Interface

T2S Interface

Zusätzlich stellt das Plugin eine bi-direktionale Schnittstelle für Drittanwendungen bzw. Plugins zur Verfügung. Mit Hilfe dieser Schnittstelle können T2S Requests per http-POST Befehl oder JSON request geschickt werden, welche dann vom T2S Plugin in MP3 Dateien umgewandelt und gespeichert werden. Die notwendigen Infos zum weiteren Prozessieren in Drittanwendung/Plugin werden dann in einem JSON Response zur Verfügung gestellt.

T2S Request

Der Request um eine T2S zu generieren muss über http-POST im JSON Format an folgende URL gesendet werden. Wie man diesen Request in der jeweiligen Programmiersprache umsetzt muss der Entwickler eroieren, das würde a: hier den Rahmen sprengen und b: ich kenne nicht alle Sprachen:

URL to be called

```
http://<LOXBERRY IP>/plugins/text2speech/index.php
```

Um den Text, der in eine MP3 umgewandelt werden soll, mitzuschicken muss in der URL-Syntax der Drittanwendung/Plugin der Text deklariert sein z.B. über einen Parameter ***&text=Dies ist ein test***

Zusätzlich liefert das Plugin per Zufallsgenerator 5 verschiedene Grußtexte je nach Tageszeit die ebenfalls mit generiert werden können (näheres dazu siehe oben). Um diese Grußtexte vor der eigentlichen MP3 miteinzubinden muss zusätzlich der Parameter ***&greet*** angegeben werden. Wenn ***&greet*** mitangeben wurde muss dieser zur Vorbereitung des Requests entweder mit einer 0 (Null) oder einer 1 (Eins) gefüllt werden. **&greet=1**.

Beispiel Syntax aus dem Sonos Plugin heraus. Der Text "Dies ist ein Test" mit einer vorangestellten Grußformel soll in eine MP3 gewandelt und anschließend auf dem Player "kueche" angesagt werden:

http://<LOXBERRY-IP/plugins/sonos4lox/index.php/?zone=kueche&action=ttsp&t2stext=Dies ist ein test$greeting

Der Text "Dies ist ein Test" ohne einer vorangestellten Grußformel soll in eine MP3 gewandelt und anschließend auf dem Player "kueche" angesagt werden:

http://<LOXBERRY-IP/plugins/sonos4lox/index.php/?zone=kueche&action=ttsp&t2stext=Dies ist ein test

generiere Parameter aus URL

```
$text = ($_GET['t2stext']);

// prüfe ob &greet angegeben wurde
if (isset($_GET['greet']))  {
    // &greet wurde in der Syntax mitangegeben
    $greet = 1;
} else {
    // &greet wurde in der Syntax nicht mitangegeben
    $greet = 0;
}
```

Die verwendeten Parameter sind grundsätzlich frei wählbar, müssen dann nur für den POST request in die entsprechenden Variablen geschrieben werden.

Generiere JSON Daten

Zur Vorbereitung des Requests müssen die gelesenen Parameter aus der Syntax in eine Array übertragen und anschließend JSON encoded werden. Dies erfolgt wie im obigen Abschnitt beschrieben über das Auslesen der Syntax und mit anschließender Übergabe an das zu erstellende Array, welches dann in das JSON Format konvertiert werden muss. Hier ein PHP code Beispiel:

generiere JSON Daten

```
// erstelle ein Array (named keys) mit den ausgelesenen Parametern.
// Die Keys 'text' bzw. 'greet' müssen verwendet werden da sonst das
Interface den Request fehlinterpretiert und keine T2S generiert wird
$jsonData = array('text' => $text,'greet' => $greet);

// Wandele das erstellte Array in das JSON Format.
$jsonDataEncoded = json_encode($jsonData);
```

