

"C:\Program Files\Fairmat Academic" directory

Add Reference	? 💌
.NET COM Projects Browse Recent	
Look in: 🚺 Faimat Academic <	G 🤣 📂 🛄 -
Name Recent Items	Date modified
Plug Stop	09/09/2010 15:34
🚳 Drav 📄 Libraries	09/09/2010 11:05
🚳 DVP 🔒 axl	09/09/2010 11:05
DVP My Documents	09/09/2010 11:05
DVP Visual Studio 2008 Projecte	09/09/2010 11:05
DVP Computer	09/09/2010 11:05
Floppy Disk Drive (A:)	00/00/2010 11:05
Local Disk (C:)	4
File name Program Files	
Files of typ CD Drive (D:) 43	(e;*.manifest) ▼
shots	
	OK Cancel
	Cancel

Select DVPLI.dll library and press OK button

Add Reference	
Look in: 🌗 Fairmat Academic	- 🕝 🎓 📂 🛄-
Name	Date modified 🔺
DVPLDOM.dll DVPLDOME.dll	09/09/2010 11:05 09/09/2010 11:05
DVPLI.dll	09/09/2010 11:05
DVPLInterfaces.dll	09/09/2010 11:05
DVPLModeling.dll	09/09/2010 11:05
DVPLSolverC.dll	09/09/2010 11:05
N/DI Triggers dll	00/00/2010 11:05
<	4
File name: DVPLI	•
Files of type: Component Files (*.dll;*.tlb;*.olb;*.d	ocx;*.exe;*.manifest) ▼
-	OK Cancel

In Solution Explorer, right-click on "DVPLI.dll" and select "Properties"



Below Solution Explorer tab you can see the Properties tab that shows properties of DVPLI.dll library, change "CopyLocal" property to False, because this library is already shipped with software (in future versions it will be signed and will be available on the GAC)

As you did with DPLI.dll, you may reference also DVPLDOM.dll, Mono.Addins.dll, DVPLInterfaces.dll libraries.

4 Writing the plug-in

Begin to code a simple plug-in that adds a constant to Fairmat workspace:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

using DVPLI; using DVPLDOM; using Mono.Addins; using System.Windows.Forms;

```
[assembly:Addin("FairmatPluginTest","1.0",Category="Test")]
[assembly:AddinDependency("Fairmat","1.0")]
```

```
namespace FairmatPluginTest
```

```
{
  [Extension("/Fairmat/UIConstant")]
  public class TestPluginConstant : IUIConstant
  {
    #region IUIConstant Members
    public double Value
    {
      get { return 3.1415; }
    }
    #endregion
    #region ISymbolDefinition Members
    public string Description
    {
      get { return "PI Constant"; }
    }
    public string Name
    {
      get { return "PI"; }
    }
```

```
#endregion
```

}

}

Now that we have a complete program, let's look at the code to see what each part is for—all of the pieces are things you'll deal with every time you write in C#. Starting from the top, the code has several lines beginning with using:

using System; using System.Collections.Generic; using System.Linq; using System.Text;

These *using directives* help the C# compiler work out what external code this particular source file will use. No code is an island—to get any useful work done, your programs will rely on other code. All C# programs depend on the .NET Framework class library.

Using directives can declare an intent to use classes from any library—yours, Microsoft's, or anyone's. All the directives in our example start with System, which indicates that we want to use something from the .NET Framework. This text that follows the using keyword denotes a *namespace*.

Fairmat plugins need to reference the following libraries:

using DVPLI; using DVPLDOM; using Mono.Addins; using System.Windows.Forms;

Fairmat extensions (plug-ins) uses Mono.Addins extension model that is based on four concepts:

- Add-in host: an application or library which can be extended by add-ins. Extension possibilities are declared using extension points.
- **Extension point**: a placeholder where add-ins can register extension nodes to provide extra functionality. Extension points are identified using type names or extension paths.
- **Extension node**: an attribute-decorated element that describes an extension. Extension nodes are typed. Extension points may declare which types of extension nodes do they accept.
- Add-in: An assembly and/or other files which register new nodes in one or several extension points defined by add-in hosts. An add-in can also act as an add-in host, and as such it can be extended by other add-ins.

