

Press Release

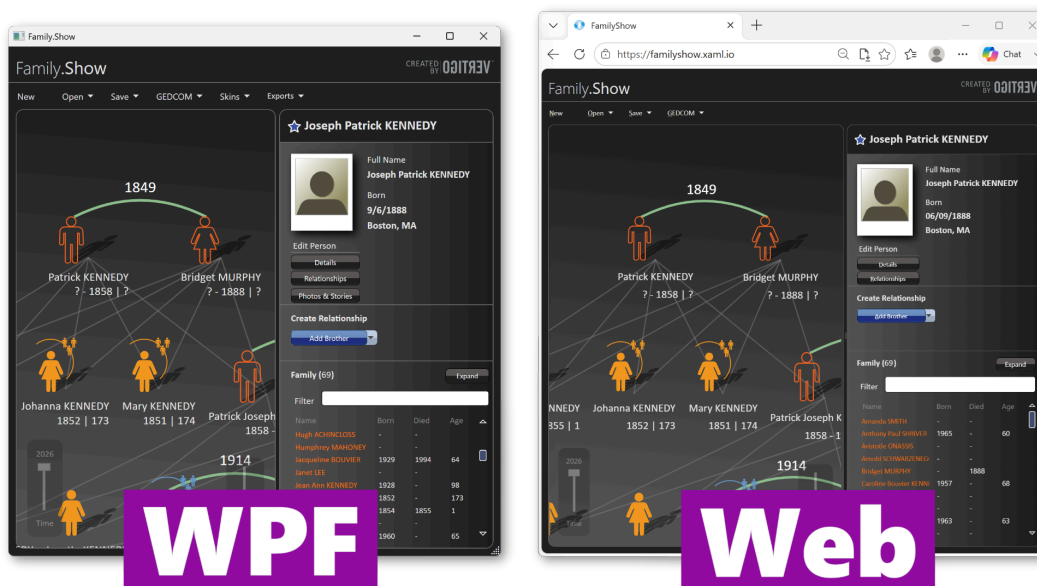
EMBARGOED UNTIL MONDAY, JULY 6, 2026 (8:00 a.m. ET)

XAML.io adds "Migrate from WPF": run an existing WPF application on the web

The free web-based .NET IDE analyzes WPF apps for web compatibility and imports them to run in any browser via the open-source OpenSilver framework: cross-platform, web-native, with minimal code changes

Paris, France — July 6, 2026. Userware today released version 0.8 of XAML.io, the free web-based .NET IDE, adding **Migrate from WPF**: free, in-browser tools that bring an existing Windows Presentation Foundation (WPF) application to the web. Developers can analyze a compiled WPF app for web compatibility, import a WPF project to run it in the browser, or engage Userware for an end-to-end migration. The technology is powered by OpenSilver, Userware's open-source framework that runs WPF-style C# and XAML on the web by compiling to WebAssembly and rendering XAML as real HTML DOM elements.

To show how little of an application typically changes, Userware migrated **Family.Show**, the advanced WPF reference application Vertigo originally built for Microsoft soon after WPF launched. The result runs live in the browser, with 97% of the original code unchanged, a figure measured as a line-by-line diff of the C# and XAML files between the original and migrated codebases:



