PowerShell Conference Europe

Basic Toolmaking: robust scripting for unattended execution

Evgenij Smirnov

VIENNA

20 - 23 June 2022

PSCONF.EU

PowerShell Conference Europe

Basic Toolmaking: robust scripting for unattended execution

Evgenij Smirnov

VIENNA

20 - 23 June 2022

Many thanks to our sponsors:

SquaredUp

au2mator

PATCH MY PC

Chocolatey

System Frontier

ScriptRunner®

```
PS> Get-SpeakerBio -Brief
FullName : Evgenij Smirnov
YearsInIT: 30
Location : Berlin
Employer : SVA GmbH
JobTitle : {Consultant,Architect}
UserGroup: {PSUGB,WSUG-B,EXUSG,VMUGB}
SpeakerAt: {PSConfEU,PSDAY.UK,CIM,…}
Twitter  : @cj_berlin
BlogURL  : https://it-pro-berlin.de
```

Microsoft® MVP Most Valuable Professional

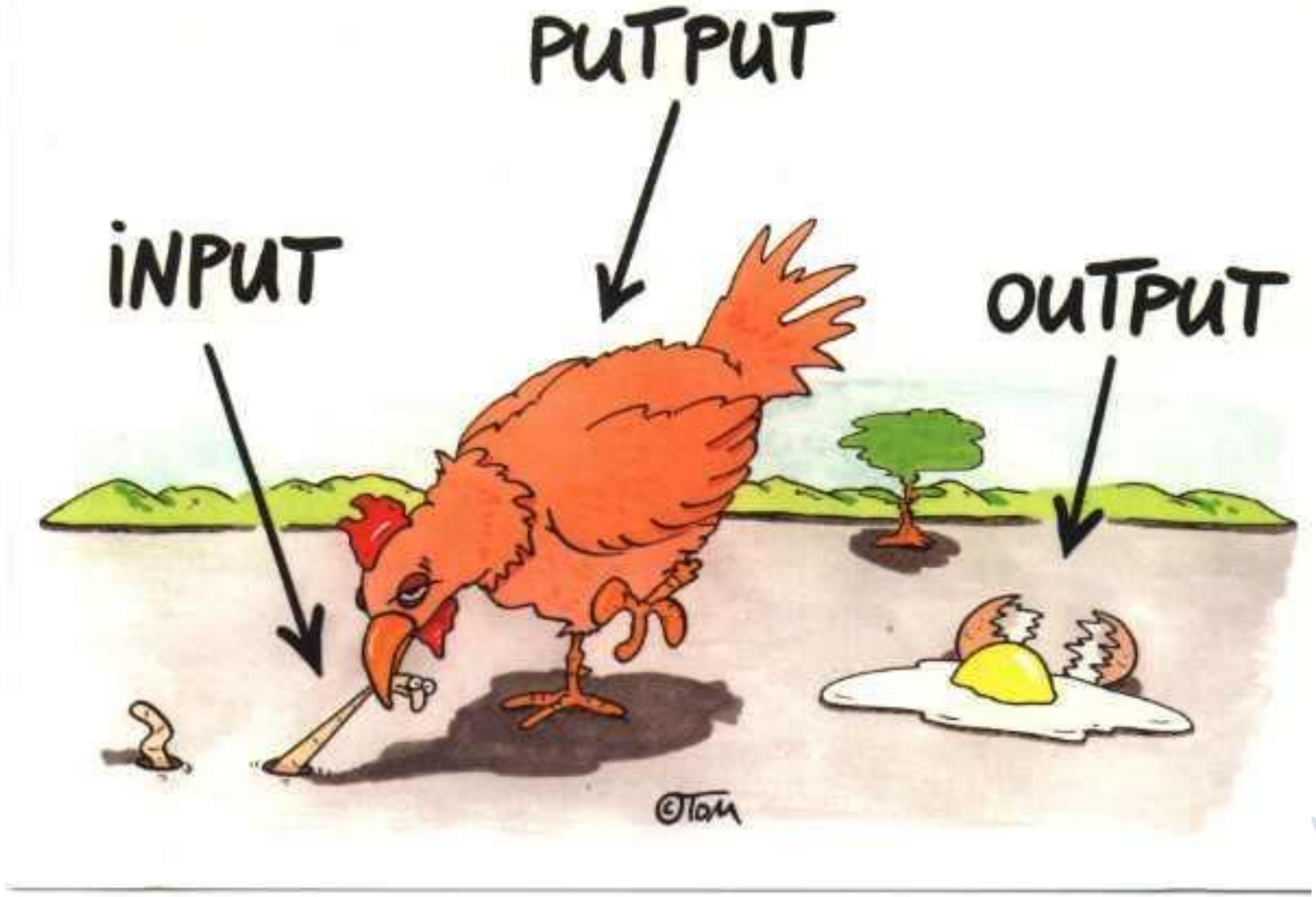
@cj_berlin

In this session

- Why is unattended execution special?
- Strategies to avoid failure
- Strategies to maximise success
- Monitoring unattended execution