Mono.Addin defines an Add-in Description Model, which is used by add-ins and add-in hosts to declare all extensibility information. Add-in descriptions can be represented either using an XML manifest, or by applying custom attributes to assemblies and types.

Mono.Addins also provides an API (implemented in Mono.Addins.dll) which can be used at runtime to query and handle add-in extensions



Add-ins have to be marked with the [Addin] attribute to be recognized as add-ins. Add-ins must also declare the add-in hosts they are extending by using the [AddinDependency] attribute. Notice that an add-in can extend several add-in hosts, and it can even extend other add-ins which declare their own extension points. The AddinDependency attribute must specify the host/add-in id and its version number.

In our example code we can see the lines:

```
[assembly:Addin("FairmatPluginTest","1.0",Category="Test")]
[assembly:AddinDependency("Fairmat","1.0")]
```

where the first line declares that we are implementing a plug-in (extension) with following properties:

Name: "FairmatPluginTest"(Plugin Name)Version: "1.0"(Plugin Version)Category: "Test"(Plugin Type)

and the second line declares that the plug-in depends on add-in host implemented in Fairmat.

The plug-in's category output can be defined by the user and represents the extension type, actually Fairmat use following values:

- Core [reserved value]
- Modeling
- Stochstic Process
- Calibration

By applying the **[Extension]** attribute to a class we are declaring a new extension node of that type which is added to an extension point. The extension point is determined by looking at the base types and interfaces of the class. So in this example the class implements the ICommand interface, and there is an extension point defined for that interface, so that's the extension point where the type will be registered. If a class implements several interfaces, we can select the extension point by specifying the type name in the attribute.

In our plug-in example we extend "/Fairmat/UIConstant" node using:

[Extension("/Fairmat/UIConstant")] public class TestPluginConstant : IUIConstant

Note that the extension class must implement methods and properties defined by IUIConstant interface associated to "/Fairmat/UIConstant" extension node.

In documentation reference (<u>http://www.fairmat.com/software/DVPL.web</u>) you can see that IUIConstant interface defines only the property "Value" that contain the numerical value for the constant symbol we are defining and inherit from interface ISymbolDefinition these properties:

- Description
- Name

that define name and description of the symbol that represents our constant in the Fairmat workspace.

Note:

You have to add the following lines:

```
[assembly:Addin("FairmatPluginTest","1.0",Category="Test")]
[assembly:AddinDependency("Fairmat","1.0")]
```

once for each assembly, alternatively you can add these to AssemblyInfo.cs file if present.

Declaration S	yntax			
C#	Visual Basic	Visual	C++	
public interface	IUIConstant : IS	nbolDe	fini	tion
Public Interface Implement	IUIConstant _ ts ISymbolDefinit;	ion		
public interface	class IUIConstant	t : ISy	mboli	Definition
Members				
All Members	Properties			
V Public		🔽 Inst	tance	🗹 Declared
Protected		🗸 Sta	tic	🗵 Inherited
Icon Member			Desc	ription
Pescription			(Inhe	rited from ISymbolDefinition.)
🚰 Name			(Inhe	rited from ISymbolDefinition.)
🚰 Value				

4.1 Programming Tricks

You can obtain a skeleton implementation of properties and methods defined in an interface by right-clicking on the interface name in the class declaration and selecting "Implement Interface" as depicted in figure:

```
[Extension("/Fairmat/UIConstant")]
public class TestPluginConstant : IUIC
                                                   Implement Interface >
                                                                            Implement Interface
{
                                                   Refactor
                                                                    ۲
                                                                            Implement Interface Explicitly
}
                                                   Organize Usings
                                                                    Þ
                                                   Insert Snippet...
                                               ٩,
                                               Surround With...
                                               Go To Definition
```

This is the result:

```
[Extension("/Fairmat/UIConstant")]
public class TestPluginConstant : IUIConstant
{
  #region IUIConstant Members
  public double Value
  {
    get { throw new NotImplementedException(); }
  }
  #endregion
  #region ISymbolDefinition Members
  public string Description
  {
    get { throw new NotImplementedException(); }
  }
  public string Name
  {
    get { throw new NotImplementedException(); }
  }
  #endregion
}
```

5 Build & Deployment

5.1 Testing setup

To test your plug-in you have to copy the generated plug-in in the Fairmat plug-in directory which is "<u>C:\Program</u> Files\Fairmat Academic\Plugins" or setup your development environment to do this programmatically for you using Project Properties:



Select "Build" tab, set the field "Output path" to:

"C:\Users\<user>\AppData\Roaming\Fairmat Academic\Plugins\addins" (where <user> is your user name) and save the project.

