Google Chrome (macOS)

- Google Chrome Configuration recipe (macOS)
- Google Chrome Managed (macOS)
- Google Chrome Extension Management (macos)
- Google Chrome Install Recipe (macOS)

Google Chrome Configuration recipe (macOS)

Description

After installing Chrome, some configuration may be desirable, examples may include:

- Block auto updates
- Set home page
- Disable Welcome Page

The below shows configuration for macOS. For Windows you may wish to consider GPO: Manage Chrome updates (Windows)

Ingredients

- FW Central
- Chrome installer application already as a Fileset
- The provided configuration Fileset:

macOS

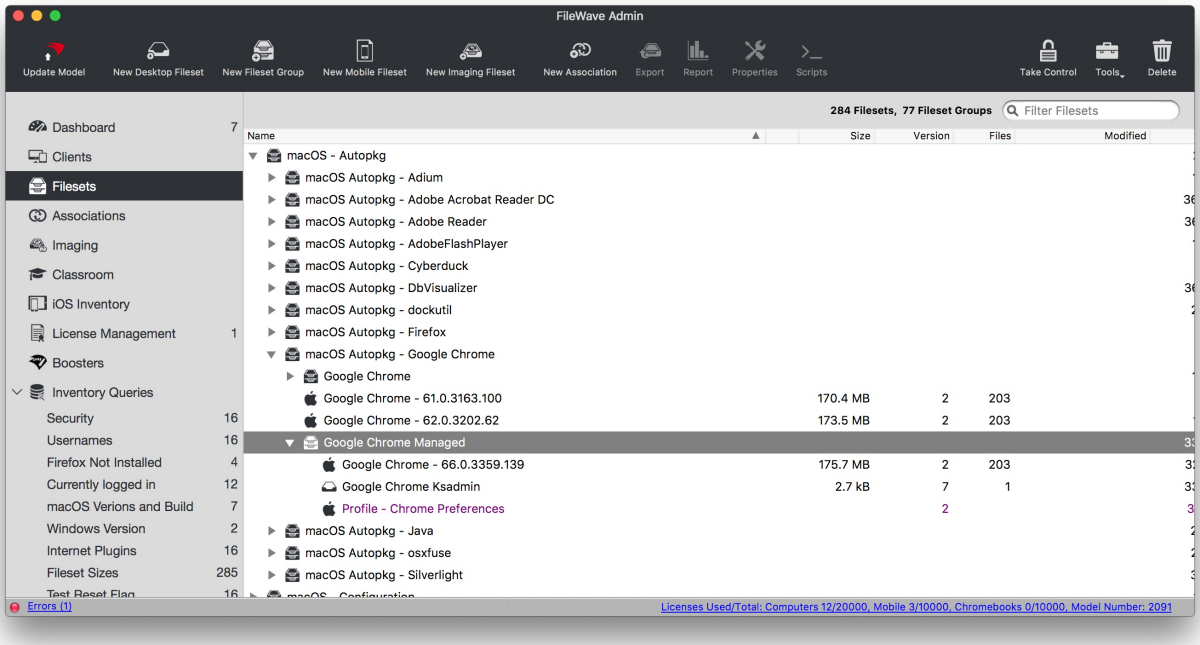
Directions

1. Download the Fileset zip file, expand and drag both the Fileset and Configuration Profile into FileWave > Filesets
2. The 'Google Chrome Ksadmin' contains 'ksadmin.sh' script which configures ksadmin to allow silently disabling Chrome updates
3. The Configuration Profile has example setups that may require editing, e.g Welcome and Home Page; replacing www.filewave.com

