ENGLISH VERSION PAGE 17…

CleanMac Assistant – Alles-in-één Script (De Middelvinger naar Pay-for-Free Scripts zoals CleanMyMac X)

Een open-source AppleScript- en shell-script toolkit om je Mac schoon en snel te houden— zonder te betalen voor wat gratis zou moeten zijn.
Ontwikkeld door EasyComp Zeeland •
https://easycompzeeland.nl

Easy Readme – Nerdy details hieronder…

1. Overzicht

CleanMac Assistant is een gratis macOS-onderhoudsscript. Het helpt je om:

- Prullenmand leeg te maken en caches te verwijderen
- Logbestanden en ongebruikte taalbestanden op te schonen
- Browsercaches leeg te maken (Chrome, Firefox)
- RAM vrij te maken en systeemonderhoudsscripts uit te voeren

- DNS-cache te flushen en kernservices te herstarten (Finder, Dock)
- macOS-updates te installeren en Homebrew-pakketten bij te werken
- Safari-geschiedenis, iMessage-logs, browsercookies en FaceTime-logs te verwijderen
- Malware te scannen met ClamAV en LaunchAgents te verwijderen
- Apps te deïnstalleren of te resetten per bundle-identifier
- Schijfgebruik te analyseren met ncdu

In plaats van te betalen voor CleanMyMac X of vergelijkbare tools, gebruik je dit ene script.

2. Afhankelijkheden

Voordat het script draait, controleert CleanMac Assistant of de volgende tools geïnstalleerd zijn:

- Homebrew
- ClamAV (voor malware-scans)
- ncdu (voor schijfgebruik-analyse)

Ontbreekt er iets? Dan installeert het script de tool automatisch via Homebrew en toont een macOS-notificatie.

3. Werking

- Dubbelklik op CleanMacAssistant.applescript in Script Editor.
- Er verschijnt een menu met deze categorieën:
  - ℹ️ Over
  - 🧹 Systeemopschoning
  - 🚀 Prestatie-optimalisatie
  - 🛡️ Privacybeheer
  - 🔒 Beveiliging
  - 📦 Appbeheer
  - 💾 Schijfgebruik-analyse
  - 🚪 Stoppen

- Kies een categorie.
- Voor elke taak krijg je een notificatie en een dialoogvenster met opties:
  • Uitvoeren om de taak te starten
  • Overslaan om deze over te slaan
  • Terug om terug te keren naar het hoofdmenu
- Als een taak mislukt, kun je kiezen voor Stoppen, Overslaan of Terug.
- Herhaal tot je "Stoppen" kiest.

4. ECZ Hulp op Afstand

Heb je extra hulp nodig? EasyComp Zeeland ontwikkelde twee gratis tools:

- ECZ QHOATOOL (Quick Help On a Distance): snelle, veilige afstandsondersteuning door ECZ-experts.
- ECZ HOATOOL (Help On a Distance): gratis publiek hulpmiddel zodat iedereen (vrienden of familie) elkaar kan helpen.

Meer informatie en downloaden via:
https://easycompzeeland.nl/services/hulp-op-afstand/

5. Bijdragen

Verbeteringen zijn welkom!

- Open een issue als je een bug vindt of een functie wilt voorstellen.
- Fork deze repository, breng wijzigingen aan op een nieuwe branch en dien een pull request in.

6. Licentie

Gepubliceerd onder de MIT License. Zie het LICENSE-bestand voor details.

Nerdy details…

CleanMac Assistant – Alles-in-één Script
(De Middelvinger naar Pay-for-Free Scripts zoals CleanMyMac X)
Een open-source AppleScript- en shell-script toolkit om je Mac volledig schoon te houden—zonder te betalen voor wat gratis zou moeten zijn.
Ontwikkeld door EasyComp Zeeland • https://easycompzeeland.nl

INHOUD

1. Overzicht

Bij het opstarten van CleanMac Assistant verschijnt een welkomstdialoog die je herinnert: je hoeft geen geld uit te geven aan dure onderhoudssoftware, want je krijgt hetzelfde (of beter) resultaat gratis. Dit script geeft credits aan de open-source community—Homebrew, ClamAV, ncdu en meer—en verwijst naar de

website van EasyComp Zeeland voor details. Ook wordt ECZ's hulp-op-afstandservice uitgelicht als je zelf ondersteuning nodig hebt.

Voordat een opschonings- of optimalisatieroutine start, controleert CleanMac Assistant automatisch of de drie essentiële afhankelijkheden—ClamAV, ncdu en Homebrew—geïnstalleerd zijn. Ontbreekt er iets, dan installeert het script dit via Homebrew en toont steeds een macOS-notificatie. Zo werkt elke functie—malware-scans, schijfgebruik-analyse, Homebrew-updates—direct, elke keer.

2. Afhankelijkheden

CleanMac Assistant maakt gebruik van drie belangrijke commandoregeltools:

- Homebrew
- ClamAV (voor malware-scans)
- ncdu (voor interactieve schijfgebruik-analyse)

Bij het opstarten voert het script de handler checkDependencies() uit:

```
set dependencies to {"clamav", "ncdu", "brew"}
repeat with i from 1 to count of dependencies
   set dependency to item i of dependencies
   try
      do shell script "which " & dependency
   on error
      display notification "Installing " & dependency & "…" with title "CleanMac
Assistant"
      do shell script "/usr/local/bin/brew install " & dependency
   end try
end repeat
```

Als een van deze tools ontbreekt, wordt deze via Homebrew geïnstalleerd en zie je een notificatie.

