
Rafał Klinowski
WPF
i Material Design
Od podstaw do tworzenia praktycznych aplikacji
Helion

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite/Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Skład komputerowy w systemie LaTeX wykonał autor.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/wpfmat
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-289-2398-0

Printed in Poland.

Spis treści

Rozdział 1

Wstęp do WPF

1.1. Czym jest WPF?

WPF jest frameworkiem do tworzenia aplikacji desktopowych (lub „okienkowych", to znaczy takich, które są bezpośrednio uruchamiane w systemie operacyjnym komputerów osobistych) stworzonym i rozwijanym przez firmę Microsoft. Po raz pierwszy pojawił się na rynku w 2006 roku jako następca WinForms, dość popularnej w tamtych czasach technologii desktopowej. WPF wyróżniał się m.in. tym, że oferował wsparcie dla grafiki wektorowej, stylów, animacji oraz renderowania elementów interfejsu przy pomocy DirectX.

Dzięki pełnej integracji z XAML (eXtensible Application Markup Language) WPF umożliwiał oddzielenie logiki biznesowej aplikacji od warstwy wizualnej. Takie podejście nie tylko ułatwiało pracę zespołów projektowych, ale także pozwalało na dynamiczne tworzenie i modyfikowanie interfejsu użytkownika, co było dużym krokiem naprzód w porównaniu do poprzednich technologii. WPF stał się także podstawą dla bardziej zaawansowanych koncepcji, które zostaną poruszone w dalszej części książki, takich jak wzorzec MVVM (Model-View-ViewModel), który pozwala na lepsze organizowanie kodu aplikacji.

Należy podkreślić, że **WPF nie jest technologią wieloplatformową, a aplikacje tworzone za jego pomocą można uruchomić tylko na systemie Windows**.

Od 2018 roku WPF stał się projektem open-source, co oznacza, że jego kod źródłowy jest dostępny publicznie w repozytorium na platformie Git-

Hub[1]. Było to częścią strategii otwierania ekosystemu .NET przez firmę Microsoft, rozpoczętej wraz z wydaniem .NET Core w 2016 roku.

1.2. Dlaczego WPF?

WPF nie jest jedynym frameworkiem w środowisku .NET, który nadaje się do tworzenia praktycznych aplikacji — nie jest nawet najmłodszy! Skoro istnieją inne rozwiązania, które mają dodatkowe plusy: są młodsze, nowocześniejsze i wieloplatformowe, to dlaczego warto nauczyć się WPF? Dlaczego warto pracować akurat w tej technologii?

1.2.1. Połączenie prostoty i możliwości

WPF jest zarówno bardzo prostym, jak i złożonym frameworkiem pozwalającym na tworzenie funkcjonalnych aplikacji bez posiadania zaawansowanej wiedzy i jednocześnie umożliwia zastosowanie zaawansowanych mechanizmów do tworzenia profesjonalnych systemów — zarówno od strony danych, jak i wyglądu. Jest to sporą zaletą, która umożliwia naukę krok po kroku poprzez praktykę, a zdobyta wiedza przyda się podczas tworzenia większych, bardziej skomplikowanych projektów.

Dodatkowym atutem WPF, jak opisano wyżej, jest podobieństwo do innych technologii, takich jak AvaloniaUI czy MAUI. Pozwala to na wykorzystanie WPF nie tylko jako ścieżki kariery zawodowej, ale również jako progu wejścia — nauki umiejętności, które okażą się przydatne podczas tworzenia aplikacji z innymi narzędziami — raz poznane elementy, właściwości oraz sposoby tworzenia aplikacji nadal będą przydatne, nawet przy pracy z innymi frameworkami.

1.2.2. Rozwój poprzez społeczność

Jak już wspomniano, od 2018 roku WPF jest w pełni open-source. Główną korzyścią płynącą z takiego rozwiązania jest zwiększenie transparentności projektu i umożliwienie społeczności programistów bezpośredniego wpływu na rozwój i ulepszanie WPF. Dzięki otwartemu kodowi źródłowemu każdy może zgłaszać propozycje zmian, poprawki błędów oraz uczestniczyć w testach nowych funkcji, co nie tylko przyspiesza jego rozwój, ale również lepiej dostosowuje go do potrzeb użytkowników.

[1] Repozytorium można znaleźć na stronie: `https://github.com/dotnet/wpf`.

Ponadto rozwój poprzez społeczność otwiera dodatkowe możliwości integracji narzędzi i bibliotek w WPF. Przykładem takiej biblioteki jest tytułowy MaterialDesign, któremu poświęcimy sporo czasu w dalszej części książki. Warto jednak zauważyć następującą kwestię: otwartość WPF sprawia, że nadal jest to istotny framework do tworzenia aplikacji okienkowych, który można rozszerzyć na wiele sposobów.

1.2.3. Dojrzałość i ilość dostępnej dokumentacji

WPF pojawił się na rynku stosunkowo dawno, ponieważ w 2006 roku, co sprawia, że ilość dostępnej dokumentacji na jego temat jest ogromna. Nie chodzi tylko o kursy i książki, ale również o dokumentację techniczną kontrolek czy opisy często występujących błędów.

