

Scripting Languages supported in FileWave

Description

FileWave provides the ability to leverage certain scripting languages on macOS and Windows. This includes:

	macOS	Windows
Perl	✓	✓
Python	✓	✓
Shell	✓	
Bat		✓
PowerShell		✓

FileWave does not include these languages, they are either installed by the OS vendor or will need to be installed separately.



Apple indicated they would be deprecating pre-installation of certain runtimes, for example:
https://developer.apple.com/documentation/macos-release-notes/macos-catalina-10_15-release-notes

When testing any scripts locally prior to deployment through FileWave, it is imperative that they are tested in the same context/environment as if they were being ran by FileWave, as indicated in our [Script Best Practices](#) KB.

Script Extensions

Fileset scripts are best added using the Script button from the Fileset view. Doing so, ensures the script will also log all output automatically, as well as setting a default [shebang](#) for macOS shell scripts.

Additionally, when defining the name of the script, the extension is required to match the language.



Only one of the following extensions will allow the editing of the Executable tab, regardless of how they the scripts are added.

Language	Extension
Perl	.pl
Python	.py
Shell	.sh
Batch	.bat or .cmd
PowerShell	.ps1



EXE files may also benefit from the Executables tab



Specify the type of Shell script, by way of the shebang. To keep in line with Apple, FileWave will automatically add ZSH as the default shebang: `#!/bin/zsh`

Example

The below provides examples of how to instal Python on endpoints. This is one way to achieve this goal, but by no means the only way. Professional Services requests can be raised for items beyond standard FileWave Support if desired.

Ingredients

- FileWave Central
- Windows EXE

Installers available from either:

- <https://www.python.org/ftp/python/>
- <https://www.python.org/downloads/windows/>

Directions

macOS Python Installer

▼ macOS Python

Apple used to provide a version of Python pre-installed, however as noted, this was an old version and is now deprecated. It is possible to download Python installers, however Apple provide Python in the Xcode Command Line Tools. The following method demonstrates how the softwareupdate mechanism may be used to list (and therefore instal) the command line tools:

```
# touch /tmp/.com.apple.dt.CommandLineTools.installondemand.in-progress
# softwareupdate -l
Software Update Tool

Finding available software
Software Update found the following new or updated software:
* Label: Command Line Tools for Xcode-12.4
  Title: Command Line Tools for Xcode, Version: 12.4, Size: 440392K, Recommended: YES,
* Label: Command Line Tools for Xcode-13.2
  Title: Command Line Tools for Xcode, Version: 13.2, Size: 577329K, Recommended: YES,
* Label: Command Line Tools for Xcode-12.5
  Title: Command Line Tools for Xcode, Version: 12.5, Size: 470966K, Recommended: YES,
* Label: Command Line Tools for Xcode-12.5
  Title: Command Line Tools for Xcode, Version: 12.5, Size: 470820K, Recommended: YES,
* Label: macOS Big Sur 11.6.2-20G314
```

Note that more than one version may be returned. If Xcode is installed a version may already be in place, so care should be taken if Xcode is already installed. After installing the chosen version, the temporary file created may then be removed.

Instal Python Packages

Python scripts import packages, not all of which will be installed by default. Any additional packages will require installation. PIP may achieve this and may be used in a Fileset script; PIP is also installed when installing the Command Line tools (or Xcode)


It is possible to check pip with the following command, note it may require upgrading, but again take careful consideration if Xcode is installed:

```
# pip3 list
Package      Version
-----
pip          20.2.3
setuptools  49.2.1
six          1.15.0
wheel        0.33.1

WARNING: You are using pip version 20.2.3; however, version 23.1.2 is available.
You should consider upgrading via the '/Library/Developer/CommandLineTools/usr/bin/python3 -m pip install --
upgrade pip' command.
```

For example, to instal pyobjc:

```
# Instal pyobjc
pip3 install pyobjc-core
pip3 install pyobjc-framework-Cocoa
pip3 install pyobjc-framework-Quartz
pip3 install pyobjc-framework-SystemConfiguration
```

 In some instances, pip3 may fail. This can be due to the link created in:
/Library/Developer/CommandLineTools/SDKs
Remove and recreate the link to the correct installed version if need be.

