UML for REALbasic

Excel Software

www.excelsoftware.com

REALbasic is an object-oriented programming language for Mac OS X, Windows and Linux computers. The Unified Modeling Language (UML) is the industry standard notation for modeling object-oriented software. This paper shows how to extract a UML design from a REALbasic project, model REALbasic software using UML and generate REALbasic code from UML. Furthermore, it lays the foundation for a code translation project to or from REALbasic, Objective-C, Java, Delphi, C#, C++ or any other object-oriented programming language.

REALbasic is a rich programming language that supports windows, classes, class properties, class methods, class event, class interfaces, data structure, modules and public, private and protected scope for programming constructs.

UML specifies a family of graphical notations for describing and designing software systems. it defines many diagram types that present different views of the software design. This paper will focus on the UML class diagram, essential to any object-oriented design.

MacA&D is a complete UML modeling tool with a REALbasic code generator. MacTranslator is a companion reengineering tool that scans source code (including REALbasic) to extract design information from which UML class diagrams are automatically generated. WinA&D and WinTranslator offer similar REALbasic support on Windows.

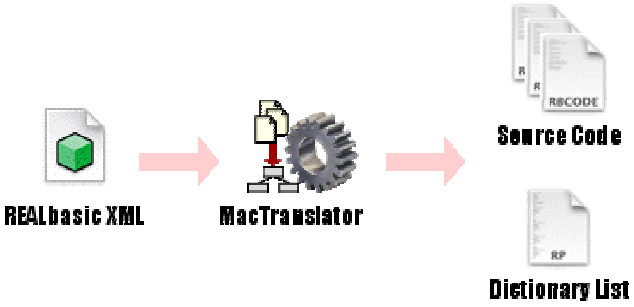
REALbasic Source Files

Most programming environments for C++, Java and Objective-C store source code in plain text files that can be edited from within the IDE, by any text editor or manipulated by various programming tools. A REALbasic project stores classes, windows, modules, etc. as binary files. Editing within the IDE is much like any programming environment, but source code is not as easily accessible to other tools and editors.

Accessing source code as plain text has many benefits when translating between languages, devices or operating systems, understanding a large code base or using other programming and design tools. The REALbasic IDE allows the source code for a method to be dragged and dropped as plain text. The Save As command also provides an XML Project format option for textual representation.

MacTranslator scans a REALbasic XML project file and outputs a dictionary list file that can be imported into the MacA&D modeling tool to generate a class diagram. MacTranslator also has the option of generating source code files that include all the essential declaration and programming logic. Likewise, MacA&D can generate source file files from a class diagram and data dictionary.

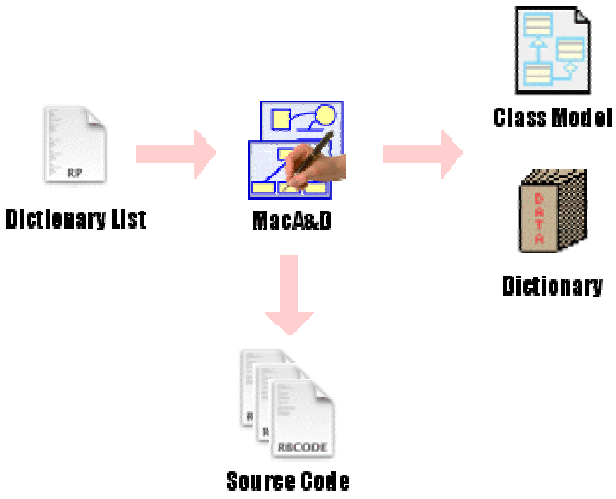
MacA&D and MacTranslator



Extract Design and Code from REALbasic Project

The process of extracting code files and a dictionary list from a REALbasic project is very simple. Open the project into REALbasic and save it as an XML Project file from the File menu. If the project references external code files, you may want to first create a temporary project file that directly includes all of the code files so a complete set of source code files and dictionary list can be extracted.

From MacTranslator, choose the REALbasic XML to Dictionary command and select the XML project file. After a few minutes of processing, the source code and dictionary list files are output to a folder.



