Nuke

Что такое Nuke

- Не система сборки
- Автономный автоматизатор процессов

Альтернативы

- Psake – powershell 30.05.2008
- Fake – F# 08.02.2010?
- Cake – C# DSL 27.05.2014

*указаны даты первых коммитов

Скелет Nuke

Workflow движок – рантайм и язык описания

Program.Main()

```
class Build : NukeBuild
{
    public static int Main () =>
        Execute<Build>(x => x.Compile);
...
```

Target

```
public static int Main () => Execute<Build>(x => x.Compile);

Target Clean => _ => _
    .Before(Restore)
    .Executes(() =>
    {
    });

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

Target Compile => _ => _
    .DependsOn(Restore)
    .Executes(() =>
    {
    });
```

Target

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

public delegate ITargetDefinition Target(ITargetDefinition definition);
```

Описание Target

```
interface ITargetDefinition
```

- DependsOn
- Before
- After
- …

```
Target Compile => _ => _
    .DependsOn(Restore)
    .Executes(() =>
    {
        Serilog.Log.Information("Hey!");
    });
```

Before и After

```
Target Clean => _ => _
    .Before(Restore)
    .Executes(() =>
    {
    });

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

А что ещё?

- Execute
- Triggers
- OnlyWhenStatic
- OnlyWhenDynamic
- AssuredAfterFailure
- and moar…

Мясо

Хэлперы, врапперы и всё-всё-всё

Параметры

```
[Parameter]
[Secret]
readonly string NugetApiKey;


Target Restore => _ => _
    .Requires(() => NugetApiKey)
    .Executes(() =>
    {
    });
```

Как сделать dotnet build?

```
using Nuke.Common.ProjectModel;
using static Nuke.Common.Tools.DotNet.DotNetTasks;


[Solution(GenerateProjects = true)]
Solution Solution;

Target Compile => _ => _
   .DependsOn(Restore)
   .Executes(() =>
   {
      DotNetBuild(settings => settings
         .SetConfiguration(Configuration)
         .SetProjectFile(Solution.Demo));
   });
```

Поддерживаемые утилиты

Генерация кода

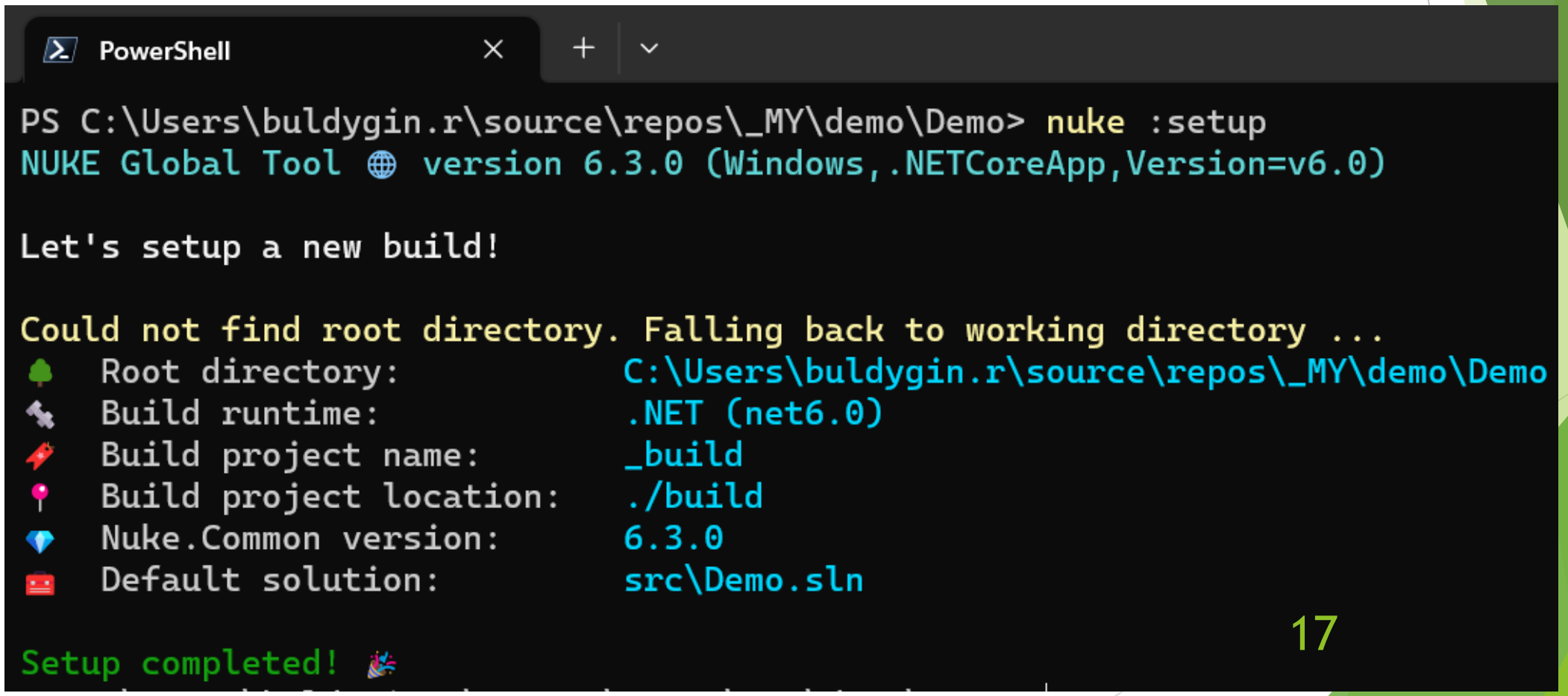

```
{
 "$schema": "https://raw.githubusercontent.com/nuke-build/nuke/master/source/Nuke.Tooling.Generator/schema.json",
 "name": "DotNet",
 "officialUrl": "https://docs.microsoft.com/en-us/dotnet/core/tools/",
 "pathExecutable": "dotnet",
 "customLogger": true,
 "tasks": [
  {
   "help": "<p>The <c>dotnet build</c> command builds the project and its dependencies into a set of binaries...",
   "postfix": "Build",
   "commonPropertySets": [
    "restore"
   ],
   "definiteArgument": "build",
   "settingsClass": {
    "properties": [
     {
      "name": "ProjectFile",
      "type": "string",
      "format": "{value}",
      "createOverload": true,
      "help": "The project file to build. If a project file is not specified, MSBuild searches the current working directory for a file that has a file extension that ends in proj and uses that file."
     },
     {
      "name": "Configuration",
      "type": "string",
      "format": "--configuration {value}",
      "help": "Defines the build configuration. If omitted, the build configuration defaults to <c>Debug</c>. Use <c>Release</c> build a Release configuration."
     },
```