3. Scriptstructuur & Functies

3.1 Controle van afhankelijkheden

checkDependencies() controleert of ClamAV, ncdu en Homebrew aanwezig zijn. Ontbreekt een tool, dan installeert het script deze via Homebrew en toont een notificatie.

3.2 Mensvriendelijke takenamen

humanNameFor(taskID) zet een interne taak-identifier (bijvoorbeeld “trash”) om in een mensvriendelijke Engelse naam (bijvoorbeeld “Empty Trash”). Dit wordt gebruikt in dialoogvensters en notificaties. De handler is:

```
on humanNameFor(taak)
    if taak is "trash" then
        return "Empty Trash"
    else if taak is "checkDependencies" then
        return "Check Dependencies"
    else if taak is "cache" then
        return "Clean Cache"
    else if taak is "logs" then
        return "Clean Log Files"
    else if taak is "localizations" then
        return "Remove Localization Files"
    else if taak is "chrome" then
        return "Clean Chrome Cache"
    else if taak is "firefox" then
        return "Clean Firefox Cache"
    else if taak is "ram" then
        return "Free RAM"
    else if taak is "scripts" then
        return "Run Maintenance Scripts"
    else if taak is "dns" then
        return "Flush DNS Cache"
    else if taak is "restart" then
        return "Restart Finder & Flush"
    else if taak is "update" then
        return "Install macOS Updates"
    else if taak is "brew" then
        return "Update Homebrew Packages"
    else if taak is "safari" then
        return "Clear Safari History"
    else if taak is "imessage" then
        return "Delete iMessage Logs"
    else if taak is "cookies" then
        return "Delete Browser Cookies"
    else if taak is "facetime" then
        return "Delete FaceTime Logs"
    else if taak is "malware" then
        return "Perform Malware Scan"
    else if taak is "agents" then
        return "Remove LaunchAgents"
    else if taak is "uninstall" then
        return "Uninstall Application"
    else if taak is "reset" then
        return "Reset Application Preferences"
```

```
    else if taak is "disk" then
        return "Analyze Disk Usage"
    else
        return "Unknown Task"
    end if
end humanNameFor
```

3.3 Opschoon- & onderhoudsfuncties

cleanupTrash()
Leegt de prullenmand van de gebruiker (~/.Trash/) *met administrator-bevoegdheden.*

```
on cleanupTrash()
    do shell script "rm -rf ~/.Trash/" with administrator privileges
end cleanupTrash
```

cleanupCache()
Leegt de user-cache door xcrun -k te draaien, met notificaties voor en na.

```
on cleanupCache()
    display notification "Starting cache cleanup…" with title "CleanMac Assistant"
    do shell script "xcrun -k"
    display notification "Cache cleanup completed." with title "CleanMac Assistant"
end cleanupCache
```

cleanupLogs()
Verwijdert alle systeemlogbestanden onder /private/var/log/.

```
on cleanupLogs()
    do shell script "sudo rm -rf /private/var/log/" with administrator privileges
end cleanupLogs
```

cleanupLocalizations()
Zoekt en verwijdert alle .lproj-directories in /Applications, behalve en.lproj.

```
on cleanupLocalizations()
    do shell script "find /Applications -name '*.lproj' ! -name 'en.lproj' -type d -exec
rm -rf {} +" with administrator privileges
end cleanupLocalizations
```

cleanupChrome()
Verwijdert de cachemap van Chrome (~/Library/Caches/Google/Chrome/).

```
on cleanupChrome()
    do shell script "rm -rf ~/Library/Caches/Google/Chrome/" with administrator
privileges
end cleanupChrome
```

