

White Labeling FileWave Components

Starting with FileWave 12.7, it is possible to customize end-user components and change FileWave defaults.

- [White Labeling FileWave components](#)
- [White Labeling the iOS, macOS, and Android enrollment pages](#)
- [White Labeling the OTA Enrollment profile and MDM profile](#)
- [White Labeling the Chromebook Extension](#)

White Labeling FileWave components

Starting with FileWave 12.7, it is possible to customize end-user components and change FileWave defaults.

Please use the How-To articles below for detailed customization instructions.

FileWave Feature	Components	How-To Articles to Follow
iOS Management	MDM enrollment profile, App Portal Web Clip	White Labeling the OTA Enrollment profile, MDM profile, and App Portal Web Clip
	App portal as native app (logo and color)	Setting the Primary Colour, Name and Logo in Kiosk/App Portal (16.0+)
	Enrollment page	White Labeling the iOS, macOS and Android enrollment pages
Android Management	Enrollment page	White Labeling the iOS, macOS and Android enrollment pages
macOS Management	MDM enrollment profiles	White Labeling the OTA Enrollment profile and MDM profile
	Self Service Kiosk	Setting the Primary Colour, Name and Logo in Kiosk/App Portal (16.0+)
	Enrollment page	White Labeling the iOS, macOS and Android enrollment pages
Windows Management	Self Service Kiosk	Setting the Primary Colour, Name and Logo in Kiosk/App Portal (16.0+)

White Labeling the iOS, macOS, and Android enrollment pages

Getting Started

This article explains how to customize the enrollment page for iOS, macOS (MDM), and Android devices - the enrollment page is what is made visible for your end users when they enroll their devices manually (BYOD or manual enrollment).

Feature	Customization type	Technology
iOS, macOS MDM, Android	Server template	HTML

Managed Platform	Server Platform	Location
Android	macOS and Linux	<code>/usr/local/filewave/django/user_templates/android/welcome.html</code>
iOS and macOS	macOS and Linux	<code>/usr/local/filewave/django/user_templates/ios/welcome.html</code>

This HTML document is actually a template, containing variables that will be replaced by the real values when processed by the FileWave server. We are using [Django template language](#) - but in a very limited way; only a few variables are important.

iOS and MacOS

variable	usage
<code>static</code>	Path where FileWave stores static files like icons, css files
<code><already_trusted_ca</code>	True if the server has a valid, trusted SSL certificate

The template will show one or two steps depending on the SSL certificate your server is using.

In case of self signed certificate:

- step 1 is to download the certificate (to be manually installed in the device trust store); link must point to "/CA"
- step 2 is to enroll the device; link must point to "/enroll"

In case of trusted certificate (recommended):

- step 1 is to enroll the device; link must point to "/enroll"

Android

variable	usage
<code>static</code>	Path where FileWave stores static files like icons, css files
<code>filewave_version</code>	FileWave server current version. "12.4.0" for instance
<code>server_host</code>	FileWave server FQDN as set in preferences. " server.filewave.ch " for instance
<code>server_port</code>	FileWave server port as set in preferences. 20443 by default.

Android template always shows two steps:

- step 1 is to download FileWave Android Native App Portal on FileWave CDN, using `filewave_version` variable. You may want to change this link if you have a custom Android Native App Portal.
- step 2 is to automatically configure the Android Native App Portal, once it has been installed, using `server_host` and `server_port` variables.

Additional files

Additional files can be placed in the same directory and will be served with the `/user_templates/ios/` prefix. For instance, if you want to include a CSS file, you can add to your `index.html`:

```
<link rel='stylesheet' href="/user_templates/ios/enroll.css" />
```

And copy "enroll.css" in the same directory as `index.html`.

White Labeling the OTA Enrollment profile and MDM profile

Getting Started

It is possible to white label the OTA enrollment profile, the MDM profile, and the Web Clip for the App Portal. These profiles are the ones used while enrolling an Apple device. White Labeling the MDM profile also allows customizing the information displayed for the profile in the Settings iOS app.

The process of White Labeling these profiles consists of copying template files located on the MDM server to the correct location and editing them with a plain text editor. These files are read by the MDM server and persist server upgrades.

