Dostępne Technologie i Biblioteki na Platformie .NET

Ecosystem rozwijany przez Microsoft i społeczność open source

Wprowadzenie do Ekosystemu .NET

Platforma **.NET** to ekosystem programistyczny o otwartym kodzie źródłowym, zaprojektowany do budowy różnych typów aplikacji. Oferuje solidny zestaw narzędzi, bibliotek i języków, które pozwalają deweloperom tworzyć rozwiązania dla wielu platform i scenariuszy.

Unifikowane podejście

Spójny zestaw narzędzi i bibliotek dla różnych typów aplikacji - od konsolowych po webowe i mobilne.

Wiele języków

Obsługa różnych języków programowania, w tym C#, F#, Visual Basic .NET i innych.

Rozbudowane narzędzia

Narzędzia do tworzenia, debugowania, profilowania i wdrażania aplikacji na różnych platformach.

Multiplatformowość

Wsparcie dla Windows, macOS i Linux, umożliwiające tworzenie aplikacji dla różnych systemów operacyjnych.

Dokumentacja i społeczność

Aktywna społeczność deweloperów i obszerna dokumentacja wspierają proces tworzenia aplikacji.

Bezpieczeństwo

Zintegrowane mechanizmy bezpieczeństwa i zarządzania pakietami, zapewniające bezpieczne i stabilne działanie aplikacji.

Aplikacje Konsolowe

Charakterystyka

69→ Aplikacje konsolowe to programy, które interagują z użytkownikiem przez interfejs wiersza poleceń (CLI) bez graficznego interfejsu użytkownika (GUI). Skupiają się na wejściu i wyjściu tekstu.

Główne biblioteki

Biblioteka `System.Console` 83→ dostarcza metody do:

- `Console.ReadLine()` - odczyt wejścia
- `Console.WriteLine()` - wyświetlanie wyjścia

Przypadki użycia

- Narzędzia linii komend do automatyzacji zadań
- Skrypty do przetwarzania danych
- Utilitary systemowe
- Usługi backendowe działające w tle

Przykład implementacji

```
// .NET 6+
using System ;
namespace HelloWorld
{
static void Main ( string [] args)
{
Console.WriteLine ( "Hello World!" );
}
}
```

W .NET 6+ można napisać "Hello World" w jednej linii: `Console.WriteLine("Hello World!");`

Aplikacje Desktopowe: Windows Forms

Windows Forms (WinForms) to dojrzała i solidna technologia do szybkiej tworzenia aplikacji desktopowych dla systemu Windows. Zapewnia deweloperom intuicyjny sposób budowania interfejsów użytkownika oraz model programowania oparty na zdarzeniach.

Kluczowe cechy

Wizualny projektant

Intuicyjny design interfejsu użytkownika za pomocą technologii drag-and-drop.

Architektura oparta na zdarzeniach

Ułatwia interakcję z komponentami interfejsu użytkownika poprzez zdarzenia.

Szybki prototyp

Możliwość szybkiego tworzenia prototypów i aplikacji produkcyjnych.

Zastosowania i zalety

Typowe zastosowania

- Aplikacje business line
- Systemy zarządzania
- Utilitary aplikacje

Zalety

- Szybkość desenvolvimento
- Integracja z Windows
- Długa historia i aktywna obsługa

Aplikacje Desktopowe: Windows Presentation Foundation (WPF)

Windows Presentation Foundation (WPF) to nowoczesny framework do tworzenia aplikacji desktopowych Windows, oferujący zintegrowane podejście do programowania interfejsów użytkownika.

Przykład XAML:

```
<Window x:Class="MojaAplikacja.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Moja Aplikacja" Width="300" Height="200">
    <Grid>
        <TextBlock Text="Witaj świecie!"
                   HorizontalAlignment="Center"
                   VerticalAlignment="Center"
                   FontSize="18" />
    </Grid>
</Window>
```

Język XAML

Definiowanie interfejsów użytkownika w języku XAML (Extensible Application Markup Language) oddzielone jest od logiki biznesowej napisanej w C#.

Powerful Data Binding

WPF oferuje zaawansowany system bindingu danych, który upraszcza synchronizację danych między interfejsem a modelem danych aplikacji.

Zaawansowane grafiki

Możliwości graficzne WPF obejmują 2D/3D grafikę, animacje, efekty wizualne i kompleksowe systemy rysowania.

Motywy i stylowanie

Możliwość tworzenia nowoczesnych motywów i stylów interfejsu użytkownika, umożliwiająca dostosowanie wyglądu aplikacji do potrzeb.

Aplikacje Wieloplatformowe: .NET MAUI

.NET MAUI (Multi-platform App UI) to evolucja **Xamarin.Forms** i reprezentuje zintegrowane rozwiązanie .NET do tworzenia natywnych aplikacji wieloplatformowych.

Główne Zalety

Jedna baza kodu

Logika biznesowa, interfejs użytkownika i zasoby wspólne są pisane raz i uruchamiane na wielu platformach.

Skracenie czasu developmentu

Zmniejszenie czasu i kosztów programowania dzięki wykorzystaniu istniejącego kodu i zespołów.

107→ Znane technologie

Użycie znanych technologii .NET (C# i XAML) znanych deweloperom Windows Forms i WPF.