TestPlugin.c [*] Fairm	atPluginTest* Start Page	
(
Application	warning ievei:	4
Build*	Suppress warnings:	
Build Events	Treat warnings as errors	
Debug	None	
Resources	Specific warnings:	
Services	C:\Us	ers\ <user>\AppData\Roaming\Fairmat Academic\Plugins\addins</user>
Settings	Output	
Reference Paths	Output path:	C:\Users\ <user>\AppData\Roaming\Fairmat Academic Browse</user>
Signing	XML documentation file:	
	Register for COM interop	
	Generate serialization assembly:	Auto
		Advanced

If you have modified the project output path, you can build it simply by pressing F6 keyboard otherwise you must copy the generated *.dll to path "C:\Users\<user>\AppData\Roaming\Fairmat Academic\Plugins\addins" manually.

Note that the folder "AppData" is a hidden folder and By default and Windows 7 does not show hidden files.

If you need to manage files under a hidden folder, you'll need to be able to view that folder.

Here's how:

- 1. Type folder options in the search box after clicking Start.
- 2. Choose Folder Options under Control Panel from the list of results and
- 3. Click on the View tab in the Folder Options window.
- 4. In the Advanced settings: area, locate the Hidden files and folders category.
- 5. You should also see two options under the folder.
- 6. Choose the Show hidden files, folders, and drives radio button under the Hidden files and folders category.
- 7. Click OK at the bottom of the Folder Options window.

5.2 Testing the plug-in

Run Fairmat, open or create a new document and in the "Settings" menu you select "Plugin Settings", the opened window shows the plugin list where you can check if your new plug-in was loaded

stalled plugins Available online	plugins Adv	vanced				
Name	Version	Category	Author	Description	Enabled	Refresh
DVPLUtils	1.0.1	Core			True	
Fairmat Core Extensions	1.0	Core			True	Disable
airmat Core Solver	1.0	Solver			True	
airmat Core	1.0	Core			True	Enable
aimat Modeler	1.0	Modeling			True	
aimatPluginTest	1.0	Test			True	
Nonte Carlo Simulation Solver	1.0	Solver			Tide	

Select the Fairmat Console Tab, insert the name of the constant defined in our plug-in and press the

Enter Key.

🔋 untitled - Fairmat Academic			- • •
File Edit Analysis Settings	Tools Help		
i 🗋 💕 🔒 🚷 i 🔶 🔶 🔶	🗮 Σ < 🏪 IF F0		
Current scenario Default	 Analysis Valuation 	 Starting node 	• ₌
Structure Data Sc			
Parameters &			
Stochastic Processes			
Discounting			
Option Map > PI			
Manage Scenarios 3,14			
	-		
> PI			
Valuations	Errors Parameters Log Fa	airmat Console	

If all works correctly, you can see on the video the constant value we previously defined.

5.3 Notes on Fairmat Log:

All Fairmat programmatic calls to methods Console.Write or Console.WriteLine are redirected to Fairmat Log tab.



In order to control the amount of outputs a plug-in should write, you can condition quantity of output's to be written by checking the integer value of DVPLI.Engine.Verbosity defined in Fairmat using for example the following code block:

```
if(Engine.Verbose >= threshold)
{
    Console.Write("Comment...");
}
```

where "threshold" is an integer value that can be: 0 (no output),1 (contained) or 2 (verbose) You can control the value of this variable from the user interface, selecting "Fairmat Settings" in "Settings" menu.

Fairmat Preferences	×
Modeler Preferences Plug-ins preferences	
General Preview Advanced	
Number of decimals shown on outputs 2	
Repeated Analysis: number of replications 10	
Generate Latex Friendly outputs	
Verbosity level (0: no output 1: contained , 2: more information	
Ok Canc	al

6 Publish the plug-in in the Fairmat website

If you think your plug-in can be useful to others, you can publish it in the Fairmat website and if your plug-in is open-source, you may join the Fairmat revenue sharing program agreement (see http://www.fairmat.com/revenue-sharing/ for more details).

