How to Use launchd to Run Services in macOS

Introduction to manage macOS service with examples.

> *In computing,* ***launchd****, a unified operating system service management framework, starts, stops and manages daemons, applications, processes, and scripts in macOS. It was introduced with Mac OS X Tiger and is licensed under the Apache License.—* ***Wikipedia***

If you've familiar with any version of Linux, surely you have worked with cron jobs. Basically, **launchd** is the cron in macOS. Other than executing scripts cron-style, **launchd** does a lot more. Like **systemd** on Linux, **launchd** is a replacement for a lot of old school Unix tools, like init, inetd, cron, etc.

At it's core, **launchd** distinguishes daemons and agents. Daemons are system-wide services that always run in the background, while agents describe regular services that are to be executed on user-specific events. Daemons and agents are managed by **launchd** framework which can be controlled using the **launchctl** terminal command.

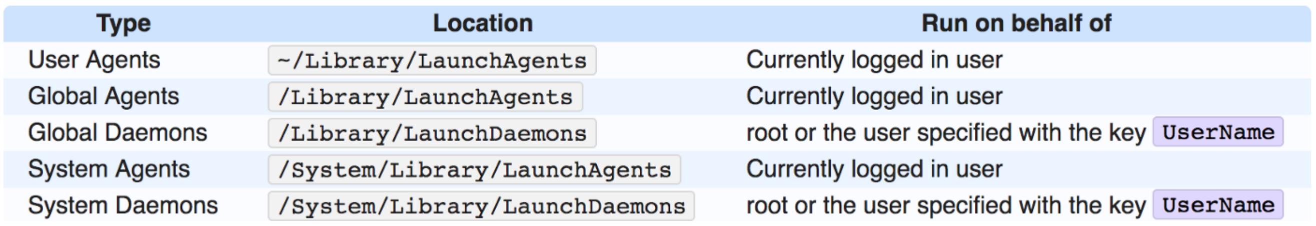
There are two main components in the launchd service creating process that you will need to follow.

1. **Job Definition** : Define the service in special XML file.

2. **Operations** : Control the service using command line utility.

📋 Job Definition

The behavior of a daemons or agents are defined in a special XML file called a **property list** (***.plist***) file. Depending on where it is stored it will be treated as a daemon or an agent. Services (Daemons and Agents) are saved as **.plist** files and can reside in one of the following folders according to your requirement as given below.



Type	Location	Run on behalf of
User Agents	`~/Library/LaunchAgents`	Currently logged in user
Global Agents	`/Library/LaunchAgents`	Currently logged in user
Global Daemons	`/Library/LaunchDaemons`	root or the user specified with the key `UserName`
System Agents	`/System/Library/LaunchAgents`	Currently logged in user
System Daemons	`/System/Library/LaunchDaemons`	root or the user specified with the key `UserName`

launchd supports more than 36 different configuration keys in **property list** file and only main keys that required for a basic service are described in the below. (from launchd.plist(5) manual)

```
Label <string>
     This required key uniquely identifies the job to launchd.

Program <string>
     This key defines what to start. If this key is missing, then
the first element of the array of strings provided to the
ProgramArguments will be used instead.  This key is required in the
absence of the ProgramArguments key.

ProgramArguments <array of strings>
     Use this one if your executable requires command line options.
This key is required in the absence of the Program key.

RunAtLoad <boolean>
     This optional key is used to control whether your job is
launched once at the time the job is loaded. The default is false.

StartInterval <integer>
     This optional key causes the job to be started every N seconds.
If the system is asleep, the job will be started the next time the
computer wakes up.  If multiple intervals transpire before the
computer is woken, those events will be coalesced into one event
upon wake from sleep.

StartCalendarInterval <dictionary of integers or array of dictionary
of integers>
     This optional key causes the job to be started every calendar
interval as specified. Missing arguments are considered to be
wildcard. The semantics are much like crontab(5).

  Minute <integer>
     The minute on which this job will be run.
  Hour <integer>
     The hour on which this job will be run.
  Day <integer>
     The day on which this job will be run.
  Weekday <integer>
     The weekday on which this job will be run (0 and 7 are Sunday).
  Month <integer>
     The month on which this job will be run.
```

Please refer launchd.plist(5) manual or launchd.info for detailed definitions of other keys in property list files.

⚠️ ***Please note:*** property list files are expected to have their name end in "***.plist***". Also please note that it is the expected convention for launchd property list files to be named **<Label>.plist**. Thus, as an example if your job label is "*org.wso2.am*", your **plist** file should be named "org..wso2.am.plist".

