Apple Skripting mit Jamf 201:

Ein Leitfaden zum Automatisieren allgemeiner Aufgaben

Wenn Sie ein Apple-Administrator sind, der die grundlegenden Skripting-Fähigkeiten, die Sie in Scripting 101 erlernt haben, auf die nächste Stufe heben möchte, ist dieses Handbuch genau das Richtige für Sie!

Sie können drei gängige IT-Aufgaben bei der Verwaltung von Apple-Geräten mit drei einfachen, leistungsstarken Tools für die Skripterstellung automatisieren:

Schreiben von Text in eine Datei und Abrufen des Textes zur späteren Verwendung

Verwendung von Skript-Parametern in Jamf Pro

Erstellen Sie interaktive Dialoge mit jamfHelper und osascript

Schreiben und Lesen einer Datei

Manchmal müssen Sie Informationen auf einem Mac speichern, damit Sie später darauf zugreifen können. Sie können zum Beispiel einen Bereitstellungsbeleg hinterlassen, aus dem hervorgeht, wann ein Mac vorbereitet und bereitgestellt wurde und wer ihn gebaut hat.

Hier sind einige Möglichkeiten, genau das zu tun:

Über das Terminal

Erinnern Sie sich an den Befehl "echo" aus unserem Webinar Skripting 101?
Dieser Befehl echot, oder druckt, was Sie eingeben. Wenn Sie eingeben:

```
echo 'Hallo, Welt!'
```

...und drücken die Eingabetaste, wird ausgegeben:

```
Hallo, Welt.
```

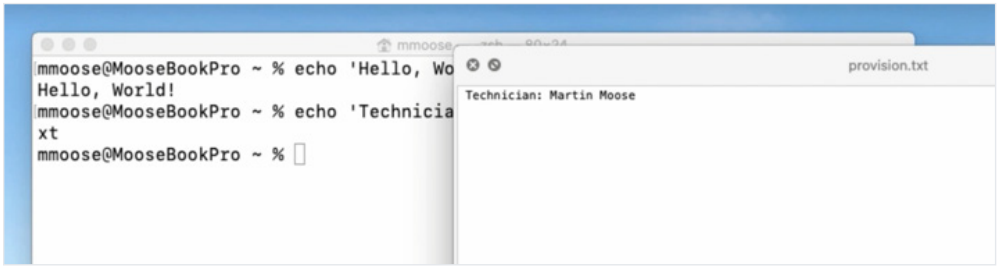
Speichern in einer Textdatei mit dem Befehl echo

Mit dem Zeichen >>, dem sogenannten "Umleitungssymbol", können Sie in einer Datei speichern. Wenn die Datei nicht existiert, wird sie durch das Umleitungssymbol erstellt. Wenn sie bereits vorhanden ist, wird das, was Sie echoten, an das Ende der Datei angehängt. (Wenn ich nur eine Umleitung verwende, >, wird der Inhalt der Datei überschrieben).

```
echo Techniker: Martin Moose >> ~/Desktop/provision.txt
echo "Date: $( /bin/date +%y-%m-%d )" >> !$
echo Abteilung: Grafiken >> !$
```

Wenn Sie die Eingabetaste drücken, wird die Datei auf Ihrem Desktop angezeigt.

Sie können die Datei auswählen und die Leertaste drücken, um sie in QuickLook anzuzeigen.



Hinzufügen des aktuellen Datums zu einer Datei

Um das Hinzufügen des aktuellen Datums zu einer Datei zu automatisieren, verwenden Sie den Befehl:

```
echo "Date: $( /bin/date +%y-%m-%d )"
```

Die "%y-%m-%d" stehen für Jahr, Monat und Datum.

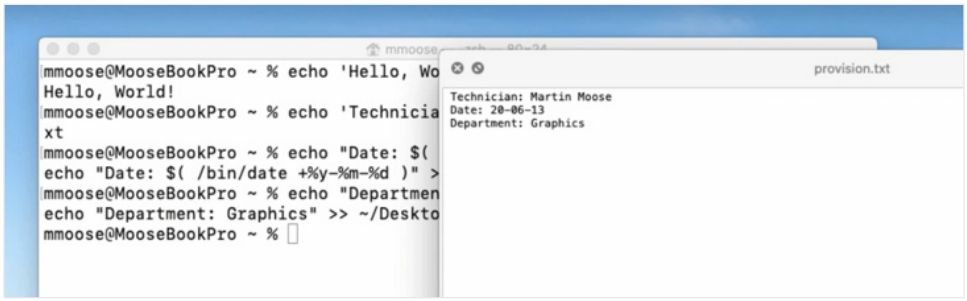
Um ein paar Tastenanschläge zu sparen, verwenden Sie eine Tastenkombination, die das letzte Argument des vorherigen Befehls wiederholt, um den Text in die Datei umzuleiten:

```
>> !$
```

Und dann schreiben Sie einen Abteilungsnamen in die Datei:

```
echo Abteilung: Grafiken >> !$
```

Wenn wir die Datei jetzt mit QuickLook betrachten, ist unser gesamter Text in der Reihenfolge enthalten, in der wir ihn hinzugefügt haben:



Um sie wieder auszulesen, verwenden Sie den Befehl "cat", gefolgt von dem Pfad zur Datei:

```
/bin/cat ~/Desktop/provision.txt
```

Der Inhalt dieser Datei wird dann direkt in Terminal angezeigt:

```
Techniker: Martin Moose
Datum: 20-06-13
Abteilung: Grafiken
```

Überprüfen

echo	Gibt aus, was Sie in das Terminalfenster eingegeben haben
>>	Leitet die Ausgabe in eine Datei um/fügt eine vorhandene Datei an
>	Leitet die Ausgabe in eine Datei um/überschreibt eine vorhandene Datei
$(Befehl)	Führt einen Befehl aus
!$	Wiederholt das letzte Argument
cat	Liest eine Datei

Standardwerte Befehl

Mit dem Befehl "cat" können Sie die gesamte Datei lesen, aber was ist, wenn Sie nur eine bestimmte Information aus der Datei benötigen?

Der Befehl "defaults" ist derselbe Befehl, den Sie verwenden würden, um Listen in Ihrem Einstellungsordner zu lesen, und kann auch verwendet werden, um Informationen zurückzulesen.

Wir können damit auch unsere eigenen Listen schreiben.

Um eine neue Datei zu erstellen, verwenden Sie "defaults write" und teilen Sie dem Programm mit, wo Sie die Datei speichern möchten. Danach geben Sie mit einem Wort "Techniker" den Techniker an, der den Computer gebaut hat, und anschließend den Namen des Technikers.

```
/usr/bin/defaults write ~/Desktop/provision.plist Techniker 'Martin Moose'
```

Drücken Sie die Eingabetaste.

Die Datei wird auf dem Desktop angezeigt, und wenn Sie QuickLook verwenden, um sie zu betrachten, sieht sie ganz anders aus als die erste Datei:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Techniker</key>
    <string>Martin Moose</string>
</dict>
</plist>
```

Der Befehl "defaults" fügt eine Menge Formatierungen hinzu, aber wenn man genau hinsieht, sind die Informationen alle vorhanden.

Fügen wir das Datum und die Abteilung hinzu und sehen wir uns die Datei erneut an:

```
/usr/bin/defaults write ~/Desktop/provision.plist Techniker 'Martin Moose'
/usr/bin/defaults write ~/Desktop/provision.plist Datum $ ( /bin/date '+%y-%m-%d' )
/usr/bin/defaults write ~/Desktop/provision.plist Abteilung 'Grafiken'
```

Die Datei enthält nun alle drei Informationen:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Datum</key>
    <string>20-06-13</string>
    <key>Abteilung</key>
    <string>Grafiken</string>
    <key>Techniker</key>
    <string>Martin Moose</string>
</dict>
</plist>
```

Da es sich um eine Liste handelt, ist jede Art von Information — Techniker, Datum und Abteilung — als "Schlüssel" aufgeführt, und der Wert für diese Art von Information ist unter jedem Schlüssel aufgeführt. (Der Befehl defaults sortiert die Tasten automatisch alphabetisch. Unabhängig von der Reihenfolge, in der Sie sie hinzufügen, werden sie in alphabetischer Reihenfolge aufgeführt).

Die plist sieht viel komplexer aus als unsere erste Textdatei, aber hier zeigt sie sich von ihrer besten Seite:

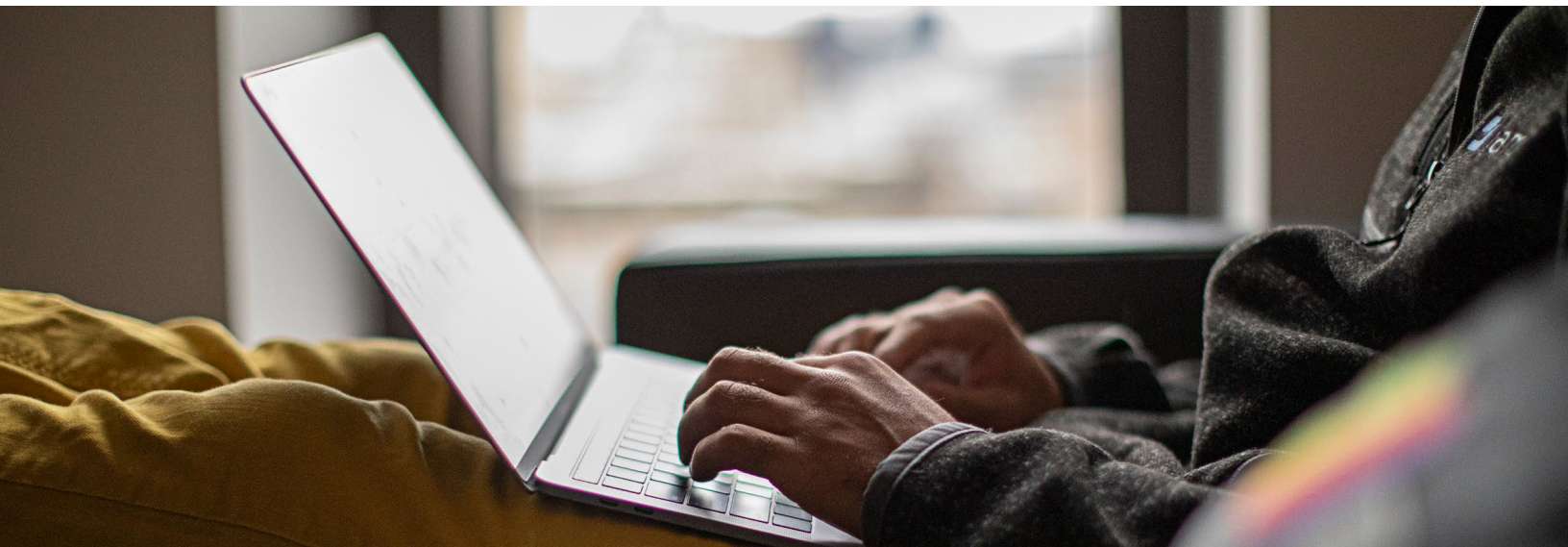
Verwenden Sie "defaults read", um die Datei zu lesen, und geben Sie dann "Techniker", "Datum" oder "Abteilung" an, um den Wert nur eines Elements zurückzugeben. Dies ist besonders nützlich für etwas wie ein Erweiterungsattribut, bei dem ein Administrator*in diesen Befehl in einem kurzen Skript verwenden würde, um das Ergebnis zurück an Jamf Pro zu senden.