All templates are located in the following directory on the server:

- macOS and Linux: `/usr/local/filewave/django/user_templates`

In the "user_templates" directory, you will see the following templates that can be used as reference point when making your customizations:

- enrolment_ios.example.plist
- enrolment_macos.example.plist
- mdm_profile.example.plist

In order to customize the profile, find the file "xxxx.example.plist". Copy the file to another one in the same directory removing ".example" from the name, e.g. copy "xxxx.example.plist" to "xxxx.plist"). You can then edit the file in a text editor.

For the purpose of demonstrating the feature, we will use the fake company name "Argon5".

OTA enrollment profile

The OTA enrollment profile is used during the first step of MDM enrollment. The information it contains is displayed by the OS while the enrollment is in progress. There are 2 files you can customize:

- "enrolment_ios.example.plist", for iOS
- "enrolment_macos.example.plist", for macOS

These files are plists that look like this:

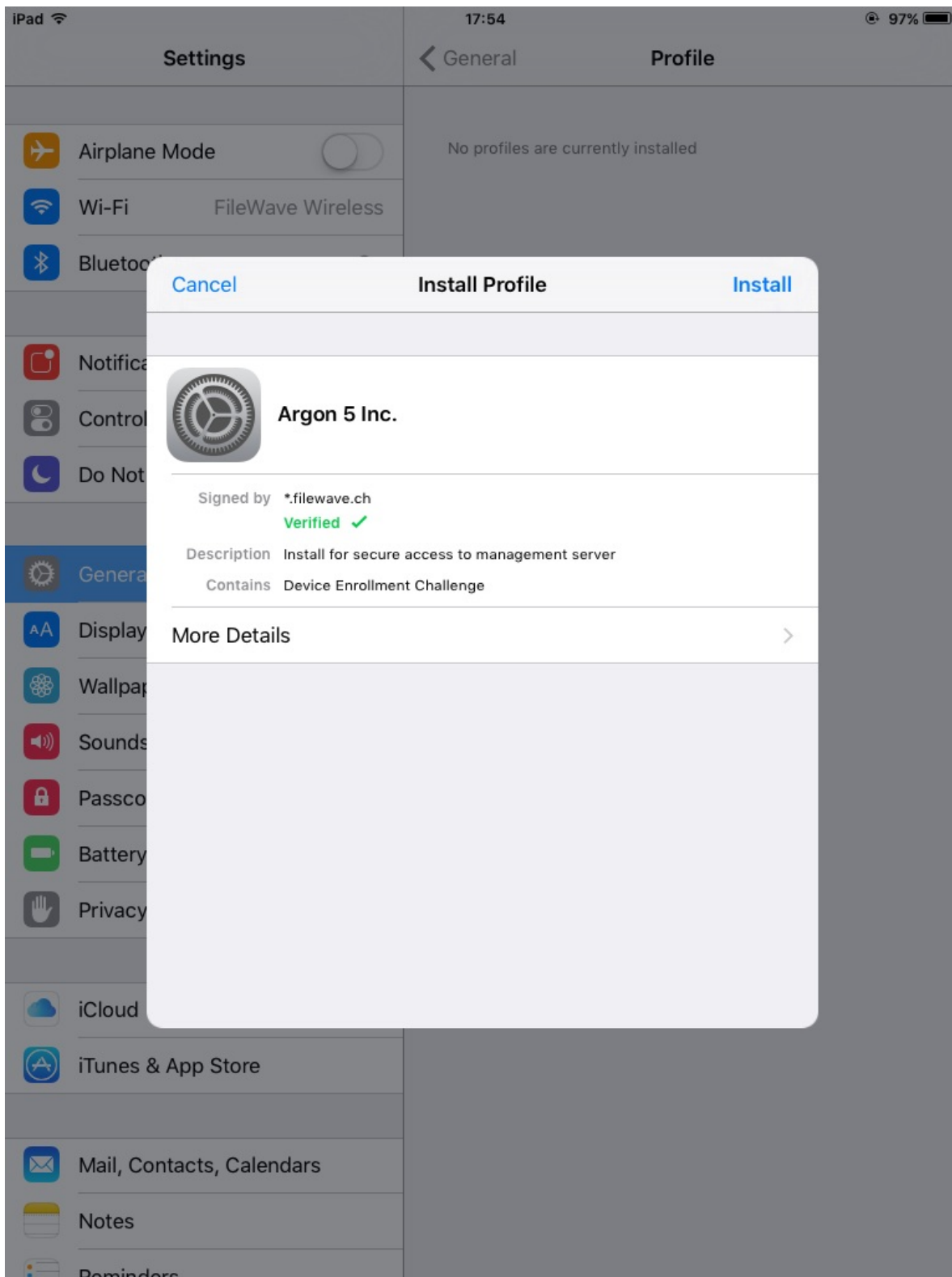
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Inc//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">

