

Unleash your build with NUKE

Todor Todorov | @totollygeek
Cloud Principal Engineer | DELL Technologies



Todor Todorov
@totollygeek

- » .NET developer;
- » clean code fanatic;
- » DevOps evangelist;
- » speaker;
- » father of 3 boys;
- » karaoke enthusiast;

How do we do
builds?

Problem?



Search or jump to...



[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)



[totollygeek](#) / [awesome-app](#)

[Unwatch](#) 1

[Star](#) 0

[Fork](#) 0

[Code](#)

[Issues](#)

[Pull requests](#)

[Actions](#)

[Projects](#)

[Wiki](#)

[Security](#)

[Insights](#)

[Settings](#)

[awesome-app](#) / [.github](#) / [workflows](#) / `continuous.yml`

in `main`

[Cancel changes](#)

[Start commit](#)

[Edit file](#)

[Preview changes](#)

Spaces

2

No wrap

[Marketplace](#)







[Documentation](#)

```
29 - uses: actions/checkout@v1
30 - name: Cache .nuke/temp, ~/.nuget/packages
31   uses: actions/cache@v2
32   with:
33     path: |
34       .nuke/temp
35       ~/.nuget/packages
36     key: ${{ runner.os }}-${{ hashFiles('global.json', 'source/**/*.csproj') }}
37 - name: Run './build.cmd Test PushPackages'
38   run: ./build.cmd Test PushPackages
39   env:
40     NuGetApiKey: ${{ secrets.NUGET_API_KEY }}
41 - uses: actions/upload-artifact@v1
42   with:
43     name: test-results
44     path: output/test-results
45 - uses: actions/upload-artifact@v1
46   with:
47     name: packages
48     path: output/packages
49 ubuntu-latest:
50   name: ubuntu-latest
```

Use `Control` + `Space` to trigger autocomplete in most situations.

Search Marketplace for Actions

Featured Actions

-  **Close Stale Issues** ☆ 389
By actions 
Close issues and pull requests with no recent activity
-  **Setup Java JDK** ☆ 383
By actions 
Set up a specific version of the Java JDK and add the command-line tools to the PATH
-  **Download a Build Artifact** ☆ 305
By actions 
Download a build artifact that was previously uploaded in the workflow by the upload-artifact action

<> Edit file

Preview changes

Spaces

2

No wrap

```
29   - uses: actions/checkout@v1
30   - name: Cache .nuke/temp, ~/.nuget/packages
31     uses: actions/cache@v2
32     with:
33       path: |
34         .nuke/temp
35         ~/.nuget/packages
36       key: ${{ runner.os }}-${{ hashFiles('global.json', 'source/**/*.csproj') }}
37   - name: Run './build.cmd Test PushPackages'
38     run: ./build.cmd Test PushPackages
39     env:
40       NuGetApiKey: ${{ secrets.NUGET_API_KEY }}
41   - uses: actions/upload-artifact@v1
42     with:
43       name: test-results
44       path: output/test-results
45   - uses: actions/upload-artifact@v1
46     with:
47       name: packages
48       path: output/packages
49 ubuntu-latest:
50   name: ubuntu-latest
```

Use **Control** + **Space** to trigger autocomplete in most situations.

```
40     NUGETAPIKEY: ${{ secrets.NUGET_API_KEY }}
41   - uses: actions/upload-artifact@v1
42     with:
43       name: test-results
44       path: output (test-results)
45   - uses: actions/upload-artifact@v1
46     with:
47       name: packages
48       path: output (packages)
49   ubuntu-latest:
50     name: ubuntu-latest
```

Use **Control** + **Space** to trigger auto-complete in most situations.

Try a broken yaml .github/workflows/continuous.yml #4

Summary

Jobs

Triggered via push 16 minutes ago	Status	Total duration	Artifacts
totollygeek pushed -> 45cb452 main	Failure	-	-



This workflow graph cannot be shown

A graph will be generated the next time this workflow is run.

Annotations

1 error

- .github/workflows/continuous.yml#L46**
You have an error in your yaml syntax on line 46

This workflow graph cannot be shown

A graph will be generated the next time this workflow is run.