Das war es im Grunde schon zur Vorbereitung des T2S Requests. Mit Hilfe der URL und der JSON Daten muss jetzt der Request an das Text2speech Plugin zur Generierung der MP3 Files gesendet werden. Um eine erfolgreiche Übertragung zu gewährleisten muss dem http-request im content header noch das entsprechende Datenformat mitgegeben werden. Dieses lautet '**Content-Type: application/json**'. und der komplette Request muss noch als Typ **POST** deklariert werden

Kompletter Request

Hier ein komplettes Bespiel eine T2S Requests aus PHP (Sonos Plugin) heraus welches zusätzlich die cURL Library nutzt:

kompletter T2S Request

```
// API Url

// full example w/o greeting:
//
http://<LOXBERRY-IP>/plugins/sonos4lox/index.php/?zone=schlafen&action=ttsp&
text=Dies%20ist%20ein%20test
// full example include random greeting:
//
http://<LOXBERRY-IP>/plugins/sonos4lox/index.php/?zone=schlafen&action=ttsp&
text=Dies%20ist%20ein%20test&greet

$url = 'http://<LOXBERRY-IP>/plugins/text2speech/index.php';

// Get text from syntax
$text = ($_GET['text']);

// Get greeting if requested
if (isset($_GET['greet']))  {
    $greet = 1;
} else {
    $greet = 0;
}

// Initiate cURL.
$ch = curl_init($url);

// populate JSON data.
$jsonData = array(
    'text' => $text,
    'greet' => $greet
);

// Encode the array into JSON.
$jsonDataEncoded = json_encode($jsonData);

// Tell cURL that we want to send a POST request.
curl_setopt($ch, CURLOPT_POST, 1);

// Attach our encoded JSON string to the POST fields.
curl_setopt($ch, CURLOPT_POSTFIELDS, $jsonDataEncoded);

// Set the content type to application/json
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:
application/json'));

// Execute the request
$result = curl_exec($ch);

// Optional
// was the request successful?
if($result === false)  {
```

```
    echo "ERROR: The POST wasn't successfull. Please check your code!";
} else {
    echo "The POST was successfull";
}
// close cURL
curl_close($ch);
```

Validierung des Requests

Das Text2speech Plugin validiert ob der empfangene Request korrekt empfangen wurde und loggt den jeweiligen Status. Hier der entsprechende Code, der aber für den Entwickler nicht maßgeblich ist:

T2S Request validierung

```
// Make sure that it is a POST request.
if(strcasecmp($_SERVER['REQUEST_METHOD'], 'POST') != 0){
    LOGERR("T2S Interface ** Request method must be POST!");
    exit;
}

// Make sure that the content type of the POST request has been set to
application/json
$contentType = isset($_SERVER["CONTENT_TYPE"]) ?
trim($_SERVER["CONTENT_TYPE"]) : '';
if(strcasecmp($contentType, 'application/json') != 0){
    LOGERR("T2S Interface ** Content type must be: application/json");
    exit;
}

// Receive the RAW post data.
$content = trim(file_get_contents("php://input"));

// Attempt to decode the incoming RAW post data from JSON.
$decoded = json_decode($content, true);

// If json_decode failed, the JSON is invalid.
if(!is_array($decoded)){
    LOGERR("T2S Interface ** Received content contained invalid JSON!");
    exit;
}
LOGOK("T2S Interface ** POST request has been successful processed!");
return ($decoded);
```

T2S Answer

Das Text2speech Plugin nimmt den Request welcher per Interface reinkommt, wandelt den Text, ggf. mit vorangestellter Grußformel, in eine MP3 um und speichert diese. Den Speicherort der MP3 kann sich der User im Plugin selber aussuchen, es werden aber nur read AND write Verzeichnisse unterstützt. Parallel wird der Symlink "**interfacedownload**" dem neuem Speicherpfad angepasst, so dass der Interface Pfad statisch bleibt. Im Falle der User hat ein USB device als Speicherort gewählt, wird das MP3 file in den Standard plugindata folder (/opt/loxberry/data/plugins/text2speech/) verschoben und ist somit vom Interface nutzbar.

