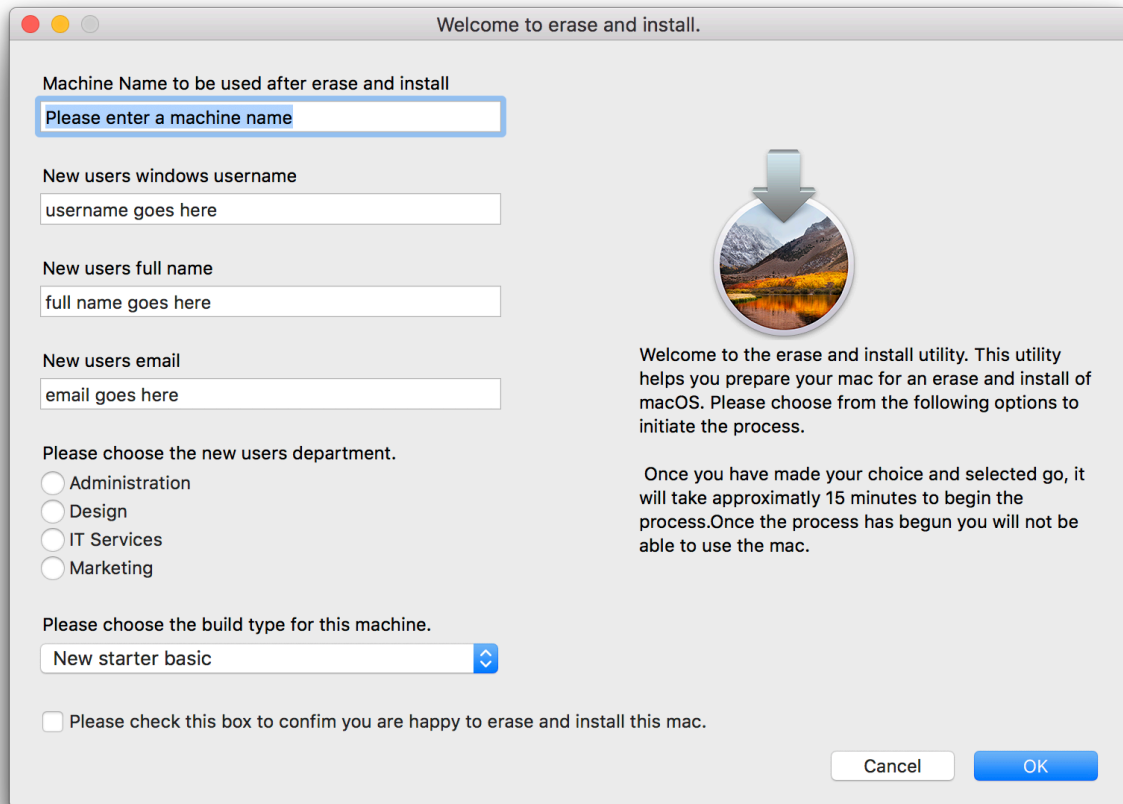


# Erase Install Utility



Welcome to erase and install.

Machine Name to be used after erase and install

Please enter a machine name

New users windows username

username goes here

New users full name

full name goes here

New users email

email goes here

Please choose the new users department.

☐ Administration

☐ Design

☐ IT Services

☐ Marketing

Please choose the build type for this machine.

New starter basic

☐ Please check this box to confirm you are happy to erase and install this mac.

Cancel OK

Welcome to the erase and install utility. This utility helps you prepare your mac for an erase and install of macOS. Please choose from the following options to initiate the process.

Once you have made your choice and selected go, it will take approximately 15 minutes to begin the process. Once the process has begun you will not be able to use the mac.

## Documentation v1

## Written by Daniel Mintz

Welcome to the documentation for the erase install utility. The purpose of this utility is to allow an administrator or maybe a team lead to re provision a machine for a new user and have it erase to factory fresh, re install applications according to a specified build and be ready in the Jamf Pro console showing the correct user assigned.