Beispiel:

```
/usr/bin/defaults read~/Desktop/provision.plist Datum
```

Das Terminal liefert nur das Datum:

```
20-06-13
```



Verwendung von Skript-Parametern in Jamf Pro

Was sind Skript-Parameter?

Lassen Sie uns das anhand eines kurzen Skripts demonstrieren.

```
#!/bin/zsh
```

Die Abkürzung ist zsh oder zee-shell, wie “seashell.”

Ich füge eine echo-Anweisung hinzu, die nur eine leere Zeile ausgibt, um die Sache übersichtlicher zu machen:

```
echo
```

Dann füge ich eine weitere Echo-Anweisung mit den Parametervariablen "1" bis "5" hinzu.

```
Echo $1 $2 $3 $4 $5
```

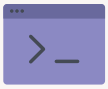
Denken Sie daran, wenn etwas in einem Skript mit einem "$" beginnt, ist das normalerweise ein Indikator dafür, dass etwas eine Variable oder ein Platzhalter für etwas ist.

Das vollständige Skript:

```
#!/bin/zsh
echo
Echo $1 $2 $3 $4 $5
```

Speichern Sie das Skript als parameters.zsh auf dem Desktop.

Jedes Mal, wenn Sie eine neue Skript-Datei erstellen, müssen Sie einen Befehl namens "chmod" in Terminal verwenden, um sie ausführbar zu machen. Andernfalls denkt Terminal, dass es sich nur um eine einfache Textdatei handelt und nicht um ein Skript, das es ausführen kann.



Verwenden Sie Ihren bevorzugten Code-Editor oder einen einfachen Texteditor wie BBEdit oder sogar TextEdit, der mit Ihrem Mac geliefert wird. Schalten Sie alle Einstellungen für die Autokorrektur aus, die gerade Anführungszeichen in geschweifte Anführungszeichen umwandeln. Hierfür sind nur direkte Zitate erforderlich. (Ein Skript-Editor erledigt dies automatisch für Sie.)

Geben Sie immer am Anfang jedes Skripts ein sogenanntes Shebang ein: Raute + Bang #! Gefolgt von /bin/zsh (oder bash oder welchen Skript-Typ Sie auch immer verwenden), sodass das Terminal bei der Verwendung Ihrer Skripte weiß, dass es seashell (oder welchen Skript-Typ Sie auch immer verwenden) als Interpreter verwenden soll.

Geben Sie das Wort "chmod" ein, was "Änderungsmodus" bedeutet, und fügen Sie dann +x hinzu, was "ausführbar machen" bedeutet, und ziehen Sie Ihr Skript in das Fenster.

Wenn Sie nun das Skript gefolgt von "Mary had a little lamb" ausführen, wird "Mary had a little lamb:" zurückgesendet.

```
chmod +x ~/Desktop/parameters.zsh Mary had a little lamb
Mary had a little lamb
```

Das ist nicht sehr aufregend, also lasst uns ein bisschen Spaß haben.

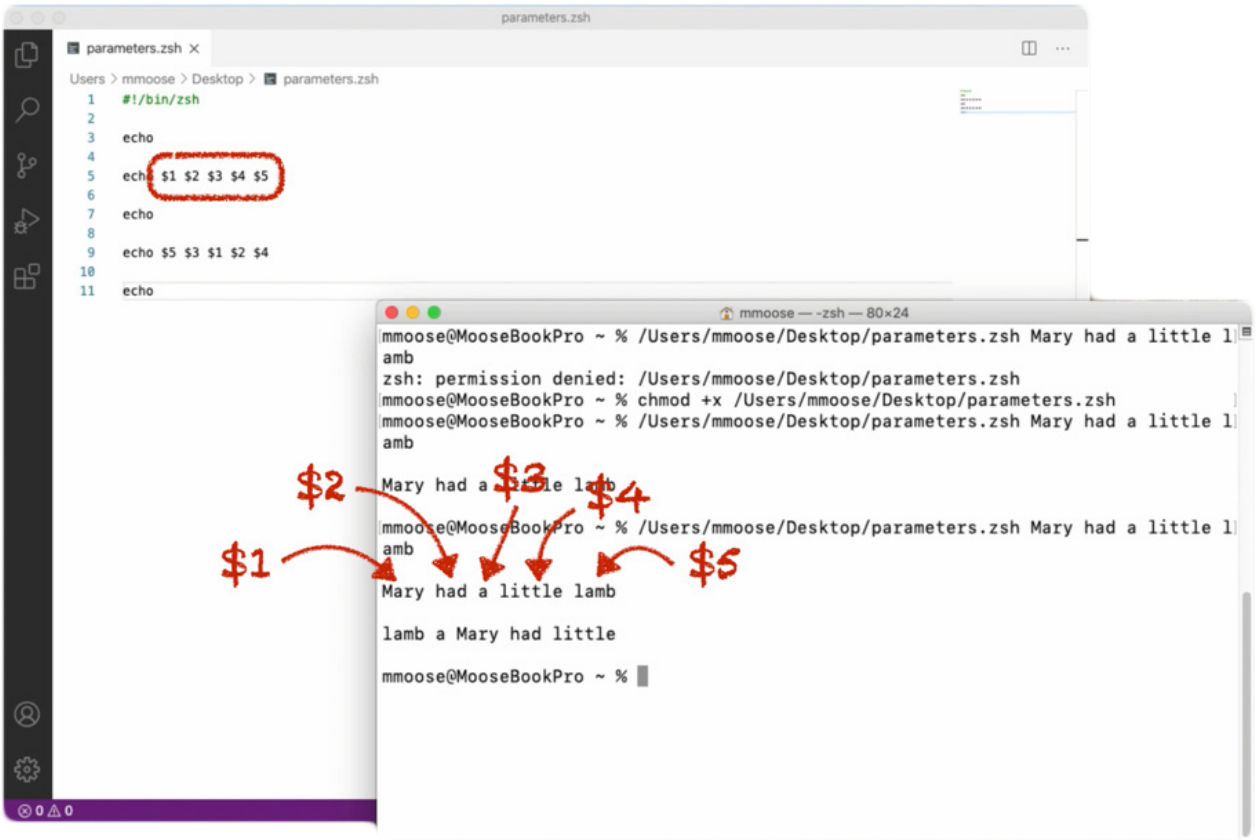
Zurück im Skript fügen Sie einen weiteren Echo-Befehl hinzu. Aber dieses Mal sollten Sie die Reihenfolge der Skript-Variablen vertauschen:

```
#!/bin/zsh
echo
Echo $1 $2 $3 $4 $5
echo
Echo $5 $3 $1 $2 $4
echo
```

Speichern Sie nun und führen Sie das Skript erneut in Terminal aus, gefolgt von "Mary had a little lamb".

```
~/Desktop/parameters.zsh Mary had a little lamb
```

Hier sehen Sie, was Sie sehen werden:



