

PRESS RELEASE



OpenSilver 3.3: Blazor Components Now Run Directly Inside XAML Applications

Open-source framework combines XAML's layout and binding system with Blazor's component ecosystem in a single runtime

Highlights:

- Use any Blazor component library directly in XAML applications
- Modernize WPF and Silverlight apps incrementally, one control at a time
- Full support for .NET 10, C# 14, and Visual Studio 2026
- Live samples with DevExpress, Syncfusion, MudBlazor, Radzen, and Blazorise at [OpenSilverShowcase.com](https://www.opensilver.com/showcase)
- Already in production at Cegid with DevExpress RichEdit integration

Paris, France – January 27, 2026 – Userware today released OpenSilver 3.3, introducing native integration between XAML and Blazor. Developers can now embed Blazor components from libraries like DevExpress, Syncfusion, MudBlazor, Radzen, and Blazorise directly inside XAML applications, with no JavaScript bridges or performance overhead.

OpenSilver is an open-source framework that runs WPF-style C# and XAML applications in web browsers via WebAssembly. It supports deployment to iOS, Android, Windows, macOS, and Linux through .NET MAUI Hybrid. With version 3.3, developers can also leverage the Blazor component ecosystem while keeping XAML as their primary UI technology.

"Blazor has an incredible component ecosystem. XAML has a powerful layout and binding system that developers love," said Giovanni Albani, CEO of Userware. "With 3.3, you don't have to choose. Use XAML where it excels, drop in Blazor components where you need them. Your ViewModels and architecture stay the same."

How It Works

Because OpenSilver renders UI elements to the HTML DOM (just like Blazor), both technologies share the same rendering model. Developers have two options for integration:

- **Option1: Inline Razor code** - Write Razor markup directly inside XAML files within <RazorComponent> tags. XAML markup extensions like {Binding} and {StaticResource} work directly in the embedded Razor code.

Here's an example of inline Razor with XAML data binding:

```
<UserControl
  x:Class="DemoApp.MainPage"
  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:razor="clr-namespace:OpenSilver.Blazor;assembly=OpenSilver.Blazor">

  <!-- XAML CODE -->
  <Grid RowDefinitions="Auto,*">

    <TextBlock Text="Hello from XAML"/>

    <Border Grid.Row="1">
      <StackPanel>

        <!-- INLINE RAZOR CODE -->
        <razor:RazorComponent>
          @using Radzen
          @using Radzen.Blazor
          <RadzenButton Text="This is a Blazor button."
            Click="{Binding OnClick, Type=Action}" />
        </razor:RazorComponent>

        <!-- DEVEXPRESS BLAZOR GRID -->
        <razor:RazorComponent>
          @using DevExpress.Blazor
          <DxGrid
            Data="{Binding Employees,
              Type=IQueryable<DemoApp.Model.Employee>}"
            PageSize="15">
```

```

        <Columns>
            <DxGridDataColumn FieldName="FirstName"
                              Caption="First Name"/>
            <DxGridDataColumn FieldName="LastName"
                              Caption="Last Name"/>
            <DxGridDataColumn FieldName="BirthDate"
                              Caption="Birth Date"
                              DisplayFormat="{0:d}"/>
        </Columns>
    </DxGrid>
    </razor:RazorComponent>
</StackPanel>
</Border>
</Grid>
</UserControl>

```

Figure 1 - Inline Razor code with XAML binding to ViewModel

- **Option 2: Reference .razor files** - Create standard Blazor components in .razor files and reference them from XAML using:

```
<RazorComponent ComponentType="{x:Type local:MyBlazorComponent}"/>
```

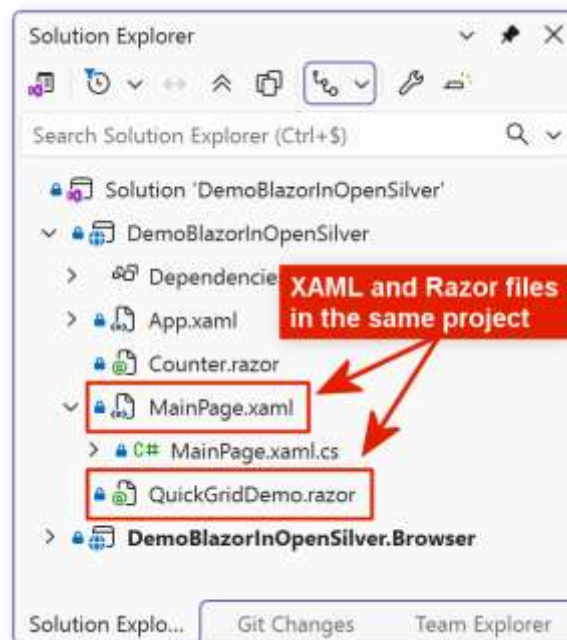


Figure 2 - Visual Studio Solution Explorer showing XAML and Razor files in the same OpenSilver project

Access to the Blazor Ecosystem

OpenSilver 3.3 provides immediate access to popular Blazor component libraries including DevExpress, Syncfusion, MudBlazor, Radzen, and Blazorise. No special builds or glue code required: add the NuGet package, reference the component, and use it in XAML.

