I Wish I Knew How To …

Program Plugins

With Xojo on Windows

May 2015 Edition (2.1)

For Visual Studio 2013 Community Edition

By Eugene Dakin

Table of Contents

Module Properties

Properties are variables which can be used in the dll or within Xojo. An example would be the radius of a circle which can be changed – not all circle radiuses are the same. A property is able to have its value changed, many times, according to what the user and/or programmer would like. Data types which can be created in a property are listed in the Data Types section of Chapter 3.

The quick steps to create a module property are shown below.

Condensed Notes

1) Start Visual Studio 2013 Community Edition
2) Start *File->New->Project*
3) Select *Templates->Visual C++,* and select *Win32 Project*
4) Choose the location for the files to be placed, I made Example5-3 folder at the location: D:\user\Xojo\Eugene\Plugins\Chapter5\Example5-3
5) Name the project *Prop*
6) Press the *OK* button
7) Press *Next* button
8) Select *DLL, Export Symbols, and Precompiled Header (unselect everything else)*
9) Press Finish
10) Copy the following files from the directory: C:\Program Files (x86)\Xojo\Xojo 2015r1\Extras\PluginsSDK\Includes and paste them in the directory: D:\user\Xojo\Eugene\Plugins\Chapter5\Example5-3\Prop\Prop (rb_plugin.h, rb_plugin_cpp.h, REALplugin.h, and WinHeader++.h)
11) Copy the following file from the directory: C:\Program Files (x86)\Xojo\Xojo 2015r1\Extras\PluginsSDK\GlueCode and paste them in the directory: D:\user\Xojo\Eugene\Plugins\Chapter5\Example5-3\Prop\Prop (PluginMain.cpp)
12) Remove the following files from Visual Studio: stdafx.h, stdafx.cpp, ReadMe.txt, targetver.h, and dllmain.cpp.
13) Drag and drop files from the folder: D:\user\Xojo\Eugene\Plugins\Chapter5\Example5-3\Prop\Prop into the Visual Studio Solution Explorer: rb_plugin.h, rb_plugin_cpp.h, REALplugin.h, WinHeader++.h, and PluginMain.cpp.
14) Delete the file Prop.h
15) Delete all the code in Prop.cpp and add the following code:

```
#include "WinHeader++.h"
#include "rb_plugin.h"

// Dimension the methods
```

```
// use DOUBLE (not double) for Xojo
static DOUBLE RadiusGetter(void);
static void RadiusSetter(DOUBLE RadiusValue);

//Make properties
REALproperty PropModule[] = {
	{ "", "RadiusValue", "double", REALconsoleSafe, (REALproc)RadiusGetter,
(REALproc)RadiusSetter },
};

//Create property module
REALmoduleDefinition PropModuleDefinition = {
	kCurrentREALControlVersion, //Version
	"Prop", //Module name
	nil, //Method names
	0, //Name count
	nil, //Constant array
	0, //constant count
	PropModule, //Module properties
	sizeof(PropModule) / sizeof(REALproperty), //Property count
	nil, //Structure array
	0, //structure count
	nil, //Enum array
	0, //enum count
};

//Perform property functions
static DOUBLE DBRadius;
static DOUBLE RadiusGetter(void)
{
	//Return the DOUBLE value
	return DBRadius;
}
static void RadiusSetter(DOUBLE AValue)
{
	  //Store the double value
	DBRadius = AValue;
}
void PluginEntry(void){
	REALRegisterModule(&PropModuleDefinition);
}
```

16) Change Prop Property->C/C++->Precompiled Headers to *Not Using Precompiled Headers* in Debug and Release Configuration.
17) In the main IDE, select Release Win32, and *Rebuild* solution. There should be two *#pragma deprecated* warnings and there should be 1 succeeded, 0 failed, and 0 skipped
18) Copy the dynamic link library Prop.dll from the D:\user\Xojo\Eugene\Plugins\Chapter5\Example5-3\Prop\Release folder and place it in the Xojo plugin folder at: C:\Program Files (x86)\Xojo\Xojo 2015r1\Plugins
19) Start Xojo and add a pushbutton action event and add the following code and run the program

