Plugin für den Loxberry entwickeln

☐ HELP-Link in der plugin.cfg ergänzen (zugewiesen an Michael Schlenstedt)

Der LoxBerry kann über Plugins erweitert werden. Plugins bestehen aus einer Reihe von Dateien in einem Standard-ZIP-Archiv, die bei der Installation an die notwendigen Orte im Dateisystem des LoxBerry kopiert werden. Bei diesen Dateien kann es sich z. B. um PHP- oder CGI-Skriptdateien handeln. Des weiteren kann durch das Plugin Software aus dem Software-Repository der Linuxdistribution auf dem LoxBerry nachinstalliert werden.

Der folgende Artikel beschreibt den Aufbau eines Plugin-Archivs und beschreibt, wie ein Plugin für den LoxBerry entwickelt werden kann, sodass es sich in die Struktur des LoxBerry nahtlos einfügt. Um die einzelnen Schritte des Artikels nachvollziehen zu können solltest Du Dir die sogenannten "Sample Plugins" herunterladen, mit dem Du alle Schritte nachvollziehen kannst. Wir haben ein Sample Plugin, was die Entwicklung unter PERL beschreibt sowie ein Sample Plugin, was die Entwicklung unter PHP beschreibt. Natürlich kannst Du später die beiden Skriptsprachen auch mischen, aber für den Anfang ist es am einfachsten wenn Du die Sparache wählst, die Du am Besten beherrscht.

- PERL: https://github.com/mschlenstedt/LoxBerry-Plugin-SamplePlugin-V2-Perl
- PHP: https://github.com/christianTF/LoxBerry-Plugin-SamplePlugin-V2-PHP

Benutzung des Plugin Generators

Wenn du NPM hast, kannst du den Plugin Generator für Perl, PHP oder Node benutzen. Du wirst durch eine Reihe an Fragen geführt die du beantworten kannst, darauf hin wird die komplette Struktur mit den entsprechenden Dateien angelegt. Es sollte dir das initiale setup vereinfachen und du kannst direkt loslegen.

- Installation von NPM
- Dokumentation des Generators

```
npm init loxberry-plugin <ordner des Plugins>
```

Was ist ein LoxBerry-Plugin?

Ein LoxBerry-Plugin ist nichts anderes als eine Sammlung von Skripten (PHP, Perl, Python, usw.), HTML- oder auch Binary-Dateien, die unterhalb der Weboberfläche des LoxBerry installiert werden. Ziel bei der Entwicklung des LoxBerry war es, die vielen sehr guten Skript-Lösungen für den Raspberry Pi, die durch die Loxone Community mittlerweile entwickelt wurden (also z. B. die Google Kalenderanbindung, der Wunderground-Wetterserver, SONOS-Anbindung usw.) für den unerfahrenen Benutzer zugänglich zu machen. D. h. diese Skripte sollen in Form von Plugins über eine grafische Oberfläche installierbar sein - ohne Linux- oder Kommandozeilenkenntnisse.

Zusätzlich haben Plugins auch die Möglichkeit auf dem System fehlende Softwarepakete aus dem Standard Raspbian Repository nachzuinstallieren. Zudem kann ein Plugin auch einen Daemon beim Systemstart starten sowie Cronjobs installieren.

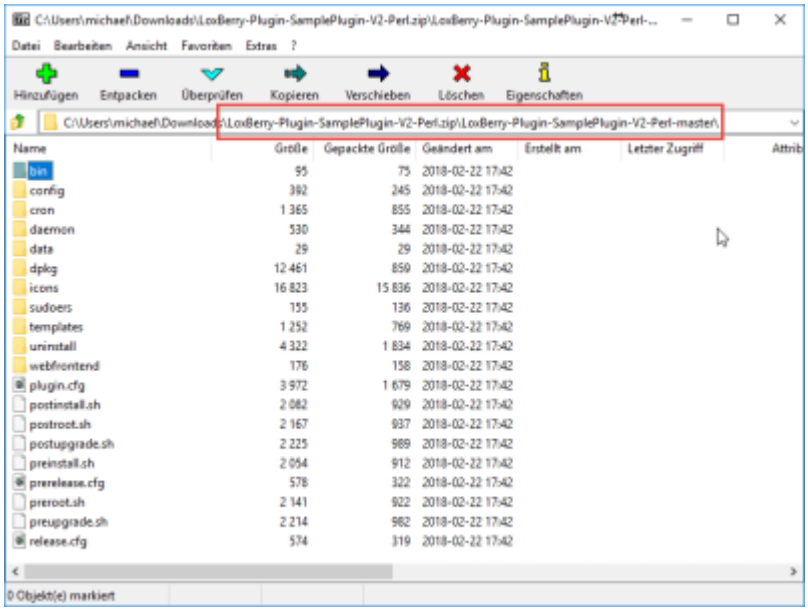
Die einzelnen Komponenten eines Plugins bzw. einer Skriptlösung werden über die Pluginschnittstelle an definierte Orte im Dateisystem des LoxBerry installiert, also HTML-Dateien ins HTML-Verzeichnis des Webservers, Perl-/PHP-Skripte ins entsprechende Verzeichnis des Webservers usw. Somit sind sie anschließend sofort einsatzbereit und auch der LoxBerry kann direkt auf diese Dateien zugreifen, z. B. um das Plugin in sein Hauptmenü zu integrieren.

Selbstverständlich sind auch Updates einzelner Plugins oder auch die Deinstallation möglich - alles ist über die Weboberfläche des LoxBerry durch den Benutzer durchführbar und konfigurierbar.

Wenn Du planst Deine Skriptlösung auch für den LoxBerry zur Verfügung zu stellen, dann schaue Dir am Besten einfach schon einmal existierende Plugins an und arbeite anschließend diese Anleitung anhand des sogenannten "Sample-Plugins" durch - das Sample Plugin behandelt alle Möglichkeiten der Pluginschnittstelle, sodass mit Hilfe dieses Beispiel-Plugins sehr gut nachvollziehbar ist, wie das Ganze funktioniert.

Aufbau der ZIP-Datei

Alle Dateien des Plugins können im Root-Verzeichnis des ZIP-Archivs oder in einem Unterverzeichnis mit beliebigen Namen liegen. Obwohl es mit Leerzeichen oder Sonderzeichen keine Probleme geben sollte, verwende bitte für das Unterverzeichnis nur ASCII-Zeichen ohne Leerzeichen. Es ist nicht möglich mehr als ein Untervezeichnis zu verwenden. Die ZIP-Datei darf nicht verschlüsselt oder Passwort-geschützt sein. Wenn Du den Sourcecode Deines Plugins auf GitHub verwaltest, kannst Du direkt aus GitHub heraus eine ZIP-Datei Deines Plugins herunterladen oder gleich die Release-Funktion von GitHub verwenden, um Dein plugin zu veröffentlichen. Die ZIP-Dateien von GitHub sind kompatibel mit LoxBerry.



Die Dateien und Verzeichnisse des ZIP-Archivs werden bei der Installation der Reihe nach abgearbeitet. Der Ablauf der Installation ist in der folgenden Auflistung dargestellt. Details zu jedem Schritt bzw. jeder Datei beschreiben die folgenden Kapitel.

