.NET CLI dotnet SDK commands

CONCEPTS

Build, run, test and publish restore NuGet packages first, so a separate restore is rarely needed.

Source .cs + .csproj
↓ restore
Packages obj/project.assets.json
↓ build
Binaries bin/<config>/<tfm>
↓ publish
Deploy folder bin/.../publish

Without a path argument, commands pick the one project or solution file in the current folder; with several, name one.

Arguments the app should get go after a literal --: dotnet run -- --help.

CREATE

`dotnet new list`
List installed templates

`dotnet new console -o <name>`
New console app in a folder

`dotnet new classlib -o <name>`
New class library

`dotnet new webapi -o <name>`
New ASP.NET Core web API

`dotnet new xunit -o <name>`
New xUnit test project

`dotnet new sln`
New solution file

BUILD AND RUN

`dotnet build`
Compile the project or solution

`dotnet build -c Release`
Build in Release configuration

`dotnet run`
Build and run the project

`dotnet run --project <path>`
Run a project from elsewhere

`dotnet run -- <args>`
Pass arguments to the app

`dotnet watch`
Rerun or hot reload on change

TEST

`dotnet test`
Build and run all tests

`dotnet test --filter <expr>`
Run only matching tests

`dotnet test --filter "Name~<text>"`
Match by test name

`dotnet test -l "console;verbosity=detailed"`
Show each test result

`dotnet test --collect "XPlat Code Coverage"`
Collect coverage with Coverlet

PACKAGES AND REFERENCES

`dotnet add package <id>`
Add a NuGet package

`dotnet add package <id> -v <version>`
Add a specific version

`dotnet remove package <id>`
Remove a package

`dotnet list package`
List referenced packages

`dotnet list package --outdated`
Show newer versions

`dotnet list package --vulnerable`
Show known vulnerabilities

`dotnet add reference <csproj>`
Reference another project

SOLUTIONS

`dotnet sln add <csproj>`
Add a project to the solution

PUBLISH

`dotnet publish -c Release`
Publish for deployment

`dotnet publish -o <dir>`
Publish into a folder

`dotnet publish -r <rid> --self-contained`
Bundle the runtime

`dotnet publish -p:PublishSingleFile=true`
One-file executable

TOOLS AND FORMAT

`dotnet tool install -g <id>`
Install a global tool

`dotnet tool install <id>`
Install a local (repo) tool

`dotnet tool restore`
Install tools from the manifest

`dotnet format`
Apply .editorconfig style fixes

`dotnet format --verify-no-changes`
CI check, fails if changes needed

SDK AND WORKLOADS

`dotnet --info`
SDK, runtime and OS details

`dotnet --version`
SDK version in use

`dotnet --list-sdks`
Installed SDKs

`dotnet --list-runtimes`
Installed runtimes

`dotnet <command> -h`
Help for any command

global.json in a parent folder pins the SDK version dotnet picks; otherwise the newest installed SDK is used.

RECIPES

New app with tests in a solution

```
dotnet new sln -n App
dotnet new console -o src/App
dotnet new xunit -o tests/
App.Tests
dotnet sln add src/App tests/
App.Tests
dotnet add tests/App.Tests
reference src/App/App.csproj
dotnet test
```

CI pipeline steps

```
dotnet restore
dotnet format --verify-no-changes
--no-restore
dotnet build -c Release --no-
restore
dotnet test -c Release --no-build
```

Single-file Linux binary

```
dotnet publish -c Release -r
linux-x64 \
  --self-contained -
p:PublishSingleFile=true \
  -o out
```