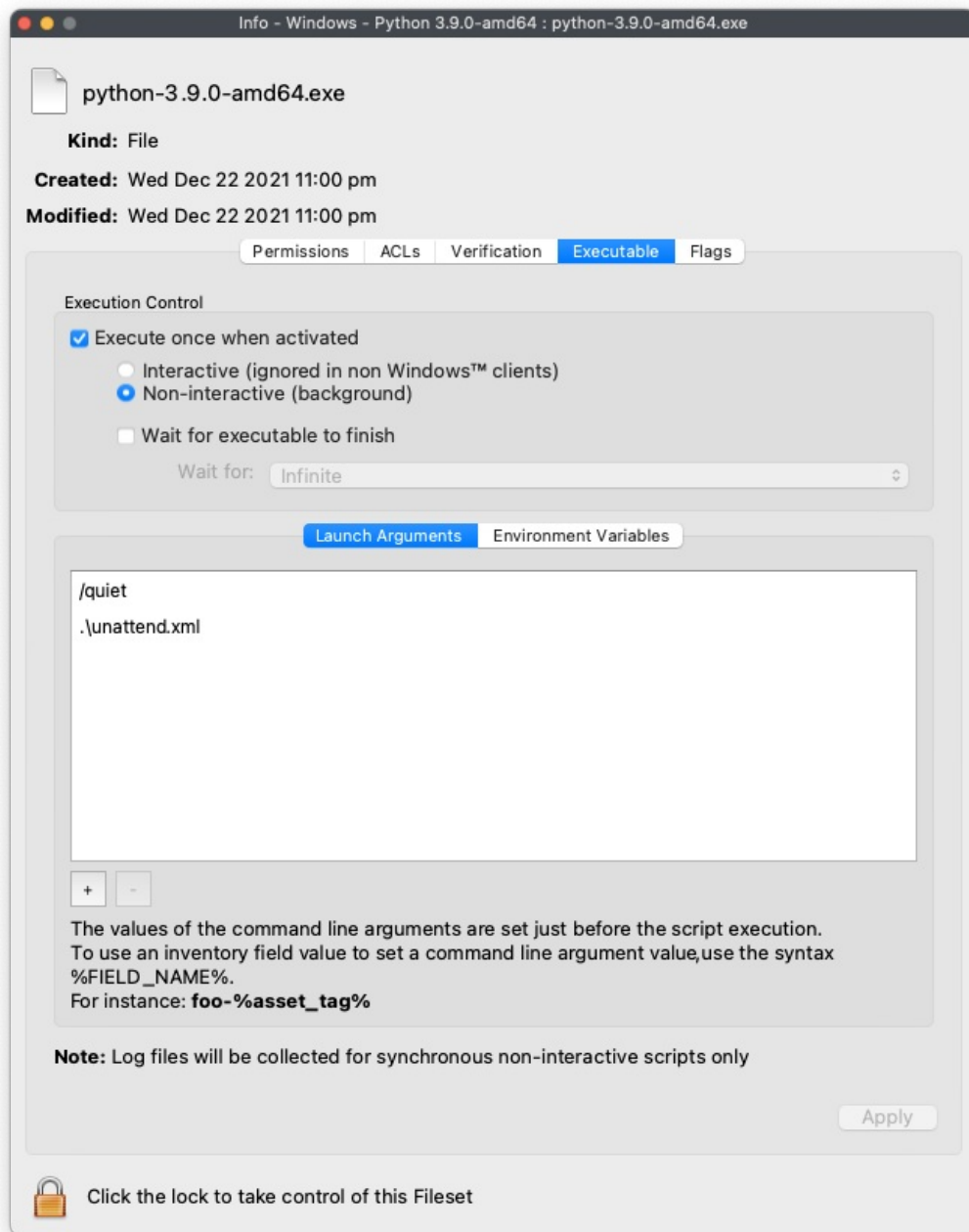
The above could be packaged into a single script which could:

- Touch the temporary file
- Run Software Update
- Obtain the desired version from the list
- Instal that version
- Remove the temporary file
- Remove the link if incorrect
- Use PIP to instal other desired packages
- Recreate the link if removed above

Windows Python Installer

▼ Windows Python

1. Download the appropriate python installer EXE
2. Create a new empty Fileset
3. Choose a location to store the installer and drag the EXE into this location
4. Highlight this EXE and from the Get Info window choose Executable
5. Tick the option to 'Execute once when activated'
6. Set two Launch Arguments as per the screen shot: /quiet and .\unattend.xml



Supporting File

An unattend.xml file may be used to pre-determine aspects of the installation, e.g. instal PIP during installation or set Paths

1. Use an editor to create a file called unattend.xml
2. Edit this file with the below details, editing to meet desired needs
3. Place this file in the Fileset within the same folder location as the EXE installer

