.NET Application Publishing Guide

Framework-Dependent vs Self-Contained Deployment

Table of Contents

Before You Start (Prerequisites)

Step 1: Check if .NET SDK is Installed

Open your command prompt/terminal and type:

dotnet --version

Expected Result: You should see a version number like 8.0.100

Step 2: Make Sure Your Project Builds

Navigate to your project folder and run:

dotnet build

Expected Result: Build should succeed with no errors

Step 3: Navigate to the Correct Folder

Important: You must be inside your project folder (where the .csproj file is)

cd ConsoleLearning

Warning: Do not run from the solution folder. Always go inside the project folder first.

Understanding the Two Publishing Methods

Method 1: Framework-Dependent Deployment (FDD)

Simple Explanation: Your app needs .NET to be already installed on the computer where it will run.

Advantages:

- **Smaller file size (5-15 MB)**
- **Faster to publish**
- **Good for company servers**

Disadvantages:

- **Target computer must have .NET installed**

Method 2: Self-Contained Deployment (SCD)

Simple Explanation: Your app includes everything it needs. No need to install .NET separately.

Advantages:

- **Runs on any computer (even without .NET)**
- **Perfect for sharing with others**
- **Best for assignments/submissions**

Disadvantages:

- **Larger file size (70-150 MB)**

Method 1: Framework-Dependent Deployment (FDD)

When to Use This Method

- **You're running the app on your own computer**
- **Company servers that already have .NET installed**
- **You want smaller file sizes**

Step-by-Step Publishing

Step 1: Make sure you're in the project folder

cd ConsoleLearning

Step 2: Run the publish command

dotnet publish -c Release

Step 3: Find your published files Your files will be in this location:

bin/Release/net8.0/publish/

What Files Will You Get

- **ConsoleLearning.dll - Your main application file**
- **ConsoleLearning.deps.json - Dependency information**
- **ConsoleLearning.runtimeconfig.json - Runtime configuration**

How to Run Your Published App

dotnet ConsoleLearning.dll

Remember: The computer must have .NET installed.

Method 2: Self-Contained Deployment (SCD)

When to Use This Method

- **Submitting assignments/projects**
- **Sharing with friends who don't have .NET**
- **Creating desktop applications**
- **Target computer doesn't have .NET**

Step 1: Choose Your Target Operating System

Operating System	Code to Use
Windows 64-bit	win-x64
Windows 32-bit	win-x86
Linux 64-bit	linux-x64
macOS 64-bit	osx-x64

Example: If you want to run on Windows 64-bit, use win-x64

Step 2: Run the Publish Command

For Windows 64-bit:

dotnet publish -c Release -r win-x64 --self-contained true

For Linux 64-bit:

dotnet publish -c Release -r linux-x64 --self-contained true

Step 3: Find Your Published Files

Your files will be in:

bin/Release/net8.0/win-x64/publish/

What Files Will You Get

- **ConsoleLearning.exe - Your application (Windows)**
- **Many .dll files - .NET runtime files (included automatically)**

How to Run Your Published App

Windows: Just double-click ConsoleLearning.exe

Linux/macOS:

./ConsoleLearning

Note: Works even if .NET is NOT installed.

Single-File Publishing (Optional)

What is This

Instead of many files, you get just ONE file that you can share easily.

Framework-Dependent Single File

dotnet publish -c Release -p:PublishSingleFile=true

Self-Contained Single File (Recommended for Sharing)

dotnet publish -c Release -r win-x64 --self-contained true -p:PublishSingleFile=true

Result: One .exe file that contains everything. Perfect for sending via email or USB drive.

Common Mistakes to Avoid

Mistake 1: Running from Wrong Folder

Problem: You run dotnet publish from solution folder instead of project folder

Error Message: "Could not find project or file to operate on"

Solution:

cd ConsoleLearning

Always go inside the project folder first.

Mistake 2: Using Wrong Operating System Code

Problem: You publish for win-x64 but try to run on Linux

Solution: Make sure you use the correct code:

- **Windows - win-x64**
- **Linux - linux-x64**
- **macOS - osx-x64**

Mistake 3: Expecting .exe File in Framework-Dependent Mode

Problem: You publish using FDD but can't find .exe file

Solution: FDD creates .dll files only. If you want .exe, use Self-Contained mode.

Quick Reference Table

Scenario	Which Method to Use
Assignment submission	Self-Contained (--self-contained true)
Sharing with friends	Self-Contained Single File
Running on your own PC	Framework-Dependent
Company servers	Framework-Dependent
Creating desktop app	Self-Contained
Target PC has .NET	Framework-Dependent
Target PC has NO .NET	Self-Contained

File Size Comparison

Publishing Method	Typical Size
Framework-Dependent	5-15 MB

Publishing Method	Typical Size
Self-Contained	70-150 MB
Self-Contained Single File	60-120 MB

For Interview Questions

Question: What's the difference between Framework-Dependent and Self-Contained deployment?

Answer: Framework-Dependent deployment creates smaller applications that require .NET to be pre-installed on the target machine. Self-Contained deployment packages the .NET runtime with the application, making it larger but able to run independently without requiring .NET installation. We choose based on the deployment environment and distribution requirements.

Final Checklist Before Publishing

- **Project builds successfully (dotnet build)**
- **I'm in the correct project folder**
- **I've chosen the right publishing method (FDD or SCD)**
- **I've chosen the correct operating system code (if using SCD)**
- **I've tested the published application**

Summary for Beginners

For Assignment Submission:

dotnet publish -c Release -r win-x64 --self-contained true -p:PublishSingleFile=true

This gives you ONE .exe file you can submit.

For Running on Your Own Computer:

dotnet publish -c Release

Then run with: dotnet ConsoleLearning.dll
