

PowerShell Multi-DLL Build Script — Rocket COBOL 11.0

PowerShell Multi-DLL Build Script - Rocket COBOL 11.0

Dit document bevat een volledig schaalbaar PowerShell-buildsysteem voor Rocket COBOL 11.0 met ondersteuning voor meerdere DLL-projecten, dynamische configuratie, foutafhandeling en nette logging.

Overzicht

Dit script ondersteunt:

- Meerdere DLL's in één build-run
 - Eigen .def-bestand per DLL
 - Eigen lijst COBOL-modules per DLL
 - Automatische compilatie en linking
 - Opruimen van objectbestanden per DLL
 - Volledige compatibiliteit met Rocket COBOL 11.0
-

PowerShell Build Script

```

<#
Rocket COBOL 11.0 - Multi-DLL Build System (PowerShell)
Ondersteunt: meerdere DLL's, meerdere modules per DLL
Auteur: Noam
#>

$CobolBin = "C:\RocketCOBOL\bin"
$CobolLib = "C:\RocketCOBOL\lib"
$OutDir = "bin"

if (-not (Test-Path $OutDir)) {
New-Item -ItemType Directory -Path $OutDir | Out-Null
}

Write-Host ""
Write-Host "=====
Write-Host " MULTI-DLL BUILD START (PowerShell)"
Write-Host "=====
Write-Host ""

# -----
# DEFINITIE VAN ALLE DLL-PROJECTEN
# -----

$DllProjects = @(
@{
Name = "mymath"
Def = "exports_math.def"
Sources = @("add.cbl", "subtract.cbl", "getver.cbl")
},
@{
Name = "datatools"
Def = "exports_data.def"
Sources = @("fill.cbl", "normalize.cbl", "stats.cbl")
}
)

# -----
# FUNCTIE: Compile COBOL module
# -----

function Compile-CobolModule {
param(
[string]$Source,
[string]$OutDir,
[string]$CobolBin
)

```

Gebruik

1. Pas de paden naar Rocket COBOL aan indien nodig.
2. Voeg extra DLL-projecten toe door een nieuw object toe te voegen aan \$DllProjects.
3. Zorg dat .def-bestanden en COBOL-modules in dezelfde map staan.
4. Voer het script uit via PowerShell:

```
powershell -ExecutionPolicy Bypass -File build.ps1
```

Conclusie

Dit PowerShell-buildsysteem biedt een schaalbare, onderhoudsvriendelijke en professionele manier om meerdere Rocket COBOL-DLL's te bouwen binnen één workflow. Het is ideaal voor CI/CD-omgevingen, grote projecten en teams die strikte ABI-consistentie vereisen.