Scripting Apple avec Jamf :

Un guide d'approfondissement pour automatiser les tâches courantes

Vous êtes un administrateur Apple et vous souhaitez approfondir vos compétences en script acquises avec L'introduction au scripting ? Ce guide est fait pour vous !

Vous pouvez automatiser trois tâches informatiques essentielles de la gestion des appareils Apple à l'aide de trois outils de script aussi simples que puissants :

Écrire du texte dans un fichier et le récupérer pour une utilisation ultérieure.

Utiliser les paramètres de script dans Jamf Pro

Créer des dialogues interactifs avec jamfHelper et osascript

Écrire et lire un fichier

Il arrive parfois que vous deviez laisser des informations sur un Mac pour y accéder plus tard. Un reçu d'approvisionnement, par exemple, qui indiquera à quel moment un Mac a été préparé et déployé, et qui l'a configuré.

Voici quelques moyens de le faire :

À l'aide du terminal

Vous vous souvenez de la commande « echo » de notre webinaire Scripting 101 ?
Cette commande rappelle, ou affiche, ce que vous avez saisi. Si vous saisissez :

```
echo 'Hello, World !'
```

...et que vous appuyez sur Entrée, la commande renvoie :

```
Hello, World !
```

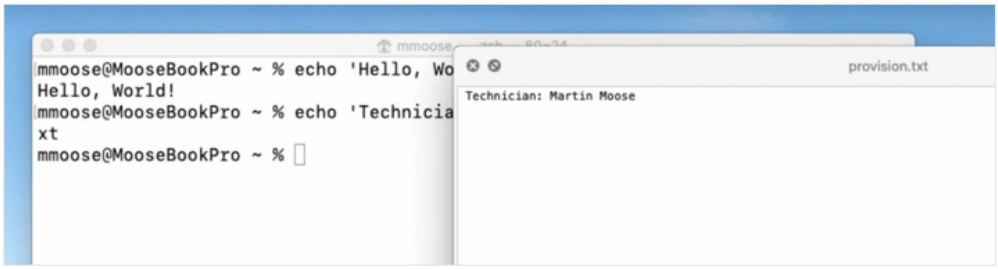
Sauvegarder un contenu dans un fichier texte avec la commande echo

En utilisant le signe >>, appelé symbole de « redirection », vous pouvez enregistrer un contenu dans un fichier. Si le fichier n'existe pas, le symbole de redirection le créera. S'il existe déjà, il ajoutera ce que vous voulez à la fin du fichier.
(Si vous utilisez un seul symbole de redirection, >, le contenu du fichier sera écrasé).

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

Lorsque vous appuyez sur Entrée, le fichier apparaît sur votre Bureau.

Vous pouvez sélectionner le fichier et appuyer sur la barre d'espace pour l'afficher dans QuickLook.



Ajouter la date du jour dans un fichier

Pour automatiser l'ajout de la date du jour dans un fichier, utilisez la commande :

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

La portion « %y-%m-%d » correspond à l'année, au mois et à la date.

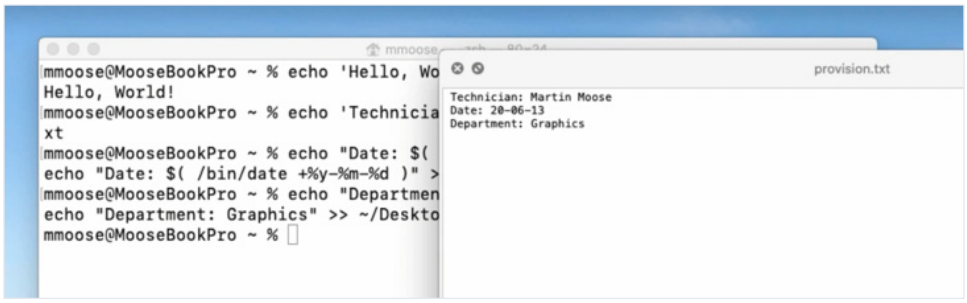
Pour vous épargner quelques frappes, utilisez un raccourci qui répète le dernier argument de la commande précédente afin de rediriger le texte dans le fichier :

```
>> !$
```

Écrivez ensuite le nom du service dans le fichier :

```
echo Service : Graphisme >> !$
```

Regardons notre fichier : tout notre texte est là, dans l'ordre où nous l'avons ajouté :



Pour le relire, utilisez la commande « cat » suivie du chemin d'accès au fichier :

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

Cette commande affiche le contenu de ce fichier directement dans le terminal :

```
Technicien : Martin Moose
Date : 20-06-13
Service : Graphisme
```

Révision

`echo`	Affiche ce que vous saisissez dans la fenêtre du terminal
`>>`	Redirige la sortie vers un fichier/l'ajoute au fichier existant
`>`	Redirige la sortie vers un fichier/écrase le fichier existant
`$( commande )`	Exécute une commande
`!$`	Répète le dernier argument
`cat`	Lit un fichier

Commande par défaut

La commande « cat » vous permet de lire l'ensemble du fichier. Mais comment faire si vous ne voulez qu'une seule information ?

La commande « defaults » est la même que celle que vous utiliseriez pour lire les plists dans votre dossier de préférences. Elle peut servir à lire des informations.

Et à écrire nos propres plists.

Pour créer un nouveau fichier, utilisez « defaults write » et indiquez-lui l'emplacement du fichier. Ensuite, donnez une description en un mot – « Technicien » – pour indiquer le technicien qui a configuré cet ordinateur, et faites-la suivre du nom du technicien.

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

Appuyez sur Entrée.

Le fichier apparaît sur le bureau. Lorsque vous utilisez QuickLook pour le visualiser, il est très différent du premier :

```
<?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>Technicien</key>
    <string>Martin Moose</string>
<dict>
</plist>
```

La commande « defaults » ajoute beaucoup de formatage, mais si vous regardez bien, les informations sont toutes là.

Ajoutons la date et le service, puis regardons à nouveau le fichier :

```
/usr/bin/defaults write ~/Desktop/provision.plist Technicien 'Martin Moose'
/usr/bin/defaults write ~/Desktop/provision.plist Date $ ( /bin/date '+%y-%m-%d' )
/usr/bin/defaults write ~/Desktop/provision.plist Service 'Graphisme'
```

Maintenant, le fichier fournit les trois informations :

```
<?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>Date</key>
    <string>20-06-13</string>
    <key>Service</key>
    <string>Graphisme</string>
    <key>Technicien</key>
    <string>Martin Moose</string>
</dict>
</plist>
```

Comme il s'agit d'une plist, chaque type d'information – technicien, date et service – est répertorié en tant que « clé », et la valeur correspondante est répertoriée sous chaque clé. (La commande defaults classe automatiquement les clés par ordre alphabétique, quel que soit l'ordre dans lequel vous les ajoutez.)

La plist semble beaucoup plus complexe que notre premier fichier texte, mais vous allez comprendre tout son intérêt :

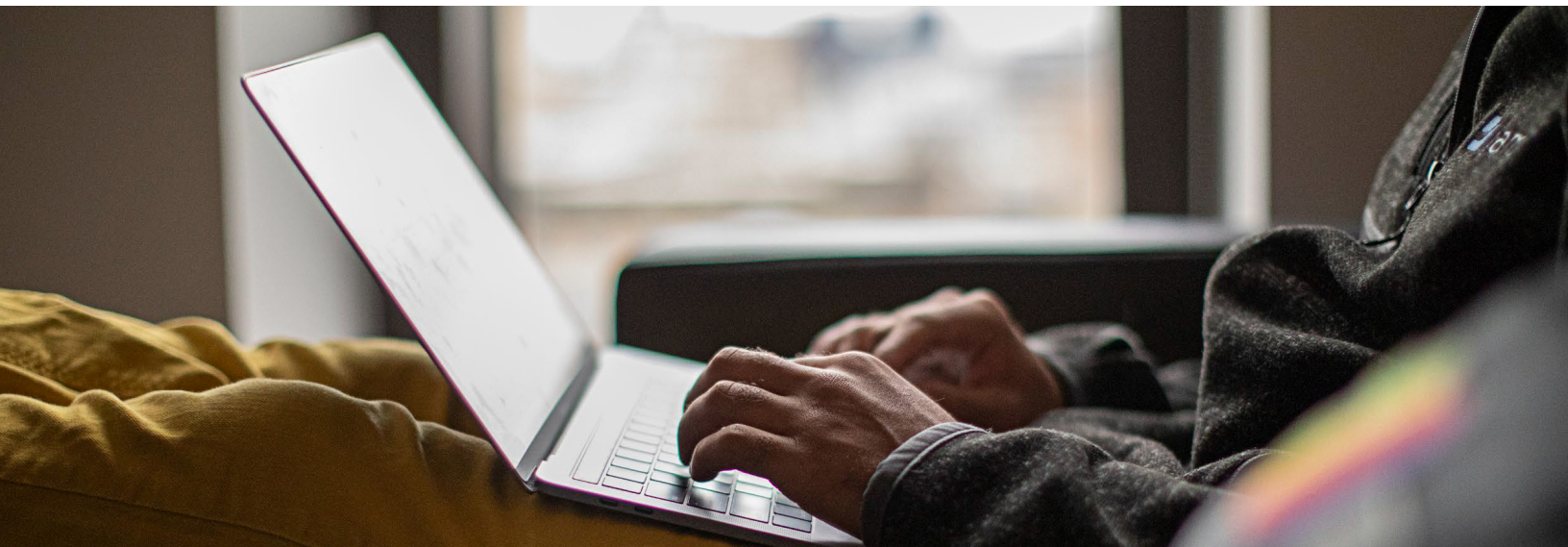
Utilisez « defaults read » pour lire le fichier, puis spécifiez « Technicien », « Date » ou « Service » pour renvoyer la valeur d'un seul élément. C'est particulièrement utile pour obtenir, par exemple, un attribut d'extension. Un administrateur peut utiliser cette commande dans un petit script pour renvoyer le résultat à Jamf Pro.