The utility is built around the Pashua Framework. You can download Pashua and get help and resources from this site

<https://www.bluem.net/en/projects/pashua/>

In addition the utility makes use of a script by Josh Roskos, Josh is a STAM at Jamf and during his professional service days he wrote a script which enabled in place OS upgrades and the erase install function. This script can be found on Josh's Github

<https://github.com/kc9wwh/macOSUpgrade>

This guide will go step by step how to set up the utility and explain the different components being used.

## A word about erase install.

eraseinstall was introduced in the 10.13.4 (High Sierra) installer. Its function is to do an in place command line driven erase and factory reset of the OS. Think of it like the erase all content and settings option on your iPhone or iPad.

eraseinstall only works with APFS drives.

The erase install function will only run if the target OS running it matches the version of the installer being used. i.e a Mac running 10.13.4 can not run the 10.13.5 installer and use the eraseinstall flag.

This script also uses the install package flag to add a custom quick add to the end of the eraseinstall and get the device back under management. If you are a DEP customer this process is not needed.

# Getting Pashua ready to deploy to the machine.

In order to use Pashua for a GUI, you must have the Pashua app on the machine in question and have the Pashua.sh script present. The Pashua.sh script holds the functions which make the gui run.

The first step to take is to package up the two components as follows



*The path used here is referenced in my script so be sure to package according to the screen shot.*

*Its totally optional if you want to change the Pashua icon. I did so its themed according to the OS version you are running.*

Now you should build this out as a package and upload it to your Jamf Pro.

# Make the extension attribute for build type in Jamf Pro

This extension attribute contains the values that you will see in the utility when you choose the build configuration that the machine will have post eraseinstall.

You can customise your extension attribute to have what ever build configurations you wish.

The screenshot shows the 'Build' configuration page in Jamf Pro. The breadcrumb trail at the top is 'Settings > Computer Management > Extension Attributes > Build'. The page title is 'Build'. The configuration fields are as follows:

- DISPLAY NAME:** Display name for the extension attribute. Value: Build.
- DESCRIPTION:** Description for the extension attribute. (Empty text area)
- DATA TYPE:** Type of data being collected. Value: String.
- INVENTORY DISPLAY:** Category in which to display the extension attribute in Jamf Pro. Value: General.
- INPUT TYPE:** Input type to use to populate the extension attribute. Value: Pop-up Menu.
- RECON DISPLAY:** Pane on which to display the extension attribute in Recon. Value: Extension Attributes.
- POP-UP MENU CHOICES:** A list of choices for the pop-up menu:
  - IT test machine
  - New starter advanced
  - New starter basic

At the bottom right, there are five buttons: Done, History, Clone, Delete, and Edit (highlighted in blue).

## Modify the script so the build types appear in the pop up menu.

Now you should modify the script so that the build types in your extension attribute appear in the utility when you choose the drop down menu for build.

NOTE: You must enter the values in the script exactly as they are in the extension attribute. Extension attribute manipulation is case sensitive.

The location in the script looks like this

```
107 # Add a popup menu
108 pop.type = popup
109 pop.label = Please choose the build type for this machine.
110 pop.width = 310
111 pop.option = New starter basic
112 pop.option = New starter advanced
113 pop.option = IT test machine
114 pop.default = New starter basic
115 pop.tooltip = This is an element of type "popup"
116
```

