Publikowanie aplikacji

Czas: 2 × 45 min | praca indywidualna | Wykład 2.20

Cel: Przygotujesz wydanie aplikacji z zadania 14/15: metadane, ikonę, dwie wersje publikacji (zależną od .NET i samodzielną), instrukcję dla użytkownika — i sprawdzisz je na innym komputerze.

Po zajęciach potrafisz:

- ustawia metadane wydania i ikonę w pliku projektu
- publikuje aplikację w konfiguracji Release w trybie framework-dependent i self-contained
- przygotowuje paczkę i instrukcję dla użytkownika
- testuje wydanie poza środowiskiem programisty

Sprzęt i materiały

- komputer z Windows 11 i dostępem do internetu
- pendrive lub konto w repozytorium do zapisania projektu
- drugi komputer (np. sąsiednie stanowisko lub laptop) do testu

Oprogramowanie

- **Visual Studio Community 2026 (obciążenie „Programowanie aplikacji klasycznych platformy .NET")** — darmowe dla uczniów i nauczycieli; w pracowni zainstalowane visualstudio.microsoft.com/pl/downloads
- **.NET 10 SDK (LTS)** — instalowany razem z Visual Studio; sprawdzenie: `dotnet --version` dotnet.microsoft.com/pl-pl/download/dotnet/10.0

Bezpieczeństwo

- Zapisuj projekt w swoim folderze (np. `C:\Users\<login>\source\repos`), a na koniec zajęć skopiuj go na pendrive lub wyślij do repozytorium.
- Nie uruchamiaj programów i pakietów z nieznanych źródeł; pakiety NuGet instaluj tylko te, które wskazuje zadanie.
- Rób przerwy dla oczu (zasada 20-20-20) — praca z projektantem okien wymaga długiego patrzenia w ekran.

Przebieg zadania

1 Metadane i ikona

Plik projektu (prawy przycisk → Edytuj plik projektu).

1. Version 1.0.0, Product, Company, Copyright, Description, AssemblyName.
2. Ikona `Assets\dziennik.ico` (kilka rozmiarów, np. wygenerowana z PNG w darmowym narzędziu) — `ApplicationIcon` i ikona okna.
3. Okno „O programie": produkt, wersja i środowisko odczytane z metadanych (klasa `InformacjeOProgramie`).

2 Publikacja

Dwa warianty.

```
dotnet publish -c Release -r win-x64 --self-contained true `
    -p:PublishSingleFile=true -p:IncludeNativeLibrariesForSelfExtract=true `
    -o publish\samodzielna
```

1. Zależna od .NET, jeden plik: `dotnet publish -c Release -r win-x64 --self-contained false -p:PublishSingleFile=true`.
2. Samodzielna, jeden plik: `--self-contained true` i `-p:IncludeNativeLibrariesForSelfExtract=true` (SQLite ma bibliotekę natywną).
3. Zapisz rozmiary obu wersji. Ten sam proces z interfejsu: prawy przycisk → Publikuj → Folder.
4. Skrypt `publikuj.ps1` z rozwiązania tworzy obie wersje i archiwa ZIP.

3 Paczka

Dla użytkownika.

1. README.md: wymagania, uruchomienie, ostrzeżenie SmartScreen, autor (szablon w rozwiązaniu).
2. Archiwa ZIP z wersją i rodzajem w nazwie.
3. Sprawdź właściwości pliku .exe w Eksploratorze (Szczegóły) — czy widać wersję i produkt?

4 Test na innym komputerze

Poza Visual Studio.

1. Uruchom obie wersje na innym komputerze (wersja zależna — bez zainstalowanego .NET 10 Desktop Runtime pojawi się komunikat; zapisz go).
2. Sprawdź: utworzenie bazy, dodanie oceny, zapis ustawień.
3. Zatwierdź pracę w Git (folder publish w .gitignore).

W sprawozdaniu umieść

- ☐ fragment pliku projektu z metadanymi
- ☐ tabela rozmiarów i wyników testu
- ☐ README
- ☐ zrzut właściwości pliku .exe i okna O programie

Pytania kontrolne

1. Kiedy wybierzesz publikację self-contained?
2. Dlaczego potrzebny jest parametr IncludeNativeLibrariesForSelfExtract?
3. Skąd okno „O programie” zna numer wersji?
4. Co oznacza ostrzeżenie SmartScreen i jak je usunąć w profesjonalnym oprogramowaniu?

Kryteria oceny

Kryterium	Punkty
Metadane, ikona, O programie	4
Dwie publikacje	5
README i paczki	3
Test na innym komputerze	5
Pytania	3
Razem	**20**

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).