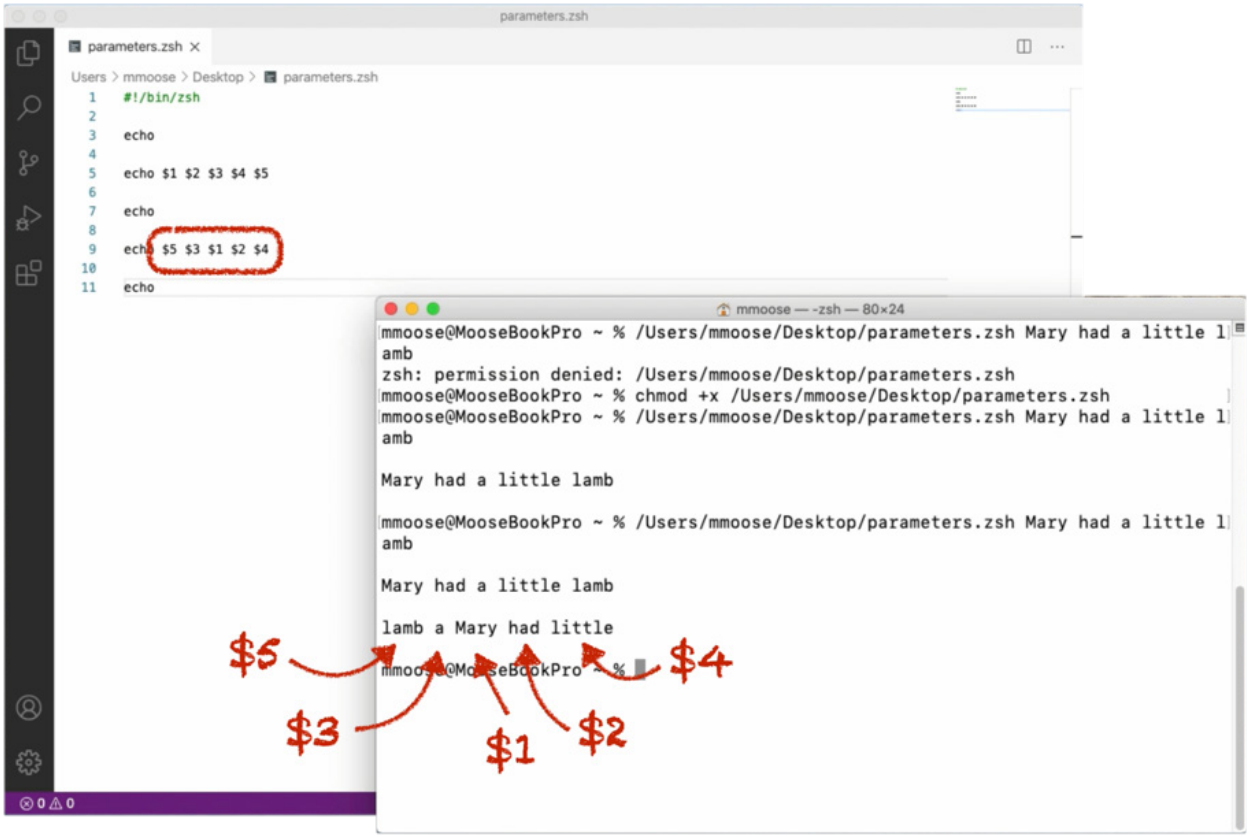
Die erste Zeile sieht richtig aus, aber jetzt ist die zweite völlig durcheinander. Das sagt uns, dass jede dieser Zahlenvariablen im Skript — $1-2-3-4-5 — der Reihenfolge der Elemente entspricht, die wir am Ende des Skripts hinzufügen.

"Mary" ist das erste Wort und beträgt "$1".

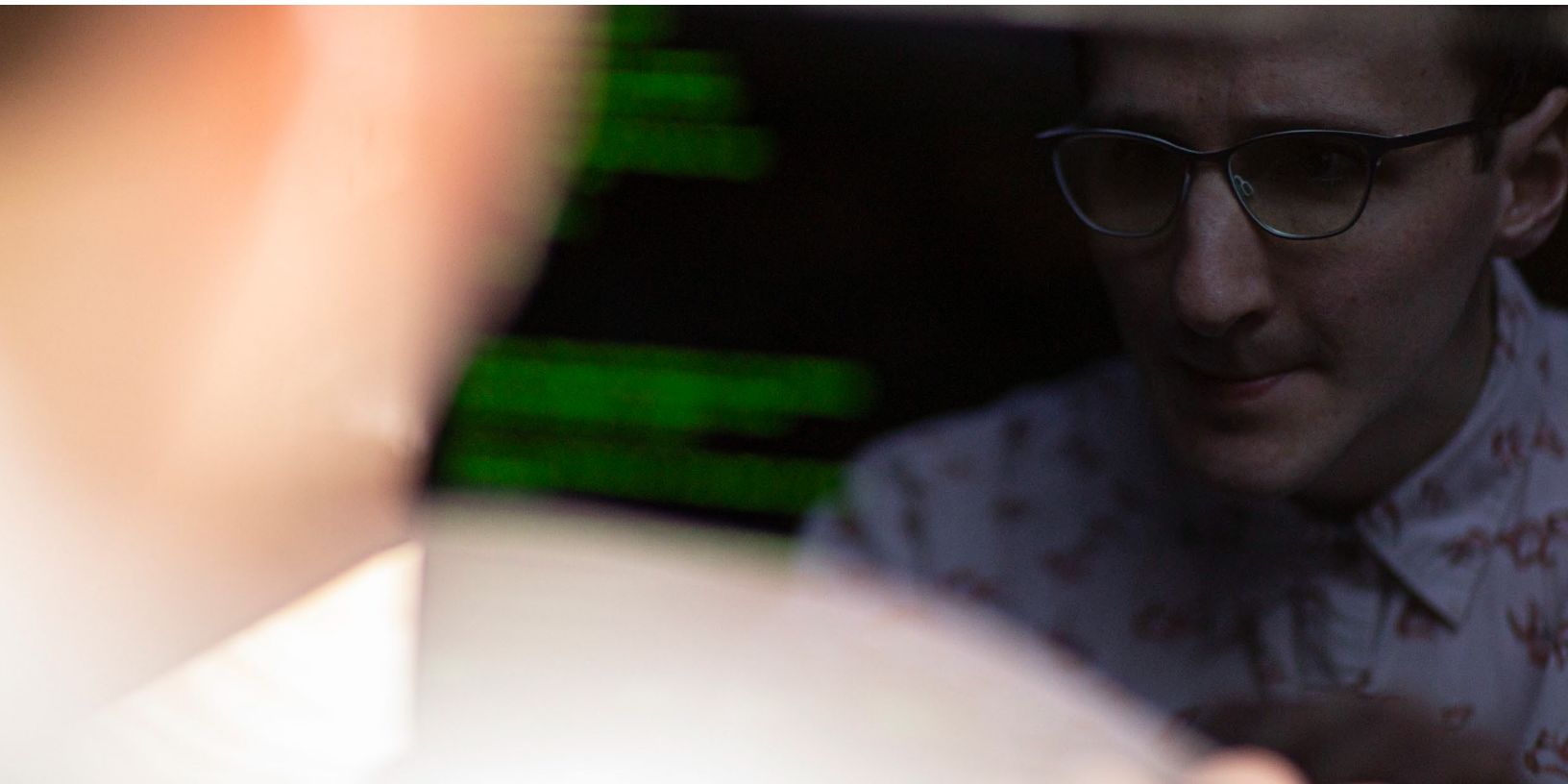
"had" ist das zweite Wort und bedeutet "2 Dollar" usw.

Jedes der Wörter in "Mary had a little lamb" ist ein "Skript-Parameter". Und wir können auf jeden Skript-Parameter durch seine Position nach dem Skript verweisen.

Wenn Sie die Reihenfolge der Variablen im Skript ändern, ändert sich auch die Reihenfolge der Wörter, wenn Sie das Skript ausführen.



Nun, da wir wissen, dass Skript-Parameter die Reihenfolge der Elemente betreffen, die einem Skript folgen, wie verwenden wir dies in Jamf Pro? Sehen wir uns das an.



Suche nach vorhandenen Skripten

Sie finden alle möglichen Skripte, die andere Jamf Pro Administrator*innen erstellt haben, unter auf jamfnation.com/de: Ressourcen ➜ Jamf Pro add-ons ➜ Skripte.

Von hier aus können Sie stöbern oder die Suchfunktion nutzen, was auch immer Sie tun möchten.

In diesem Beispiel verwenden wir “time zone” ein Parameter Skript
So werden Sie vorgehen:

- Skript zum Download
- Datei öffnen
- Gesamten Inhalt auswählen
- Kopieren

Navigieren Sie in Ihrer Jamf Pro-Instanz zu:

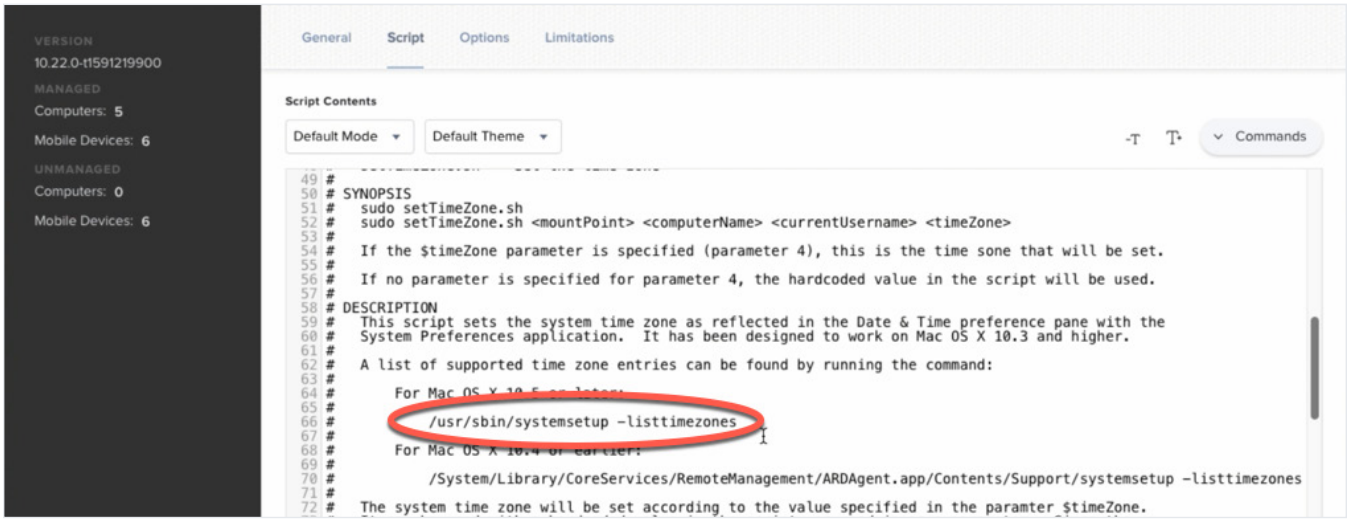
Einstellungen > Computerverwaltung > Skripte > Neues Skript

- Nennen Sie das neue Skript "Zeitzone einstellen".
- Fügen Sie das Skript auf der Registerkarte Skript ein.
- Klicken Sie auf der Registerkarte "Optionen" in das Feld "Parameter 4" und stellen Sie den Namen des Labels auf "Zeitzone" ein. Sie können es nennen, wie Sie wollen, aber es ist sinnvoll, ihm einen sinnvollen Namen zu geben. Wenn Sie das Skript ausführen, stellen Sie die gewünschte Zeitzone im vierten Parameter oder "$4" ein.
- Speichern.
- Wählen Sie erneut die Registerkarte Skript.
- Suchen Sie den Befehl zum Auflisten der Zeitzonen, die Sie kopieren möchten.

