Beyond the good ol' LaunchAgents - 9 - Preference Pane

theevilbit.github.io/beyond/beyond_0009

March 25, 2021

> This is part 9 in the series of "Beyond the good ol' LaunchAgents", where I try to collect various persistence techniques for macOS. For more background check the introduction.

Preference panes on macOS are plugins to the `System Preferences.app`. These panes can extend the functionality of the app, and typically allow you to modify configuration settings for your app. These admins are loaded when the user selects them, so they are not perfect from persistence point of view, as it requires user interaction, but can still be a thing.

Preference panes are located in:

- `/System/Library/PreferencePanes` - This folder contains the default macOS panes
- `/Library/PreferencePanes` - Here we fine the panes that are installed for all users. Note that these will still run as the logged in user, when opened, and not as root.
- `~/Library/PreferencePanes` - Finally here we can find the per-user installed preference panes.

Let's see how we can create one. Apple has a very detailed documentation about Preference Panes, which we can find here. It has a section called "Implementing a Simple Preference Pane", which can be used to create one from scratch.

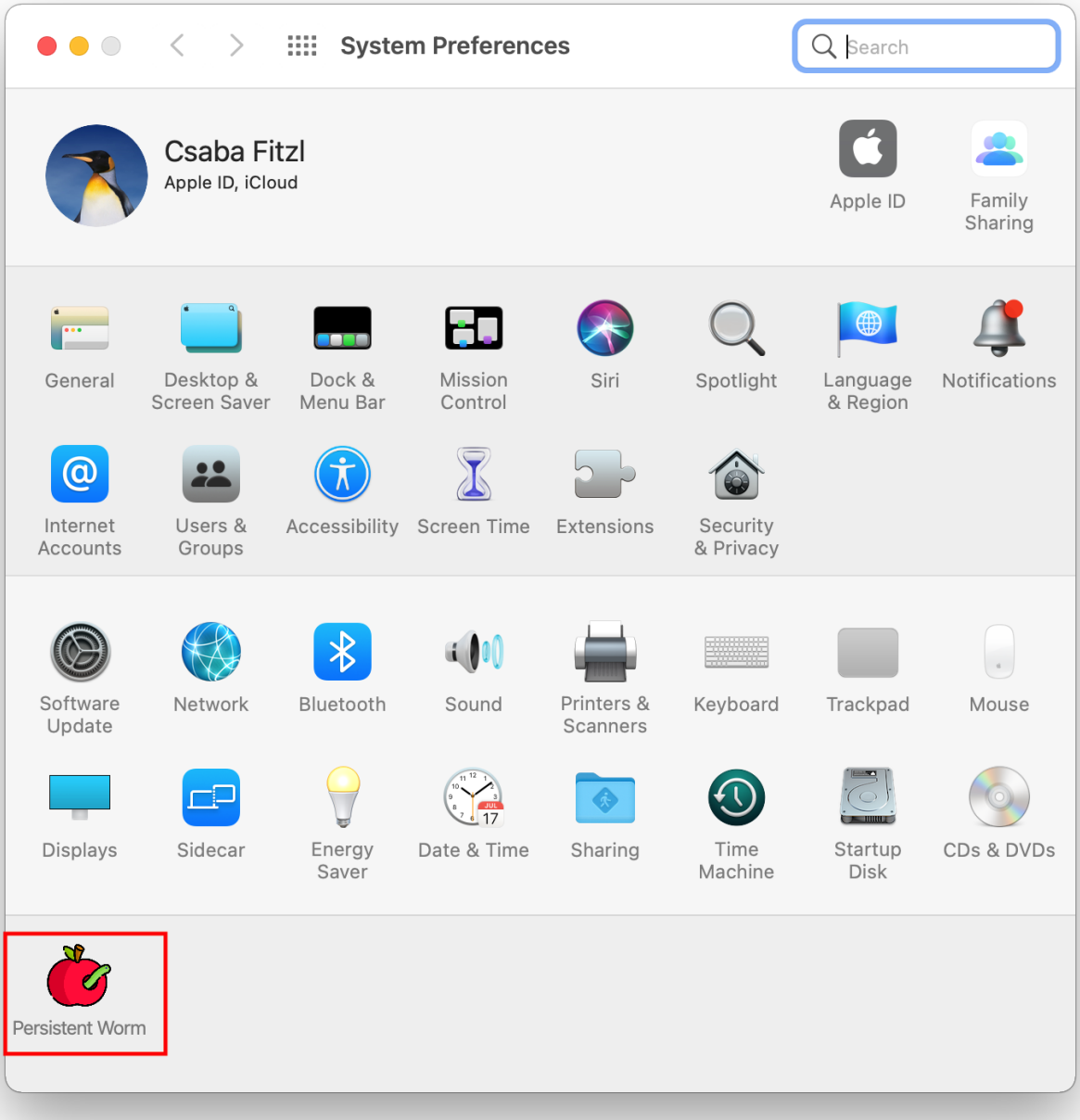
The easier method if we just want to experiment with it is to download their sample code, PrefsPane. This is an old code that was created for OS X 10.6, so we would get a few alerts, but essentially it compiles easily. I did some changes to this project, giving a new name, new icon, fixed some alerts, and uploaded it on GitHub - theevilbit/macos. It's called `Persistent Worm`.