Exemple :

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

Le terminal ne livre que la date :

```
20-06-13
```



Utiliser les paramètres de script dans Jamf Pro

Que sont les paramètres du script ?

Faisons une démonstration avec un script bref.

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

Le shebang est zsh, également prononcé zi-shell.

Je vais ajouter une instruction echo, qui me donnera juste une ligne vide, à des fins de lisibilité :

```
echo
```

J'ajoute ensuite une autre instruction echo, avec les variables « 1 » à « 5 » en paramètres.

```
echo $1 $2 $3 $4 $5
```

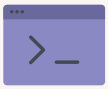
N'oubliez pas : dans un script, quand un élément commence par un « $ », c'est généralement le signe qu'il s'agit d'une variable ou d'un espace réservé.

Le script complet :

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

Enregistrez le script sur le bureau sous le nom parameters.zsh.

Chaque fois que vous créez un nouveau fichier de script, vous devez utiliser une commande dans le terminal appelée « chmod ». Cette commande va rendre votre script exécutable. Sinon, le terminal le considérera comme un simple fichier texte et non comme un script à exécuter.



Utilisez votre éditeur de code préféré ou un de texte brut comme BBEdit ou même TextEdit, fourni avec votre Mac. Désactivez tous les paramètres de correction automatique qui transforment les guillemets droits en guillemets anglais. Vous ne devez utiliser que des guillemets droits. (Un éditeur de script le fait automatiquement pour vous.)

Au début de chaque script, vous devez d'abord saisir ce qu'on appelle le shebang, un dièse suivi d'un point d'exclamation : #! Ajoutez ensuite / bin/zsh (ou bash ou tout autre type de script de votre choix). De cette façon, lorsque vous lancerez vos scripts, le terminal saura qu'il doit utiliser seashell (ou votre type de script) pour l'interpréter.

Saisissez le mot « chmod », qui signifie « changer de mode », puis ajoutez +x, qui signifie « rendre exécutable ».
Faites ensuite glisser votre script dans la fenêtre.

Désormais, lorsque vous exécutez le script suivi de « Ah vous dirais je maman », il renvoie « Ah vous dirais je maman » :

```
chmod +x ~/Desktop/parameters.zsh Ah vous dirais je maman
Ah vous dirais je maman
```

Ce n'est pas très passionnant. Amusons-nous un peu.

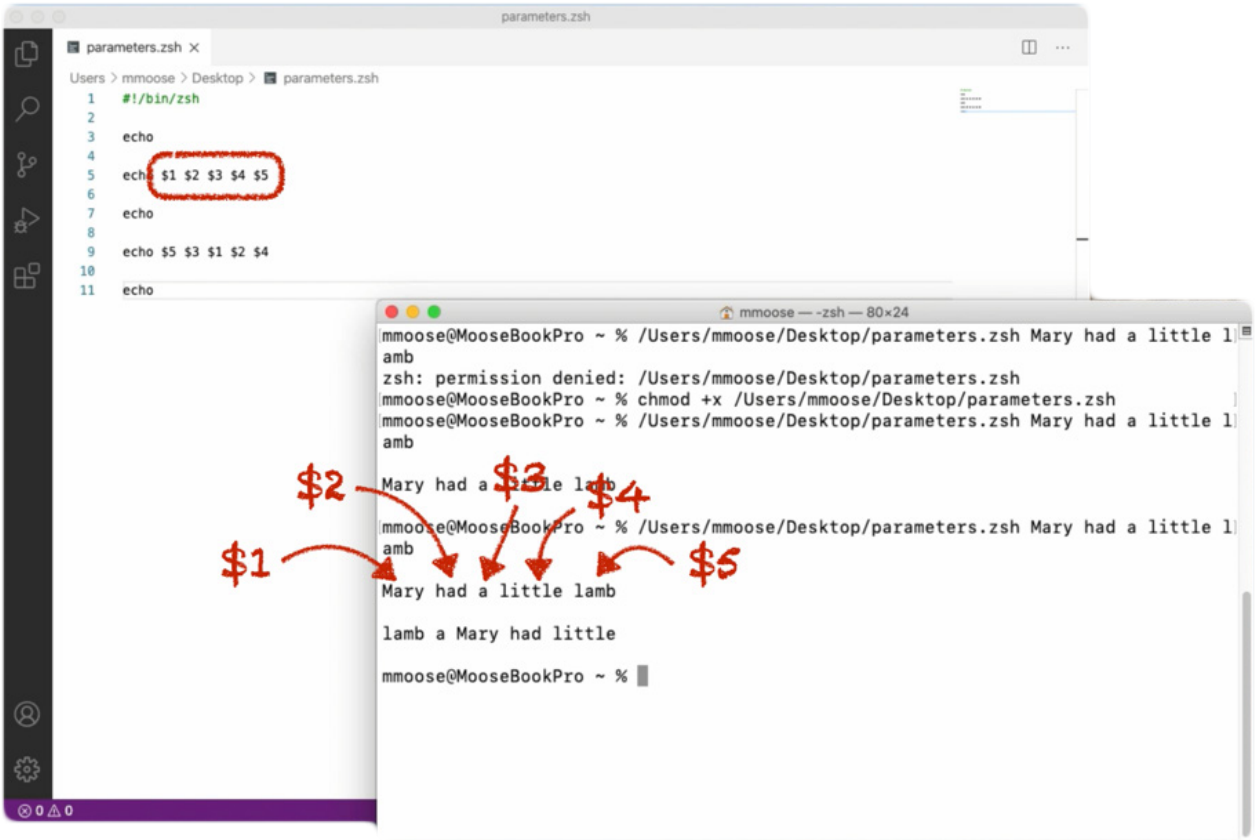
Revenez dans le script et ajoutez une autre commande echo. Mais cette fois, mélangez l'ordre des variables :

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

Enregistrez le script et relancez-le dans le terminal, suivi de suivi de « Ah vous dirais je maman ».

```
~/Desktop/parameters.zsh Ah vous dirais je maman
```

Vous obtenez ce qui suit :



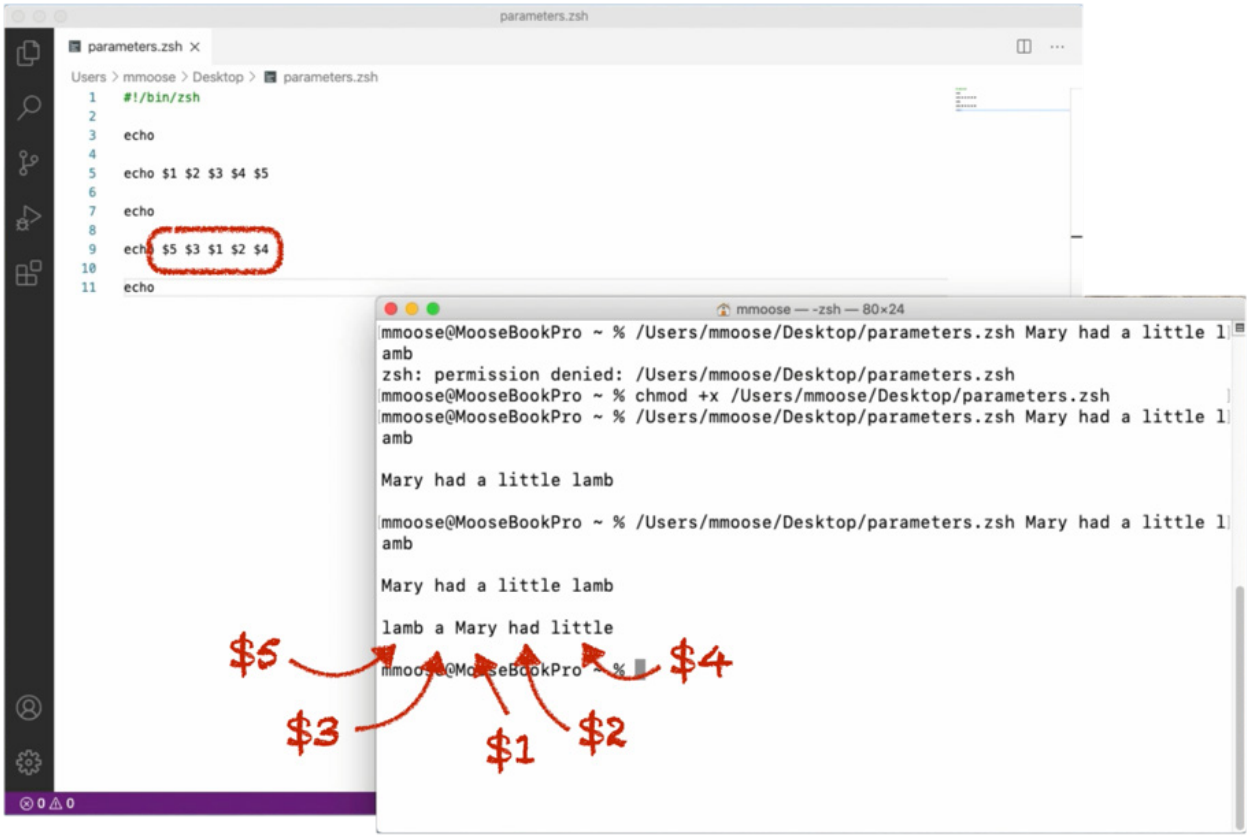
La première ligne semble correcte mais les mots sont mélangés sur la deuxième. Qu'est-ce que cela nous apprend ? Chacune des variables numériques du script – $1-2-3-4-5 – correspond à l'ordre des éléments que nous ajoutons à la fin du script.