Understand, Design and Generate REALbasic Code

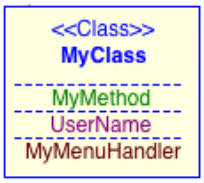
From MacA&D, use the New Project button to generate a new project that includes an empty Dictionary and Class Model document. Open these documents and choose the Import Dictionary command to select and import the dictionary list file produced by MacTranslator. From the class diagram, choose the Generate Class Model command.

REALbasic as UML

The UML class diagram graphically represents REALbasic code. Consider a class named MyClass that inherits the Control class and has one menu handler, method and property. The source code extracted by MacTranslator or generated by MacA&D would look like this.

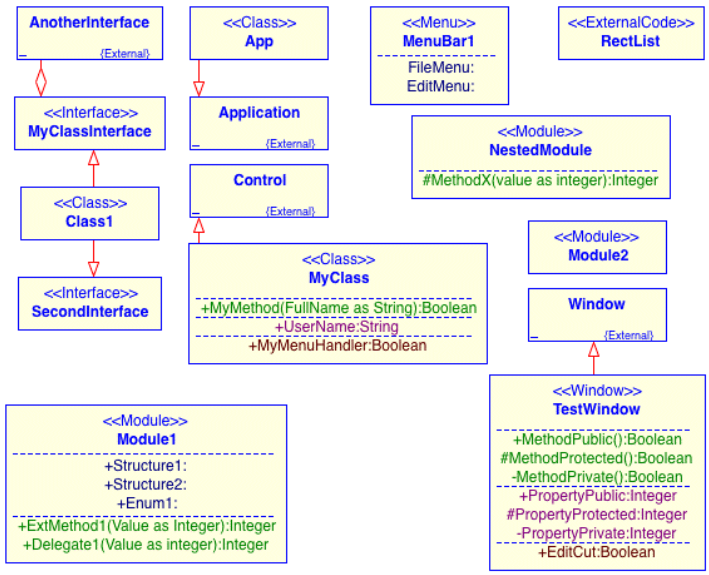
```
Class MyClass inherits Control

Public Event Function MyMenuHandler() As Boolean
MsgBox("Hello" + UserName)
Return True
End Function

Public Sub MyMethod(FullName as String)
UserName = FullName
End Sub

Public Property UserName As String
End Class
```

A UML class diagram represents the class as a named box with named class members (attributes, operations, properties and events). MacA&D has options to determine which class members are presented, the color and amount of detail (data type, arguments, etc.).



Class Object on Diagram

Many REALbasic language constructs (class, window, module, class interface, menu) are represented by a class box on the diagram with a stereotype name like <<Class>>, <<Window>> or <<Module>>) that specifies the type of language construct.



UML Class Diagram of REALbasic Language Constructs

From a quick glance, the class diagram reveals many important aspects of the project including its structure, type of constructs and important relationships. For example, you can see Class1 implements two class interfaces, MyClassInterface and SecondInterface. Furthermore, MyClassInterface aggregates another class interface named AnotherInterface. MyClass inherits the Control class that is external to the project since it is part of the REALbasic class framework.

Dictionary Information

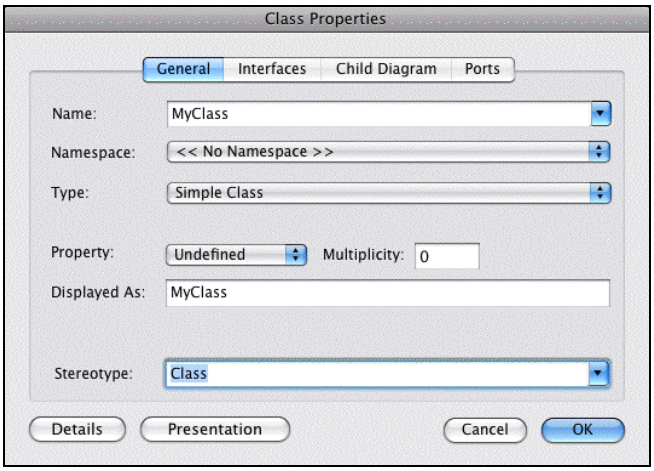
A class diagram in the MacA&D modeling tool has an associated data dictionary that contains all the detailed information behind the diagram. Each class object on the diagram has an associated dictionary entry. Likewise, class members such as attributes, operations, properties and events also have dictionary entries.