Niestety nie każda alternatywa do WPF może pochwalić się takim stanem rzeczy. W szczególności MAUI, który jest znacznie młodszy, nie jest w stanie poszczycić się dużą stabilnością czy ilością dokumentacji. Tymczasem szczególnie dla osoby zaczynającej przygodę z tworzeniem aplikacji okienkowych ilość dostępnych źródeł jest niezwykle istotna.

Warto również podkreślić, że WPF nadal jest aktywnie rozwijany. Na ten moment twórcy wraz ze społecznością pracują nad poprawą stylów (by lepiej dopasować je do wyglądu systemu Windows 11), typami dopuszczającymi wartość `null` oraz poprawkami dostępności czy stabilności. Mówiąc krótko, daleko jeszcze do zaprzestania jego aktywnego rozwoju.

1.2.4. WPF nadal jest dość popularny!

O ile zdecydowanie prawdą jest, że popularność WPF na rynku zmniejsza się, o tyle framework ten nigdzie się nie wybiera. WPF nadal jest szeroko stosowany przy tworzeniu systemów biznesowych dla firm i korporacji. Choć większość takich aplikacji nie ujrzy nigdy światła dziennego (ponieważ jest wykorzystywana wewnętrznie), nie odejmuje to wartości tej technologii. Wciąż istnieje i powstaje również wiele stanowisk pracy celujących konkretnie w WPF. Za to przykładami ogólnodostępnych aplikacji lub środowisk korzystających z WPF mogą być Stride, będący silnikiem do tworzenia gier 2D i 3D[2], czy Razer Synapse — aplikacja pozwalająca na zarządza-

[2] Więcej informacji o kwestiach technicznych Stride: `https://www.stride3d.net/blog/announcing-stride-4-2-in-dotnet-8`.

nie produktami firmy Razer[3]. W pewnym stopniu z WPF korzysta również środowisko programistyczne Visual Studio[4].

WPF nadal jest wykorzystywany nie tylko w zastosowaniach biznesowych, ale również do tworzenia personalnych projektów, w tym open-source. Aktualnie na platformie GitHub znajdują się 8524 repozytoria oznaczone jako WPF, zawierające aplikacje zbudowane z wykorzystaniem tego frameworku lub dodatkowe biblioteki. To znacznie więcej niż MAUI (1412), Xamarin (4197) czy Avalonia (609). Pomimo zanikającej popularności WPF na platformie StackOverflow nadal na porządku dziennym pojawiają się pytania dotyczące WPF, które mogą posłużyć nam jako dodatkowe wsparcie i dokumentacja przy rozwiązywaniu problemów.

Oficjalne repozytorium WPF na platformie GitHub również jest dość aktywne. Na moment pisania niniejszej książki w ciągu ostatniego roku do repozytorium wepchnięto 835 commitów, podczas gdy w ciągu ostatniego miesiąca pojawiło się 49 kwestii oraz 81 commitów wykonanych przez 13 autorów. O ile dane dotyczące wyświetlania repozytorium nie są dostępne publicznie, o tyle można się spodziewać, że jest ono daleko od rzadko odwiedzanego.

1.3. WPF kontra inne technologie w .NET

1.3.1. WPF kontra WinForms

WinForms to technologia starsza od WPF, która mimo dość dobrze rozbudowanej dokumentacji oraz dostępności narzędzi posiada wiele ograniczeń. Największym z nich jest brak zaawansowanego wsparcia dla nowoczesnych interfejsów użytkownika — nie tylko sam wygląd jest momentami dość przestarzały, lecz dodatkowym problemem może być także brak wsparcia dla złożonych animacji oraz przyspieszenia sprzętowego grafiki.

Największą zaletą WinForms w stosunku do WPF jest jego prostota — brak zaawansowanych mechanizmów, które są obecne w WPF, oraz możliwość tworzenia interfejsu metodą „drag-and-drop” przy pomocy edytora.

[3] Więcej informacji o kwestiach technicznych Synapse: `https://mysupport.razer.com/app/answers/detail/a_id/14146/open-source-software-for-razer-software`.

[4] Więcej informacji o wykorzystaniu WPF w Visual Studio: `https://web.archive.org/web/20100412115119/http://msmvps.com/blogs/carlosq/archive/2008/10/26/visual-studio-2010-extensibility-moving-beyond-add-ins-and-packages.aspx`.

Prostota ta może jednak być zgubna! W szczególności gdy mamy już bardziej zaawansowaną wiedzę, WinForms poprzez swoją prostotę w pewnym sensie „utrudnia" nam dostęp do właściwości kontrolek czy sposobu zdefiniowania całych okienek aplikacji, co w WPF jest osiągane poprzez pisanie interfejsu w XAML.

Warto jednak wiedzieć, że WinForms jest do dziś wykorzystywany i podobnie jak WPF często jest technologią do tworzenia wewnętrznego oprogramowania dla firm.