„Unattended“ use cases

- Task scheduler | Logon/Logoff | Startup/Shutdown
- Orchestration engine (au2mator, ScriptRunner etc.)
- PowerAutomate flow, GitHub action
- Monitoring agent
- IDM/IAM supplemental actions
- Backup pre-/post-execution
- SCCM detection action or configuration baseline


PUTPUT
INPUT
OUTPUT
©TOM

Things outside your control

- Version, edition and language of the OS…
- Underlying hardware…
- Security context executing the script…
- PowerShell host (powershell.exe, pwsh.exe, custom)…
- Modules installed/loaded/available…
- Connectivity to required resources…
- Early abort by the invoking facility…

Separation of responsibilities

Because Ops will take care
of everything BUT your script…

…your script has to take care of itself

Mission objective

provide a meaningful result under any circumstances

What does meaningful mean?

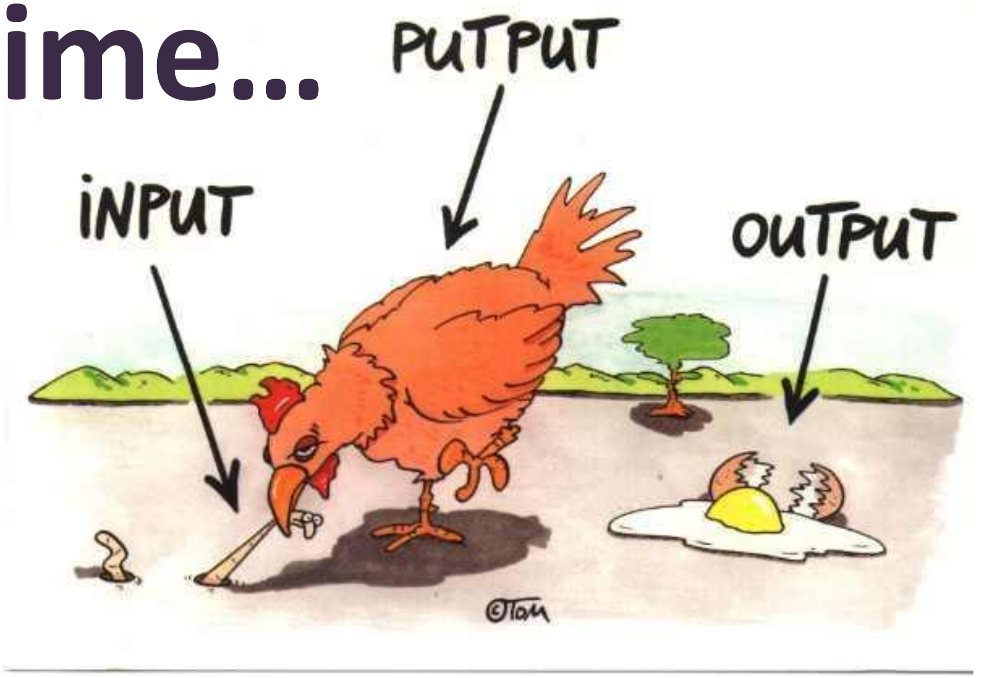
If everything works as expected

- Provide a certain RC
- Provide output in a certain format…
- …or no output at all!
- Clean up after yourself

If something goes wrong

- No unfinished business!
- Provide an error RC
- Provide an error output
- Provide troubleshooting means

One step at a time…



Init → Input validation → Input processing → Output & Cleanup

Demo

Script structure

@cj_berlin

Init and prerequisites

Make sure you actually can do what you came to do

Init Pointers

- Pre-execution checks like **`#requires`** offload checking onto environment admins who may or may not be aware of the requirements not being met
- Load modules explicitly and make sure they‘re loaded
  - Include version in the check if relevant
- Check resource access before accessing resources
  - Connectivity to systems, databases, webservices
  - Read/Write access to config/log/data folders

Demo

Init

@cj_berlin

More Init Pointers

- Do not attempt „self-repair“ if some of the prereqs aren‘t met
- Document prereqs clearly in a comment block at the beginning of the script
- Whether you output init warnings or not, depends on the intended use case (false negatives/positives)!