"We're pleased to see OpenSilver enabling new scenarios for DevExpress Blazor components," said Alexander Chuev, Blazor Team Product Manager at DevExpress. "Developers now have more options for building modern applications with familiar tools."

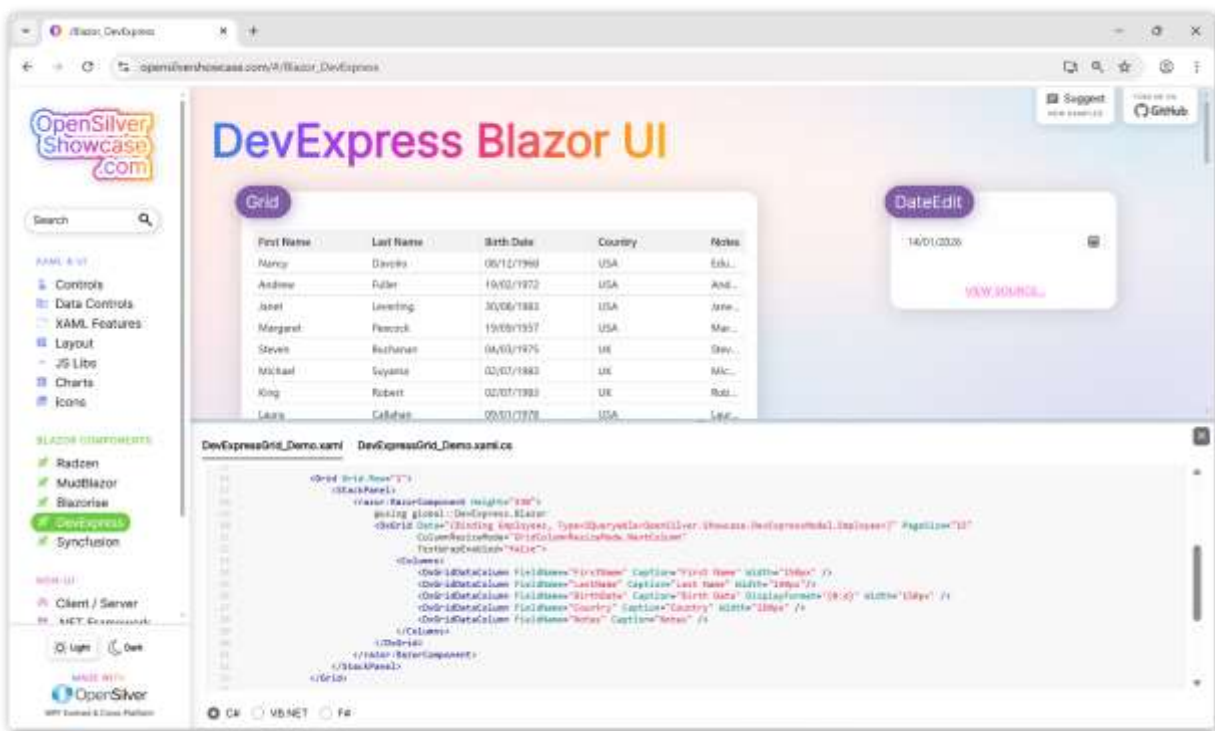


Figure 3 - OpenSilverShowcase.com displaying DevExpress Blazor components in a XAML application, with source code

A Modernization Path

For enterprises maintaining WPF or Silverlight applications, Blazor integration offers a practical modernization approach. Rather than rewriting entire applications, teams can

replace individual controls with modern Blazor equivalents while keeping their existing architecture intact.

The capability is already being used in production. Cegid, a European business software company, recently integrated the DevExpress RichEdit Blazor component into their Tax Flex application to add Word document editing capabilities. Tax Flex is a large XAML codebase that was migrated from Silverlight to OpenSilver.

"Adding the RichEdit component was straightforward," said Pascal Olivier, R&D Director at Cegid's Finance Innovation Factory. *"We added the NuGet package, placed the component in our XAML, connected it to our existing ViewModel, and it worked."*