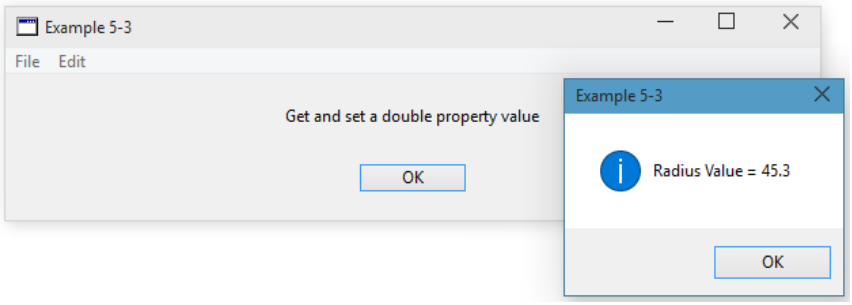
```
//Assign a value
Prop.RadiusValue = 45.3
//Get the property value
MsgBox "Radius Value = " + CStr(prop.RadiusValue)
```

The number 45.3 should appear in the messagebox which was a properties stored and retrieved from the plugin.

Explained

Example 5-3 shows how to set a property value with a Xojo Plugin and also to get the value. On Xojo the property value will use code that is the same as a module created in Xojo.

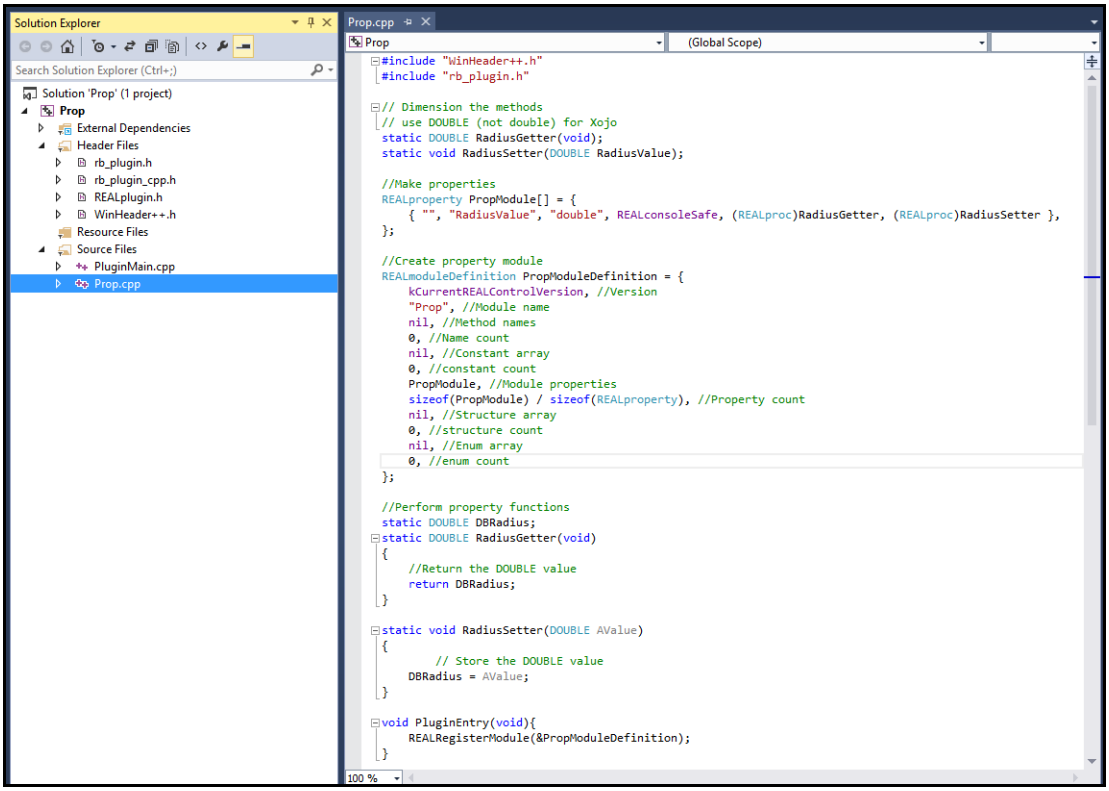
Figure 39. Example 5-3: Xojo Screen Grab



One property is created in Visual Studio 2013 and has two methods, a setter and a getter method. The setter method takes a numerical value and places it in the property value. The getter method retreives the property value and returns this value back to the variable in Xojo.

Code for this example is shown below.

Figure 40. Example 5-3: Variable Property VS 2013 Code



Above code for Example 5-3 begins the same way as other module plugins with the WinHeader++ and rb_plugin headers being *included*. Both methods are declared at the beginning of code and then the functioning code can be placed after REALmoduleDefinition. If functional code is placed after REALmoduleDefinition and no declaration of the method exists at the before REALmoduleDefinition (example: static DOUBLE RadiusGetter(void);) then an error will occur when compiling the program.

