Deliver PowerShell to your team

Bartek Bielawski & Daniël Both

Experts Live 2016

OFFICE 365

Daniël Both & Bartek Bielawski

Windows Engineers @ Optiver

Goals

- Be able to see who changed what when
- Enforce consistent use of parameters, help, credentials, etc.
- Make sure the code works
- Enable easy discovery of your functions
- Everybody uses the correct version of a script

Current situation?

- Scripts on a UNC share
- Hard to track changes
- Low or no quality control

Step 1: Convert to module

- Functions available locally in memory
- Enables discoverability

Step 2: Create a git repository

- Single source of truth
- Starting point for all automation
- Use what's already there

Step 3: Using Git

- Git branch: Your own place to edit code
- Create pull request from this branch
- Peer review on pull request to save the day
- Keeping track of pull requests

Step 4: PowerShell Gallery

- Enable easy discovery of your functions.
- Updating versions is now much easier!

Demo 1-4

Step 5: Automated distribution

- Use a build tool, it's build to do the job.

Step 6: Test your code

- For new code, this should be step #0
- Enforce code quality
- Make sure your code works

Step 7: Auto update modules

- Enforce use of the correct version
- Filter when an update is triggered

Demo 5-7

Goals achieved?

- Git repository to see who changed what when.
- Pester to enforce consistent use of parameters, help, credentials, etc. and to make sure our code works.
- Bamboo for automating testing and publishing of code.
- PowerShell Gallery to enable easy discovery of your functions.
- Module auto update to enforce the use of the correct version.

optiver

- We’re looking for colleagues, so come have a chat if you’re interested!
- Or browse to:

http://optiver.com/amsterdam

<Volgende sessie 13:30 – 14:30 uur>

Leverage Azure to power your SAP workloads on SuSe

Orhan Alici

Experts Live 2016