PSCONF.EU
VIENNA
20 - 23 June 2022

Working with dependencies

And introducing dependencies yourself

@cj_berlin

Dependencies Pointers

- Load 3rd party modules early and watch out for errors
  - **Module loading may provide output as well!**
- Module qualified cmdlet names help avoid ambiguity
- Own modules/includes are a double-edges sword
- Make your scripts as portable as possible, at least the prereqs checking part
- Try to ship as few files as possible with a script

Demo

Modules

Qualified Cmdlets

@cj_berlin

Configurations

- Simple configuration values:
  - Default values of optional parameters
- Config files:
  - Script path vs. machine-wide path vs. user path
- Registry:
  - HKLM vs. HKCU / not cross-platform
- Central configurations provider:
  - Web service with a hard-coded host name → **availability!**

Credentials

- Ideal case: Your script doesn‘t need any credentials
- Best case: The invoking facility will pass you a PSCredential object or some equivalent of it
- Otherwise: Check for availability of credentials early
  - It is a good indicator of how well the environment has been prepared
- In any case: Check the credentials‘ actual validity!

Error handling & Early Bugout

Because there ain‘t nobody there to see that red text…

@cj_berlin

Return vs. Abort

- There are four possible outcomes:
  - Init OK, script did what it came here to do
  - Init OK, script could not achieve everything
  - Init OK, something unexpected happened during execution
  - Init failed, for whatever reasons
- Providing „meaningful output“ for each is key
  - Use a result/exit function as the <u>only</u> output facility
  - Exit after providing output, but think about cleaning up!

Error Handling

- Leave no ~~man~~ unhandled error behind!
- Set `$ErrorActionPreference = "Stop"` at the start
- Use `try {} catch {}` for exceptions you anticipate and `trap {}` for those you don‘t
  - The execution will resume after the trap‘ped script block…
  - …so may want to just call the bugout function.

Early Bugout

- On early bugout, there can be a lot to clean up!
  - open remoting sessions (e.g. Exchange)
  - open database connections
  - open files or even loaded registry hives
- This may be one of the rare occasions where it‘s semi-OK to access global resources from within a function… maybe ☺

Demo

Error Handling

Bugout

Early Abort by invoking facility

This is one of the cases you can‘t do much about, but…

- all „changing“ actions as close together as possible
- if you persist state…
  - log every state change as soon as it occurs
  - check for a saved state of previous execution(s) and roll forward (or back, whichever suits the purpose better)
  - remove/update state on success

PSCONF.EU VIENNA 20 - 23 June 2022

Debugging & Troubleshooting

Because no one will be there to watch every step

@cj_berlin

Debugging Pointers

- Debugging in development:
  - All output goes to Verbose stream
- Prepare for debugging at first execution in target
  - Debugging marker redirects Verbose to log file

Troubleshooting Pointers

- Triggering debug mode
  - Not running in a particular (e.g. SYSTEM) context
  - Not running in a particular host
  - Explicitly (e.g. file marker)
- Finding a place for log files
  - In a profile-less batch, `[System.IO.Path]::GetTempPath()` may not be of much help

Demo

Debugging

Troubleshooting

Monitoring the execution

Because not running does not usually leave traces

Monitoring Pointers

- It‘s easy to monitor success/failure if the script gets to run in the first place
- Always talk to the invoking systems‘ admins and have them monitor their part
- With occasional execution like detection actions: schedule and run tests under real conditions
- Leave a breadcrumb upon each execution *(where?)*

Demo

Breadcrumb

@cj_berlin

PSCONF.EU VIENNA 20 - 23 June 2022

Wrappin‘ it up

@cj_berlin

Takeaways from this session

- Decide what a „meaningful output“ has to be for each of the four execution scenario and put it in a function
- Check prerequisites and access up front
- Prepare generic building blocks & reuse in all scripts
- „Code as Code“ script builders allow shipping singe-file
    - CI/CD or roll your own – whatever you‘re comfortable with

PSCONF.EU VIENNA 20 - 23 June 2022

@cj_berlin

Q&A

Thank you!

https://github.com/it-pro-berlin-de/esm-psconfeu22

@cj_berlin