« Ah » est le premier mot et correspond à « $1 ».

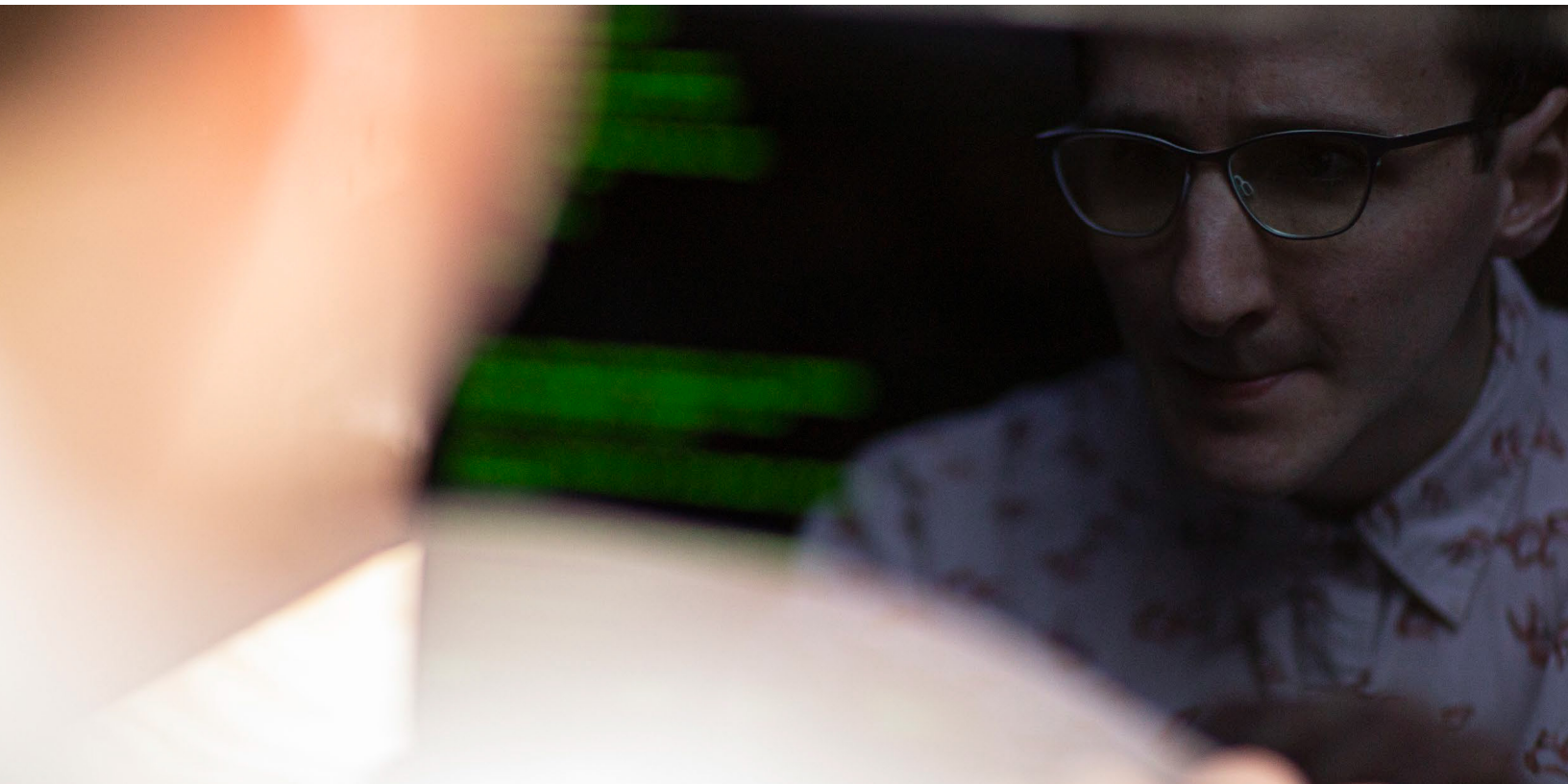
« vous » est le premier mot et correspond à « $2 », et ainsi de suite.

Chacun des mots de « Ah vous dirais je maman » est un « paramètre de script ». Et nous pouvons référencer chaque paramètre de script par sa position à la suite du script.

Si vous changez l'ordre des variables dans le script, l'ordre des mots change lorsque vous exécutez le script.



Nous avons compris que les paramètres de script concernent l'ordre des éléments qui suivent un script. Mais comment pouvons-nous utiliser cela dans Jamf Pro ? Examinons cela de plus près.



Trouver des scripts existants

Vous trouverez toutes sortes de scripts créés par d'autres administrateurs Jamf Pro sur jamfnation.com : Ressources ➔ Compléments Jamf Pro ➔ Scripts.

À partir de là, vous pouvez naviguer ou utiliser la fonction de recherche.

Dans cet exemple, nous allons utiliser un script paramétré « time zone » (fuseau horaire). Voilà ce que vous allez faire :

- Téléchargez le script
- Ouvrez le fichier
- Sélectionnez tout le contenu
- Copiez-le

Dans votre instance Jamf Pro, rendez-vous dans :

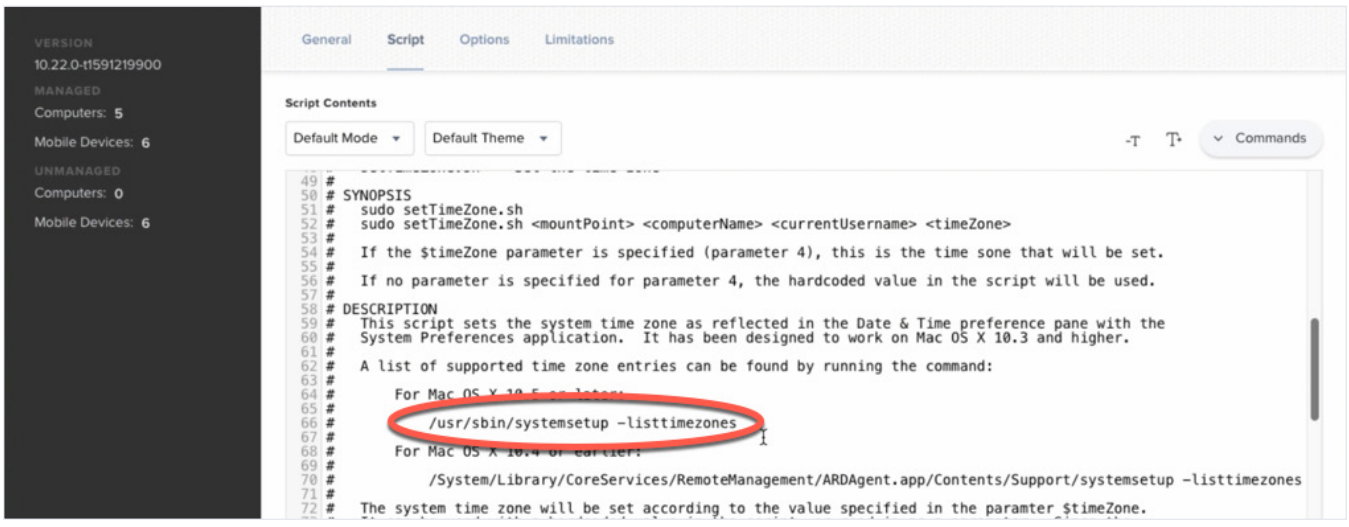
Paramètres > Gestion des ordinateurs > Scripts > Nouveau script

- Nommez le nouveau script « Régler le fuseau horaire »
- Sous l'onglet Script, collez le script.
- Sous l'onglet Options, cliquez dans le champ « Paramètre 4 » et nommez le libellé. Par exemple : « Fuseau horaire ». Vous pouvez lui donner le nom que vous voulez, mais il vaut mieux être descriptif. Dans notre cas, lorsque vous exécutez le script, vous définissez le fuseau horaire que vous souhaitez dans le quatrième paramètre ou « $4 »
- Enregistrez.
- Sélectionnez à nouveau l'onglet Script.
- Trouvez la commande permettant de lister les fuseaux horaires pour les copier.