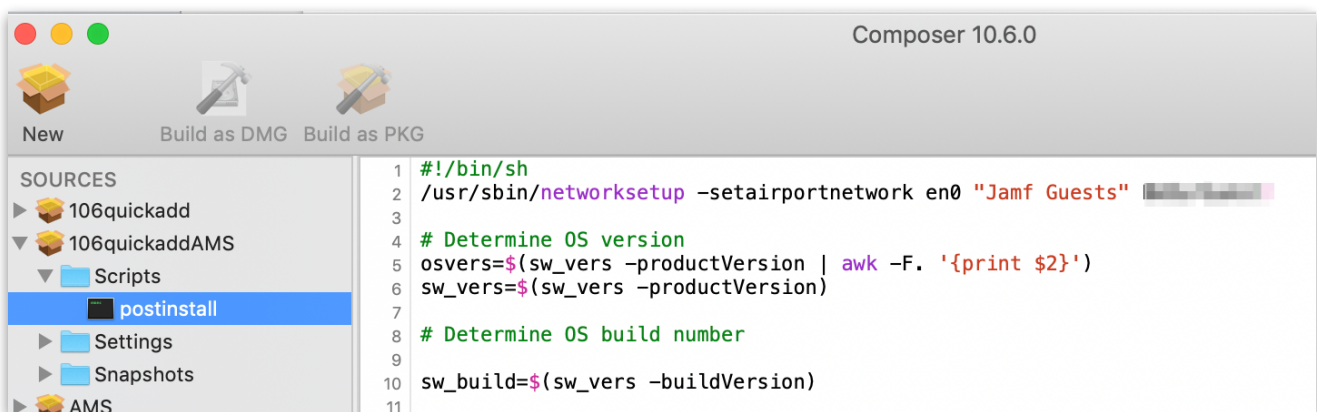
# Modify the quick add package to be used in the eraseinstall.

Now we will modify the quick add package that we use in the eraseinstall process.

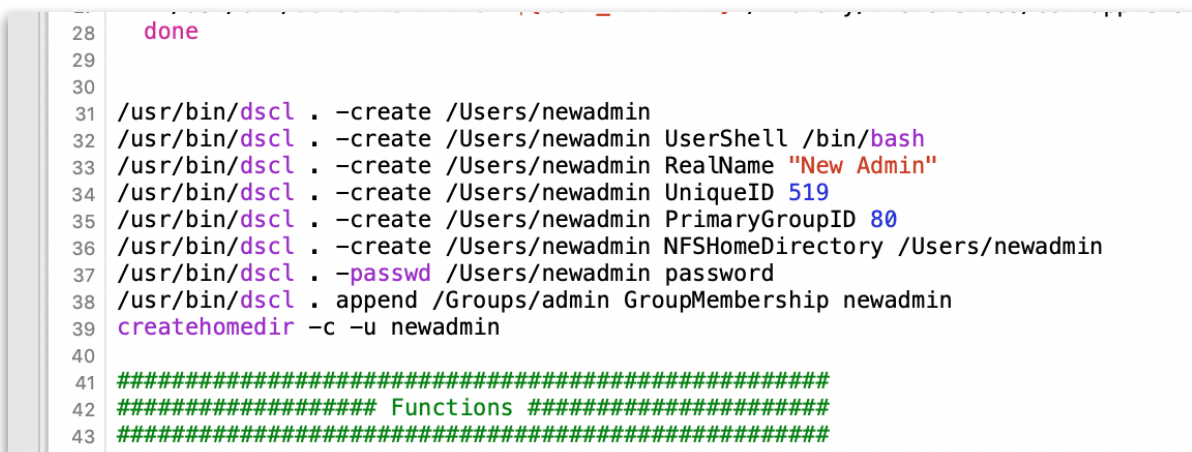
The modifications will allow us to have the device join a wireless network, create a new account for use in our build process and return the Mac to a setup assistant state after the build has finished.

First you should make a normal quick add package in Recon and convert it to source in Composer.

When you convert to source, go ahead and expand the package and edit the post install script. From here you should add the following lines.



Wifi connection added in line 2 of the script.



Commands added from lines 31-39 to create a new account for use in the build process.