There are multiple methods in the code, the one that will be called when loaded is `initWithBundle:`.

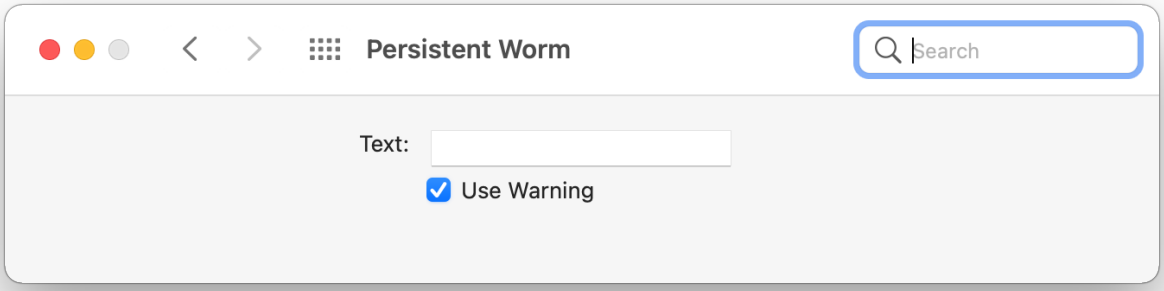
```
-(id)initWithBundle:(NSBundle*)bundle
{
        if ((self = [super initWithBundle:bundle]) != nil)
        {
                // do more initialization here
        }
    NSLog(@"PersistentWorm");
        return self;
}
```

I added some logging to it for tracking.

Once we compile the bundle will be copied into `~/Library/PreferencePanes`. Then we can go ahead and open `System Preferences.app`.



The new entry with the name `Persistent Worm` is the one we created, and it has an apple logo with a worm. We can open it.



It will display the window setup in the code.

Let's check who loads this.

```
csaby@mac ~ % log show --predicate "eventMessage contains[c] 'Worm'" --last 5m
Filtering the log data using "composedMessage CONTAINS[c] "Worm""
Skipping info and debug messages, pass --info and/or --debug to include.
Timestamp                       Thread     Type        Activity             PID    TTL
2021-03-25 09:56:37.466027+0100 0x5e6f0a   Default     0xc09cb5             16414  0    legacyLoader-x86_64: (PrefsPane)
PersistentWorm
---------------------------------------------------------------------------------------------------------------
Log      - Default:          1, Info:               0, Debug:            0, Error:          0, Fault:          0
Activity - Create:           0, Transition:         0, Actions:          0
```

If we search for our `Worm` log with the `log` utility we will see that the process loading the binary is `legacyLoader-x86_64`, at least on Big Sur 11.2. This is an XPC service used by `System Prefrences` and can be found at `/System/Library/Frameworks/PreferencePanes.framework/Versions/A/XPCServices/legacyLoader-x86_64.xpc/Contents/MacOS/legacyLoader-x86_64`.

If we check its code signing properties we will see that it allows the load of third party code due to the `com.apple.security.cs.disable-library-validation` entitlement.

```
Executable=/System/Library/Frameworks/PreferencePanes.framework/Versions/A/XPCServices/legacyLoader-
x86_64.xpc/Contents/MacOS/legacyLoader-x86_64
Identifier=com.apple.systempreferences.legacyLoader.x86_64
Format=bundle with Mach-O thin (x86_64)
CodeDirectory v=20100 size=672 flags=0x0(none) hashes=13+5 location=embedded
Platform identifier=11
Signature size=4577
Signed Time=2020. Dec 22. 1:26:09
Info.plist entries=23
TeamIdentifier=not set
Sealed Resources version=2 rules=2 files=0
Internal requirements count=1 size=96
<?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>com.apple.security.cs.disable-library-validation</key>
        <true/>
</dict>
</plist>
```

On macOS Catalina the loader was `/System/Library/Frameworks/PreferencePanes.framework/Versions/A/XPCServices/legacyLoader.xpc/Contents/MacOS/legacyLoade` with the entitlement `com.apple.security.cs.disable-library-validation`.

Update 2021.04.06.

As suggested by @xnyhps we can enable the auto-loading of our preference pane with registering an alert for it.

```
csaby@mac ~ % defaults delete com.apple.systempreferences DidShowPrefBundleIDs
csaby@mac ~ % defaults write com.apple.systempreferences AttentionPrefBundleIDs -dict 'com.yourcompany.Persistent-Worm' 1
```

In the above listing we register an alert for our pane by bundle ID. We also clear the status if it has been showed before. This will result in our preference pane loading, when the user opens `System Preferences`.