Pierwsze spojrzenie na .NET MAUI

Ten artykuł powstał w styczniu 2022. To ważne, bo nowe wersje zapoznawcze .NET MAUI powstają i publikowane są dosłownie co chwilę – bieżąca jest już jedenasta. Trzeba się więc liczyć z tym, że za kilka miesięcy, a w szczególności po oficjalnej premierze, która obecnie planowana jest na drugi kwartał 2022 roku, wiele jeszcze może się zmienić. Technologia .NET MAUI (od ang. Multi-platform App UI) to wieloplatformowa technologia umożliwiająca tworzenie aplikacji z graficznym interfejsem użytkownika kompilowanych dla .NET 6. Co oznacza wieloplatformowa? Obecnie dostępna jest na Windows, Android oraz systemy od Apple. Może być dostępna także na Linuxach, ale realizacja tego zadania oddana została w ręce społeczności. Historia .NET MAUI formalnie zaczyna się w 2020 roku, jednak rozwój tej technologii można też traktować jako kontynuację rozwoju Xamarin.Forms i przeniesienie jej z Mono do .NET 6.

MAUI

.NET MAUI pozwala tworzyć graficzny interfejs aplikacji przeznaczonych na różne platformy sprzętowe i systemowe, na których dostępne jest .NET 6. Co ważne, możemy to robić bez konieczności utrzymywania osobnych projektów dla warstwy widoku w każdej z obsługiwanych platform systemowych. Wcześniej to zadanie realizowało Xamarin.Forms, ale opinie o nim były „różne" – dość wspomnieć, że chyba jednak częściej stosowano osobne projekty warstwy widoku dla poszczególnych systemów. .NET MAUI ma zastąpić Xamarin.Forms i usunąć jego słabości, ale jest przy tym jego ewolucyjnym rozwinięciem.

Obsługiwane systemy operacyjne to Android od wersji 5.0 (API 21), iOS 10 lub nowszy, macOS 10.13 oraz Windows. W tym ostatnim przypadku chodzi zarówno o tzw. aplikacje „klasyczne" na desktop, jak i aplikacje uruchamiane na platformie UWP, a więc dystrybuowane przez Microsoft Store. Poza tym na stronie projektu znajduje się informacja, że w przyszłości wspierane będą platformy Tizen oraz Linux. Pierwsza jest systemem operacyjnym stworzonym przez firmę Samsung na bazie Linuxa i służy do obsługi smartwatchy i SmartTV; wsparcie dla niej będzie zapewniał właśnie Samsung. Natomiast wsparcie dla Linuxa ma być całkowicie w rękach społeczności. Na razie ani Tizen, ani Linux, jak również klasyczny desktopowy Windows, nie są obecne w dotychczas udostępnionych wersjach zapoznawczych .NET MAUI.

Aplikacje dla urządzeń mobilnych z systemami Android lub iOS z interfejsem budowanym za pomocą technologii MAUI mogą być tworzone za pomocą Visual Studio dla systemów Windows lub macOS w wersji 2022 Preview. Jednak jeżeli chcemy kompilować aplikacje dla iOS, musimy mieć również Mac'a z XCode 13.0 Beta 1. Tworzenie aplikacji MAUI dla Windows i macOS możliwe jest tylko w Visual Studio dla właściwego systemu.

MAUI unifikuje systemy umożliwiające tworzenie aplikacji z graficznym interfejsem użytkownika dla wymienionych wyżej platform w oparciu o język znaczników XAML. Ale nie tylko. Nowością jest również możliwość wykorzystania języka Blazor, który do tej pory kojarzyliśmy raczej z aplikacjami webowymi. .NET MAUI bazuje na rozwinięciu wieloplatformowego .NET Core, które obecnie porzuciło swój przydomek i nazywa się .NET 6, przejmując miejsce dotychczasowej flagowej platformy Microsoft .NET Framework. Aplikacje .NET MAUI mają dostęp do wspólnego dla wszystkich platform sprzętowych i systemowych zestawu klas .NET 6 (mowa oczywiście o BCL), który z kolei opiera się o Win32 z UWP w systemie Windows i Mono Runtime w pozostałych systemach. Wyżej są biblioteki służące do tworzenia aplikacji charakterystyczne dla poszczególnych systemów, a więc .NET for Android, .NET for iOS, .NET for Mac i WinUI 3 (z UWP). Na szczęście powyżej tego jest jeszcze nasze .NET MAUI, które unifikuje tworzenie aplikacji na wszystkich platformach.