1. Unzippen der Datei

2. Auslesen und Interpretation der Datei "plugin.cfg"
3. Ausführung der Datei "preroot.sh" mit Rootrechten
4. Nur bei Update: Ausführen der Datei "preupgrade.sh" mit Userrechten (loxberry)
5. Nur bei Update: Löschen der alten Installation
6. Ausführen der Datei "preinstall.sh" mit Userrechten (loxberry)
7. Installation der Dateien aus "config" nach LBHOMEDIR/config/plugins/
8. Installation der Dateien aus "bin" nach LBHOMEDIR/bin/plugins/
9. Installation der Dateien aus "template" nach LBHOMEDIR/template/plugins/
10. Installation der Dateien aus "cron" nach LBHOMEDIR/system/cron/
11. Installation der Dateien aus "data" nach LBHOMEDIR/data/plugins/
12. Installation der Dateien aus "webfrontend/htmlauth" nach LBHOMEDIR/webfrontend/htmlauth/plugins/
13. Installation der Dateien aus "webfrontend/html" nach LBHOMEDIR/webfrontend/html/plugins/
14. Installation der Dateien aus "icons" nach LBHOMEDIR/webfrontend/html/system/images/icons/
15. Installation der Datei "daemon/daemon" nach LBHOMEDIR/system/daemons/plugins
16. Installation der Datei "uninstall/uninstall" nach LBHOMEDIR/data/system/uninstall
17. Installation der Datei "sudoers/sudoers" nach LBHOMEDIR/system/sudoers
18. Installation zusätzlicher Pakete aus dem Distributions-Repository mit apt-get aus der Datei dpkg/apt
19. Installation zusätzlicher mitgelieferter DEB-Pakete aus dem Verzeichnis dpkg/<arch>
20. Ausführen der Datei "postinstall.sh" mit Userrechten (loxberry)
21. Nur bei Update: Ausführen der Datei "postupgrade.sh" mit Userrechten (loxberry)
22. Ausführung der Datei "postroot.sh" mit Rootrechten

Rootverzeichnis - Datei: plugin.cfg PFLICHT

Diese Datei ist die zentrale Konfigurationsdatei für das Plugin. Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Der Dateiaufbau ähnelt dem einer typischen INI-Datei unter Windows: Die Daten sind in Blöcke organisiert, jeder Block wird durch den Blocknamen in eckigen Klammern eingeleitet, jede Option innerhalb eines Blockes wird durch KEY=VALUE gesetzt. Kommentare werden durch eine vorangestellte Raute eingeleitet und bei der Interpretation der Datei ignoriert.

1. Block: AUTHOR

```
[AUTHOR]
# You can also use a Team/Project Name here and a generic email address
# like info@..., BUT NEVER CHANGE this information in future updates! It
# will be used to identify your Plugin, handle updates etc. If you change
# this information, LoxBerry could not identify your plugin and handle it as
# a different one - therefore updates will fail.
NAME=Max Mustermann
EMAIL=info@yourdomain.de
```

	Key	Value
1	NAME	Legt den Namen des Autors oder der Projektgruppe fest.
2	EMAIL	Email-Adresse des Autors oder der Projektgruppe

Wichtiger Hinweis!

Die beiden Variablen NAME und EMAIL werden benutzt um das Plugin im System nach der Installation eindeutig zu identifizieren. Daher dürfen diese beiden Angaben bei späteren Updates des Plugins nicht geändert werden. Ansonsten würde das Plugin nicht als Update sondern als neues, eigenständiges Plugin vom System behandelt werden und somit parallel zum bereits installierten Plugin installiert werden.

2. Block: PLUGIN

```
[PLUGIN]
# The version of your plugin - important if you would like to write an
# upgrade script. Use a correct syntax, which is supported by LoxBerry:
# More info: http://www.loxwiki.eu/x/LYG3AQ
VERSION=1.0.0

# Short name and prefered subfolder of your Plugin (do not use blanks - they
# will be filtered - and use lowercase only)! Used for script names in
# daemon or cron (NAME) and as unique installation folder. If these names
# already exist on the installation system, we will add 01, 02, 03 and so
# on. Therefore your script should check THESE TWO VARIABLES for figuring
# out subfolder and scriptnames! You will find this information in
# /data/system/plugindatabase.dat after installation!  BUT NEVER CHANGE this
# information in future updates! It will be used to identify your Plugin,
# handle updates etc. If you change this information, Loxberry could not
# identify your plugin and will handle it as a different one - therefore
# updates will definetely fail.
NAME=sampleplugin_name
FOLDER=sampleplugin_folder

# Friendly Long Name of your Plugin - 25 Characters maximum! All others
maybe
# replaced by "...". You can use blanks, uppercase/lowercase etc.
TITLE=Sample Plugin
```

	Key	Value
1	VERSION	Version des Plugins. Kann verwendet werden um spezielle Aktionen bei einem Update auszuführen (siehe unten). Wird zudem in der Plugin-Verwatung des LoxBerry angezeigt. Die Version muss eine korrekte Versionssyntax verwenden, andernfalls wird das Plugin bei automatischen Plugin-Updates ausgelassen!
2	NAME	Kurzer Plugin-Name. Keine Leerzeichen, keine Sonderzeichen, nur Kleinbuchstaben! Dieser Name wird als Dateiname verwendet, wenn z. B. in zentralen Ordnern (Cron, Daemon, siehe unten) Skripte für das Plugin angelegt werden sollen. Dieser Name muss im System einzigartig sein. Wenn festgestellt wird, dass bereits ein Plugin installiert ist, welches den gleichen Namen verwendet, wird bei der Installation automatisch "01", "02" usw. angehängt. *Wir empfehlen, NAME und FOLDER gleich zu setzen.*

3	FOLDER	Verzeichnis-Name. Keine Leerzeichen, keine Sonderzeichen, nur Kleinbuchstaben! Dieser Name wird als Name der Unterverzeichnisse erwendet, in die die einzelnen Dateien des Plugins installiert werden. Dieser Name muss im System einzigartig sein. Wenn festgestellt wird, dass bereits ein Plugin installiert ist, welches den gleichen Namen verwendet, wird bei der Installation automatisch "01", "02" usw. angehängt. *Wir empfehlen, NAME und FOLDER gleich zu setzen.*
4	TITLE	(Langer) Name des Plugins. Dieser wird in der Plugin-Verwaltung des LoxBerry angezeigt und wird auch in den LoxBerry-Menüs verwendet. Hier können Sonderzeichen, Leerzeichen etc. verwendet werden. Der Name sollte nicht länger als 25 Zeichen sein.

Wichtiger Hinweis!

Die beiden Variablen NAME und FOLDER werden benutzt um das Plugin im System nach der Installation eindeutig zu identifizieren. Daher dürfen diese beiden Angaben bei späteren Updates des Plugins nicht geändert werden. Ansonsten würde das Plugin nicht als Update sondern als neues, eigenständiges Plugin vom System behandelt werden und somit parallel zum bereits installierten Plugin installiert werden.