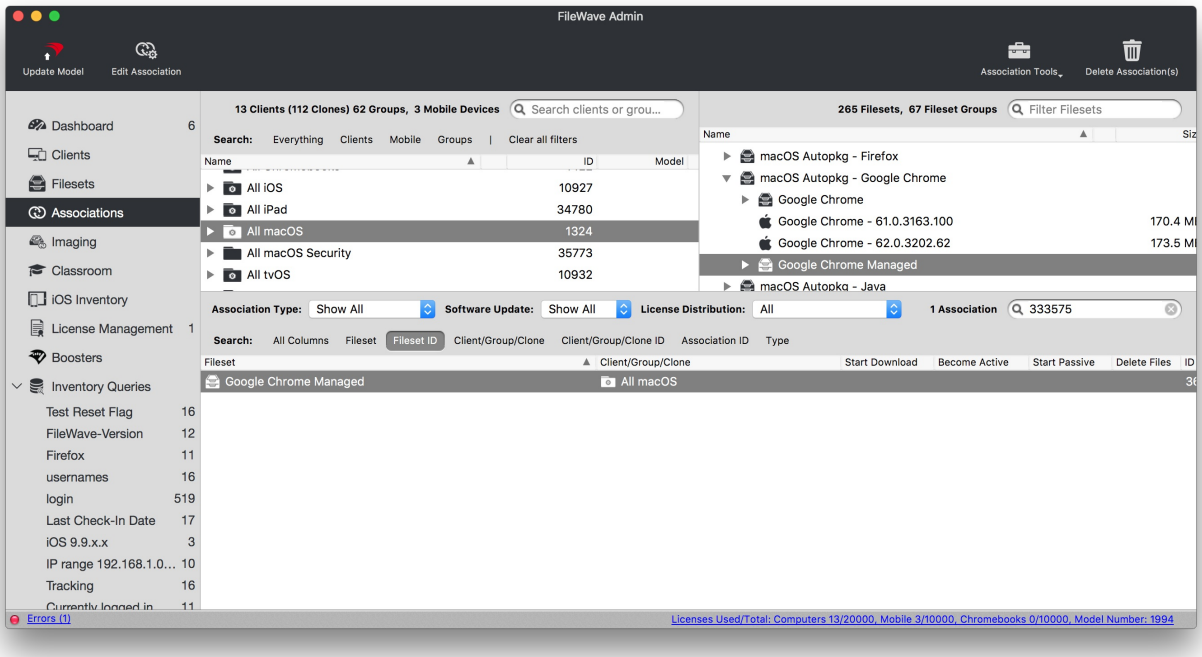
> Preference Names
> A list of policies can be found at: Chrome Enterprise policy list

Association

When associating, to ensure the configuration is installed with the Application, consider creating a Fileset Group:



Associating the 'Google Chrome Managed' Fileset Group will ensure Application and Configuration are associated to devices.



When there is an update to Google Chrome, replace the version in the Fileset Group with the latest version, after testing, and re-instal Google Chrome Ksadmin

Controlling Updates

Updates may be managed with the use of additional configuration that may be added to the Fileset Group.

The following profile contains:

```
<key>updatePolicies</key>
<dict>
  <key>com.google.Chrome</key>
  <dict>
    <key>UpdateDefault</key>
    <integer>3</integer>
  </dict>
</dict>
```

Download: Chrome Preferences Keystone.mobileconfig

Edit the profile integer to match the desired experience:

Setting	Description
0	Turns on auto-updates. Updates are always applied when detected by Google Software Update. This is the default value.
1	Updates are installed only from the scheduled update checks. Manual update checks will not install updates.
2	Turns off auto-updates. This stops Google Software Update automatically updating all users to the latest stable version of Chrome. Updates are only applied when the user manually checks for updates. For example, on the chrome://help page or by running the CheckForUpdatesNow.command utility.
3	Updates are never applied.

Add the 'mobileconfig' file as a Custom Settings Configuration Profile Payload

> Google Software Management
> Other Google software may also be managed with this process. Please read Manage Chrome updates (Mac) for full details of management options.

Google Chrome Managed (macOS)

Description

Chrome can be managed using methods outlined in the KB: Google Chrome Configuration Recipe (macOS)

However, from the Google Admin Console, it is possible to create a Management Token, push the token to devices and then control the browser experience within Google's Cloud management.

> The Chrome Browser Management Token can be used on macOS, Windows, Android and iOS.

Ingredients

- FW Central
- Chrome installer application already as a Fileset
- Google Admin Console access
- Chrome Browser Enrolment Token