An example could look as follows (see the [Python documentation](#) for a list and description of available options):

```
<Options>
<Option Name="InstallAllUsers" Value="1" />
<Option Name="TargetDir" Value="C:\Program Files\Python39" />
```

```
<Option Name="DefaultAllUsersTargetDir" Value="C:\Program Files\Python39" />
<Option Name="DefaultJustForMeTargetDir" Value="C:\Program Files\Python39" />
<Option Name="DefaultCustomTargetDir" Value="C:\Program Files\Python39" />
<Option Name="AssociateFiles" Value="1" />
<Option Name="CompileAll" Value="1" />
<Option Name="PrependPath" Value="1" />
<Option Name="Shortcuts" Value="1" />
<Option Name="Include_doc" Value="1" />
<Option Name="Include_debug" Value="1" />
<Option Name="Include_dev" Value="1" />
<Option Name="Include_exe" Value="1" />
<Option Name="Include_launcher" Value="1" />
<Option Name="InstallLauncherAllUsers" Value="1" />
<Option Name="Include_lib" Value="1" />
<Option Name="Include_pip" Value="1" />
<Option Name="Include_symbols" Value="1" />
<Option Name="Include_tcltk" Value="1" />
<Option Name="Include_test" Value="1" />
<Option Name="Include_tools" Value="1" />
<Option Name="LauncherOnly" Value="0" />
<Option Name="SimpleInstall" Value="0" />
<Option Name="SimpleInstallDescription"></Option>
</Options>
```

Edit the above example as desired, for example, set the target directory paths as desired.

Instal Python Packages

Python scripts import packages, not all of which will be installed by default. Any additional packages will require installation. PIP may achieve this and may be used in a post flight script, if the option above to include pip at installation was configured.

1. Create a PostFlight script within the Fileset
2. Edit the script to include any desired packages, e.g. to instal the 'requests' package:

```
python -m pip install requests
```

- 1 The command 'pip list' may be used to list ALL additional packages installed.
The command 'pip freeze' may be used to list packages installed by PIP.

The final Fileset will may look something like:

Fileset Contents: Windows - Python 3.9.0-amd64					
<div> <div>Import File/Folder</div> <div>New Folder</div> <div>Get Info</div> <div>Edit Registry</div> <div>Edit Text</div> <div>Export Files</div> <div>Delete</div> </div>					
Revision: <default> (Initial Revision)			Manage Revisions		
<input checked="" type="checkbox"/> Hide unused folders					
Name	Size	ID	Access	User	
temp		9890	rw-rw-r-x	root	
python-3.9.0-amd64.exe	26.9 MB	716495	rw-r--r--	root	
unattend.xml	1.1 kB	716881	rw-r--r--	root	
var		1406	rw-r-xr-x	root	
scripts		1458	rw-rw-r-x	root	
716172		717044	rw-rw-r-x	root	
py_requests.ps1	30 B	717045	r-x-----	root	

Packages, including PIP itself, may require updating over time.

Testing

The following download is a simple Python Custom Field for both macOS and Windows. It should report the date when ran.

macOS12VM - Client Info					
<div> <div>Last Connected: 04/07/2023 09:55</div> <div>From: 192.168.1.90</div> <div>Free Space: 47.1 GB</div> <div>Platform: macOS 12 Monterey</div> <div>Model: 9770</div> <div>Version: 14.7.0 (Rev. 7ae10fa7)</div> <div>Enrollment Type: User approved enrollment</div> <div>Missing Updates: 11 missing 0 critical</div> </div>					
<div> <div>Export Current Tab</div> <div>Client Monitor</div> <div>Get Log</div> <div>Verify</div> <div>Tools</div> </div>					
<div> <div>Filesets Status</div> <div>Device Details</div> <div>Command History</div> <div>Managed Apps</div> <div>Installed Apps</div> <div>Installed Profiles</div> <div>Users</div> <div>Policies</div> <div>Software Updates</div> </div>					
<div> <div>Edit Custom Field(s) Values...</div> <div>Date Time</div> </div>					
Property	Last Update Time	Status	Value		
Date Time	04/07/2023 09:48	Success	04/07/2023 09:48		



If desired, use the Custom Field Assistant window to Import this unzipped, Custom Field example, called 'Date Time'

🕒Revision #8

★Created 2 July 2023 12:57:05 by Josh Levitsky

✎Updated 9 April 2025 16:43:50 by Sean Holden