cleanupFirefox()
Verwijdert Firefox-profielcache (~/Library/Caches/Firefox/Profiles/).

```
on cleanupFirefox()
```

```
    do shell script "rm -rf ~/Library/Caches/Firefox/Profiles/" with administrator
privileges
end cleanupFirefox

freeRAM()
Voert purge uit om inactief RAM vrij te maken.
on freeRAM()
    do shell script "purge" with administrator privileges
end freeRAM

runMaintenanceScripts()
Laadt de macOS periodieke onderhoudsdaemons (daily, weekly, monthly).
on runMaintenanceScripts()
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-daily.plist"
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-weekly.plist"
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-monthly.plist"
end runMaintenanceScripts

flushDNS()
Flusht de DNS-cache door dscacheutil -flushcache; sudo killall -HUP
mDNSResponder uit te voeren.
on flushDNS()
    do shell script "dscacheutil -flushcache; sudo killall -HUP mDNSResponder" with
administrator privileges
end flushDNS

restartTools()
Herstart Finder, Dock en SystemUIServer om de UI-processen te verversen.
on restartTools()
    do shell script "killall Finder; killall Dock; killall SystemUIServer" with administrator
privileges
end restartTools

systemUpdate()
Installeert alle beschikbare macOS-updates via softwareupdate -ia.
on systemUpdate()
    do shell script "softwareupdate -ia" with administrator privileges
end systemUpdate

brewUpdate()
Voert eerst brew doctor uit, daarna brew update --verbose && brew upgrade --
verbose om Homebrew-pakketten bij te werken. Toont notificaties bij succes of
fout.
on brewUpdate()
```

```
  set brewPath to do shell script "which brew"
  try
    do shell script brewPath & " doctor" with administrator privileges
    do shell script brewPath & " update --verbose && " & brewPath & " upgrade --
verbose" with administrator privileges
    display notification "Homebrew packages updated." with title "CleanMac
Assistant"
  on error errMsg number errNum
    display notification "Error updating Homebrew packages: " & errMsg with title
"CleanMac Assistant"
  end try
end brewUpdate

safariHistory()
Verwijdert alle Safari-browsegeschiedenis via een SQLite-command op History.db.
on safariHistory()
  do shell script "sqlite3 ~/Library/Safari/History.db 'DELETE from history_items'" with
administrator privileges
end safariHistory

imessageLogs()
Verwijdert de iMessage-chatdatabasebestanden (~/Library/Messages/chat.db*).
on imessageLogs()
  do shell script "rm -rf ~/Library/Messages/chat.db*" with administrator privileges
end imessageLogs

cookiesCleanup()
Verwijdert alle browsercookies onder ~/Library/Cookies/.
on cookiesCleanup()
  do shell script "rm -rf ~/Library/Cookies/" with administrator privileges
end cookiesCleanup

facetimeLogs()
Verwijdert FaceTime-preference-logs in ~/Library/Preferences/
com.apple.FaceTime.bag.plist.
on facetimeLogs()
  do shell script "rm -rf ~/Library/Preferences/com.apple.FaceTime.bag.plist" with
administrator privileges
end facetimeLogs

scanMalware()
Gebruikt ClamAV om de hele schijf (/) recursief te scannen op malware. Toont
notificatie bij voltooiing of fout.
on scanMalware()
  try
    do shell script "/usr/local/bin/clamscan -r / --verbose" with administrator
privileges
```

```
        display notification "Malware scan completed." with title "CleanMac Assistant"
    on error errMsg number errNum
        display notification "Error during malware scan: " & errMsg with title "CleanMac
Assistant"
    end try
end scanMalware

removeLaunchAgents()
Verwijdert alle LaunchAgents in ~/Library/LaunchAgents/.
on removeLaunchAgents()
    do shell script "rm -rf ~/Library/LaunchAgents/" with administrator privileges
end removeLaunchAgents

uninstallApp()
Vraagt om een app-naam en verwijdert vervolgens de bijbehorende .app-map
uit /Applications.
on uninstallApp()
    display dialog "Enter the name of the app you want to uninstall:" default answer
""
    set appName to text returned of result
    do shell script "sudo rm -rf /Applications/" & quoted form of appName & ".app"
with administrator privileges
end uninstallApp

resetApp()
Vraagt om een bundle-identifier (bijvoorbeeld com.vendor.app) en voert defaults
delete <bundleID> uit om alle gebruikers-preferences van die app te verwijderen.
on resetApp()
    display dialog "Enter the bundle identifier of the app to reset preferences (e.g.,
com.vendor.app):" default answer ""
    set bundleID to text returned of result
    do shell script "defaults delete " & bundleID with administrator privileges
end resetApp

analyzeDisk()
Installeert ncdu via Homebrew (indien ontbrekend), opent dan Terminal en voert
ncdu / uit voor interactieve schijfgebruik-analyse. Toont notificatie bij voltooiing of
fout.
on analyzeDisk()
    try
        do shell script "/usr/local/bin/brew install ncdu"
        tell application "Terminal"
            activate
            do script "/usr/local/bin/ncdu /"
        end tell
        display notification "Disk usage analysis completed." with title "CleanMac
Assistant"
```

```
    on error errMsg number errNum
        display notification "Error analyzing disk usage: " & errMsg with title "CleanMac Assistant"
    end try
end analyzeDisk
```

4. Categorieën & Taken

CleanMac Assistant ordent zijn functies in deze categorieën (weergegeven in het "choose from list"-dialoogvenster). Elke categorie bevat een set taak-IDs die overeenkomen met de handlers hierboven:

1. Over
2. Systeemopschoning – trash, cache, logs, localizations, chrome, firefox
3. Prestatie-optimalisatie – ram, scripts, dns, restart, update, brew
4. Privacybeheer – safari, imessage, cookies, facetime
5. Beveiliging – malware, agents
6. Appbeheer – uninstall, reset
7. Schijfgebruik-analyse – disk
8. Stoppen

Hieronder volgen de weergave en de bijbehorende Engelse beschrijvingen (in het script) van elke categorie:

4.1 Over

Display "About" Dialog
– Titel: CleanMac Assistant – Your Free Mac Maintenance Tool
– Inhoud:
1. Introduceert CleanMac Assistant als EasyComp Zeeland's gratis macOS-onderhoudsscript.
2. Benadrukt dat je niet hoeft te betalen voor premium onderhoudssoftware (subtiel de middelvinger naar CleanMyMac X).
3. Credits voor Homebrew, ClamAV, ncdu en andere open-source projecten.
4. Biedt links:
• EasyComp Zeeland: https://easycompzeeland.nl
• ECZ Quick Help On a Distance: https://easycompzeeland.nl/services/hulp-op-afstand/

4.2 Systeemopschoning

Taken in "Systeemopschoning":
- Empty Trash – cleanupTrash()
- Clean Cache – cleanupCache()
- Clean Log Files – cleanupLogs()
- Remove Localization Files – cleanupLocalizations()
- Clean Chrome Cache – cleanupChrome()
- Clean Firefox Cache – cleanupFirefox()

Elke actie toont een macOS-notificatie voor de start en na voltooiing.