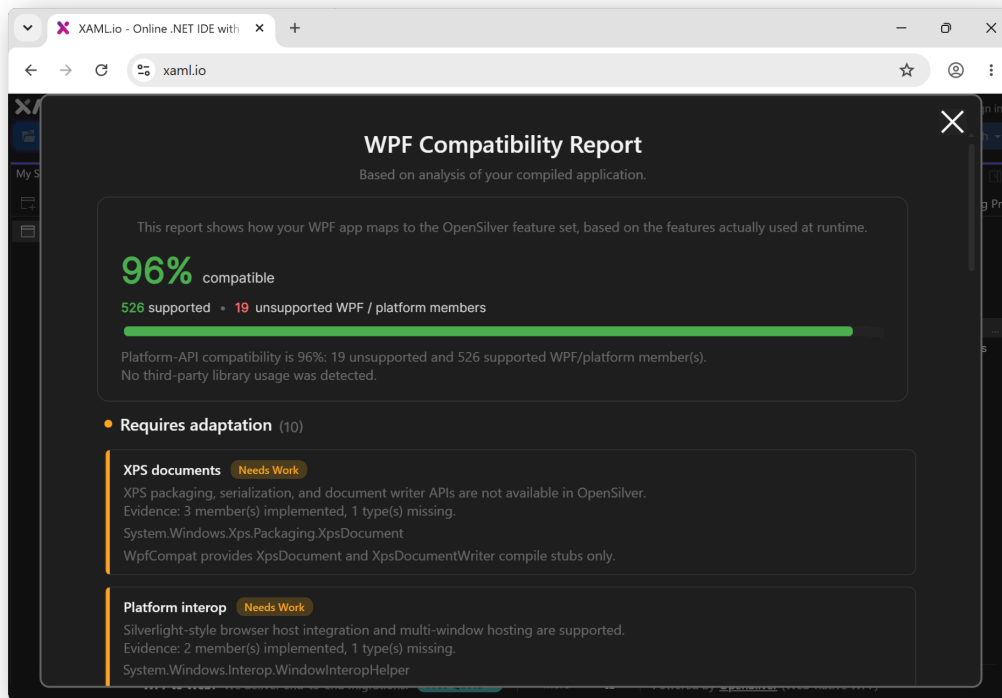
Side-by-side screenshots of Family.Show, an advanced WPF reference app, running as a native Windows desktop app (left) and as a web app in the browser via XAML.io/OpenSilver (right). 97% of the original code is unchanged.

WPF is a mature, powerful framework for Windows desktop development, and it isn't going anywhere. What pushes teams to put these applications on the web is how they get delivered, secured, and accessed. Today a WPF application has to be installed and kept up to date on every user's machine. On the web, that per-machine install disappears: IT deploys once and ships updates to everyone in a single click, so no one is left running an outdated version.

Security changes too. The app runs inside the browser's sandbox, the same model organizations already trust for every website they use. Nothing is installed on the endpoint, and the app has no standing access to the user's machine.

Reach changes as well. Users can open the app from any device with a browser (desktop, tablet, or phone), and the same C#/XAML codebase can also ship as desktop and mobile apps. And because OpenSilver renders the interface as standard HTML rather than painting it onto a canvas, migrated apps gain browser-native accessibility (a foundation for standards such as Section 508 and EN 301 549), text search, and integration with the rest of the web.

XAML.io's new migration tools are built to deliver those benefits without a rewrite, and the work begins with analysis. The new free **Compatibility Analyzer** works for an application of any size. It reads the compiled application, identifies the WPF and platform APIs the app uses, and reports, feature by feature, what runs on OpenSilver today, what needs adaptation, and what is blocking, with an exportable report. The app's binaries are processed locally in the browser; only an aggregated list of unsupported features is sent to Userware's server to generate the report.



Screenshot of a WPF compatibility report in XAML.io, showing the percentage of platform APIs supported on OpenSilver for a given WPF app, and a feature-by-feature breakdown of what works as-is and what needs adaptation.

