Grundlagen zur Erstellung eines Plugins

```php
$a = array(1, 2, 3, 17);

foreach ($a as $v) {
    echo "Aktueller Wert von \$a: $v.\n";
}
```

Wenn du die Funktionsweise dieses Codes verstehst, kannst du ein LoxBerry Plugin entwickeln!

Das Entwickeln von LoxBerry Plugins ist - wer schon mal irgendwas gescriptet oder programmiert hat - nicht all zu schwer. Praktischerweise gibt es heute das Internet, wo man Beispiele findet oder direkt übernehmen kann.

Einige LoxBerry-Plugins sind beispielsweise aus bereits bestehenden, eigenen Scripts, oder Programmen aus dem Internet hervorgegangen, bei denen primär ein Webinterface für die Konfiguration dieser Scripts erforderlich war.

Wenn du erstmals ein Plugin planst, zerlege die Entwicklung in folgende Teile:

- Das Programm, das die eigentliche Funktion deines Plugins ausführt.
- Eine Konfigurationsdatei, aus der dein Programm die Parameter ausliest, die es benötigt.

Damit hast du schon mal die wahre Funktionalität. Jetzt fehlt noch das Rundherum, deswegen sind die nächsten Schritte:

- Ein Webinterface, um die Konfigurationsdatei zu lesen und zu schreiben
- ggf. ein Cronjob, um dein Programm in Intervallen auszuführen

Schritt 0: Update deinen LoxBerry auf den aktuellsten Pre-Release

Damit hast du alle Bugfixes und neuen Funktion auf deinem Raspberry.

Schritt 1: Plugin-Verzeichnis aufbauen

Beginne damit, dass du dir das Sample-Plugin herunterlädst (nicht installierst!) und auf deinem PC entpackst. Darin ist Perl- oder PHP-Code, für die Implementierung kann aber jede beliebige Sprache verwendet werden. Perl und PHP hat den Vorteil, dass LoxBerry dafür ein SDK anbietet, das viele Aufgaben erleichtert (z.B. die Erzeugung des Webinterfaces, oder das Auslesen der LoxBerry-Konfiguration, und das Erstellen von Logfiles).

Schritt 2: plugin.cfg anpassen

In deiner Kopie des Sample-Plugins passe den Namen des Plugins, Author, Versionsnummer usw. an

Mit der Änderung dieser Daten ist es ab sofort **dein** Plugin.

Schritt 3: Dein Plugin installieren

Zippe das Verzeichnis, und installiere es am LoxBerry. Ignoriere eventuelle Installationsfehler - die stammen aus den Beispieldaten des Sample-Plugins und kannst du später korrigieren.

Danach findest du dein Plugin in der Pluginverwaltung.

Entsprechend hat LoxBerry Verzeichnisse entsprechend des Namens deines Plugins am Raspberry angelegt.

Verzeichnis	**Funktion**
/opt/loxberry/webfrontend/htmlauth/plugins/*deinplugin*/	Verzeichnis für dein Webinterface (Authentifizierung erforderlich)
/opt/loxberry/webfrontend/html/plugins/*deinplugin*/	Verzeichnis für Webseiten ohne Authentifizierung
/opt/loxberry/templates/plugins/*deinplugin*/	Speziell bei der Verwendung mit Perl und HTML::Template legst du hier HTML-Templates ab. Im Unterverzeichnis lang sind die Sprachdateien.
/opt/loxberry/config/plugins/*deinplugin*/	Verzeichnis für deine Konfigurationsdateien
/opt/loxberry/bin/plugins/*deinplugin*/	Verzeichnis für ausführbare Dateien, die nicht aus dem Web erreichbar sein sollen
/opt/loxberry/log/plugins/*deinplugin*/	Verzeichnis für deine Logdateien
/opt/loxberry/data/plugins/*deinplugin*/	Verzeichnis für jegliche andere Daten, die dein Plugin benötigt oder erzeugt

Schritt 4: Funktion implementieren

Entscheide, ob die Funktion deines Plugins zeitgesteuert, oder per Webaufruf getriggert werden soll.

