**Creating a Java Daemon (System Service) for Debian using Apache Commons Jsvc**

A [daemon](http://en.wikipedia.org/wiki/Daemon_%28computing%29) is a computer program that runs as a [background](http://en.wikipedia.org/wiki/Background_(computer_software)) [process](http://en.wikipedia.org/wiki/Process_(computing)), executing tasks on a predefined schedule or in response to particular [events](http://en.wikipedia.org/wiki/Event_(computing)), or in response to requests for information or services from other programs.

To start up a Java program as a daemon using Jsvc, we’ll need to do a few things:

a) Implement the Apache Commons Daemon [interface](http://docs.oracle.com/javase/tutorial/java/concepts/interface.html).

b) Install Jsvc by [downloading](http://commons.apache.org/proper/commons-daemon/jsvc.html)and compiling the source (C) or via the command line with the sudo apt-get install jsvc command.

c) Create a bash shell script to launch the daemon and manage the basic set of daemon controls, namely: **start**, **stop**& **restart.**Others include **reload**& **status**.

## Implementing the Java Daemon Interface

Implementing the Daemon interface is a fairly simple undertaking. Once you have downloaded the library from the [Apache Commons project site](http://commons.apache.org/proper/commons-daemon/jsvc.html) and referenced it in your Java project, you can go about creating your implementation as follows:

package *za.co.neilson*;  
  
import *org.apache.commons.daemon.Daemon*;  
import *org.apache.commons.daemon.DaemonContext*;  
import *org.apache.commons.daemon.DaemonInitException*;  
  
public class MyDaemon implements Daemon {  
  
    private [Thread](http://www.google.com/search?hl=en&q=allinurl%3Athread+java.sun.com&btnI=I%27m%20Feeling%20Lucky) myThread;   
    private boolean stopped = false;  
    private boolean lastOneWasATick = false;  
      
    @Override  
    public void init(DaemonContext daemonContext) throws DaemonInitException, [Exception](http://www.google.com/search?hl=en&q=allinurl%3Aexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {  
        */\**  
*\* Construct objects and initialize variables here.*  
*\* You can access the command line arguments that would normally be passed to your main()*  
*\* method as follows:*  
*\*/*  
        [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky)[] args = daemonContext.getArguments();   
          
        myThread = new [Thread](http://www.google.com/search?hl=en&q=allinurl%3Athread+java.sun.com&btnI=I%27m%20Feeling%20Lucky)(){  
            private long lastTick = 0;  
              
            @Override  
            public synchronized void start() {  
                MyDaemon.this.stopped = false;  
                super.start();  
            }  
  
            @Override  
            public void run() {               
                while(!stopped){  
                    long now = [System](http://www.google.com/search?hl=en&q=allinurl%3Asystem+java.sun.com&btnI=I%27m%20Feeling%20Lucky).currentTimeMillis();  
                    if(now - lastTick >= 1000){  
                        [System](http://www.google.com/search?hl=en&q=allinurl%3Asystem+java.sun.com&btnI=I%27m%20Feeling%20Lucky).out.println(!lastOneWasATick ? "tick" : "tock");  
                        lastOneWasATick = !lastOneWasATick;  
                        lastTick = now;   
                    }  
                }  
            }  
        };  
    }  
    @Override  
    public void start() throws [Exception](http://www.google.com/search?hl=en&q=allinurl%3Aexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {  
        myThread.start();  
    }  
    @Override  
    public void stop() throws [Exception](http://www.google.com/search?hl=en&q=allinurl%3Aexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {  
        stopped = true;  
        try{  
            myThread.join(1000);  
        }catch([InterruptedException](http://www.google.com/search?hl=en&q=allinurl%3Ainterruptedexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) e){  
            [System](http://www.google.com/search?hl=en&q=allinurl%3Asystem+java.sun.com&btnI=I%27m%20Feeling%20Lucky).err.println(e.getMessage());  
            throw e;  
        }  
    }  
      
    @Override  
    public void destroy() {  
        myThread = null;  
    }  
}

Jsvc seems to require that our program is exported to a jar, which in this instance I have stored at /usr/local/mydaemon/MyDaemon.jar and the Apache Commons Daemon library has been stored at/usr/local/mydaemon/lib/commons-daemon-1.0.15.jar.