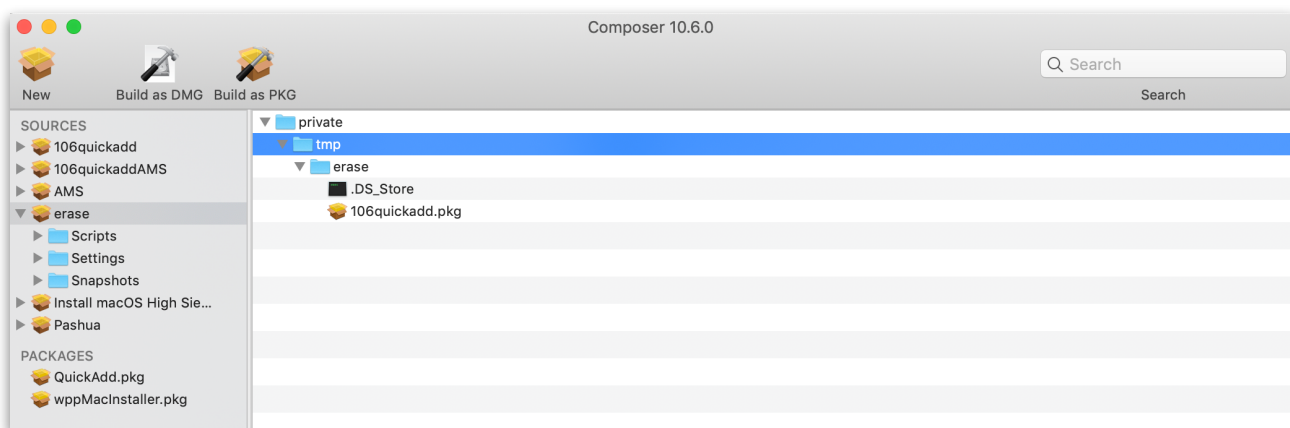
```

136 #####
137 ## Turn on SSH
138 #####
139 $jamfCLIPath startSSH
140
141 /bin/mkdir -p /private/var/db
142 /usr/bin/touch /private/var/db/.AppleSetupDone
143 /bin/chmod 0400 /private/var/db/.AppleSetupDone
144 /bin/mkdir -p /Library/Receipts
145 /usr/bin/touch /Library/Receipts/.SetupRegComplete
146

```

Commands added from lines 139-145 to return the machine to a setup assistant state after build.

Once you have modified the package please go ahead and compile this package and export it, you will then need to wrap this package in a DMG so that it deploys to a set location on the client and can be used for the installpackage part of the eraseinstall process.



# Upload the eraseinstall.sh script to Jamf

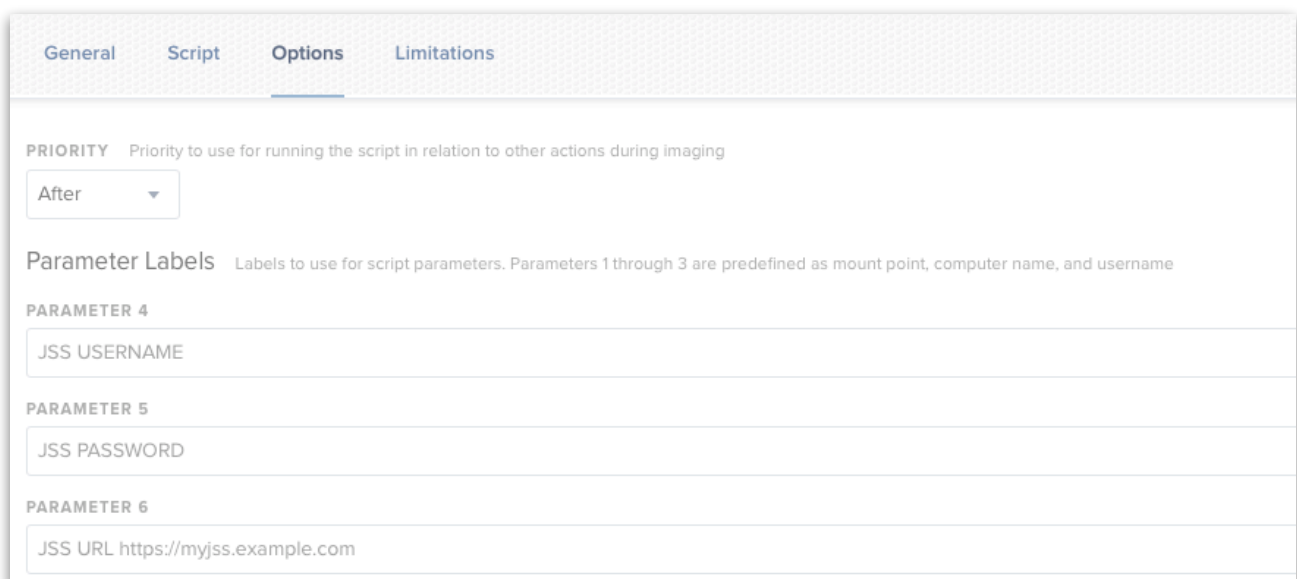
Now you should upload the script to your Jamf Pro. You should make note of the variables that are used. I use \$4,5&6 in this script

\$4 = JSS Username

\$5 = JSS Password

\$6 = JSS url.

