# pDLNA

## Installation, Configuration and Debugging Guide

**Stefan Heumader**

**version 0.64.3**

**December 29, 2013**

# Contents


**1 Introduction** **5**

**2 Requirements** **6**
2.1 Perl . . . 6
2.2 Perl modules . . . 7
2.3 Third party software . . . 7
2.3.1 FFmpeg . . . 7

**3 Installing on UNIX and Linux** **12**
3.1 Install prerequisites . . . 12
3.1.1 Debian GNU/Linux 7 . . . 12
3.1.2 CentOS 6 . . . 12
3.1.3 FreeBSD 9 . . . 13
3.2 Installing via git . . . 14
3.2.1 Cloning the git repository . . . 14
3.2.2 Finalizing the installation . . . 15
3.2.3 Updating pDLNA . . . 18
3.3 Installing the latest release . . . 18
3.3.1 Downloading latest release . . . 18
3.3.2 Installation . . . 19
3.3.3 Updating pDLNA . . . 24

**4 Installing on Windows** **25**

**5 Installing on MacOS X** **26**

**6 Configuration** **27**
6.1 Global parameters . . . 27
6.1.1 FriendlyName . . . 27
6.1.2 Check4Updates . . . 28
6.1.3 Check4UpdatesNotification . . . 28
6.1.4 EnableGeneralStatistics . . . 28
6.1.5 PIDFile . . . 29
6.1.6 TempDir . . . 29
6.2 Database configuration . . . 29
6.2.1 DatabaseType . . . 29
6.2.2 DatabaseName . . . 30
6.2.3 DatabaseUsername and DatabasePassword . . . 31
6.3 Network configuration . . . 31

6.3.1 ListenInterface . . . 32
6.3.2 ListenIPAddress . . . 32
6.3.3 HTTPPort . . . 32
6.3.4 AllowedClients . . . 32
6.4 SSDP configuration . . . 33
6.4.1 CacheControl . . . 33
6.4.2 UUID . . . 33
6.5 DLNA configuration . . . 34
6.5.1 BufferSize . . . 34
6.5.2 SpecificViews . . . 34
6.5.3 EnableImageThumbnails . . . 34
6.5.4 EnableVideoThumbnails . . . 34
6.5.5 LowResourceMode . . . 35
6.5.6 FFmpegBinaryPath . . . 35
6.6 Logging . . . 35
6.6.1 LogFile . . . 35
6.6.2 LogFileMaxSize . . . 36
6.6.3 LogLevel . . . 36
6.6.4 LogCategory . . . 36
6.6.5 DateFormat . . . 37
6.7 Media Configuration . . . 37
6.7.1 RescanMediaInterval . . . 37
6.7.2 Directory . . . 38
6.7.3 External . . . 39
6.8 Transcoding Profiles . . . 40

**7 WebUI** **43**
7.1 Configured media . . . 43
7.2 Connected devices . . . 45
7.3 Information . . . 45

**8 Preinstalled VMware images** **50**
8.1 General information . . . 50
8.1.1 Included media files . . . 51
8.2 Specific Information . . . 52
8.2.1 CentOS 6 . . . 52
8.2.2 Debian GNU/Linux 7 . . . 54
8.2.3 FreeBSD 9 . . . 55

**9 Debugging and reporting problems** **57**
9.1 Debugging . . . 57
9.1.1 My device is not able to discover *pDLNA* . . . 58
9.1.2 My media directories are not read correctly . . . 59
9.1.3 My device is not able to list the directories/items shared by *pDLNA* . . . 59
9.1.4 *pDLNA* is not able to stream media items to my device . . . 59
9.2 Reporting problems . . . 59

**10 Known Issues** **60**
10.1 FreeBSD: initscript unable to completely shut down pDLNA . . . 60
10.1.1 Description . . . 60
10.1.2 Workaround . . . 60

**11 Fixed Issues** **61**
11.1 DirectoryListings not working when mandatory data fields not defined . . . 61
11.1.1 Describtion . . . 61
11.1.2 Workaround . . . 61
11.2 Movie::Info is still used by PDLNA::Media . . . 61
11.2.1 Describtion . . . 61
11.2.2 Workaround . . . 61
11.3 parsing problem if media file name contains special regex characters like '(', ')' or others . . . 62
11.3.1 Describtion . . . 62
11.3.2 Workaround . . . 62
11.4 Use of uninitialized value $request_line in string ne at /PDLNA/HTTPServer.pm line xxx. . . . 62
11.4.1 Describtion . . . 62
11.4.2 Workaround . . . 63
11.5 AllowedClients auto detection on FreeBSD is defect . . . 63
11.5.1 Describtion . . . 63
11.5.2 Workaround . . . 63

# Chapter 1

# Introduction

This document gives detailed information and instructions regarding the requirements, the installation and the configuration of *pDLNA* in its version 0.64.3.

The following chapter 2 gives an overview regarding the necessary requirements to run *pDLNA*. The chapters 3, 4 and 5 describe the installation steps on various operating systems and chapter 6 describes all the possible configuration parameters.

While chapter 7 gives a short introduction into the functionality of the WebUI, chapter 8 gives a description about the preinstalled *VMware* images, which are available to download.

Chapter 9 gives an overview regarding debugging or reporting problems of *pDLNA* on different operating systems.

In the end of this document, chapter 10 contains a list of known issues, which affect this version of *pDLNA* and possible workarounds, to fix the issue temporarily. And chapter 11 contains a list of issues, which have been fixed by this or prior versions of *pDLNA*.

If you have any questions at all, please do not hesitate to contact me.

# Chapter 2

# Requirements

This chapter gives an overview regarding the supported Perl versions, necessary Perl modules and additional/optional third party software, which are required for specific functionalities of *pDLNA*.

## 2.1 Perl

Currently, *pDLNA* has been tested with the following Perl versions:

- 5.10
- 5.12

Your installed Perl version can be determined by executing:

```
pdlna@mediaserver:~$ perl -v
```

Additionally, for those without interest in their Perl version, *pDLNA* in its current version has been tested with the following UNIX derivates or Linux distrubutions:

- Debian GNU/Linux 6 (squeeze)
- Debian GNU/Linux 7 (wheezy)
- CentOS 6
- FreeBSD 9

NOTE: *pDLNA* propably works with any other Perl version or Linux distrubution. Please feel free to contact me about your installation environment.

## 2.2 Perl modules

There are three tables, which give an overview about the necessary Perl modules regarding a specific Linux distrubution. While table 2.1 lists the Debian GNU/Linux 7 (and its variants) package names, table 2.2 lists the CentOS 6 package names (for a i686 installation) of these modules and finally table 2.3 names the FreeBSD 9 port names and their category.

If your favorite Linux distrubution is not one of those, you will need make sure, that these Perl modules are installed anyway.

As already mentioned you need to install these Perl modules. You are able to install these modules using *CPAN (Comprehensive Perl Archive Network)*[1] or even via the package management of your favourite Linux distribution.

For example, installing the *XML::Simple* Perl module can be installed via *CPAN* by using the following command

```
pdlna@mediaserver:~$ sudo cpan
cpan[1]> install XML::Simple
```

or by executing the following command

```
pdlna@mediaserver:~$ sudo apt-get install libxml-simple-perl
```

on the Debian GNU/Linux distribution and its variants.

For Perl modules without a package provided by your Linux distribution, you need to install it via *CPAN*.

## 2.3 Third party software

*pDLNA* requires for specific functionalities third party software, which is open source software.

### 2.3.1 FFmpeg

For enabling transcoding[2] of video and audio files, `FFmpeg` (`ffmpeg.org`) is required.

If no Transcoding Profiles (see section 6.8 for detailed information) are configured, `FFmpeg` is not required. If `LowResourceMode` is enabled, configured Transcoding Profiles will be ignored.

The `FFmpeg`'s source code or binaries can be obtained from the project's official website `ffmpeg.org` or can be installed via the package management of your favorite Linux distribution. The following command will install the `FFmpeg` package on the Debian GNU/Linux distribution and its variants.

[1] `cpan.perl.org/`

[2] Transcoding is converting specific media items on the fly to a (by the *DLNA* aware device) supported media format.

| Perl module name | Debian 7 package name |
|---|---|
| Audio::FLAC::Header | libaudio-flac-header-perl |
| Audio::Wav | libaudio-wav-perl |
| Audio::WMA | libaudio-wma-perl |
| Config | |
| Config::ApacheFormat | libconfig-apacheformat-perl |
| Data::Dumper | |
| Date::Format | |
| DBD::mysql | libdbd-mysql-perl |
| DBD::SQLite | libdbd-sqlite3-perl |
| DBI | libdbi-perl |
| Digest::MD5 | libdigest-md5-perl |
| Digest::SHA | libdigest-sha-perl |
| Fcntl | |
| File::Basename | |
| File::Glob | |
| File::MimeInfo | libfile-mimeinfo-perl |
| GD | libgd-gd2-perl |
| GD::Graph::area | libgd-graph-perl |
| Getopt::Long::Descriptive | libgetopt-long-descriptive-perl |
| Image::Info | libimage-info-perl |
| IO::Interface::Simple | libio-interface-perl |
| IO::Select | |
| IO::Socket | |
| IO::Socket::INET | |
| IO::Socket::Multicast | libio-socket-multicast-perl |
| LWP::UserAgent | |
| MP3::Info | libmp3-info-perl |
| MP4::Info | libmp4-info-perl |
| Net::IP | libnet-ip-perl |
| Net::Netmask | libnet-netmask-perl |
| Ogg::Vorbis::Header::PurePerl | libogg-vorbis-header-pureperl-perl |
| POSIX | |
| Proc::ProcessTable | libproc-processtable-perl |
| SOAP::Lite | libsoap-lite-perl |
| Socket | |
| Sys::Hostname | |
| Sys::Syslog | libsys-syslog-perl |
| threads | |
| threads::shared | |
| Time::HiRes | |
| URI::Split | |
| XML::Simple | libxml-simple-perl |

Table 2.1: Necessary Perl modules on Debian GNU/Linux 7

| Perl module name | CentOS 6 package name |
|---|---|
| Audio::FLAC::Header | |
| Audio::Wav | |
| Audio::WMA | |
| Config | |
| Config::ApacheFormat | |
| Data::Dumper | |
| Date::Format | |
| DBD::mysql | perl-DBD-MySQL.i686 |
| DBD::SQLite | perl-DBD-SQLite.i686 |
| DBI | perl-DBI.i686 |
| Digest::MD5 | |
| Digest::SHA | perl-Digest-SHA.i686 |
| Fcntl | |
| File::Basename | |
| File::Glob | |
| File::MimeInfo | |
| GD | perl-GD.i686 |
| GD::Graph::area | perl-GDGraph.noarch |
| Getopt::Long::Descriptive | |
| Image::Info | perl-Image-Info.noarch |
| IO::Interface | |
| IO::Select | |
| IO::Socket | |
| IO::Socket::INET | |
| IO::Socket::Multicast | |
| LWP::UserAgent | |
| MP3::Info | |
| MP4::Info | |
| Net::IP | perl-Net-IP.noarch |
| Net::Netmask | |
| Ogg::Vorbis::Header::PurePerl | |
| POSIX | |
| Proc::ProcessTable | |
| SOAP::Lite | perl-SOAP-Lite.noarch |
| Socket | |
| Sys::Hostname | |
| Sys::Syslog | |
| threads | |
| threads::shared | |
| Time::HiRes | perl-Time-HiRes.i686 |
| URI::Split | |
| XML::Simple | perl-XML-Simple.noarch |

Table 2.2: Necessary Perl modules on CentOS 6

| Perl module name | FreeBSD 9 port name |
|---|---|
| Audio::FLAC::Header | audio/p5-Audio-FLAC-Header |
| Audio::Wav | audio/p5-Audio-Wav |
| Audio::WMA | audio/p5-Audio-WMA |
| Config | |
| Config::ApacheFormat | devel/p5-Config-ApacheFormat |
| Data::Dumper | devel/p5-Data-Dumper |
| Date::Format | |
| DBD::mysql | databases/p5-DBD-mysql |
| DBD::SQLite | databases/p5-DBD-SQLite |
| DBI | databases/p5-DBI |
| Digest::MD5 | |
| Digest::SHA1 | security/p5-Digest-SHA1 |
| Fcntl | |
| File::Basename | |
| File::Glob | |
| File::MimeInfo | devel/p5-File-MimeInfo |
| GD | graphics/p5-GD |
| GD::Graph::area | graphics/p5-GD-Graph |
| Getopt::Long::Descriptive | devel/p5-Getopt-Long-Descriptive |
| Image::Info | graphics/p5-Image-Info |
| IO::Interface | net/p5-IO-Interface |
| IO::Select | |
| IO::Socket | |
| IO::Socket::INET | |
| IO::Socket::Multicast | net/p5-IO-Socket-Multicast |
| LWP::UserAgent | |
| MP3::Info | audio/p5-MP3-Info |
| MP4::Info | multimedia/p5-MP4-Info |
| Net::IP | net-mgmt/p5-Net-IP |
| Net::Netmask | net-mgmt/p5-Net-Netmask |
| Ogg::Vorbis::Header::PurePerl | audio/p5-Ogg-Vorbis-Header-PurePerl |
| POSIX | |
| Proc::ProcessTable | devel/p5-Proc-ProcessTable |
| SOAP::Lite | net/p5-SOAP-Lite |
| Socket | net/p5-Socket |
| Sys::Hostname | |
| Sys::Syslog | sysutils/p5-Sys-Syslog |
| threads | |
| threads::shared | |
| Time::HiRes | devel/p5-Time-HiRes |
| URI::Split | |
| XML::Simple | textproc/p5-XML-Simple |

Table 2.3: Necessary Perl modules on FreeBSD 9

```
pdlna@mediaserver:~$ sudo apt-get install ffmpeg
```

The official website of `FFmpeg` does also provide information, where to get binaries for your Windows operating system.