Annotations

1 error

- ✘ `.github/workflows/continuous.yml#L46`
You have an error in your yaml syntax on line 46

<> Edit file

Preview changes

Spaces

2

No wrap

```
29   - uses: actions/checkout@v1
30   - name: Cache .nuke/temp, ~/.nuget/packages
31     uses: actions/cache@v2
32     with:
33       path: |
34         .nuke/temp
35         ~/.nuget/packages
36       key: ${{ runner.os }}-${{ hashFiles('global.json', 'source/**/*.csproj') }}
37   - name: Run './build.cmd Test PushPackages'
38     run: ./build.cmd Test PushPackages
39     env:
40       NuGetApiKey: ${{ secrets.NUGET_API_KEY }}
41   - uses: actions/upload-artifact@v1
42     with:
43       name: test-results
44       path: output/test-results
45   - uses: actions/upload-artifact@v1
46     with:
47       name: packages
48       path: output/packages
49 ubuntu-latest:
50   name: ubuntu-latest
```

Use **Control** + **Space** to trigger autocomplete in most situations.

```
env:
  NuGetApiKey: ${ secrets.NUGET_API_KEY }
- uses: actions/upload-artifact@v1
  with:
    name: test-results
    path: output/test-results
- uses: actions/upload-artifact@v1
  with:
    name: packages
    path: output/packages
ubuntu-latest:
```

Dear
YAML,



```
7
8 pool:
9   --name: - 'Tarox3-(2019-1tsc)'
```

```
10
11 variables:
12   --solution: - '**/Scheduler.sln'
```

```
13   --buildPlatform: - 'Any-CPU'
```

```
19 on:
20   push:
21     branches:
22       - main
23
24 jobs:
25   windows-latest:
26     name: windows-latest
```

```
39 build_job:
40   stage: build
41   only:
42     - tags # the build process will only be started by git tag commits
43   script:
44     - '& "$env:NUGET_PATH" restore' # restore Nuget dependencies
45     - '& "$env:MSBUILD_PATH" /p:Configuration=Release' # build the project
46   artifacts:
47     expire_in: 1 week # save gitlab server space, we copy the files we need to deploy folder later on
48     paths:
49     - '$env:EXE_RELEASE_FOLDER\YourApp.exe' # saving exe to copy to deploy folder
```



Dear
YAML,



Debugging

IDE Support

OnPrem

Portability

Customization

Highlighting

The only
problem?

Reusability

Libraries

Extensibility

Build sharing

Learning curve



/nju:k/

1. The **cross-platform** build automation solution for .NET with C# DSL.
2. An approach to **embrace** existing IDE tooling.
3. A state where **everyone in a team** is able to **manage and change** the build.

