DuPSUG

Testing in PowerShell

Unit testing, integration tests and Operational testing

Jaap Brasser

@Jaap_Brasser

About_Jaap

- Dutch PowerShell User Group (!!!)
- Blogging
  - PowerShell Magazine
  - JaapBrasser.com
- Slack
- Reddit
- GitHub
- PowerShell Gallery
- TechNet Forums/Gallery

@Jaap_Brasser

Agenda

- Why testing and what are some benefits
- Basic concepts of Pester
- Writing Unit tests
- Writing Integration tests
- Operational testing

Why testing and what are some benefits

Reasons for testing

- Provide a framework for updating scripts
- Write integration tests to verify scripting functionality
- Verify operational functionality
- Analyze and improve scripting habits

What is Pester

Pester provides a framework for **running unit tests to execute and validate PowerShell commands from within PowerShell**. Pester consists of a simple set of functions that expose a testing domain-specific language (DSL) for isolating, running, evaluating and reporting the results of PowerShell commands.

Advantages of Pester

- Is shipped with Windows
- Used by the PowerShell team
- Works in familiar notation
- Actively maintained and updated on GitHub

Basic concepts of Pester

- Describe
- Context
- It
- Should
- BeforeEach
- AfterEach
- In

- New-Fixture
- Invoke-Pester
- New-PesterOption
- Mock
- InModuleScope
- Assert-MockCalled
- Assert-VerifiableMocks

Should operators

- Different syntax when compared to PowerShell
- Different naming from the operators in PowerShell
- Mostly similar results when compared to PowerShell operators

Demo 1 – Successful Tests

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- How to create a simple pester test
- How the Pester Syntax differs from regular PowerShell

Demo 2 – Failed Tests

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- What happens if a Pester Test fails
- Error messages for curly brace on next line

Demo 3 – Mocking and ModuleScope

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- How to mock a function or module
- Mock the smallest component
- How to mock when working with modules

Demo 4 – Pester TestDrive:\

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- View how the TestDrive: is initialized
- Use for temporary storage in the current Pester Scope
- How to use this in your testing