# Chapter 3

# Installing on UNIX and Linux

This chapter gives an overview about the different methods for installing *pDLNA* on your favorite Linux distribution. The first section gives an overview about installing the necessary prerequisites for various Linux distributions. The second section describes the installation steps via the official `git` repository, while the third section describes installing *pDLNA* from a packed tarball.

## 3.1 Install prerequisites

### 3.1.1 Debian GNU/Linux 7

On a fresh installed Debian GNU/Linux 7 operating system, you are able to execute the following command to install necessary Debian GNU/Linux packages and all the necessary Perl modules from the Debian repository:

```
pdlna@mediaserver:~$ sudo apt-get install \
 ffmpeg libaudio-flac-header-perl libaudio-wav-perl \
 libaudio-wma-perl libconfig-apacheformat-perl libdbd-mysql-perl \
 libdbd-sqlite3-perl libdbi-perl libdigest-md5-perl \
 libdigest-sha-perl libfile-mimeinfo-perl libgd-gd2-perl \
 libgd-graph-perl libgetopt-long-descriptive-perl \
 libimage-info-perl libio-interface-perl \
 libio-socket-multicast-perl libmp3-info-perl libmp4-info-perl \
 libnet-ip-perl libnet-netmask-perl \
 libogg-vorbis-header-pureperl-perl libproc-processtable-perl \
 libsoap-lite-perl libsys-syslog-perl libxml-simple-perl
```

### 3.1.2 CentOS 6

On a fresh CentOS 6 (mininal) installation, the following command installs necessary CentOS packages and some necessary Perl Modules from the official repository:

```
pdlna@mediaserver:~$ sudo yum install \
 git bind-utils sudo cpan make gcc \
 libogg-devel.i686 libogg.i686 libvorbis.i686 \
 libvorbis-devel.i686 vorbis-tools.i686 \
 perl-Digest-SHA.i686 perl-GD.i686 \
 perl-Image-Info.noarch perl-Net-IP.noarch \
 perl-XML-Simple.noarch shared-mime-info.i686 \
 perl-DBI.i686 perl-DBD-SQLite.i686 \
 perl-GDGraph.noarch perl-SOAP-Lite.noarch \
 perl-DBD-MySQL.i686
```

IMPORTANT NOTE: In the default repositories of CentOS, there is no package for `FFmpeg`. Either install `FFmpeg` manually or run *pDLNA* in `LowResourceMode` (see section 6.5.5 for detailed information).

Afterwards you still need to install a lot of Perl modules via *CPAN* itself, since they are not included in the official repository:

```
pdlna@mediaserver:~$ sudo cpan
cpan[1]> install YAML
cpan[2]> install Getopt::Long::Descriptive
cpan[3]> install Audio::FLAC::Header
cpan[4]> install Audio::Wav
cpan[5]> install Audio::WMA
cpan[6]> install Config::ApacheFormat
cpan[7]> install Date::Format
cpan[8]> install File::MimeInfo
cpan[9]> install IO::Interface
cpan[10]> install IO::Socket::Multicast
cpan[11]> install MP3::Info
cpan[12]> install MP4::Info
cpan[13]> install Net::Netmask
cpan[14]> install Inline::MakeMaker
cpan[15]> install Ogg::Vorbis::Header::PurePerl
cpan[16]> install Proc::ProcessTable
```

### 3.1.3 FreeBSD 9

Since FreeBSD 9 brings `portsnap`, which is a fast and user-friendly tool for retrieving the *Ports Collection*, installing required software, which is part of the *Ports Collection* is easy. The following three commands will download all *Ports*, extract them and will also update the the *Ports* tree.

```
[pdlna@mediaserver ~]$ sudo portsnap fetch
[pdlna@mediaserver ~]$ sudo portsnap extract
[pdlna@mediaserver ~]$ sudo portsnap update
```

Afterwards, you will be able to look, if there is already a *Port* for the required software available, by simply visiting `http://www.freebsd.org/cgi/ports.cgi`. If the *Port* is available, it will list the category and its name. Afterwards you are able to compile and install the software. For instance, if you would like to install `git`, which is available in the category `devel`, this can be done by the following command:

```
[pdlna@mediaserver ~]$ cd /usr/ports/devel/git
[pdlna@mediaserver /usr/ports/devel/git]$ sudo make install clean
```

IMPORTANT NOTE: FreeBSD 9 comes with a **non-threaded** Perl version. Please ensure to deinstall the running Perl version and recompile it with threads. This can be done by the command `[pdlna@ms /usr/ports/lang/perl5.12/]$ make WITH_THREADS=yes`.

To install all required Perl modules, which are included in the *Ports Collection* (see table 2.3), follow the two steps listed above for compiling and installing. Additionally, you are able to install the Perl module `File::Copy::Recursive` (necessary for the `install.pl` script) which is named `p5-File-Copy-Recursive` and member of the `devel` category.

Afterwards you still need to install some Perl modules via *CPAN* itself, since they are not included in the *Ports Collection*:

```
[pdlna@mediaserver ~]$ sudo cpan
cpan[1]> install Date::Format
cpan[2]> install LWP::UserAgent
```

If you would like *pDLNA* to use `FFmpeg` (see section 2.3.1), you are also able to install them via the *Ports Collection* by the following commands:

```
[pdlna@mediaserver ~]$ cd /usr/ports/multimedia/ffmpeg
[pdlna@mediaserver /usr/ports/multimedia/ffmpeg]$
  sudo make install clean
```

## 3.2 Installing via git

Installing *pDLNA* via a `git clone` is a simple way to install *pDLNA* und keep it up to date. `git` is a distributed revision control system and is developed by Linus Torvalds. The official `git` repository is hosted on GitHub (`github.com/geuma/pDLNA/`), which is a web-based hosting service for software development projects.

### 3.2.1 Cloning the git repository

At first the `git` software must be installed on the computer, *pDLNA* should be run at. The `git` source code can be obtained from the project's official website `git-scm.com/` or can be installed via the package management of your favorite Linux distribution. The following command will install the `git` package on the Debian GNU/Linux distribution and its variants.

```
pdlna@mediaserver:~$ sudo apt-get install git
```

After installing `git` you need to clone the repository by executing the following command:

```
pdlna@mediaserver:~$ git clone git://github.com/geuma/pDLNA.git
```

In the end, a directory named *pDLNA* has been created, which should look like the following directory listing.

```
pdlna@mediaserver:~/pDLNA$ ls -lah
total 120K
drwxr-xr-x 6 pdlna pdlna 4.0K Mar  5 07:56 .
drwxr-xr-x 5 pdlna pdlna 4.0K Aug 15  2012 ..
-rwxr-xr-x 1 pdlna pdlna  15K Mar  5 07:56 CHANGELOG
drwxr-xr-x 2 pdlna pdlna 4.0K Feb 11 08:45 external_programs
drwxr-xr-x 8 pdlna pdlna 4.0K Mar  5 07:56 .git
-rwxr-xr-x 1 pdlna pdlna 1.4K Feb 11 08:45 INSTALL
-rwxr-xr-x 1 pdlna pdlna 6.1K Mar  5 07:56 install.pl
-rwxr-xr-x 1 pdlna pdlna  35K Aug  7  2012 LICENSE
drwxr-xr-x 2 pdlna pdlna 4.0K Mar  5 07:56 PDLNA
-rwxr-xr-x 1 pdlna pdlna 9.4K Mar  5 07:56 pdlna.conf
-rwxr-xr-x 1 pdlna pdlna 2.9K Mar  5 07:56 pDLNA.pl
-rwxr-xr-x 1 pdlna pdlna 2.2K Feb 11 08:45 rc.pDLNA
-rwxr-xr-x 1 pdlna pdlna  831 Feb 13 20:27 README
-rwxr-xr-x 1 pdlna pdlna 1.3K Mar  5 07:56 TODO
-rwxr-xr-x 1 pdlna pdlna   22 Mar  5 07:56 VERSION
```

### 3.2.2 Finalizing the installation

The easiest way to finalize the installation is to copy the default configuration file from the `git clone` to the `/etc/` directory by executing the following commands. For copying the configuration file you might need superuser rights.

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ sudo cp pdlna.conf /etc/
```

Additionally you should copy the sample initscript from the `git clone` to the `/etc/init.d/` directory and setting the execute bit by executing the following commands. This step might also require superuser rights.

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ sudo cp rc.pDLNA /etc/init.d/
pdlna@mediaserver:~/pDLNA$ sudo chmod +x /etc/init.d/rc.pDLNA
```

In the end you need to change the `DIR` variable in the initscript (line 20) by editing the file `/etc/init.d/rc.pDLNA` with your favourite editor like *vim*, *nano* or whatever you like. The following snippet shows you an example command:

```
pdlna@mediaserver:~$ sudo vim /etc/init.d/rc.pDLNA
```

After opening the file, jump to line 20 and edit the path for the *DIR* variable to the path, where the *pDLNA.pl* is stored. If you have cloned the `git` repository to `/home/pdlna/pDLNA/` you need to set the variable to the following value:

```
DIR="/home/pdlna/pDLNA/"
```

Additionally you need to check for the dependencies (see chapter 2) you are able to run the `install.pl` script with the (`-c` or `--checkrequirements`) parameter, which is stored in the `git` repository.

```
pdlna@mediaserver:~/pDLNA$ perl install.pl -c
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use DBD::mysql;
ok 9 - use DBD::SQLite;
ok 10 - use DBI;
ok 11 - use Digest::MD5;
ok 12 - use Digest::SHA;
ok 13 - use Fcntl;
ok 14 - use File::Basename;
ok 15 - use File::Glob;
ok 16 - use File::MimeInfo;
ok 17 - use GD;
ok 18 - use GD::Graph::area;
ok 19 - use Getopt::Long::Descriptive;
ok 20 - use Image::Info;
ok 21 - use IO::Interface::Simple;
ok 22 - use IO::Select;
ok 23 - use IO::Socket;
ok 24 - use IO::Socket::INET;
ok 25 - use IO::Socket::Multicast;
ok 26 - use LWP::UserAgent;
ok 27 - use MP3::Info;
ok 28 - use MP4::Info;
ok 29 - use Net::IP;
ok 30 - use Net::Netmask;
ok 31 - use Ogg::Vorbis::Header::PurePerl;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use SOAP::Lite;
ok 35 - use Socket;
ok 36 - use Sys::Hostname;
ok 37 - use Sys::Syslog;
ok 38 - use threads;
ok 39 - use threads::shared;
ok 40 - use Time::HiRes;
ok 41 - use URI::Split;
ok 42 - use XML::Simple;
1..42
```

The output above shows the executed `install.pl` script with its output. As long as all of these checks return `ok`, all necessary requirements are fullfilled.

IMPORTANT NOTE: The checking will not include checking if `FFmpeg` is installed. For detailed information regarding the usage of `FFmpeg` see section 2.3.1.

After the initial configuration (see chapter 6), you are able to start *pDLNA* by executing the following command:

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

### 3.2.3 Updating pDLNA

Because of the `git clone` updating *pDLNA* is pretty easy if there have not been any changes to your `git clone`. By executing the following commands

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ git pull
```

*pDLNA* will be updated to the latest version, which is currently pushed to the git repository. After checking for requirements by executing `install.pl -c` and restarting *pDLNA* via

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA restart
```

the new version is going to be started.