Warum beginnen diese also mit dem Parameter "4" und nicht mit "1"?
Wenn Sie sich die winzige Schrift direkt über den Parameterbeschriftungen genau ansehen, werden Sie einen Text erkennen, der besagt: "Parameter 1 bis 3 sind als Einhängepunkt, Computername und Benutzername vordefiniert."

Das bedeutet, dass Jamf Pro diese Informationen immer als die ersten drei Parameter mit jedem Skript sendet, unabhängig davon, ob das Skript sie verwendet oder nicht.

Sie haben keine Kontrolle über diese Parameter, aber Sie haben Kontrolle über den Rest.
Sie haben bis zu 8 weitere Parameter, die Sie nach Belieben definieren können.



So führen Sie den Befehl aus

- Terminal öffnen
- Geben Sie "sudo" ein, was "mit erweiterten Rechten ausführen" bedeutet, und fügen Sie dann den Befehl ein:

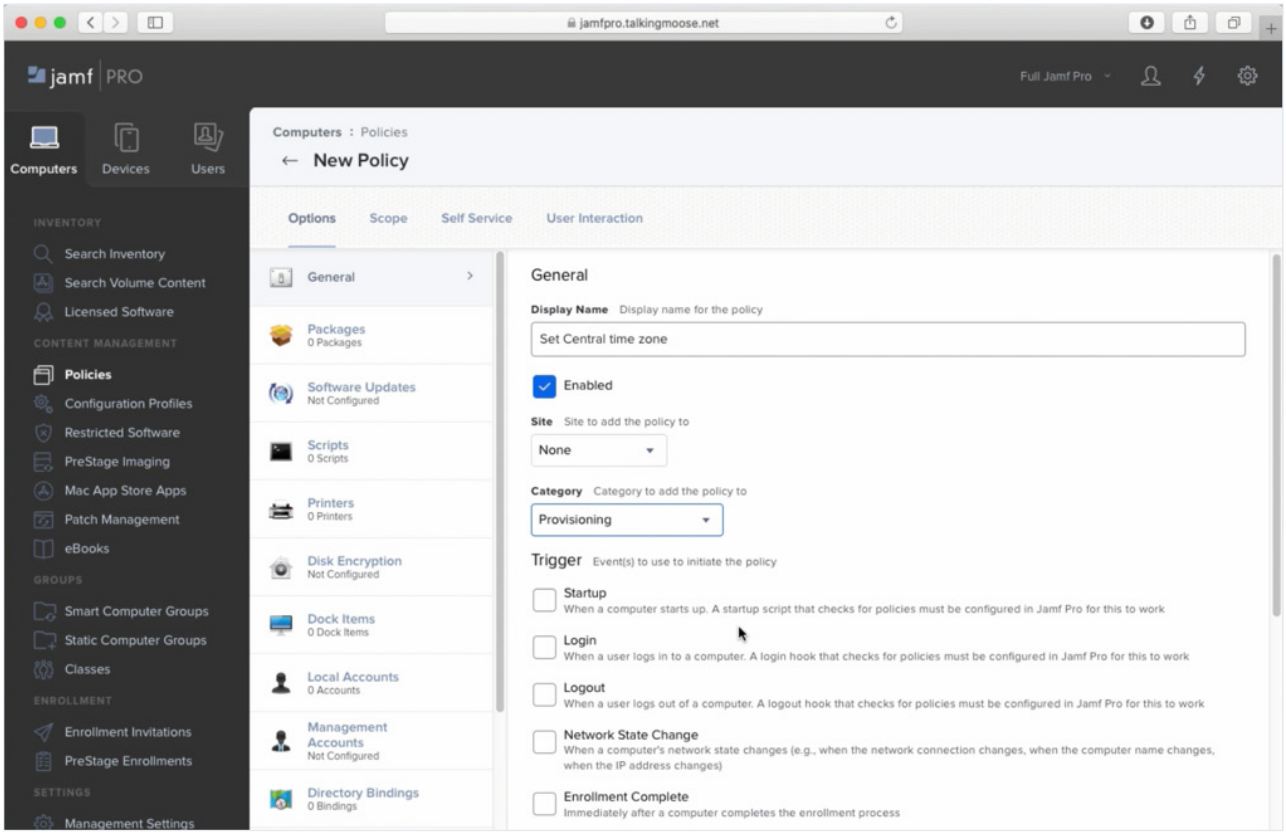
```
/usr/sbin/systemsetup -listtimezones
```

Dadurch wird eine lange Liste der verfügbaren Zeitzonen im richtigen Format für das Skript angezeigt. In diesem Beispiel wählen wir Chicago, das repräsentativ für die zentrale Zeitzone der Vereinigten Staaten ist, und kopieren es.

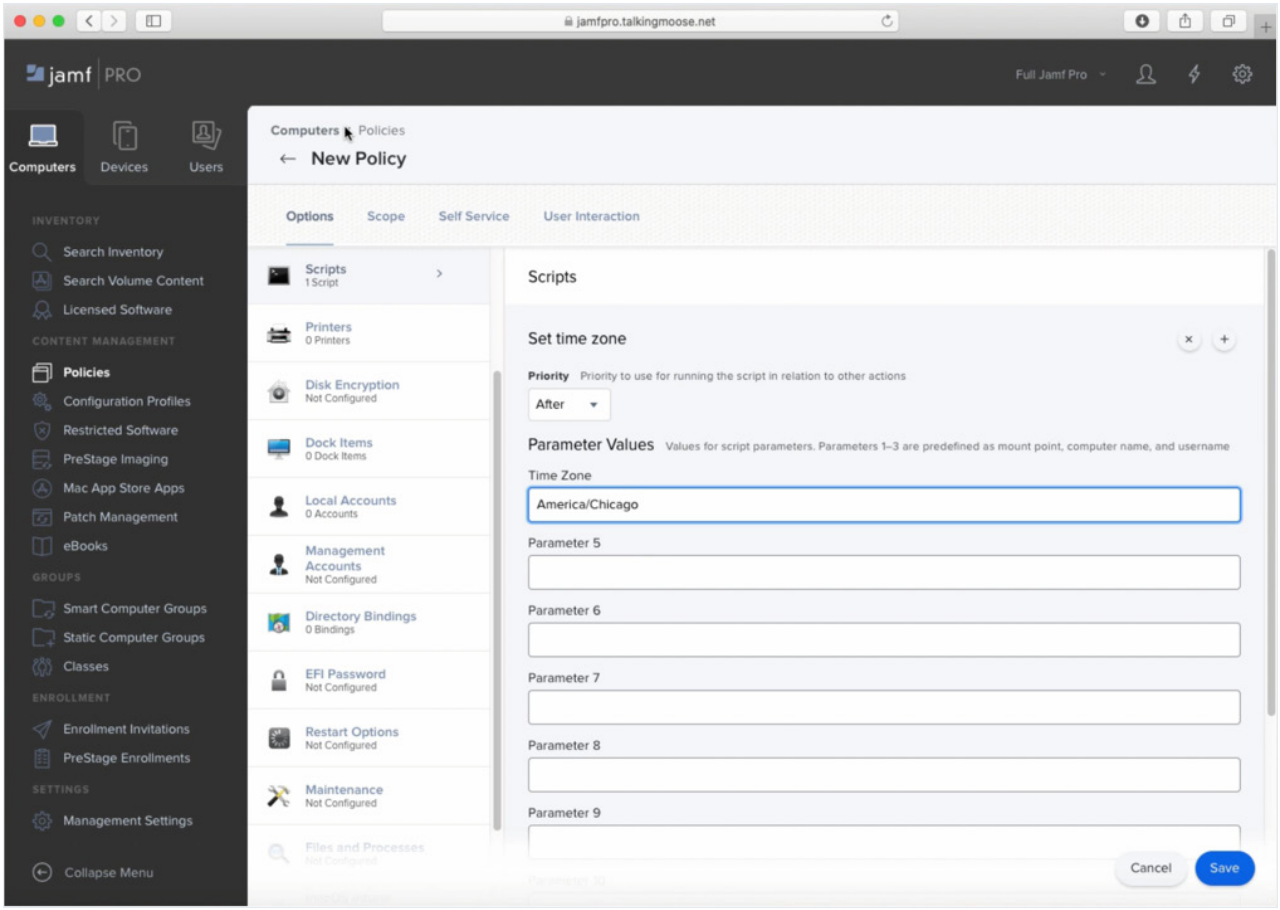
Erstellung einer Richtlinie

Lassen Sie uns nun eine Richtlinie erstellen, um unser Skript auszuführen.

- Wählen Sie in Ihrer Jamf-Instanz aus dem linken Menü "Richtlinien" und dann die Kategorie. In diesem Beispiel fügen wir sie der Kategorie "Bereitstellung" hinzu.
- Nennen Sie sie "Zentrale Zeitzone festlegen", und fügen Sie sie der Kategorie "Bereitstellung" hinzu.



- Legen Sie einen benutzerdefinierten Auslöser fest, damit ich diese Richtlinie später in einem Skript namentlich aufrufen kann, z. B. setcentraltimezone.
- Wählen Sie die Option Skripte und fügen Sie die Zeitzone ein, die Sie aus dem Terminal-Befehl kopiert haben.



Beachten Sie, dass in diesem Feld die Bezeichnung angezeigt wird, die wir beim Einfügen des Skripts auf der Registerkarte Optionen hinzugefügt haben.
Das Etikett sagt Ihnen, was Sie hier eintragen sollen.

