Publishing .NET Applications (Self-Contained vs Framework-Dependent)

---

1. Prerequisites

Before publishing, ensure:

1. .NET SDK is installed

   ```
   dotnet --version
   ```

2. Your project builds successfully

   ```
   dotnet build
   ```

3. You are inside the **project folder** (where `.csproj` exists)

   ```
   cd ConsoleLearning
   ```

⚠️ `dotnet publish` must be run from the **project directory**, not the solution directory.

---

2. Key Concepts (Quick Understanding)

Framework-Dependent Deployment (FDD)

- Application **depends on .NET runtime installed on target machine**
- Smaller output size
- Faster publish
- Runtime must be installed separately

Self-Contained Deployment (SCD)

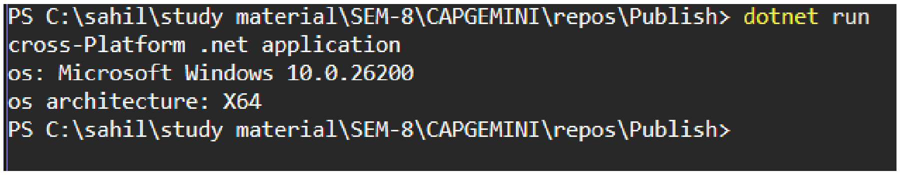
- Application **includes .NET runtime inside output**
- Larger output size
- Runs on machine **without .NET installed**
- OS-specific

---

3. Framework-Dependent Deployment (FDD)

3.1 When to Use FDD

Use FDD when: - Target machine already has .NET installed - Internal/company servers - Docker images with .NET base image - You want smaller binaries

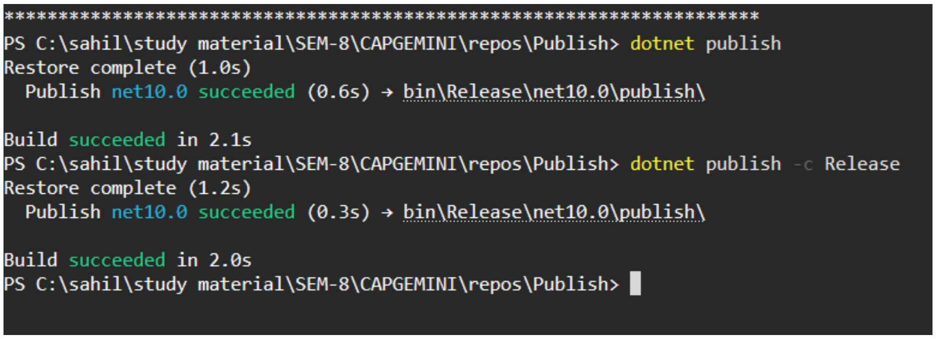


```
PS C:\sahil\study material\SEM-8\CAPGEMINI\repos\Publish> dotnet run
cross-Platform .net application
os: Microsoft Windows 10.0.26200
os architecture: X64
PS C:\sahil\study material\SEM-8\CAPGEMINI\repos\Publish>
```

3.2 Command to Publish (Framework-Dependent)

```
dotnet publish -c Release
```

This publishes the application using default framework-dependent mode.

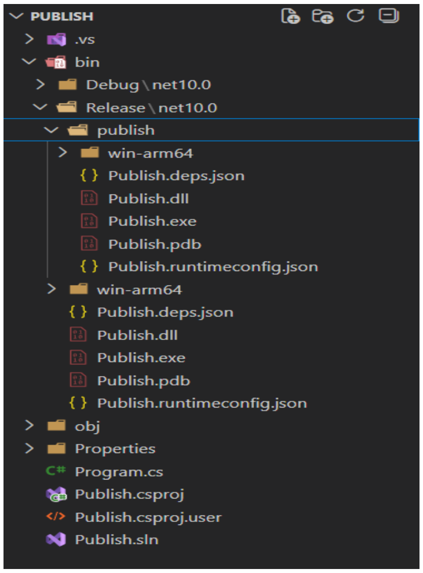


```
**********************************************************************
PS C:\sahil\study material\SEM-8\CAPGEMINI\repos\Publish> dotnet publish
Restore complete (1.0s)
  Publish net10.0 succeeded (0.6s) → bin\Release\net10.0\publish\

Build succeeded in 2.1s
PS C:\sahil\study material\SEM-8\CAPGEMINI\repos\Publish> dotnet publish -c Release
Restore complete (1.2s)
  Publish net10.0 succeeded (0.3s) → bin\Release\net10.0\publish\

Build succeeded in 2.0s
PS C:\sahil\study material\SEM-8\CAPGEMINI\repos\Publish>
```

