#!/bin/bash

#####################################################################################

# #

# Name of Script: checkstatusTomcat.sh #

#####################################################################################

##Purpose : To check status of tomcat

kill -0 `cat $CATALINA\_PID` > /dev/null 2>&1

if [ $? -gt 0 ]

then

echo "Check tomcat status " | mailx -s "Tomcat not running" support@dom.com

fi

##Parameter : $1-Error status mail

MailIt ()

{

Subject=`hostname`

Body=$1

echo $Body | mail -s ${Subject} -c "awinash.kumar@tieto.com" -- -r "from the xpris"

}

#!/bin/sh  
PID=$(ps -eo pid,comm | awk '$2=="jboss" {print $2}')  
echo $PID  
if [ -z "$PID"]; then  
echo No Jboss running  
else  
echo All done  
fi

PID=$(ps -ef | awk '/jbos[s]/ {print $2}')

#!/bin/bash

# Check if gedit is running

# -x flag only match processes whose name (or command line if -f is

# specified) exactly match the pattern.

if pgrep -x "gedit" > /dev/null

then

echo "Running"

else

echo "Stopped"

fi

SERVICE='httpd'

if ps ax | grep -v grep | grep $SERVICE > /dev/null

then

echo "$SERVICE service running, everything is fine"

else

echo "$SERVICE is not running"

echo "$SERVICE is not running!" | mail -s "$SERVICE down" root

fi

**Check if program is running with bash shell script**

|  |  |  |
| --- | --- | --- |
| abstract  This is an example of a **bash script** which checks for some **running process**  (daemon or service) and does specific actions (reload, sends mail) if there is no such process running.   |  |  | | --- | --- | | compatible | http://www.anyexample.com/i/dar.gif |    Any modern **Linux** distribution   Nearly any unix sistem with recent bash version, grep, ps  We call *ps* (process list) command to list every system process and then use grep(filter)  command to check if our process name is in that list, for example:    $ ps ax | grep httpd  ps ax|grep httpd  592 ?? Ss 8:22.25 /usr/local/apache/bin/httpd  3706 ?? I 0:58.06 /usr/local/apache/bin/httpd  3707 ?? I 0:40.12 /usr/local/apache/bin/httpd  3708 ?? I 0:49.56 /usr/local/apache/bin/httpd  3709 ?? I 1:02.86 /usr/local/apache/bin/httpd  3710 ?? I 1:12.26 /usr/local/apache/bin/httpd  3711 ?? I 1:20.03 /usr/local/apache/bin/httpd  3712 ?? I 0:48.79 /usr/local/apache/bin/httpd  3714 ?? I 1:01.60 /usr/local/apache/bin/httpd  3715 ?? I 0:58.91 /usr/local/apache/bin/httpd  3716 ?? I 0:42.88 /usr/local/apache/bin/httpd  28363 p3 S+ 0:00.01 grep httpd |

But if there is no such process, grep (filter) command will match itself, because command argument has exactly same string which is searched for:

$ ps ax|grep nosuchprocess

28372 p3 S+ 0:00.01 grep nosuchprocess

So, we have to add additional 'grep -v grep' filter that will remove process which command line contains 'grep' i.e. grep itself.

Here is a script wich checks if SERVICE (variable at the beginning of script which contains service name) is running, and if it's not running, mails warning message to root

source code: bash script

#!/bin/sh

SERVICE='httpd'

if ps ax | grep -v grep | grep $SERVICE > /dev/null

then

echo "$SERVICE service running, everything is fine"

else

echo "$SERVICE is not running"

echo "$SERVICE is not running!" | mail -s "$SERVICE down" root

fi

|  |  |
| --- | --- |
| warning | http://www.anyexample.com/i/dar.gif |

 If you process name is too short and is substring of some other process command line, suggested approach could fail (for example, you checking for 'testd' process, but have running 'testdoo-doo-doo" which will result false positive)

 Though, highly unlikely, but if your process name will begin with 'grep', suggested approach could fail