Mais pourquoi commence-t-on par le paramètre « 4 » et non « 1 » ? Regardez bien les petits caractères juste au-dessus des libellés des paramètres : le texte indique « Les paramètres 1 à 3 sont prédéfinis comme point de montage, nom d'ordinateur et nom d'utilisateur ».

Autrement dit, Jamf Pro enverra toujours ces informations en tant que trois premiers paramètres, que le script les utilise ou non.

Vous n'avez pas de contrôle sur ces paramètres, mais vous en avez sur le reste. Vous pouvez définir 8 paramètres supplémentaires comme bon vous semble.



Comment exécuter la commande

- Ouvrez le terminal
- Saisissez « sudo », ce qui signifie « exécuter avec des privilèges supérieurs », puis collez la commande suivante :

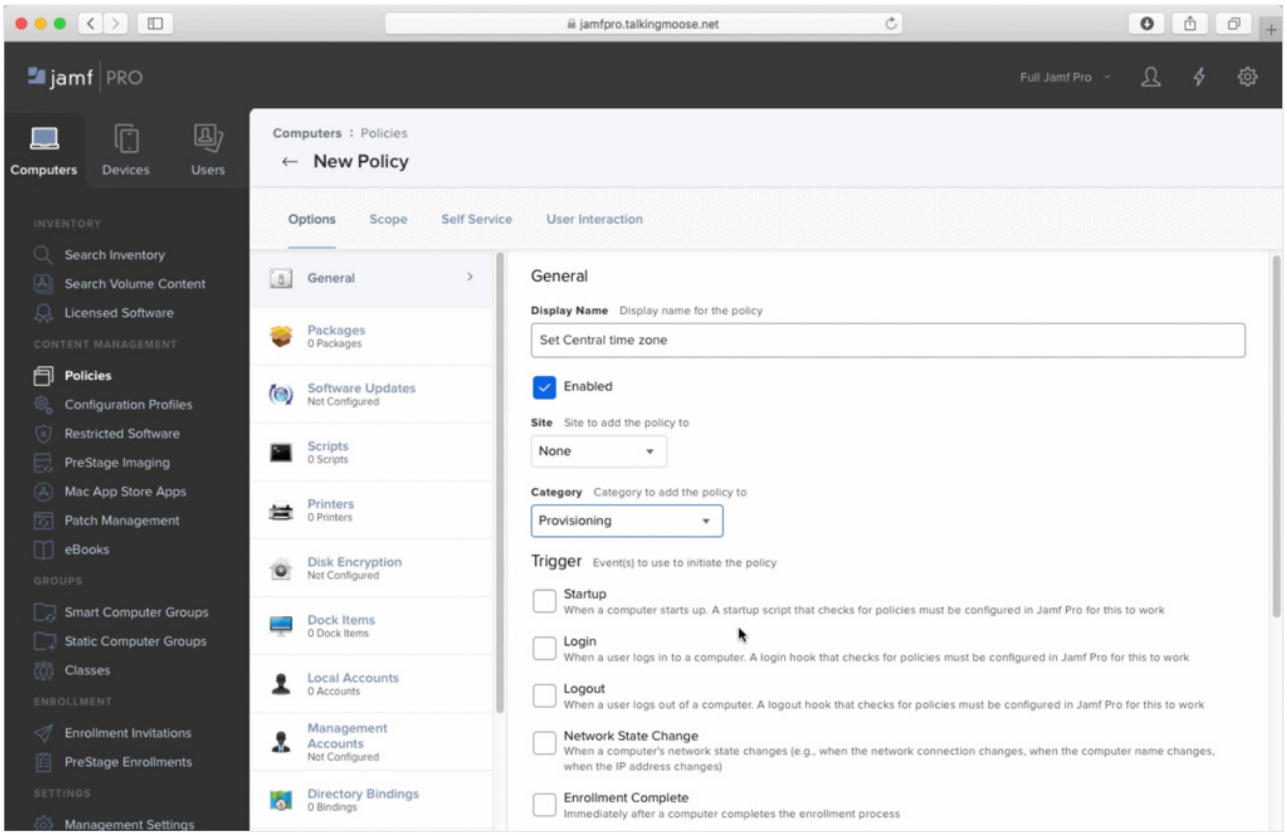
```
/usr/sbin/systemsetup -listtimezones
```

Vous obtenez une longue liste de fuseaux horaires, présentés dans un format correct pour le script. Dans notre exemple, nous choisissons Chicago, qui correspond au fuseau horaire central des États-Unis, et nous le copions.

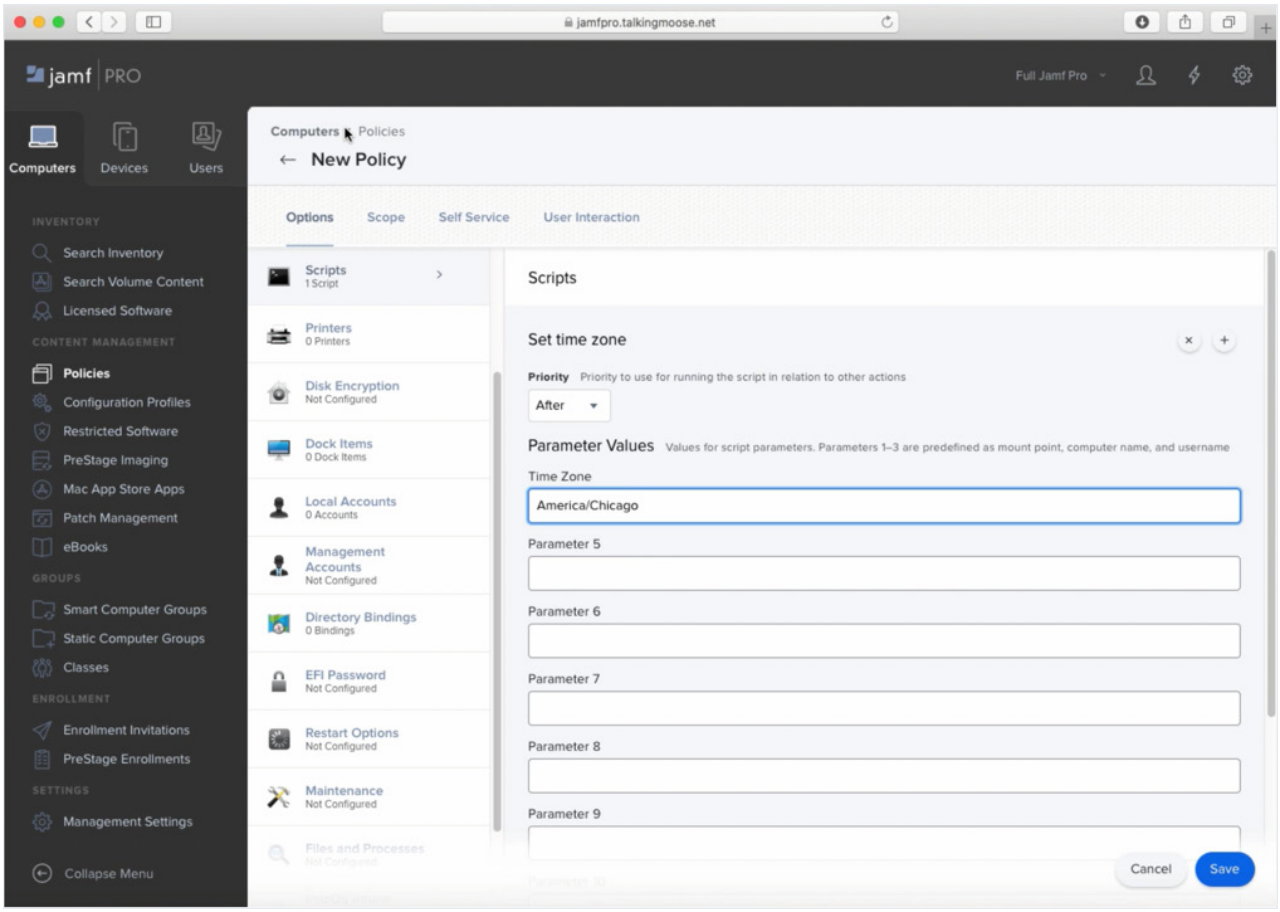
Créer une règle

Maintenant, créons une politique pour exécuter notre script.

- Dans votre instance Jamf, sélectionnez « Règles » dans le menu de gauche, puis la catégorie. Dans cet exemple, nous allons l'ajouter à la catégorie « Approvisionnement ».
- Nommez la règle « Attribuer le fuseau horaire Central », et ajoutez-la à la catégorie « Approvisionnement ».



