PSCONF.EU

PowerShell Conference Europe

New assertions in Pester

Jakub Jareš

@nohwnd

Antwerp 24

PSCONF.EU

Antwerp24

Many thanks to our sponsors:

Chocolatey

PATCH MY PC

semperis

ScriptRunner®
Your #1 for PowerShell Automation

@nohwnd

Jakub Jareš

Pester and Profiler owner and maintainer.

Senior software engineer, developing VSTest, Testing Platform and MSTest at Microsoft. All opinions are mine.

Please consider sponsoring my open-source development:

Sponsor @nohwnd on GitHub Sponsors

@nohwnd
@pspester
me@jakubjares.com

𝕏 @nohwnd

Frode in the house!

- Maintains pester and pester/docs
- 200 PRs created
- Many many issues
  and discussions solved!

github/fflaten
twitter/FrodeFlaten

Frode Flaten
fflaten · he/him

Pester 6 - alpha

New syntax:

`Should-Be`

No space.
Side-by-side with `Should -Be`.

@nohwnd

Why?

Every release of Pester needs new Assert syntax 😁

```
$true.Should.Be($false)    - v1, v2
$true | Should Be $false   - v3
$true | Should -Be $false  - v4, v5
$true | Should-Be $false   - v6
```

Why?

- You can have only 32 parameter sets in PowerShell.

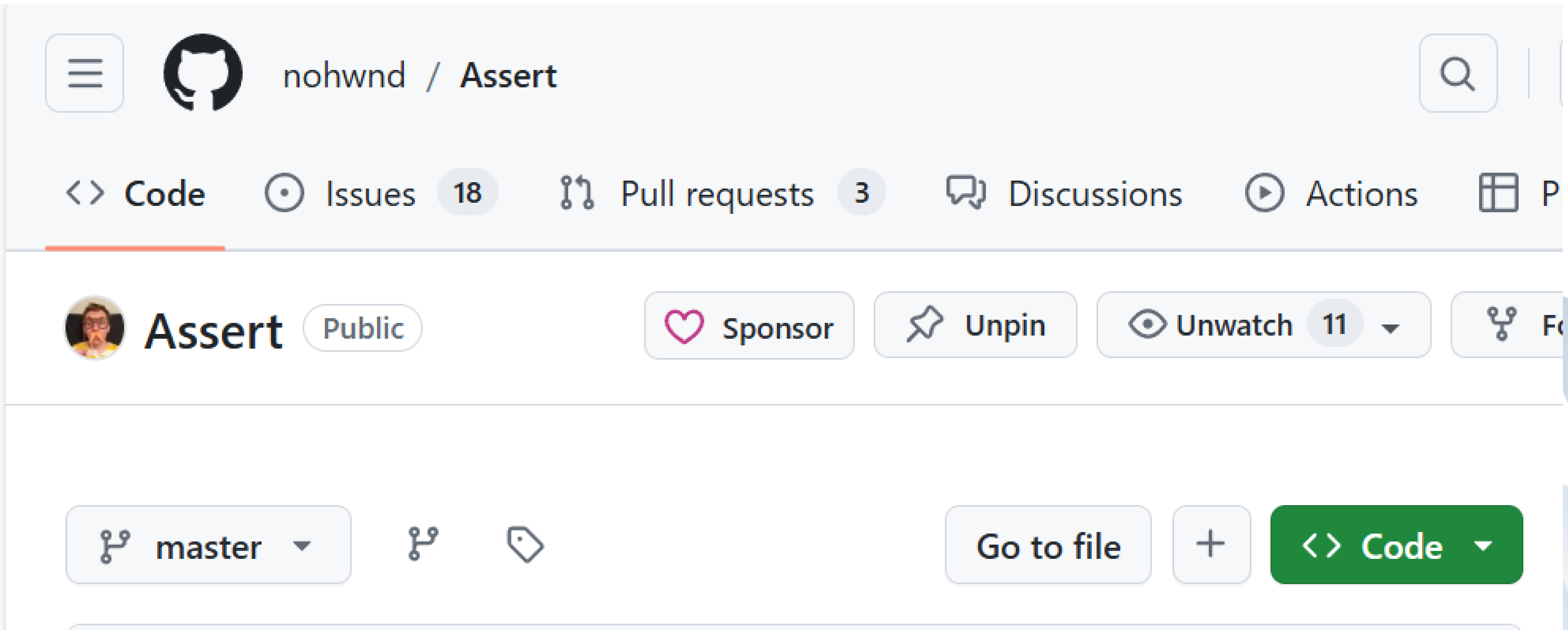
```
Should -Be
0000 0000 0000 0000 0000 0000 0000 0001
```

- Current Should has a lot of problems with consistency, array comparisons etc.