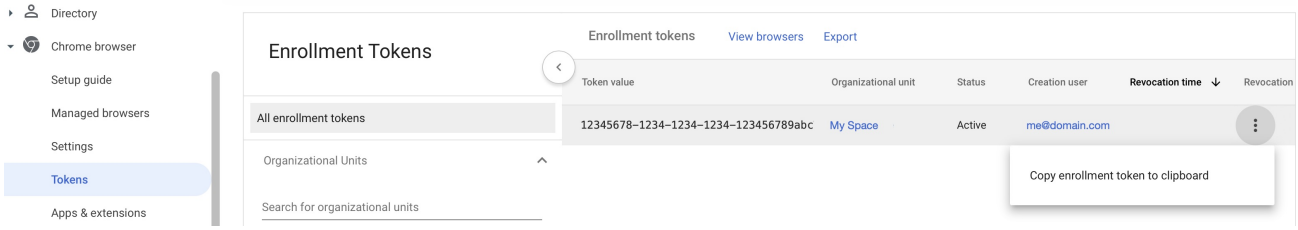
macOS

Directions

Google

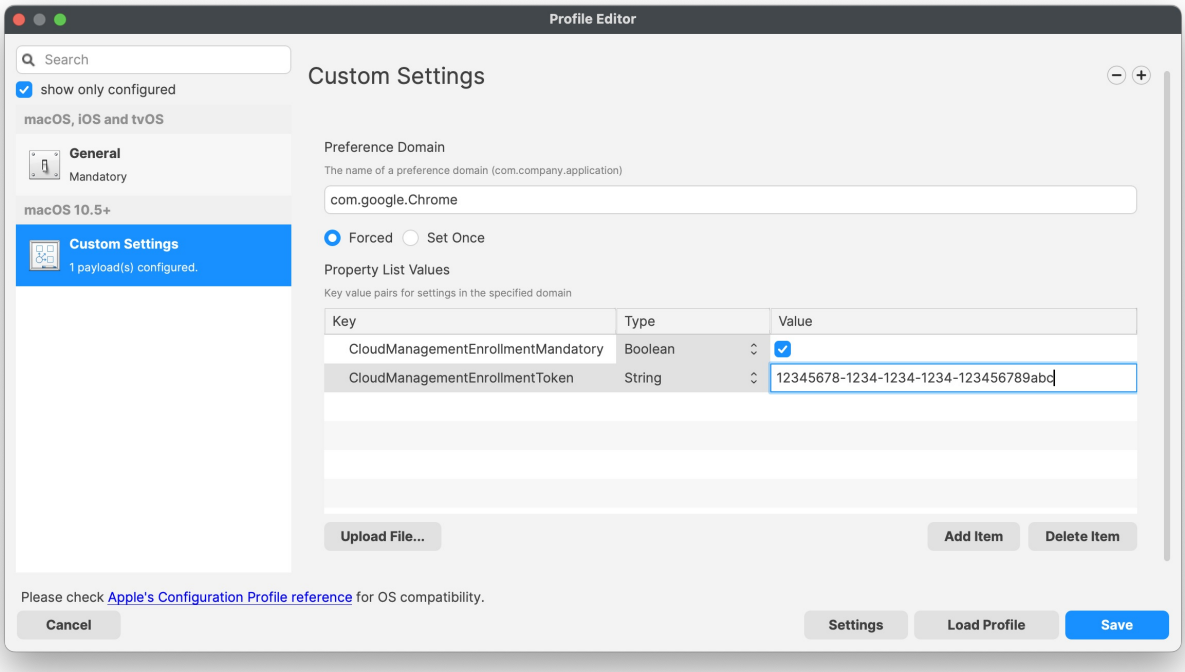
Google have their own KB on this topic: Enroll cloud-managed Chrome browsers

1. From the Chrome Browser view in the Google Admin console, use the 'kebab' (vertical ellipsis) to copy the token



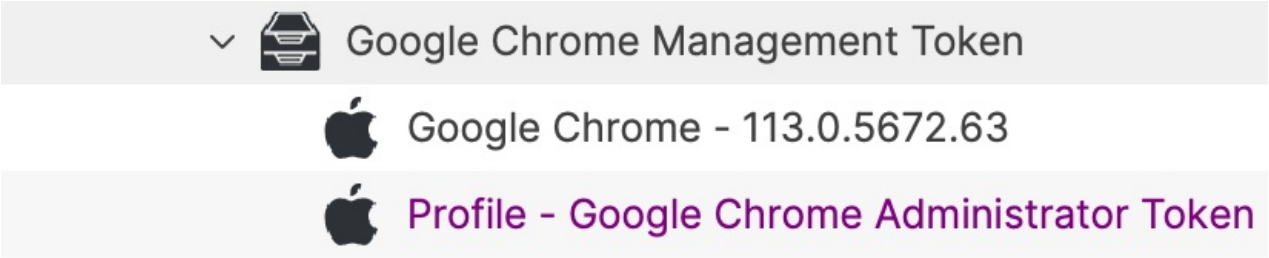
FileWave

1. Download the above provided Fileset zip file, expand and drag into FileWave > Filesets
2. Open the Profile for editing and access the Custom Settings Payload
3. Edit the CloudManagementEnrollmentToken String, entering the token from the Google Admin console