- Définissez un déclencheur personnalisé : il vous permettra d'appeler cette règle par son nom dans un script ultérieur. Par exemple : attribuerfuseaucentral.
- Sélectionnez l'option Scripts et collez le fuseau horaire que vous avez copié depuis la commande du terminal.



Remarquez que ce champ présente le libellé que nous avons ajouté dans l'onglet Options en collant le script.
Le libellé vous indique ce que vous devez mettre ici.

Il ne vous reste plus qu'à définir la portée de la règle et à l'enregistrer.

Vous disposez maintenant d'une nouvelle règle dans la catégorie « Approvisionnement ».

Les paramètres de script offrent d'innombrables possibilités.

Le script a été écrit pour accepter des paramètres : vous pouvez donc le réutiliser à loisir pour différents fuseaux horaires. Il vous suffit de créer une nouvelle politique pour chaque fuseau horaire, d'ajouter le même script, puis de renseigner la valeur Fuseau horaire correspondante.

Créer des boîtes de dialogue avec jamfHelper et osascript

Les boîtes de dialogue sont très pratiques : elle permettent aussi bien d'afficher un message pour vos utilisateurs finaux que de leur demander des informations.

Voici deux façons de réaliser des boîtes de dialogue, chacune ayant ses propres avantages.

jamfHelper

jamfHelper est un outil en ligne de commande.

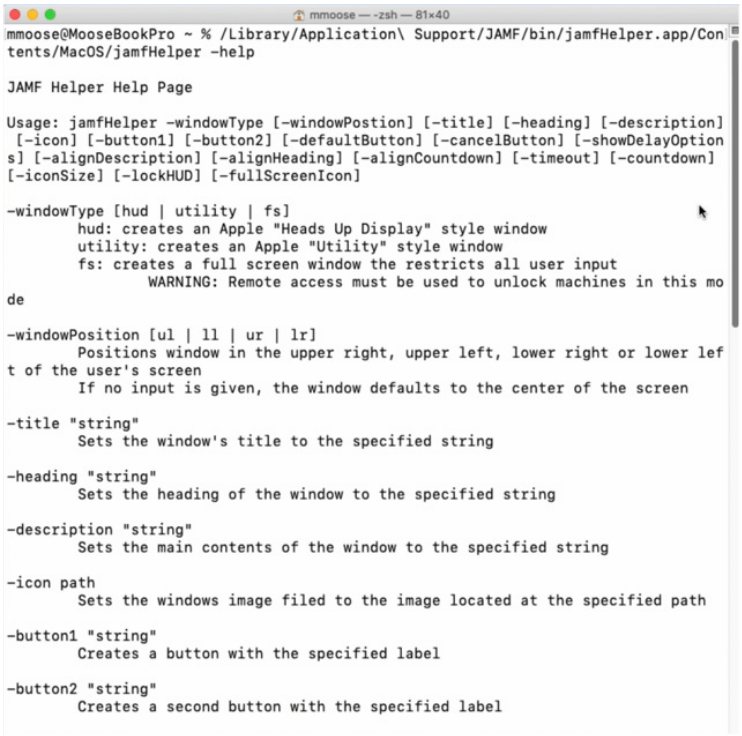
Ouvrez le terminal. Ensuite :

- Rendez-vous dans Bibliothèque > Application Support > JAMF > bin > jamfHelper
- Faites un clic droit sur jamfHelper et choisissez « Afficher le contenu du paquet ».
- Naviguez jusqu'à Contents > MacOS où vous trouverez l'outil de ligne de commande.
- Faites-le glisser dans le Terminal.

À la fin de la ligne de commande qui s'affiche, ajoutez « -help » et appuyez sur la touche Entrée :

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

Vous aurez alors toutes les instructions nécessaires pour utiliser jamfHelper.



Le chemin d'accès à jamfHelper est très long. Pour plus de simplicité, placez-le dans un nom de variable plus court (« jamfHelper ») à l'aide de ce script :

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

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

À partir de maintenant, pour appeler jamfHelper, il suffit de faire précéder le nom de la variable du signe dollar et de le mettre entre guillemets :

```
"$jamfHelper"
```

> Voici une astuce pour obtenir le bon chemin. Cliquez avec le bouton droit de la souris sur jamfHelper, placez votre curseur sur « Copier », puis enfoncez également sur la touche Option avant de cliquer pour copier le chemin plutôt que le fichier. Dans ce script, il vous suffit simplement de copier le chemin entre les guillemets.

Que peut faire jamfHelper ?

Imaginons que vous voulez ajouter des options à jamfHelper.
La première consiste à définir un type de fenêtre – il en propose trois.
Pour commencer, nous allons choisir « affichage tête haute ».
Ajoutez -windowType à votre script, suivi du type de fenêtre : « hud ».

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

> L'antislash en fin de ligne vous permet de répartir sur plusieurs lignes ce qui serait normalement une très longue commande.

Ensuite, ajoutez un titre. Vous devez le mettre entre guillemets :

```
-heading "Preparing your computer…" \
```

Ensuite, ajoutez une description.

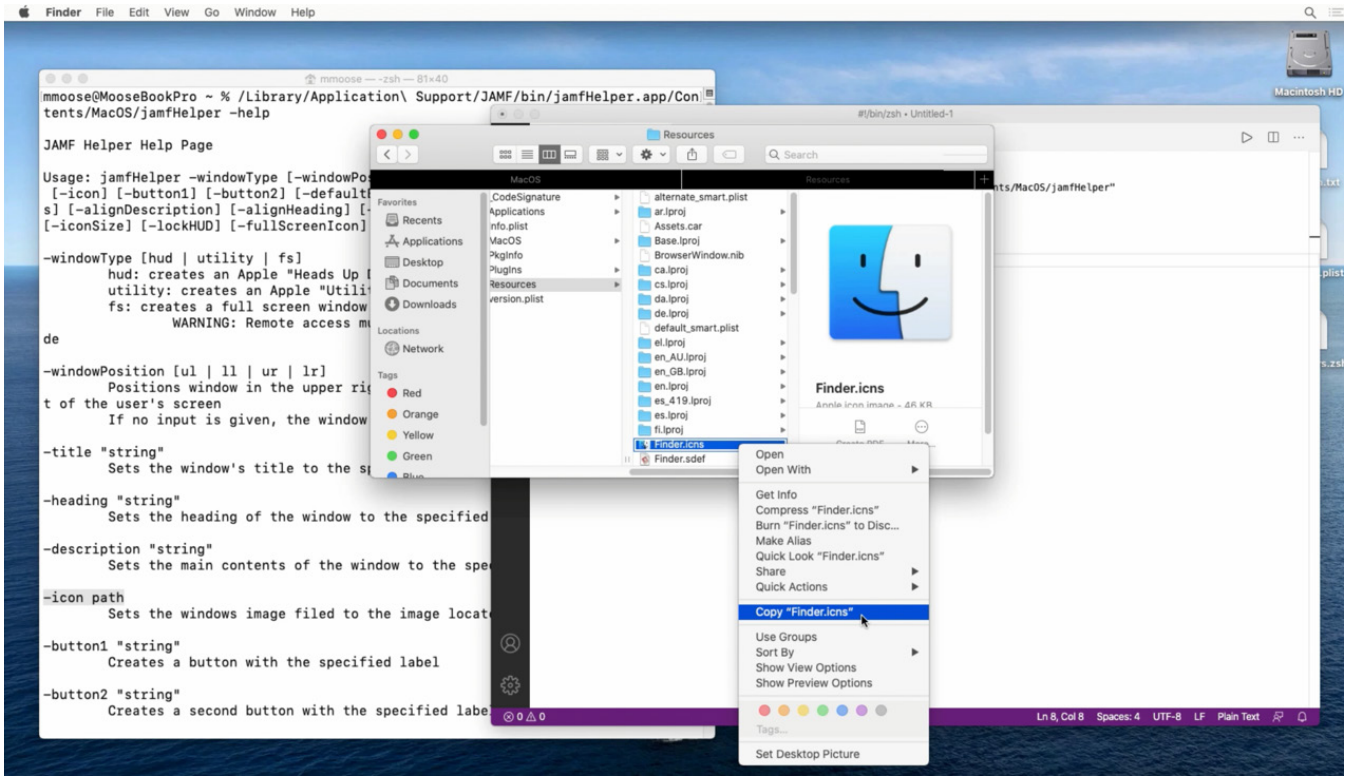
```
-description "Installation de Microsoft Office 2019"\
```

Si vous voulez ajouter de l'intérêt à la boîte de dialogue, ajoutez une icône. Choisissez une icône et insérez son chemin ici. Mettez le chemin entre guillemets au cas où il contiendrait des espaces.

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

Comment trouver les icônes préseentes sur votre Mac

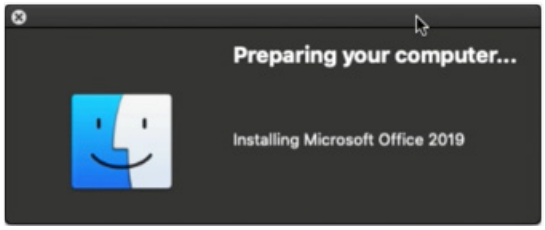
- Dans Finder, accédez à Système > Bibliothèque > CoreServices et localisez l'application Finder.
- Faites un clic droit sur Finder, choisissez « Afficher le contenu du paquet » puis Contents > Resources
- Sélectionnez l'icône Finder.
- Faites un clic droit sur l'icône Finder, maintenez la touche Option enfoncée et copiez le chemin.
- Collez ensuite le chemin dans votre script.



