

Plugin Documentation Linux MTPCenter 2.0

Version 0.3.0

© 2006 - 2007 Matthias Panczyk

Table of Contents

General comments on this documentation.....	3
General comments for the Plugin interface.....	3
How does it work in general.....	3
A sample php script.....	3
How to manipulate the framework header.....	4
Important system variables.....	4
The Output Screen.....	4
Graphical overview of the main screen:.....	5
Output Array.....	5
Naming and directory structure.....	6
Directory structure for plugins.....	6
Naming convention for configurationfiles and plugin php scripts.....	6
Summary.....	8

General comments on this documentation

This documentation is a living document and may contain errors and for sure it will never be complete. Its function is only to help with the development for plugins.

In case you like to work with special Media Client features like playlists etc. you should ask the producer of your Media Client for a SDK. I for example have an SDK from Pinnacle but as it is under NDA I'm not allowed to share it.

General comments for the Plugin interface

Plugins for the Linux MTPCenter can be developed on your own. There are no rules unless you would like to share them on the official MTPCenter Homepage.

If you like to share your plugin with others I'm happy to upload them on my server including a short description. However in this case there are some rules

- The plugin must be licensed under GLP, the license text need to be in the tarball
- I need to have your complete mail address verified in case of legal issues
- You are the owner of the plugin, I will only distribute them for free on my webpage. Any legal problems with your plugin are coming back to you!
- Your real name be listed on the web as Author (developer) of the Plugin
- I will check the plugin from the technical side and whenever I have concerns I'm able to decline the publication on my web
- You need to use the standard output function of the MTPCenter, any own developed user interfaces will not be accepted.

How does it work in general

You will have more or less a framework where your applications are plugged in. This means there is no need for your own session control, for reading system wide configuration files and also there is one single output routine which is building the user interface.

A sample php script

Include all systemwide configuration like background picture, stylesheet information, session information, selected language, selected theme etc.

```
include ("../..../dialog_header.php");
```

Now it's place for your own stuff. You can add your own configuration files, your own application code etc.

```
....  
....
```

At the end we will have the call of the standard output function, more details about this a while later in the document.

```
Print_Table($Freebox_Table, $out_array, $MyCount_Start,  
            count($Channel_Array), $RowsAvailable, $Browser,  
            $Headline, $Key_Mapping,$Freebox_Key_Text, $OnLoadSet, $logo);
```

Plugins are detected automatically, also a needed database table will be created when defined.

How to manipulate the framework header

When calling additional php scripts there are two important variables:

session=1

When this variable is added to the url all old session information are deleted and a new session is started.

SH=1

With this variable in the url no headers are build, no background, no stylesheet, just nothing. You need to add this to your url whenever you want to feed multimedia content directly or whenever you need to send a playlist to the Browser or the media client.

Important system variables

There are some variables where you should take care of. Others here not documented variables can be found in the system configuration files.

<i>Variable</i>	<i>Use</i>
\$DocumentRoot	This is the root directory of the MTPCenter webserver
\$RemoteHost	The host which is calling the plugin
\$TMPDir	The host specific temp directory in case you need to store streams or whatever
\$ServerName	Webserver Server-IP
\$ServerPort	Webserver Port
\$ServerIP	Webserver_IP:Webserver_Port
\$Server	Browser String
\$Client	0 = Browser, 1 = Pinnacle Showcenter 200 ... Details can be found in config/internal/function.inc
\$Browser	Easy Check if it is browser or Media Client
\$Theme	Theme which is Used at the moment
\$LangUsed	Used Language

There are also some additional Variables available, but I think the most important ones are now listed here for a draft overview.

The Output Screen

This is the most complex and most difficult issue as the Media Clients are only using a subset of available html features, the browser is pretty simple. Because of this the output screen is pretty complex but also very limited. I tried to use stylesheets to make things easier, but at the end the stylesheet information are mainly exchanged with standard tags.

Graphical overview of the main screen:

	<i>Custom Logo</i>	<i>MTPCentzer Logo</i>	
	Headline + Dummy Row for Media Client Cursor Travelling(hidden)		
	Content		
	Content		
	Content		
	Content		
	Content		
	Content		
	Content		
	Content		
	Content		
	Dummy Row for Media Client Cursor Travelling(hidden)		
	List of available function Keys		

The things which you can influence are the Custom Logo, the Headline and the real content. The number of Content lines is limited by \$RowsAvailable. Please do not change this value to a higher number of Lines as the layout will then be destroyed.

You can defined the number of Columns you would like to us, you can also combine Rows and Lines. If you combine Lines this must be done for all Columns, otherwise Media Clients will not be able to show it correct.

If the Number of Content Lines is bigger then \$RowsAvailable a page up and / or pagedown button is automatically added.