4.3 Prestatie-optimalisatie

Taken in "Prestatie-optimalisatie":

1. Free RAM – freeRAM()
   - Voert purge uit om inactief RAM vrij te maken.
2. Run Maintenance Scripts – runMaintenanceScripts()
   - Laadt Apple's periodieke onderhoudsdaemons (daily, weekly, monthly).
3. Flush DNS Cache – flushDNS()
   - Voert dscacheutil -flushcache; sudo killall -HUP mDNSResponder uit.
4. Restart Finder & Flush – restartTools()
   - Herstart Finder, Dock en SystemUIServer om UI-processen te verversen.
5. Install macOS Updates – systemUpdate()
   - Voert softwareupdate -ia uit om alle updates te installeren.
6. Update Homebrew Packages – brewUpdate()
   - Voert brew doctor uit, vervolgens brew update --verbose && brew upgrade --verbose. Toont notificatie bij succes of fout.

4.4 Privacybeheer

Taken in "Privacybeheer":
- Clear Safari History – safariHistory()
  - Verwijdert alle Safari-geschiedenis met een SQLite-command.
- Delete iMessage Logs – imessageLogs()
  - Verwijdert ~/Library/Messages/chat.db*.
- Delete Browser Cookies – cookiesCleanup()
  - Verwijdert alle bestanden in ~/Library/Cookies/*.
- Delete FaceTime Logs – facetimeLogs()
  - Verwijdert ~/Library/Preferences/com.apple.FaceTime.bag.plist.

4.5 Beveiliging

Taken in "Beveiliging":

- Perform Malware Scan – scanMalware()
  - Voert ClamAV clamscan -r / --verbose uit en toont notificatie bij voltooiing of fout.
- Remove LaunchAgents – removeLaunchAgents()
  - Verwijdert alles in ~/Library/LaunchAgents/*.

4.6 Appbeheer

Taken in "Appbeheer":

- Uninstall Application – uninstallApp()
  - Vraagt om een app-naam, vervolgens wordt /Applications/<app>.app verwijderd.
- Reset Application Preferences – resetApp()
  - Vraagt om een bundle-identifier en verwijdert de opgeslagen voorkeuren.

4.7 Schijfgebruik-analyse

Taak in "Schijfgebruik-analyse":

- Analyze Disk Usage – analyzeDisk()
  - Installeert ncdu als deze ontbreekt, opent Terminal en voert ncdu / uit. Toont notificatie bij voltooiing of fout.

4.8 Stoppen

Stop Script – Beëindigt de run-loop en sluit het script.

5. Gebruiksaanwijzing

1. Open CleanMacAssistant.applescript in Apple's Script Editor.
2. Klik op de Run-knop (▶) in de toolbar.
3. Een welkomstdialoog verschijnt waarin je een categorie kunt kiezen.
4. Selecteer een categorie (bijvoorbeeld "Systeemopschoning").
5. Voor elke taak in die categorie:
   – Er verschijnt een notificatie: "Task X/X: <mensvriendelijke taak> starting…"
   – Een dialoog vraagt: "Wat wil je doen met <mensvriendelijke taak>?" met knoppen: Uitvoeren, Overslaan, Terug.
   – Bij keuze "Uitvoeren" wordt de betreffende handler uitgevoerd (eventueel met administrator-rechten).
   – Bij voltooiing verschijnt een notificatie. Bij een fout toont het dialoogvenster het foutbericht met opties: Stoppen, Overslaan, Terug.

6. Herhaal tot je "Stoppen" kiest of het dialoogvenster annuleert.
7. Administrator-rechten worden enkel gevraagd wanneer strikt nodig.

6. ECZ Hulp op Afstand

Bij EasyComp Zeeland ontdekten we dat populaire afstandshulpmiddelen (AnyDesk, TeamViewer) soms:

- Traag zijn (langzame verbindingen)
- Kostbaar (abonnementskosten, gebruiksbeperkingen)
- Incompatibel (occasionele issues op macOS, Linux, Windows)

Om dit op te lossen, ontwikkelde EasyComp Zeeland twee gespecialiseerde tools:

- ECZ QHOATOOL (Quick Help On a Distance Tool)
  • Snel: lage latency voor realtime ondersteuning.
  • Veilig: end-to-end encryptie houdt je data beschermd.
  • Expert Support: directe verbinding met EasyComp Zeeland-technici.
  • Cross-Platform: werkt op macOS, Linux en Windows.
- ECZ HOATOOL (Help On a Distance Tool)
  • Gratis & Publiek: iedereen—familie, vrienden, collega's—kan het downloaden en gebruiken.
  • Gebruiksvriendelijk: eenvoudige interface voor het geven of ontvangen van hulp.
  • Cross-Platform: compatibel met macOS, Linux en Windows.