Jetzt müssen Sie die Richtlinie nur noch einrichten und speichern.

Jetzt haben Sie eine neue Richtlinie in Ihrer Kategorie "Bereitstellung".

Mit Skript-Parametern können wir etwas wirklich Cooles machen.

Da das Skript so geschrieben wurde, dass es Skriptparameter akzeptiert, können Sie es beliebig oft für verschiedene Zeitzonen wiederverwenden. Sie müssen lediglich eine neue Richtlinie für jede Zeitzone erstellen, das gleiche Skript hinzufügen und dann den Wert für die Zeitzone ausfüllen.

Dialoge mit jamfHelper und osascript erstellen

Dialoge sind nicht nur nützlich, um Ihren Endbenutzer*innen eine Nachricht anzuzeigen, sondern auch, um von ihnen Informationen anzufordern.

Es gibt zwei Möglichkeiten, Dialoge zu erstellen, die jeweils ihre eigenen Vorteile haben.

jamfHelper

jamfHelper ist ein Befehlszeilenwerkzeug.

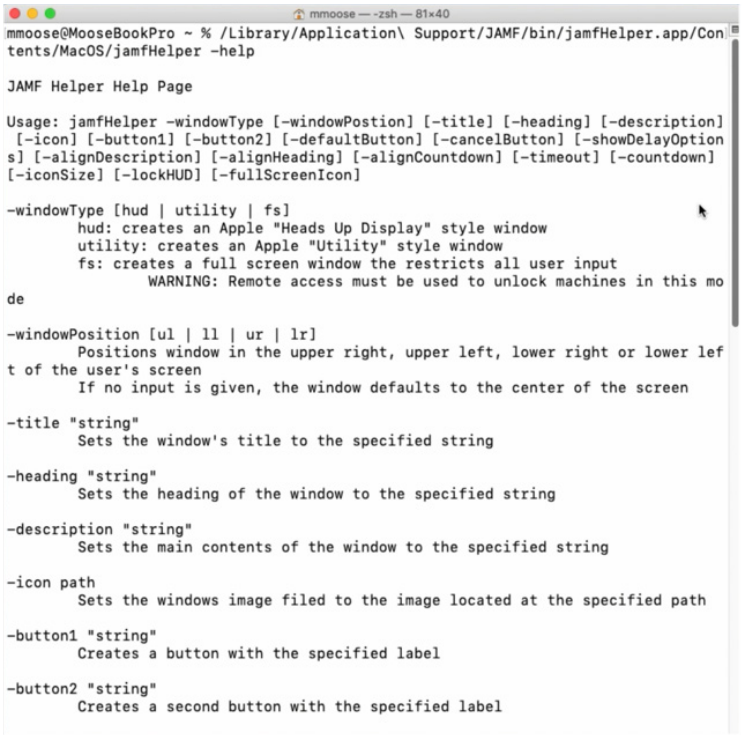
Terminal öffnen. Dann:

- Navigieren Sie zu Bibliothek > App Support > JAMF > bin > jamfHelper
- Klicken Sie mit der rechten Maustaste auf jamfHelper und wählen Sie 'Paketinhalt anzeigen'.
- Navigieren Sie zu Inhalt > macOS, wo Sie das Befehlszeilentool finden.
- Ziehen Sie es in das Terminal.

Fügen Sie am Ende der eingeblendeten Befehlszeile "-help" hinzu und drücken Sie die Eingabetaste:

```
/Library/Application\ Support/JAMF/bin/jamfHelper.app/
Contents/MacOS/jamfHelper -help
```

Hier erfahren Sie alles, was Sie wissen müssen, um jamfHelper zu benutzen.



Da der Pfad zu jamfHelper so lang ist, können Sie ihn mit diesem Skript in einen kürzeren Variablennamen packen, der nur jamfHelper heißt:

```
#!/bin/zsh
```

```
jamfHelper="/Library/Application
Support/JAMF/bin/jamfHelper.app/Contents/MacOS/
jamfHelper"
```

Um jamfHelper aufzurufen, müssen Sie von nun an nur noch ein Dollarzeichen vor den Variablennamen setzen und ihn in Anführungszeichen setzen:

```
"$jamfHelper"
```

Hier ist ein netter Trick, um den richtigen Pfadnamen zu erhalten. Klicken Sie mit der rechten Maustaste auf jamfHelper und zeigen Sie auf "Kopieren", drücken Sie dann aber auch die Optionstaste, um den Pfad zu kopieren. In diesem Skript müssen Sie nur den Pfad zwischen den Anführungszeichen einfügen.

Was kann jamfHelper tun?

Nehmen wir an, Sie möchten dem jamfHelper einige Optionen hinzufügen.
Die erste ist die Definition eines Fenstertyps — es werden drei Typen unterstützt.
Beginnen wir mit dem "Heads-Up-Display".
Fügen Sie -windowType zu Ihrem Skript hinzu und geben Sie die Art des Fensters an: "hud".

```
"$jamfHelper" -windowType hud  \
```

Ein Backslash am Ende einer Zeile erlaubt es Ihnen, einen normalerweise sehr langen Befehl auf mehrere Zeilen zu verteilen.

Fügen Sie dann eine Überschrift hinzu. Dies muss in Anführungszeichen stehen:

```
-Überschrift "Vorbereiten des Computers..." \
```

Fügen Sie dann eine Beschreibung hinzu.

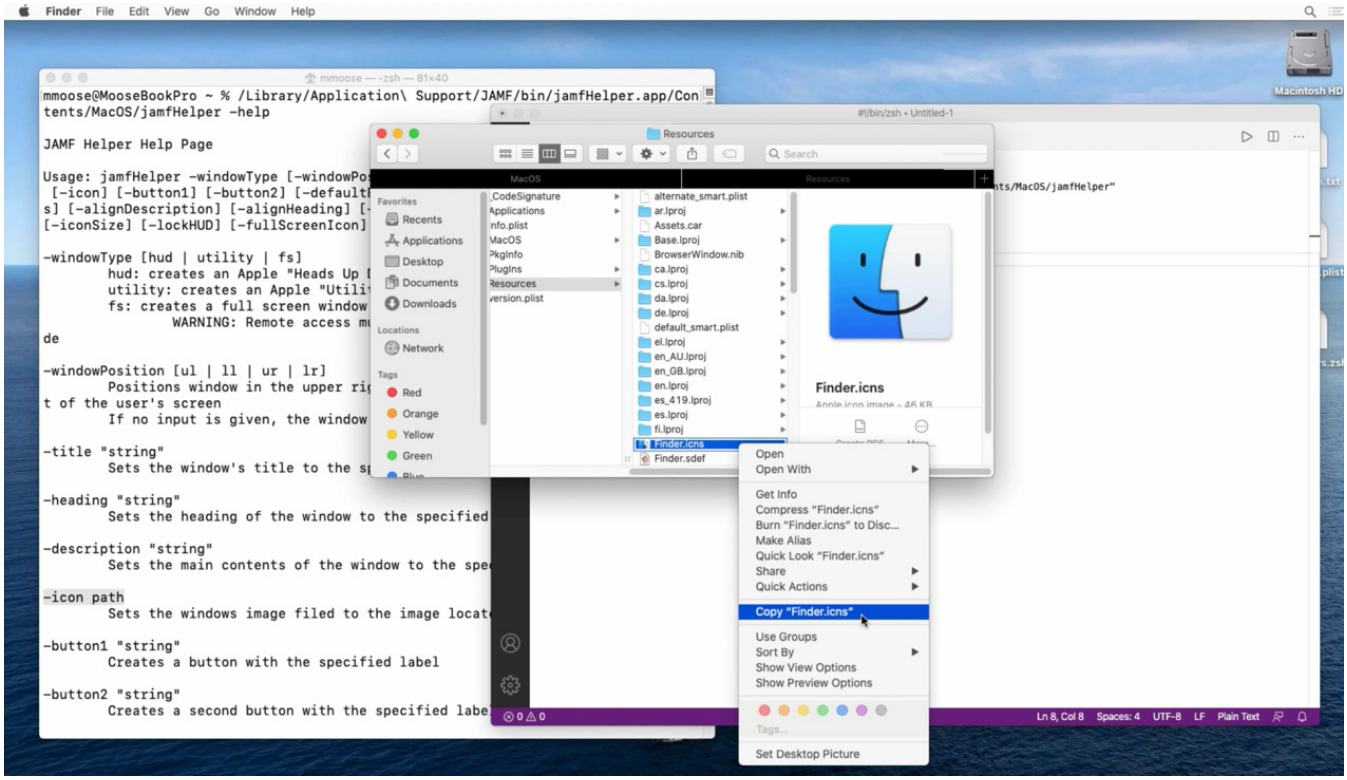
```
-description "Installation von Microsoft Office 2019"\
```

Wenn Sie das Dialogfeld interessanter gestalten wollen, fügen Sie ein Symbol hinzu. Wählen Sie ein Symbol und geben Sie dessen Pfad hier ein. Setzen Sie den Pfad in Anführungszeichen, falls er Leerzeichen enthält.

```
-icon "/System/Library/CoreServices/Finder.app/
Contents/Resources/Finder.icns"
```

So finden Sie bereits vorhandene Symbole auf Ihrem Mac

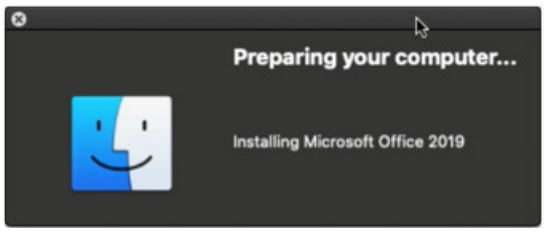
- Navigieren Sie im Finder zu System > Bibliothek > CoreServices und suchen Sie die Finder-App.
- Klicken Sie mit der rechten Maustaste auf den Finder, wählen Sie "Paketinhalt anzeigen" und dann Inhalt > Ressourcen
- Wählen Sie das Finder-Symbol.
- Klicken Sie mit der rechten Maustaste auf das Finder-Symbol, halten Sie die Optionstaste gedrückt und kopieren Sie den Pfad.
- Fügen Sie dann den Pfad in Ihr Skript ein.