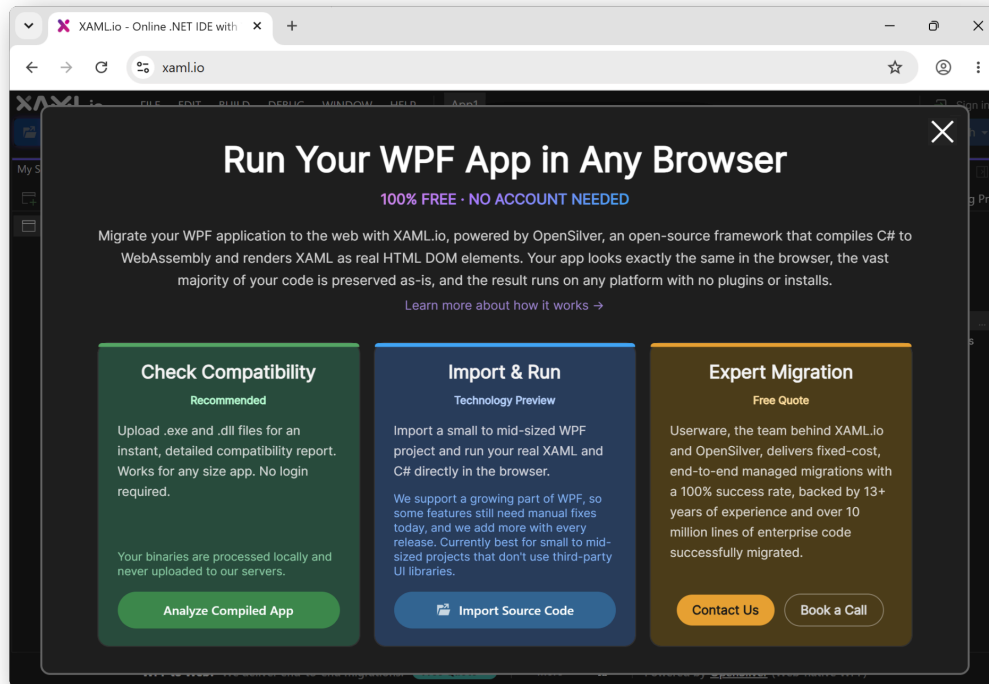
The migration approach is deliberately not an AI rewrite. Rather than regenerating code, XAML.io preserves the original C# and XAML and makes the framework support it, applying only small, visible changes a developer can review: unsupported C# is wrapped in compiler directives, unsupported XAML is annotated with comments, and every automatic edit is surfaced in the IDE as a warning. A runtime alert system goes further, flagging stubbed or disabled code as the application runs, with the exact file and line, and without crashing it. The compatibility layer behind the tooling currently ships 8 Roslyn analyzers and 16 automatic code fixes, and its WPF coverage keeps expanding.

The app also looks the same after it moves to the web. On import, XAML.io automatically applies a WPF look-alike theme so the web version matches the desktop original; teams that find the classic WPF look dated can switch to a modern theme and add responsive layouts to refresh the interface, without rewriting the application.

The full Family.Show migration is published, so anyone can check the claim instead of taking an adjective for it: compare the [live migrated app](#), its [source on XAML.io](#), and the [original WPF source](#) side by side. The migrated version is also relatively light for what it is: the running app, including the entire .NET runtime, is 8.1 MB compressed, and because OpenSilver needs no browser plugin, there is nothing for the user to install. Served from a CDN, it loads in a few seconds on a good connection and is then cached for subsequent visits; runtime performance for an app of this kind is solid. A short video of the end-to-end migration is in the launch write-up.

"WPF isn't going anywhere on the desktop. But a lot of these apps now need to live on the web too. That means reachable from any device, centrally updated, and running inside the browser's security sandbox instead of with full access to each user's machine, with the accessibility and auditability the desktop never gave them," said Giovanni Albani, CEO of Userware. "What's always made that painful is that it meant a rewrite, and that's exactly what we remove. We keep your original C# and XAML and make the framework support what your app already does, so you avoid regressions and keep a codebase your team still understands."

"For a serious application, the honest answer is that the migration is a project, and that's exactly what we do," said Darshin Vyas, VP of Sales at Userware. "It starts with a free compatibility analysis and a fixed-cost quote. We deliver in milestones you can compile and test, and your team can keep working the whole time. We've done this for more than thirteen years, on over ten million lines of enterprise code."