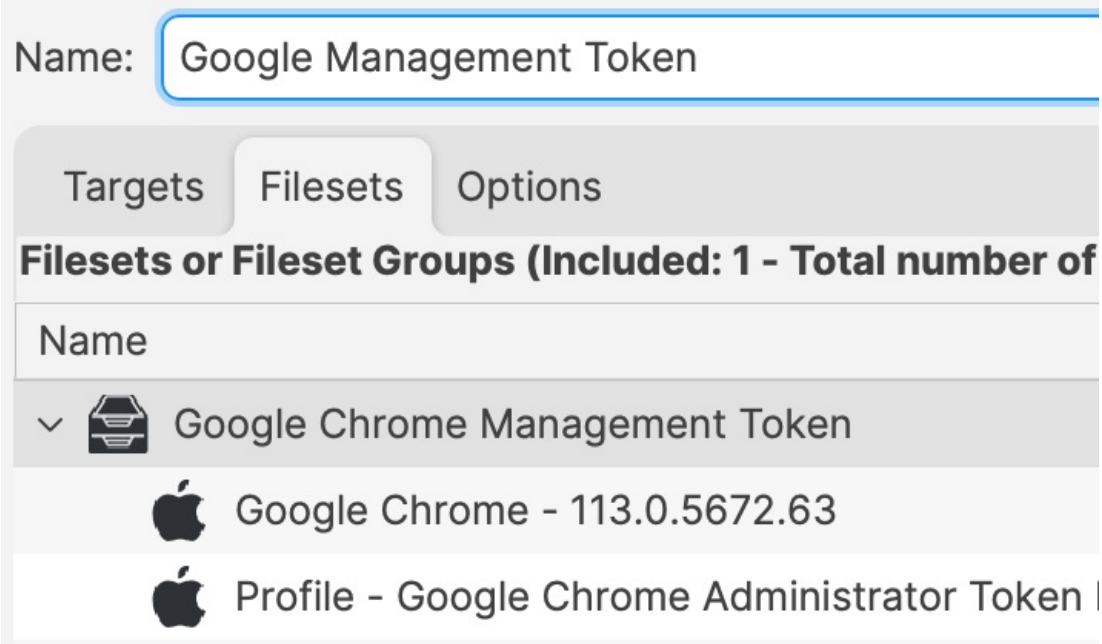


Association

When associating, to ensure the configuration is installed with the Application, consider creating a Fileset Group:



Assigning the 'Google Chrome Management Token' Fileset Group will ensure Application and Configuration are associated to devices. Below is an example using Deployments:



When there is an update to Google Chrome, replace the version in the Fileset Group with the latest version (after testing) or add a new Fileset Revision if desired.

Google Chrome Extension Management (macos)

Description

Chrome Extensions, like ClassSpaces, can be managed via FileWave on multiple device types.

If the Chrome browser is already managed using Google Admin Console, then all management may be applied to devices via the Google Admin Console. It is though possible to apply the management of Chrome to macOS and Windows devices via FileWave also.

Ingredients

- Chrome installation
- Provided downloads

↓ macOS

macOS

Drag the download Profile into the Admin Console, associate to test devices and deploy once tested.

Contents of Profile:

```
Dict {
    PayloadDisplayName = Google Chrome Classspaces
    PayloadScope = System
    PayloadType = Configuration
    PayloadRemovalDisallowed = false
    ConsentText = Dict {
        default =
    }
    PayloadContent = Array {
        Dict {
            PayloadVersion = 1
            PayloadDisplayName = Custom: (com.google.Chrome)
            PayloadType = com.apple.ManagedClient.preferences
            PayloadContent = Dict {
                com.google.Chrome = Dict {
                    Forced = Array {
                        Dict {
                            mcx_preference_settings = Dict {
                                ExtensionSettings = Dict {
                                    * = Dict {
                                        installation_mode = blocked
                                    }
                                    obeophmpnnhboefjagnpbllfbbaeodnn = Dict {
                                        installation_mode = force_installed
                                        update_url = https://clients2.google.com/service/update2/crx
                                        comment = Classspaces
                                    }
                                }
                            }
                        }
                    }
                }
            }
```

```
            PayloadEnabled = true
            PayloadIdentifier = ml1063.lan.c7256e52-e8f0-4b6b-b48d-
5ec98c03ff8a.com.apple.ManagedClient.preferences.87789162-48a1-42bf-b444-ff6567a9e7f0
            PayloadUUID = 87789162-48a1-42bf-b444-ff6567a9e7f0
        }
    }
    PayloadIdentifier = ml1063.lan.c7256e52-e8f0-4b6b-b48d-5ec98c03ff8a.Configuration.c7256e52-e8f0-4b6b-b48d-
5ec98c03ff8a
    PayloadVersion = 1
    PayloadUUID = c7256e52-e8f0-4b6b-b48d-5ec98c03ff8a
}
```