Demo 5 – Pester output

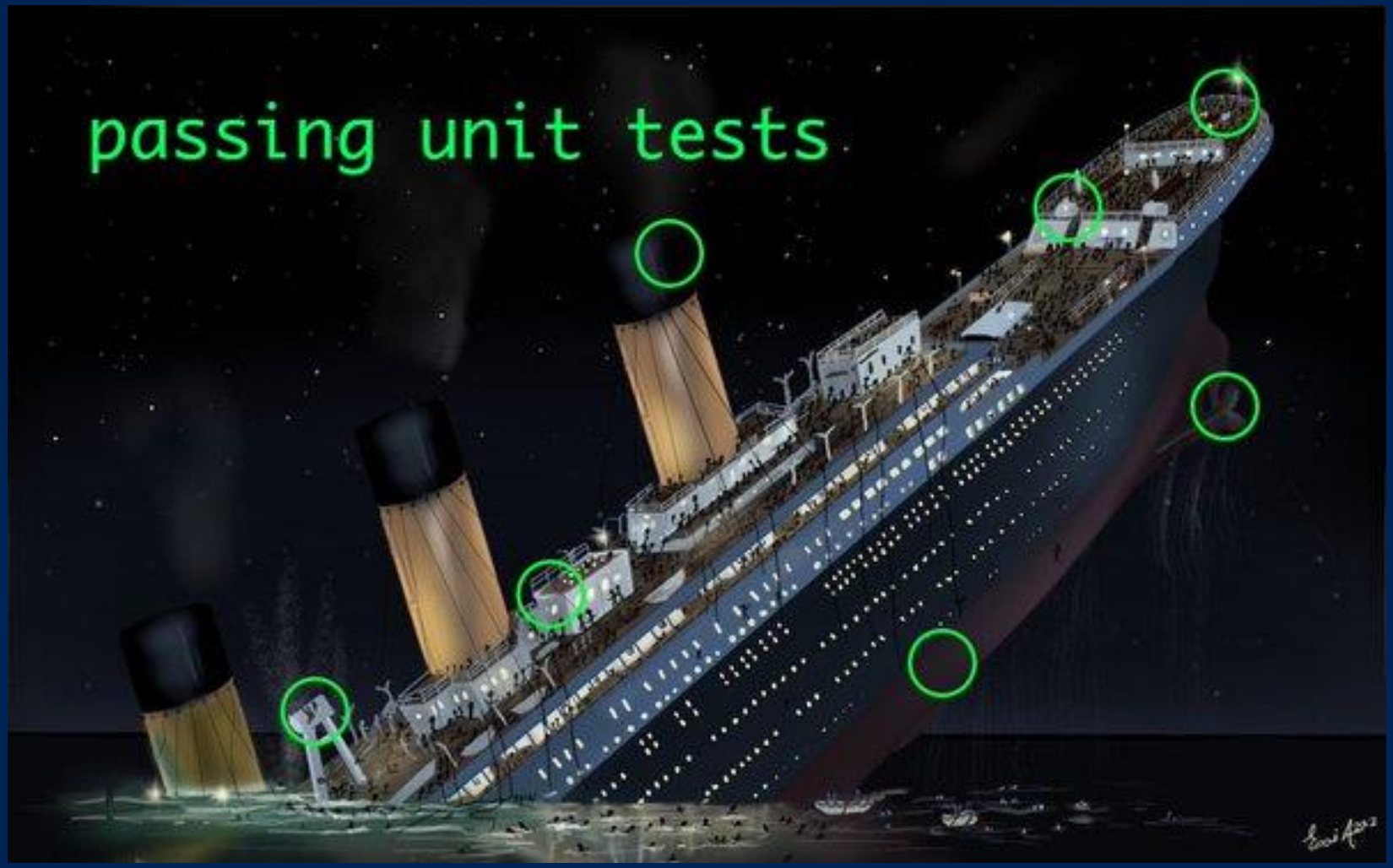
```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- Run Pester using Invoke-Pester
- Define different output type
- Access the different test results using the XML output
- Useful for automating your testing

Write unit tests



Why unit tests

- A way of testing components in your scripts and functions
- Can be fully automated
- Tests a single component
- Hard to write (when inexperienced)

Demo Unit Testing

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- Function that exports dates based on strings
- Singled out a component to test
- Verified that breaking code intentionally was seen in the tests

Write integration tests

- Used to test functionality of a function or collection of functions
- Test functionality of a script

Demo – Integration Test

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo Summary

- Run the entire function
- Test various aspects of the function
  - Is there output
  - Type of output

Operational testing

- Test your infrastructure
- Verify changes in infrastructure
- Useful in the following scenarios:
  - Migrations
  - New deployments
  - When incidents occur
  - Automatically verify critical systems

Demo 1 – Test installed components

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo summary

- Verified that required software is installed
- Verified version of PowerShell
- Verified various registry settings
- Added PowerShell to Win+X

Demo 2 – Test Fileshares

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo summary

- Show content of the CliXml file
- Show similar results
- Use Pester tests
- Investigated the differences

Demo 3 – Linux testing

```
1  Describe 'This is my Pester test' {
2      Context 'Certainly nothing will fail' {
3          It 'This almost never fails' {
4              42.0001 | Should Be 42
5          }
6          It 'Simple text comparison' {
7              'Hello' | Should Match 'World'
8          }
9          It 'Compare two arrays' {
10             @(1,2) | Should BeExactly @(1,2,3)
11         }
12     }
13 }
```

```
Describing This is my Pester test
   Context Certainly nothing will fail
    [-] This almost never fails 76ms
      Expected: {42}
      But was:  {42.0001}
      4:              42.0001 | Should Be 42
      at <ScriptBlock>, <No file>: line 4
    [-] Simple text comparison 69ms
      Expected: {Hello} to match the expression {World}
      7:              'Hello' | Should Match 'World'
      at <ScriptBlock>, <No file>: line 7
    [+] Compare two arrays 28ms
```

Demo summary

- Run PowerShell scripts / Pester tests on Linux
- Verify software installation on Linux
- Check PowerShell version and edition

Questions?