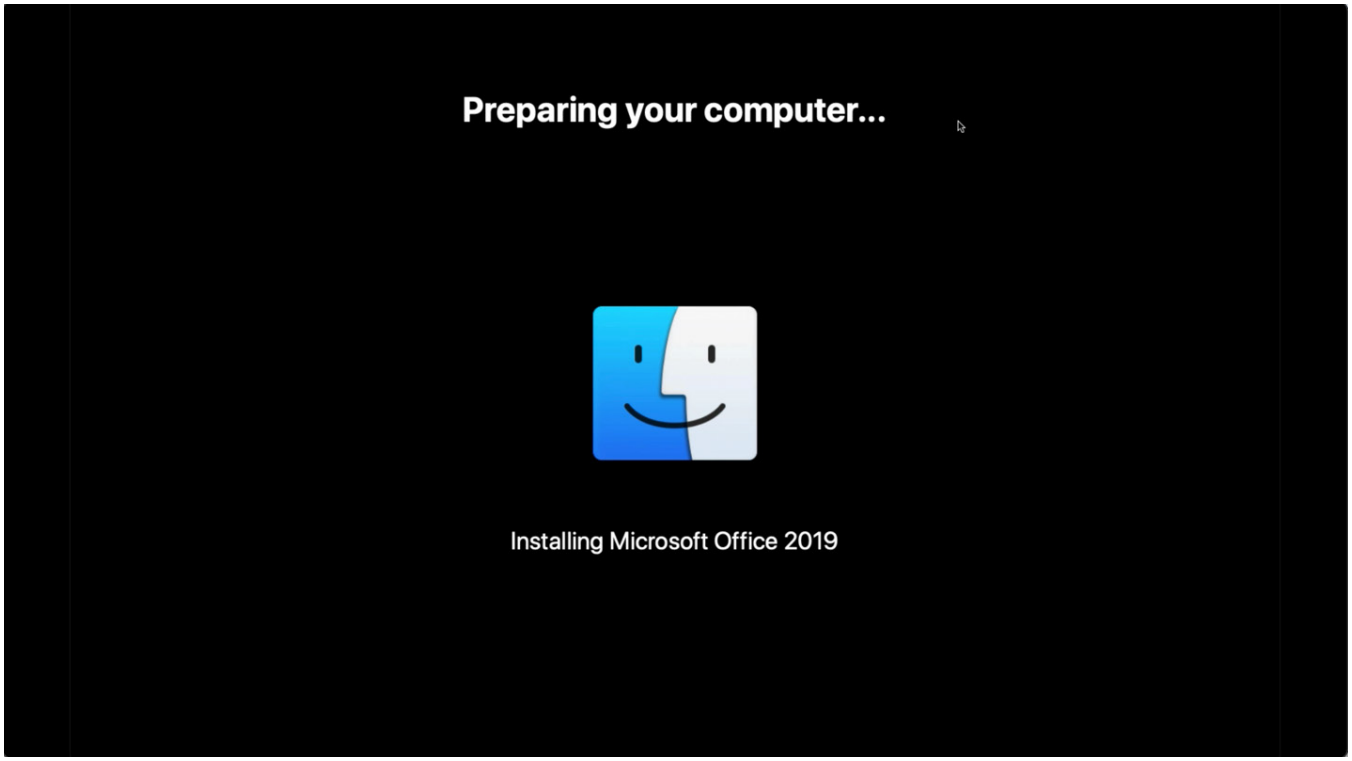
Voici comment tout cela se présente :

```
#!/bin/zsh

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

"$jamfHelper" -windowType hud  \
-heading "Préparation de votre ordinateur…" \
-description "Installation de Microsoft Office 2019"\
-icon "/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns"
```

Exécutez votre script directement depuis votre éditeur de script.
Vous obtenez un dialogue très simple :



Pour une boîte de dialogue en plein écran, choisissez « fs » (« full screen »)
comme type de fenêtre plutôt que « hud » et exécutez-la à nouveau :



La boîte de dialogue en plein écran offre un avantage de poids : elle empêche la personne qui se trouve devant l'ordinateur de l'utiliser pendant l'approvisionnement. Elle peut également se mettre à jour pour informer l'utilisateur de l'avancement de tâches telles que l'installation de logiciels.

Si, à tout moment, vous avez besoin de voir ce qui se passe derrière la boîte de dialogue, il suffit d'appuyer sur Commande + q pour quitter.

La commande osascript

Vous pouvez également créer des boîtes de dialogue via AppleScript, en utilisant la commande osascript du Terminal.

Elle offre davantage de possibilités que jamfHelper.

Tout d'abord, rédigez une commande pour créer une boîte de dialogue « choisir » proposant trois noms de service.

Vous devez utiliser des guillemets doubles pour commencer et terminer cette commande. Si des guillemets doubles figurent également dans la commande, faites-les précéder d'antislashs pour qu'ils soient interprétés comme des guillemets littéraux, et non comme des délimiteurs de commande :

```
asCommand="choose from list {\"Comptabilité\", \"Ventes\", \"Kiosque\"}
```

Ensuite, ajoutez le message à présenter à votre utilisateur final pour explorer ce que le Mac est sur le point de faire :

```
with prompt \"Bonjour, nous allons préparer votre Mac.
Pour commencer, choisissez votre service ci-dessous...\"
```

La dernière partie de la commande ajoute un titre à la fenêtre de dialogue. N'oubliez pas de mettre le guillemet fermant à la fin de la commande :

```
with title \"Préparez votre Mac\""
```

Cette ligne indique à osascript d'exécuter la commande et de placer le résultat dans la variable « service » :

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

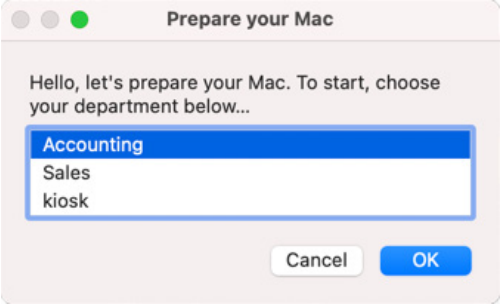
Enfin, utilisez echo avec la variable « service » pour afficher la sélection de l'utilisateur :

```
echo "$service"
```

Voici à quoi ressemble le script complet :

```
#!/bin/zsh

asCommand="choose from list {\"Comptabilité\", \"Ventes\", \"Kiosque\"} with prompt \
"Bonjour, nous allons préparer votre Mac. Pour commencer, choisissez votre service
ci-dessous...\" with title \"Préparez votre Mac\""

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

echo "$service"
```

Maintenant, testez le script dans votre éditeur de script. Ici, l'utilisateur a choisi « Comptabilité ».
Voici ce qu'il voit :



AppleScript et osascript peuvent faire bien plus avec les boîtes de dialogue, mais voilà déjà un bon point de départ.

Récapitulons

Vous pouvez ajouter ce script à Self Service pour permettre à un nouvel employé de provisionner son Mac en choisissant son service.

Il recueille d'abord le nom complet et le nom d'utilisateur de l'utilisateur connecté. Il utilise ensuite le nom complet dans l'invite osascript pour le choix du service:

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

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

nomComplet=$( /usr/bin/id -F "$utilisateurActuel" ) # p. ex. "mmoose"

echo "Approvisionnement de l'utilisateur $nomComplet ($utilisateurActuel)"
```

Ensuite, il demande à l'utilisateur de choisir un service :

```
asCommand="choose from list {\"Comptabilité\", \"Ventes\", \"Kiosque\"} with prompt \"Bonjour, $nomComplet ! Préparons votre Mac. Pour commencer, choisissez votre service ci-dessous...\"  with title \"Préparez votre Mac\""

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

echo "Approvisionné pour le service $department"
```

Un autre osascript demande le libellé d'actif de l'ordinateur :

```
asCommand="text returned of (display dialog \"Saisissez le libellé d'actif de ce Mac
(voir sous l'ordinateur)...\" default answer \"\" with title \"Préparez votre Mac\")"

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

echo "Le libellé de l'appareil est $libelléActif"
```

Le script va ensuite définir le fuseau horaire pour tous les ordinateurs :

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

Ensuite, il évalue le service choisi et installe les logiciels nécessaires :

```
if [[ "$service" = "Comptabilité" ]] ; then
          echo "Approvisionnement du Mac pour la Comptabilité"

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

elif [[ "$service" = "Ventes" ]]; then

          echo "Approvisionnement de ce Mac pour les Ventes"

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

else

          echo "Approvisionnement de ce Mac en tant que kiosque"

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

fi
```

Une fois le logiciel installé, il effectuera une mise à jour de l'inventaire qui comprendra le libellé d'actif, le service et le nom d'utilisateur de l'ordinateur :