You set the values for 4,5 & 6 when you build the policy to run this script.



The screenshot shows the 'Options' tab in the Jamf Pro interface. It features a 'PRIORITY' dropdown menu set to 'After'. Below this is a 'Parameter Labels' section with three input fields: 'PARAMETER 4' containing 'JSS USERNAME', 'PARAMETER 5' containing 'JSS PASSWORD', and 'PARAMETER 6' containing 'JSS URL https://myjss.example.com'.

General	Script	Options	Limitations
<b>PRIORITY</b> Priority to use for running the script in relation to other actions during imaging			
After ▼			
<b>Parameter Labels</b> Labels to use for script parameters. Parameters 1 through 3 are predefined as mount point, computer name, and username			
<b>PARAMETER 4</b>			
JSS USERNAME			
<b>PARAMETER 5</b>			
JSS PASSWORD			
<b>PARAMETER 6</b>			
JSS URL https://myjss.example.com			



# Upload and modify the script to perform the eraseinstall.

Go to the following GitHub and copy the script macOSUpgrade.sh

<https://github.com/kc9wwh/macOSUpgrade/blob/master/macOSUpgrade.sh>

Once you have copied this into the script editor in your Jamf Pro, you should make some changes to the script.

First we must set up the \$ variables that this script uses. To do this navigate to the options tab and provide the labels needed for the parameters. We will set the parameters later when we make the policy to run this script.

The screenshot shows the 'Options' tab of the 'MacOS High Sierra Upgrade Script' in Jamf Pro. The breadcrumb trail at the top is 'Settings > Computer Management > Scripts > MacOS High Sierra Upgrade Script'. Below the breadcrumb trail are four tabs: 'General', 'Script', 'Options' (which is selected and underlined), and 'Limitations'. The 'Options' tab contains several configuration fields:

- PRIORITY**: A dropdown menu with the value 'Before' selected. The description is 'Priority to use for running the script in relation to other actions during imaging'.
- Parameter Labels**: A text field with the value 'Install Location E.G /Applications/Install ...'. The description is 'Labels to use for script parameters. Parameters 1 through 3 are predefined as mount point, computer name, and username'.
- PARAMETER 4**: A text field with the value 'OS Installer Version that will be run.'
- PARAMETER 5**: A text field with the value 'Custom Trigger to trigger a download of the OS installer'.
- PARAMETER 6**: A text field with the value 'Set this to 1 to enable eraseinstall . If left at 0 it will be man upgrade NOT an eraseinstall.'

Now we must edit the script to include the `install package` option that will run the quick add to re enrol the machine back into Jamf.

To do this navigate to line 380 and add the following `after the nointeraction` option.

```
--nointeraction --installpackage /private/tmp/erase/106quickadd.pkg
```

Then repeat the action on line 382.

## Create the policy for Self Service to run the erase and install utility.

This policy appears in Self Service and runs the erase and install utility.

The Policy contains the script that runs the utility (`eraseinstall.sh`) as well as the Pashua Package that you created earlier. The scope for this policy should be set to a smart group which identifies Machines which meet the criteria for `eraseinstall`.

Remember, the criteria is to be running the same OS version as the installer you are using, so if your OS installer is 10.14.0 you will want a smart group that targets machines with that OS.

When setting up this policy you should also remember to add in the variables as needed for \$4,5&6

\$4=API Username

\$5=API Password

\$6=Jamf Pro URL

Computers > Policies > Erase and Install macOS

Options

Scope

Self Service

User Interaction

Scripts  
1 Script

Printers  
0 Printers

Disk Encryption  
Not Configured

Dock Items  
0 Dock Items

Local Accounts  
0 Accounts

Management Accounts  
Not Configured

Directory Bindings  
0 Bindings

EFI Password  
Not Configured

Restart Options

Scripts

Erase and Install.

