COMPARATIVE GUIDE

The VB6 AI Migrator vs. the Visual Basic Upgrade Wizard

A guide to deterministic application modernization

Executive summary

For organizations relying on aging Visual Basic 6.0 (VB6) applications, selecting the right modernization path is a critical technical and business decision. Historically, many teams relied on the Visual Basic Upgrade Wizard (VBUW), a migration tool that shipped with Microsoft Visual Studio through its 2008 release. The VBUW has received no improvements since 2008, leaving modern enterprises without a viable path to current cloud architectures.

To solve this, Growth Acceleration Partners (GAP) and GAPVelocity AI offer the **VB6 AI Migrator** — a continuously upgraded, deterministic transformation platform. It generates native, idiomatic .NET code without proprietary runtimes, drastically reducing manual effort while preserving your proven business logic.

This document highlights the definitive feature and productivity differences between the legacy VBUW and the modern VB6 AI Migrator.

Step 1 — Assess before you migrate with ByteInsight

You can't deal with what you have until you know what you have. Before comparing migration tools, organizations must deeply understand their legacy codebase. **ByteInsight** is a free static code-analysis tool that builds a complete inventory of your source code so you can make an informed modernization decision.

- Walks your entire folder tree, reporting lines of code per file and identifying the specific technology footprint.
- Extracts metadata only — your source code never leaves your secure environment.
- Produces a data-driven inventory used to size the modernization effort and accurately scope your VB6 AI Migrator license.

High-level feature comparison

The table below summarizes the stark differences in capability between the modern VB6 AI Migrator and the retired VBUW.

Feature	VB6 AI Migrator	Legacy VBUW
C# code output	**Supported**	Not supported
VB.NET code output	**Supported**	Supported
Extensions supported	**7,000+**	~250
Type inference	**Supported**	Not supported
Modern data access	**EF Core / ADO.NET**	Legacy ADO wrappers
Structured error handling	**Try…Catch generation**	Legacy On Error GoTo
.NET native library support	**Supported**	Not supported
.NET enumerations	**Supported**	Not supported
Code refactoring & readability	**Comprehensive**	Minimal

Core capabilities of the VB6 AI Migrator

1. Extensibility and component mapping

Unlike the VBUW, which offers no extensibility, the VB6 AI Migrator can be customized to meet specific enterprise needs. It automatically migrates programming patterns and legacy ActiveX controls to their native .NET equivalents — or to newer third-party components such as Infragistics or ComponentOne — saving immense manual effort, time and money.

2. Intelligent type inference

An AI-based type-inference engine analyzes variables, parameters and return values. Instead of defaulting to generic types like `Object` or `Variant` (as the VBUW does), it declares variables with the exact, appropriate type — avoiding unnecessary compilation errors and warnings.

3. Modern data access (ADO to ADO.NET / EF Core)

The legacy VBUW generates a target application that still uses outdated ADO via COM Interop wrapper calls. The VB6 AI Migrator fundamentally upgrades the data-access model to ADO.NET and Entity Framework Core, delivering:

- **Performance and scalability:** disconnected datasets and automatic connection pooling remove database bottlenecks.
- **Interoperability:** data is transported in standard XML formats.

4. Direct C# generation

Because some VB6 features have no native C# equivalent (e.g. `Optional` parameters, `Modules`, `ReDim`), the legacy VBUW could not output C#. The VB6 AI Migrator performs sophisticated automated transformations to generate clean, native C# directly from VB6 source.

5. Structured error handling and native libraries

- **Error handling:** the VBUW rigidly maintains the outdated `On Error GoTo` pattern; the VB6 AI Migrator recognizes these and cleanly replaces them with modern .NET `Try…Catch` blocks.
- **Native libraries:** instead of relying on VB Compatibility Libraries, it maps legacy functions to native .NET classes — e.g. `Len` and `InStr` become `.Length` and `.IndexOf` — improving readability and performance.

6. Readability and automated refactoring

The VB6 AI Migrator executes advanced structural refactoring to produce code that looks hand-written:

- Upgrades long `If…ElseIf` constructs to `switch` / `Select Case` statements.
- Converts iterative loops into clean `For…Each` blocks.
- Replaces generic numeric literals with clearly defined .NET enumerations.
- Consolidates nested `If` statements using short-circuit operators (`AndAlso`, `OrElse`).

Step 2 — Beyond VB6: full-scale modernization with VELO

While the VB6 AI Migrator handles deterministic legacy transformations, GAPVelocity AI also offers **VELO**, an agentic AI modernization platform built on Microsoft Foundry. For sprawling enterprise ecosystems — including Access, PowerBuilder or WinForms — VELO orchestrates a multi-agent pipeline:

- **Scout Agent:** scans the application to map dependencies and complexities.
- **Architect Agent:** proposes optimal target architectures, such as Blazor and Azure SQL.
- **Translation & Quality Agents:** convert legacy logic to modern C# while generating automated tests and verifying functional parity.

Begin your modernization journey

Stop managing legacy risk and start powering your growth. With the VB6 AI Migrator, you achieve the speed of AI with the quality of human engineering.

- **Try ByteInsight:** run our free code-analysis tool to safely inventory your legacy codebase.
- **Schedule a consultation:** speak with a GAP migration engineer for a data-driven path to the cloud.

gapvelocity.ai