Рюшечки

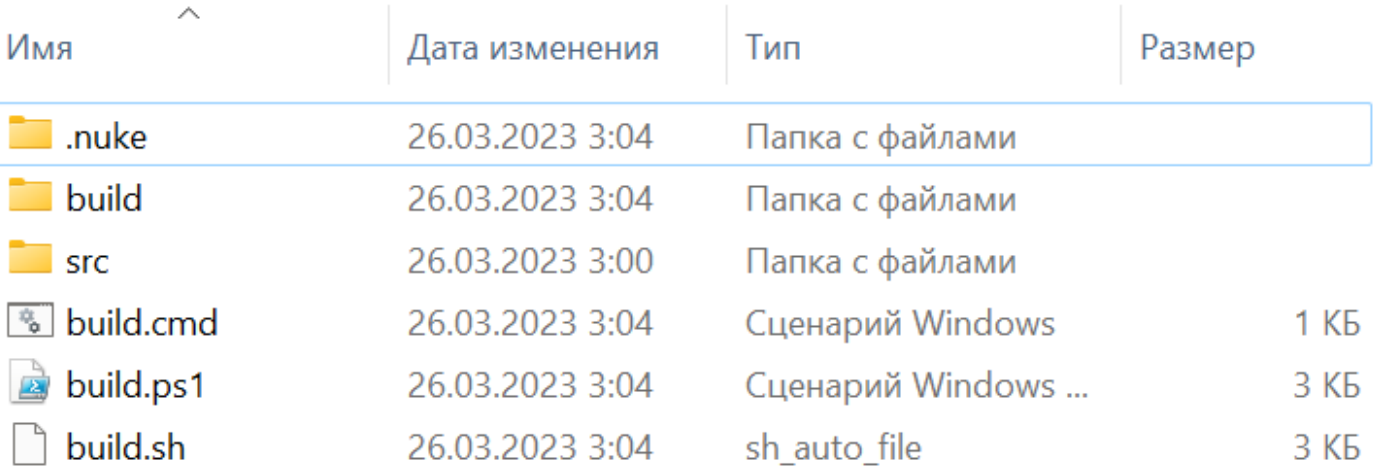
Как nuke делает жизнь проще

Tool и установка

- dotnet tool install Nuke.GlobalTool --global
- nuke :setup



Tool и установка



Имя	Дата изменения	Тип	Размер
.nuke	26.03.2023 3:04	Папка с файлами	
build	26.03.2023 3:04	Папка с файлами	
src	26.03.2023 3:00	Папка с файлами	
build.cmd	26.03.2023 3:04	Сценарий Windows	1 КБ
build.ps1	26.03.2023 3:04	Сценарий Windows ...	3 КБ
build.sh	26.03.2023 3:04	sh_auto_file	3 КБ

```
:; set -eo pipefail
:; SCRIPT_DIR=$(cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd)
:; ${SCRIPT_DIR}/build.sh "$@"
:; exit $?

@ECHO OFF
powershell -ExecutionPolicy ByPass -NoProfile -File "%~dp0build.ps1" %*
```

Логирование

- Serilog one love
- Уровни «разговорчивости» **nuke --verbosity xxx:**

  verbose, normal, minimal, quiet

```
Target Compile => _ => _
    .DependsOn(Restore)
    .Executes(() =>
    {
        Serilog.Log.Information("Hey!");
    });
```

Интеграции с серверами сборки

- **AppVeyor**
- **Azure Pipelines**
- Bitbucket
- **GitHub Actions**
- GitLab
- Jenkins
- JetBrains Space
- **TeamCity**

Тестировани

- Скрипт сборки – код
- Тестирование кода можно автоматизировать

Расшаривание пайплайнов

- .NET global tool
- Отдельные target в nuget пакетах

Пример из жизни

https://github.com/OpenHD/OpenHD-WebUI

https://github.com/buldo/RtpToWebRtcRestreamer

Выводы / IMHO

- Когда использовать:
  - Разработчик == DevOps
  - Проект с уникальной кучерявой сборкой
  - Хочется автоматизации, не хочется изучать очередной сервер сборки
  - Необходимость защиты от vendor-lock
- Когда подумать:
  - Всё и так отлично работает
  - Есть отличная DevOps команда
  - Вы хотите использовать специфичный функционал текущей системы сборки