Obsługiwane Platformy

Windows

macOS

iOS

Android

Przykład

Formularz logowania napisany w XAML i C# będzie działał natywnie na iPhone'u, telefonie Android, komputerze Windows i Macu, automatycznie dostosowując się do cech każdego systemu operacyjnego.

Tworzenie Aplikacji Webowych z ASP.NET Core

51→ ASP.NET Core to framework open-source i multiplatformowy dla .NET, zaprojektowany do budowy nowoczesnych aplikacji webowych, usług i API. Oferuje on ekosystem solidnych narzędzi do tworzenia skalowalnych i wydajnych rozwiązań webowych.

ASP.NET Core MVC

Wzorzec architektoniczny **Model-Widok-Kontroler** , który organizuje kod aplikacji webowej i API w trzy główne komponenty:

- **Model** - reprezentuje dane aplikacji i logikę biznesową
- **Widok** - odpowiedzialny za interfejs użytkownika i wyświetlanie danych
- **Kontroler** - zarządza interakcjami użytkownika i aktualizuje model

ASP.NET Core Razor Pages

Model programowania bardziej prosty i skierowany na strony, idealny dla mniejszych aplikacji webowych lub z prostszymi przepływami pracy:

- Każda strona Razor to autonomiczna jednostka kombinująca logikę biznesową z kodem HTML
- Uproszcza programowanie interfejsów użytkownika, ponieważ logika i prezentacja są w jednym miejscu
- Bardziej dostępny dla deweloperów preferujących podejście bezpośrednie w porównaniu do MVC

Blazor

Innowacyjna technologia umożliwiająca budowanie interfejsów użytkownika webowych z użyciem C# zamiast JavaScript:

- **Blazor Server** - aplikacja uruchamiana na serwerze, a interakcje użytkownika obsługiwane via SignalR
- **Blazor WebAssembly** 171→ - aplikacja uruchamiana bezpośrednio w przeglądarce klienta przy użyciu WebAssembly
- Umożliwia deweloperom .NET rozwijanie aplikacji full-stack z użyciem jednego języka

Entity Framework Core

Entity Framework Core (EF Core) to standardowy, open-source ORM (Object-Relational Mapping) dla .NET, który umożliwia interakcję z bazami danych za pomocą obiektów C# zamiast pisania zapytań SQL bezpośrednio.

Obsługiwane bazy danych

EF Core obsługuje różne bazy danych: SQL Server, SQLite, PostgreSQL, MySQL i inne.

Śledzenie zmian

Automatyczne śledzenie zmian obiektów i synchronizacja z bazą danych.

Migracje

Zarządzanie zmianami schematu bazy danych przez migracje kodu, które można automatyzować.

LINQ

Obsługa zapytań LINQ do manipulacji danymi, które są konwertowane na SQL.

Korzyści z użycia EF Core

- 136→ Abstrakcja nad złożonością operacji na bazie danych
- Możliwość pracy z bazą danych w sposób obiektowy
- Ograniczenie potrzeby pisania SQL

Chmura Obliczeniowa i Usługi Azure

Platforma **.NET** oferuje silną integrację z usługami chmurowymi, szczególnie Microsoft Azure, umożliwiając deweloperom budowanie, wdrażanie i zarządzanie skalowalnymi i odpornymi na błędy aplikacjami w chmurze.

Azure Functions

Rozwiązanie do obliczeń bezserwerowych, które umożliwia uruchamianie kodu na żądanie bez konieczności provisioningu lub zarządzania infrastrukturą.

79→ Wykonywanie kodu na żądanie

Azure App Service

Pełnowartościowa platforma zarządzana do hostowania aplikacji webowych, API REST i back-endów mobilnych.

Hostowanie aplikacji webowych

SDK .NET dla Azure

113→ Biblioteki ułatwiające interakcję z różnymi usługami Azure, takimi jak magazyn, bazy danych i usługi AI, bezpośrednio z kodu .NET.

Integracja z usługami Azure

Korzyści z używania .NET z Azure

- Skalowalność aplikacji w chmurze bez zarządzania infrastrukturą
- Odporność na błędy i wysoka dostępność aplikacji
- Szybsze tempo wdrażania i skalowania aplikacji
- Integracja z ekosystemem .NET i narzędziami developmentowymi

Inne Technologie: Gry i Sztuczna Inteligencja

Tworzenie Gier

- **Język C#** - szeroko wykorzystywany w developmencie gier
- **Unity** - jeden z najpopularniejszych platformów do tworzenia gier
- Obsługa gier 2D, 3D, VR i AR
- Krzywa nauki - łatwy dostęp do zaawansowanych funkcji

Unity to jedna z najpopularniejszych platform do tworzenia gier na świecie, umożliwiająca rozwój dla wielu platform.

Sztuczna Inteligencja

- **ML.NET** - biblioteka open-source do uczenia maszynowego
- Integracja modeli AI bez opuszczenia ekosystemu .NET
- Obsługa zadań: klasyfikacja, regresja, grupowanie
- Detekcja anomalii i przetwarzanie języka naturalnego

ML.NET pozwala deweloperom .NET integrować modele AI do aplikacji bez konieczności uczenia się nowych języków.

Platforma .NET sięga poza tradycyjne zastosowania, oferując solidne rozwiązania w dziedzinie gier i sztucznej inteligencji, co podkreśla jej wszechstronność i elastyczność.