<!--
Note: define your values to substitute the OTA enrolment profile here.
Values present on the enrolment profile and not here will not be substituted.
Every key/value set here is final: they will be present on the enrolment profile.
-->

<plist version="1.0">
  <dict>
    <key>PayloadIdentifier</key>
    <string>com.argonfive.mobileconfig.profile-service</string>
    <key>PayloadDisplayName</key>
    <string>Argon 5 Inc.</string>
  </dict>
</plist>
```

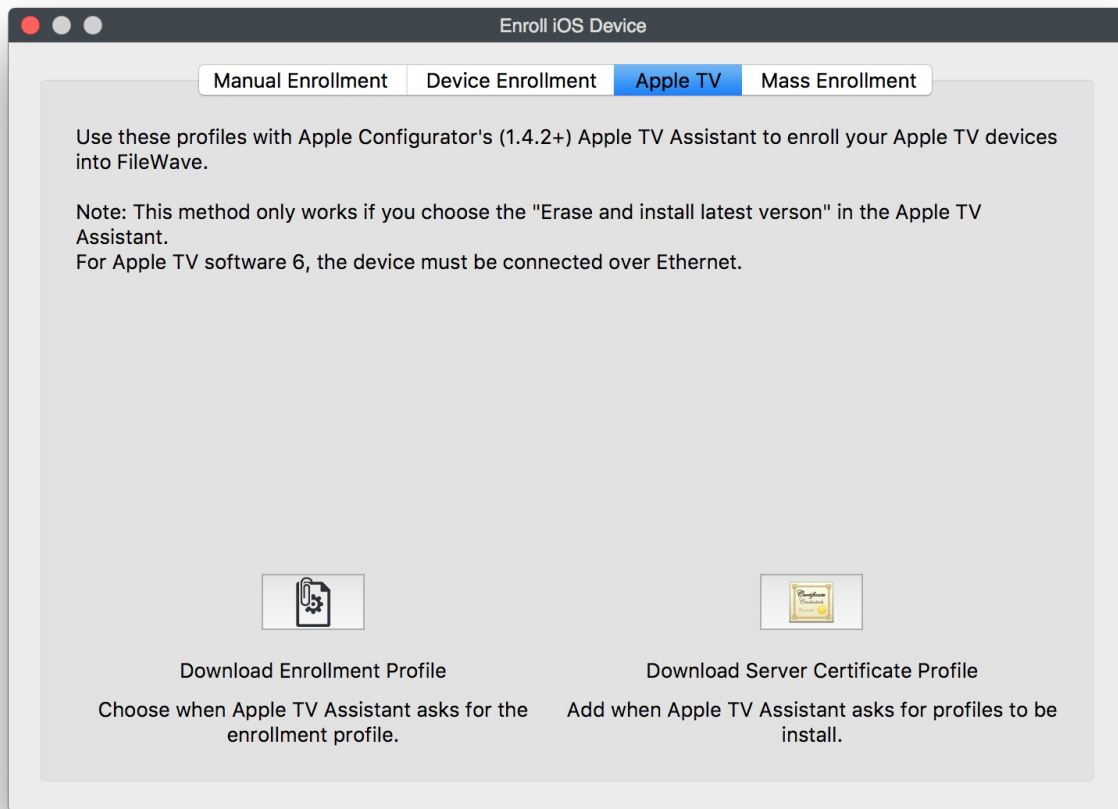
To customize the profile, replace the strings with the value that you would like to see displayed. If you don't want to customize any key, it can safely be removed from the .plist file. This will cause the default value (with FileWave theming) to be sent instead.

The result can be seen while enrolling an iOS device:



Apple TV Enrollment Profile

The file `enrollment_ios.plist` is also used to customize the profile to enroll Apple TVs with Apple Configurator.



MDM profile

Basic customization

The MDM profile template file is "mdm_profile.example.plist". You can replace any keys in the same way as explained in the previous section.

Sample customized MDM profile

