

Du hast versucht, ein Windows Forms-Projekt nach WPF umzuwandeln – das ist möglich, aber nicht per Mausklick. Das erfordert klare Schritte, vor allem, wenn es bereits viele System.Windows.Forms-Abhängigkeiten gibt.

Deine Fehler zeigen, dass du derzeit eine Mischung aus WinForms und WPF-Namespace im Projekt hast, was zu Namenskonflikten, falscher Vererbung und fehlenden Methoden führt.

 **Ziel:** Von Windows Forms zu echtem WPF-Projekt umstellen

 **1. Alte WinForms-Struktur entfernen**

 **Entferne oder ersetze:**

- **MainForm.cs, MainForm.Designer.cs, MainForm.resx**
 - Alle Steuerelemente mit **System.Windows.Forms.***
 - Verweise auf **System.Windows.Forms** (über using oder im Designer)
 - Eigenschaft **<UseWindowsForms>true</UseWindowsForms>** aus der .csproj
-

 **2. .csproj für WPF anpassen**

Beispiel:

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net9.0-windows</TargetFramework>
    <UseWPF>true</UseWPF>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="OpenTK" Version="4.9.4" />
  </ItemGroup>
```

```
<PackageReference Include="OpenTK.WinForms" Version="4.0.0-pre.6" />  
<PackageReference Include="geometry3Sharp" Version="1.0.324" />  
</ItemGroup>  
  
</Project>
```

⚠️ OpenTK.WinForms ist nicht für WPF gedacht – verwende alternativ eine reine OpenGL-Integration für WPF oder trenne das Rendering in ein WinForms-Host-Control, wenn es unbedingt nötig ist.

3. Neue WPF-Hauptfenster-Struktur erstellen

Ersetze MainForm.cs durch MainWindow.xaml und MainWindow.xaml.cs:

MainWindow.xaml

```
<Window x:Class="ManufacturingTool.MainWindow"  
       xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  
       xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  
       Title="MainWindow" Height="450" Width="800">  
  
<Grid>  
    <!-- Dein WPF-Layout hier -->  
</Grid>  
</Window>
```

MainWindow.xaml.cs

```
using System.Windows;  
  
namespace ManufacturingTool  
{  
    public partial class MainWindow : Window  
    {  
        public MainWindow()  
        {
```

```

        InitializeComponent();
    }

}

}

```

4. Fehlermeldungen korrigieren (Beispiele)

Fehler	Ursache	Lösung
Dictionary< TKey >	falsche Syntax	Verwende Dictionary< TKey, TValue >
Directory nicht gefunden	Schreibfehler	Wahrscheinlich meinst du System.IO.Directory
"Draw()" wurde nicht überschrieben	Basis-Klasse verlangt override	Prüfe Signatur & Rückgabetyp, schreibe korrekt override Draw()
mehrdeutiger Verweis auf "Button"	Mischung von WPF und WinForms	Entferne alle using System.Windows.Forms; oder wähle explizit System.Windows.Controls.Button

5. Aufräumen: Feature.cs-Fehler

Die Fehler wie

Feature.Drwaw()

zeigen Schreibfehler (vermutlich sollte es Draw() heißen). Kontrolliere alle Basisklassen und Methoden auf:

- korrekte Signatur
- override-Angabe
- ob die Basisklasse abstract ist

Beispiel:

```

public abstract class Feature
{
    public abstract void Draw();

    public abstract void SetProperties(Directory dir); // Directory wahrscheinlich falsch!

```

}

→ Directory ist kein Typ, du meinst vermutlich:

```
public abstract void SetProperties(string directoryPath);
```

Fazit: Checkliste zur Umstellung

1.  Entferne alle .Designer.cs, .resx, Forms-Referenzen
 2.  Erstelle MainWindow.xaml + .cs neu
 3.  Passe .csproj an (<UseWPF>true</UseWPF>)
 4.  Korrigiere Methodensignaturen (kein Drwaw, sondern Draw)
 5.  Keine Namespace-Dopplungen (z. B. Button oder Panel)
 6.  System.Windows.Forms → raus; System.Windows.Controls → rein
-

Wenn du magst, kannst du mir deinen aktuellen MainForm.Designer.cs zeigen, und ich helfe dir, ihn in XAML zu übersetzen – damit hast du eine funktionierende WPF-Basis.