Notes

⚠ As part of the example, an additional key has been added to block all other extensions. Remove or edit as desired

✅ If Chrome is already open, the settings will not be applied until after the browser has been restarted

Related Content

- Google Chrome Extension Management (Windows)

Google Chrome Install Recipe (macOS)

Description

Installation of Chrome or any app that simply goes in /Applications can be fairly simple to do with a Fileset, but if the application is presented as a PKG that does other things besides simply copying a file to /Applications, then this can be a good example of how to accomplish the install. Note that this Fileset also will download the latest Google Chrome at install time, so depending on your use case, this may not be a good solution. An alternative would be to put the PKG inside of the FileSet and use the same script to install it or make a PKG-based FileSet in FileWave. There are many roads to success. This example will show several features of Filesets for teaching purposes.

Ingredients

- FW Central
- Chrome already as a Fileset (below)

Directions

1. Download the Chrome Fileset zip file, expand and drag the Fileset into FileWave > Filesets
2. Create an Association between the Fileset and one or more macOS systems
3. See Chrome installed on them.

What does this Fileset do?

The Fileset is entirely 2 scripts. If you highlight the Fileset in the Native Admin and pick the Scripts button in the toolbar you can edit the 2 scripts. The first is the Activation Script. This will create a directory in /private/tmp/ and download Chrome to it. It will install and then delete the installer from that folder. Because of the way this works you may want to make this script a little fancier with error handling, but the below works and is a good starting point. You will also see that it logs what it is doing to /Library/Libgs/GoogleCrhomeInstallScript.log so that you can review the date and time that events happened.

```
#!/bin/zsh

pkgfile="GoogleChrome.pkg"
logfile="/Library/Logs/GoogleChromeInstallScript.log"
url='https://dl.google.com/chrome/mac/stable/gcem/GoogleChrome.pkg'

/bin/echo "--" >> ${logfile}
/bin/echo "`date`: Downloading latest version." >> ${logfile}
mkdir /private/tmp/chrome_install/
/usr/bin/curl -s -o /private/tmp/chrome_install/${pkgfile} ${url}

/bin/echo "`date`: Installing..." >> ${logfile}
/usr/sbin/installer -pkg /private/tmp/chrome_install/GoogleChrome.pkg -target /

/bin/sleep 5

/bin/echo "`date`: Deleting package installer." >> ${logfile}
/bin/rm -rf /private/tmp/chrome_install

exit 0
```