Co ciekawe, unifikacja ta nie dotyczy tylko kontrolek, z których budujemy interfejs. .NET MAUI dostarcza również zunifikowany dostęp do sensorów (akcelerometr, żyroskop, kompas itp.), umożliwia sprawdzanie stanu sieci z wykrywaniem jego zmian, dostęp do informacji o urządzeniu, dostęp do plików i mechanizmów przechowywania danych, możliwość użycia wbudowanego syntezatora mowy itp. To oznacza, że, przynajmniej w pewnym zakresie, aplikacje napiszemy bez konieczności różnicowania kodu dla poszczególnych platform. Na razie trudno jednak ocenić, na ile jest to wiarygodna obietnica.

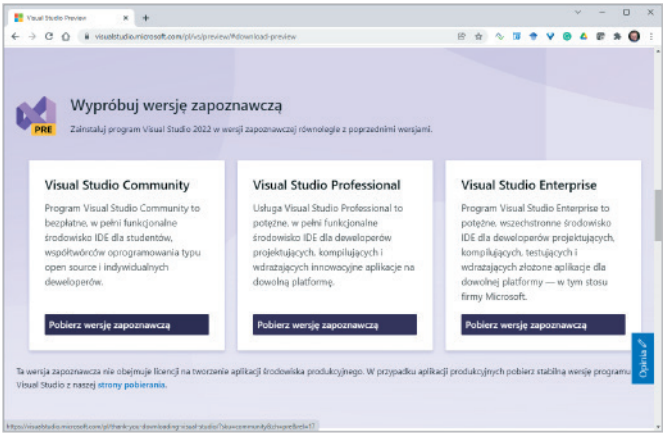
Konsekwencją nowego podejścia są rozwiązania, w których aplikacja znajduje się w pojedynczym projekcie, w którym różne platformy obsługiwane są w razie konieczności za pomocą „równoległych" plików umieszczonych w przypisanych platformom podfolderach, a nie osobnych projektów. To oczywiście powinno uprościć rozwój niezbyt rozbudowanych aplikacji.

Wartą odnotowania nowością .NET MAUI jest dostosowanie aplikacji do potrzeb osób z niepełnosprawnościami, w szczególności wsparcie dla czytników ekranów (ang. *screen reader*).

Więcej aktualnych informacji można znaleźć na stronie:
https://docs.microsoft.com/pl-pl/dotnet/maui/.

INSTALACJA VISUAL STUDIO 2022 PREVIEW

Zabawę z .NET MAUI musimy zacząć od instalacji Visual Studio 2022 w wersji Preview (VS). Jest ona konieczna, bo pakiety .NET MAUI nie są jeszcze dostępne w pełnej wersji tego środowiska. Wersję Preview możemy pobrać za darmo w różnych edycjach (Rysunek 1). Ja wybrałem Visual Studio Community.