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

<!--
Note: define your values to substitute the MDM profile here.
Values present on the MDM profile and not here will not be substituted.
Every key/value set here is final: they will be present on the MDM profile.
-->

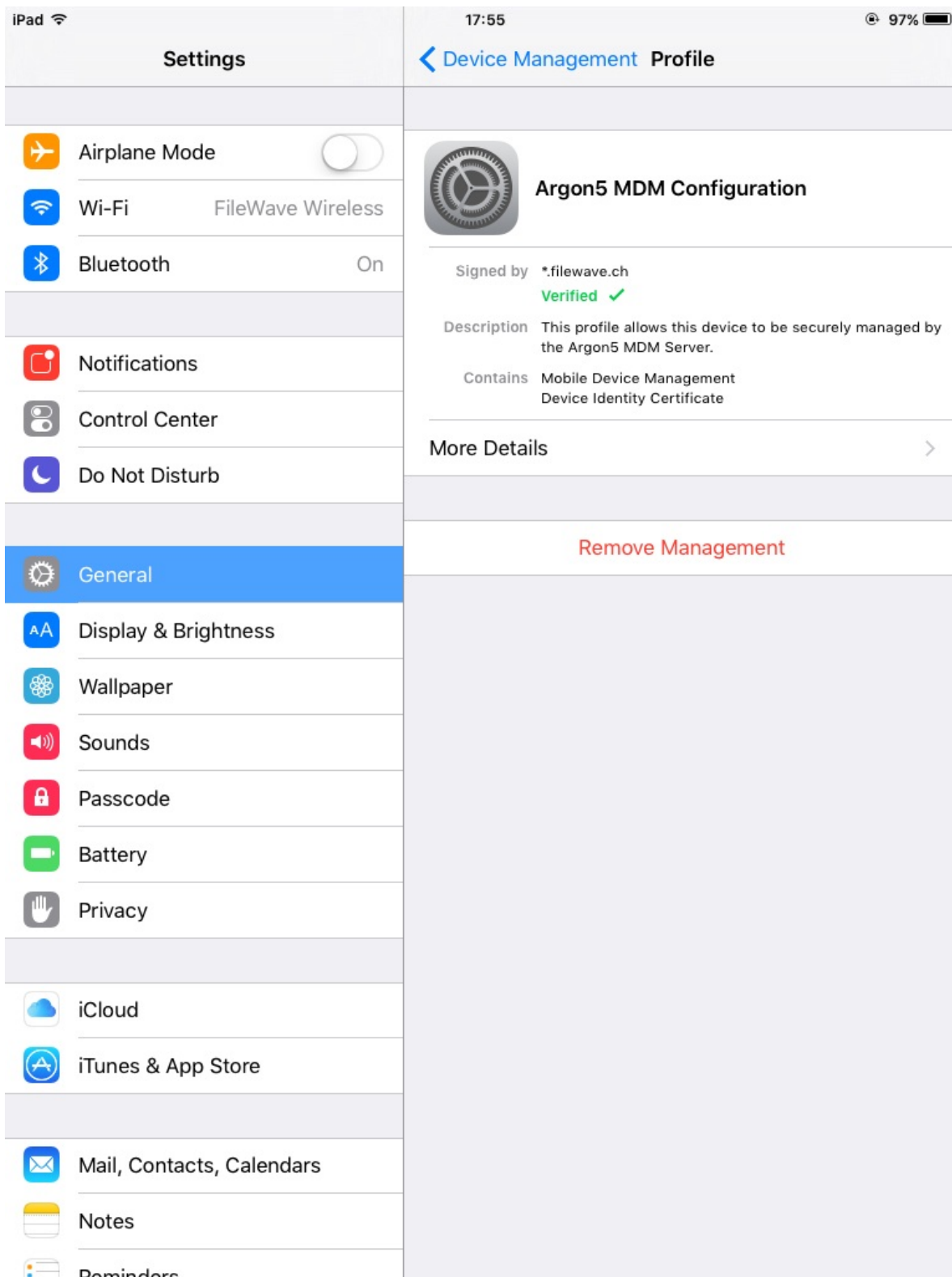
<plist version="1.0">
<dict>
  <key>PayloadContent</key>
  <array>
    <dict>
      <key>PayloadContent</key>
      <dict>
        <key>Name</key>
        <string>Argon5 SCEP</string>
        <key>Subject</key>
        <array>
          <array>
            <array>
              <string>0</string>
              <string>Argon5 SCEP</string>
```

```

        </array>
    </array>
    <array>
        <array>
            <string>CN</string>
            <string>argonfivemdm</string>
        </array>
    </array>
</dict>
</dict>
<dict>
    <key>PayloadDescription</key>
    <string>Installs the Root certificate for the Argon5 MDM solution.</string>
</dict>
</array>
<key>PayloadDescription</key>
<string>This profile allows this device to be securely managed by the Argon5 MDM Server.</string>
<key>PayloadDisplayName</key>
<string>Argon5 MDM Configuration</string>
</dict>
</plist>