REALproperty defines the properties used for a single property. When a number is to be placed in RadiusValue, the RadiusSetter method is called. Getting data from the property will require the RadiusGetter method to be called. The array PropModule is can store many properties.

REALproperty has no value (“”) for the group, the sub name to be used in Xojo for accessing this property is RadiusValue, and this is a double value. There are a seven possible flags for REALproperty (see Chapter 3 PluginSDK Reference) and this is console safe, which can be used in console and desktop projects. Both RadiusGetter and RadiusSetter procedures (REALproc) are declared.

Code 8. Property Module Definition

```
//Create property module
REALmoduleDefinition PropModuleDefinition = {
	kCurrentREALControlVersion, //Version
	"Prop", //Module name
	nil, //Method names
	0, //Name count
	nil, //Constant array
	0, //constant count
	PropModule, //Module properties
	sizeof(PropModule) / sizeof(REALproperty), //Property count
	nil, //Structure array
	0, //structure count
	nil, //Enum array
	0, //enum count
};
```

The property definition begins with the version of the module which is almost always used the constant kCurrentREALControlVersion. The name of the module is the word that is typed before the added methods and in this case is *Prop,* which is the shortened version for property. All of the properties have been defined in the array *PropModule* and are added and the number of constants (using sizeof) are also added. Since there are no methods, constants, structures, or enums, then these are all left as Nil and 0 values. Later examples in this chapter will show how some of these are used.

The functional part of the properties are then coded.

Code 9. Functional Properties

```
//Perform property functions
static DOUBLE DBRadius;
static DOUBLE RadiusGetter(void)
{
	//Return the DOUBLE value
	return DBRadius;
}
static void RadiusSetter(DOUBLE AValue)
{
		// Store the DOUBLE value
	DBRadius = AValue;
}
```

The variable DBRadius (shortened for DoubleRadius) is only used in VS 2013 which holds the RadiusValue number. When Xojo wants to retrieve the value from the property RadiusValue, the method RadiusGetter *gets* the value by Xojo and returns it in this method. Similarly, when a

new value is placed in the property (Prop.RadiusValue = 45.3) then the value is stored with the RadiusSetter method.

All of the created properties which have been built need to be registered in the module.

```
void PluginEntry(void){
	REALRegisterModule(&PropModuleDefinition);
}
```

The pluginEntry is where the properties are registered with the REALRegisterModule command. All of the properties are added – usually in arrays.

Once the dll has been built with the Build-> Rebuild Solution menu item, the dll is copied to the Xojo plugin folder. Xojo is restarted to load the plugin data and the following code is added to a pushbutton in a new Xojo program.

Code 10. Example 5-3: Xojo Program with dll Properties

```
  //Assign a value
 Prop.RadiusValue = 45.3
 //Get the property value
 MsgBox "Radius Value = " + CStr(Prop.RadiusValue)
```

Code where the value of 45.3 is placed in the property calls the RadiusSetter method in the dll, and retrieving the data (Cstr(Prop.RadiusValue)) gets the value by calling RadiusGetter in the dll.

Example 5-3 shows how to create a double numerical property that can get and set the value.

Module String Properties

String properties in a module are slightly different than numerical values because the string reference should be locked and/or unlocked with use, otherwise the string can be released which could cause errors with a dangling pointer. This method also includes the previous

Index

The 'I Wish I Knew' series contains technical data and advice that makes sense and contains practical and numerous examples with explanations to allow you to ease into the steep programming curve. You can extend Xojo applications today!

This book "I Wish I Knew How to … Program Plugins with Visual Studio 2013 with Xojo for Windows" shows you how to create a plugin with FREE community Visual Studio 2013. All code has been tested on Xojo 2015 r2.

The book is written as a guide and reference to Xojo programmers who program Desktop Applications in Windows. Dynamic Link Libraries are created with Visual Studio to allow extra functionality in Xojo.

There are 8 chapters and contains over 300 pages with over 27 example programs.

Examples include creating a manual plugin, plugins from prebuilt templates, class constants, classes, modules, and controls. Many screenshots have been added to show the results of the code with an index to help find topics quickly.

This is one of many books at Great White Software. This book can be purchased at http://great-white-software.com/rblibrary/ where many great Xojo and Real Studio resources are available.

Happy programming!

Eugene

Eugene Dakin MBA, Ph.D., P.Chem., is an author of Xojo and Real Studio reference materials and has many years of experience in the programming industry. Another great reference book is *I Wish I Knew How To … Program the Canvas*.

ISBN: 978-1-927924-09-9