Matthias
Koch

@matkoch87



@totollygeek



totollygeek

1 commit 31 ++ 23 --

#46

40

20

July

April

2019

October

July

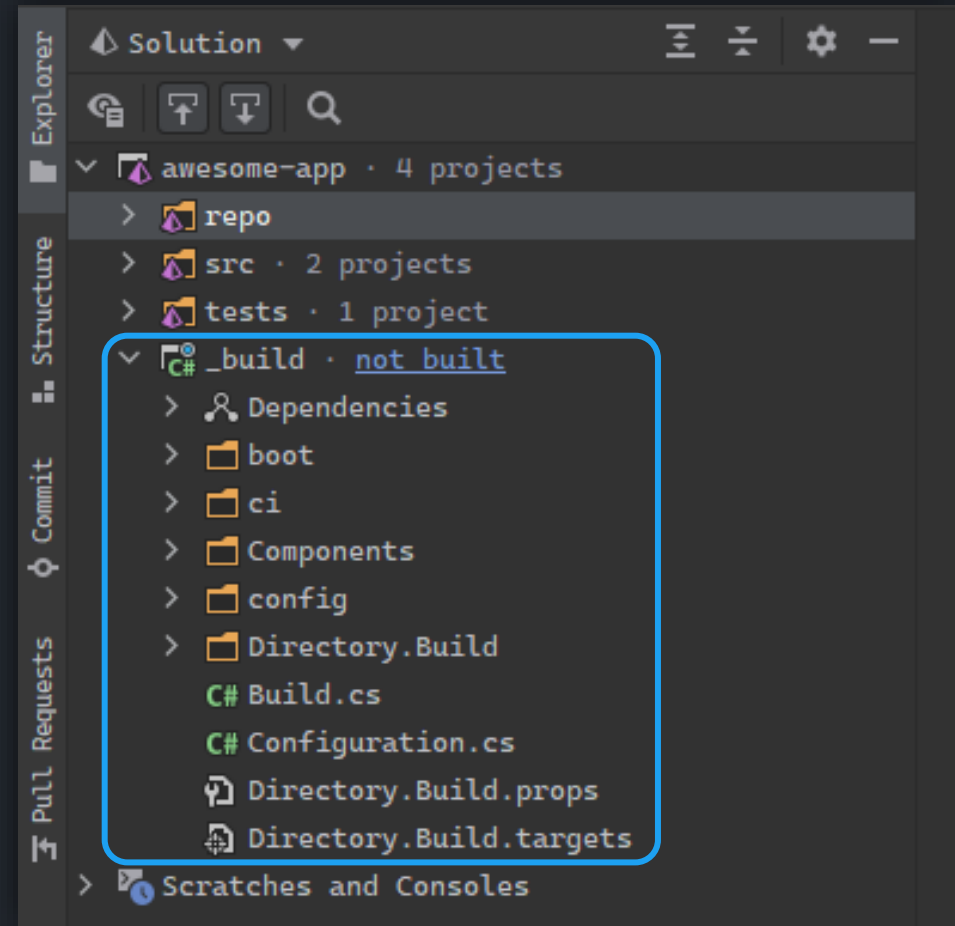
April





What is it?

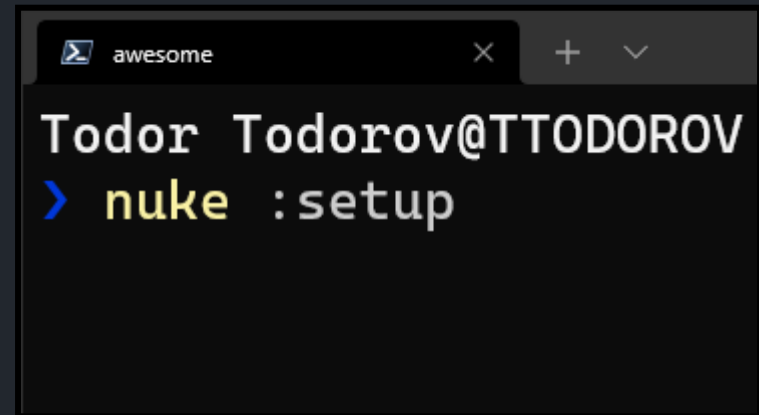
What is it?



But it is also so much more!

- » Global tool
- » Extensive libraries
- » CLI tool support
- » Fluent syntax
- » CI systems integration
- » Path handling
- » Solutions & projects models
- » Slack notifications
- » IDE extensions
- » Build sharing
- » ... and more

Setup & Global Tool

A terminal window titled 'awesome' with window control buttons (close, maximize, minimize) in the top right. The terminal shows the user 'Todor Todorov@TTODOROV' and a prompt '>' followed by the command 'nuke :setup' in yellow text.

```
awesome x + v  
Todor Todorov@TTODOROV  
> nuke :setup
```

Installing a
global tool is
just one
command
away....

```
PowerShell
todor@TCentral C: > git > awesome-app }main ≡
> dotnet tool install Nuke.GlobalTool --global
You can invoke the tool using the following command: nuke
Tool 'nuke.globaltool' (version '5.1.2') was successfully installed.
todor@TCentral C: > git > awesome-app }main ≡
> nuke --help
PowerShell Desktop version 5.1.19041.906
Microsoft (R) .NET Core SDK version 5.0.203

NUKE

NUKE Execution Engine version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)

Targets (with their direct dependencies):

Clean (default)
Restore
Compile          → Restore
Test             → Compile
Pack             → Test
PushPackages (default) → Test, Pack
```

awesome



Todor Todorov@TTODOROV

C:\git\awesome

master

[14:59]

> nuke :setup

Todor Todorov@TTODOROV C:\git\awesome master [14:59]

> nuke :setup

NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)

How should the build project be named?

>> [default: _build]

Todor Todorov@TTODOROV C:\git\awesome master [14:59]

> nuke :setup

NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)

How should the build project be named?

→ _build

Where should the build project be located?