## Creating a bash shell script to launch the daemon and manage the basic set of daemon controls

Debian daemon init scripts are stored in /etc/init.d/. A skeleton init script, (/etc/init.d/skeleton) is available to copy and modify according to your needs and generally would provide a good starting point for your services’ init script, however, writing an init script for [Jsvc](http://commons.apache.org/proper/commons-daemon/jsvc.html" \o "Daemon: Java Service" \t "_blank)is a little different.

So assuming that that your Daemon implementation’s jar file and the Apache Commons Daemon library are stored at the above locations A basic example of a bash init script for our MyDaemon service could look like this:

*#! /bin/sh*  
*#  /etc/init.d/mydaemon*  
  
*### BEGIN INIT INFO*  
*# Provides:          mydaemon*  
*# Required-Start:    $remote\_fs $syslog*  
*# Required-Stop:     $remote\_fs $syslog*  
*# Short-Description: Starts the MyDaemon service*  
*# Description:       This file is used to start the daemon*  
*#                    and should be placed in /etc/init.d*  
*### END INIT INFO*  
  
*# Author:   Sheldon Neilson <sheldon[AT]neilson.co.za>*  
*# Url:      www.neilson.co.za*  
*# Date:     25/04/2013*  
  
NAME="mydaemon"  
DESC="MyDaemon service"  
  
*# The path to Jsvc*  
EXEC="/usr/bin/jsvc"  
  
*# The path to the folder containing MyDaemon.jar*  
FILE\_PATH="/usr/local/$NAME"  
  
*# The path to the folder containing the java runtime*  
JAVA\_HOME="/usr/lib/jvm/default-java"  
  
*# Our classpath including our jar file and the Apache Commons Daemon library*  
CLASS\_PATH="$FILE\_PATH/MyDaemon.jar:$FILE\_PATH/lib/commons-daemon-1.0.15.jar"  
  
*# The fully qualified name of the class to execute*  
CLASS="za.co.neilson.MyDaemon"  
  
*# Any command line arguments to be passed to the our Java Daemon implementations init() method*  
ARGS="myArg1 myArg2 myArg3"  
  
*#The user to run the daemon as*  
USER="root"  
  
*# The file that will contain our process identification number (pid) for other scripts/programs that need to access it.*  
PID="/var/run/$NAME.pid"  
  
*# System.out writes to this file...*  
LOG\_OUT="$FILE\_PATH/log/$NAME.out"  
  
*# System.err writes to this file...*  
LOG\_ERR="$FILE\_PATH/err/$NAME.err"  
  
jsvc\_exec()  
{     
    cd $FILE\_PATH  
    $EXEC -home $JAVA\_HOME -cp $CLASS\_PATH -user $USER -outfile $LOG\_OUT -errfile $LOG\_ERR -pidfile $PID $1 $CLASS $ARGS  
}  
  
case "$1" in  
    start)    
        echo "Starting the $DESC..."          
          
        *# Start the service*  
        jsvc\_exec  
          
        echo "The $DESC has started."  
    ;;  
    stop)  
        echo "Stopping the $DESC..."  
          
        *# Stop the service*  
        jsvc\_exec "-stop"         
          
        echo "The $DESC has stopped."  
    ;;  
    restart)  
        if [ -f "$PID" ]; then  
              
            echo "Restarting the $DESC..."  
              
            *# Stop the service*  
            jsvc\_exec "-stop"  
              
            *# Start the service*  
            jsvc\_exec  
              
            echo "The $DESC has restarted."  
        else  
            echo "Daemon not running, no action taken"  
            exit 1  
        fi  
            ;;  
    \*)  
    echo "Usage: /etc/init.d/$NAME {start|stop|restart}" >&2  
    exit 3  
    ;;  
esac

The init script above should be stored in /etc/init.d/ along with the system’s other start-up scripts.