3. Block: SYSTEM

```
[SYSTEM]
# If a reboot is needed after the plugin was installed, enable the following
# option:
REBOOT=true

# If your Plugin runs only on special LoxBerry Versions, please set the
# following options. If you plugin is compatible with all versions or at
least
# with all future versions, please set this to false or leave it empty..
# Note! If you use the new Plugin Interface V2  - I think you will, because
you
# currently read THIS file ;-) - please set the Minimum version to 0.3.0!
LB_MINIMUM=0.3.1
LB_MAXIMUM=false

# If your plugin runs only on a special architecture (e.g. if you use the
GPIOs
# on a Raspberry platform), please set the architecture here. You can
seperate
# different architectures with a comma. Set to false or leave it empty if
your
# plugin do not use any special features of an architecture.
# PUT in QUOTES ""!!!
ARCHITECTURE="aarch64,x86_64"

# If you are using the LoxBerry::Log Modul in PHP or Perl and you would like
to use User-defined loglevels,
# enable this option. If enabled, the user will be able to choose a loglevel
in the Plugin Management Widget.
```

```
CUSTOM_LOGLEVELS=true

# Plugin Interface
# Do not change this if you are not knowing what you are doing!
INTERFACE=2.0
```

	Key	Value	
1	REBOOT	Wenn true, wird nach der Plugin-Installation zum Reboot aufgefordert. Mögliche Werte: true	false
2	LB_MINIMUM	Minimalversion des LoxBerry, ab der das Plugin lauffähig ist, z. B. 1.0.1. Mögliche Werte: Versionstring	false
3	LB_MAXIMUM	Maximalversion des LoxBerry, bis zu der das Plugin lauffähig ist, z. B. 1.0.1. Mögliche Werte: Versionstring	false
4	ARCHITECTURE	Falls das Plugin nur auf bestimmten Hardware-Plattformen lauffähig ist (z. B. weil es die GPIOs des Raspberry voraussetzt), kann das hier angegeben werden. Mehrere Plattformen können durch Kommata getrennt werden. Aktuell unterstützt: armv6l, armv7l, aarch64, x86_64, riscv64. Es wird das Kürzel erwartet, welches der Befehl "uname -m" zurückliefert. Siehe auch: https://github.com/MichaIng/DietPi/blob/master/dietpi/func/dietpi-obtain_hw_model	
5	CUSTOM_LOGLEVELS	Wenn true, wird im Plugin Management Widget dem User eine Auswahl an verschiedenen Loglevels angezeigt. Kann in Verbindung mit den LoxBerry Libraries LoxBerry::Log verwendet werden, siehe Perl-Modul LoxBerry::Log bzw. PHP Module loxberry_log.php. Es kann auch als Ersatz einer eigenen Loglevel-Einstellung im Plugin verwendet werden, siehe LoxBerry::System::pluginloglevel bzw. LBSystem::pluginloglevel. Mögliche Werte: true	false
6	INTERFACE	Die Interface-Version, für die das Plugin entwickelt wurde. Wenn Du die Features aus dieser anleitung verwenden möchtest, muss hier 2.0 eingetragen werden.	

4. Block: AUTOUPDATE

```
[AUTOUPDATE]
# If your plugin offers automatic updates, please enable the following
option.
# Details here: http://www.loxwiki.eu/x/WoG3AQ
AUTOMATIC_UPDATES=true

# This is the URL to your release.cfg file. This file will be checked for
# new releases if the user eneables autoupdates. If the version number
# given in the release.cfg file is newer than the installed one, the
# plugin archive will be downloaded from the URL given in the release.cfg
file
# and will be installed automatically
RELEASECFG=https://raw.githubusercontent.com/mschlenstedt/LoxBerry-Plugin-Sa
mplePlugin-V2-Perl/master/release.cfg

# This is the URL to your prerelease.cfg file. This file will be checked for
# new prereleases if the user enables autoupdates for prereleases. If the
# version number given in the release.cfg file is newer than the installed
# one, the plugin archive will be downloaded from the URL given in the
# release.cfg file and will be installed automatically
PRERELEASECFG=https://raw.githubusercontent.com/mschlenstedt/LoxBerry-Plugin
-SamplePlugin-V2-Perl/master/prerelease.cfg
```

	Key	Value	
1	AUTOMATIC_UPDATES	Legt fest, ob das Plugin die AutoUpdate-Funktion zur Verfügung stellt oder nicht. Details hier: AutoUpdate Funktion für Plugins Mögliche Werte: true	false
2	RELEASECFG	Eine URL, unter der die Datei mit den Einstellungen zum Release vom System heruntergeladen werden kann. Details hier: AutoUpdate Funktion für Plugins	
3	PRERELEASECFG	Eine URL, unter der die Datei mit den Einstellungen zum PreRelease vom System heruntergeladen werden kann. Details hier: AutoUpdate Funktion für Plugins	

Rootverzeichnis - Datei: preroot.sh OPTIONAL

Bei der Datei *preroot.sh* handelt es sich um ein Shell-Skript, welches vor der Installation ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). Die Datei wird mit Root-Rechten ausgeführt, hat also umfangreiche Rechte im System. Sie sollte nur verwendet werden, wenn unbedingt Kommandos als User root ausgeführt werden müssen. Ansonsten sollte man das Skript "preinstall.sh" verwenden (siehe unten). Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird vor der Installation als Benutzer "root" ausgeführt. Es kann verwendet werden um Dinge für die Installation vorzubereiten oder Kommandos auf dem System auszuführen.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	$1
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	$2
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	$3
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	$4
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	$5
6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv (sicherere Alternative zu $1)	$6

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBHOMEDIR`	Homeverzeichnis von LoxBerry	`/opt/loxberry`

LBPHTMLAUTH	Plugin-HTML-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/htmlauth/plugins
LBPHTML	Plugin-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/plugins
LBPTEMPL	Plugin-Template-Verzeichnis	/opt/loxberry/templates/plugins
LBPDATA	Plugin-Data-Verzeichnis	/opt/loxberry/data/plugins
LBPLOG	Plugin-Log-Verzeichnis	/opt/loxberry/log/plugins
LBPCONFIG	Plugin-Config-Verzeichnis	/opt/loxberry/config/plugins
LBPBIN	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin/plugins

System-Pfade

Variable	Bedeutung	LoxBerry-Original
LBSHTMLAUTH	System-HTMLAUTH-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/cgi/system
LBSHTML	System-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/system
LBSTEMPL	System-Template-Verzeichnis	/opt/loxberry/templates/system
LBSDATA	System-Data-Verzeichnis	/opt/loxberry/data/system
LBSLOG	System-Log-Verzeichnis	/opt/loxberry/log/system
LBSTMPFSLOG	System-Log-Verzeichnis (tmpfs)	/opt/loxberry/log/system_tmpfs
LBSCONFIG	System-Config-Verzeichnis	/opt/loxberry/config/system
LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei preroot.sh

```
#!/bin/bash

# Shell script which is executed *BEFORE* installation is started
# (*BEFORE* preinstall and *BEFORE* preupdate). Use with caution and
remember,
# that all systems may be different!
#
# Exit code must be 0 if executed successfull.
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
# Will be executed as user "root".
# !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
$1)

# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR
```

```
echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

exit 0
```

Rootverzeichnis - Datei: preupgrade.sh OPTIONAL

Bei der Datei *preupgrade.sh* handelt es sich um ein Shell-Skript, welches vor der Installation und nur im Falle eines Plugin-Updates ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). D. h. das Plugin muss schon auf dem System existieren, damit dieses Skript bei der Installation ausgeführt wird. Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird vor einem Update als Benutzer "loxberry" ausgeführt. Es kann verwendet werden, um im Falle eines Updates Konfigurationsdateien des Plugins zu sichern.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	$1
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	$2
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	$3
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	$4
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	$5
6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv (sicherere Alternative zu $1)	$6

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
LBHOMEDIR	Homeverzeichnis von LoxBerry	/opt/loxberry
LBPHTMLAUTH	Plugin-HTML-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/htmlauth/plugins
LBPHTML	Plugin-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/plugins
LBPTEMPL	Plugin-Template-Verzeichnis	/opt/loxberry/templates/plugins
LBPDATA	Plugin-Data-Verzeichnis	/opt/loxberry/data/plugins
LBPLOG	Plugin-Log-Verzeichnis	/opt/loxberry/log/plugins
LBPCONFIG	Plugin-Config-Verzeichnis	/opt/loxberry/config/plugins
LBPBIN	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin/plugins

