



2018

# Containers! Where is my PowerShell?



Jan Egil Ring & Øyvind Kallstad

# Agenda

- Management of containers using PowerShell
- Using PowerShell inside containers
- Windows & Linux containers side-by-side

# Demo

# Discussion point

- PSDockerTools – new community-based module
  - Go or no-go?
  - Are there any value in creating a «proper» Docker module for PowerShell, or will most people use the native Docker CLI (docker.exe) anyway?

# Summary

- PowerShell is great for managing containers
  - ConvertFrom-Json is your friend
  - PowerShell can be the glue for integrations
- PowerShell can be used inside containers
  - "Business as usual" – can be used like we are used to
  - With PowerShell Core, it is also possible to leverage PowerShell inside Linux and Nano Server containers

# about\_Author

PowerShell-6.0.0

```
PS C:\> Get-ContactInfo
```

```
Name      : Jan Egil Ring  
E-mail    : jan.egil.ring@crayon.com  
Twitter   : @JanEgilRing  
Website   : {www.powershell.no}
```

PowerShell-6.0.0

```
PS C:\> Get-ContactInfo
```

```
Name      : Øyvind Kallstad  
E-mail    : okallstad@gmail.com  
Twitter   : @OKallstad  
Website   : {communary.net}
```

# Next Steps

- Now: 15 min break
- Grab a coffee
- Stay here to enjoy next presentation
- Change track and switch to another room
- Ask me questions or meet me in a breakout session room afterwards

**Questions?**



# Bonus tip: VS Code Extension for Docker

## Docker commands

Many of the most common Docker and docker compose commands are built right into the Command Palette (F1).

>docker|

Docker: Compose Down

recently used

Docker: Compose Up

Docker: Build Image

Docker: Inspect Image

Docker: Remove Image

Docker: Run Interactive

Docker: Add docker files to workspace

Docker: Attach Shell

other commands

Docker: Azure CLI

Docker: Deploy Image to Azure App Service


Docker: DockerHub Logout

Docker: Push

Docker: Refresh

# VS Code Extension for Docker

IntelliSense (completions) for `Dockerfile` and `docker-compose.yml` files.

 Dockerfile x

```
1 FROM node:6-alpine
2 ENV NODE_ENV production
3 WORKDIR /usr/src/app
4 COPY ["package.json", "npm-shrinkwrap.json*", "./"]
5 RUN npm install --production --silent && mv node_modules ../
6 COPY . .
7 EXPOSE 3000
```

```
8
9
10 ADD source dest
11 ARG name
12 ARG name=defaultValue
13 CMD [ "executable" ]
14 COPY source dest
15 ENTRYPOINT [ "executable" ]
16 ENV key=value
17 EXPOSE port
18 FROM baseImage
19 HEALTHCHECK --interval=30s --timeout=30s
20 HEALTHCHECK NONE
21 LABEL key="value"
```

Copy files, folders, or remote URLs from `source` to the `dest` path in the image's filesystem.

ADD hello.txt /absolute/path  
ADD hello.txt relative/to/workdir