# FeatureFlags for Avalonia

FeatureFlags for Avalonia is a lightweight library that allows you to enable or disable UI components dynamically based on feature flags. It provides a simple and intuitive way to control the visibility and activity of controls directly from your AXAML files using attached properties.

## Getting Started

Installation

Add the library to your Avalonia project via NuGet:

`Install-Package FeatureFlags.Avalonia`

Usage

1. Define Feature Flags

Define your feature flags in your Program.cs:

```
public static AppBuilder BuildAvaloniaApp()
=> AppBuilder.Configure<App>()
    .UsePlatformDetect()
    .WithInterFont()
    .LogToTrace()

    .UseIniForFeatureFlags("FeatureFlags.ini")
    .UseJsonForFeatureFlags("FeatureFlags.json")
    .UseDicionaryForFeatureFlags(new Dictionary<string, bool>
    {
        { "Feature1", true },
        { "Feature2", false }
    })
;
```

2. Use in AXAML

Add the FeatureFlags namespace to your AXAML file and use the attached properties:

```
<Window xmlns="https://github.com/avaloniaui"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:local="using:YourNamespace">
    <StackPanel>
        <Button Content="Visible if FeatureA is Enabled"
                local:FeatureFlags.IsVisible="FeatureA" />
        <Button Content="Active if FeatureB is Enabled"
                local:FeatureFlags.IsActive="FeatureB" />
    </StackPanel>
</Window>
```

IsVisible: Controls the visibility of a control based on a feature flag.

IsActive: Enables or disables a control based on a feature flag.

FeatureManager Methods:

`IsFeatureEnabled(string featureName)`: Checks if a feature flag is enabled.

---

Happy coding! 🚀