>> [default: ./build]

```
awesome x + v
Todor Todorov@TTODOROV C:\git\awesome master [14:59]
> nuke :setup
NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)
How should the build project be named?
- _build
Where should the build project be located?
- ./build
Which NUKE version should be used?
» 5.1.1 (latest release)
```

```
awesome x + v
Todor Todorov@TTODOROV C:\git\awesome master [14:59]
> nuke :setup
NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)
How should the build project be named?
- _build
Where should the build project be located?
- ./build
Which NUKE version should be used?
- 5.1.1 (latest release)
Which solution should be the default?
>> awesome.sln
None
```



```
awesome x + v
Todor Todorov@TTODOROV C:\git\awesome master [14:59]
> nuke :setup
NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)
How should the build project be named?
- _build
Where should the build project be located?
- ./build
Which NUKE version should be used?
- 5.1.1 (latest release)
Which solution should be the default?
- awesome.sln
Do you need help getting started with a basic build?
>> Yes, get me started!
    No, I can do this myself ...
```

```
awesome x + v - □ ×
> nuke :setup
NUKE Global Tool version 5.1.1 (Windows, .NETCoreApp, Version=v2.1)
How should the build project be named?
→ _build
Where should the build project be located?
→ ./build
Which NUKE version should be used?
→ 5.1.1 (latest release)
Which solution should be the default?
→ awesome.sln
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
» dotnet CLI
  MSBuild/Mono
  Neither
```

```
awesome x + v - □ ×
How should the build project be named?
→ _build
Where should the build project be located?
→ ./build
Which NUKE version should be used?
→ 5.1.1 (latest release)
Which solution should be the default?
→ awesome.sln
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
→ dotnet CLI
Source files are located in ...
  ./source
»  ./src
  Neither
```

```
awesome x + v - □ ×
Where should the build project be located?
→ ./build
Which NUKE version should be used?
→ 5.1.1 (latest release)
Which solution should be the default?
→ awesome.sln
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
→ dotnet CLI
Source files are located in ...
→ ./src
Move packages to ...
» ./output
  ./artifacts
  Neither
```

```
awesome x + v - □ ×
→ ./build
Which NUKE version should be used?
→ 5.1.1 (latest release)
Which solution should be the default?
→ awesome.sln
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
→ dotnet CLI
Source files are located in ...
→ ./src
Move packages to ...
→ ./output
Where do test projects go?
» ./tests
   Same as source
```

```
awesome x + v - □ ×
→ 5.1.1 (latest release)
Which solution should be the default?
→ awesome.sln
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
→ dotnet CLI
Source files are located in ...
→ ./src
Move packages to ...
→ ./output
Where do test projects go?
→ ./tests
Do you use GitVersion?
» Yes, just not setup yet
   No, custom versioning
```

```
awesome x + v - □ ×
Do you need help getting started with a basic build?
→ Yes, get me started!
Restore, compile, pack using ...
→ dotnet CLI
Source files are located in ...
→ ./src
Move packages to ...
→ ./output
Where do test projects go?
→ ./tests
Do you use GitVersion?
→ Yes, just not setup yet
Creating directory 'C:\git\awesome\.nuke' ...
Creating directory 'C:\git\awesome\build' ...
Todor Todorov@TTODOROV C:\git\awesome ↗ master ≠ +2 ~0 -0 | +3
~1 -0 ! [16:07]
>
```