@nohwnd

Based on Assert module

- Assert was fixed for PowerShell 7.
- Assert is now maintenance only.



Assertion categories

- Value assertions
  - generic
  - type specific
- Collection assertions
  - generic
  - combinator
- Equivalency

Value assertions

Accept single value for $Expected

Unwrap one-item arrays received through pipeline

🟢
```
1 | Should-Be -Expected 1
```
🟢
```
@(1) | Should-Be -Expected 1
```
🟢
```
Should-Be -Actual 1 -Expected 1
```
❌
```
Should-Be -Actual @(1) -Expected 1
```

@nohwnd

Generic value assertions

`Should-Be`

Accept multiple types of input data.

Convert `$Actual` value to the type of `$Expected`.

Act very similarly to PowerShell operators.

PSCONF.EU Antwerp24

DEMO 01

@nohwnd

Generic value assertions

- `$true | Should-Be $true`
- `1 | Should-Be $true`
- `"false" | Should-Be $true`

Assertion failure

short type

value

```
Expected [int] 1,
but got [Object[]] @(1, 2).
```

array

null "type"

```
Expected [null] $null,
but got [string] '$null'.
```

Type specific value assertions

```
Should-BeTrue
Should-BeString
```

Accept single value.

Don't cast the input to the `$Expected` type.

Provide more options specific to the type of data.

PSCONF.EU

Antwerp24

DEMO 02

@nohwnd

Type specific value assertions

🟢 `$true | Should-BeTrue`
❌ `1 | Should-BeTrue`
❌ `"false" | Should-BeTrue`
❌ `$null | Should -BeTrue`

Collection assertions

Accept array for $Expected

Keep one-item arrays received through pipeline

🟢 `@(1) | Should-BeCollection -Expected @(1)`

🟢 `1 | Should-BeCollection -Expected @(1)`

🟢 `Should-BeCollection @(1) -Expected @(1)`

❌ `Should-BeCollection 1 -Expected @(1)`

Generic collection assertions

```
Should-BeCollection
Should-ContainCollection
```

Accept multiple type of input data.

Convert `$Actual` items to the type of `$Expected` item.

Act similarly to PowerShell operators.

@nohwnd

PSCONF.EU Antwerp24

DEMO 03

@nohwnd

Collection combinator assertions

`Should-Any`
`Should-All`

Accept multiple type of input data.

Convert `$Actual` items to the type of `$Expected` item.

Take a scriptblock as a filter (predicate).

PSCONF.EU
Antwerp24

DEMO 04

@nohwnd

Collection combinator assertions

🟢 `@(1, 2) | Should-Any { $_ -eq 1 }`

❌ `@(1, 2) | Should-All { $_ -eq 1 }`

❌ `@(1, 2) | Should-All { $_ | Should-Be 1 }`

Equivalency

`Should-BeEquivalent`

Compare deeper object structures for equivalency or equality.

PSCONF.EU

Antwerp24

DEMO 05

@nohwnd

Time assertions, fluent syntax

```
$file.LastWriteTime |
    Should-BeAfter 1week -Ago

{ Start-Sleep -Second 1 }
        | Should-BeFasterThan 1s
```

ms/mil, s, m, h, d, w

1min, 2mins, 1minute, 2minutes, 2minuetes

PSCONF.EU

Antwerp24

DEMO 06

@nohwnd

Negating assertions

```
Should-Be
Should-NotBe
```

Not every assertion has Not:

```
Should-BeGreaterThan
Should-BeLessThanOrEqual
```

Migration

Not yet. I got ahead of myself. 😰

If you do migrate, you can disable the old assertions by:

```
$c = New-PesterConfiguration
$c.Should.DisableV5 = $true
```

Q&A

15 minutes

@nohwnd