1.3.2. WPF kontra MAUI

MAUI to dość nowa technologia (wydana w 2022 roku) od firmy Microsoft. Jego główną zaletą jest wieloplatformowość — aplikacje pisane w MAUI można w prosty sposób uruchamiać na każdej platformie (Windows, MacOS, iOS, Android, a nawet WebAssembly z wykorzystaniem innego frameworku Blazor), z jednoczesną możliwością modyfikowania wyglądu interfejsu na każdej z nich (np. w celu wykorzystania dostępnego w urządzeniu sposobu nawigacji) czy wykorzystanie usług dostępnych tylko na platformach mobilnych.

To, że jest to świeża i młoda technologia, jest w przypadku MAUI problemem. Niewielka ilość dokumentacji, błędy z mało opisowymi komunikatami czy problemy z funkcjonalnością kontrolek to główne punkty, które sprawią, że pisanie aplikacji w MAUI będzie uciążliwym wyzwaniem. Na dodatek MAUI ma zdecydowanie wyższy próg wejścia w stosunku do podobnych technologii. Zdaniem autora MAUI ma potencjał, żeby zostać w przyszłości wartym wykorzystywania frameworkiem do tworzenia różnego typu aplikacji, jednak ta przyszłość jeszcze nie nastąpiła.

1.3.3. WPF kontra Avalonia

Dużo lepszą alternatywą, na czas pisania niniejszej książki, jest framework Avalonia, który jest mocno inspirowany WPF-em (tak naprawdę jest forkiem oficjalnego repozytorium WPF!). O ile w żadnym wypadku nie można powiedzieć, że już teraz wypiera on WPF, o tyle w taki właśnie sposób zapowiada się przyszłość.

Avalonia również jest frameworkiem open-source, który wykazuje się bardzo dużą aktywnością zarówno pod względem liczby osób wspierających projekt, jak i ilości zmian w repozytorium [1]. Podobnie jak w WPF interfejs użytkownika pisany jest w XAML, występują zaawansowane me-

chanizmy, takie jak wiązanie danych czy przyspieszenie sprzętowe renderowania. Framework mocno wspiera też architekturę MVVM, tak więc jest — pod względem teoretycznym — bardzo zbliżony do WPF.

Co istotne, migracja istniejących aplikacji z WPF do Avalonii jest możliwa, a to oznacza, że istniejące aplikacje WPF mogą w przyszłości zostać *relatywnie* łatwo napisane w Avalonii. W szczególności możliwości takie oferuje wersja Avalonia XPF, która jest jednak płatna i przeznaczona w szczególności dla większych firm, które chcą przekonwertować aplikacje biznesowe z WPF na ten wieloplatformowy framework.

Podsumowując, Avalonia nie jest w tym momencie obiektywnie lepszym frameworkiem od WPF, jednak wkrótce może nim zostać. Ponieważ jest jednak bardzo podobny do WPF, to wiedza nabyta podczas tworzenia aplikacji w tym drugim może zostać łatwo wykorzystana w przyszłości do tworzenia aplikacji wykraczających poza system Windows.

Zacznij programować aplikacje dla systemu Windows!

WPF (ang. Windows Presentation Foundation) jest frameworkiem przygotowanym przez firmę Microsoft do tworzenia aplikacji desktopowych lub „okienkowych" — to znaczy takich, które są bezpośrednio uruchamiane w systemie operacyjnym komputerów osobistych. Umożliwia on budowanie funkcjonalnych aplikacji także osobom nieposiadającym wysoce specjalistycznej wiedzy — wystarczą podstawy języka C#. Dzięki użyciu dostarczonych narzędzi zaczniesz budować profesjonalne oprogramowanie, zaawansowane zarówno pod względem funkcjonalności, jak i interfejsu.

Ta książka stanowi przewodnik po technologii WPF. Znajdziesz w niej:

- **najważniejszą wiedzę dotyczącą WPF — czym jest ten framework i jak się w nim tworzy interfejs użytkownika**
- **objaśnienie sposobu działania zaawansowanych mechanizmów WPF — w tym architektury MVVM, wiązania danych czy poleceń i szablonów**
- **informacje na temat standardu projektowania Material Design — jako narzędzia do tworzenia eleganckich interfejsów w prosty sposób**

Rafał Klinowski — programista specjalizujący się w .NET, przetwarzaniu obrazów i uczeniu maszynowym. Ma ponad trzyletnie doświadczenie zawodowe w tworzeniu aplikacji WPF dla systemu Windows, zarówno w projektach komercyjnych, jak i *open source*. Od 2024 roku związany z Uniwersytetem Bielsko-Bialskim, gdzie prowadzi badania nad sztuczną inteligencją. Pasjonat nauki, dzielenia się wiedzą i wspierania innych w rozwoju. W wolnym czasie podróżuje, gra na pianinie i ćwiczy jogę.

Helion

helion.pl

HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI
Sięgnij po więcej!

ISBN 978-83-289-2398-0

9 788328 923980

Cena: 89,00 zł