System-Pfade

Variable	Bedeutung	LoxBerry-Original
LBSHTMLAUTH	System-HTMLAUTH-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/cgi/system
LBSHTML	System-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/system
LBSTEMPL	System-Template-Verzeichnis	/opt/loxberry/templates/system
LBSDATA	System-Data-Verzeichnis	/opt/loxberry/data/system
LBSLOG	System-Log-Verzeichnis	/opt/loxberry/log/system
LBSTMPFSLOG	System-Log-Verzeichnis (tmpfs)	/opt/loxberry/log/system_tmpfs
LBSCONFIG	System-Config-Verzeichnis	/opt/loxberry/config/system
LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei preupgrade.sh

```
#!/bin/bash

# Shell script which is executed in case of an update (if this plugin is
already
# installed on the system). This script is executed as very first step
(*BEFORE*
# preinstall.sh) and can be used e.g. to save existing configfiles to /tmp
# during installation. Use with caution and remember, that all systems may
be
# different!
#
# Exit code must be 0 if executed successfull.
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# Will be executed as user "loxberry".
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
$1)
```

```
# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR

echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

exit 0
```

Rootverzeichnis - Datei: preinstall.sh OPTIONAL

Bei der Datei *preinstall.sh* handelt es sich um ein Shell-Skript, welches vor der Installation ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird vor der Installation als Benutzer "loxberry" ausgeführt. Es kann verwendet werden um Dinge für die Installation vorzubereiten oder Kommandos auf dem System auszuführen.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	$1
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	$2
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	$3
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	$4
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	$5

6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv (sicherere Alternative zu $1)	$6

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
LBHOMEDIR	Homeverzeichnis von LoxBerry	/opt/loxberry
LBPHTMLAUTH	Plugin-HTML-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/htmlauth/plugins
LBPHTML	Plugin-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/plugins
LBPTEMPL	Plugin-Template-Verzeichnis	/opt/loxberry/templates/plugins
LBPDATA	Plugin-Data-Verzeichnis	/opt/loxberry/data/plugins
LBPLOG	Plugin-Log-Verzeichnis	/opt/loxberry/log/plugins
LBPCONFIG	Plugin-Config-Verzeichnis	/opt/loxberry/config/plugins
LBPBIN	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin/plugins

System-Pfade

Variable	Bedeutung	LoxBerry-Original
LBSHTMLAUTH	System-HTMLAUTH-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/cgi/system
LBSHTML	System-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/system
LBSTEMPL	System-Template-Verzeichnis	/opt/loxberry/templates/system
LBSDATA	System-Data-Verzeichnis	/opt/loxberry/data/system
LBSLOG	System-Log-Verzeichnis	/opt/loxberry/log/system
LBSTMPFSLOG	System-Log-Verzeichnis (tmpfs)	/opt/loxberry/log/system_tmpfs
LBSCONFIG	System-Config-Verzeichnis	/opt/loxberry/config/system
LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei preinstall.sh

```
#!/bin/bash

# Shell script which is executed by bash *BEFORE* installation is started
(but
# *AFTER* preupdate). Use with caution and remember, that all systems may be
# different!
#
# Exit code must be 0 if executed successfull.
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# Will be executed as user "loxberry".
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
```

```
$1)

# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR

echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

exit 0
```

Rootverzeichnis - Datei: postinstall.sh OPTIONAL

Bei der Datei *postinstall.sh* handelt es sich um ein Shell-Skript, welches nach der Installation ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird nach der Installation als Benutzer "loxberry" ausgeführt. Es kann verwendet werden um Dinge nach der Installation aufzubereiten oder Kommandos auf dem System auszuführen.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	$1
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	$2

3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	$3
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	$4
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	$5
6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv (sicherere Alternative zu $1)	$6

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBHOMEDIR`	Homeverzeichnis von LoxBerry	`/opt/loxberry`
`LBPHTMLAUTH`	Plugin-HTML-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/htmlauth/plugins`
`LBPHTML`	Plugin-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/plugins`
`LBPTEMPL`	Plugin-Template-Verzeichnis	`/opt/loxberry/templates/plugins`
`LBPDATA`	Plugin-Data-Verzeichnis	`/opt/loxberry/data/plugins`
`LBPLOG`	Plugin-Log-Verzeichnis	`/opt/loxberry/log/plugins`
`LBPCONFIG`	Plugin-Config-Verzeichnis	`/opt/loxberry/config/plugins`
`LBPBIN`	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/bin/plugins`

System-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBSHTMLAUTH`	System-HTMLAUTH-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/cgi/system`
`LBSHTML`	System-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/system`
`LBSTEMPL`	System-Template-Verzeichnis	`/opt/loxberry/templates/system`
`LBSDATA`	System-Data-Verzeichnis	`/opt/loxberry/data/system`
`LBSLOG`	System-Log-Verzeichnis	`/opt/loxberry/log/system`
`LBSTMPFSLOG`	System-Log-Verzeichnis (tmpfs)	`/opt/loxberry/log/system_tmpfs`
`LBSCONFIG`	System-Config-Verzeichnis	`/opt/loxberry/config/system`
`LBSBIN`	System-Bin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/sbin`
`LBSSBIN`	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/bin`

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation

geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei postinstall.sh

```
#!/bin/bash

# Shell script which is executed by bash *AFTER* complete installation is
done
# (but *BEFORE* postupdate). Use with caution and remember, that all systems
may
# be different!
#
# Exit code must be 0 if executed successfull.
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# Will be executed as user "loxberry".
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
```

```
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
$1)

# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR

echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

# Exit with Status 0
exit 0
```

Rootverzeichnis - Datei: postupgrade.sh OPTIONAL

Bei der Datei *postupgrade.sh* handelt es sich um ein Shell-Skript, welches nach der Installation und nur im Falle eines Plugin-Updates ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). D. h. das Plugin muss schon auf dem System existieren, damit dieses Skript nach der Installation ausgeführt wird. Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird nach einem Update als Benutzer "loxberry" ausgeführt. Es kann verwendet werden, um im Falle eines Updates Konfigurationsdateien des Plugins nach der Installation wieder zurückzuspielen.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	`$1`
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	`$2`
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	`$3`
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	`$4`
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	`$5`
6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv (sicherere Alternative zu `$1`)	`$6`

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBHOMEDIR`	Homeverzeichnis von LoxBerry	`/opt/loxberry`
`LBPHTMLAUTH`	Plugin-HTML-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/htmlauth/plugins`
`LBPHTML`	Plugin-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/plugins`
`LBPTEMPL`	Plugin-Template-Verzeichnis	`/opt/loxberry/templates/plugins`
`LBPDATA`	Plugin-Data-Verzeichnis	`/opt/loxberry/data/plugins`
`LBPLOG`	Plugin-Log-Verzeichnis	`/opt/loxberry/log/plugins`
`LBPCONFIG`	Plugin-Config-Verzeichnis	`/opt/loxberry/config/plugins`
`LBPBIN`	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/bin/plugins`

System-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBSHTMLAUTH`	System-HTMLAUTH-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/cgi/system`
`LBSHTML`	System-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/system`
`LBSTEMPL`	System-Template-Verzeichnis	`/opt/loxberry/templates/system`
`LBSDATA`	System-Data-Verzeichnis	`/opt/loxberry/data/system`
`LBSLOG`	System-Log-Verzeichnis	`/opt/loxberry/log/system`
`LBSTMPFSLOG`	System-Log-Verzeichnis (tmpfs)	`/opt/loxberry/log/system_tmpfs`
`LBSCONFIG`	System-Config-Verzeichnis	`/opt/loxberry/config/system`

LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei preupgrade.sh

```
#!/bin/bash

# Shell script which is executed in case of an update (if this plugin is
already
# installed on the system). This script is executed as very first step
(*BEFORE*
# preinstall.sh) and can be used e.g. to save existing configfiles to /tmp
# during installation. Use with caution and remember, that all systems may
be
# different!
#
# Exit code must be 0 if executed successfull.
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# Will be executed as user "loxberry".
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
```

```
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
$1)

# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR

echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

exit 0
```

Rootverzeichnis - Datei: postroot.sh OPTIONAL

Bei der Datei *postroot.sh* handelt es sich um ein Shell-Skript, welches nach der Installation ausgeführt wird. Ab LoxBerry 3.0 sind auch andere Dateiendungen erlaubt (zum Beispiel .pl oder .py - in diesem Fall muss der Shebang im Skript korrekt gesetzt sein). Die Datei wird mit Root-Rechten ausgeführt, hat also umfangreiche Rechte im System. Sie sollte nur verwendet werden, wenn unbedingt Kommandos als User root ausgeführt werden müssen. Ansonsten sollte man das Skript "postinstall.sh" verwenden (siehe oben). Es handelt sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF). Das Skript wird nach der Installation als Benutzer "root" ausgeführt. Es kann verwendet werden um Dinge nach der Installation aufzubereiten oder Kommandos auf dem System auszuführen.

Dem Skript werden auf der Kommandozeile folgende Parameter übergeben, die innerhalb des Bashskripts verwendet werden können:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner, in den das Plugin-Archiv zur Installation entpackt wurde (`/tmp/uploads/$1`)	`$1`
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	`$2`
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	`$3`
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	`$4`
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	`$5`
6	Voller Pfad zum temporären Ordner, in den das Plugin-Archiv zur Installation entpackt wurde - inklusive eventuellem Unterordner aus dem ZIP-Archiv	`$6`

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBHOMEDIR`	Homeverzeichnis von LoxBerry	`/opt/loxberry`
`LBPHTMLAUTH`	Plugin-HTML-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/htmlauth/plugins`
`LBPHTML`	Plugin-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/plugins`
`LBPTEMPL`	Plugin-Template-Verzeichnis	`/opt/loxberry/templates/plugins`
`LBPDATA`	Plugin-Data-Verzeichnis	`/opt/loxberry/data/plugins`
`LBPLOG`	Plugin-Log-Verzeichnis	`/opt/loxberry/log/plugins`
`LBPCONFIG`	Plugin-Config-Verzeichnis	`/opt/loxberry/config/plugins`
`LBPBIN`	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/bin/plugins`

System-Pfade

Variable	Bedeutung	LoxBerry-Original
LBSHTMLAUTH	System-HTMLAUTH-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/cgi/system
LBSHTML	System-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/system
LBSTEMPL	System-Template-Verzeichnis	/opt/loxberry/templates/system
LBSDATA	System-Data-Verzeichnis	/opt/loxberry/data/system
LBSLOG	System-Log-Verzeichnis	/opt/loxberry/log/system
LBSTMPFSLOG	System-Log-Verzeichnis (tmpfs)	/opt/loxberry/log/system_tmpfs
LBSCONFIG	System-Config-Verzeichnis	/opt/loxberry/config/system
LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Die Auflistung von REPLACExxx Tags zum automatischen Ersetzen der Verzeichnisse während der Installation findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

Alle Ausgaben auf STDOUT und STDERR des Skriptes werden in die Logdatei der Plugininstallation geschrieben. So kann man Statusmeldungen mittels "echo" absetzen. Um die Meldungen im Logfile farblich hervorzuheben kann und sollte man die folgenden einleitenden Tags am Zeilenanfang des Logeintrags verwenden:

Logfile Tags	Hervorhebung
<OK>	Grün
<INFO>	Schwarz/Neutral
<WARNING>	Rot
<ERROR>	Rot
<FAIL>	Rot

Das Skript muss mit einem Exit-Status von "0" (Null) enden, damit es als erfolgreich beendet wird. Bei einem Exit-Statis von 1 wird ein Fehler ausgegeben, aber die Installation wird fortgesetzt. Bei einem Exit-Status > 1 wird die weitere Installation des Plugins abgebrochen. Benötigt man das Skript nicht, so ist es ratsam die Datei zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen, um die Installation nicht unnötig zu verzögern.

Beispiel: Datei postroot.sh

```
#!/bin/bash

# Shell script which is executed by bash *AFTER* complete installation is
done
# (*AFTER* postinstall and *AFTER* postupdate). Use with caution and
remember,
# that all systems may be different!
#
# Exit code must be 0 if executed successfull.
```

```
# Exit code 1 gives a warning but continues installation.
# Exit code 2 cancels installation.
#
# !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
# Will be executed as user "root".
# !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#
# You can use all vars from /etc/environment in this script.
#
# We add 5 additional arguments when executing this script:
# command <TEMPFOLDER> <NAME> <FOLDER> <VERSION> <BASEFOLDER>
#
# For logging, print to STDOUT. You can use the following tags for showing
# different colorized information during plugin installation:
#
# <OK> This was ok!"
# <INFO> This is just for your information."
# <WARNING> This is a warning!"
# <ERROR> This is an error!"
# <FAIL> This is a fail!"

# To use important variables from command line use the following code:
COMMAND=$0    # Zero argument is shell command
PTEMPDIR=$1   # First argument is temp folder during install
PSHNAME=$2    # Second argument is Plugin-Name for scipts etc.
PDIR=$3       # Third argument is Plugin installation folder
PVERSION=$4   # Forth argument is Plugin version
#LBHOMEDIR=$5 # Comes from /etc/environment now. Fifth argument is
              # Base folder of LoxBerry
PTEMPPATH=$6  # Sixth argument is full temp path during install (see also
$1)

# Combine them with /etc/environment
PHTMLAUTH=$LBPHTMLAUTH/$PDIR
PHTML=$LBPHTML/$PDIR
PTEMPL=$LBPTEMPL/$PDIR
PDATA=$LBPDATA/$PDIR
PLOG=$LBPLOG/$PDIR # Note! This is stored on a Ramdisk now!
PCONFIG=$LBPCONFIG/$PDIR
PSBIN=$LBPSBIN/$PDIR
PBIN=$LBPBIN/$PDIR

echo -n "<INFO> Current working folder is: "
pwd
echo "<INFO> Command is: $COMMAND"
echo "<INFO> Temporary folder is: $PTEMPDIR"
echo "<INFO> (Short) Name is: $PSHNAME"
echo "<INFO> Installation folder is: $PDIR"
echo "<INFO> Plugin version is: $PVERSION"
echo "<INFO> Plugin CGI folder is: $PHTMLAUTH"
echo "<INFO> Plugin HTML folder is: $PHTML"
```

```
echo "<INFO> Plugin Template folder is: $PTEMPL"
echo "<INFO> Plugin Data folder is: $PDATA"
echo "<INFO> Plugin Log folder (on RAMDISK!) is: $PLOG"
echo "<INFO> Plugin CONFIG folder is: $PCONFIG"

# Exit with Status 0
exit 0
```

Unterverzeichnis: config OPTIONAL

Alle Dateien aus diesem Verzeichnis werden im System unter */opt/loxberry/config/plugins/NAME/* installiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg). In diesem Verzeichnis sollten Konfigurstionsdateien abgelegt werden, die vom Plugin verwendet werden. Benötigt das Plugin keine Konfigurationsdateien, so kann dieses Verzeichnis auch leer sein.

