
mumble-ruby-pluginbot Documentation

Release master

Apr 22, 2017

1	Contents	3
1.1	Features	3
1.2	Compatibility	4
1.3	Usage	4
1.3.1	Basic overview	4
1.3.2	Usage examples	5
1.3.2.1	Example 1 – Volume control	5
1.3.2.2	Example 2 – Change state	5
1.3.2.3	Example 3 – Listen to a radio station	5
1.3.2.4	Example 4 – Download music from Youtube	6
1.4	Plugins	6
1.4.1	General information	6
1.4.2	Plugin specific information	7
1.4.2.1	Bandcamp	7
1.4.2.2	Control	7
1.4.2.3	Ektoplazm	7
1.4.2.4	Idle	7
1.4.2.5	Messages	8
1.4.2.6	Mixcloud	8
1.4.2.7	Mpd	8
1.4.2.8	Null	8
1.4.2.9	Radiostream	8
1.4.2.10	Soundcloud	8
1.4.2.11	Timer	9
1.4.2.12	Version	9
1.4.2.13	Youtube	9
1.5	How to run the bot	9
1.5.1	Installation options	9
1.5.1.1	Option 1: Install it on your own – Installation Howto	9
1.5.1.2	Option 2: Use a VirtualBox Virtual Appliance – Download a Fully set up Mumble Ruby Pluginbot	9
1.5.1.3	Option 3: Use Docker	9
1.5.1.4	Option 4: Preconfigured images for different systems	10
1.5.2	Maintenance	10
1.5.2.1	Automatic update	10
1.5.2.2	Manual update (not recommended)	10

1.6	Installation HowTo	10
1.6.1	How much time will you need?	10
1.6.2	How will it look like?	10
1.6.3	On Arch Linux only: Install and set up system package dependencies	11
1.6.4	On Debian/Ubuntu based Distributions only: Install and set up system package dependencies	11
1.6.5	Create a user which should contain all the relevant bot structures	12
1.6.6	Create all needed directories and subdirectories for MPD and the bot(s)	12
1.6.7	Install and set up ruby and all needed libraries	13
1.6.7.1	Set up a Ruby environment	13
1.6.7.2	Get and build mumble-ruby and ruby-mpd and other dependencies	13
1.6.8	Download and set up celt-ruby and libcelt	14
1.6.9	Download and set up opus-ruby	14
1.6.10	Download and set up mumble-ruby-pluginbot	15
1.6.11	Set up MPD (Music Player Daemon)	16
1.6.11.1	Set up the script to start your bot(s) and MPD instance(s)	16
1.6.12	Install a custom version of youtube-dl	17
1.6.12.1	Only on Arch Linux: Install dependencies for youtube-dl	17
1.6.12.2	Only on Debian/Ubuntu based distributions: Install the dependencies (if ffmpeg is available for your distribution)	17
1.6.12.3	Only on Debian/Ubuntu based distributions (OPTIONAL): Install the dependencies (if ffmpeg IS NOT available for your distribution)	17
1.6.12.4	Login as botmaster	17
1.6.12.5	Install the youtube-dl script	18
1.6.13	Almost done, start your bot(s) for the first time	18
1.6.14	Set up bot to start automatically on system startup	18
1.6.14.1	Start everything automatically – if your system is NOT systemd	18
1.6.14.2	Start everything automatically – if your system is systemd	18
1.6.15	Controlling the bot(s) from your shell	18
1.6.16	Configuration settings of your bot	19
1.6.17	Known problems	19
1.7	Explain the configs	19
1.7.1	Configuration files	19
1.7.2	Default configuration and override configuration	19
1.7.3	Syntax within this help	19
1.7.4	Notes for editing the configuration	20
1.7.5	Default config/config.yml	20
1.7.6	Main settings	21
1.7.6.1	config:version	21
1.7.6.2	debug	21
1.7.6.3	language	21
1.7.6.4	main:tempdir	21
1.7.6.5	main:logfile	22
1.7.6.6	main:ducking	22
1.7.6.7	main:automute_if_alone	22
1.7.6.8	main:stop_on_unregistered	22
1.7.6.9	main:channel_notify	22
1.7.6.10	main:controllable	23
1.7.6.11	main:whitelist_enabled	23
1.7.6.12	main:control:string	23
1.7.6.13	main:control:message:private_only	23
1.7.6.14	main:control:message:registered_only	23
1.7.6.15	main:control:historysize	24
1.7.6.16	main:display:comment:set	24
1.7.6.17	main:user:whitelisted	24