## Automatically Starting the Daemon at System Boot

With our init script done, we need to change it’s permissions to make it executable using chmod. chmod 755 /etc/init.d/mydaemon. And register it for execution as the system’s run level changes.

*“Debian uses a*[*Sys-V*](http://en.wikipedia.org/wiki/UNIX_System_V)*like system for executing commands when the system [runlevel](http://www.debianhelp.co.uk/runlevels.htm" \o "Debian Run Levels" \t "_blank)changes, for example at boot and shutdown time.” – Source:*[*Making scripts run at boot time with Debian*](http://www.debian-administration.org/articles/28)

As the system’s [runlevel](http://www.debianhelp.co.uk/runlevels.htm" \o "Debian Run Levels" \t "_blank) changes, any scripts within corresponding /etc/rc\*.d/ directory will be executed, where \* is the number of the runlevel.

The easiest way to have our script executed appropriately is to use the Debian command [update-rc.d](http://manpages.debian.net/cgi-bin/man.cgi?sektion=8&query=update-rc.d&apropos=0&manpath=sid&locale=en) to add the appropriate symlinks for us.

To enable our daemon at boot use update-rc.d mydaemon defaults:

root@sheldon-pc:/etc/init.d# update-rc.d mydaemon defaults

Adding system startup for /etc/init.d/mydaemon...

/etc/rc0.d/K20mydaemon -> ../init.d/mydaemon

/etc/rc1.d/K20mydaemon -> ../init.d/mydaemon

/etc/rc6.d/K20mydaemon -> ../init.d/mydaemon

/etc/rc2.d/S20mydaemon -> ../init.d/mydaemon

/etc/rc3.d/S20mydaemon-> ../init.d/mydaemon

/etc/rc4.d/S20mydaemon -> ../init.d/mydaemon

/etc/rc5.d/S20mydaemon-> ../init.d/mydaemon

If you should ever wish to disable it, run update-rc.d -f mydaemon remove:

root@sheldon-pc:/etc/init.d# update-rc.d -f mydaemon remove

update-rc.d: /etc/init.d/mydaemon exists during rc.d purge (continuing)

Removing any system startup links for /etc/init.d/blah ...

/etc/rc0.d/K20mydaemon

/etc/rc1.d/K20mydaemon

/etc/rc2.d/S20mydaemon

/etc/rc3.d/S20mydaemon

/etc/rc4.d/S20mydaemon

/etc/rc5.d/S20mydaemon

/etc/rc6.d/S20mydaemon

Our daemon should now start when the system boots. To view it’s output, first we should stop it to prevent any concurrent access issues when we open the log file: /etc/init.d/mydaemon stop.

As we know that our log file is being created at /usr/local/mydaemon/log/mydaemon.out, we can view it using the command vi /usr/local/mydaemon/log/mydaemon.out

root@sheldon-pc:/etc/init.d# vi /usr/local/mydaemon/log/mydaemon.out

tick

tock

tick

tock

tick

tock...

This entry was posted in **[Debian](http://www.neilson.co.za/category/linux/debian/" \o "View all posts in Debian)**, [**Java**](http://www.neilson.co.za/category/java/), [**Linux**](http://www.neilson.co.za/category/linux/) and tagged [**Bash**](http://www.neilson.co.za/tag/bash/), [**Daemon**](http://www.neilson.co.za/tag/daemon/), [**Debian**](http://www.neilson.co.za/tag/debian/), [**Java**](http://www.neilson.co.za/tag/java/), [**Jsvc**](http://www.neilson.co.za/tag/jsvc/), [**Linux**](http://www.neilson.co.za/tag/linux/), [**Service**](http://www.neilson.co.za/tag/service/), [**Unix**](http://www.neilson.co.za/tag/unix/), [**update-rc.d**](http://www.neilson.co.za/tag/update-rc-d/) by [**Sheldon Neilson**](http://www.neilson.co.za/author/sheldon-neilson/). Bookmark the [**permalink**](http://www.neilson.co.za/creating-a-java-daemon-system-service-for-debian-using-apache-commons-jsvc/).