```
"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Mise à jour de l'inventaire" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/local/bin/jamf recon -assetTag "$libelléActif" -department "$service"
-endUsername "$utilisateurActuel"
```

Il crée ensuite un dossier dans la bibliothèque pour les outils et les informations administratives :

```
"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Création du dossier d'administration" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

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

Et laisse un reçu d'aprovisionnement sur la machine :

```
"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Écriture du reçu d'approvisionnement" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

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

Enfin, il informe l'utilisateur et redémarre l'ordinateur :

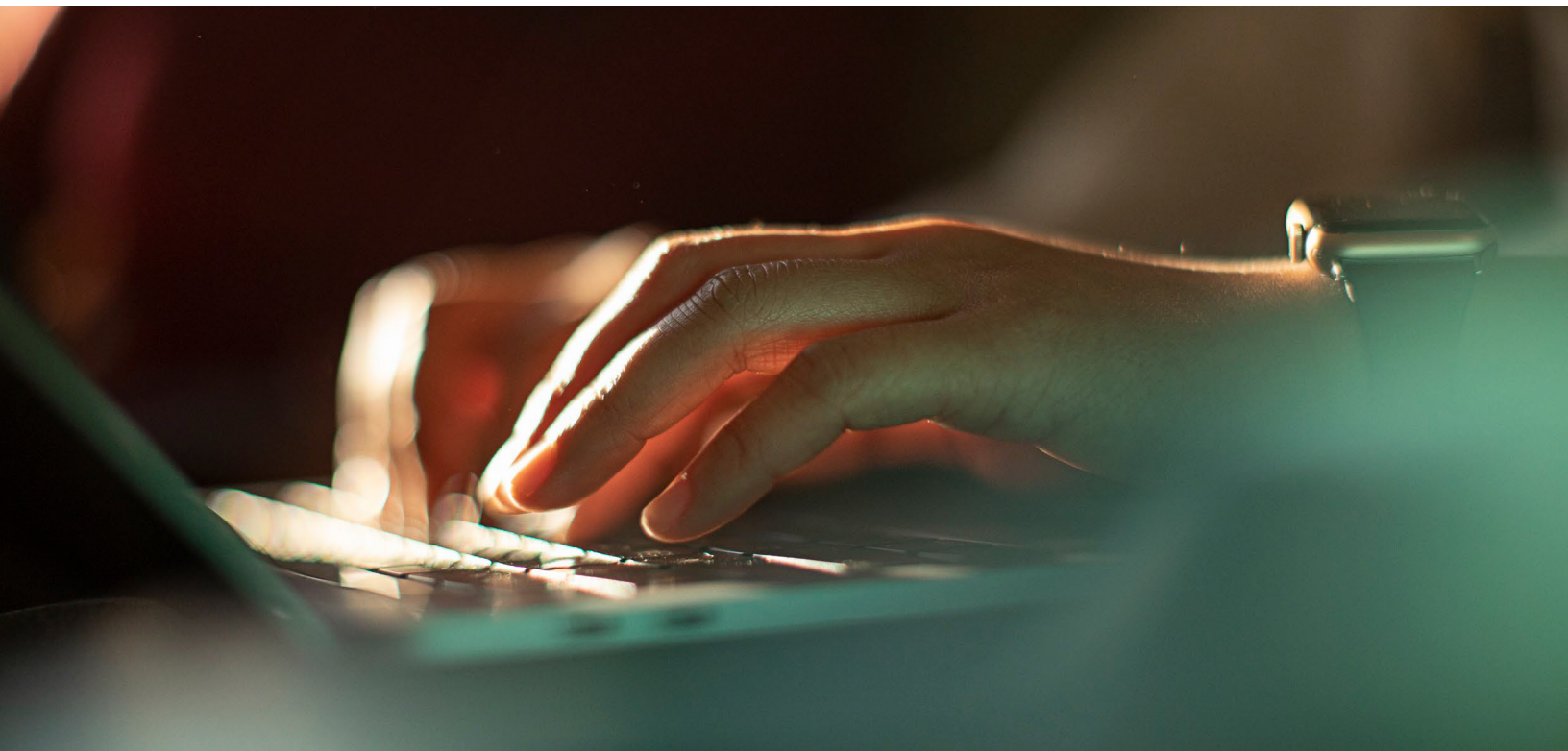
```
"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Redémarrage de votre Mac dans une minute" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &
/sbin/shutdown -r +1 &

exit 0
```

Script complet, avec descripteurs incorporés :

```
#!/bin/zsh

# assigner le long chemin de JamfHelper à une variable plus courte

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

# obtenir des informations sur l'utilisateur actuel

utilisateurActuel=$( /usr/bin/stat -f "%Su" /dev/console )
nomComplet=$( /usr/bin/id -F "$utilisateurActuel" ) # p. ex. "Martin Moose"

echo "Approvisionnement de l'utilisateur $nomComplet ($utilisateurActuel)"

# Commande AppleScript pour demander à l'utilisateur actuellement connecté de choisir
son service asCommand="choose from list {\"Comptabilité\", \"Ventes\", \"Kiosque\"}
with prompt \"Bonjour, $nomComplet ! Préparons votre Mac. Pour commencer, choisissez
votre service ci-dessous...\" with title \"Préparez votre Mac\""

# exécuter la commande

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

echo "Approvisionné pour le service $department"

# Commande AppleScript pour demander à l'utilisateur actuellement connecté d'entrer le
libellé d'actif

asCommand="text returned of (display dialog \"Saisissez le libellé d'actif de ce Mac
(voir sous l'ordinateur)...\" default answer \"\" with title \"Préparez votre Mac\")"

# exécuter la commande

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

echo "Le libellé de l'appareil est $libelléActif"
```

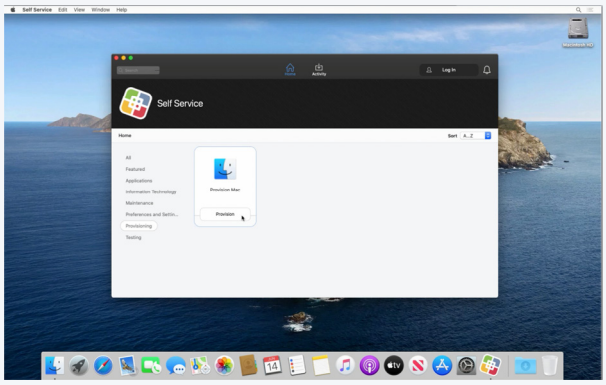
```
# configurer ce Mac pour le département sélectionné

# paramètres globaux

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

if [[ "$service" = "Comptabilité" ]]; then

          echo "Approvisionnement de ce Mac pour la Comptabilité"

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

elif [[ "$service" = "Ventes" ]]; then

          echo "Approvisionnement de ce Mac pour les Ventes"

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

else

          echo "Approvisionnement de ce Mac en tant que kiosque"

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

fi

# mettre à jour l'inventaire de Jamf Pro

"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Mise à jour de l'inventaire \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/local/bin/jamf recon -assetTag "$libelléActif" -department "$service"
-endUsername "$utilisateurActuel"

# créer le dossier admin

"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Création du dossier d'administration" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

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

```
# créer un reçu d'approvisionnement

"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Écriture du reçu d'approvisionnement" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisiondate -date $( /bin/date "+%Y-%m-%d" )
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
provisioner -string "$utilisateurActuel"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
department -string "$service"
/usr/bin/defaults write "/Library/Talking Moose Industries/receipt.plist"
assettag -string "$libelléActif"

"$jamfHelper" -windowType "fs" \
-heading "Préparation de votre Mac..." \
-description "Redémarrage de votre Mac dans une minute" \
-icon
"/System/Library/CoreServices/Finder.app/Contents/Resources/Finder.icns" &

# redémarrer le Mac
/sbin/shutdown -r +1 &

exit 0
```

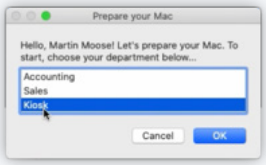


Voyons voir à quoi tout cela ressemble !

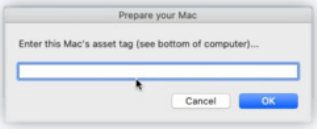
Ouvrez le Self Service et lancez le processus en cliquant sur le bouton Approvisionner.



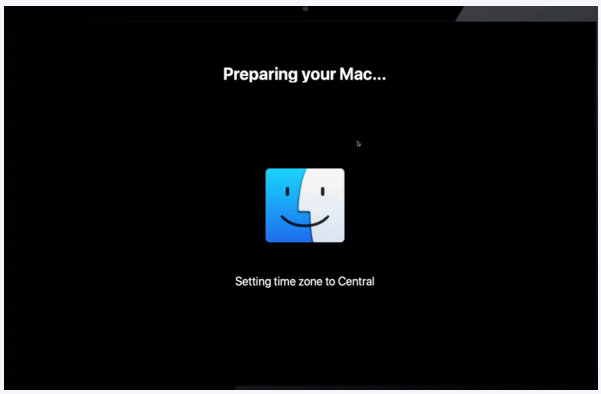
ÉTAPE 1.