## 3.3 Installing the latest release

Installing the latest release of *pDLNA* is the simpliest way to install *pDLNA*. Every release of *pDLNA* will be packaged as a tarball and available via the changelog section on `www.pdlna.org/cgi-bin/index.pl?menu=changelog` or can be downloaded directly via `http://www.pdlna.org/cgi-bin/index.pl?menu=download&type=release&version=latest`.

### 3.3.1 Downloading latest release

The latest release of *pDLNA* can be downloaded via `http://www.pdlna.org/cgi-bin/index.pl?menu=download&type=release&version=latest`. The changelog section (`www.pdlna.org/cgi-bin/index.pl?menu=changelog`) gives detailed information about all releases.

The following snippet shows the command to download the latest version using `wget`:

```
pdlna@mediaserver:~$ cd /tmp/
pdlna@mediaserver:/tmp$ wget \
  http://www.pdlna.org/cgi-bin/index.pl?menu=download&type=release&version=latest
```

### 3.3.2 Installation

After downloading the latest version of *pDLNA* to the `/tmp` directory, you should extract the tarball and change to the extracted directory by executing the following commands:

```
pdlna@mediaserver:/tmp$ tar xzf pDLNA-$pDLNAversion.tgz
pdlna@mediaserver:/tmp$ cd pDLNA
```

In this directory, there is a script called `install.pl`, which supports checking for requirements (`-c` or `--checkrequirements`) and installing the application (`-i` or `--install`). Additionally the help function (`-h` or `--help`) prints out more detailed information.

So the first step to install *pDLNA* should be to run the installation script to check for the necessary requirements (see chapter 2) by executing the following command:

```
pdlna@mediaserver:/tmp/pDLNA$ perl install.pl -c
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use DBD::mysql;
ok 9 - use DBD::SQLite;
ok 10 - use DBI;
ok 11 - use Digest::MD5;
ok 12 - use Digest::SHA;
ok 13 - use Fcntl;
ok 14 - use File::Basename;
ok 15 - use File::Glob;
ok 16 - use File::MimeInfo;
ok 17 - use GD;
ok 18 - use GD::Graph::area;
ok 19 - use Getopt::Long::Descriptive;
ok 20 - use Image::Info;
ok 21 - use IO::Interface::Simple;
ok 22 - use IO::Select;
ok 23 - use IO::Socket;
ok 24 - use IO::Socket::INET;
ok 25 - use IO::Socket::Multicast;
ok 26 - use LWP::UserAgent;
ok 27 - use MP3::Info;
ok 28 - use MP4::Info;
ok 29 - use Net::IP;
ok 30 - use Net::Netmask;
ok 31 - use Ogg::Vorbis::Header::PurePerl;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use SOAP::Lite;
ok 35 - use Socket;
ok 36 - use Sys::Hostname;
ok 37 - use Sys::Syslog;
ok 38 - use threads;
ok 39 - use threads::shared;
ok 40 - use Time::HiRes;
ok 41 - use URI::Split;
ok 42 - use XML::Simple;
1..42
```

The attached output, shows the different tests and their results, which were performed by the installation script. For a complete overview regarding the requirements please see chapter 2.

IMPORTANT NOTE: The checking will not include checking if `FFmpeg` is installed. For detailed information regarding the usage of `FFmpeg` see section 2.3.1.

When all requirements have been fixed and installed, you are able to rerun the installation script by executing the installation script with the `--install` parameter. By default, the script will install *pDLNA* to the `/opt` directory, which can be changed by setting the `--prefix=/path/to/your/directory` parameter. The following output shows the installation process, which is started by checking the requirements for *pDLNA*. While step 2 is checking for two more Perl modules for installation, step 3 installs the necessary files from *pDLNA* to the specified directory and step 4 modifies the installed files for the installation specific parts. In the end, step 5 verifies the installation.

```
pdlna@mediaserver:/tmp/pDLNA$ sudo perl install.pl -i
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use DBD::mysql;
ok 9 - use DBD::SQLite;
ok 10 - use DBI;
ok 11 - use Digest::MD5;
ok 12 - use Digest::SHA;
ok 13 - use Fcntl;
ok 14 - use File::Basename;
ok 15 - use File::Glob;
ok 16 - use File::MimeInfo;
ok 17 - use GD;
ok 18 - use GD::Graph::area;
ok 19 - use Getopt::Long::Descriptive;
ok 20 - use Image::Info;
ok 21 - use IO::Interface::Simple;
ok 22 - use IO::Select;
ok 23 - use IO::Socket;
ok 24 - use IO::Socket::INET;
ok 25 - use IO::Socket::Multicast;
ok 26 - use LWP::UserAgent;
ok 27 - use MP3::Info;
ok 28 - use MP4::Info;
ok 29 - use Net::IP;
ok 30 - use Net::Netmask;
ok 31 - use Ogg::Vorbis::Header::PurePerl;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use SOAP::Lite;
ok 35 - use Socket;
ok 36 - use Sys::Hostname;
ok 37 - use Sys::Syslog;
ok 38 - use threads;
ok 39 - use threads::shared;
ok 40 - use Time::HiRes;
ok 41 - use URI::Split;
ok 42 - use XML::Simple;
-------------------------------------------------------
```

```
Step 2:
Testing for necessary Perl Modules for installation ...
-------------------------------------------------------
ok 43 - use File::Copy;
ok 44 - use File::Copy::Recursive;
-------------------------------------------------------
Step 3:
Installing files ...
-------------------------------------------------------
ok 45 - Installed './PDLNA' to '/opt/pDLNA/PDLNA'.
ok 46 - Set rights for '/opt/pDLNA/PDLNA'.
ok 47 - Installed './external_programs' to
 '/opt/pDLNA/external_programs'.
ok 48 - Set rights for '/opt/pDLNA/external_programs'.
ok 49 - Installed './VERSION' to '/opt/pDLNA/VERSION'.
ok 50 - Set rights for '/opt/pDLNA/VERSION'.
ok 51 - Installed './pDLNA.pl' to '/opt/pDLNA/pDLNA.pl'.
ok 52 - Set rights for '/opt/pDLNA/pDLNA.pl'.
ok 53 - Installed './LICENSE' to '/opt/pDLNA/LICENSE'.
ok 54 - Set rights for '/opt/pDLNA/LICENSE'.
ok 55 - Installed './README' to '/opt/pDLNA/README'.
ok 56 - Set rights for '/opt/pDLNA/README'.
ok 57 - Installed './pdlna.conf' to '/etc/pdlna.conf'.
ok 58 - Set rights for '/etc/pdlna.conf'.
ok 59 - Installed './rc.pDLNA' to '/etc/init.d/rc.pDLNA'.
ok 60 - Set rights for '/etc/init.d/rc.pDLNA'.
-------------------------------------------------------
Step 4:
Setting of relevant paths ...
-------------------------------------------------------
ok 61 - Changed path for binary in '/etc/init.d/rc.pDLNA'.
ok 62 - Changed path for lib in '/opt/pDLNA/pDLNA.pl'.
-------------------------------------------------------
Step 5:
Checking for pDLNA Perl Modules ...
-------------------------------------------------------
ok 63 - use PDLNA::Config;
ok 64 - use PDLNA::ContentLibrary;
ok 65 - use PDLNA::Daemon;
ok 66 - use PDLNA::Database;
ok 67 - use PDLNA::Devices;
ok 68 - use PDLNA::FFmpeg;
ok 69 - use PDLNA::HTTPServer;
ok 70 - use PDLNA::HTTPXML;
ok 71 - use PDLNA::Log;
ok 72 - use PDLNA::Media;
ok 73 - use PDLNA::SOAPClient;
ok 74 - use PDLNA::SOAPMessages;
ok 75 - use PDLNA::SpecificViews;
ok 76 - use PDLNA::SSDP;
ok 77 - use PDLNA::Statistics;
ok 78 - use PDLNA::Status;
ok 79 - use PDLNA::Utils;
ok 80 - use PDLNA::WebUI;
1..80
```

After configuring your *pDLNA* installation (see chapter 6), you are able to start *pDLNA* by executing the following command:

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

### 3.3.3 Updating pDLNA

Currently, there is no simple way to update this *pDLNA* installation. Actually, you should do a backup of the `/etc/pdlna.conf` configuration file by executing for instance

```
pdlna@mediaserver:~$ sudo cp /etc/pdlna.conf /etc/pdlna.conf.bak
```

and rerun the installation process for the new version again (like described in section 3.3). In the end, you are able to restore the old configuration file by simple executing the following command:

```
pdlna@mediaserver:~$ sudo cp /etc/pdlna.conf.bak /etc/pdlna.conf
```

# Chapter 4

# Installing on Windows

Currently installing and running *pDLNA* has not been tested on a Microsoft Windows operation system yet. Please contact me about your experiences regarding installation and/or executing *pDLNA* on Windows.

NOTE: If somebody is interrested into porting and maintaining *pDLNA* for Windows, please contact me and start doing it. Thanks.

# Chapter 5

# Installing on MacOS X

*pDLNA* has not been tested on a MacOS X operating system yet either. So please contact me about your experiences regarding installation and/or executing *pDLNA* on a Apple operating system.

NOTE: In fact, you should be able to install *pDLNA* similar as described in chapter 3.

# Chapter 6

# Configuration

This chapter gives an overview about the possible parameters to configure *pDLNA* and describes their functionality and their possible impact on the installation. By default, the configuration file is stored in `/etc/pdlna.conf`. If you would like to change the location of the configuration file, you need to change the following line in the initscript (by default `/etc/init.d/rc.pDLNA`) to the correct location:

```
CFGFILE="/etc/pdlna.conf"
```

Some of the available configuration parameters are binary values, which can be enabled or disabled. To enable one of those parameters, simple configure the parameter to one of the following values:

- `on`
- `true`
- `yes`
- `enabled`
- `enable`
- `1`

The parsing of these values is case insenstive and if you would like to disable one of these parameters, simple use another value (.e.g. `disabled` or `Off`).

## 6.1 Global parameters

### 6.1.1 FriendlyName

The *FriendlyName* configures a name, which will be shown by all capable DNLA devices to identify a running digital media server. The maximum length of the FriendlyName is limited to 32 characters and the allowed characters are:

- letters and number

| VARIABLE | DESCRIPTION |
|---|---|
| `$VERSION` | the currently running version, for instance v0.64.3 |
| `$HOSTNAME` | the systems hostname |
| `$OS` | the operating system's name |

Table 6.1: Available FriendlyName configuration variables

- spaces and dots
- dashes and underscores

Additionally, you are able to use the in table6.1 listed variables in the *FriendlyName*.

By default, the *FriendlyName* will be set to `pDLNA $VERSION on $HOSTNAME` or you are able to define your *FriendlyName* by configuring it like this:

```
FriendlyName 'pDLNA $VERSION on $HOSTNAME'
```

### 6.1.2 Check4Updates

If *Check4Updates* is enabled, the running *pDLNA* installation will check every 24 hours, if there is a new version of *pDLNA* available. Therefore, *pDLNA* will do a HTTP request to `www.pdlna.org/cgi-bin/status.pl` and transmits its **version number** including the information if it is a **beta release** and the configured or generated **UUID** as XML data to the server. The response from the server is XML data too, which will be evaluated by the running *pDLNA* installation and the result will be logged by the running *pDLNA* installation.

If you do not like *pDLNA* to check for a new version, you are able to deactivate this feature by adding the following line to your configuration file:

```
Check4Updates Off
```

### 6.1.3 Check4UpdatesNotification

If *Check4Updates* and *Check4UpdatesNotification* is enabled and a device with a `urn:samsung.com:serviceId:MessageBoxService` is connected, *pDLNA* will send a message regarding an available update to this service.

You are also able to deactivate this *FunFeature* by adding the following line to your configuration file:

```
Check4UpdatesNotification Off
```

### 6.1.4 EnableGeneralStatistics

With the *EnableGeneralStatistics* configuration parameter, you are able to specify if *pDLNA* should store statistics data like

- memory usage

- amount of media items

in the database and should draw graphs in the WebUI.

IMPORTANT NOTE: This feature is currently not available on FreeBSD.