Screenshot of XAML.io's "Migrate from WPF" window, offering three paths: analyze a compiled WPF app for web compatibility, import and run a project in the browser, or request an end-to-end migration.

The self-serve tools are completely free, with no time limit or feature paywall. The Compatibility Analyzer is robust for apps of any size, while in-browser source import is currently a Technology Preview best suited to small to mid-size, self-contained projects. For production migrations of any size, Userware offers optional professional services and support, and customers can sponsor the specific WPF features they need, which then ship to everyone in the open-source framework. That model is what funds the free tooling. Third-party UI control suites are not yet supported in the self-serve tools and are handled within managed migrations. Userware's roadmap includes broadening WPF coverage, one-click web publishing, mobile and signed desktop publishing, and VB.NET support.

XAML.io is free to use and runs in any modern browser. The entire migration workflow, from analysis through importing, fixing, running, and exporting a Visual Studio solution, can run locally in the browser with no signup, and source code is not sent to Userware's servers; cloud save, sharing, and AI features are opt-in. This keeps proprietary code on the developer's machine, which matters for regulated industries. There is also no lock-in: at any point, developers can download a standard Visual Studio solution and continue in Visual Studio, VS Code, Rider, or another IDE.

OpenSilver, the framework behind the migration, is an open-source, web-native implementation of WPF: it runs WPF-style C# and XAML on WebAssembly and the standards-based web, with no plugin and nothing to install. It first emerged in the Silverlight era, but Userware has long since refocused it on WPF, the much larger framework of which Silverlight was only ever a subset.

Availability: XAML.io v0.8 is available now at <https://xaml.io>
Full technical write-up: <https://blog.xaml.io/post/migrate-wpf-to-the-web/>
Family.Show live demo: <https://familyshow.xaml.io>
Family.Show source code: <https://xaml.io/s/Samples/Source/FamilyShow>
Free migration assessment: <https://opensilver.net/talk-to-expert/>

About Userware

Userware is the company behind OpenSilver, the open-source framework that runs C# and XAML in the browser via WebAssembly, and XAML.io, the free web-based .NET IDE built on it. For more than 13 years, Userware has migrated enterprise XAML applications (Silverlight, LightSwitch, and WPF) to modern platforms for organizations including DENSO, Tata Communications, Symcor, LiveData (healthcare), Repton (education), and many more across industries such as financial services, manufacturing, and the public sector. Userware is based in Paris, France.

Media contact

Vasil Buraliev
Media Relations, Userware
vasil.buraliev@userware.dev

High-resolution screenshots and logos are available now in the press kit at <https://blog.xaml.io/post/press-kit/> (alongside the press release, fact sheet, and quotes), and may be used freely for editorial coverage. More screenshots and a short migration video are in the technical write-up, which goes live July 6 at <https://blog.xaml.io/post/migrate-wpf-to-the-web/>

Additional quotes (for editors, optional)

Offered so different outlets can choose the angle that fits their story. All attributable to Giovanni Albani, CEO of Userware.

- *On why move a desktop app to the web at all:* "Most teams aren't moving these apps to the web for novelty. They're moving them because the business needs the app on any device, behind central one-click updates, inside the browser's security sandbox rather than with full access to each PC, with the accessibility and auditability the Windows desktop never gave them."
- *On the AI-rewrite trend:* "The instinct today is to let AI rewrite a legacy app. For a working business app, that's a brand-new app you have to re-test and re-trust. We'd rather keep your code and change where it runs."
- *On the value of existing code:* "A working WPF app is an asset, not a liability. We help companies bring it to the web and keep it, instead of throwing it away in a rewrite."
- *On what has changed:* "For years, the only way to get a WPF app onto the web was to rewrite it in something else. That's the assumption we're ending."
- *Short pull-quote:* "The web shouldn't cost you a rewrite: keep your WPF code, and change where it runs."