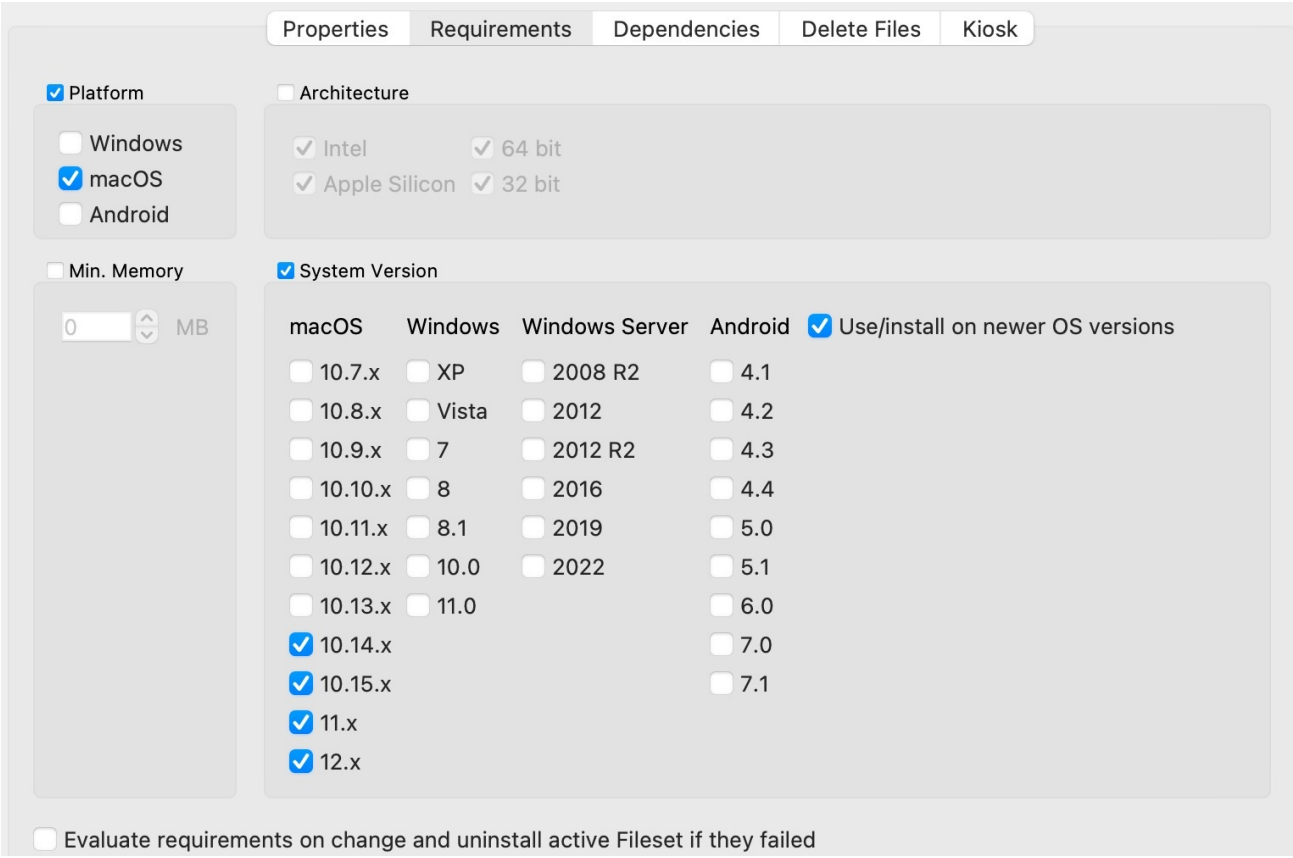
Note that in these days of Intel vs. Apple Silicon you may want to add something to behave differently depending on which processor is present below is an easy way in scripting to do this;

```
if [[ $(uname -p) == 'arm' ]]; then
  echo M1
else
  echo Intel
fi
```

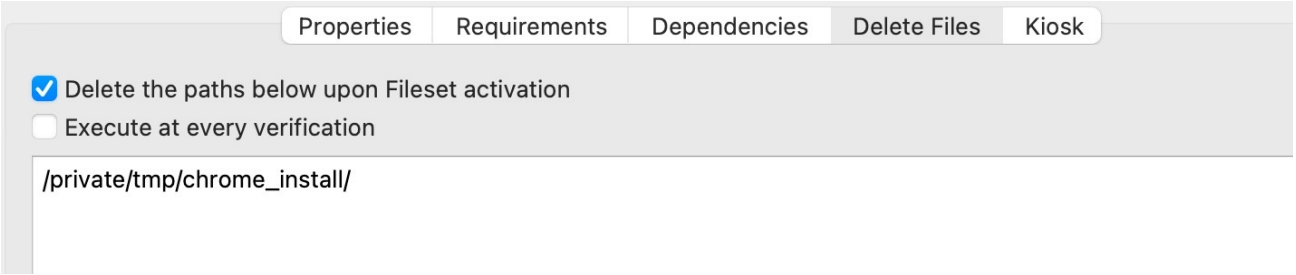
The second script is for when you remove the Association for the Fileset. It will kill Chrome if it is running and then delete it from /Applications. If you don't want this behavior then just remove that script from the Fileset.

```
#!/bin/zsh

logfile="/Library/Logs/GoogleChromeInstallScript.log"

/bin/echo "--" >> ${logfile}
/bin/echo "`date`: Removing Google Chrome." >> ${logfile}

killall Google\ Chrome

rm -rf "/Applications/Google Chrome.app"

exit 0
```