By default, this feature is activated and you are able to deactivate it by adding the following line to your configuration file.

```
EnableGeneralStatistics Off
```

### 6.1.5 PIDFile

This parameter specifies the location of the *PIDFile*, which is used to store the process ID of a running *pDLNA* installation. By default, the parameter is set to `/var/run/pdlna.pid` or you are able to specify a different location by configuring it like this:

```
PIDFile /var/run/pdlna.pid
```

IMPORTANT NOTE: Please ensure, that the user, which is running *pDLNA*, has permissions to write to the configured path.

### 6.1.6 TempDir

This parameter defines the directory, where *pDLNA* will store temporary files for thumbnails and so on. By default, this parameter is set to `/tmp` or you are able to specify a different location by configuring it like this:

```
TempDir /tmp
```

IMPORTANT NOTE: Please ensure, that the user, which is running *pDLNA*, has permissions to write to the configured path.

## 6.2 Database configuration

### 6.2.1 DatabaseType

The parameter *DatabaseType* defines the database, which is used by *pDLNA* to store all th e necessary information (like the media items, which are member of the media library). Currently the following database types (which are listed in table 6.2) are available.

By default, this parameter is set to `SQLITE3` or you are able to specify a different database type by configuring it like this:

```
DatabaseType MYSQL
```

| PARAMETER | DESCRIPTION |
|---|---|
| SQLITE3 | SQLite database in version 3 |
| MYSQL | MySQL database |

Table 6.2: Available DatabaseType configuration parameters

NOTE: Using MySQL as a database for *pDLNA* improves the performance. A short performance comparison regarding *pDLNA* can be accessed on `http://www.pdlna.org/cgi-bin/index.pl?menu=news&id=v0.64b`.

#### Preparing MySQL for pDLNA

To use MySQL as a database server for *pDLNA*, creating a database and a seperate user is recommended. This can be achieved by logging in as an administrative user (for instance `root`) and executing the following commands:

```
pdlna@mediaserver:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8896

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database pdlna;
mysql> grant usage on pdlna.* to pdlna@localhost identified by
 '<yourpassword>';
mysql> grant all privileges on pdlna.* to pdlna@localhost;
```

Finally, *pDLNA* needs to be configured to use the created MySQL database with the according username and password. This is described in the following two sections 6.2.2 and 6.2.3.

### 6.2.2 DatabaseName

Configuring the parameter *DatabaseName* depends on the configuration parameter *DatabaseType*.

#### SQLite database in version 3

This parameter defines the location, where *pDLNA* will store the SQLite database file itself. By default, this parameter is set to `/tmp/pdlna.db` or you are able to specify a different location by configuring it like this:

```
DatabaseName /tmp/pdlna.db
```

IMPORTANT NOTE: Please ensure, that the user, which is running *pDLNA*, has permissions to write to the configured path.

#### MySQL database

This configuration parameter defines the name of the database, which should be used by *pDLNA*. By default, this parameter is set to `pdlna` or you are able to specify a different database by configuring it like this:

```
DatabaseName pdlna
```

### 6.2.3 DatabaseUsername and DatabasePassword

These two configuration parameters need to be configured, if the *DatabaseType* is configured to `MYSQL` (see section 6.2.1) and represent the authentication information for the configured database server.

These configuration parameters can be configured like this:

```
DatabaseUsername pdlna
DatabasePassword <yourpassword>
```

By default, if no *DatabaseUsername is configured, it will be set to* `pdlna`. *And if no* DatabasePassword is configured, no password will be used to authenticate to the database.

IMPORTANT NOTE: It is not recommended to use the database server's administrative (for instance `root`) user.

## 6.3 Network configuration

If you are not sure about the installed network interfaces and configured IP addresses, execute the following command:

```
pdlna@mediaserver:~$ sudo ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state
 UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
 pfifo_fast state UNKNOWN qlen 1000
    link/ether 00:0c:29:bc:fc:da brd ff:ff:ff:ff:ff:ff
    inet 192.168.145.139/24 brd 192.168.145.255 scope global eth0
    inet6 fe80::20c:29ff:febc:fcda/64 scope link
       valid_lft forever preferred_lft forever
```

In the following two sections regarding the configuration of *ListenInterface* and *ListenIPAddress* this output will be taken as an example.

### 6.3.1 ListenInterface

The *ListenInterface* parameter specifies the network interface, *pDLNA* should be using. If the parameter is **not** specified, *pDLNA* will try to determine the first active network interface with an IP address.

In case, that *pDLNA* was not able to determine the correct network interface, you need to specify it by the following configuration line:

```
ListenInterface eth0
```

### 6.3.2 ListenIPAddress

The *ListenIPAddress* parameter specifies the IP address, *pDLNA* should use to communicate with the *DLNA* capable devices. If *ListenIPAddress* is **not** specified, *pDLNA* will try to determine the first IP address on the configured or even detected *ListenInterface*. Otherwise you are able to define it by configuring *ListenIPAddress* like this:

```
ListenIPAddress 192.168.145.139
```

### 6.3.3 HTTPPort

The *HTTPPort* parameter defines the TCP port, which should be used by the integrated HTTP server. By default it will be set to `8001` or you are able to define another by the following parameter:

```
HTTPPort 8001
```

IMPORTANT NOTE: Please ensure, that the configured TCP port is **not** used by any other application, otherwise *pDLNA* will not be able to start up correctly.

### 6.3.4 AllowedClients

For reasons of data privacy, you need to specify IP address(es) and/or subnet(s), which should be able to communicate with *pDLNA*. *SSDP* is limited to the local subnet because of the Multicast communication (as long as there is no Multicast Routing), but HTTP is not limited to the local subnet.

Nevertheless, *pDLNA* listens to UDP communication on port 1900 and it is recommended to restrict the *AllowedClients* as good as possible.

```
AllowedClients 192.168.145.2, 192.168.145.128/26, \
 192.168.145.192/255.255.255.252
```

IMPORTANT NOTE: if **none** are specified, the *ListenIPAddresses* local subnet will be configured. Please limit the number of hosts.

IMPORTANT NOTE: the hosts, which should access the WebUI (see chapter 7) have also to be added to the *AllowedClients* configuration parameter.

| PARAMETER | DESCRIPTION |
|---|---|
| `Version3` | the hostname's MD5 checksum |
| `Version4` | random generated UUID |
| `Version4MAC` | random generated UUID including the MAC address of the configured *ListenInterface* in the end |
| `Version5` | the hostname's SHA-1 checksum |
| `<staticUUID>` | define a static UUID, which is formated like `56657273-696f-6e34-4d41-000c29bcfcda` |

Table 6.3: Available UUID configuration parameters

## 6.4 SSDP configuration

### 6.4.1 CacheControl

*CacheControl* represents a parameter in the *Simple Service Discovery Protocol (SSDP)* and defines the time in seconds, clients will cache the server's information. The value has impact on the interval *pDLNA* is going to send out his *SSDP* alive messages.

Most devices and *pDLNA* use as default value `1800` seconds. If you would like to use a different value, define it in seconds like this:

```
CacheControl 1800
```

IMPORTANT NOTE: Changing this value **may** result into malfunction of *pDLNA*.

### 6.4.2 UUID

The *Universally Unique Identifier (UUID)* is used in the *Simple Service Discovery Protocol (SSDP)* as an unique identifier. RFC 4122[1] describes the format and the different methods to generate a UUID.

The available configuration parameters are listed in table 6.3. If **none** is specified, `Version4` will be used or you are able to change the parameter by the following line:

```
UUID Version4MAC
```

IMPORTANT NOTE: the method for generating the UUIDs are **not** (completely) compliant to RFC 4122 and are **only** pseudo-random.

[1] `www.ietf.org/rfc/rfc4122.txt`

## 6.5 DLNA configuration

### 6.5.1 BufferSize

The *BufferSize* defines the default size of a chunk, which is used to transfer streaming data. The default value for the *BufferSize* is `32768`. The following example shows how to change the value:

```
BufferSize 1337
```

IMPORTANT NOTE: Changing this value **may** result into malfunction of *pDLNA* or may even result into a crash of your system because of high memory usage.

### 6.5.2 SpecificViews

The *SpecificViews* configuration parameters enables for specific *DLNA* aware devices, like

- Samsung TV

some different method for directory listings. By default, this value is disabled. To enable this feature, simple configure it by adding the following line to your configuration file:

```
SpecificViews Off
```

IMPORTANT NOTE: Enabling *SpecificViews* in the current version will result in not being able to fulfill a directory listing request of a *Samsung TV*.

### 6.5.3 EnableImageThumbnails

By switching the parameter *EnableImageThumbnails* on or off, you are able to decide if preview thumbnails of images should be displayed on capabale devices. By default, this feature is deactivated.

IMPORTANT NOTE: Enabling *EnableImageThumbnails* might decrease the performance of directory listings.

```
EnableImageThumbnails On
```

### 6.5.4 EnableVideoThumbnails

By switching the parameter *EnableVideoThumbnails* on or off, you are able to decide if preview thumbnails of video files should be displayed on capabale devices. By default, this feature is deactivated.

IMPORTANT NOTE: Enabling *EnableVideoThumbnails* might decrease the performance of directory listings and will require `FFmpeg`.

```
EnableVideoThumbnails On
```

### 6.5.5 LowResourceMode

By default, *pDLNA* opens every single video and audio file with `FFmpeg` to determine its codecs, length and various other attributes. This behaviour is not really IO friendly and so crawling and indexing the media library is slow. Because of small devices, with limited resource like CPU and IO, enabling the *LowResourceMode* increases crawling and indexing the media library.

By default, *LowResourceMode* is disabled.

```
LowResourceMode On
```

IMPORTANT NOTE: Enabling *LowResourceMode* will decrease the usability: No information like codecs, length and various other attributes will be gathered from the media itself. Because of this limited information, directory listings might not be as beautiful as when *LowResourceMode* is disabled.
When *LowResourceMode* is enabled, *pDLNA* will ignore *External* and *Transcode* configuration blocks. *pDLNA* will also **not** index audio or video streams as elements from a playlist. But `FFmpeg` is not required.

### 6.5.6 FFmpegBinaryPath

*pDLNA* has to use `FFmpeg` for transcoding. So it is necessary to configure *FFmpegBinaryPath* with the correct path of `FFmpeg`'s binary. The default path for `FFmpeg`'s binary is set to `/usr/bin/ffmpeg`. For detailed information about `FFmpeg` please see section 2.3.1.

```
FFmpegBinaryPath /usr/bin/ffmpeg
```

## 6.6 Logging

### 6.6.1 LogFile

The *LogFile* configuration parameter defines the logging location of *pDLNA*. Table 6.4 gives an overview about the available configuration options.

IMPORTANT NOTE: Please ensure, that the user, which is running *pDLNA* has permissions to write to the configured path.

If *LogFile* is not specified, *pDLNA* will print all logging output to *STDERR*.

```
LogFile /var/log/pdlna.log
```

| PARAMETER | DESCRIPTION |
|---|---|
| `STDERR` | all logging output will be printed to *STDERR* |
| `SYSLOG` | all logging output will be logged to syslog |
| `<full path to log file>` | specify a full path to a file, *pDLNA* should use as a logfile (like `/var/log/pdlna.log`) |

Table 6.4: Available LogFile configuration parameters

| LOGLEVEL | DESCRIPTION |
|---|---|
| `0` | normal |
| `1` | debug |
| `2` | debug 2 |
| `3` | debug 3 |

Table 6.5: Available LogLevel configuration parameters

### 6.6.2 LogFileMaxSize

*pDLNA* is able to keep track of the logfile's size and clear it, if the size exceeds a value specified by *LogFileMaxSize* in Megabytes. The configured value has to be **greater than 0 and less than 100**.

By default, *LogFileMaxSize* is set to 10 Megabytes.

```
LogFileMaxSize 10
```

### 6.6.3 LogLevel

*pDLNA* is capable to differentiate between different kind of log messages by their *LogLevel*. Specifying a higher *LogLevel* will result in more detailed logging messages including the messages of the lower *LogLevels*. Please see table 6.5 for detailed information about the different *LogLevel*

If *LogLevel* is **not** specified, it will be set to 0.

```
LogLevel 1
```

### 6.6.4 LogCategory

*pDLNA* is capable to differentiate between different kind of log messages by their *LogCategory*. To activate log messages of one of the *LogCategory* listed in table 6.6, simple configure them via a comma seperated list, like in the following example, which enables all categories.

```
LogCategory discovery,httpdir,httpstream,library,httpgeneric,
 database,transcoding,soap
```

By default, only some generic messages will be logged and **none** of these categories are enabled.

| LogCategory | Description |
|---|---|
| `discovery` | SSDP messages |
| `httpdir` | messages from the directory listings via HTTP |
| `httpstream` | messages from the streaming via HTTP |
| `library` | messages from building the media library |
| `httpgeneric` | generic HTTP messages |
| `database` | Database log messages, like queries and so on |
| `transcoding` | Transcoding log messages |
| `soap` | SOAP log messages |

Table 6.6: Available LogCategory configuration parameters

| Variables | Description |
|---|---|
| `%m` | number of month |
| `%d` | numeric day of the month |
| `%H` | hour, 24 hour clock |
| `%I` | hour, 12 hour clock |
| `%p` | AM or PM |
| `%M` | minute |
| `%S` | second |
| `%s` | seconds since the epoch, UCT (aka unixtimestamps) |
| `%o` | ornate day of month – 1st, 2nd, 25th, etc. |
| `%Y` | year |
| `%Z` | timezone in ascii. eg: PST |
| `,-_:` and spaces | characters to format the date string |

Table 6.7: Available DateFormat configuration variables

### 6.6.5 DateFormat

By configuring *DateFormat*, you are able to define the format of time and date in log or debug messages. The following table 6.7 shows valid variables, their value and other valid characters to format the *DateFormat* string.