When modeling an object-oriented project in MacA&D, class objects are drawn on the diagram and connected with relationship lines. Class members are added by typing data into a details dialog. As information is added to the diagram, MacA&D automatically constructs the dictionary.

While a class diagram can be language independent, different programming languages can be selected to specialize the detail dialogs to the specific constructs of that language. Language specific details like data types, method arguments or stereotype names are stored in the dictionary entry. MacA&D will allow the designer to switch from one language to another and can store language specific information for multiple languages in each dictionary entry.

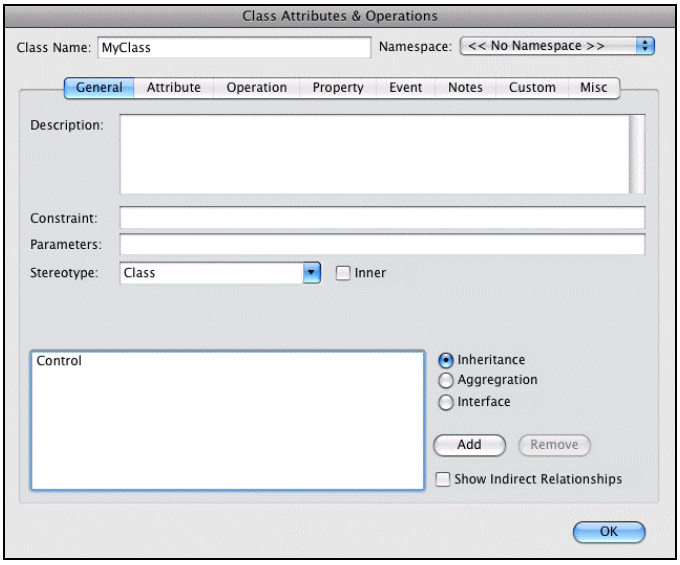
To understand the class modeling and code generation process, the next set of screens will demonstrate how to add a class object to the diagram, define REALbasic specific details and generate the source code. This discussion assumes that a project with a Dictionary and Class document has been created and the REALbasic language is selected.

Click the Class tool in the diagram to add a class object and present the Class Properties dialog. Type the name of the new class object and select the Class stereotype to indicate what type of REALbasic object it is. Click OK to dismiss the dialog and see the class object on the diagram.



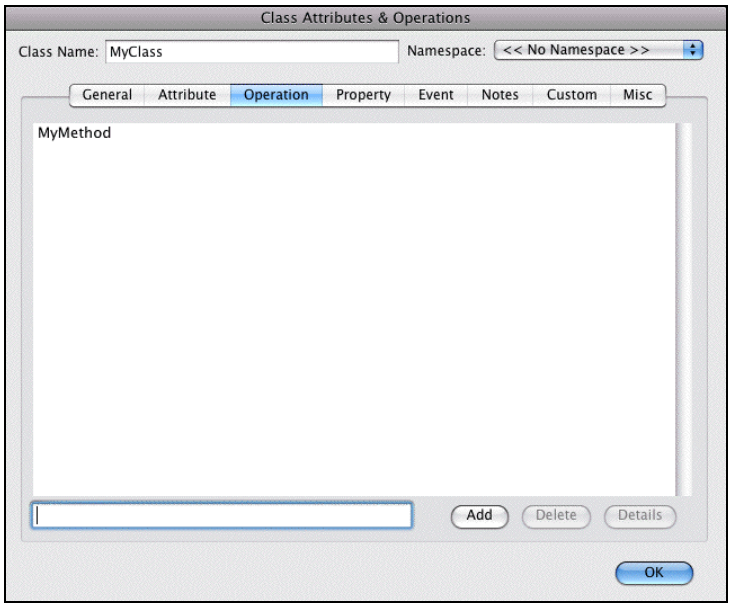
Name Class Object in Class Diagram

MyClass is a subclass of the Control class. The generalization relationship between MyClass and Control can be drawn with the Connection tool. To create an associated dictionary entry for the class object, click the Merge button in the tool bar. Now the Details button is enabled and when clicked the Class Attributes and Operations dialog is presented. This dialog shows the class name and any relationships with other classes. You can also add Inheritance, Aggregation and Interface relationships with other classes from the General panel.