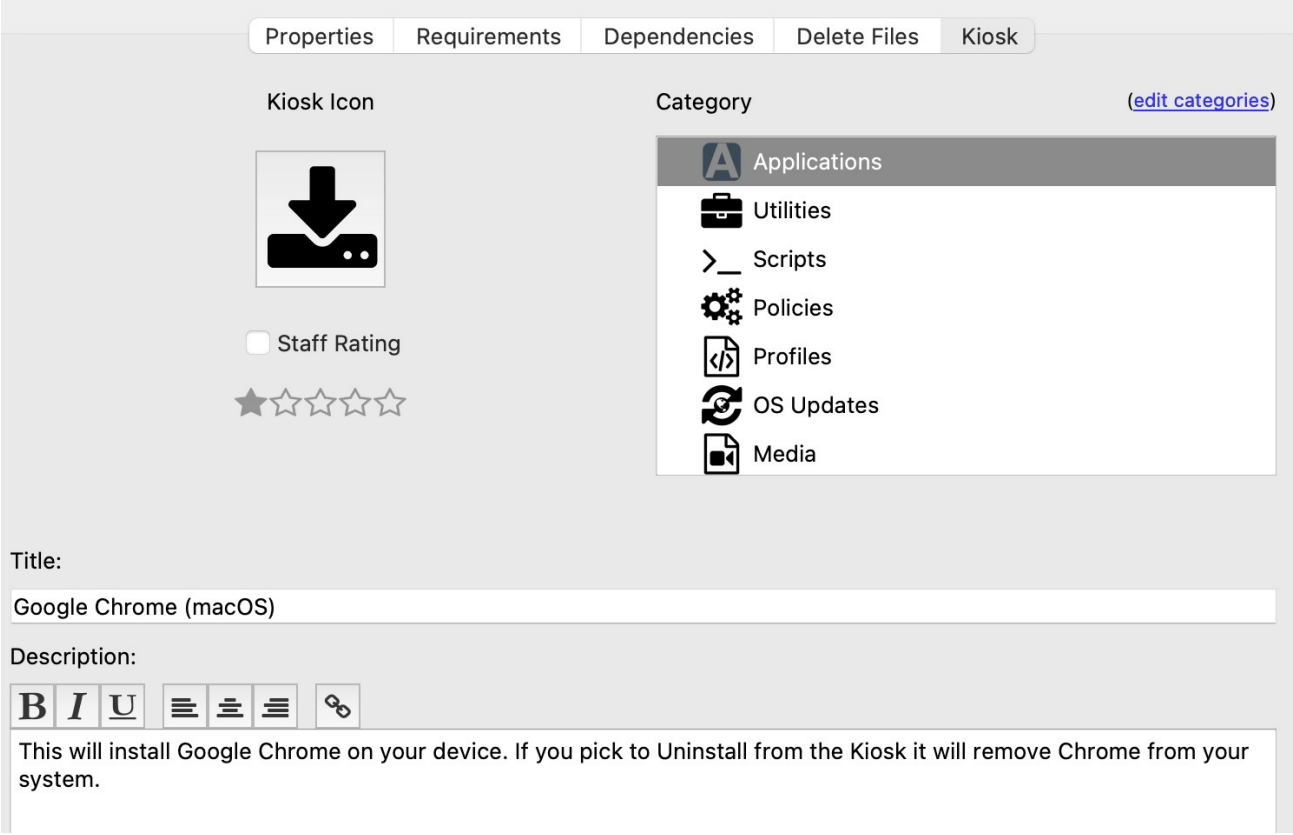
For Requirements, I set it for macOS and only the most recent versions since (1) Anything below macOS 10.14.x is insecure to use, and (2) I don't know that Chrome will work well on older macOS, but adjust this to your needs.



For Delete Files, I have it purging the download files so that when the Fileset activates, the folder is clear every time.



On the Kiosk tab, I set this to Applications and put a description in. Using a Kiosk Association is a great way to make an application installable by non-admins and they can even uninstall it if they aren't using it any more.



So with this example, you can see how you might create a scripted install of an application and handle uninstall of it as well. Keep in mind you don't have to do this exactly as I did, and these concepts can be applied to other applications.

> After installing Chrome, some configuration may be desirable, examples may include:
> * Block auto updates
> * Set home page
> * Disable Welcome Page
>
> This article discusses configuration options: Google Chrome Configuration Recipe

Related Content

- Google Chrome Configuration recipe (macOS)
- Google Chrome Extension Management (macos)