awesome



Todor Todorov@TTODOROV

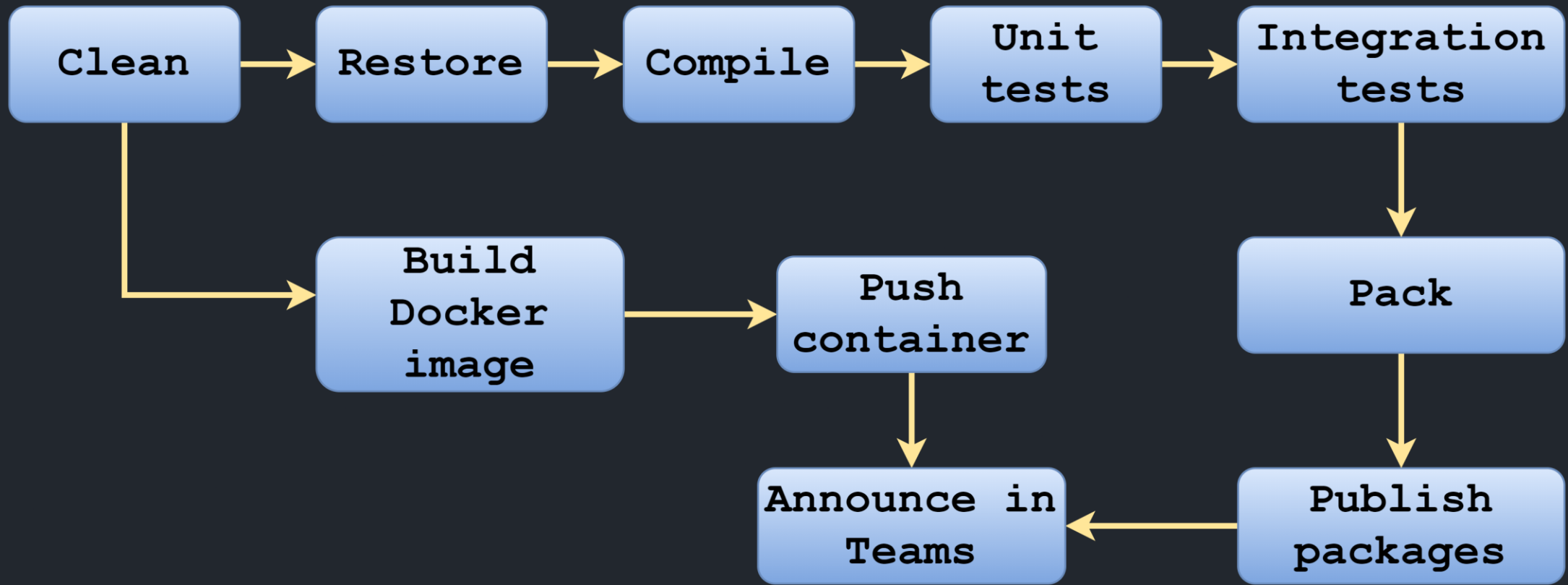
C:\git\awesome

master

[14:54]



Build model tree



Targets

```
Target Restore => _ => _  
    .Executes(() =>  
    {  
        DotNetRestore(_ => _  
            .SetProjectFile(Solution));  
    });
```

```
Target Clean => _ => _
    .Before(Restore)
    .Executes(() =>
    {
        SourceDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        TestsDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        EnsureCleanDirectory(OutputDirectory);
    });
```

```
Target Clean => _ => _
    .Before(Restore)
    .Executes(() =>
    {
        SourceDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        TestsDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        EnsureCleanDirectory(OutputDirectory);
    });

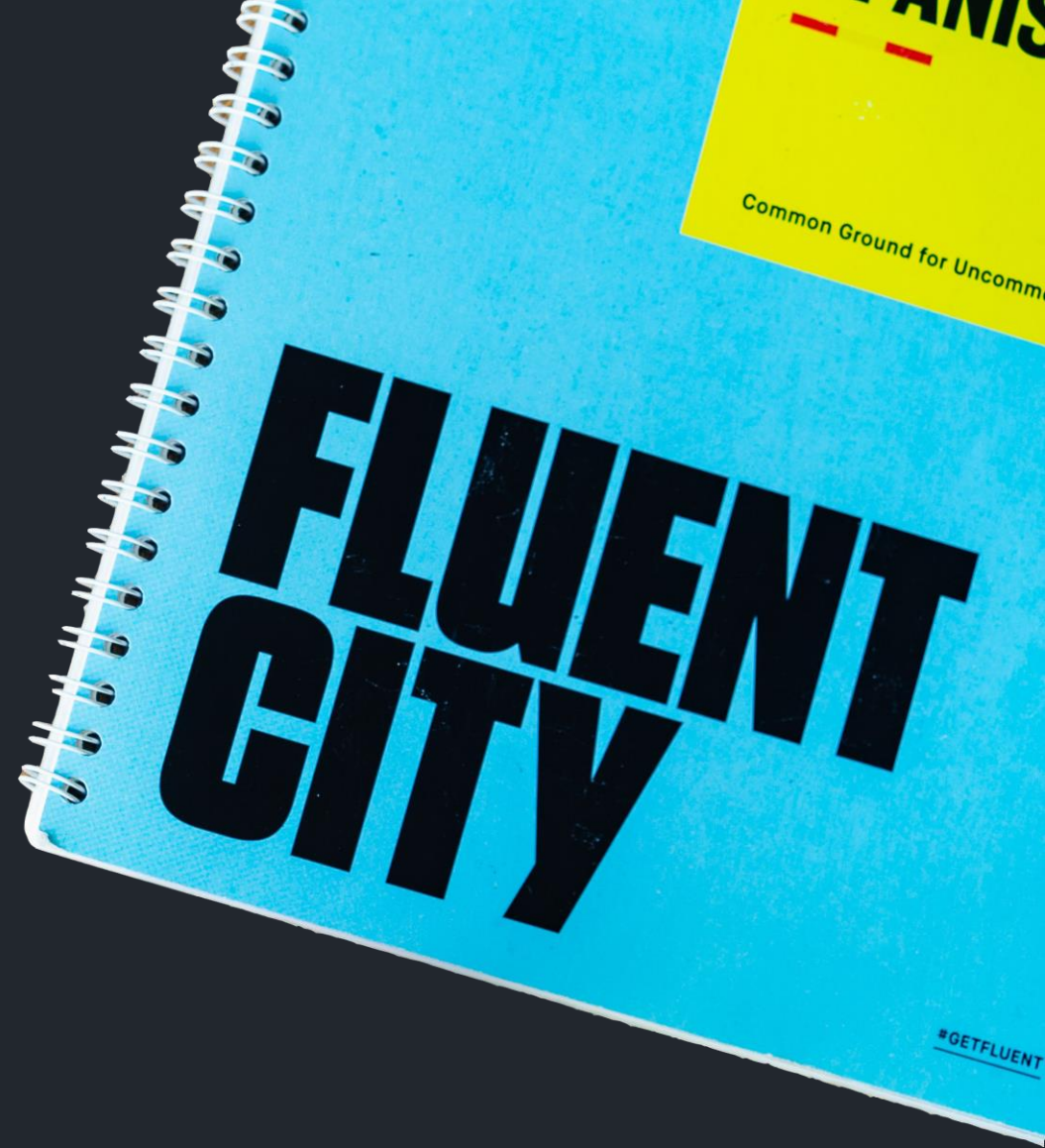
Target Restore => _ => _
    .Executes(() =>
    {
        DotNetRestore(_ => _
            .SetProjectFile(Solution));
    });
```

```
Target Clean => _ => _
    .Before(Restore)
    .Executes(() =>
    {
        SourceDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        TestsDirectory.GlobDirectories("**/bin", "**/obj").ForEach(DeleteDirectory);
        EnsureCleanDirectory(OutputDirectory);
    });

Target Restore => _ => _
    .Executes(() =>
    {
        DotNetRestore(_ => _
            .SetProjectFile(Solution));
    });

Target Compile => _ => _
    .DependsOn(Restore)
    .Executes(() =>
    {
        DotNetBuild(_ => _
            .SetProjectFile(Solution)
            .SetConfiguration(Configuration)
            .SetAssemblyVersion(GitVersion.AssemblySemVer)
            .SetFileVersion(GitVersion.AssemblySemFileVer)
            .SetInformationalVersion(GitVersion.InformationalVersion)
            .EnableNoRestore());
    });
```

