Filename: watchdog_wifi.command
Use for MacBook Air 2013 that loose WiFi connection
This will detect automatically the router's IP and the WiFi network card and if the MBA cannot ping the router, it will power off the WiFi network card and restart it 3 seconds later.
This is a workaround of the problem of the MacBook Air 2013 and mostly not useful for any other Mac.

==========
watchdog_wifi.command - script start after this
==========

```
#!/bin/bash

###### parameters that you can adjust ######

# Auto-detection of Wi-Fi card ID
# (usually "en0" on macbook air and "en1" on other macbook)
network_id=$(networksetup -listnetworkserviceorder \
    | grep Hardware \
    | sed -e 's/^(H.*rt: \(.*\), Device: \(.*\))/\1 \2/' -e 's/[()\*#]//g' -e 's/[-]/_/g' \
    |grep Wi_Fi| awk '{print $2'})

# If you want to set manually the ID of the network card, use this command and remove #
#network_id=en0

# Auto-detection of the router"s IP (gateway)
router_ip=$(netstat -rn | grep "default" | grep "$network_id" | awk '{print $2}')

# If you want to set manually the IP to check, use this line and remove #
#router_ip=192.168.1.1

# delay between each check (in seconds)
delay_between_ping=5

# delay before switching back the network card on  (in seconds)
delay_switch_on=3
```

```
###### you should not modify anything else after ######

if [ "$router_ip" == "" ]; then
    echo
    echo "********************************************"
    echo " You need to connect to a WiFi network first! "
    echo "********************************************"
    echo
    echo press enter to finish...
    read nothing
    exit
fi

if [ "$network_id" == "" ]; then
    echo
    echo "*****************************************************************************"
    echo " Wi-Fi network card not detected, you have to manually configure the script "
    echo "*****************************************************************************"
    echo
    echo press enter to finish...
    read nothing
    exit
fi


echo
echo "*****************************************************************************"
echo " This script test if the WiFi network card $network_id is working. "
echo " It will ping the router's IP to check it."
echo " Router IP: $router_ip"
echo " If it cannot ping the router, the WiFi card will be turned Off then back On"
echo "*****************************************************************************"
echo
echo "------------------------"
echo " Press Control+C to quit "
echo "------------------------"
echo
```

```
while true
do

#ping the router
# -t 1 = wait 1 second max for the reply
# -c 1 = only ping once
# $router_ip = IP to ping
# grep "1 packets received" check if we get the right response
echo $(date)
echo Pinging the router to see if WiFi card $network_id is still alive...
ping -t 1 -c 1 $router_ip | grep "1 packets received"

#set the exit code in the variable
exitcode=$?
echo Exit Code: $exitcode

#if exit code is not 0, then we restart the WiFi network card
if [[ $exitcode != 0 ]] ; then

    echo $(date) >> watchdog_wifi.log

    echo
    echo "**************"
    echo Stopping WIFI
    echo "**************"
    /usr/sbin/networksetup -setairportpower $network_id off



    echo
    echo Waiting $delay_switch_on seconds...
    echo
    sleep $delay_switch_on

    echo "****************"
    echo Restarting WIFI
    echo "****************"
    /usr/sbin/networksetup -setairportpower $network_id on

    echo
```

```
    echo Waiting for network to reconnect...
    echo
    router_ip=$(netstat -rn | grep "default" | grep "$network_id" | awk '{print $2}')
    while [ "$router_ip" == "" ]; do
    sleep 1
        echo "..."
    router_ip=$(netstat -rn | grep "default" | grep "$network_id" | awk '{print $2}')
    done

fi

echo
echo Waiting $delay_between_ping seconds
echo
sleep $delay_between_ping

done
```