Beide tools, inclusief stapsgewijze handleidingen en FAQ's, vind je op: https://easycompzeeland.nl/services/hulp-op-afstand/

Of je nu rechtstreeks hulp van een EasyComp Zeeland-expert wilt of iemand op afstand wilt ondersteunen: ECZ Hulp op Afstand heeft je gedekt.

7. Richtlijnen voor bijdrages

We verwelkomen bijdragen om CleanMac Assistant te verbeteren op het gebied van functionaliteit, betrouwbaarheid en gebruiksgemak. Zo kun je helpen:

1. **Rapporteer Issues**
   – Heb je een bug ontdekt of wil je een nieuwe functie, open dan een issue op GitHub met een duidelijke beschrijving en eventueel reproduceerstappen.

2. **Fork & Commit**
   – Fork deze repository naar je eigen GitHub-account.
   – Maak een nieuwe branch (bijvoorbeeld feature/add-new-task).
   – Implementeer je wijzigingen en hou je aan de bestaande stijl.

3. **Pull Request**
   – Push je branch naar je fork.
   – Open een pull request naar de main-branch van deze repository.
   – Geef een beschrijvende titel en een samenvatting van wat je hebt aangepast en waarom.

4. **Review & Merge**
   – Maintainers reviewen je pull request en kunnen om wijzigingen vragen.
   – Na goedkeuring wordt je bijdrage samengevoegd en word je in de commit-geschiedenis vermeld.

Alle bijdragers worden opgenomen in een CONTRIBUTORS-sectie zodra de community groeit.

8. Licentie

Dit project is gelicenseerd onder de MIT License. Zie het LICENSE-bestand voor volledige voorwaarden.

CleanMac Assistant – All-In-One Script (Flipping the Finger to Pay-for-Free Scripts Like CleanMyMac X)

An open-source AppleScript and shell-script toolkit to keep your Mac clean and fast—without paying for what should be free.
Developed by EasyComp Zeeland • https://easycompzeeland.nl

Easy Readme - Nerdy details below…

5. Overview

CleanMac Assistant is a free macOS maintenance script. It helps you:

- Empty trash and remove caches
- Clear log files and unused language files
- Clean browser caches (Chrome, Firefox)
- Free up RAM and run system maintenance scripts
- Flush DNS cache and restart core services (Finder, Dock)
- Install macOS updates and update Homebrew packages
- Remove Safari history, iMessage logs, browser cookies, FaceTime logs
- Scan for malware with ClamAV and remove LaunchAgents
- Uninstall or reset apps by bundle identifier
- Analyze disk usage with ncdu

Instead of paying for CleanMyMac X or similar tools, use this single script.

2. Dependencies

Before running, CleanMac Assistant checks for:

1. **Homebrew**

2. **ClamAV** (for malware scans)
3. **ncdu** (for disk usage analysis)

If any are missing, the script installs them via Homebrew and shows a macOS notification.

3. How It Works

1. Double-click **CleanMacAssistant.applescript** in Script Editor.
2. A menu appears with these categories:
    - ℹ️ About
    - 🧹 System Cleanup
    - 🚀 Performance Optimization
    - 🛡️ Privacy Management
    - 🔒 Security
    - 📦 App Management
    - 💾 Disk Usage Analysis
    - 🚪 Exit
3. Pick a category.
4. For each task, you get a notification and a dialog:
    - **Execute** to run the task
    - **Skip** to skip it
    - **Back** to return to the category menu
5. If a task fails, you can **Stop**, **Skip**, or **Back**.
6. Repeat until you choose "Exit."

4. ECZ Remote Help

Need extra assistance? EasyComp Zeeland built two free tools:

- **ECZ QHOATOOL** (Quick Help On a Distance): fast, secure remote support from ECZ experts.
- **ECZ HOATOOL** (Help On a Distance): free, public tool so anyone (friends or family) can help each other.

Learn more and download at:
https://easycompzeeland.nl/services/hulp-op-afstand/

5. Contributing

We welcome improvements!

1. Open an **issue** if you find a bug or want a feature.
2. Fork this repository, make changes on a new branch, and submit a **pull request**.

6. License

Released under the **MIT License**. See the LICENSE file for details.

Nerdy details…
CleanMac Assistant – All-In-One Script (Flipping the Finger to Pay-for-Free Scripts Like CleanMyMac X)

An open-source AppleScript and shell-script toolkit to keep your Mac pristine—without paying for what should be free.
Developed by EasyComp Zeeland • https://easycompzeeland.nl

CONTENTS

1. Overview

Upon launching CleanMac Assistant, you are greeted by a welcome dialog that reminds you: you don't need to shell out money for premium maintenance software when you can get the same (or better) results for free. This script credits the open-source community—Homebrew, ClamAV, ncdu, and more—and directs

you to EasyComp Zeeland's website for details. It also highlights ECZ's remote assistance services if you ever need a hand.

Before any cleanup or optimization routine runs, CleanMac Assistant automatically checks that its three essential dependencies—ClamAV, ncdu, and Homebrew—are installed. If any are missing, the script quietly installs them via Homebrew, showing a macOS notification each time. That guarantees every feature—malware scanning, disk usage analysis, Homebrew updates—works out of the box, every time.