Fluent API



FluentInterface

20 December 2005



Martin Fowler

- API DESIGN
- DOMAIN SPECIFIC LANGUAGE

A few months ago I attended a workshop with Eric Evans, and he talked about a certain style of interface which we decided to name a fluent interface. It's not a common style, but one we think should be better known. Probably the best way to describe it is by example.

Source: <https://www.martinfowler.com/bliki/FluentInterface.html>



```
DotNetTest(_ => _
    .SetConfiguration(Configuration)
    .SetNoBuild(SucceededTargets.Contains(Compile))
    .ResetVerbosity()
    .SetResultsDirectory(TestResultDirectory)
    .When(IsServerBuild, _ => _
        .EnableCollectCoverage()
        .SetCoverletOutputFormat(CoverletOutputFormat.cobertura)
        .SetExcludeByFile("*.Generated.cs")
        .SetCoverletOutputFormat(
            $"\\\\"{CoverletOutputFormat.cobertura},{CoverletOutputFormat.json}\\\\"")
        .EnableUseSourceLink())
    .CombineWith(TestProjects, (_, p) => _
        .SetProjectFile(p)
        .SetLogger($"{{logger}};LogFileName={{p.Name}}.{{logger}}")),
    completeOnFailure: true);
```


Fluent tasks convert to CLI call

```
DotNetRestore(_ => _  
    .SetProjectFile(Solution));
```



```
> "C:\Program Files\dotnet\dotnet.exe" restore C:\git\awesome-app\awesome-app.sln
```

CLI Tools



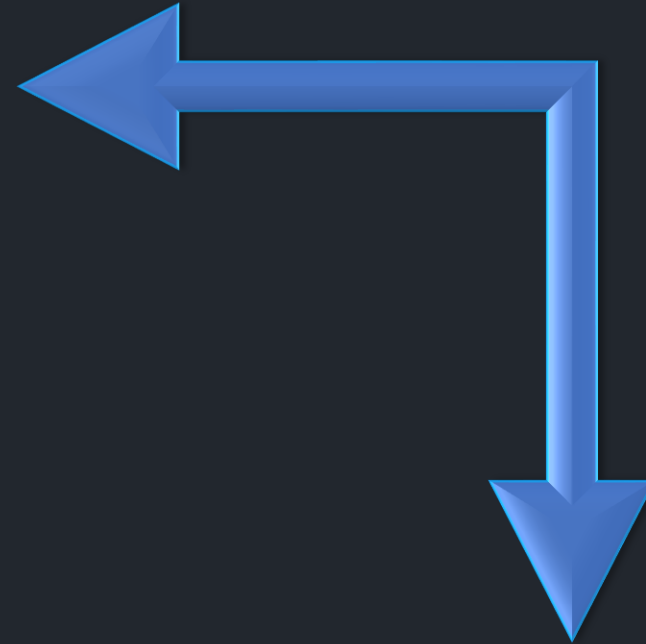
DotCover GitVersion DotMemory SignTool SonarScanner
SpecFlow Npm
DotNet VSWhere Paket
MSBuild Helm DocFX GitHub
Docker Unity Squirrel OpenCover
Xunit NuGet Git
Kubernetes
ReSharper Coverlet Octopus ReportGenerator VSTest

```
Todor Todorov
Target BuildDockerImage => _ => _
  .Executes(() =>
  {
    DockerBuild(configurator: _ => _
      .SetFile(Dockerfile));
  });
```