Output Array

Each Cell can have Text or Images, formatting information and rowspan / colspan information. Please see the description below for details

Your information is feeded to the output function with an array which has always three planes.

`$output_array[x-position][y-position][type of content]`

Example:

```
$output_array[1][2][0]="Hello"  
Write "Hallo" to line Number 2 and column Number 3  
(as starting with 0 for first entry)
```

Type of Content:

- [0] Text which should be shown in cell
- [1] Colspan (Number of Columns combined)
- [2] Rowspan (Number of Lines combined)
- [3] Cell Width (don't use)
- [4] Link
- [5] Horizontal alignment

- [6] Text Color (no longer used in latest version)
- [7] Picture Width
- [8] Picture Height

Comments

- [0] When Using a picture instead of text the link is a bit special (sorry)

Example:

```
"<img src='http://".$ServerIP."/". $Theme."picture/icon_music.png'";
```

The following part will be added by the output function!!!

```
" height='".$output_table[$i][$j][8]."px'
  width='".$output_table[$i][$j][7]."px' border=0>";
```

- [2] Rowspan: If you use this function ensure that is it identical for all Columns
- [3] Please do not use this yet, the cell width is calculated automatically!!!!
- [5] Standard is left, valid entries are “textright” or “textcenter”
- [8] Please ensure the picture height is not more then \$TextFrameDimension[2]-2 [which is the content height]

Naming and directory structure

In general there are only a few rules you need to know. However we really suggest to use the structure presented here, then it will be easier for other developer to help with plugin development and easier for user to do the configuration.

Directory structure for plugins

The plugin directory structure is simple and described below. Feel free to add additional subdirectories below your plugin main directory.

Main plugin folder: `plugin`

Each plugin has it's own subdirectory in the plugin folder. Please keep the name simple and don't use special character or spaces!

Specific Plugin folder: `plugin - <pluginname>`

Example: `plugin/myplugin`

Plugin Configuration files: `plugin - <pluginname> - config`

Example: `plugin/myplugin/config`

Picture/Icon Directory: `plugin - <pluginname> - picture`

Example: `plugin/myplugin/picture`

Naming convention for configuration files and plugin php scripts

Ensure that you follow the guidelines here, otherwise your plugin may conflict with other plugins or with the MTPCenter main application.

Main configuration file config.ini

Each plugin MUST have a file named config.ini in the main plugin folder. The MTPCenter is reading this configuration file whenever it is started again.

Sample config.ini:

```
// Config File for MTPCenter freebox Plugin
// (c) 2006 Matthias Panczyk / Olivier Djian

// Lines with # at the beginning will be interpreted as plugin information for MTPCenter
// and will need some time to be interpreted

#Plugin_Startscript#freebox_main.php?session=1
#Plugin_Version#0.1
#Plugin_Active#yes
#Plugin_Icon#picture/freebox.png

#Plugin_Title#de#Freebox Fernsehen
#Plugin_Title#eng#Freebox TV

#Plugin_Database#freeboxtv
#Plugin_DB_Fields#Channelnumber INT
#Plugin_DB_Fields#Channelname VARCHAR(200)
#Plugin_DB_Fields#Link VARCHAR(200)
#Plugin_DB_Index#Primary
```

Explanation of the needed entries:

- Plugin_Startscript
 - The main plugin script which need to be called. Please add always ?session=1 so that all session variables are deleted.
- Plugin_Version
 - The version number of your plugin
- Plugin_Active
 - with yes you will see the plugin listed in the plugin menu of the MTPCenter, otherwise it will not be shown. By default active should be yes.
- Plugin_Title
 - The real Name of your plugin which will be shown in MTPCenter, you should define this for de (german) and eng (english). Please also ensure you have a fallback to english if the used language is not available
- Plugin_Database
 - This feature is optional so please use it only if you are working with a database table. In case you need a database table this will be the name of your table. The database will automatically be created with the next MTPCenter Start
- Plugin_Database_Fields
 - Here you can define all needed database fields in your Table. It's possible to add new fields in later versions of your plugin, however you cannot modify fields once they have been created.

config/plugin.inc

Use this configuration file for any configurable constants in your application.

config/plugin_lang.inc

Sample:

```
<?php
    $Freebox_Main_Header =
        array('de'=> "Freebox TV Kanäle", 'eng'=> "Freebox TV Channels");
    $Freebox_Key_Text     = array ("");
?>
```

This file should contain any language dependant text information.

config/plugin_internal.inc

If you need additional constants which should not be changed by the user you should create and use this file. However normally this will only be needed when you have created a more complex plugin.

Summary

I think now we have described the most important things for your plugin development. Please install our demo plugin freebox to see some more information how finally to code a plugin, the sources should be well documented.

If you think this documentation should have additional information feel free to add parts and send them to software@panczyk.org, I will add them to the original document.

Have fun

Matthias