3.3 Output Location

After publishing, files are created at:

```
bin/
 └── Release/
      └── net8.0/
           └── publish/
```



```
PUBLISH
> .vs
v bin
  > Debug\net10.0
  v Release\net10.0
    v publish
      > win-arm64
        Publish.deps.json
        Publish.dll
        Publish.exe
        Publish.pdb
        Publish.runtimeconfig.json
    > win-arm64
      Publish.deps.json
      Publish.dll
      Publish.exe
      Publish.pdb
      Publish.runtimeconfig.json
> obj
> Properties
  Program.cs
  Publish.csproj
  Publish.csproj.user
  Publish.sln
```

3.4 Files Generated (Important)

- `.dll` → Main application
- `.deps.json`
- `.runtimeconfig.json`

To run the app:

```
dotnet ConsoleLearning.dll
```

> ❗ Requires matching .NET runtime installed on the machine

3.5 Verify Framework Dependency

Check runtime config:

```
{
  "framework": {
    "name": "Microsoft.NETCore.App",
    "version": "8.0.0"
  }
}
```

This confirms **framework-dependent mode**.

---

4. Self-Contained Deployment (SCD)

4.1 When to Use SCD

Use SCD when: - Target machine has **no .NET installed** - Client systems - Desktop utilities - Exam / submission / offline delivery

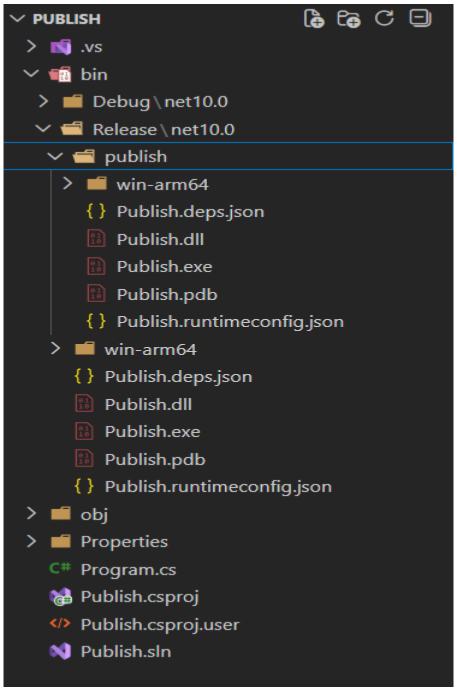
---

4.3 Command to Publish (Self-Contained)

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

4.4 Output Location

```
bin/
 └── Release/
      └── net8.0/
           └── win-x64/
                └── publish/
```



```
PUBLISH
> .vs
v bin
  > Debug \ net10.0
  v Release \ net10.0
    v publish
      > win-arm64
        { } Publish.deps.json
        Publish.dll
        Publish.exe
        Publish.pdb
        { } Publish.runtimeconfig.json
    > win-arm64
      { } Publish.deps.json
      Publish.dll
      Publish.exe
      Publish.pdb
      { } Publish.runtimeconfig.json
> obj
> Properties
  Program.cs
  Publish.csproj
  Publish.csproj.user
  Publish.sln
```

4.5 Files Generated

- `.exe` (Windows)
- .NET runtime files
- No dependency on external runtime

Run directly:

```
ConsoleLearning.exe
```

✅ Works even if .NET is NOT installed

4.6 Verify Self-Contained Mode

Check runtime config file:

```
{
  "includedFrameworks": [
    {
      "name": "Microsoft.NETCore.App",
      "version": "8.0.0"
    }
  ]
}
```

This confirms **self-contained deployment**.

---

5. Optional: Single-File Publishing

5.1 Framework-Dependent Single File

```
dotnet publish -c Release -p:PublishSingleFile=true
```

5.2 Self-Contained Single File

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

Output: - One `.exe` file - Best for distribution

---

6. Common Mistakes & Fixes

Mistake 1: Running from solution folder

❌ `dotnet publish` fails

✅ Fix:

```
cd ConsoleLearning
```

Mistake 2: Expecting `.exe` in FDD

❌ Only `.dll` generated

✅ Fix: Use self-contained mode

---

7. Recommended Usage (Industry Practice)

Scenario	Recommended Mode
Internal servers	Framework-Dependent
Client machines	Self-Contained
Exam submission	Self-Contained

Scenario	Recommended Mode
Docker	Framework-Dependent
Utilities	Self-Contained Single File

8. Final Checklist

✓ Build succeeds ✓ Correct project folder ✓ Correct RID ✓ Correct deployment mode ✓ Verified output