[Pure]
public static T SetFile<T>(this T toolSettings, string file)
in class DockerBuildSettingsExtensions

T is DockerBuildSettings

Sets File
Name of the Dockerfile (Default is 'PATH/Dockerfile').
[`DockerBuildSettingsExtensions.SetFile` on google.com](#)



🏠 / Reference / Command-line reference / Docker CLI (docker) / docker build

docker attach	--cpuset-mems	MEMs in which to allow execution (0-3, 0,1)
docker build	--disable-content-trust true	Skip image verification
docker builder	--file, -f	Name of the Dockerfile (Default is 'PATH/Dockerfile')
docker buildx	--force-rm	Always remove intermediate containers
docker checkpoint	--iidfile	Write the image ID to the file

Working with other executables

```
    // This will locate it in PATH environment variable
    [PathExecutable]
    readonly Tool Docker;

    // This will locate it via absolute/relative path
    [LocalExecutable("./tools/confd.exe")]
    readonly Tool Confd;

    // This will get it from a NuGet package
    [PackageExecutable("ReportGenerator", "ReportGenerator.exe")]
    readonly Tool ReportGenerator;
```

Working with other executables

```
var imageName = "alpine";  
var command = "sh";  
Docker($"run -it {imageName} {command}");  
// This will run: "docker run -it alpine sh"
```

CI Systems Integration

TeamCity
GitHub Actions
Azure Pipelines
AppVeyor

Prepare
yourself...




```

[GitHubActions(
    "continuous",
    GitHubActionsImage.WindowsLatest,
    GitHubActionsImage.UbuntuLatest,
    GitHubActionsImage.MacOsLatest,
    OnPushBranches = new[] { MainBranch },
    ImportSecrets = new[] { "NuGetApiKey" },
    PublishArtifacts = true,
    InvokedTargets = new[] { nameof(Test), nameof(PushPackages) },
    CacheKeyFiles = new[] { "global.json", "source/**/*.csproj" })]
[CheckBuildProjectConfigurations]
[ShutdownDotNetAfterServerBuild]
class Build : NukeBuild, IHaveGit
{

```

```
Solution ▾
  awesome-app · 4 projects
    repo
      .gitattributes
      .gitignore
      continuous.yml
      Directory.build.props
      GitVersion.yml
      global.json
      LICENSE
      NuGet.config
      README.md
    src · 2 projects
    tests · 1 project
    _build · not built
  Scratches and Consoles

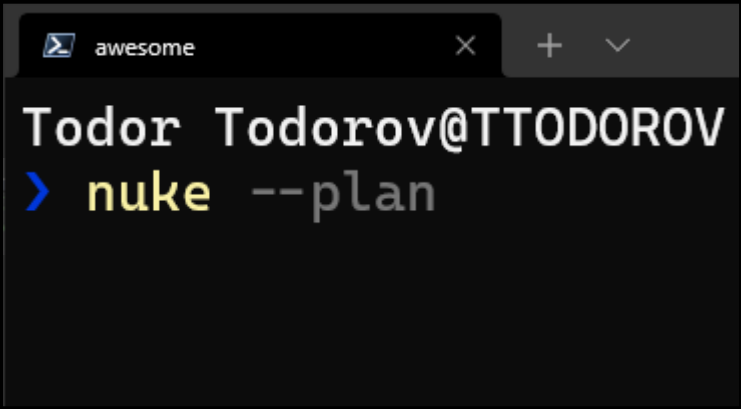
1 # -----
2 # <auto-generated>
3 #
4 #   This code was generated.
5 #
6 #   - To turn off auto-generation set:
7 #
8 #     [GitHubActions (AutoGenerate = false)]
9 #
10 #   - To trigger manual generation invoke:
11 #
12 #     nuke --generate-configuration GitHubActions_continuous --host GitHubActions
13 #
14 # </auto-generated>
15 # -----
16
17 name: continuous
18
19 on:
20   push:
21     branches:
22       - main
23
24 jobs:
25   windows-latest:
26     name: windows-latest
27     runs-on: windows-latest
28     steps:
29       - uses: actions/checkout@v1
30       - name: Cache_nuke/temp_~/nuget/packages

Document 1/1 > jobs: > windows-latest: > steps: > Item 5/5 > with:
```

DEMO



View your build tree

A terminal window titled 'awesome' with window control buttons (close, maximize, minimize) in the top right. The terminal shows the user 'Todor Todorov@TTODOROV' and a prompt '>' followed by the command 'nuke --plan' in yellow text.