2. Dependencies

CleanMac Assistant relies on three key command-line tools:

1. Homebrew
2. ClamAV (for malware scanning)
3. ncdu (for interactive disk usage analysis)

At startup, the script runs on checkDependencies() which does the following:

```
set dependencies to {"clamav", "ncdu", "brew"}
repeat with i from 1 to count of dependencies
   set dependency to item i of dependencies
   try
      do shell script "which " & dependency
   on error
      display notification "Installing " & dependency & "…" with title "CleanMac
Assistant"
      do shell script "/usr/local/bin/brew install " & dependency
   end try
end repeat
```

If any of those tools are missing, it invokes Homebrew to install them, displaying a notification.

3. Script Structure & Functions

3.1 Dependency Check

checkDependencies() verifies presence of ClamAV, ncdu, and Homebrew. If any is missing, it installs it via Homebrew—and displays a notification.

3.2 Human-Readable Task Names

humanNameFor(taskID) converts an internal task identifier (for example, “trash”) into a human-readable English string (for example, “Empty Trash”). This is used when showing dialogs and notifications. The handler is:

```
on humanNameFor(taak)
    if taak is "trash" then
        return "Empty Trash"
    else if taak is "checkDependencies" then
        return "Check Dependencies"
    else if taak is "cache" then
        return "Clean Cache"
    else if taak is "logs" then
        return "Clean Log Files"
    else if taak is "localizations" then
        return "Remove Localization Files"
    else if taak is "chrome" then
        return "Clean Chrome Cache"
    else if taak is "firefox" then
        return "Clean Firefox Cache"
    else if taak is "ram" then
        return "Free RAM"
    else if taak is "scripts" then
        return "Run Maintenance Scripts"
    else if taak is "dns" then
        return "Flush DNS Cache"
    else if taak is "restart" then
        return "Restart Finder & Flush"
    else if taak is "update" then
        return "Install macOS Updates"
    else if taak is "brew" then
        return "Update Homebrew Packages"
    else if taak is "safari" then
        return "Clear Safari History"
    else if taak is "imessage" then
        return "Delete iMessage Logs"
    else if taak is "cookies" then
        return "Delete Browser Cookies"
    else if taak is "facetime" then
        return "Delete FaceTime Logs"
    else if taak is "malware" then
        return "Perform Malware Scan"
    else if taak is "agents" then
        return "Remove LaunchAgents"
    else if taak is "uninstall" then
        return "Uninstall Application"
    else if taak is "reset" then
        return "Reset Application Preferences"
```

```
    else if taak is "disk" then
        return "Analyze Disk Usage"
    else
        return "Unknown Task"
    end if
end humanNameFor
```