PRIORITY

Priority to use for running the script in relation to other actions

After

Parameter Values

Values for script parameters. Parameters 1–3 are predefined as mount point, computer name, and username

JSS USERNAME (\$4)

jssadmin

JSS PASSWORD (\$5)

JSS URL https://myjss.example.com (\$6)

https://danielmintz.jamfcloud.com

Parameter 7

Cancel

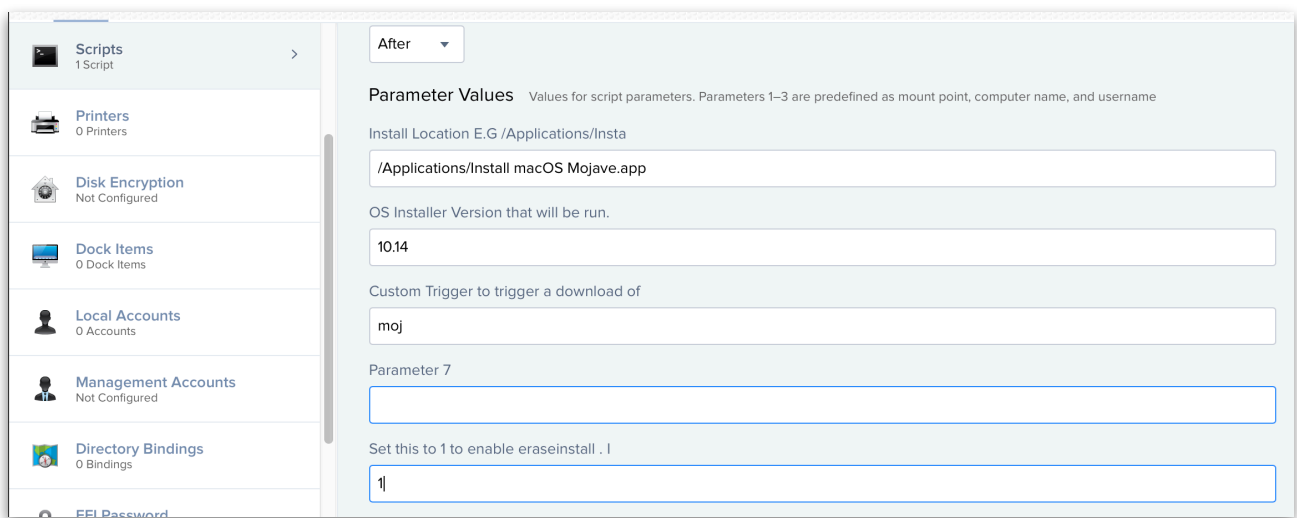
Save

# Create the policy to start the erase install.

This policy is called using a custom trigger which runs at the end of the utility script above. The custom trigger to use is 'starterase'.

This policy contains the script macOSUpgrade.sh that we got from Josh Roskos, it also contains the custom quick add package that we uploaded earlier in this guide.

You should set the parameters for the script as follows:



The screenshot shows the macOS Policy Management interface. On the left is a sidebar with navigation options: Scripts (1 Script), Printers (0 Printers), Disk Encryption (Not Configured), Dock Items (0 Dock Items), Local Accounts (0 Accounts), Management Accounts (Not Configured), Directory Bindings (0 Bindings), and FDI Password. The main area is titled 'After' and contains the following configuration fields:

- Parameter Values**: Values for script parameters. Parameters 1–3 are predefined as mount point, computer name, and username.
- Install Location E.G /Applications/Insta**: /Applications/Install macOS Mojave.app
- OS Installer Version that will be run.**: 10.14
- Custom Trigger to trigger a download of**: moj
- Parameter 7**: (Empty field)
- Set this to 1 to enable eraseinstall . I**: 1

Note that the script is set to run after. This means the custom quick add package will be deployed before the script runs.

The scope for this policy can be to all devices. This is safe to do as the policy is only run by a custom trigger which is called at the end of the utility script.

Once this is complete then you should be able to run the self service policy to launch the utility.

The following sequence of screen shots represents the user experience seen.