If *DateFormat* is **not** specified, it will be set to `%Y-%m-%d %H:%M:%S`, which will result for instance in `2012-12-21 13:37:00`.

```
DateFormat '%Y-%m-%d %H:%M:%S'
```

## 6.7 Media Configuration

### 6.7.1 RescanMediaInterval

By configuring *RescanMediaInterval*, you are able to define the interval, when *pDLNA* recrawls the configured media directories (including found playlist files) and external media items.

The following table 6.8 shows valid variables and their descriptions to configure *RescanMediaInterval*.

If *RescanMediaInterval* is **not** specified, it will be set to `daily`.

| RescanMediaInterval | Description |
|---|---|
| `never` | never recrawl the media directories |
| `hourly` | media library will be marked as expired after 60 minutes |
| `halfdaily` | media library will be marked as expired after 12 hours |
| `daily` | media library will be marked as expired after 24 hours |

Table 6.8: Available RescanMediaInterval configuration variables

| Parameter | Information | Description |
|---|---|---|
| `MediaType` | obligatory | specify the type of media files, which should be crawled: `video`, `audio`, `image`, `all` |
| `Recursion` | optional, by default: `yes` | specify if these directories should be crawled recursively (by `yes` or `no`) |
| `ExcludeDirs` | optional | exclude a comma seperated list of directory names from being crawled and added to the media library |
| `ExcludeItems` | optional | exclude a comma seperated list of file names from being crawled and added to the media library |
| `AllowPlaylists` | optional, by default: `no` | enable AllowPlaylists configuration parameter to initialize playlist files |

Table 6.9: Available Directory block configuration parameters

```
RescanMediaInterval daily
```

### 6.7.2 Directory

By default and for reasons of data privacy, *pDLNA* will **not** scan automatically for media files and will **not** add them automatically to the media library.

Therefore you need to configure your media directories in the configuration file in a configuration block called `<Directory>`. Each of these blocks can be configured separately with parameters, which are described in table 6.9.

Additionally, table 6.10 gives an overview about the different media types and their mime types.

The following configuration block will crawl the directory `/media/video/` **recursively** for only **video** files.

```
<Directory "/media/video/">
  MediaType      video
  Recursion      yes
</Directory>
```

| MediaType | MimeTypes |
|---|---|
| video | video/x-msvideo, video/x-matroska, video/mp4, video/mpeg, video/x-flv |
| audio | audio/mpeg, audio/mp4, audio/x-ms-wma, audio/x-flac, audio/x-wav, video/x-theora, audio/ac3, audio/x-aiff |
| image | image/jpeg, image/gif |
| Playlist | audio/x-scpls, audio/x-mpegurl, application/vnd.apple.mpegurl, audio/x-ms-asx, video/x-ms-asf, application/xspf+xml |

Table 6.10: Supported MimeTypes

The next configuration snippet will crawl the directory `/media/music/` **recursively** for only **music** files. Additionally *pDLNA* excludes directories which are called `Justin Bieber` or `Lady Gaga` and also ignores files with the following names: `justin_bieber.mp3` or `lady_gaga.mp3`. Additionally, if *pDLNA* finds a **Playlist file**, *pDLNA* will also add included streams to the media library[2].

```
<Directory "/media/music/">
  MediaType      audio
  ExcludeDirs    "Justin Bieber,Lady Gaga"
  ExcludeItems   "justin_bieber.mp3,lady_gaga.mp3"
  AllowPlaylists true
</Directory>
```

The third configuration example will crawl the directory `/media/images/` **recursively** for only **images**.

```
<Directory "/media/images/">
  MediaType      image
</Directory>
```

And the last snippet crawls the directory `/media/misc/` **not recursively** for all sort of media files but not for **Playlist files**.

```
<Directory "/media/misc/">
  MediaType      all
  Recursion      no
  AllowPlaylists off
</Directory>
```

### 6.7.3 External

Additionally you are able to add external media items to the media library. Therefore there are currently two kind of external media items:

[2]For detailed information regarding supported streaming URLs please see section 6.7.3.

- scripts
- streams

These configured external media items will be added to the root directory of the media library.

A valid configuration block for a script has to look like the following example. Since *pDLNA* is not able to determine the type of the media, which will be served by the script, it is needed to define the media type by the *MediaType* parameter.

```
<External "mpegstream">
  Executable external_programs/mpegstream_src.pl
  MediaType  video
</External>
```

For configuring a stream as an external media item, a valid configuration block looks like the following example. Therefor the only parameter needed is the `StreamingURL` parameter with a valid streaming URL[3].

```
<External "FM4">
  StreamingURL http://mp3stream1.apasf.apa.at:8000/
</External>
```

IMPORTANT NOTE: External media items are not available, when `LowResourceMode` is enabled.

**Streams**

Currently *pDLNA* only supports `http://` and `mms://` streams.

## 6.8 Transcoding Profiles

Since not all *DLNA* aware devices support all different kinds of audio or video codecs, transcoding these media files allows playing these files on these devices anyway.

Table 6.11 gives an overview about the currently tested and supported transcoding profiles for audio files. The green colored table cells in this table show the currently supported and tested transcoding possiblites for audio items.

The following configuration snippet will transcode all audio files, which are *vorbis* encoded audio data to a *flac* encoded audio stream **on condition that** the *DLNA* client IP address matches the comma seperated list of the configured `ClientIPs` parameter.

[3]For detailed information regarding supported streaming URLs please see section 6.7.3.

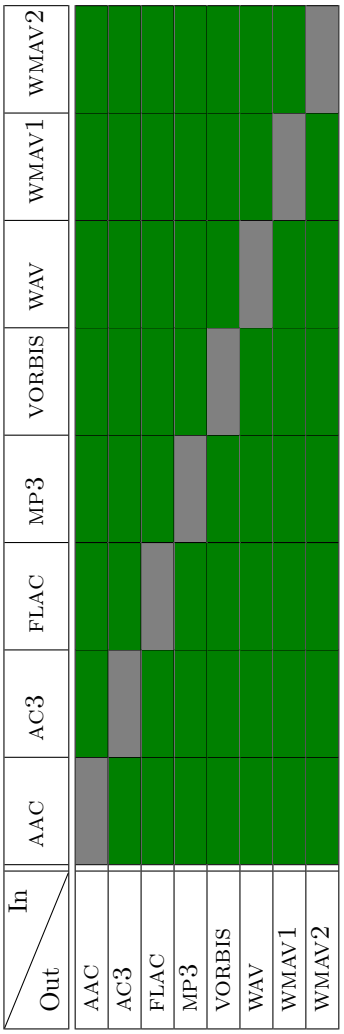



Table 6.11: Supported Audio Transcoding Profiles

| Transcoded Codec | FFmpeg audio codec | FFmpeg format |
|---|---|---|
| AAC | libfaac | m4v |
| AC3 | ac3 | ac3 |
| FLAC | flac | flac |
| MP3 | libmp3lame | mp3 |
| VORBIS | libvorbis | ogg |
| WAV | pcm_s16le | wav |
| WMAV1 | wmav1 | asf |
| WMAV2 | wmav2 | asf |

Table 6.12: Required encoding audio codecs and formats by your FFmpeg installation

```
<Transcode "ogg2flac">
  MediaType      audio
  AudioCodecIn   vorbis
  AudioCodecOut  flac
  ClientIPs      192.168.0.1,192.168.0.128/25
</Transcode>
```

Since it is possible to define multiple transcoding profiles, the first transcoding profile, which matches the criteria, will be used. So please ensure to configure the more specific on the top of the transcoding profiles.[4]

IMPORTANT NOTE: Transcoding depends on the supported audio and video codecs of your `FFmpeg` installation and some validation in the source code to ensure its functionality. Table 6.12 gives an overview regarding the required `FFmpeg` encoding audio codecs and formats to encode audio files.

IMPORTANT NOTE: Transcoding is not available, when `LowResourceMode` is enabled.

An easy way to get the information, which transcoding profiles are supported by your `FFmpeg` installation, is by using the WebUI (see chapter 7).

[4]This functionality can be compared with a firewall rule set.

# Chapter 7

# WebUI

*pDLNA* includes a small WebUI, which is accessable via the following URL `http://ListenIPAddress:HTTPPort/webui/`.

IMPORTANT NOTE: If your client's IP address, which you would like to use to access the WebUI, is not included in the *AllowedClients* configuration parameter, you will not be able to access it.

This small WebUI gives the chance to navigate through the indexed media items, have a detailed look on connected *UPnP* and *DLNA* devices and determine some process information like memory usage, the amount and size of indexed media items in the database or even the supported codecs of your `FFmpeg` installation.

The WebUI does have a navigation part (coloured in blue) on the left side of the website. The information itself will be shown on the right side (coloured in white). The navigation is split in three different sections, which will be described in the following three sections.

## 7.1 Configured media

The screenshot in figure 7.1 shows the landing page of the WebUI, which show the root directory of the media library. It also shows the external media items, which are configured in the configuration file (see section 6.7.3). In the navigation of the WebUI, in the *Configured media* section, all in the configuration file configured directories (see section 6.7.2) are listed. The number in the brackets after the directory name gives the information, how much items are in this specific directory.

By clicking (in this case) on the *media* directory, the directory tree will be shown as it appears in figure 7.1. With an additional click on the *audio* directory, the files, which are in this directory will be shown in the content part of the WebUI. The content table, as can be seen, shows the information about the filename, the size and the date of the media file.




Figure 7.1: WebUI: Landing page




Figure 7.2: WebUI: Directory listing

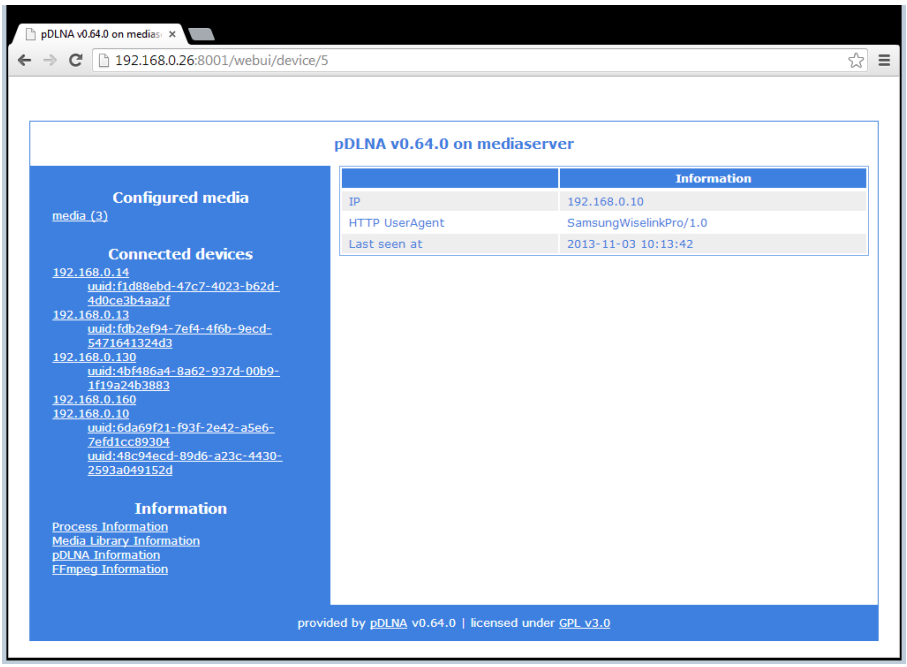



Figure 7.3: WebUI: Information about a connected device per IP address

## 7.2 Connected devices

The second section in the navigation of the WebUI lists all discovered IP addresses, which are part of your network and do interact somehow with the running *pDLNA* installation. This includes normal clients, which access the WebUI, other *UPnP* devices, *DLNA* aware TVs or even other *DLNA* media servers.