3.3 Cleanup & Maintenance Functions

```
cleanupTrash()
Empties the user's Trash folder (~/.Trash/) with administrator privileges.
on cleanupTrash()
    do shell script "rm -rf ~/.Trash/" with administrator privileges
end cleanupTrash

cleanupCache()
Clears user caches by running xcrun -k, with notifications before and after
execution.
on cleanupCache()
    display notification "Starting cache cleanup…" with title "CleanMac Assistant"
    do shell script "xcrun -k"
    display notification "Cache cleanup completed." with title "CleanMac Assistant"
end cleanupCache

cleanupLogs()
Deletes all system log files under /private/var/log/.
on cleanupLogs()
    do shell script "sudo rm -rf /private/var/log/" with administrator privileges
end cleanupLogs

cleanupLocalizations()
Finds and removes all .lproj directories in /Applications, excluding en.lproj.
on cleanupLocalizations()
    do shell script "find /Applications -name '*.lproj' ! -name 'en.lproj' -type d -exec
rm -rf {} +" with administrator privileges
end cleanupLocalizations

cleanupChrome()
Deletes Chrome's cache folder at ~/Library/Caches/Google/Chrome/.
on cleanupChrome()
    do shell script "rm -rf ~/Library/Caches/Google/Chrome/" with administrator
privileges
end cleanupChrome

cleanupFirefox()
Deletes Firefox's profile cache directories at ~/Library/Caches/Firefox/Profiles/.
on cleanupFirefox()
```

```
    do shell script "rm -rf ~/Library/Caches/Firefox/Profiles/" with administrator
privileges
end cleanupFirefox

freeRAM()
Executes purge to free up inactive memory.
on freeRAM()
    do shell script "purge" with administrator privileges
end freeRAM

runMaintenanceScripts()
Loads macOS periodic maintenance daemons (daily, weekly, monthly).
on runMaintenanceScripts()
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-daily.plist"
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-weekly.plist"
    do shell script "launchctl load /System/Library/LaunchDaemons/
com.apple.periodic-monthly.plist"
end runMaintenanceScripts

flushDNS()
Flushes the DNS cache by running dscacheutil -flushcache; sudo killall -HUP
mDNSResponder.
on flushDNS()
    do shell script "dscacheutil -flushcache; sudo killall -HUP mDNSResponder" with
administrator privileges
end flushDNS

restartTools()
Restarts Finder, Dock, and SystemUIServer to refresh the UI.
on restartTools()
    do shell script "killall Finder; killall Dock; killall SystemUIServer" with administrator
privileges
end restartTools

systemUpdate()
Installs all available macOS updates via softwareupdate -ia.
on systemUpdate()
    do shell script "softwareupdate -ia" with administrator privileges
end systemUpdate

brewUpdate()
Runs brew doctor, then updates and upgrades all Homebrew packages. Displays
notifications on success or error.
on brewUpdate()
    set brewPath to do shell script "which brew"
```

```
    try
        do shell script brewPath & " doctor" with administrator privileges
        do shell script brewPath & " update --verbose && " & brewPath & " upgrade --
verbose" with administrator privileges
        display notification "Homebrew packages updated." with title "CleanMac
Assistant"
    on error errMsg number errNum
        display notification "Error updating Homebrew packages: " & errMsg with title
"CleanMac Assistant"
    end try
end brewUpdate

safariHistory()
Deletes all Safari browsing history by executing an SQLite command on the History
database.
on safariHistory()
    do shell script "sqlite3 ~/Library/Safari/History.db 'DELETE from history_items'" with
administrator privileges
end safariHistory

imessageLogs()
Deletes the iMessage chat database files (~/Library/Messages/chat.db*).
on imessageLogs()
    do shell script "rm -rf ~/Library/Messages/chat.db*" with administrator privileges
end imessageLogs

cookiesCleanup()
Deletes all browser cookies stored under ~/Library/Cookies/.
on cookiesCleanup()
    do shell script "rm -rf ~/Library/Cookies/" with administrator privileges
end cookiesCleanup

facetimeLogs()
Deletes FaceTime preference logs at ~/Library/Preferences/
com.apple.FaceTime.bag.plist.
on facetimeLogs()
    do shell script "rm -rf ~/Library/Preferences/com.apple.FaceTime.bag.plist" with
administrator privileges
end facetimeLogs

scanMalware()
Uses ClamAV to recursively scan the entire file system (/) for malware. Displays a
notification on completion or error.
on scanMalware()
    try
        do shell script "/usr/local/bin/clamscan -r / --verbose" with administrator
privileges
```

```
        display notification "Malware scan completed." with title "CleanMac Assistant"
    on error errMsg number errNum
        display notification "Error during malware scan: " & errMsg with title "CleanMac
Assistant"
    end try
end scanMalware

removeLaunchAgents()
Deletes all LaunchAgents in the user's library at ~/Library/LaunchAgents/.
on removeLaunchAgents()
    do shell script "rm -rf ~/Library/LaunchAgents/" with administrator privileges
end removeLaunchAgents

uninstallApp()
Prompts the user for an application name, then deletes the corresponding .app
bundle from /Applications.
on uninstallApp()
    display dialog "Enter the name of the app you want to uninstall:" default answer
""
    set appName to text returned of result
    do shell script "sudo rm -rf /Applications/" & quoted form of appName & ".app"
with administrator privileges
end uninstallApp

resetApp()
Prompts for a bundle identifier (for example, com.vendor.app) and runs defaults
delete <bundleID> to remove all user preferences for that app.
on resetApp()
    display dialog "Enter the bundle identifier of the app to reset preferences (e.g.,
com.vendor.app):" default answer ""
    set bundleID to text returned of result
    do shell script "defaults delete " & bundleID with administrator privileges
end resetApp

analyzeDisk()
Installs ncdu via Homebrew (if missing), then opens Terminal to run ncdu /. Allows
interactive disk usage analysis, and issues a notification on completion or error.
on analyzeDisk()
    try
        do shell script "/usr/local/bin/brew install ncdu"
        tell application "Terminal"
            activate
            do script "/usr/local/bin/ncdu /"
        end tell
        display notification "Disk usage analysis completed." with title "CleanMac
Assistant"
    on error errMsg number errNum
```

```
        display notification "Error analyzing disk usage: " & errMsg with title "CleanMac Assistant"
    end try
end analyzeDisk
```

4. Categories & Tasks

CleanMac Assistant organizes its functions into these categories (shown in the "choose from list" dialog). Each category contains a set of task IDs that map to the functions above:

1. About
2. System Cleanup – trash, cache, logs, localizations, chrome, firefox
3. Performance Optimization – ram, scripts, dns, restart, update, brew
4. Privacy Management – safari, imessage, cookies, facetime
5. Security – malware, agents
6. App Management – uninstall, reset
7. Disk Usage Analysis – disk
8. Exit

Below is how each category is presented and the English descriptions of the tasks within:

4.1 About

Display "About" Dialog
– Title: CleanMac Assistant – Your Free Mac Maintenance Tool
– Content:
1. Introduces CleanMac Assistant as EasyComp Zeeland's free macOS maintenance script.
2. Emphasizes that you don't need to pay for premium maintenance software (subtly flipping the finger to CleanMyMac X).
3. Credits Homebrew, ClamAV, ncdu, and other open-source projects.
4. Provides links:
- EasyComp Zeeland: https://easycompzeeland.nl
- ECZ Quick Help On a Distance: https://easycompzeeland.nl/services/hulp-op-afstand/

4.2 System Cleanup

Tasks in "System Cleanup":
- Empty Trash – cleanupTrash()
- Clean Cache – cleanupCache()
- Clean Log Files – cleanupLogs()
- Remove Localization Files – cleanupLocalizations()
- Clean Chrome Cache – cleanupChrome()
- Clean Firefox Cache – cleanupFirefox()

Each action displays a macOS notification before starting and after completing.