Unterverzeichnis: bin OPTIONAL

Alle Dateien aus diesem Verzeichnis werden im System unter */opt/loxberry/bin/plugins/NAME/* installiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg). In diesem Verzeichnis sollten ausführbare Skripte und Binaries abgelegt werden, die vom Plugin verwendet werden und nicht über das Webfrontend erreichbar sein sollen. Man kann hier z. B. Skripte ablegen, die aus einem Cronjob heraus aufgerufen werden. Benötigt das Plugin keine entsprechenden Skripte, so kann dieses Verzeichnis auch leer sein.

Unterverzeichnis: templates OPTIONAL

Dein Plugin sollte Template-Dateien verwenden, um seine Konfiguration im LoxBerry-Menü darzustellen. Bitte stelle wenn möglich mindestens Templates in englischer Sprache zur Verfügung. Das ist die Default-Sprache ab LoxBerry V1.0. Die Dateien aus diesem Unterverzeichnis werden bei der Installation nach */opt/logberry/templates/plugins/NAME* kopiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg).

Wie Templates für Dein Plugin am effektivsten erstellt werden beschreiben die folgenden Wiki-Artikel:

Perl: /wiki/spaces/LOXBERRY/pages/1193708779

PHP: PHP functions to create your webpage with LoxBerry design

Unterverzeichnis: cron OPTIONAL

Die Dateien aus diesem Verzeichnis werden im System als Cronjob installiert. Es muss sich dabei jeweils um ein Shell-Skript oder um ein Skript mit korrektem Shebang handeln. Es muss sich um ASCII-Dateien im Unix-Dateiformat (Zeilenende: Linefeed, LF) handeln, die in der Shell ausgeführt werden können. Der Name der Datei entscheidet, für welchen Cronjob die Datei installiert wird. Die Cronjobs werden als Benutzer "loxberry" ausgeführt. Bei der Installation wird das Skript umbenannt in den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg) und im korrekten Unterverzeichnis unter */opt/loxberry/system/cron* installiert.

Des weiteren kann eine crontab-Datei im Verzeichnis hinterlegt werden, um volle Flexibilität bei der Generierung von Cronjobs zu erhalten. Bei dieser Datei handelt es sich um eine vollwertige crontab-Datei, die im System nach der Installation nach `$lbhomedir/system/cron/cron.d/<pluginname>` kopiert wird. Es existieren zahlreiche HOWTOs, wie diese Dateien aufgebaut sind. Die crontab-Manpage findet man hier: http://man7.org/linux/man-pages/man5/crontab.5.html

Wichtig ist, dass die Syntax analog der Datei */etc/crontab* bzw. */etc/cron.d* erfolgen muss, d. h. es muss der ausführende User mit angegeben werden. Hier ist ausschließlich der User "loxberry" erlaubt. Der folgende Wiki-Artikel erklärt, wie die Datei verwendet werden muss: Eigene Cronjobs im Plugin-Code pflegen

Folgende Dateien bzw. Cronjobs stehen zur Verfügung:

Cronjob	Dateiname
Vollwertige Crontab	`crontab`
Minütlich	`cron.01min`
Alle 3 Minuten	`cron.03min`
Alle 5 Minuten	`cron.05min`
Alle 10 Minuten	`cron.10min`
Alle 15 Minuten	`cron.15min`
Alle 30 Minuten	`cron.30min`
Stündlich	`cron.hourly`
Täglich	`cron.daily`
Wöchentlich	`cron.weekly`
Monatlich	`cron.monthly`
Jährlich	`cron.yearly`
Bei Reboot *(Achtung, ab LoxBerry **2.0**)*	`cron.reboot`

Cronjobs, die unmittelbar nach Reboot ausgeführt werden, können sich nicht darauf verlassen, dass Netzwerk, alle LoxBerry-Plugin-Verzeichnisse und -Variablen bereits existieren. Entweder mit sleep den Start eigener Scripts verzögern, oder im Script prüfen, ob Verzeichnisse bereits angelegt sind.

Ab **LoxBerry 2.2.1** ist sichergestellt, dass Netzwerk und Plugin-Verzeichnisse beim Aufruf des Cronjobs (auch nach Reboot) verfügbar sind.

Unterverzeichnis: data OPTIONAL

Hier können sämtliche Dateien, die das Plugin zusätzlich benötigt, abgelegt werden. Denkbar sind zum Beispiel ASCII oder auch Binary-Dateien. Auch zur Zwischenspeicherung kann dieses Verzeichnis später vom Plugin verwendet werden (Cache-Dateien oder ähnliches). Bei der Installation werden alle Dateien aus diesem Verzeichnis nach */opt/loxberry/data/plugins/NAME* installiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg).

Unterverzeichnis: webfrontend/htmlauth PFLICHT

Dieses Verzeichnis enthält Script-Dateien (Perl oder PHP), die per Webserver ausgeführt werden sollen (vergleichbar mit */usr/lib/cgi-bin*), statische HTML-Dateien sowie sonstige Dateien, die per Webserver ausgeliefert werden sollen (etwa Bilddateien). Dieses Verzeichnis ist Passwort-geschützt (LoxBerry Webuser). Das Hauptmenü Deines LoxBerrys sowie das Plugin Management Widget verweist auf dieses Verzeichnis. Daher sollte bzw. muss es die Hauptkonfigurationsseite Deines Plugins enthalten. **Es muss mindestens die Datei *index.cgi, index.php* oder *index:start* exisiteren.**

Die Dateien werden bei der Installation ins Verzeichnis */opt/loxberry/webfrontend/htmlauth/plugins/NAME* kopiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg).

Unterverzeichnis: webfrontend/html OPTIONAL

Dieses Verzeichnis enthält Script-Dateien (Perl oder PHP), die per Webserver ausgeführt werden sollen (vergleichbar mit */usr/lib/cgi-bin*), statische HTML-Dateien sowie sonstige Dateien, die per Webserver ausgeliefert werden sollen (etwa Bilddateien). Dieses Verzeichnis ist **nicht** Passwort-geschützt, ist also von jedem aus den lokalen Netzwerk aufrufbar.

Die Dateien werden bei der Installation ins Verzeichnis */opt/loxberry/webfrontend/htmlauth/plugins/NAME* kopiert. NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg).

Unterverzeichnis: icons PFLICHT

Dieses Verzeichnis enthält die Icons im PNG-Format, die im Menü des LoxBerrys verwendet werden.

Es müssen Icon-Dateien in folgenden Größen hinterlegt werden: 64x64, 128x128, 256x256 und 512x512. Die Dateien müssen im PNG-Format mit transparentem Hintergrund vorliegen. Die Dateien müssen folgende Namen haben: "icon_64.png", "icon_128.png" usw.

Bitte verwendet Icons im Stil der anderen LoxBerry-Icons, um ein einheitliches Aussehen des Menüs zu erhalten. Passende Icons können hier kostenlos heruntergeladen werden:

- Vollständige Icon-Library: http://download.loxberry.de/development/icons/
- Leere Vorlage für eigene Gestaltung: Blank_icon.zip

Werden keine Icons durch das Plugin zur Verfügung gestellt, verwendet der LoxBerry Default-Icons.