In the navigation, a list of IP addresses is shown. Additionally, all discovered *UPnP* services, which are served by a specifc IP address are also shown in the navigation as subitems.

Figure 7.2 shows in the content part the information after clicking on the IP address *192.168.0.130* in the navigation. This information includes the IP address itself, the *HTTP UserAgent*, which was used by the device and when communication from the IP address was seen last.

Figure 7.2 shows information, including the friendly name, the device type or the device description URL, after clicking on a service UDN.

## 7.3 Information

The third section in the navigation includes some various information about the running*pDLNA* installation.

While figure 7.3 shows information about the process itself (including for instance memory and cpu usage), figure 7.3 delivers information about the amount and size of media items included in the media library.

Additionally, the following figure 7.3 shows information about the installed

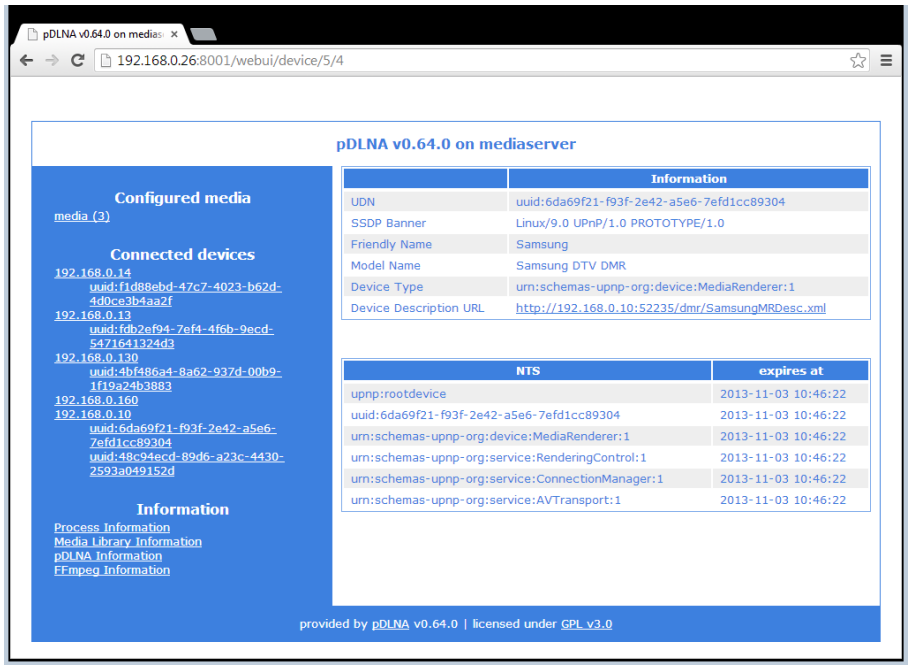



Figure 7.4: WebUI: Information about a connected device per service

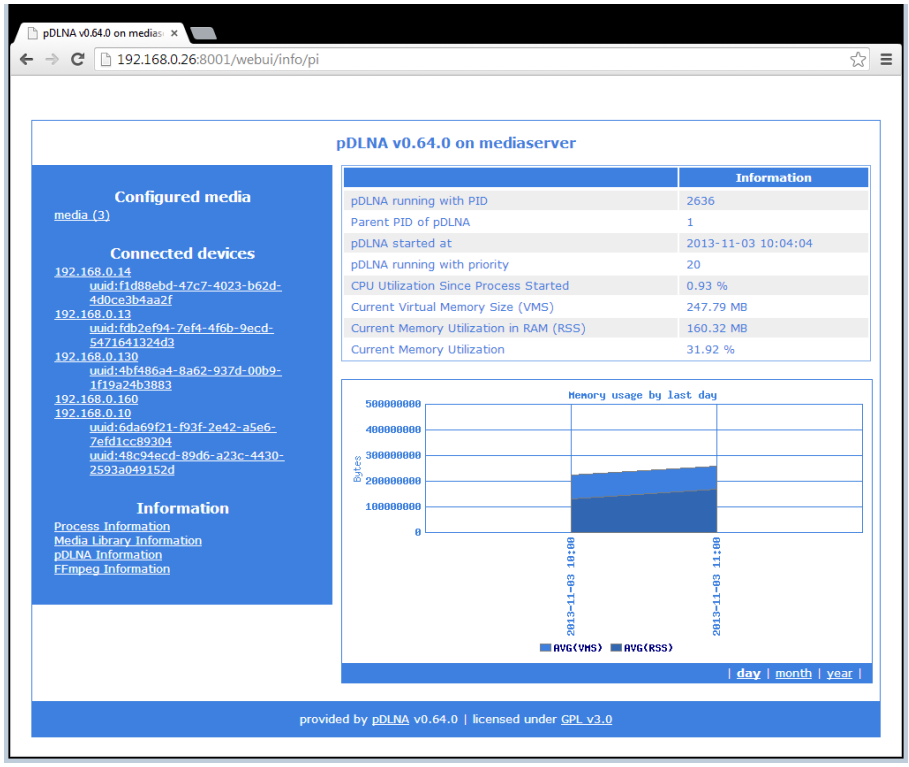



Figure 7.5: WebUI: Process Information

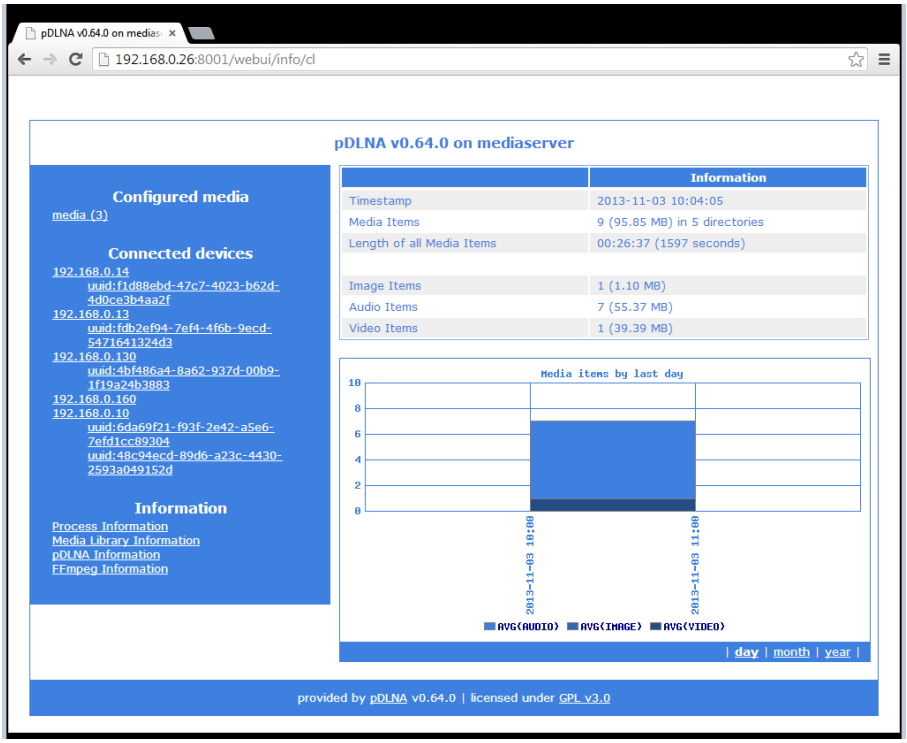



Figure 7.6: WebUI: Media Library Information

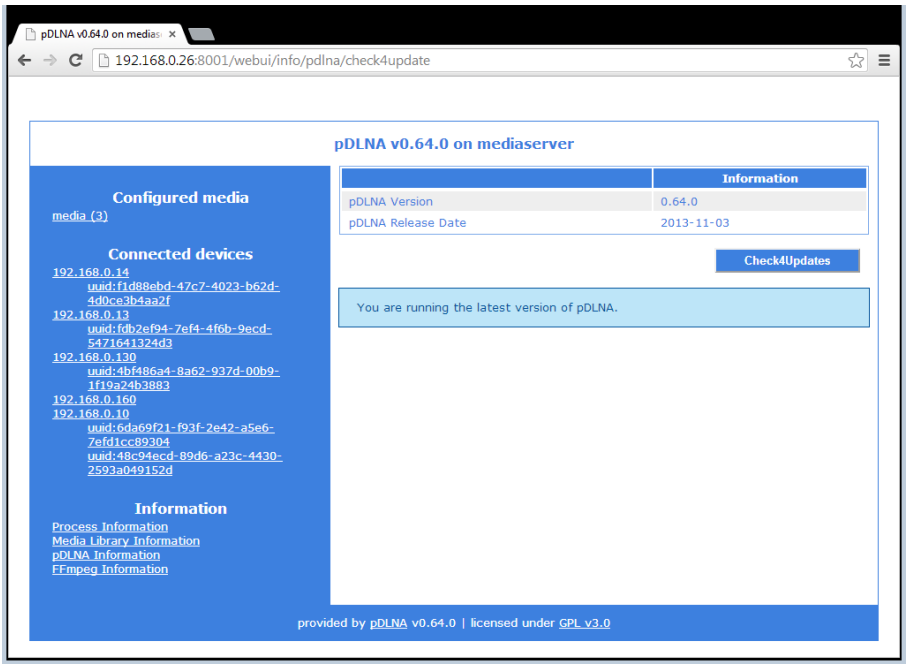



Figure 7.7: WebUI: pDLNA Information

*pDLNA* version and its release date. You are also able to execute a *Check4Updates*, which works as described in section 6.1.2, but shows the result in the WebUI.

And finally, the following figure 7.3 shows information regarding the used `FFmpeg` installation, if the usage of `FFmpeg` is configured. This information provides easier access to supported transcoding profiles (see section 6.8).

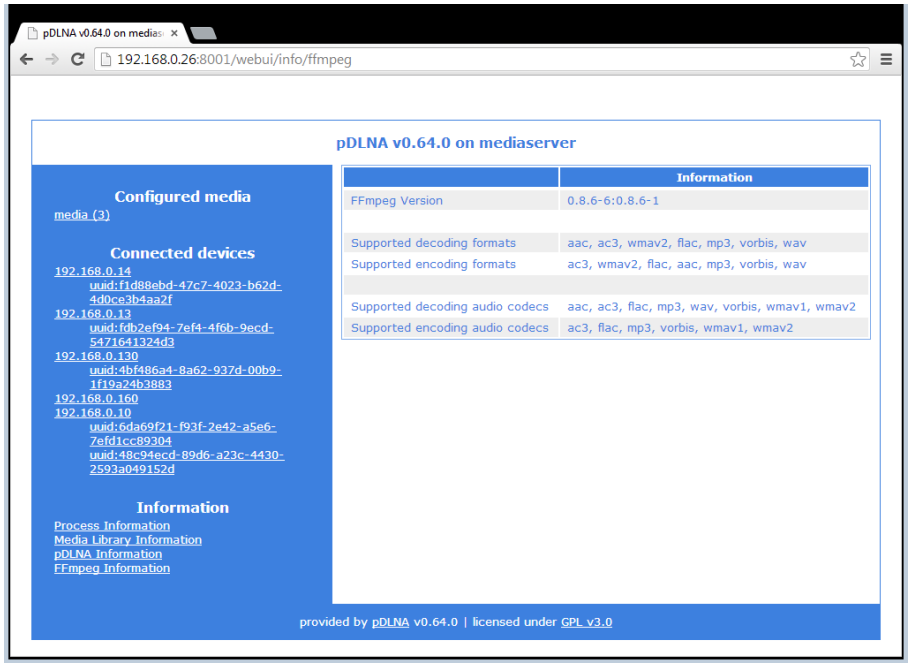



Figure 7.8: WebUI: FFmpeg Information

# Chapter 8

# Preinstalled VMware images

Since not everybody is willing to install *pDLNA* on a system, just to verify if *pDLNA* fullfills his needs or just likes to test *pDLNA* for the first time, there are some preinstalled *VMware* images available to download from the *pDLNA* website available under the following URL: `www.pdlna.org/cgi-bin/index.pl?menu=vm`

In the case, that somebody does not know *VMware*: VMware Player is free for personal use and is available to download via the following URL: `www.vmware.com/products/player/overview.html`

You also might be able to convert the available *VMware* images and open it with *VirtualBox*[1] or any other virtualization product.