Note that the functionalities described below are not being ported to our new system, if you want to publish a plug-in just contact us.

In order to officially release a plug-in you have to go to Fairmat web site at address: <u>http://www.fairmat.com</u> and register as developer selecting "Plug-ins - area" on the "Developers" menu.



After that you can login to developers area or if you don't have a developer account you have to sign up by clicking the link depicted in figure:

Farmat Plug-in area	Home	Project	Features	Plugins	
Email Password					
Login In order to access the plug-in area you must log-in. If you do not have a developer account yet, you must sign up here.					

To sign up you have to fill the registration form

In order to access to the plug-in's devel compile the following form.	oper area, where you can upload and manage your own plug-ins please
First name	John
Last name	Doe
Email	john.doe@example.com
Please write a valid email address given activation codes will be emailed to you la	that account information and ater.
Address	Camelot str.
City	Avalon
Zip code	12345
Country	HOLY SEE (VATICAN CITY S

you have to agree on the license and press the "Submit" button

REGISTRATION (in accordance to page (PLUG-IN H the time of reg same may be free parties. In to developed under commercial pote Fermat, at its	On releasing the plug-in(s) developed with the procedure set out in the web RELEASE PAGE) the User, as detailed at pistration, hereby accepts that the eely accessed and used by third the case of plug-ins that have been of GPL3 license and that may have ential, these will be selected by sole discretion, and a request to
If you want to prin	t the registration disclaimer press here
To sign up you hav	e to agree to the above statements.
💽 I agree	💿 I don't agree
	Submit

then you can see the registration confirmation and a mail with password is sent automatically to the mail address you provided in registration form.

Developer sign-up
You have succesfully signed up. Your developer account details have been emailed to you. You can proceed in the plug-in area.

Once you are logged in you can manage and upload your plug-ins



In order to upload your plug-in, you have to prepare the following packages:

- Binary Package: after compilation, go to your project's binary folder and create a zip file containing the plug-in libraries and its dependencies. Note that the system will not accept zip archives containing more than on plug-in. This means that if you have more than one assembly containing the attribute assembly: Addin you have to separate them in different archives.
- Source Package: clean the project and create a zip file containing the entire solution. If you're using VC# 2008 Express Edition, then the Clean commands are not on any menus by default. You can customize your menus by right-clicking on empty space on either the menu bar or toolbar area, and selecting "Customize...". You'll be able to search for the Clean commands in the dialog that appears, and then you can drag-and-drop it to the menu of your choice.
- Documentation: you can add a Pdf or a text file which explains how to use your plug-in in Fairmat.

In the "Requirements" area you must select the packages that are required for the execution of your plugin and that will be automatically provided when download it.

Press upload button, then compile the fields "Description" and "Version Date" of the resulting form and press "Save" button.

Name **	
Description *	Simple Plugin Test
Tags (comma separated)	
Repository for Fairmat V.	1.0 💌
Plug-in Version **	
Version date *	16/09/2010
Requirements	 Binomial Lattice Solver Pelsser model Heston model Pelsser model estimator Libor Market Model IR model EstimateDB TestExtensions FairmatPluginTest
Upload files Source * C:\Users\ Documentation	axl\Documents\Visual Studio 2008\Projects\FairmatPlus
Binaries * C:\Users\	axl\Downloads\FairmatPluginTest.zip
© 5.4	
Source *	
Documentation	
Binaries *	
 * Mandatory field. ** Grayed fields are filled a Save Cancel 	utomatically from the information present in the plug-ins binary.

When your plug-in is uploaded correctly, you can see the confirmation:

fair	mət ""	ne Project Feature	es Plugins	Developers	News	Fo
Plug-in ar	ea					
In this area you can i	manage your plug-ins					
Plugin succesfully se Upload	aved					
Name	Status	Created at	Updated at			
FairmatPluginTest	waiting for validation	16/09/2010 10:17:5	6	Modify	Delete	

Notes:

Your plug-in have to wait for validation from Fairmat Srl before it would be available to the others. You can check this on the plug-in status area in your personal page.