Unterverzeichnis: daemon OPTIONAL

Hier kann eine Datei mit Namen "daemon" abgelegt werden. Es muss sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF) handeln, die in der Shell ausgeführt werden kann. Es kann sich dabei um ein beliebiges Script (Bash, Perl, PHP usw.) handeln. Die erste Zeile muss den entsprechenden Shebang enthalten.

Die Datei wird während des Bootvorgangs des LoxBerrys als Init-Skript ausgeführt. Bei der Installation wird die Datei umbenannt in den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg) und im Verzeichnis */opt/loxberry/system/daemons/plugins* installiert.

Die Besonderheit: Da dieses Skript während des Bootvorgangs als Init-Skript ausgeführt wird, wird sie mit Root-Rechten ausgeführt.

Es stehen sämtliche LoxBerry Environment-Variablen zur Verfügung: Systemweite Pfade in Environmentvariablen

Systemweite Pfade in Environmentvariablen

Plugin-Pfade

Variable	Bedeutung	LoxBerry-Original
`LBHOMEDIR`	Homeverzeichnis von LoxBerry	`/opt/loxberry`
`LBPHTMLAUTH`	Plugin-HTML-Verzeichnis (Authentifizierung)	`/opt/loxberry/webfrontend/htmlauth/plugins`
`LBPHTML`	Plugin-HTML-Verzeichnis (ohne Auth)	`/opt/loxberry/webfrontend/html/plugins`
`LBPTEMPL`	Plugin-Template-Verzeichnis	`/opt/loxberry/templates/plugins`
`LBPDATA`	Plugin-Data-Verzeichnis	`/opt/loxberry/data/plugins`
`LBPLOG`	Plugin-Log-Verzeichnis	`/opt/loxberry/log/plugins`
`LBPCONFIG`	Plugin-Config-Verzeichnis	`/opt/loxberry/config/plugins`
`LBPBIN`	Plugin-Bin-Verzeichnis **(ab LB 1.2.2)**	`/opt/loxberry/bin/plugins`

System-Pfade

Variable	Bedeutung	LoxBerry-Original
LBSHTMLAUTH	System-HTMLAUTH-Verzeichnis (Authentifizierung)	/opt/loxberry/webfrontend/cgi/system
LBSHTML	System-HTML-Verzeichnis (ohne Auth)	/opt/loxberry/webfrontend/html/system
LBSTEMPL	System-Template-Verzeichnis	/opt/loxberry/templates/system
LBSDATA	System-Data-Verzeichnis	/opt/loxberry/data/system
LBSLOG	System-Log-Verzeichnis	/opt/loxberry/log/system
LBSTMPFSLOG	System-Log-Verzeichnis (tmpfs)	/opt/loxberry/log/system_tmpfs
LBSCONFIG	System-Config-Verzeichnis	/opt/loxberry/config/system
LBSBIN	System-Bin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/sbin
LBSSBIN	System-Sbin-Verzeichnis **(ab LB 1.2.2)**	/opt/loxberry/bin

Außerdem können die Perl-Funktionen (use LoxBerry::System) oder PHP-Funktionen (require_once "loxberry_system.php") und deren Variablen bzw. Konstanten verwendet werden.

Ab LoxBerry 2.0: Eventuell reicht es für deinen Anwendungsfall aus, einen Cronjob in `cron.reboot` abzulegen. Dieser wird bei jedem Start im Kontext des `loxberry`-Benutzers gestartet. Siehe Cron.

Unterverzeichnis: uninstall OPTIONAL

Hier kann eine Datei mit Namen "**uninstall**" abgelegt werden. Es muss sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF) handeln, die in der Shell ausgeführt werden kann. Diese Datei wird am Ende einer Deinstallation (nicht bei einem Update!) als User **root** ausgeführt. Sie kann benutzt werden, um Dinge, die durch das Plugin verändert worden sind, wieder rückgängig zu machen. Bitte achte darauf, dass diese Änderungen gegebenenfalls auch von anderen Plugins verwendet werden könnten! Bei der Installation wird die Datei umbenannt in den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg) und im Verzeichnis */opt/loxberry/data/system/uninstall* gespeichert.

Ab LoxBerry 2.0: Dem uninstallscript werden folgende Commandline-Parameter übergeben:

Nr.	Inhalt	Bash-Variable
1	Temporärer Ordner (`/tmp`)	$1
2	Tatsächlich verwendeter Plugin-Name (siehe Kapitel zur Datei //plugin.cfg//)	$2
3	Tatsächlich verwendeter Plugin-Installationsordner (siehe Kapitel zur Datei //plugin.cfg//)	$3
4	Plugin-Version (siehe Kapitel zur Datei //plugin.cfg//)	$4
5	LoxBerry Basis-Ordner (normalerweise `/opt/loxberry`)	$5

Ab LoxBerry 2.0: Bei der Ausführung des uninstall-Scripts sind außer der Cronjob-Dateien noch alle Daten vorhanden. Es können (und sollten) eigene Daemons, Scripts, Services etc. beendet bzw. gekillt

werden.

Uninstall Ablauf (**ab LoxBerry 2.0**):

1. Löschen der Cronjobs
2. Löschen des individuellen crontab-Files
3. **Ausführen des Uninstall-Scripts (als root)**
4. Löschen des Uninstall-Scripts
5. Löschen des Daemons
6. Löschen des Sudoers-Files
7. Löschen der Plugin-Ordner (config, bin, data, templates, webfrontend, temporäre Backups, Plugin-Icons)
8. Entfernen des Plugins aus der Plugin-Datenbank
9. Löschen der Logfiles
10. Löschen der Notifications aus der Datenbank

LoxBerry 1.x: Dem Script werden keine Variablen übergeben. Zum Zeitpunkt des Uninstall-Scriptlaufs sind bereits alle Plugin-Daten gelöscht.

Unterverzeichnis: sudoers OPTIONAL

Hier kann eine Datei mit Namen "sudoers" abgelegt werden. Es muss sich um eine ASCII-Datei im Unix-Dateiformat (Zeilenende: Linefeed, LF) handeln. Mit der sudoers-Datei können System-Kommandos oder eigene Scripte festgelegt werden, die vom Plugin mit Rootrechten ausgeführt werden. Da es sich hierbei um ein Sicherheitsrisiko handelt, solltest Du diese Funktion nur im äußersten notfall verwenden und ausschließlich, wenn unbedingt Rootrechte erforderlich sind. Meist reicht es auch aus, den User "loxberry" in einer entsprechenden Gruppe einzutragen, um gewisse Befehle im System ausführen zu können.

Die Syntax des Sudoers-File beschreibt die Manpage: https://www.sudo.ws/man/1.8.15/sudoers.man.html Eine recht gute Dokumentation findet man auch hier: https://help.ubuntu.com/community/Sudoers Ein weiteres beispiel findest Du im LoxBerry-Sourcecode, der ebenfalls ein System-Sudoers-File verwendet: https://github.com/mschlenstedt/Loxberry/blob/master/system/sudoers/lbdefaults

Bei der Installation wird die Datei umbenannt in den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg) und im Verzeichnis */opt/loxberry/system/sudoers* gespeichert.

Unterverzeichnis: dpkg OPTIONAL

Über dieses Unterverzeichnis können vom Plugin fehlende Softwarepakete nachinstalliert werden. Hierbei kann sowohl auf Standardpakete aus dem Raspbian Repository zurückgegriffen werden (dringend empfohlen!) oder (im Notfall) auch direkt DEB-Pakete mitgeliefert werden.