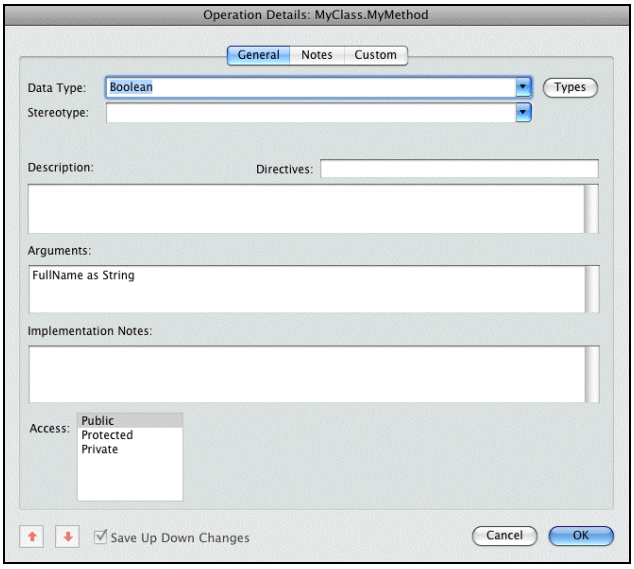
Class Attributes & Operations

Click the Operation panel to add a REALbasic method to the class. To add each method, type its name in the bottom edit field and click the Add button. As each method name is added to the list within the dialog, a corresponding dictionary entry is created in the Dictionary window.



Add Methods to a REALbasic Class

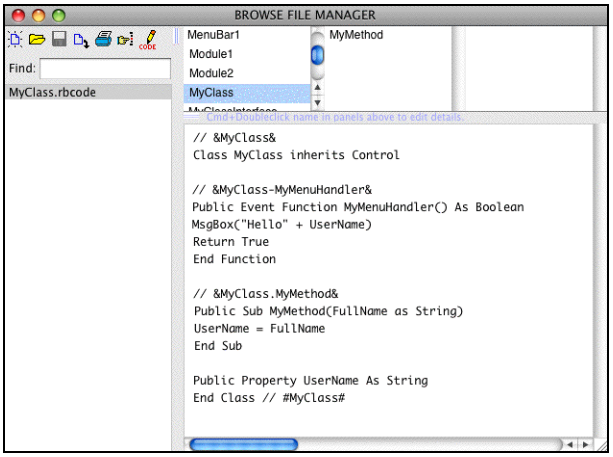
Double-click on a method name in the Operation panel to present the Operation Details dialog.



Operation Details

Use the Operation Details dialog to select the Data Type, enter the Arguments and set the Access type of the REALbasic method. The Notes panel of this dialog can be used to add programming notes or the actual source code within the method. During code generation, the Notes text can be inserted directly into the code.

The same process is used to define and detail attributes, properties and events that specify REALbasic programming constructs. When the design of a class object is complete, click the Generate Code button to immediately view the generated code in the integrated code browser.



Code Browser Showing Generated Source Code

Alternatively, if MacTranslator was used to capture the design of an existing REALbasic project, the class diagrams and dictionary details are automatically generated. The designer can select a class object and click the Code button to see the associated source code.

Map REALbasic to UML

Programming constructs in the REALbasic environment are represented as class objects and class members in class diagram. Stereotype names on class objects and class members add additional clues to the visualization and code generation process. For a new project, MacA&D offers to generate the needed Stereotype definition entries in the Dictionary when you select the REALbasic language, otherwise they get generated by MacTranslator for an existing project.

