Visual Basic Upgrade Companion Version 8.0

Release Notes

Welcome to Version 8 of the VBUC. This major update includes improvements in three key areas:

1. Performance
2. Code generation
3. Coverage

Performance

- Because the VBUC uses an interative process of analysis in order to understand code, very large projects (> 2MLOC) can require lengthy migration processing times. Version 8 includes a new memory model that can handle extremely large applications better.
- Substantial performance improvements (10x) are now possible by converting multi-project solutions in parallel—up to 12 projects simultaneously with 8 cores and 32GB of RAM.
- Performance improvements were validated on our internal testbed of over 24MLOC of real-world customer code, representing 100s of VB6 projects.

Code generation

- Shared files bridge: In VB6 you can have a shared file with a declaration for Foo…then in different implementations Foo can be a method or a property, ie:

  ```
  Module1.Foo() 'in this module Foo is a method

  Module2.Globals.Foo 'In this module Foo is a property
  ```

  This caused .NET code generation to be clumsy. In VBUC 8.0, we now create a "shared files bridge" class so the reference is converted to `SharedFilesBridge.Foo` and each module gets its own implementation of Foo. Resulting code is:

  Project1: `return Module1.Foo()`
  Project2: `return Globals.Foo`
- Drag and drop: Some VB6 controls support drag and drop functionality. This code will now successfully migrate to .NET to both activate the behavior in the controls and create the event handlers in .NET.
- ByRef parameters: Function parameters in VB6 are passed as references by default (ByRef), rather than as copies. This creates potential problems since a function can change the value of an object which will persist after the function terminates. However, the conversion to C# requires lots of ugly transformations when the arguments sent were not really references, but other expressions such as function return values, literal expressions, constant references, operations, property access, etc. Previous versions of VBUC created sub-optimal C# code to safely pass arguments as values rather than references. Version 8.0 improves code generation by recognizing when parameters can be safely passed as values, creating more readable code.

- In VB6 MousePointer types were an enum, while in .NET they are Cursor class instances. These are now converted correctly to prevent compiler errors, while preserving intended behavior.
- Event renaming: The VBUC now applies the necessary transformation to event handler method names so that they are generated in a consistent way, preventing lots of compilation errors.

Coverage

- New library mappings: The following libraries are now supported to convert to similar .Net libraries: ActiveBar, ActiveInput, ActiveScroll, PvColorComboBox, PvExplorerLib, SQLDMO, StdPicker, TDBCombo, TLI.
- Improved mappings: The following existing library mappings have been improved to reduce compilation errors and enhance automation on conversion to equivalent .NET libraries: VB6 default libraries, Accusoft, CWGraph, CrViewer, FPSpread, MDIExtenderLib, MSComCtlLib, MSDataGridLib, MSFlexGrid, MSXML, SSActiveTreeView, SSActiveTabs, SSCalendarWidgets, SSDataWidgets, Threed, and more.
- Data Access helpers: Several behaviors have been improved in the DataAccess helpers to fix behavior differences, in areas such as: Data controls enhanced bindings, better schema loading and usage, filter behaviors, “FindNext” support, “MovePrevious” support, “AbsolutePage” and “AbsolutePosition” support, SQLTransactions improved, Timeout in commands and connections, DAO Clone support, Firing missing events, other big fixes.
- Reflection helper: some bugs were fixed in this helper class.

Additional improvements and bug fixes

- ComVisible feature improvements
- Interfaces Implementation: several bugs fixed
- Typing inference fixes with shared files
- ControlArrray events indexes issues fixed
- Get/Set properties to methods by ComInterop
- Date optional parameters supported
- Variables being used before the declaration
- Improved handling of variables declarations/initializations for both C# and VB.Net
- Identification and migration of Alias Types
- ADODB “Error” and “Errors” migration issues
- AddressOf migration for C# and VB.Net
- TimeZone and GetTimeZoneInformation structs and related functions conversion
- Several robustness fixes to avoid untranslated-statements
- Several assorted bug fixes

Clean up before remigrating

Note: before rerunning the VBUC on a project you have previously migrated with an older version of the tool, either generate the code in a different output directory or delete the existing output directory to avoid potential version conflicts with some helper classes.