Der genaue Speicherort der MP3, und alle anderen notwendigen Details, werden in Form einer JSON Datei auf dem LoxBerry abgelegt UND parallel per **echo** Befehl ausgegeben. Mit Hilfe dieser beiden Informationen kann dann in der Drittanwendung/Plugin die Verarbeitung fortgesetzt werden.

Antwortdetails

```
Array
(
    [fullttspath] =>
/opt/loxberry/data/plugins/text2speech/tts/1d91695724e14c372529572496cc4048.
mp3
    [path] => /opt/loxberry/data/plugins/text2speech/tts/
    [fullcifsinterface] =>
\\192.168.50.xx\plugindata\text2speech\interfacedownload\1d91695724e14c37252
9572496cc4048.mp3
    [cifsinterface] =>
\\192.168.50.xx\plugindata\text2speech\interfacedownload\
    [fullhttpinterface] =>
http://192.168.50.xx:80/plugins/text2speech/interfacedownload/1d91695724e14c
372529572496cc4048.mp3
    [httpinterface] =>
http://192.168.50.xx:80/plugins/text2speech/interfacedownload/
    [mp3filenameMD5] => 1d91695724e14c372529572496cc4048
    [jsonfilenameMD5] => 1d91695724e14c372529572496cc4048.json
    [jsonlogfileMD5] => 1d91695724e14c372529572496cc4048_log.json
    [durationms] => 2712
    [bitrate] => 192000
    [samplerate] => 48000
    [text] => Dies ist ein Interface Test
    [warning] =>
    [success] => 1
    [logging] => Array
        (
            [0] => Array
                (
                    [OK] => tts.php: Set Loglevel temporally to level 7 to
process Interface logging
                )
```

```
            [1] => Array
                (
                    [INFO] => tts.php: T2S Interface: POST request has been
received and will be processed!
                )

            [2] => Array
                (
                    [OK] => T2S Interface: Incoming POST request has been
successful processed, let's go ahead!
                )

            [3] => Array
                (
                    [DEB] => tts.php: fullmessageid:
1d91695724e14c372529572496cc4048 textstring: Dies ist ein Interface Test
                )

            [4] => Array
                (
                    [INFO] => tts.php: Processing time of creating MP3 file
tooks: 0.66518783569336 ms
                )

            [5] => Array
                (
                    [DEB] => tts.php: MS Azure has been successful selected
                )

            [6] => Array
                (
                    [DEB] => tts.php: Expected filename:
/opt/loxberry/data/plugins/text2speech/tts/1d91695724e14c372529572496cc4048.
mp3
                )

            [7] => Array
                (
                    [DEB] => tts.php: T2S will be called with 'Dies ist ein
Interface Test'
                )

            [8] => Array
                (
                    [DEB] => voice_engines/MS_Azure.php: Microsoft TTS has
been successful selected
                )

            [9] => Array
                (
                    [OK] => voice_engines/MS_Azure.php: The text has been
```

```
passed to Microsoft engine for MP3 creation
                )

            [10] => Array
                (
                    [OK] => voice_engines/MS_Azure.php: Everything went well
during TTS creation!
                )

            [11] => Array
                (
                    [INFO] => tts.php: Processing time of the complete T2S
request tooks: 0.84 Sek.
                )

            [12] => Array
                (
                    [DEB] => tts.php: filename of MP3 file:
1d91695724e14c372529572496cc4048
                )

            [13] => Array
                (
                    [OK] => tts.php: T2S Interface: JSON has been
successfully responded to Request
                )

        )

)
```

Hier einige Details zu den jeweiligen Keys:

- im Key **fullttspath** befindet sich der komplette LoxBerry Pfad und der Dateiname der erstellten MP3 Datei (nur für LoxBerry Plugins nutzbar)
- im Key **path** befindet sich nur der Pfad (nur für LoxBerry Plugins nutzbar)
- im Key **fullcifsinterface** befindet sich die komplette URL mit Pfad und der Dateiname der erstellten MP3 Datei (für Drittanwendungen nutzbar)
- im Key **cifsinterface** befindet sich die komplette URL mit Pfad (für Drittanwendungen nutzbar)
- im Key **fullhttpinterface** befindet sich die komplette http URL mit Pfad und der Dateiname der erstellten MP3 Datei (für Drittanwendungen nutzbar)
- im Key **httpinterface** befindet sich die komplette http URL mit Pfad (für Drittanwendungen nutzbar)
- im Key **mp3filenameMD5** befindet sich der MP3 Dateiname OHNE Endung (.mp3)
- im Key **jsonfilenameMD5** befindet sich der komplette JSON Dateiname mit dem Return Content
- im Key **jsonlogfileMD5** befindet sich der komplette JSON Dateiname NUR mit dem Logging Return (für Drittanwendungen nutzbar)
- im Key **durationms** befindet sich die Nettospieldauer der MP3
- im Key **bitrate** befindet sich die bitrate der MP3

- im Key **samplerate** befindet sich die samplerate der MP3
- im Key **text** befindet sich der Text der in eine MP3 umgewandelt wurde.
- im Key **warning** befindet sich die Fehlerbeschreibung (im Falle eines Fehlers bei der Voice Datei Erstellung).
- im Key **success** befindet sich der Rückgabecode (**1 = Success, 2 = Warning, 3 = Error**)
- im Key **logging** befindet sich das komplette Logging (für Drittanwendungen nutzbar)

Das generierte JSON file hat einen Dateinamen analog zum Dateinamen der generierten MP3 Dieser Name ist der MD5 gehashte Text, beide Files liegen im Loxberry unter folgendem Pfad (Samba share).

/opt/loxberry/data/text2speech/interfacedownload

Von hier kann es von der Drittanwendung bzw. dem Plugin eingelesen und weiterverarbeitet werden.

Beispiel einer Weiterverarbeitung der JSON Datei aus dem Sonos Plugin auf demgleichen Loxberry.

Beispiel aus Sonos Plugin (PHP)

```
// NOTE: path been generated by variables

// Get text from syntax and decrypt by using MD5 Hash
$text = md5(($_GET['text']));

// siehe weiter oben unter "Kompletter Request!!!

// ** Weiterverarbeitung des Returns **

// decode JSON data into Array
$json_a = json_decode($string, true);

// Logging Result vom Text-to-speech Plugin in Plugin inkludieren

$file = fopen($lbplogdir."/sonos.log","a",1);
# if error from Text-to-speech been received
if ($json_a['success'] == "2" or $json_a['success'] == "3")   {
    fwrite($file, date("H:i:s")." <ERROR> Interface: Text-to-speech
".$json_a['warning']."\n");
}
# loop through Logging array and write entries to Log File
foreach($json_a['logging'] as $key => $value) {
    foreach($value as $log => $text) {
        fwrite($file, date("H:i:s")." <".$log."> Interface: Text-to-speech
".$text."\n");
    }
}
# close file
fclose($file);
```

```
// select key full-server-URL and add to Play Queue
$sonos->AddToQueue("x-file-cifs://".$json_a['full-server-URL']);

// set Player to preselected Sonos Zone 'kueche'
$sonos->SetQueue("x-rincon-queue:".trim($sonoszone[$master][1])."#0");

// set Track to play from the actual position of the playlist
$sonos->SetTrack(1);

// set Zone specific Standard Volume for the T2S
$sonos->SetVolume($volume);

// Play T2S
$sonos->Play();

// wait until fully played
usleep($json_a['durationms'] * 1000);

// clear Queue
$sonos->ClearQueue();
```

Wenn Unterstützung während der Entwicklung notwendig ist bin ich gerne behilflich.