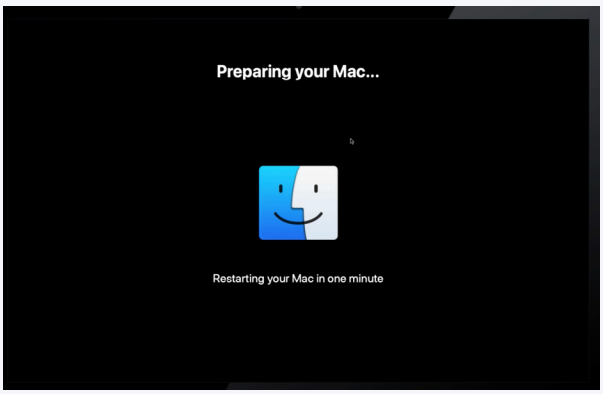
ÉTAPE 2.



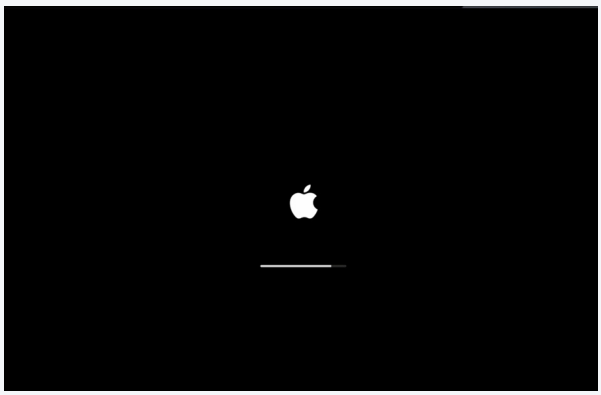
ÉTAPE 3.



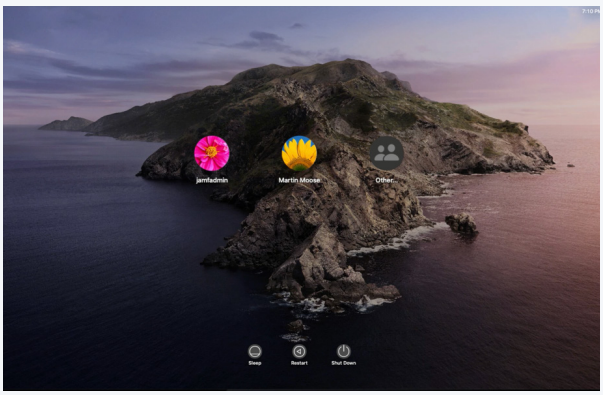
ÉTAPE 4.



ÉTAPE 5.



ÉTAPE 6.

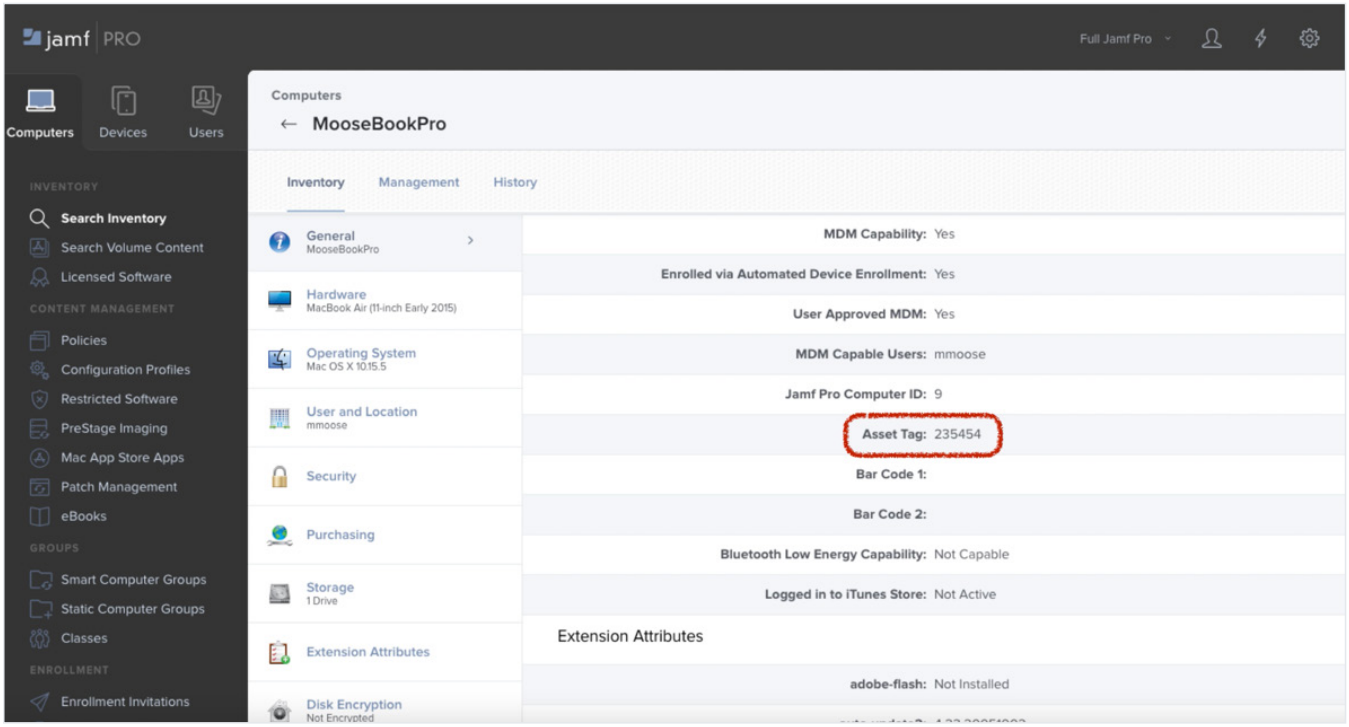


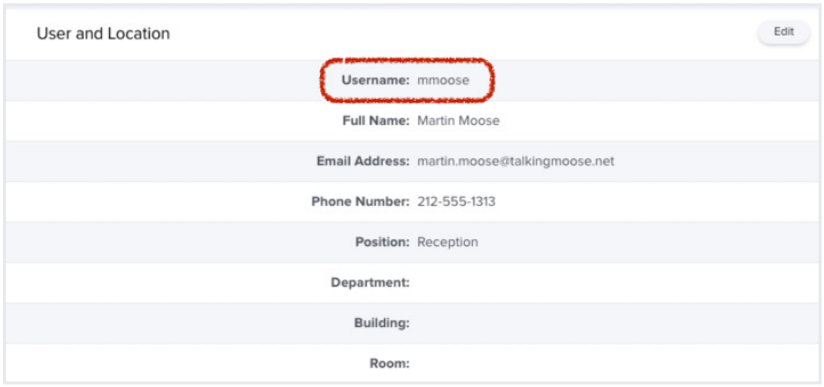
ÉTAPE 7.

Quand l'approvisionnement est terminé, l'ordinateur redémarre et le Mac revient à la fenêtre de connexion, prêt à fonctionner.

Une fois l'utilisateur connecté, les logiciels sont installés et le reçu d'approvisionnement est disponible dans le dossier Bibliothèque.

Lorsque l'administrateur Mac consulte l'enregistrement de l'ordinateur dans Jamf Pro, le libellé d'actif est rempli à partir de la boîte de dialogue osascript, tout comme le champ du nom d'utilisateur. Il peut alors effectuer une recherche LDAP et obtenir des informations supplémentaires : adresse électronique, numéro de téléphone, nom principal de l'utilisateur et autres informations provenant d'Active Directory.





Et c'est tout !

Autres ressources et supports d'apprentissage

Catalogue de formation de Jamf

N'oubliez pas que tous les clients de Jamf ont accès à l'ensemble de notre catalogue de formations. Il contient plus de 15 heures de petites vidéos pratiques pour tous les rôles, du service d'assistance à l'ingénieur en passant par l'administrateur : **trainingcatalog.jamf.com**

Jetez un coup d'œil à la série sur les scripts, qui comprend 15 modules et leçons vidéo dont la réalisation prend moins de 30 minutes. Vous pourrez ainsi approfondir votre apprentissage des scripts.

Github

Explorez la communauté open-source sur **GitHub.com**. Les logiciels libres sont gratuits, et vous trouverez des centaines de scripts et de projets sur la page GitHub de Jamf : **github.com/jamf**

Ressources de l'auteur Bill Smith

Si vous voulez découvrir des exemples de scripts courts ou plus étoffés répondant à un large éventail de besoins, consultez le **dépôt gist de Bill Smith sur GitHub**. Vous y retrouverez l'exemple de script d'approvisionnement de ce livre blanc et une partie du code qu'il a utilisé : **https://gist.github.com/talkingmoose**

Nous espérons que ce guide vous a aidé à passer au niveau supérieur en matière de script.

Combinés à un bon système de gestion d'entreprise, les scripts peuvent vraiment faire la différence entre des heures de travail et un simple clic.

Demander une version d'essai

Nous aimerions vous suggérer Jamf Pro.