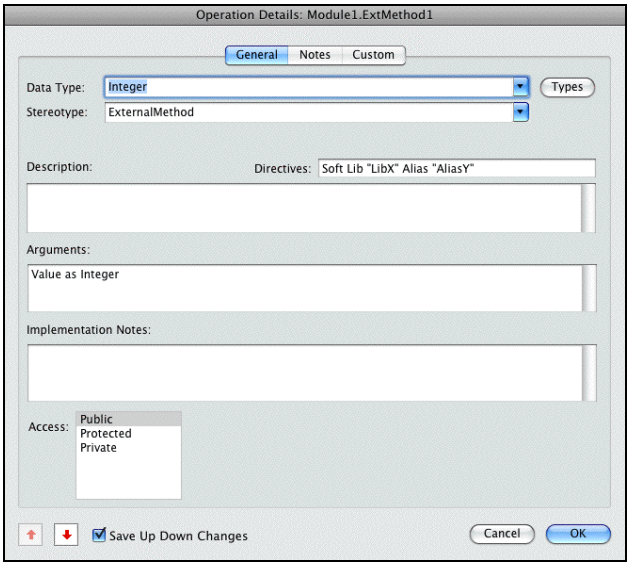
Modules, Windows, Classes, Class Interfaces and Menus in REALbasic are represented as class objects with the Module, Window, Class, Interface and Menu stereotype, respectively. If the REALbasic project references external code files they get represented as a class with the ExternalCode stereotype.

In the Class Properties dialog, select the Interface class type for class interfaces. For all other constructs, choose the Simple Class type. This information enables MacA&D to maintain the appropriate relationships between classes and interfaces within the dictionary and during code generation.

REALbasic properties are represented as a property class member. REALbasic menu handlers are represented as an event class member.

REALbasic methods are represented as an operation class member. A REALbasic delegate or external method is represented as an operation class member with the DelegateDeclaration or ExternalMethod stereotype, respectively. As you can see in a few specific cases, class members can also have a selected stereotype that further defines the kind of class member it is.

The Operation Details screen below shows the ExternalMethod stereotype selected for an external method, its data and arguments. For this special case, additional information is needed to define the library that contains the external method. That data is stored in the Directives field.



Operation Details for External Method

REALbasic Structures and Enums are represented as class attributes of the "Module" class in which they exist. The source code of a structure is shown below that contains two fields named fieldName1 and fieldName2. MacTranslator captures these two lines into the Notes section of the associated dictionary entry. Likewise the REALbasic code generator in MacA&D emits the Notes text to fill in the contents of the structure. The Enums construct works the same way.

```
Public Structure Structure1
fieldName1 As Integer
fieldName2 As Integer
End Structure
```

As indicated above, Menu objects in REALbasic are represented as a class with the Menu stereotype. Each menu item is represented as a class attribute.

A class diagram can use three kinds of relationships between class objects, inheritance, interface and aggregation. Relationships are shown on the diagram as connecting lines and represented in the dictionary with special characters in a class entry's composition field.

- ^ - Inheritance
- ! – Interface
- & - Aggregation

In REALbasic, if a class has a superclass, it inherits from the superclass. If a class implements a class interface, that is represented with the Interface relationship. If a class interface uses other class interfaces that gets represented by the Aggregation relationship.

When capturing the design of a REALbasic project with MacTranslator, notes and constants are captured into the Notes field of a class object. Likewise, during code generation, MacA&D emits the Notes text that may contain note comments and constants.

Summary

A REALbasic project can be represented as a UML class diagram. MacTranslator can capture the design from a REALbasic XML project for presentation within MacA&D. This automated process requires little human effort and can also generate a plain text source code file for each Window, Class, Module, Menu or Class Interface in the REALbasic project.

MacA&D is a complete UML modeling tool with language specific modeling and code generation for REALbasic and other programming languages. Collectively, a class diagram and associated dictionary has the information needed to generate REALbasic source code. Code is really just a textual representation of the design and can be emitted by clicking the Generate Code button for a selected class object.

MacA&D and MacTranslator can be used by developers to model and generate REALbasic code, capture the design of existing projects, understand the structure of a program or automate the translation process to or from other programming languages.