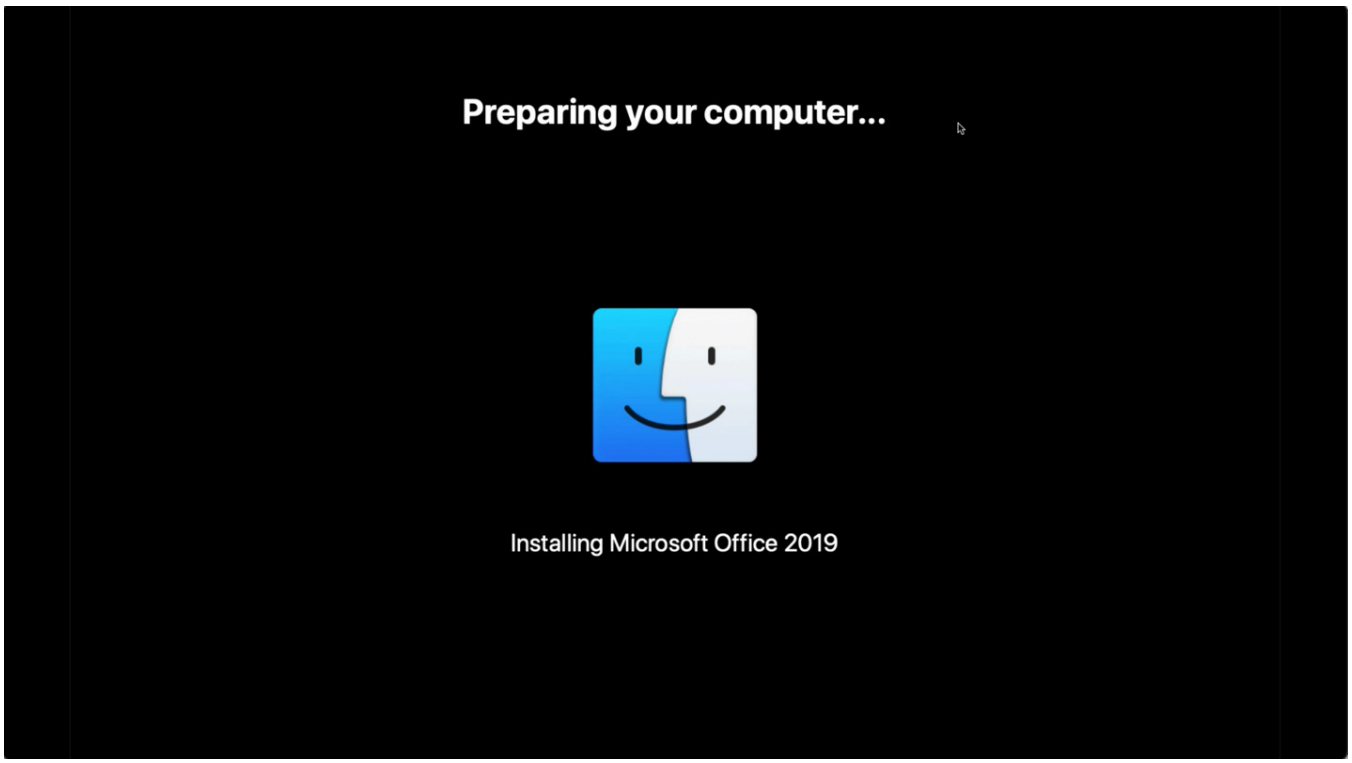
So sieht das Ganze zusammen aus:

```
#!/bin/zsh

jamfHelper="/Library/Application
Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper"

"$jamfHelper" -windowType hud  \
-heading "Preparing your computer..." \
-description "installing Microsoft Office 2019"\
-icon "/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns"
```

Führen Sie Ihr Skript direkt von Ihrem Skript-Editor aus.
Sie erhalten einen sehr einfachen Dialog:



Um ein Vollbild-Dialogfeld zu erhalten, ändern Sie den Fenstertyp von "hud" in "fs",
, was für "Full Screen" (Vollbild) steht, und führen Sie es erneut aus:



Ein Vollbilddialog wie dieser ist eine gute Möglichkeit, um zu verhindern, dass die Person, die vor dem Computer sitzt, diesen während der Bereitstellung benutzt. Es kann auch aktualisiert werden, um den Benutzer über den Fortschritt zu informieren, wenn Aufgaben wie die Installation von Software hinter ihm laufen.

Wenn Sie zu irgendeinem Zeitpunkt sehen möchten, was hinter dem Dialog passiert, drücken Sie einfach Befehl + q, um den Vorgang zu beenden.

Der osascript-Befehl

Sie können Dialoge auch über AppleScript mit dem Befehl osascript im Terminal erstellen.

Er kann etwas, was jamfHelper nicht kann.

Erstellen Sie zunächst einen Befehl für einen "Auswahl"-Dialog, der drei Abteilungsnamen enthält.

Sie müssen diesen Befehl in doppelten Anführungszeichen beginnen und beenden, und wenn Sie auch doppelte Anführungszeichen als Teil des Befehls haben, müssen Sie ihnen Backslashes voranstellen, um sie als buchstäbliche doppelte Anführungszeichen zu behandeln - nicht als die doppelten Anführungszeichen, die den Befehl umgeben:

```
asCommand="choose from list {\"Buchhaltung\", \"Sales\", \"kiosk\"}
```

Fügen Sie dann eine Meldung für Ihre Endbenutzer*innen hinzu, die erklärt, was der Mac gerade tut:

```
mit der Aufforderung \"Hallo, bereiten wir Ihren Mac vor.
Um zu beginnen, wählen Sie unten Ihre Abteilung...\"
```

Der letzte Teil des Befehls fügt dem Dialogfenster einen Titel hinzu. Denken Sie daran, den Befehl mit einem doppelten Anführungszeichen zu beenden:

```
mit dem Titel \"Bereiten Sie Ihren Mac vor\""
```

In dieser Zeile wird osascript angewiesen, den Befehl auszuführen, und das Ergebnis wird in die Variable "Abteilung" eingetragen:

```
department=$( /usr/bin/osascript -e "$asCommand" )
```

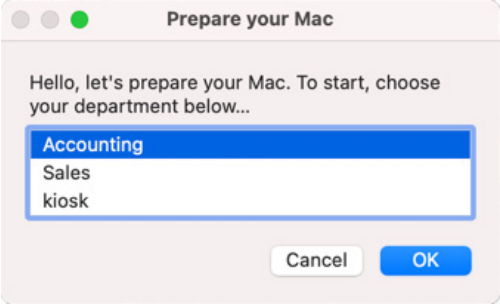
Zuletzt geben Sie die Variable "Abteilung" aus, um anzuzeigen, was der Benutzer*innen ausgewählt hat:

```
echo "$Abteilung"
```

So sieht das gesamte Skript aus:

```
#!/bin/zsh

asCommand="Wählen Sie aus der Liste {\"Abteilung\", \"Sales\", \"Kiosk\"} mit der
Aufforderung \ "Hallo, bereiten wir Ihren Mac vor. Um zu beginnen, wählen Sie unten
Ihre Abteilung aus...\" mit dem Titel \"Bereiten Sie Ihren Mac vor\""

department=$( /usr/bin/osascript -e "$asCommand" )

echo "$Abteilung"
```

Testen Sie nun das Skript in Ihrem Skript-Editor. Hier wählt der Benutzer*innen "Buchhaltung", und dies ist das Ergebnis, das er sehen wird:



AppleScript und osascript können noch viel mehr mit Dialogen machen, aber dies ist ein guter Anfang.

Alles zusammenfügen

Hier ist ein Skript, das Sie zum Self Service hinzufügen können, damit ein neuer Mitarbeiter*innen beim Onboarding seinen Mac durch die Auswahl seiner Abteilung bereitstellen kann.

Zunächst werden der vollständige Name und der Benutzername des angemeldeten Benutzers erfasst. Es verwendet dann den vollständigen Namen in der osascript-Eingabeaufforderung, um eine Abteilung auszuwählen:

```
jamfHelper="/Library/Application Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper"

currentUser=$( /usr/bin/stat -f "%Su" /dev/console )

fullName=$( /usr/bin/id -F "$currentUser" ) # z.B. "mmoose"

echo "Bereitstellender Benutzer ist $fullName ($currentUser)"
```

Dann wird der Benutzer*innen aufgefordert, eine Abteilung zu wählen:

```
asCommand="Wählen Sie aus der Liste {\"Buchhaltung\", \"Verkauf\", \"Kiosk\"} mit der Aufforderung\"Hallo, $fullName! Lassen Sie uns Ihren Mac vorbereiten. Um zu beginnen, wählen Sie unten Ihre Abteilung aus...\"mit dem Titel \"Bereiten Sie Ihren Mac vor\""

department=$( /usr/bin/osascript -e "$asCommand" )

echo "Bereitgestellt für Abteilung $Abteilung"
```

Ein weiteres osascript fragt nach dem Asset-Tag des Computers:

```
asCommand="text returned of (Dialogfeld \"Geben Sie das Asset-Tag dieses Macs ein (siehe
unten am Computer)...\" Standardantwort \"\" mit Titel \"Bereiten Sie Ihren Mac vor\")"

assetTag=$( /usr/bin/osascript -e "$asCommand" )

echo "Geräte-Asset-Tag ist $assetTag"
```

Das Skript wird dann die Zeitzone für alle Computer einstellen:

```
/usr/local/bin/jamf policy -event settimezonechicago
```