IMPORTANT NOTE: These preinstalled *VMware* images are prepared to have a quick look on *pDLNA* and are not designed to be used in a productive environment or for any other purpose.

All preinstalled *VMware* images are configured to use SQLite3 as a database.

While the following section 8.1 gives an overview about the general information, like installation of the operating system or the included media files, section 8.2 describes some per operating system relevant information.

## 8.1 General information

All the available preinstalled *VMware* images are configured the same and do have the following hardware applied:

- 512 megabytes of memory
- harddisk with 10 gigabytes capacity
- one network card which is bridged

[1] *VirtualBox* is an open source virtualization product, which is available on the product website `www.virtualbox.org`.

Afterwards the operating system has been installed, **english** as language is chosen. Additionally the timezone is set to **Central European Summer Time (CEST)** and the keyboard layout is set to **english** too. Finally, if a package management tool is used by the operating system itself, a mirror from **Austria** is configured. If these settings does not match your needs, please feel free to modify these settings. The following section 8.2 gives you some short information how to change some of these configurations.

With the installation, the password for the superuser **root** is set to **pdlna**. Additionally, another user **pdlna** with the password **pdlna** is created. The networkcard is configured to aquire an IP address via *DHCP*. Also a *SSH* server for remote access and a *NTP* server to time synchronization is installed.

Afterwards, all the necessary packages and Perl modules were installed like described int section 3.1. And finally, *pDLNA* was installed via a `git clone`, by the user **pdlna** to the directory `/home/pdlna/pDLNA/`, which is described in section 3.2. So the configuration file is stored in `/etc/pdlna.conf` and the initscript to start and stop *pDLNA* is stored in `/etc/init.d/rc.pDLNA` or `/etc/rc.d/rc.pDLNA restart`.

IMPORTANT NOTE: Every preinstalled *VMware* image will not start the *pDLNA* installation automatically at startup. So please start *pDLNA* by executing `/etc/init.d/rc.pDLNA start` as a superuser.

Because of installing *pDLNA* via `git`, updating the *pDLNA* version of an already downloaded *VMware* image can be done, by logging in as the user **pdlna**, changing to the directory `/home/pdlna/pDLNA/` and executing the command `git pull`. And after executing `/etc/init.d/rc.pDLNA restart` or `/etc/rc.d/rc.pDLNA restart` as a superuser, the new version of *pDLNA* should be up and running again. This is described in detail in section 3.2.3.

### 8.1.1 Included media files

Each *VMware* image comes with the same media files (to test the functionality), which are licensed under the *Creative Commons* licence[2]. In each *VMware* image, there is a `DISCLAIMER.txt` file in the directory `/home/pdlna/media/`, which lists the included media files, their source and their exact *Creative Commons* licence.

The following list also lists the included media files, their source and their exact *Creative Commons* licence:

- `/home/pdlna/media/audio/Bryyn_-_Giraffe.mp3`
  - Source: `www.jamendo.com/de/track/725574/giraffe`
  - License: `creativecommons.org/licenses/by-nc-sa/3.0/`
- `/home/pdlna/media/audio/Fhernando_-_Ride_My_Tempo.mp3`
  - Source: `www.jamendo.com/de/track/944721/ride-my-tem`
  - License: `creativecommons.org/licenses/by-nc-sa/3.0/`

[2] `www.creativecommons.org`

- /home/pdlna/media/audio/Manolis_Moumouzias_-_In_Love.mp3
  - Source: www.jamendo.com/de/track/917546/in-love
  - License: creativecommons.org/licenses/by-nc-sa/3.0/
- /home/pdlna/media/images/Two-toed_sloth_Costa_Rica_-_cropped.jpg
  - Source: upload.wikimedia.org/wikipedia/commons/8/8b/Two-toed_sloth_Costa_Rica_-_cropped.jpg
  - License: creativecommons.org/licenses/by/2.5/deed.de
- /home/pdlna/media/video/episode2.0_xvid.avi
  - Source: archive.org/details/welcometothescene_version2.0_xvid
  - License: creativecommons.org/licenses/by-nc-nd/3.0/
- /home/pdlna/media/video/episode2.0_xvid.mp4
  - Source: archive.org/details/welcometothescene_version2.0_xvid
  - License: creativecommons.org/licenses/by-nc-nd/3.0/

The included media files are chosen by random.

IMPORTANT NOTE: If you are the copyright owner or an agent thereof and do not want your creation to be distributed like this, please contact me and I will remove your creation.

## 8.2 Specific Information

The following sections describe some specific settings, which are made to the different preinstalled *VMware* images and also describes how to change some of the settings like the timezone or the keyboard layout.

### 8.2.1 CentOS 6

IMPORTANT NOTE: Since FFmpeg is not part of the package management in CentOS 6, *pDLNA* is only able run in *LowResourceMode* in this preinstalled *VMware* image.

#### Logon customization

The logon customization, as can be seen in figure 8.2.1, is handeled by the script /etc/rc.local, which was modified to call /usr/local/bin/update-issue.sh, which updates the /etc/issue and /etc/issue.net files. The following three lines have also been added to this file, to delete sensitive data on startup:

```
/bin/rm -f /var/log/pdlna.log
/bin/rm -f /tmp/pdlna.db
/bin/rm -f /var/run/pdlna.pid
```

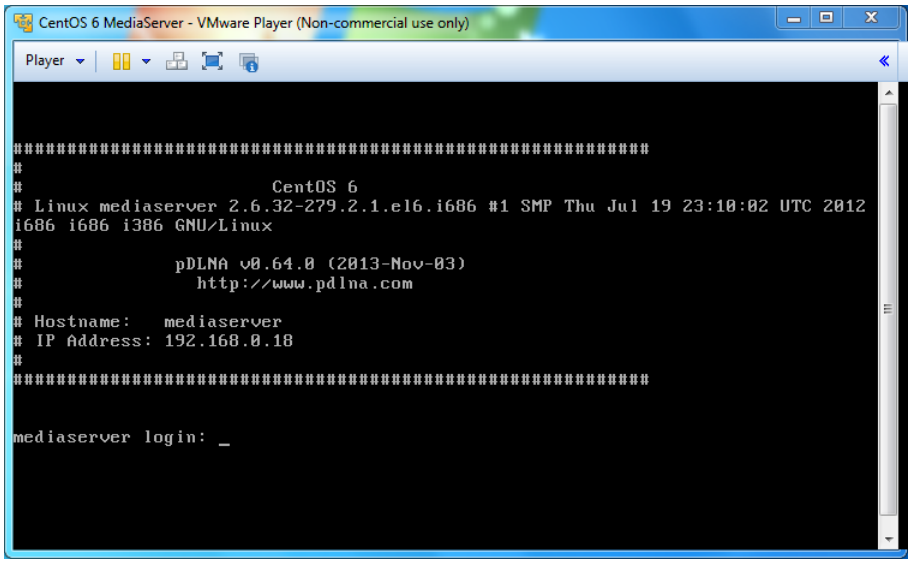



Figure 8.1: Boot-up screen of preinstalled CentOS 6 VMware image

Additionally, the `OpenSSH` server's configuration file (`/etc/ssh/sshd_config`) has been modified to use the `/etc/issue.net` file as a banner:

```
Banner /etc/issue.net
```

The `/usr/local/bin/update-issue.sh` script can also be found on GitHub (`github.com/geuma/pDLNA-utils/blob/master/preinstalled-VMs/CentOS6/update-issue.sh`).

### Changing the timezone

In CentOS 6, all the available timezones are stored in the directory `/usr/share/zoneinfo/`. Because of this, changing the timezone can be done via modifying the sysmlink for `/etc/localtime` to the corresponding timezone file. An example is listed in the following snippet:

```
pdlna@mediaserver:~$ sudo ln -sf /usr/share/zoneinfo/EST
  /etc/localtime
```

### Changing the keyboard layout

In CentOS 6, all the available keyboard layouts are stored in the directory `/lib/kbd/keymaps/i386`. Changing the keyboard layout can be done via modifying the `KEYTABLE` key in the file `/etc/sysconfig/keyboard` by opening the file with your favourite file editor:

```
pdlna@mediaserver:~$ sudo vi /etc/sysconfig/keyboard
```

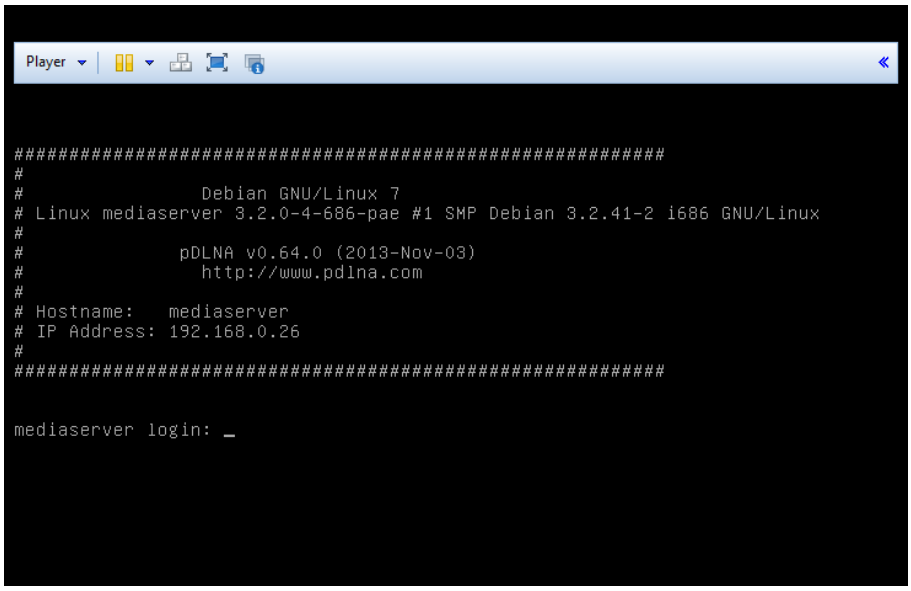



Figure 8.2: Boot-up screen of preinstalled Debian 7 VMware image

Additional documentation can be found here: `www.centos.org/docs/5/html/5.1/Deployment_Guide/s2-sysconfig-kybd.html`

### 8.2.2 Debian GNU/Linux 7

#### Logon customization

The logon customization, as can be seen in figure 8.2.2, is handeled by the script `/etc/rc.local`, which was modified to call `/usr/local/bin/update-issue.sh`, which updates the `/etc/issue` and `/etc/issue.net` files. The following three lines have also been added to this file, to delete sensitive data on startup:

```
/bin/rm -f /var/log/pdlna.log
/bin/rm -f /tmp/pdlna.db
/bin/rm -f /var/run/pdlna.pid
```

Additionally, the `OpenSSH` server's configuration file (`/etc/ssh/sshd_config`) has been modified to use the `/etc/issue.net` file as a banner:

```
Banner /etc/issue.net
```

The `/usr/local/bin/update-issue.sh` script can also be found on GitHub (`github.com/geuma/pDLNA-utils/blob/master/preinstalled-VMs/Debian7/update-issue.sh`).

#### Changing the timezone

Changing the timezone in Debian GNU/Linux 7 can be done by executing the following command as a superuser and use its wizard to configure the correct

timezone.

```
pdlna@mediaserver:~$ sudo dpkg-reconfigure tzdata
```

#### Changing the keyboard layout

Changing the keyboard layout is as easy as changing the timezone. Just execute the following command as a superuser and use its wizard to configure the correct keyboard layout.

```
pdlna@mediaserver:~$ sudo dpkg-reconfigure keyboard-configuration
```

### 8.2.3 FreeBSD 9

#### Logon customization

The logon customization, as can be seen in figure 8.2.3, is handeled by the script `/etc/rc.local`, which was created to call `/usr/local/bin/update-issue.sh`, which updates the `/etc/issue` and `/etc/issue.net` files and delete sensitive data. The script `/etc/rc.local` has been created with the following content and the executable bit has been set. By default, FreeBSD 9 executes this script at startup via `/etc/rc.d/local`.

```
#!/bin/sh

/usr/local/bin/update-issue.sh
/bin/rm -f /var/log/pdlna.log
/bin/rm -f /tmp/pdlna.db
/bin/rm -f /var/run/pdlna.pid
```

Since the configuration file `/etc/gettytab` is already configured to use the `/etc/issue` file as a banner, no configuration to `getty` was needed.