- Zeitgesteuert: Entwickle dein Programm in deinem bin-Verzeichnis
- Webaufruf: Entwickle dein Programm in deinem webfrontend/htmlauth-Verzeichnis

Programmiere deine Funktionalität, und verwende für alles, was später vom Benutzer im Web anpassbar sein soll, ein Config-File, welches du in dein config-Verzeichnis legst.

Bei Perl und PHP binde die LoxBerry-Library LoxBerry::System bzw. loxberry_system.php ein - verwende dessen globale Variablen für deine Verzeichnisse! Bei anderen Programmiersprachen musst du dir selbst Routinen schreiben, die dein Plugin-Verzeichnis herausfinden. Der Name deines Plugin-Verzeichnisses ist variabel, deswegen darfst du das nicht hardcoden.

Schau dir das SDK für deine Sprache durch - Wenn du beispielsweise die Miniserver-Konfiguration vom LoxBerry benötigst, gibt es die SDK-Funktion get_miniservers. Zum Erstellen von Logfiles gibt es LogBerry::Log bzw. loxberry_log.php.

Schritt 5: Webinterface implementieren

Wenn deine Funktionen laufen, entwickle dafür dein Webinterface.

Verwende dafür die Perl-Lib LoxBerry::Web bzw. für PHP loxberry_web.php. **Schau dir die Funktionen des SDK's durch** und verwende Code des Sample-Plugins weiter.

Verwende readlanguage (Perl, PHP) um alternative Sprachen für dein Plugin anzubieten.

Schritt 6: Installation anpassen

Wenn am installierten LoxBerry deine Sachen halbwegs funktionieren, aktualisiere dein Plugin-Paket:

- Kopiere die neuen Programmdateien in das Verzeichnis, wovon du anfangs das ZIP mit der geänderten plugin.cfg erstellt hast.
- Wenn während der Installation, oder während des Plugin-Updates spezielle Schritte nötig sind, aktualisiere in diesem Verzeichnis die Install-Scripts (z.B. Sichern der Config während des Updates)
- Wenn spezielle Debian-Pakete für dein Plugin nötig sind, aktualisiere im Ordner apt das apt-File und füge dort die Paketnamen ein.

Schritt 7: Testen und weitermachen

Bevor zu deinen Quellordner neu zippst und installierst, stelle sicher, dass wirklich alle Dateien vom LoxBerry in deiner Quelle am PC sind - während des Updates werden alle Ordner am LoxBerry gelöscht, um diese bei der Installation wieder anzulegen.

Danach kannst du die Installation testen. Meist kommt man dabei auf etwaige Fehler in den Installationsscripten drauf. Du kannst danach alles so durchtesten, wie es sich dann auch bei anderen Benutzern verhält.

Schritt 8: Hilfe holen

Manche Sachen sind beim ersten Mal ungewöhnlich, deswegen nicht verzweifeln sondern Rat einholen.

Alle LoxBerry-Plugin-Entwickler schauen regelmäßig im LoxForum in den Entwickler-Bereich für LoxBerry: https://www.loxforum.com/forum/projektforen/loxberry/entwickler

Außerdem betreiben wir eine WhatsApp-Gruppe für LoxBerry-Entwickler, wo das Troubleshooting viel schneller geht. Bitte dafür mit Christian FenzlKontakt aufnehmen (inkl. Name und Telefonnummer).

Schritt 9: Öffentlicher Test

Schlussendlich würden wir vorerst deine Alpha/Beta-Version im LoxForum anbieten zum Testen (mit entsprechendem Disclaimer, dass es noch Alpha-Stand ist). Wenn die gröberen Probleme ausgeräumt

sind, solltest du im LoxWiki dein neues Plugin anlegen und dokumentieren. Machs so, wie es die anderen Plugin-Autoren machen.

Empfehlungen

Codeverwaltung mit GitHub

Mit der Windows-Software GitHub Desktop kannst du dein Plugin-Quellverzeichnis als öffentliches Repository bei Github hosten. Vorteile sind: Du hast eine echte Code-Verwaltung mit Änderungsverfolgung und der Möglichkeit, Änderungen rückgängig zu machen. Außerdem können dich andere Entwickler bei Problemen oder Fehlern leichter unterstützen, wenn sie direkt Einblick in deinen Code bekommen.

Die in LoxBerry eingebaute Update-Funktion für Plugins ist für GitHub optimiert.