Anschließend wird die gewählte Abteilung bewertet und die erforderliche Software installiert:

```
if [[ "$Abteilung" = "Buchhaltung" ]]; thenecho
          "Bereitstellung dieses Macs für die Buchhaltung"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installoffice

elif [[ "$Abteilung" = "Vertrieb" ]]; dann

          echo "Bereitstellung dieses Macs für den Verkauf"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installoffice
          /usr/local/bin/jamf policy -event installzoom

sonst

          echo "Bereitstellung dieses Macs als Kiosk"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installzoom

fi
```

Sobald die Software installiert ist, führt sie eine Aktualisierung des Bestands durch, die das Asset-Tag, die Abteilung und den Benutzernamen des Computers enthält:

```
"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Updating inventory" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/local/bin/jamf recon -assetTag "$assetTag" -department "$department"
-endUsername "$currentUser"
```

Anschließend wird in der Bibliothek ein Ordner für administrative Tools und Informationen erstellt:

```
"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Creating admin folder" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/bin/mkdir -p "/Library/Talking Moose Industries"
```

Und hinterlassen Sie eine Quittung für die Bereitstellung:

```
"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Writing provisioning receipt" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisiondate -date $( /bin/date "+%Y-%m-%d" )
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisioner -string "$currentUser"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
Abteilung -string "$Abteilung"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
assettag -string "$assetTag"
```

Schließlich informiert es die Benutzer*innen und startet dann den Computer neu:

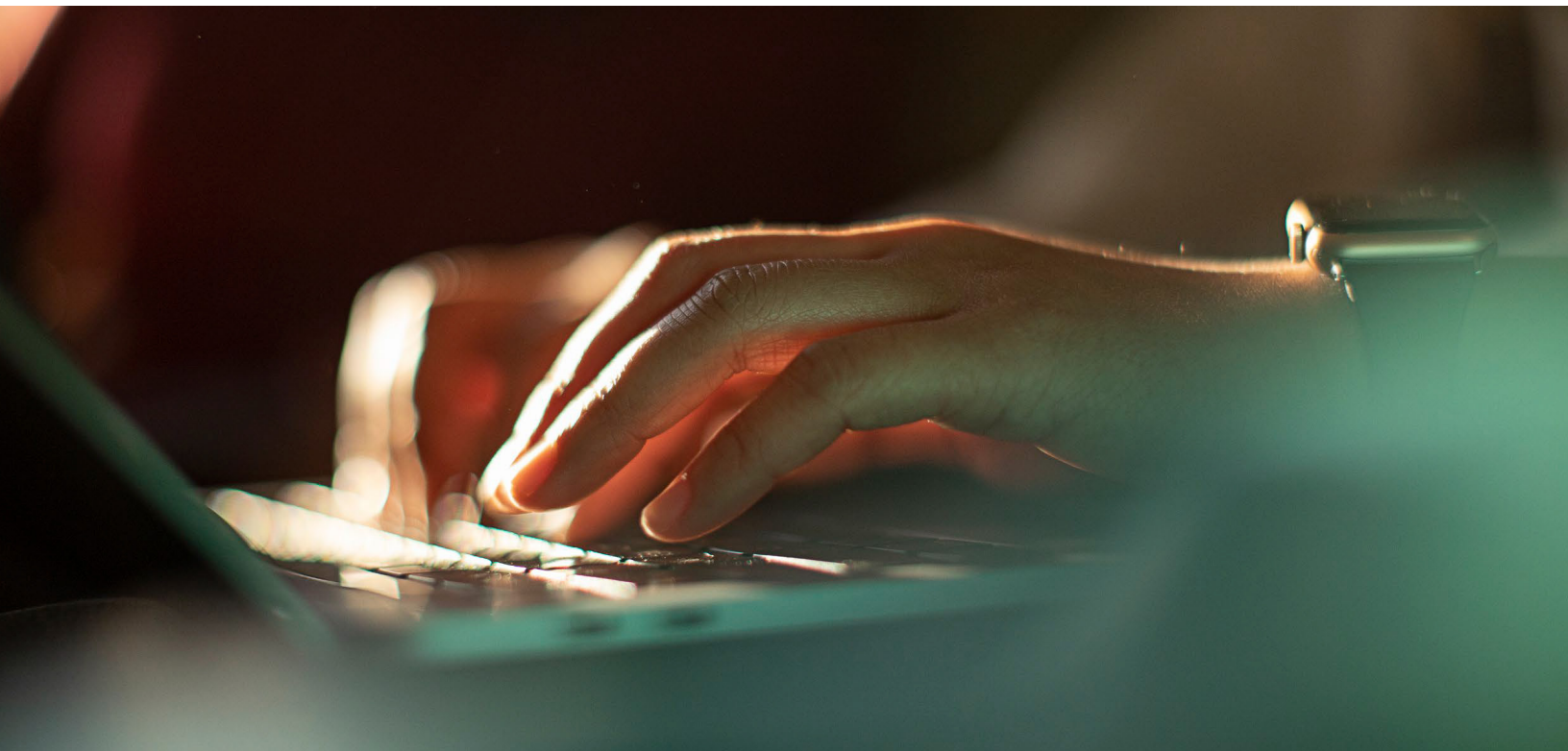
```
"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Neustart des Macs in einer Minute" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &
/sbin/shutdown -r +1 &

Ausgang 0
```

Vollständiges Skript, mit eingebetteten Deskriptoren:

```
#!/bin/zsh

# langen Pfad zu JamfHelper einer kürzeren Variablen zuweisen

jamfHelper="/Library/Application
Support/JAMF/bin/jamfHelper.app/Contents/MacOS/jamfHelper"\

# Informationen über den aktuellen Benutzer abrufen

currentUser=$( /usr/bin/stat -f "%Su" /dev/console )
fullName=$( /usr/bin/id -F "$currentUser" ) # z.B. "Martin Moose"

echo "Bereitstellender Benutzer ist $fullName ($currentUser)"

# AppleScript-Befehl, um den aktuell angemeldeten Benutzer aufzufordern, eine Abteilung
auszuwählen

asCommand="Wählen Sie aus der Liste {\"Buchhaltung\", \"Verkauf\", \"Kiosk\"} mit der
Aufforderung\"Hallo, $fullName! Lassen Sie uns Ihren Mac vorbereiten. Um zu beginnen, wählen
Sie unten Ihre Abteilung aus...\"mit dem Titel \"Bereiten Sie Ihren Mac vor\""

# den Befehl ausführen

department=$( /usr/bin/osascript -e "$asCommand" )

echo "Bereitgestellt für Abteilung $Abteilung"

# AppleScript-Befehl, um den aktuell angemeldeten Benutzer aufzufordern, ein Asset-Tag
einzugeben

asCommand="text returned of (Dialogfeld \"Geben Sie das Asset-Tag dieses Macs ein (siehe
unten am Computer)...\" Standardantwort \"\" mit Titel \"Bereiten Sie Ihren Mac vor\")"

# den Befehl ausführen

assetTag=$( /usr/bin/osascript -e "$asCommand" )

echo "Geräte-Asset-Tag ist $assetTag"
```

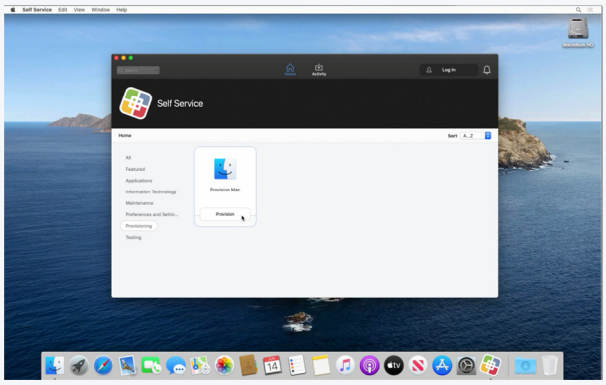
```
# Diesen Mac für die ausgewählte Abteilung bauen

# globale Einstellungen

/usr/local/bin/jamf policy -event settimezonechicago

if [[ "$Abteilung" = "Buchhaltung" ]]; then

          echo "Bereitstellung dieses Macs für die Buchhaltung"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installoffice

elif [[ "$Abteilung" = "Vertrieb" ]]; dann

          echo "Bereitstellung dieses Macs für den Verkauf"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installoffice
          /usr/local/bin/jamf policy -event installzoom

sonst

          echo "Bereitstellung dieses Macs als Kiosk"

          /usr/local/bin/jamf policy -event installchrome
          /usr/local/bin/jamf policy -event installzoom

fi

# Jamf Pro-Inventar aktualisieren

"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Updating inventory" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/local/bin/jamf recon -assetTag "$assetTag" -department "$department"
-endUsername "$currentUser"

# Admin-Ordner erstellen

"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Creating admin folder" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/bin/mkdir -p "/Library/Talking Moose Industries"

# Bereitstellungsbeleg erstellen
```

```
"$jamfHelper" -windowType "fs" \
-heading "Preparing your Mac..." \
-description "Writing provisioning receipt" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisiondate -date $( /bin/date "+%Y-%m-%d" )
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisioner -string "$currentUser"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
Abteilung -string "$Abteilung"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
assettag -string "$assetTag"

"$jamfHelper" -windowType "fs" \
-Überschrift "Vorbereiten des Macs..." \
-Beschreibung "Neustart des Macs in einer Minute" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

# Neustart des Macs
/sbin/shutdown -r +1 &

Ausgang 0
```

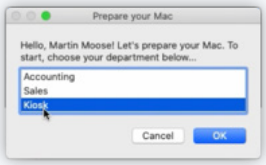


Mal sehen, wie das alles aussieht!

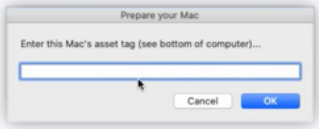
Wenn Self Service geöffnet ist, starten Sie das Verfahren, indem Sie auf die Schaltfläche Bereitstellung klicken.