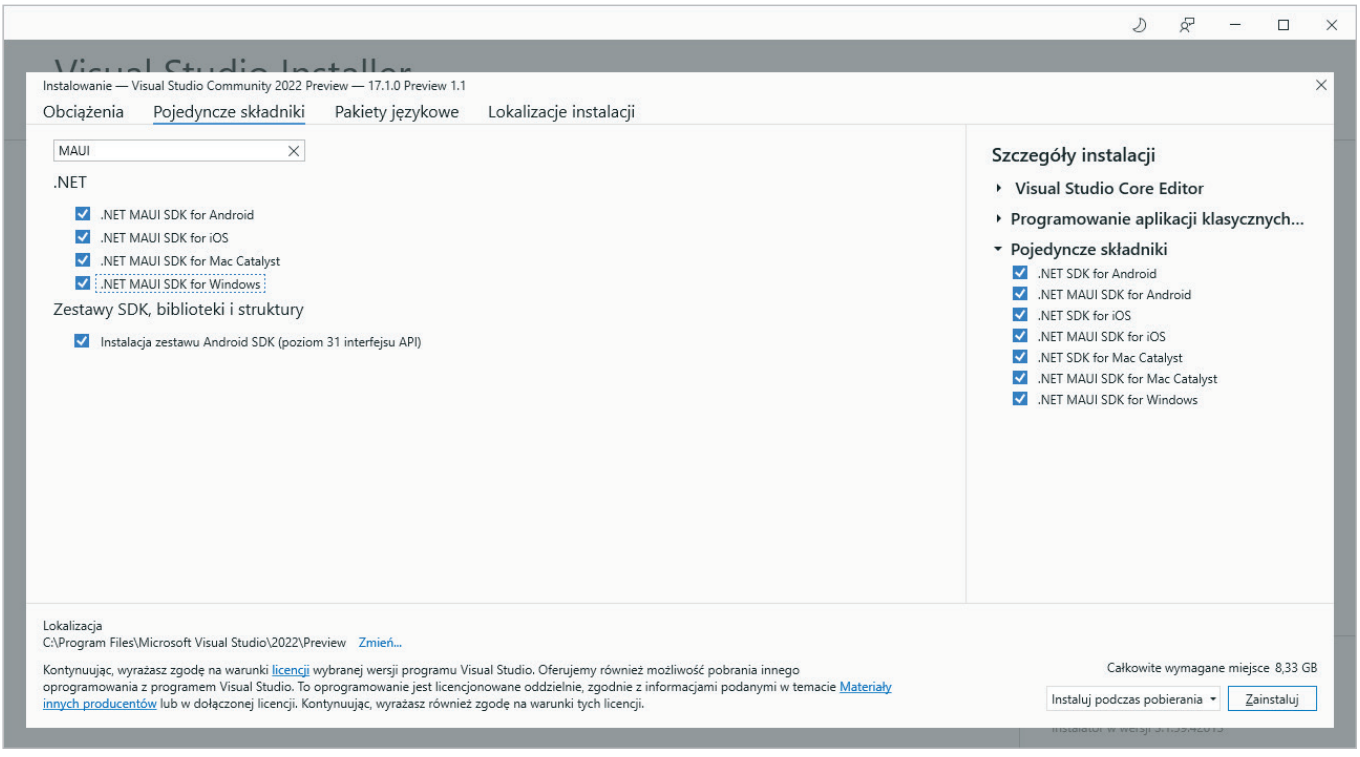
Rysunek 1. Dostępne do pobrania wersje Visual Studio

Podczas instalacji należy wybrać zestawy *Opracowywanie aplikacji mobilnych za pomocą środowiska .NET* oraz *Programowanie aplikacji klasycznych dla platformy .NET*. Obecnie ten pierwszy zestaw zawiera już domyślnie składniki wymagane do tworzenia aplikacji .NET MAUI, jednak we wcześniejszych wersjach – albo gdy tę możliwość chcemy dodać do już zainstalowanego zestawu – należy je dodać samodzielnie. W tym celu w instalatorze należy przejść na zakładkę *Pojedyncze składniki* i wpisać hasło „MAUI". Wówczas pojawią się cztery pakiety widoczne na Rysunku 2. Oprócz nich należy zainstalować także pakiet *Android SDK (poziom 31 interfejsu API)* – Rysunek 2. Ale bez obawy, tak wysokie API wcale nie jest potrzebne do uruchomienia aplikacji .NET MAUI – wystarczy API 20 odpowiadające Androidowi 5.0.

Z góry uprzedzam, że .NET MAUI potrafi być bardzo kapryśne. Projekty czasem nie chcą się kompilować, choć bez problemu można je skompilować po ponownym uruchomieniu Visual Studio. Bywa również tak, że w edytorze są widoczne fragmenty kodu podkreślane na czerwono, a projekt kompiluje się bez błędów (i podkreślenia wówczas wcale nie znikają). Przełączanie między platformami także nie zawsze działa idealnie – po przełączeniu nie można wdrożyć aplikacji na wybrany system itd. Należy zatem uzbroić się w cierpliwość, pamiętając, że nadal mamy do czynienia tylko z wersją zapoznawczą.

TWORZENIE PROJEKTU

Aby przyjrzeć się .NET MAUI, proponuję zbudować prostą aplikację, od której zawsze rozpoczynam poznawanie nowych technologii pozwalających na tworzenie graficznych interfejsów użytkownika (GUI). W aplikacji tej za pomocą trzech suwaków będziemy kontrolować kolor widocznego w oknie prostokąta. To prosty projekt, ograniczony wyłącznie do kontrolek i manipulacji nimi bez odwoływania się do bardziej złożonych zagadnień, charakterystycznych dla poszczególnych platform, ale umożliwia rozpoznanie podstawowych mechanizmów i narzędzi służących do budowania GUI. Zwykle oznacza to naukę wizualnych narzędzi projektowania interfejsów, które obsługiwane są myszką. Te jednak w przypadku MAUI nie są jeszcze udostępnione – w chwili pisania tego artykułu nie ma nawet podglądu projektowanego interfejsu.



Rysunek 2. Pakiety .NET MAUI w Visual Studio 2022 Preview