4.3 Performance Optimization

Tasks in "Performance Optimization":

1. Free RAM – freeRAM()
   - Runs purge to free inactive memory.
2. Run Maintenance Scripts – runMaintenanceScripts()
   - Loads Apple's periodic maintenance daemons (daily, weekly, monthly).
3. Flush DNS Cache – flushDNS()
   - Executes dscacheutil -flushcache; sudo killall -HUP mDNSResponder.
4. Restart Finder & Flush – restartTools()
   - Restarts Finder, Dock, and SystemUIServer to refresh UI processes.
5. Install macOS Updates – systemUpdate()
   - Runs softwareupdate -ia to install all pending updates.
6. Update Homebrew Packages – brewUpdate()
   - Runs brew doctor, then brew update --verbose && brew upgrade --verbose. Notifies on success or error.

4.4 Privacy Management

Tasks in "Privacy Management":
- Clear Safari History – safariHistory()
  - Deletes all Safari browsing history items via an SQLite command.
- Delete iMessage Logs – imessageLogs()
  - Removes ~/Library/Messages/chat.db*.
- Delete Browser Cookies – cookiesCleanup()
  - Erases all files in ~/Library/Cookies/*.
- Delete FaceTime Logs – facetimeLogs()
  - Deletes ~/Library/Preferences/com.apple.FaceTime.bag.plist.

4.5 Security

Tasks in "Security":
- Perform Malware Scan – scanMalware()
  - Runs ClamAV clamscan -r / --verbose and notifies when finished or on error.
- Remove LaunchAgents – removeLaunchAgents()
  - Deletes everything in ~/Library/LaunchAgents/*.

4.6 App Management

Tasks in "App Management":
- Uninstall Application – uninstallApp()
  - Prompts for an app name, then deletes /Applications/<app>.app.
- Reset Application Preferences – resetApp()
  - Prompts for a bundle identifier and deletes stored preferences.

4.7 Disk Usage Analysis

Task in "Disk Usage Analysis":
- Analyze Disk Usage – analyzeDisk()
  - Installs ncdu if missing, then opens Terminal to run ncdu /. Notifies on completion or error.

4.8 Exit

Stop Script – Ends the run loop and closes the script.

5. Usage Instructions

1. Open CleanMacAssistant.applescript in Apple's Script Editor.
2. Click the Run button (▶) in the toolbar.
3. A welcome dialog appears, prompting you to choose one of the categories above.
4. Select a category (for example, "System Cleanup").
5. For each task in that category:
   – A notification appears: "Task X/X: <Human-Readable Task> starting…"
   – A dialog asks: "What would you like to do with <Human-Readable Task>?" with buttons: Execute, Skip, Back.
   – If you choose Execute, the function runs (with administrator privileges if required).
   – On completion, a notification indicates success. If an error occurs, a dialog shows the error message with options: Stop, Skip, Back.
6. Repeat until you select "Exit" or cancel a dialog.

7. Administrator privileges are requested only when strictly necessary.

6. ECZ Remote Help

At EasyComp Zeeland, we discovered that popular remote assistance tools (AnyDesk, TeamViewer) can be:
• Slow (laggy connections)
• Costly (subscription fees, usage limitations)
• Incompatible (occasional issues on macOS, Linux, Windows)

To fix this, EasyComp Zeeland developed two specialized tools:

1. ECZ QHOATOOL (Quick Help On a Distance Tool)
   • Fast: Low-latency connections for real-time support.
   • Secure: End-to-end encryption keeps your data safe.
   • Expert Support: Connect directly to EasyComp Zeeland technicians.
   • Cross-Platform: Works on macOS, Linux, and Windows.

2. ECZ HOATOOL (Help On a Distance Tool)
   • Free & Public: Anyone—family, friends, or colleagues—can download and use it at no cost.
   • Easy to Use: Simple interface for offering or receiving remote help.
   • Cross-Platform: Compatible with macOS, Linux, and Windows.

Both tools, plus step-by-step guides and FAQs, are available at:
https://easycompzeeland.nl/services/hulp-op-afstand/

Whether you need direct assistance from an EasyComp Zeeland expert or want to help someone remotely, ECZ Remote Help has you covered.

7. Contribution Guidelines

We welcome contributions to improve CleanMac Assistant's functionality, reliability, and usability. Here's how you can help:

1. Report Issues
   – If you find a bug or want a new feature, open an issue on GitHub with a clear description and, if applicable, steps to reproduce.

2. Fork & Commit
   – Fork this repository to your GitHub account.
   – Create a new branch (for example, feature/add-new-task).
   – Implement your changes, following the existing style.

3. Pull Request
   – Push your branch to your fork.
   – Open a pull request against the main branch of this repo.
   – Provide a descriptive title and a summary of what you changed and why.

4. Review & Merge
   – Maintainers will review your pull request and may request changes.
   – Once approved, your contribution will be merged, and you'll be credited in commit history.

All contributors will be listed in a CONTRIBUTORS section once the community grows.

8. License

This project is licensed under the **MIT License**. See the LICENSE file for full terms and conditions.

Wijzigingslogboek

- **09 jun 2025 – v1.0:** Eerste release.
- **09 Jun 2025 – v1.1**: Een probleem opgelost waarbij scripts niet correct werden verplaatst naar de map Bronnen in de app. (Beginnersfout - oeps!)

Changelog

- **09 Jun 2025 – v1.0**: Initial release.
- **09 Jun 2025 – v1.1**: Fixed an issue where scripts were not correctly moved to the Resources folder within the app. (Beginner's mistake—oops!)