Erweitertes Script - Mit Cleanup alter Versionen

Hier ist das erweiterte Script, das nach der Installation alle anderen Node.js Versionen entfernt:

? `update-node.ps1` (Mit Cleanup)

```
# ============================================
# Node.js Update Script für Windows 11 (NVM)
# LTS VERSION + CLEANUP
# ============================================

Write-Host "`n□□Node.js Update Script gestartet...`n" -ForegroundColor Cyan

# 1. Aktuelle installierte Versionen anzeigen
Write-Host "□□Aktuell installierte Node.js Versionen:" -ForegroundColor Yellow
$installedBefore = cmd /c "nvm list" 2>&1
Write-Host $installedBefore -ForegroundColor DarkGray

# 2. NVM-Liste anzeigen
Write-Host "`n□□Suche neueste Node.js LTS Version..." -ForegroundColor Yellow

$nvmOutput = cmd /c "nvm list available" 2>&1

Write-Host "`nVerfügbare Versionen:" -ForegroundColor Gray
Write-Host $nvmOutput -ForegroundColor DarkGray

# ============================================
# NVM LIST AVAILABLE Format:
# |   CURRENT    |     LTS      |  OLD STABLE  | OLD UNSTABLE |
# |--------------|--------------|--------------|--------------
# |    24.1.0    |   22.16.0    |   0.12.18    |   0.11.16    |
# ============================================
```

```
# Die LTS Version ist in der ZWEITEN Spalte!
$lines = $nvmOutput -split "`n"
$latestLTS = $null

foreach ($line in $lines) {
    # Suche nach Zeilen mit dem Tabellenformat und extrahiere die ZWEITE Versionsnummer (LTS)
    if ($line -match '^\s*\|\s*[\d.]+\s*\|\s*([\d]+\.[\d]+\.[\d]+)\s*\|') {
        $latestLTS = $matches[1]
        break
    }
}

# Fallback: Alternative Methode
if (-not $latestLTS) {
    foreach ($line in $lines) {
        $versions = [regex]::Matches($line, '(\d+\.\d+\.\d+)')
        if ($versions.Count -ge 2) {
            $latestLTS = $versions[1].Value
            break
        }
    }
}

# Wenn immer noch nichts gefunden, manuelle Eingabe
if (-not $latestLTS) {
    Write-Host "`n⚠  Konnte LTS Version nicht automatisch ermitteln." -ForegroundColor Red
    Write-Host "   Bitte schaue oben in der Tabelle - LTS ist die ZWEITE Spalte.`n" -
ForegroundColor Yellow
    $latestLTS = Read-Host "   Gib die gewünschte LTS Version ein (z.B. 22.16.0)"
}

if (-not $latestLTS) {
    Write-Host " Keine Version angegeben. Abbruch." -ForegroundColor Red
    exit 1
}

Write-Host "`n   ➡  Ausgewählte LTS Version: $latestLTS" -ForegroundColor Green

# Bestätigung
```

```
Write-Host ""
$confirm = Read-Host "   Fortfahren mit Version $latestLTS? (J/n)"
if ($confirm -eq "n" -or $confirm -eq "N") {
    $latestLTS = Read-Host "   Gib die gewünschte Version ein"
}

# 3. LTS Version installieren
Write-Host "`n□□Installiere Node.js $latestLTS (LTS)..." -ForegroundColor Yellow
cmd /c "nvm install $latestLTS"

# 4. LTS Version aktivieren
Write-Host "`n□ Aktiviere Node.js $latestLTS..." -ForegroundColor Yellow
cmd /c "nvm use $latestLTS"

# Kurze Pause damit NVM die Änderung übernimmt
Start-Sleep -Seconds 3

# 5. Überprüfen
Write-Host "`n□□Aktuelle Versionen:" -ForegroundColor Yellow
$nodeVersion = cmd /c "node -v" 2>&1
$npmVersion = cmd /c "npm -v" 2>&1
Write-Host "   Node.js: $nodeVersion" -ForegroundColor Green
Write-Host "   NPM: $npmVersion" -ForegroundColor Green

# ========================================
# 6. CLEANUP - Alte Versionen entfernen
# ========================================
Write-Host "`n□□Cleanup: Entferne alte Node.js Versionen..." -ForegroundColor Yellow

# Installierte Versionen abrufen
$nvmList = cmd /c "nvm list" 2>&1
$installedVersions = @()

foreach ($line in ($nvmList -split "`n")) {
    # Versionsnummern extrahieren (Format: "  * 22.16.0 (Currently using...)" oder "
20.10.0")
    if ($line -match '(\d+\.\d+\.\d+)') {
        $version = $matches[1]
        if ($version -ne $latestLTS) {
```

```
            $installedVersions += $version
        }
    }
}

# Duplikate entfernen
$installedVersions = $installedVersions | Select-Object -Unique

if ($installedVersions.Count -eq 0) {
    Write-Host "   ✓ Keine alten Versionen gefunden - alles sauber!" -ForegroundColor Green
} else {
    Write-Host "`n   Folgende alte Versionen werden entfernt:" -ForegroundColor Yellow
    foreach ($version in $installedVersions) {
        Write-Host "   • $version" -ForegroundColor DarkGray
    }

    Write-Host ""
    $confirmDelete = Read-Host "   Alle alten Versionen löschen? (J/n)"

    if ($confirmDelete -ne "n" -and $confirmDelete -ne "N") {
        foreach ($version in $installedVersions) {
            Write-Host "   □□□ Entferne Node.js $version..." -ForegroundColor Gray
            cmd /c "nvm uninstall $version" 2>&1 | Out-Null
        }
        Write-Host "`n   ✓ Alle alten Versionen entfernt!" -ForegroundColor Green
    } else {
        Write-Host "   □□  Cleanup übersprungen." -ForegroundColor Yellow
    }
}

# 7. NPM selbst aktualisieren
Write-Host "`n□□Aktualisiere NPM auf neueste Version..." -ForegroundColor Yellow
cmd /c "npm install -g npm@latest"

# 8. NPM Cache leeren (optional aber empfohlen)
Write-Host "`n□□Leere NPM Cache..." -ForegroundColor Yellow
cmd /c "npm cache clean --force" 2>&1 | Out-Null
Write-Host "   ✓ Cache geleert" -ForegroundColor Green
```

```
# 9. Globale Pakete installieren/aktualisieren
Write-Host "`n□□Installiere globale Pakete..." -ForegroundColor Yellow

$packages = @(
    "less",
    "sass",
    "typescript",
    "npm-check-updates",
    "nodemon",
    "live-server"
)

foreach ($package in $packages) {
    Write-Host "   → Installiere $package..." -ForegroundColor Gray
    cmd /c "npm install -g $package" 2>$null
}

# 10. Zusammenfassung
Write-Host "`n" -NoNewline
Write-Host ("=" * 50) -ForegroundColor Cyan
Write-Host "□ FERTIG! Zusammenfassung:" -ForegroundColor Green
Write-Host ("=" * 50) -ForegroundColor Cyan

$finalNode = cmd /c "node -v" 2>&1
$finalNpm = cmd /c "npm -v" 2>&1

Write-Host "`n   Node.js Version: $finalNode (LTS)" -ForegroundColor White
Write-Host "   NPM Version: $finalNpm" -ForegroundColor White

Write-Host "`n   Installierte Node.js Versionen:" -ForegroundColor White
cmd /c "nvm list"

Write-Host "`n   Installierte globale Pakete:" -ForegroundColor White
cmd /c "npm list -g --depth=0"

Write-Host "`n□□Alles erledigt!" -ForegroundColor Green
Write-Host "□□Tipp: Öffne ein NEUES Terminal damit alle Änderungen aktiv sind.`n" -
ForegroundColor Yellow
```

