I Wish I Knew How To …

Begin Object Oriented Programming With Xojo

July 2015 Edition (1.0)

By Eugene Dakin

Table of Contents

Chapter 4 - Encapsulation

Encapsulation is a common word when Object Oriented Programming is used. Encapsulation has all of the objects data hidden within the object and can only be accessed through the methods of the class. Here are some of the many benefits to hiding (making properties Private) which allows the validation of changes to the data, protecting the data from unauthorized access, and allows the data to be consistent.

Scope

With classes there are three types of scope in Xojo: Public, Protected, and Private. Scope is also called the 'visibility' of the entity (eg. Property, Method, etc.).

Scope is very important when working with encapsulation. The general definitions of scope are:

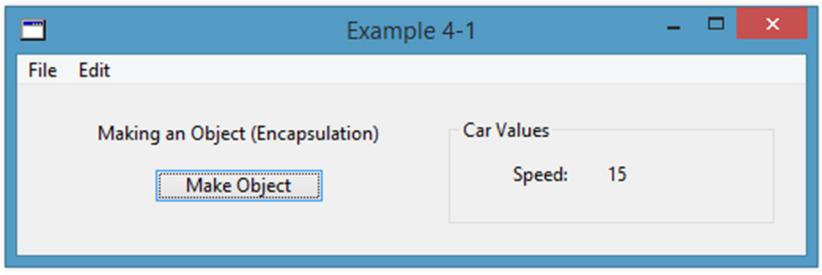
a) Public – All methods, Windows, and Objects can see and likely access the entity (i.e. Property) within a class.
b) Protected – The class and any subclasses can access and modify the property/method.
c) Private – Only the class can access and modify the property/method.

None of the properties/methods are given access from outside of the program.

Encapsulated Property

This example will make a property (Speed) which is invisible to the outside program and is only accessible by calling methods in the Class by setting the property to Private.

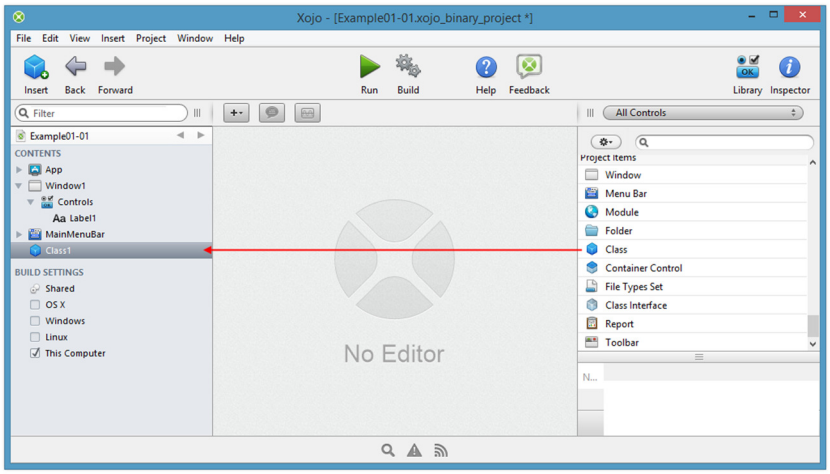
Figure 14. Example 4-1: Encapsulating a Property Screen Grab



The object is made and property is set with a private scope, meaning only a class method can change the property value. Lets show how this program is made.

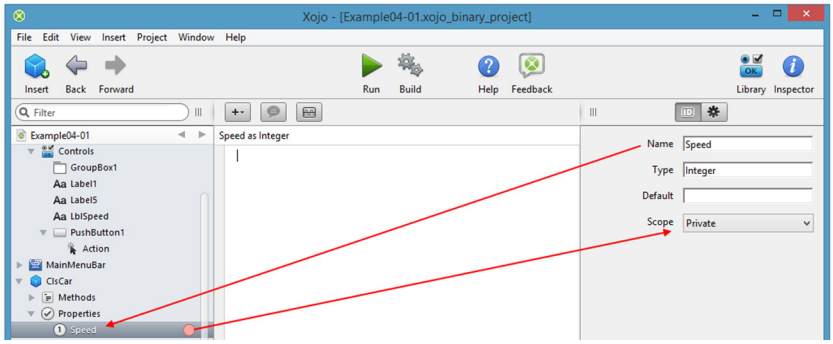
The first step is to create a blank desktop application and drag-and-drop a class

Figure 15. Example 4-1: Encapsulated Property Screen Grab



Class1 was renamed to ClsCar and a property called *Speed* was added to the car. When the properties scope was changed to *Private*, then a red dot appears to the right of the property.

Figure 16. Adding a Private Class Property



A private property can only be changed by the methods within the class. Two methods with public properties were created that can be accessed outside of the class.

Code 24. Example 4-1: Class Constructor Method

```
Function Constructor()
```

```
  //Set PRIVATE Default Class Property value
  Me.Speed = 15 //Car has zero speed
End Function
```

Using our knowledge from the previous chapter, a method called a Constructor is added to the class and is automatically called when a new object from a class is created. Code in the Constructor method sets the speed of the object to 15.

Another method is created to update the Window label from the class.

Code 25. Example 4-1: Class UpdateLabels Method

```
  //Show MyFastCar Settings in Window
  Window1.LblSpeed.Text = Me.Speed.ToText
```

This is the part of the method where the value in the speed variable is shown on a label in the window.

To make all of the methods and property work together, the following code is added to the pushbutton action event.

Code 26. Example 4-1: Create Encapsulated Object

```
  //Create a Fast Car
  Dim MyFastCar as New ClsCar
  //Property values are set with the method
  MyFastCar.UpdateLabels()
```

When the MyFastCar variable is created from the Class ClsCar, then the Constructor fires and places a default value into the *Speed* property. The Class *UpdateLabels* method is then called to show the property value on the label in Window1 which has the value of 15.

This example shows how to encapsulate (hide) a property so that only methods in the property can change or be updated by public methods in the class.

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 … Begin Object Oriented Programming with Xojo" shows you how to program with object in the three major operating systems. All code has been tested on Xojo 2015 r2.2 with Windows 8.1, OS X Yosemite 10.10.4, and Ubuntu 15.04 32-bit versions.

The book is written as a guide and reference to Xojo programmers who program Desktop Applications.

There are 12 chapters and contains over 90 pages with 29 example programs.

Examples include topics such as Polymorphism, Introspection, Extends, Explicit Unboxing, Overriding, and definitions of common terms used in OOP. 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 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 Win32 Declares for Windows*.

ISBN: 978-1-927924-12-9