Endlich sichtbar

.NET Core zeigt sein Gesicht: Version 3 spendiert zwei grafische Oberflächentechnologien.

Zwar gilt das nicht unter macOS und Linux, aber Windows-Desktop-Entwickler können frohlocken. Mit Windows Forms und Windows Presentation Foundation stehen nun zwei Technologien zur Verfügung, mit denen sich grafische Oberflächen bauen lassen. Zeit wird's!

Während mit den ersten Versionen des modularen Frameworks nur Programme für die Kommandozeile geschrieben werden konnten, klappt das mit der Preview von .NET Core 3.0 auch für grafische Oberflächen [1].

Zusätzliche Templates helfen dabei.

Mit dem Befehl „dotnet new winforms" wird beispielsweise ein neues WinForms-Projekt angelegt.

Entsprechend lautet der Befehl für WPF *dotnet new wpf*. Das angelegte Projekt enthält im Fall von Windows Forms eine Datei *Form1.Designer.cs*, die die Oberfläche des Hauptfensters definiert – wie gewohnt in Form von Codezeilen. Legen Sie hingegen ein WPF-Projekt an, finden Sie im Projektordner eine Datei *MainWindow.xaml*, die das Fenster in der Sprache XAML beschreibt.

Bislang konnte eine .NET-Core-Anwendung nicht einfach durch einen Doppelklick auf die entsprechende EXE-Datei gestartet werden. Der Grund: Es gab schlicht keine EXE-Datei. Stattdessen mussten Sie den Befehl *dotnet myapp.dll* bemühen, wobei *myapp.dll* die Assembly Ihrer Anwendung ist.

Auch das hat sich mit .NET Core 3.0 geändert. Wer sich die Preview herunterlädt und damit ein Projekt übersetzt, findet im *bin*-Ordner eine EXE-Datei, die den Namen des Projekts trägt. Doppelklick … Sie kennen den Rest.

Mit einer Umfrage [2] sucht Microsoft den Dialog mit den Entwicklern, um nach eigener Aussage zu erreichen, dass möglichst viele Anwendungen auf .NET Core umgezogen werden können. Auf der Website hebt das Team explizit noch einmal die Vorzüge von .NET Core hervor, wie zum Beispiel zusätzliche Features wie *Span<T>* oder die höhere Performance.

Ein erster Versuch mit der Preview ließ mein Herz hüpfen: Eine bestehende WPF-Anwendung mit XML-Serialisierung, Dateioperationen, Komponenten und Drag-and-drop läuft ohne Veränderung (!) des Codes unter .NET Core. Lediglich die Projektdatei musste leicht angepasst werden. Einfach nur klasse.

Viel Spaß mit der dotnetpro!

Tilman Börner
Chefredakteur dotnetpro

[1] www.dotnetpro.de/SL1903Edi1
[2] www.dotnetpro.de/SL1903Edi2

Andreas Maslo

hat zusammengesucht, was es an Konfigurationsformaten so gibt (S. 10)

Dr. Holger Schwichtenberg

klärt auf, worin der Unterschied zwischen Razor Components und Blazor liegt (S. 46)

Patrick A. Lorenz

gibt einen Einblick, wie der Entwicklungsprozess in seinem Unternehmen aussieht (S. 96)