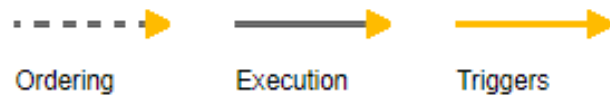
```
awesome x + v  
Todor Todorov@TTODOROV  
> nuke --plan
```

Execution Plan

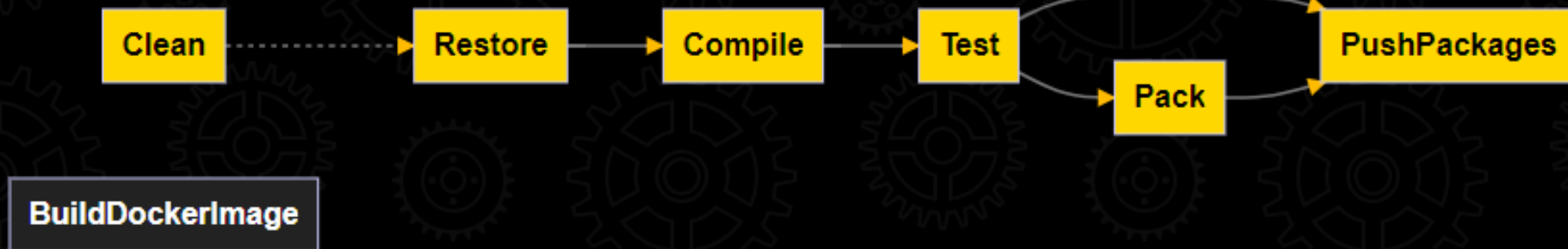
Hovering a target highlights its associated execution plan.



Target edges denote how they are related to each other.



[More information](#)

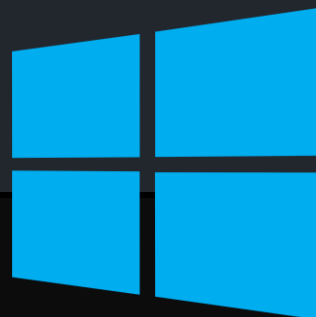


Division
operator
magic

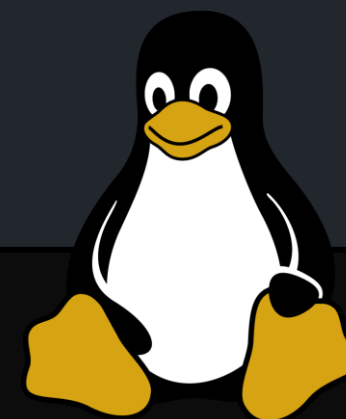




```
AbsolutePath SourceDirectory => RootDirectory / "src";  
AbsolutePath TestsDirectory => RootDirectory / "tests";  
AbsolutePath OutputDirectory => RootDirectory / "output";  
AbsolutePath TestResultDirectory => OutputDirectory / "test-results";  
AbsolutePath PackagesDirectory => OutputDirectory / "packages";  
AbsolutePath Dockerfile => SourceDirectory / "awesome.app" / "Dockerfile";
```



```
RootDirectory:    C:\git\awesome-app
TestsDirectory:  C:\git\awesome-app\tests
OutputDirectory: C:\git\awesome-app\output
Dockerfile:      C:\git\awesome-app\src\awesome.app\Dockerfile
```



```
RootDirectory:    /home/totollygeek/git/awesome-app
TestsDirectory:  /home/totollygeek/git/awesome-app/tests
OutputDirectory: /home/totollygeek/git/awesome-app/output
Dockerfile:      /home/totollygeek/git/awesome-app/src/awesome.app/Dockerfile
```


IDE Extensions



NUKE Support

Maintainers of NUKE 2021.1.0

Enabled for all projects

NUKE Support

This extension adds several features related to the NUKE build automation system:

DEMO



Together, We Create!



<https://github.com/nuke-build/nuke>

@totollygeek

Thank you! (●'∩'●)

Where to find me:

blog:

todorov.bg

twitter:

twitter.com/totollygeek

linkedin:

linkedin.com/in/totollygeek

github:

github.com/totollygeek



Image sources: rawpixel  Unsplash pixabay  Pexels

@totollygeek