? Neue Features:

Feature	Beschreibung
□□ **Vorher-Übersicht**	Zeigt installierte Versionen vor dem Update
□□ **Auto-Cleanup**	Findet und listet alle alten Versionen
□ **Bestätigung**	Fragt vor dem Löschen nach Bestätigung
□□ **Batch-Deinstallation**	Entfernt alle alten Versionen automatisch
□□ **NPM Cache**	Leert auch den NPM Cache
□□ **Nachher-Übersicht**	Zeigt am Ende nur noch die aktive Version

?? Ausführen:

```
.\update-node.ps1
```

? Beispiel-Ablauf:

```
□□Node.js Update Script gestartet...

□□Aktuell installierte Node.js Versionen:
    20.10.0
    18.19.0
  * 22.11.0 (Currently using 64-bit executable)

□□Suche neueste Node.js LTS Version...
   →□  Ausgewählte LTS Version: 22.16.0
   Fortfahren mit Version 22.16.0? (J/n): J

□□Installiere Node.js 22.16.0 (LTS)...
□ Aktiviere Node.js 22.16.0...

□□Cleanup: Entferne alte Node.js Versionen...

   Folgende alte Versionen werden entfernt:
   • 20.10.0
   • 18.19.0
```

```
    • 22.11.0

    Alle alten Versionen löschen? (J/n): J
    □□□ Entferne Node.js 20.10.0...
    □□□ Entferne Node.js 18.19.0...
    □□□ Entferne Node.js 22.11.0...

    ✓ Alle alten Versionen entfernt!

□□Alles erledigt!
```

Jetzt hast du ein sauberes System mit nur einer Node.js Version! □□□

Revision #1
Created 2026-05-31 14:29:45 UTC by art10m
Updated 2026-05-31 14:29:58 UTC by art10m