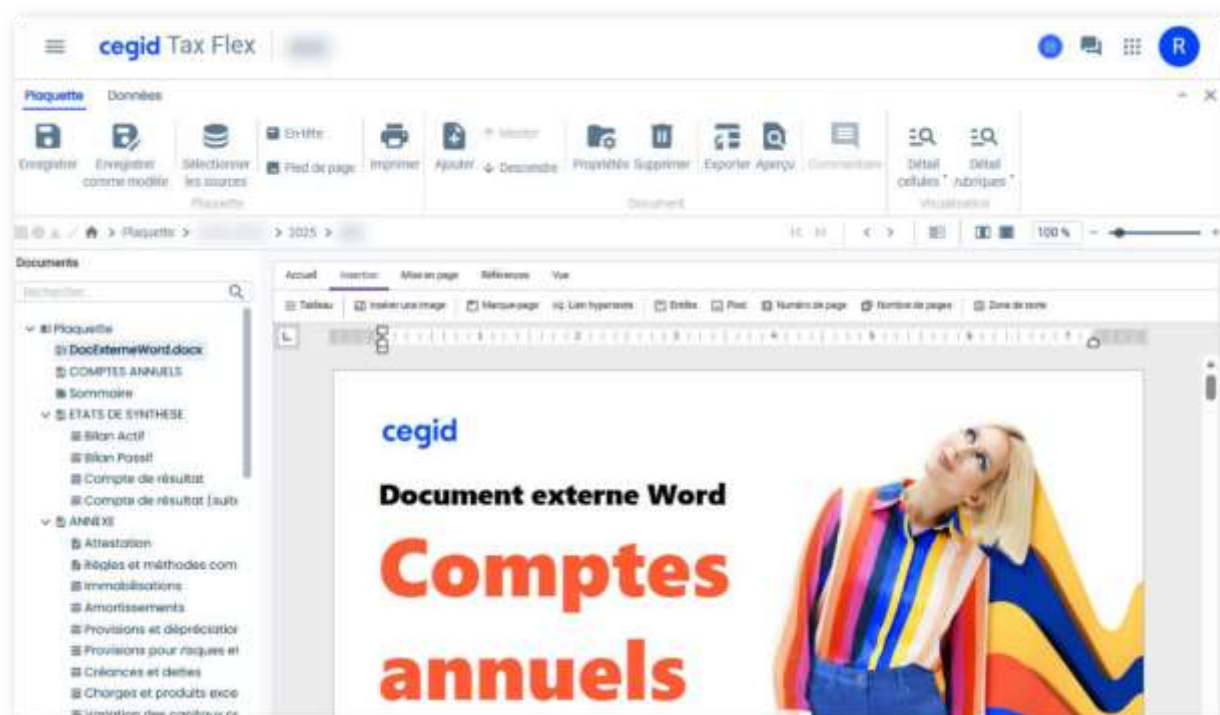


Figure 4 - DevExpress RichEdit Blazor component integrated in a XAML application (Cegid Tax Flex)

Also in This Release

OpenSilver 3.3 includes significant updates beyond Blazor integration:

- **Platform support:** Full compatibility with .NET 10 (released November 2025), C# 14, and Visual Studio 2026.

- **Responsive layouts:** A new Responsive markup extension enables adaptive Uis in XAML:

```
<Border Width="{Responsive Mobile=200, Desktop=500}"  
        Visibility="{Responsive Mobile=Collapsed, Desktop=Visible}"/>
```

- **WPF compatibility:** Complete CommandManager and RoutedCommand infrastructure, template selectors, preview/tunneling events, and numerous API additions including WeakEventManager, DependencyPropertyDescriptor, and a rewritten CollectionViewSource.
- **Modern layout properties:** Grid and StackPanel now support BorderThickness, BorderBrush, Spacing, Padding, and CornerRadius, along with MAUI-style shorthand syntax for row and column definitions.

Try It

Live samples demonstrating Blazor component integration are available in the OpenSilver Showcase, with source code for each example:

- **Web:** [OpenSilverShowcase.com](https://opensilvershowcase.com)
- **iOS:** [App Store](#)
- **Android:** [Google Play](#)

Availability

OpenSilver 3.3 is available now on NuGet.org, in the [Visual Studio](#) and [VS Code](#) marketplaces (with XAML designer and project templates), and online at [XAML.io](#) for browser-based development without installation.

Blazor integration documentation and limitations:

doc.opensilver.net/documentation/general/opensilver-blazor.html

About OpenSilver



OpenSilver is an open-source implementation of a WPF-compatible subset for the web. Using C# and XAML, developers can build applications that run in any modern browser via WebAssembly and deploy to mobile and desktop platforms through .NET MAUI Hybrid.

The framework is used in production by enterprises including government agencies, healthcare organizations, and Fortune 500 companies to modernize WPF, Silverlight, and LightSwitch applications, and to build new cross-platform business applications.

Website: opensilver.net

About Userware



Founded in 2007, Userware specializes in .NET application modernization. The company maintains OpenSilver as an open-source project and provides migration services for organizations bringing WPF, Silverlight, and LightSwitch applications to modern platforms. Userware also develops [XAML.io](https://xaml.io), an online development environment where developers can build and run C# and XAML applications directly in the browser using a drag-and-drop XAML designer.

Website: userware.dev

Migration services: opensilver.net/contact

Press Contact

Vasil Buraliev

Media Relations at Userware

Email: vasil.buraliev@userware.dev