Additionally, the `OpenSSH` server's configuration file (`/etc/ssh/sshd_config`) has been modified to use the `/etc/issue.net` file as a banner:

```
Banner /etc/issue.net
```

The `/usr/local/bin/update-issue.sh` script can also be found on GitHub (`github.com/geuma/pDLNA-utils/blob/master/preinstalled-VMs/FreeBSD9/update-issue.sh`).

#### Changing the timezone

In FreeBSD 9, all the available timezones are stored in the directory `/usr/share/zoneinfo`. Because of this, changing the timezone can be done via copying the corresponding timezone file to `/etc/localtime`. An example is listed in the following snippet:

```
[pdlna@mediaserver ~]$ sudo copy /usr/share/zoneinfo/EST
  /etc/localtime
```

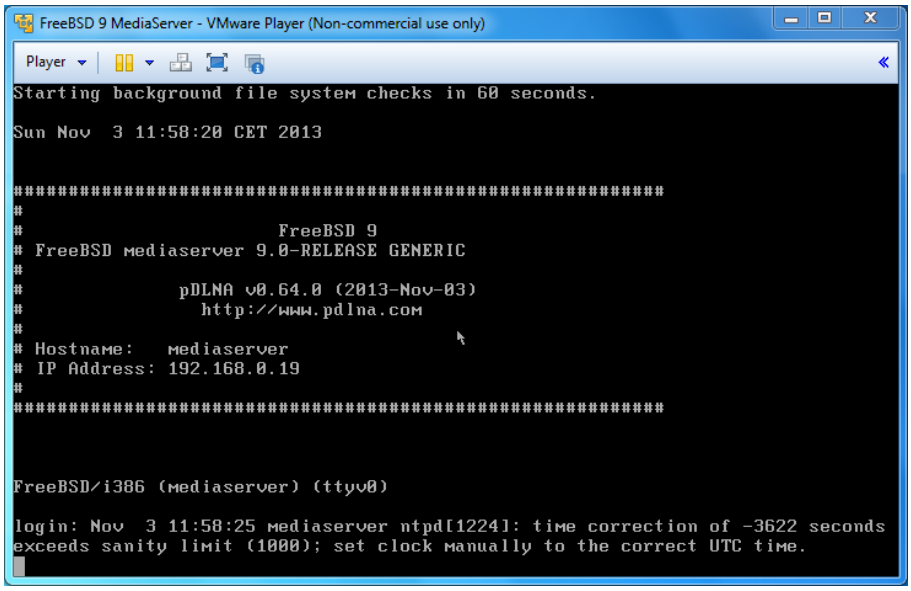



Figure 8.3: Boot-up screen of preinstalled FreeBSD 9 VMware image

### Changing the keyboard layout

Changing the keyboard layout is as easy as changing the timezone. Just execute the following command as a superuser and use its menu to configure the correct keyboard layout.

```
[pdlna@mediaserver ~]$ sudo kbdmap
```

If you would like to change the keyboard layout in a persistent way, modify the file `/etc/rc.conf`.

# Chapter 9

# Debugging and reporting problems

This is not (yet) a real troubleshooting guide to *pDLNA*. Currently it is more a small handbook to do some general debugging and how to gather some necessary information on a running Linux operating system. You are also able to forward this information to me.

## 9.1 Debugging

When starting *pDLNA*, the process will parse the configuration file[1] and will not start until everything is configured correctly. In some cases *pDLNA* might not be able to determine the correct *ListenInterface*, where you need to configure those information by hand. The following command starts *pDLNA*.

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

Once *pDLNA* is running successfully, you are able to verify this by running the following two commands. The first one checks all the running processes for the *pDLNA* process, while the second one prints out the stored PID. If both PID match each other, everything should be fine.

```
pdlna@mediaserver:~$ sudo ps -ef | grep pDLNA
root     13162     1  3 09:18 pts/2    00:00:50
  /usr/bin/perl ./pDLNA.pl -f /etc/pdlna.conf
```

```
pdlna@mediaserver:~$ sudo cat /var/run/pdlna.pid
13162
```

The following listings are based on the network configuration listed in 6.3. So, the next step to identify a lowlevel network problem of *pDLNA* will be to check if *pDLNA* is listening to the two necessary network ports. On the one

[1] Please see chapter 6 for more details.

hand, there is **port 1900 UDP** needed for the *SSDP* communication and **port 8001 TCP**[2] for the *DLNA* communication via HTTP. The following command checks the network statistics and filters for the lines including the PID of the running *pDLNA* installation.

```
pdlna@mediaserver:~$ sudo netstat -taunp | grep 13162
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address
 Foreign Address         State       PID/Program name
tcp        0      0 192.168.145.139:8001
 0.0.0.0:*               LISTEN      13162/perl
udp        0      0 192.168.145.139:37302
 239.255.255.250:1900    ESTABLISHED 13162/perl
udp    53368      0 0.0.0.0:1900
 0.0.0.0:*                           13162/perl
```

If your network statistics output shows, that *pDLNA* is correctly listening to **port 1900 UDP** and the configured *HTTPPort*, the network communication should be working.

IMPORTANT NOTE: *AllowedClients* is configured to the local subnet by default.

IMPORTANT NOTE: Please also check your firewall configuration.

As a simple test you can visit the WebUI (see chapter 7), which can be accessed by the following URL: `http://ListenIPAddress:HTTPPort/webui/`. If you are receiving an *HTTP Error Code 403*, your host is not configured as an *AllowedClients*. If your Browser is running in a *timeout*, there might be a problem with your configuration.

After checking the general process and network information, the next step is about increasing the *LogLevel* and configuring the necessary *LogCategory* from table 6.6. At first you need to configure *LogLevel* to the highest *LogLevel* available in table 6.5 for the most detailed log messages.

### 9.1.1 My device is not able to discover *pDLNA*

If your *DLNA* capable device is not able to discover *pDLNA*, please set the configuration parameter *LogCategory* to `discovery, httpgeneric` and restart *pDLNA*. Additionally you are able to do a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -i ListenInterface \
 -s 1500 -w capture_full.pcap
```

Please ensure to fill in the configured *ListenInterface*.

[2]Please ensure that this port can be changed in the configuration file and *8001* is the default one.

### 9.1.2 My media directories are not read correctly

In the case, that *pDLNA* was not able to read in your shared directories correctly, enable the *LogCategory* `library` in the configuration file and restart the installed version of *pDLNA*. An easy way to navigate quickly through the media library and check for problems is by using the WebUI (see chapter 7).

For a detailed look on the structure and contents of the media library, check your database.

### 9.1.3 My device is not able to list the directories/items shared by *pDLNA*

If browsing the shared media directories is not working properly by your *DLNA* aware devices, configure the *LogCategory* to `httpgeneric, httpdir`, restart *pDLNA* and do a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -p HTTPPort -s \
 1500 -w capture_http.pcap
```

Please ensure to fill in the configured *HTTPPort* (by default set to `8001`).

For a detailed look on the structure and contents of the media library, check your database.

### 9.1.4 *pDLNA* is not able to stream media items to my device

In any case, that streaming of videos, music or images is not working properly, please set the *LogCategory* parameter to `httpgeneric, httpstream`, restart *pDLNA* and start a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -p HTTPPort \
 -s 1500 -w capture_http.pcap
```

Please ensure to fill in the configured *HTTPPort* (by default set to `8001`).

## 9.2 Reporting problems

If you are not able to fix the problem or just want me to take a closer look, please supply all mentioned information, like network statistics, packet captures, logs and so on.

In case of having a *DLNA* aware device, which is not working properly with *pDLNA* please supply a full packet capture, including the whole *SSDP* and *DLNA* packets of this specific device communicating with another digital media server (which is working properly). In this case I might be able to determine the differences and supply a new version of *pDLNA* which is supporting this device.

# Chapter 10

# Known Issues

## 10.1 FreeBSD: initscript unable to completely shut down pDLNA

### 10.1.1 Description

On FreeBSD, there seems to be a problem shutting down *pDLNA* via the included initscript[1]. After stopping and starting *pDLNA*, the following error message will occur:

```
[root@mediaserver ~]# /etc/rc.d/rc.pDLNA start
Starting pDLNA ...
[root@mediaserver ~]# /etc/rc.d/rc.pDLNA restart
Starting pDLNA ...
[root@mediaserver ~]# Cannot bind to Multicast socket:
 Address already in use
Going to terminate pDLNA/v0.63.0 on freebsd/9.0-release
 with FriendlyName 'pDLNA v0.63.0 on mediaserver' ...
```

### 10.1.2 Workaround

Since there is no fix for this issue today, this problem can be handled by a workaround. Get the process ID of the still running *pDLNA* process, kill it, remove the PID file (if any) and use the initscript afterwards again to start *pDLNA*:

```
[root@mediaserver ~]# ps | grep pDLNA
2327   0  I    0:22.46 /usr/bin/perl ./pDLNA.pl -f /etc/pdlna.conf
2651   0  R+   0:00.01 grep pDLNA
[root@mediaserver ~]# kill -9 2327
[root@mediaserver ~]# rm /var/run/pdlna.pid
[root@mediaserver ~]# /etc/rc.d/rc.pDLNA start
Starting pDLNA ...
```

[1] https://github.com/geuma/pDLNA/issues/20

# Chapter 11

# Fixed Issues

## 11.1 DirectoryListings not working when mandatory data fields not defined

IMPORTANT NOTE: This issue has been fixed with *pDLNA* in version 0.64.2.

### 11.1.1 Describtion

Directory listings are not working, when *mandatory* data fields are not defined in the XML `<FILTER>` tag of the browse request from a DLNA media renderer.[1].

### 11.1.2 Workaround

There is no workaround for this issue.

## 11.2 Movie::Info is still used by PDLNA::Media

IMPORTANT NOTE: This issue has been fixed with *pDLNA* in version 0.64.1.

### 11.2.1 Describtion

The `PDLNA::Media` Perl module is still using `Movie::Info` Perl module, even if it is not required any more. So *pDLNA* will not start up if the `Movie::Info` Perl module is not installed[2].

### 11.2.2 Workaround

Install the missing Perl module `Movie::Info`.

[1] `github.com/geuma/pDLNA/issues/24`
[2] `github.com/geuma/pDLNA/issues/23`

## 11.3 parsing problem if media file name contains special regex characters like '(', ')' or others

IMPORTANT NOTE: This issue has been fixed with *pDLNA* in version 0.64.0.

### 11.3.1 Describtion

This issue has been reported on a `Raspberry Pi` running Linux with Perl 5.14.2 and locales configured to `de_DE.UTF-8`.

The following error appears, when files with special characters (for instance brackets) are existing, which break some regular expressions in *pDLNA*[3]:

```
Issuing rollback() due to DESTROY without explicit disconnect()
 of DBD::SQLite::db handle dbname=/media/TREK_2/Audio/pdlna.db at
 ./pDLNA.pl line 63.
Thread 1 terminated abnormally: Unmatched ) in regex; marked by
 <-- HERE in m/^18-This Live I) <-- HERE 'm Living.mp3$/ at
 PDLNA/ContentLibrary.pm line 152.
```

### 11.3.2 Workaround

There is no workaround for this issue. Or would you like to rename your files?

## 11.4 Use of uninitialized value $request_line in string ne at /PDLNA/HTTPServer.pm line xxx.

IMPORTANT NOTE: This issue has been fixed with *pDLNA* in version 0.62.0.

### 11.4.1 Describtion

Apparently *pDLNA* sometimes starts to print, for unknown reasons, the following warning message to `STDOUT` infinitely[4]:

```
Use of uninitialized value $request_line in string ne at
  /PDLNA/HTTPServer.pm line xxx.
```

[3] `github.com/geuma/pDLNA/issues/21`
[4] `github.com/geuma/pDLNA/issues/13`

### 11.4.2 Workaround

Since, this message is only a warning message, it does not affect the functionality of *pDLNA*. But actually, it increases the system load. A restart of *pDLNA* will fix the problem temporarily, but the issue might take place again.

## 11.5 AllowedClients auto detection on FreeBSD is defect

IMPORTANT NOTE: This issue has been fixed with *pDLNA* in version 0.61.0.

### 11.5.1 Describtion

Apparently *pDLNA* dies with the following error message on *FreeBSD* if *AllowedClients* auto detection is enabled[5]:

```
AllowedClients(ip_address_sv) at PDLNA/Config.pm line xxx.
```

This seems to be a problem with the usage of `AllowedClients(ip_address_sv)` on *FreeBSD*.

### 11.5.2 Workaround

As a workaround you are able to define the *AllowedCLients* configuration parameter in the configuration file. For detailed information see section 6.3.4.

[5] `github.com/geuma/pDLNA/issues/4`