Integrate your plugin to LoxBerry's healthcheck

Compatibility

This feature is available from LoxBerry 2.2. LoxBerry below 2.2 ignores your healthcheck, but does not affect the functionality of your plugin.

LoxBerry's healthcheck can be run by the user interactively (located in *My LoxBerry / Healthcheck*), and additionally is automatically started daily by LoxBerry. If one or more healthchecks report an error, a notification is sent (also by email, if the user configured mail settings). If errors persist, the notification is repeated every week. A warning does not trigger a notification.

Additionally, the user can query the summary of the healthcheck with an LoxBerry Virtual HTTP Input, or can make use of the MQTT information if the user has the MQTT Gateway plugin installed.

You can provide an own plugin healthcheck with your plugin, that integrates into LoxBerry's healthcheck. All the functionality (notification, MQTT support, error reminder by email) of the internal checks are also available for your plugin healthcheck.

Implement your healthcheck

- Place a file named `healthcheck` (without file extension) to the `bin` directory of your plugin
- The `healthcheck` file needs the correct shebang on top of the script, to run the script with the correct interpreter
- Be sure to have UNIX line endings enabled

You can write your healthcheck code in any language, be sure that your shebang is correct.

Parameters and return values

LoxBerry's call of your healthcheck contains parameters, and expects specific return values. Your healthcheck requires to parse these options, and return the requested answer!

The return needs to be a line by line output to STDOUT, or a json output.

Input parameter	Return value (line-by-line output)	JSON output	Explanation
`title`	- line: Description of your healthcheck	`{ "desc" : "Description of your check" }`	With the `title` parameter, your healthcheck should return *only a description* without performing the check. The response should take place immediately. Do not use linefeeds (cr/lf) inside of the description.
`check`	- line: Description of your healthcheck - line: The **status code** of the check - line: Text result of your check	`{ "desc" : "Description of your check", "status" : statuscode, "result" : "The result of your check" }`	With the `check` parameter, you really should run the check. The output to STDOUT also needs to contain the 1. line with the description. The 2. line is the numeric status code (see below), the 3. line is a textual description of the result. Do not use linefeeds (cr/lf) in all of the text responses!

Status codes

Code	Status	Color	Description
0/nothing	UNKNOWN	GREY	This code shouldn't be returned by your healthcheck. It is set, if LoxBerry cannot call your healthcheck, or your healthcheck returns unexpected output.
3	ERROR	RED	If your healthcheck could figure out an error, set this status. The error will trigger a notification and email to the user. Only use the error state if it is really an error that requires user interaction.
4	WARNING	YELLOW	If you found something suspect, but cannot absolutely detect an error, use the warning flag.
5	OK	GREEN	Your healthcheck detected no problems.
6	INFO	BLUE	Use the Info state if your healthcheck does not check anything, but only shows information.

Test your healthcheck

Run your healthcheck script

For testing, do a `chmod +x healthcheck` so your script is executable. On the users LoxBerry's, the plugin installation will do this automatically.

First try to run your healthcheck script directly:

```
./healthcheck title
```

```
./healthcheck check
```

The scripts should only return the lines or json explained above.

Run your healthcheck within LoxBerry's healthcheck

Running `/opt/loxberry/sbin/healthcheck.pl` will always run all healthchecks available.

To speed up and only test your healtcheck, run

- `/opt/loxberry/sbin/healthcheck.pl action=titles check=plugincheck_<pluginfolder>`
- `/opt/loxberry/sbin/healthcheck.pl action=check check=plugincheck_<pluginfolder>`

where `<pluginfolder>` is the folder name of your plugin.

What to check

This is completely individual to your plugin. We have some ideas for you:

- Write a state file in your plugin (e.g. in /tmp/ - this is the ram disk). Check and/or read the state file in the healthcheck.
- If your plugin runs a daemon/service, check if it is still running (by name or pid).
- If you provide a TCP/UDP port, check if you can reach the port of your plugin.
- You can combine multiple checks in your healthcheck script. On the end of your script, return the status code of the highest severity.

The healthcheck should **not** fix a problem with your plugin. Otherwise, the user wouldn't take notice of the error, the error re-occurs, is re-fixed, re-occurs, re-fixed,... and you won't get any feedback of an error from the users.

Things to consider

- Your healthcheck shouldn't take too long. Users expect a runtime of 5 to 10 seconds maximum, otherwise it may be interpreted that the check has stalled/failed.
- The healthcheck itself should not influence your own or other plugins. Do not try to fix errors in the healthcheck (users won't take notice of errors otherwise).
- Be clear in your result text! If an error occurs, give some information for the user to fix the problem, and possibly for yourself, as the user may contact **you** if the check returns an error.
- Don't give a persistent warning or error because of an issue in your plugin code that the user cannot fix. Fix the reason for a persistent warning/error in your plugin!
- Your healthcheck script must write only the lines explained above. Prevent to write other output, especially if calling other commands inside of your healthcheck (e.g. in bash, use `> /dev/null` calling other commands)
- Description and result lingo: To keep it easy, all response should be kept in English. LoxBerry's internal healthchecks also only return English description and results.

Example healthcheck trunks

Bash

```
#!/bin/bash
echo "Checks if Alexa2Lox is working"  # First line: Description

if [ "$1" == "title" ]; then
    exit
fi

echo "6"  # Second line: Status
echo "Alexa2Lox is installed, but no further healthcheck installed."  #
Third line: Test result

exit
```

Perl

```
#!/usr/bin/perl

print "Checks if Alexa2Lox is working\n";  # First line: Description
if( $ARGV[0] eq "title" ) {
    exit;
}

print "6\n";  # Second line: Status
print "Alexa2Lox is installed, but no further healthcheck installed.\n";  #
Third line: Test result

exit;
```

PHP

```
#!/usr/bin/php

echo "Checks if Alexa2Lox is working\n";  // First line: Description
if( $argv[1] == "title" ) {
    exit();
}

echo "6\n";  // Second line: Status
echo "Alexa2Lox is installed, but no further healthcheck installed.\n";  //
Third line: Test result
```

```
exit();
```

Another way: Notification from within the plugin

The healthcheck is an external option to check your plugin. The check only runs once a day.

If your plugin itself can detect a critical error, and you want to inform the user immediately, you can use LoxBerry's Notify (Perl, PHP, Bash). Notify is also sent by email directly. Notifications are available since LoxBerry 1.0