```

Which is displayed in iOS like this:



Consent text

An additional key that you might want to add is "ConsentText". When this key is defined, iOS and macOS display a consent text during the enrollment process. The consent text must be explicitly accepted by the user. This can be used to request the user to accept terms and conditions before continuing with the enrollment. If this key is missing, this step will be skipped.

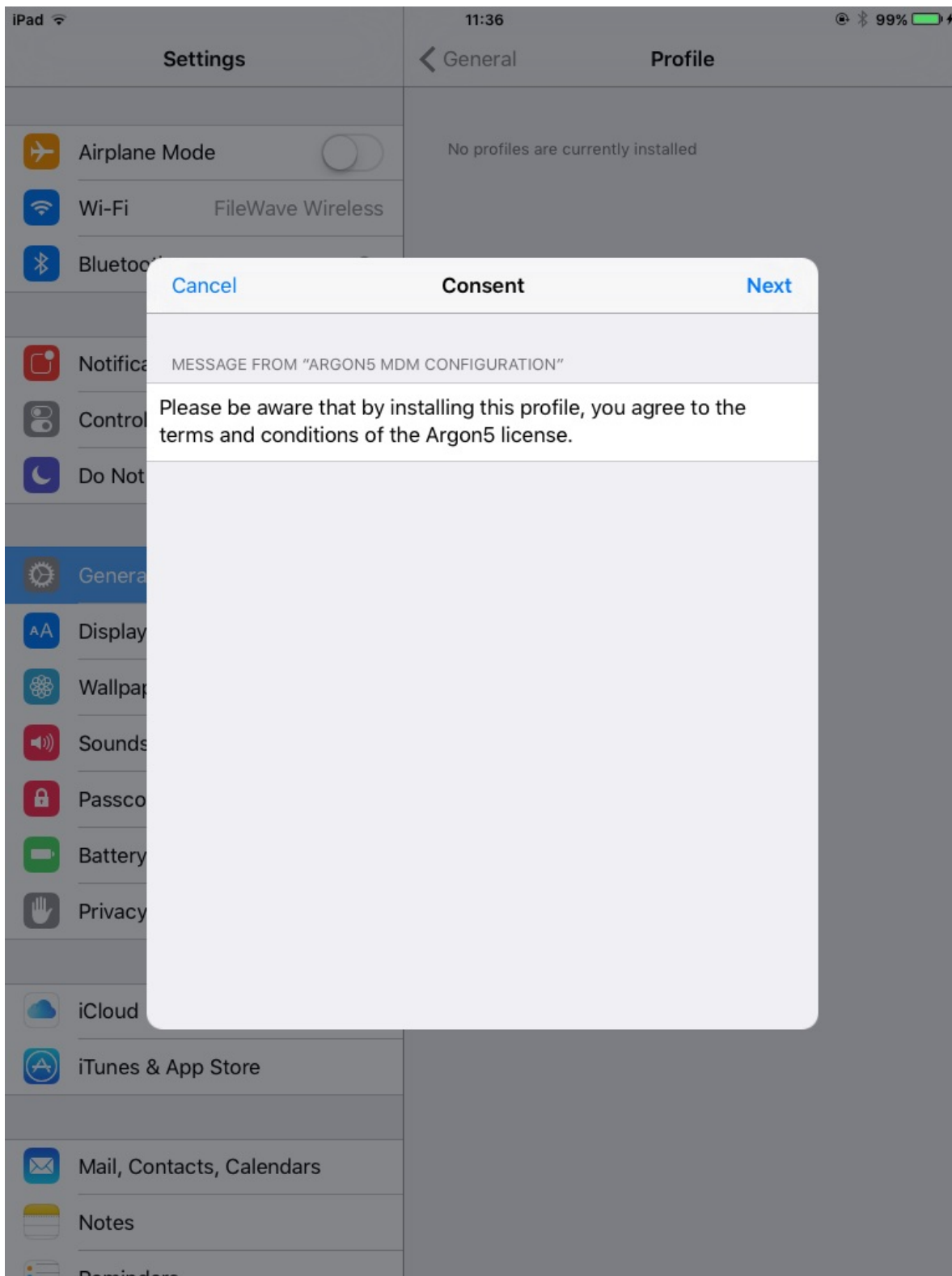
Here is an example of how to add the ConsentText at the end of the MDM profile:

```
...  
  <key>ConsentText</key>  
  <dict>  
    <key>default</key>  
    <string>Please be aware that by installing this profile, you agree to the terms and conditions of the  
Argon5 license.</string>  
  </dict>
```



```
</dict>  
</plist>
```