Software aus dem Standard Raspbian Repository:

Hierzu muss im Verzeichnis die Datei "`apt`" existieren. Es handelt sich um eine ASCII-Datei im Unix-

Dateiformat (Zeilenende: Linefeed, LF). Kommentare werden durch eine vorangestellte Raute eingeleitet und bei der Interpretation der Datei ignoriert. Jedes zu installierende Paket muss in einer separaten Zeile eingetragen werden.

Es ist auch möglich, abhängig von der **LoxBerry** Major-Versionsnummer unterschiedliche Pakete zu installieren. Dazu wird die Major Versionsnummer an den Dateinamen der apt-Datei angehängt, also z. B. "`apt3`" für LoxBerry 3.x, "`apt2`" für LoxBerry 2.x usw. *Dieser Weg ist mittlerweile obsolete - siehe nächsten Abschnitt.*

Zusätzlich ist es auch möglich, abhängig von der **Debian** Major-Versionsnummer unterschiedliche Pakete zu installieren. Diese Änderung war notwendig, weil nicht mehr zwingend anhand der LoxBerry Major-Versionsnummer auf die darunter liegende Debian-Versionsnummer geschlossen werden kann (seit DietPi kann z. B. LoxBerry 3.0 auf unterschiedlichen Distributionsversionen laufen). Dazu wird die Debian Major Versionsnummer an den Dateinamen der apt-Datei angehängt, also z. B. "`apt11`" für Debian 11 (Bullseye), "`apt12`" für Debian 12 (Bookworm) usw.

Vor Installation der Pakete wird der Befehl `apt-get -q -y update` sowie der Befehl `dpkg –configure -a` ausgeführt, um die Paketquellen zu aktualisieren. Anschließend werden alle Pakete gemeinsam an den Befehl `apt-get -q -y install` weitergeleitet und somit die Pakete installiert.

Wenn vom Plugin keine zusätzlichen Pakete benötigt werden, ist es ratsam die Datei `apt` zu löschen und nicht im Pluginarchiv zur Verfügung zu stellen. So wird die Installation des Plugins deutlich beschleunigt, da bei Existenz der Datei auf jeden Fall der Befehl zum Aktualisieren der Paketquellen ausgeführt wird (egal, ob anschließend ein Paket installiert werden soll oder nicht). Die Aktualisierung der Paketquellen dauert je nach Internetverbindung mehrere Sekunden bis Minuten, sodass es zu einer deutlichen Verzögerung während der Installation des Plugins kommt.

Beispiel: Datei apt

```
# These packages will be installed by apt-get -y install PACKAGENAME
# One line per package, use exact packagename as you would do for apt-get.
#
package1
package2
package3
```

Software aus selbst mitgelieferten DEB-Paketen:

Möchte man direkt DEB-Pakete mitliefern, z. B. weil die benötigte Software im Standard-Repository nicht zur Verfügung steht, muss für jede Hardware-Architektur ein entsprechendes Unterverzeichnis angelegt werden, in dem dann die DEB-Pakete liegen. Es werden dabei Verzeichnisse mit dem Architektur-Namen erwartet, die auch der Befehl uname -m zurückliefert. Mit DietPi wurden die unterstützten Architekturen erweitert (ab LoxBerry 3.0). Aktuell (Stand: 07.03.2023) sind das:

Architektur	**Unterverzeichnis**	**Beispiel**
Armv6l	`armv6l`	Raspberry 1 oder Zero
Armv7l	`armv7l`	Raspberry 2 Rev. 1
Aarch64 bzw. Arm64	`aarch64`	Raspberry 2-4 und Zero2
x86_64	x86_64	Virtual Box VM (u. a.)

Architektur	Unterverzeichnis	Beispiel
Riscv64	`riscv64`	

In die Unterverzeichnisse werden dann die DEB-Pakete direkt abgelegt. Diese werden auf der jeweiligen Hardware-Plattform in alphabetischer Reihenfolge per `dpkg -i` installiert.

Besonderheiten / Hinweise

Log-Dateien

Seit LoxBerry V1.0 werden Logdateien von Plugins in einer RAM-Disk gespeichert. Daher können Sie auch nicht mehr mit dem Plugin mitgeliefert werden (wie noch bei der Pluginschnittstelle V1 des LoxBerry 0.2.3). Bei jedem Booten werden die Logdateien gelöscht. Daher muss die Existenz einer Logdatei von den Pluginskripten unbedingt kontrolliert werden und die Logdatei bei Bedarf erzeugt werden. Jedes Plugin hat ein eigenes Verzeichnis für Logdateien: */opt/loxberry/log/plugins/NAME.* NAME wird dabei ersetzt durch den Pluginnamen aus plugin.cfg (vergleiche Kapitel plugin.cfg).

Der Verzeichnisname steht natürlich auch als Environment-Variable zur Verfügung: Systemweite Pfade in Environmentvariablen. Außerdem können die Perl-Funktionen (use LoxBerry::System) oder PHP-Funktionen (require_once "loxberry_system.php") verwendet werden, um in Skripten den genauen Verzeichnispfad zu ermitteln.

Zur Anzeige von Logdateien bringt der LoxBerry einen Logviewer mit, der für solche Zwecke verwendet werden sollte: http://IP_LOXBERRY/admin/system/tools/logfile.cgi

Automatisches Ersetzen der Plugin-Verzeichnisse

Die Plugin-Installation ersetzt in ALLEN mitgelieferten Text-Dateien (z.B. **auch** in den preinstall/postinstall-Dateien, in HTMLs, PHPs, Perl-Scripts) folgende Ausdrücke durch die entsprechenden Verzeichnisse bzw. Namen:

Die Auflistung der REPLACExxx Tags findest du hier: Automatisches Ersetzen der Plugin-Verzeichnisse (REPLACE)

/opt/loxberry hardcoded

Wird in einem Plugin hardcoded der Pfad `/opt/loxberry` verwendet, wird dies erkannt und als Warnung bei der Benutzerinstallation ausgegeben. Diese Überprüfung kann dabei nicht unterscheiden, ob der Code tatsächlich verwendet wird oder auskommentiert ist. Um die Warnung zu vermeiden, bitte stattdessen die Ersetzungskonstante REPLACELBHOMEDIR, oder die globale Environmentvariable $LBHOMEDIR verwenden.

Innerhalb von Perl- und PHP-Scripten können die Variablen der Module verwendet werden.

Miniserver

Bei der Einrichtung des LoxBerrys können mehrere Miniserver eingerichtet werden. Dein Plugin sollte das berücksichtigen. Die Daten der Miniserver kann mit den Modulfunktionen LoxBerry::System::get_miniservers (Perl) bzw. LBSystem::get_miniservers (PHP) abgefragt werden.

Die Module unterstützen direkt die Verwendung von Loxone Cloud DNS, um Miniserver in anderen Netzen zu verwenden.

Versenden von Emails

Soll dein Plugin E-Mails versenden, so kannst du diese einfach über *sendmail* verschicken: */usr/sbin/sendmail.* Voraussetzung ist, dass der User einen Mailserver für ausgehende E-Mails eingerichtet hat.

WhatsApp Gruppe

Wenn du so weit gelesen hast, können wir dich auch gerne einladen, der LB-Plugins WhatsApp-Gruppe beizutreten, wo sich die Plugin-Entwickler austauschen, und auch vom Core-Entwicklungsteam unterstützt werden.

Bitte um Kontaktaufnahme im LoxForum, mit Name, WhatsApp-Telefonnummer und kurzer Beschreibung, an welchem Plugin du arbeitest: https://www.loxforum.com/member/409-christian-fenzl