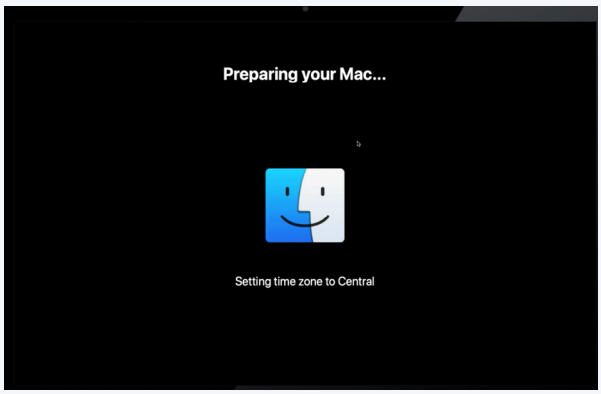
SCHRITT 1.



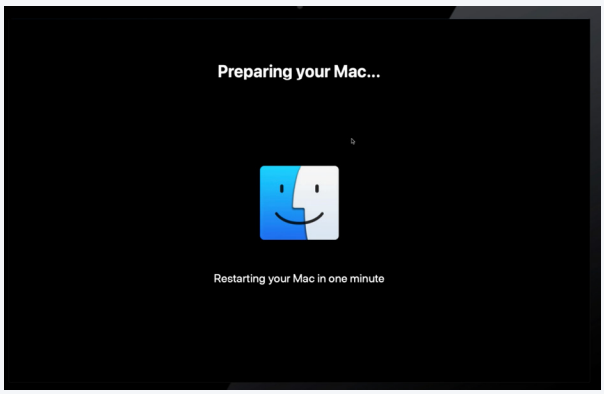
SCHRITT 2.



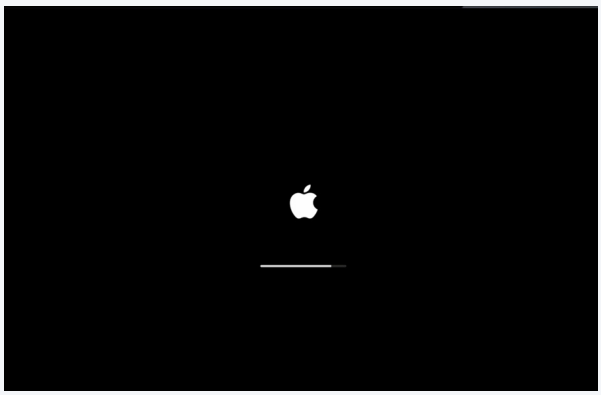
SCHRITT 3.



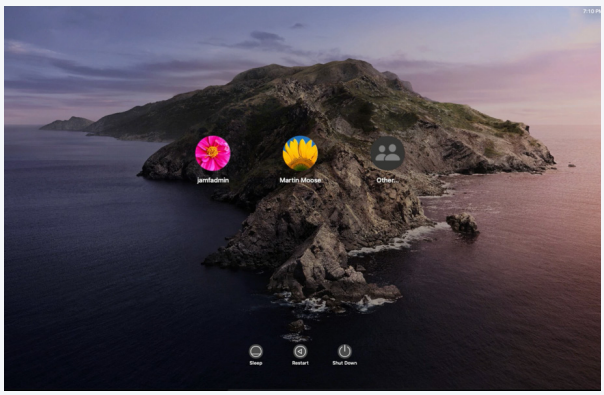
SCHRITT 4.



SCHRITT 5.



SCHRITT 6.

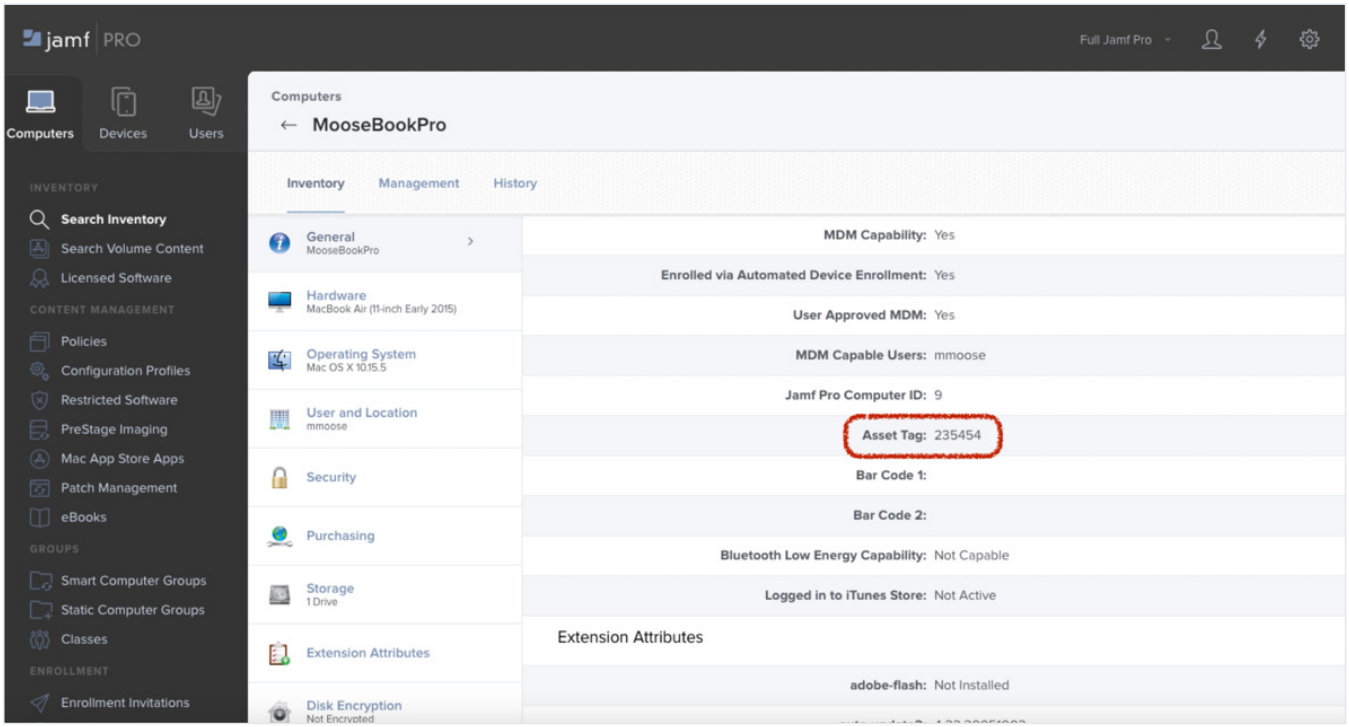


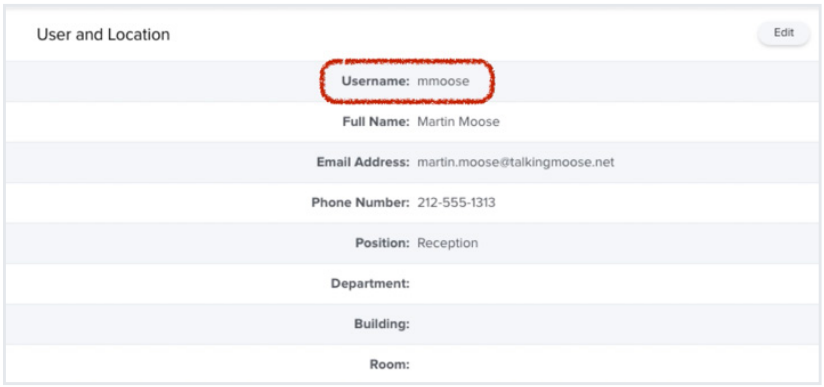
SCHRITT 7.

Wenn die Bereitstellung abgeschlossen ist, wird der Computer neu gestartet und der Mac befindet sich wieder im Anmeldefenster und ist einsatzbereit.

Nachdem sich die Benutzer*innen angemeldet haben, wird ihre Software installiert und sie können ihre Bereitstellungsbestätigung im Bibliotheksordner sehen.

Wenn der Mac-Administrator den Computerdatensatz in Jamf Pro anschaut, wird das Asset-Tag aus dem osascript-Dialogfeld ausgefüllt, das Feld Benutzername wird ausgefüllt - was es dem Administrator ermöglicht, einen LDAP-Lookup durchzuführen und zusätzliche Informationen wie E-Mail-Adresse, Telefonnummer, Hauptbenutzername und andere Details aus Active Directory zu ziehen.





Und das war's!

Weitere Ressourcen und Möglichkeiten zum Lernen

Jamf's Trainingskatalog

Vergessen Sie beim Skripten nicht, dass alle Jamf Kunden Zugang zu unserem gesamten Katalog mit mehr als 15 Stunden kurzer Anleitungsvideos haben, die alles abdecken, vom Helpdesk über den Ingenieur bis hin zum Administrator in: **trainingcatalog.jamf.com** (auf Englisch)

Sehen Sie sich die Skripting Series mit 15 Modulen und Videolektionen an, die jeweils weniger als 30 Minuten in Anspruch nehmen. So können Sie das Skripting weiter erlernen.

Github

Erkunden Sie die Open-Source Community auf **GitHub.com** (auf Englisch). Open-Source Software ist kostenlos, und Sie finden hunderte von Skripten und Projekten auf der GitHub-Seite von Jamf: **github.com/jamf** (auf Englisch)

Ressourcen des Autors Bill Smith

Wenn Sie viele Beispiele für kurze und lange Skripte für eine Vielzahl von Anforderungen sehen möchten, sehen Sie sich das Projektarchiv (gist repository) von Bill Smith **auf GitHub an**. Hier finden Sie das Beispiel-Provisioning-Skript aus diesem Whitepaper und einen Teil des anderen Codes, den er hier ebenfalls verwendet hat: **https://gist.github.com/talkingmoose** (auf Englisch)

Wir hoffen, dass dieser Leitfaden Ihnen geholfen hat, Ihre Skript-Fähigkeiten auf die nächste Stufe zu heben.

Skripte in Verbindung mit einem guten Unternehmensverwaltungssystem können den Unterschied zwischen stundenlanger Arbeit und einem Klick auf eine Schaltfläche ausmachen.