This consent text will be displayed by iOS like this:



Related Content

- [Setting the Primary Colour, Name and Logo in Kiosk/App Portal \(16.0+\)](#)

White Labeling the Chromebook Extension

What

It was previously possible to customize the user-visible information for the FileWave Inventory Extension. It was possible to customize icons, name and description. This involved:

1. Download [chromebook-whiteboxing.zip](#) and modify the data inside (adding a new icon, description, etc.)
2. Create a support Ticket titled "Chromebook Custom Extension" and attach your new modified Zip File
3. Wait till we send back your "whiteboxed" extension then continue to the "How to publish" section below

When/Why

As part of an overhaul of the extension to leverage new APIs and enhancements the ability to publish it with customizations is removed for now. If this returns in a future version then this article will be updated.

How

This customization is not possible in FileWave 15.4+ and this article remains to make customers aware of the change. You would use the inventory extension as outlined in [Chromebook Client Pre-Requisites](#) rather than this customized method.

Related Content

- [Chromebook Client Pre-Requisites](#)

Digging Deeper

This was the actual process to publish in the past.

The ZIP file contained:

- icon_.png (so far we have 19, 38, 128 as size)
- data.json

data.json

```
{
  "name": "<NAME_HERE>",
  "description": "<DESCRIPTION_HERE>"
}
```

How to publish a "whiteboxed" version of the extensions

Once you get the "whiteboxed" version of your extension, you have to follow some steps to make it available to your organization.

1. Login/Create an account to [Google Developer Dashboard](#)
2. Click on the "Add New Item" button

As of November 21st, 2016, all newly published packaged or hosted apps are restricted to Chrome OS, and are not available to users on Windows, Mac or Linux. Existing apps will continue to be available on all major platforms and will continue to receive updates. - [More Info](#)

Note: This change does not apply to Google Drive Apps or Add-Ons for Google Apps.

Warning: Recent phishing email campaign targeting Chrome Web Store developers



A number of developers have recently reported receiving [phishing emails](#) from email addresses that impersonate the Chrome Web Store policy team.

If you receive any emails that appear to be from the Chrome Web Store but do not belong to the google.com domain (for example, chromewebstore@gmail.com), please use your gmail controls to mark the email as spam, and send the [original email headers](#) to [Chrome Web Store Developer Support](#).

We also encourage you to increase your account security by [enabling 2-step verification](#). You can also consider adding the [Password Alert Chrome Extension](#), which can help identify phishing attacks.

[Dismiss](#)

Developer Dashboard

Your Listings (1 - 2 of 2)		Created	Last published	Weekly users	Status	
	FileWave Inventory					Publish Edit
	Version 0.6.1 - filewave.com Internal	11/30/16	12/21/16	1	Draft	More info
	★★★★★ (0) Target users in other languages. More info ▼					
	FileWave Engage for Chromebooks					Publish Edit
	Version 1.0.5 - Public	1/23/17		0	Draft	More info
	★★★★★ (0) Target users in other languages. More info ▼					

[Add new item](#)

Your Developer Account

3. Upload the ZIP file

[Developer Dashboard](#) > [Add new item](#)

Upload an extension or app (.zip file)

Uploading an item:

- Upload a ZIP file of your item directory, not a packaged CRX file.
- Include a well-designed product icon in your manifest ([more info](#)).
- [Read the documentation](#) about creating and packaging apps.
- Need more help? Check out the [Chrome Web Store developer documentation](#).

4. Edit/add Additional Info

Edit Item

In-app Products

Upload



FileWave Test

Short name: Not specified in manifest

Version 12.5.0 by brandony

[Upload Updated Package](#)

This is a description for this test app.

Detailed
description

Icon



[Upload new icon](#)

5. Publish the extension