⚙ Operations

This section will define how to obtain information about launchd services and how to load, unload, start and stop those jobs. All of this can be accomplished using the command line utility **launchctl**.

> ***launchctl** interfaces with **launchd** to load, unload daemons/agents and generally control launchd. launchctl supports taking subcommands on the command line, interactively or even redirected from standard input.*
> — ***launchctl(1) manual***

Some of the popular commands in **launchctl** are defined in the below.

- Getting information about available (loaded) jobs :

```
$ launchctl list
```

- Getting information about a given job :

```
$ launchctl list | grep <LABEL>
```

- Loading a job (*a global daemon*) :

```
$ launchctl load /Library/LaunchDaemons/<LABEL>.plist
```

- Unloading a job (*a global daemon*) :

```
$ launchctl unload /Library/LaunchDaemons/<LABEL>.plist
```

- Starting a job (*a loaded job*) :

```
$ launchctl start <LABEL>
```

- Stoping a job (*a loaded job*) :

```
$ launchctl stop <LABEL>
```

- Restarting a job (*a loaded job*) :

```
$ launchctl restart <LABEL>
```

⚠️ ***Please note:*** If you are working with Daemon job please use ***sudo*** permission with above commands.

🎗Simple Example (using WSO2 API Manager)

In this article, I am hoping to run a WSO2 API Manager server as a service on macOS system as an example to understand above explained methods. WSO2 API Manager addresses full API lifecycle management, monetization, and policy enforcement and it has been named as a Leader in The Forrester Wave™: API Management Solutions, Q4 2018 Report.

1. Download the Latest WSO2 API Manager distribution from WSO2 Official Website.
2. Extract the API Manager zip and go the WSO2 API Manager directory. (we will call the path to this directory as **<PRODUCT_HOME>**)
3. Download and install OpenJDK 8 or Oracle JDK 1.8.* and set the `JAVA_HOME` environment variable.
4. Create a property file named "**org.wso2.am.plist**" at "**/Library/LaunchDaemons/**". Open the "**org.wso2.am.plist**" file in a text editor and copy the following content to the property list (.plist) file. Please replace **<PRODUCT_HOME**> with path to the product directory.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
    "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Label</key>
    <string>org.wso2.am</string>
    <key>ServiceDescription</key>
    <string>WSO2 API Manager Server</string>
    <key>ProgramArguments</key>
    <array>
        <string><PRODUCT_HOME>/bin/wso2server.sh</string>
    </array>
    <key>RunAtLoad</key>
    <false/>
</dict>
</plist>
```

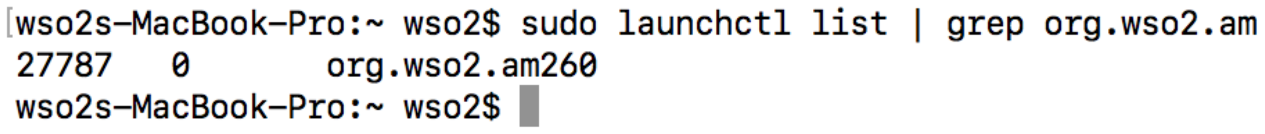
5. Run the following commands to load and start the WSO2 API Manager as service using **launchctl** utility.

```
$ sudo launchctl load /Library/LaunchDaemons/org.wso2.am.plist

$ sudo launchctl start org.wso2.am
```

6. To check the status of the job run the following command and you will able to observe need information about the jobs as given in the image.

```
$ sudo launchctl list | grep org.wso2.am
```



```
wso2s-MacBook-Pro:~ wso2$ sudo launchctl list | grep org.wso2.am
27787   0       org.wso2.am260
wso2s-MacBook-Pro:~ wso2$
```

7. Run following commands to stop the running process and unload the daemon service from **launchd** framework.

```
$ sudo launchctl stop org.wso2.am

$ sudo launchctl unload /Library/LaunchDaemons/org.wso2.am.plist
```

Therefore, with the understanding of the simple example of the launchd framework now you can use launchd to create scripts that do things like clean up files, control application servers on a schedule or run an application when a certain file appears.

Cheers!! 🍺🍺

More details on **launchd** framework and **launchctl** utility:

A launchd Tutorial

A launchd primer covering configuration, administration and troubleshooting. Complete with examples.

man page launchd.plist section 5

Daemons or agents managed by launchd are expected to behave certain ways. A daemon or agent launched by launchd MUST…

man page launchctl section 1

load [-wF] [-S sessiontype] [-D domain] paths ... Load the specified configuration files or directories of config…