1.7.6.18	main:user:superuser	24
1.7.6.19	main:user:banned	24
1.7.6.20	main:user:bound	25
1.7.6.21	main:certfolder	25
1.7.6.22	main:fifo	25
1.7.6.23	main:logo	25
1.7.6.24	main:timer:ticks	25
1.7.6.25	main:blacklisted_commands	25
1.7.6.26	mumble:use_vbr	26
1.7.6.27	mumble:bitrate	26
1.7.6.28	mumble:host	26
1.7.6.29	mumble:port	26
1.7.6.30	mumble:name	26
1.7.6.31	mumble:password	26
1.7.6.32	mumble:channel	27
1.7.7	Bandcamp plugin settings	27
1.7.7.1	plugin:bandcamp:folder:download	27
1.7.7.2	plugin:bandcamp:folder:temp	27
1.7.7.3	plugin:bandcamp:youtube_dl:path	27
1.7.7.4	plugin:bandcamp:youtube_dl:options	27
1.7.7.5	plugin:bandcamp:youtube_dl:prefixes	27
1.7.7.6	plugin:bandcamp:to_mp3	28
1.7.8	Ektoplazm plugin settings	28
1.7.8.1	plugin:ektoplazm:prefixes	28
1.7.8.2	plugin:ektoplazm:folder:download	28
1.7.8.3	plugin:ektoplazm:folder:temp	28
1.7.9	Idle plugin settings	28
1.7.9.1	plugin:idle:maxidletime	28
1.7.9.2	plugin:idle:action	29
1.7.10	Mixcloud plugin settings	29
1.7.10.1	plugin:mixcloud:folder:download	29
1.7.10.2	plugin:mixcloud:folder:temp	29
1.7.10.3	plugin:mixcloud:youtube_dl:path	29
1.7.10.4	plugin:mixcloud:youtube_dl:options	29
1.7.10.5	plugin:mixcloud:youtube_dl:prefixes	29
1.7.10.6	plugin:mixcloud:to_mp3	30
1.7.11	MPD plugin settings	30
1.7.11.1	plugin:mpd:testpipe	30
1.7.11.2	plugin:mpd:volume	30
1.7.11.3	plugin:mpd:host	30
1.7.11.4	plugin:mpd:port	31
1.7.11.5	plugin:mpd:musicfolder	31
1.7.11.6	plugin:mpd:template:comment:disabled	31
1.7.11.7	plugin:mpd:template:comment:enabled	31
1.7.12	Soundcloud plugin settings	31
1.7.12.1	plugin:soundcloud:folder:download	31
1.7.12.2	plugin:soundcloud:folder:temp	31
1.7.12.3	plugin:soundcloud:youtube_dl:path	31
1.7.12.4	plugin:soundcloud:youtube_dl:options	32
1.7.12.5	plugin:soundcloud:youtube_dl:prefixes	32
1.7.12.6	plugin:soundcloud:to_mp3	32
1.7.13	Youtube plugin settings	32
1.7.13.1	plugin:youtube:folder:download	32
1.7.13.2	plugin:youtube:folder:temp	32

1.7.13.3	plugin:youtube:youtube_dl:path	33
1.7.13.4	plugin:youtube:youtube_dl:options	33
1.7.13.5	plugin:youtube:youtube_dl:prefixes	33
1.7.13.6	plugin:youtube:youtube_dl:maxresults	33
1.7.13.7	plugin:youtube:to_mp3	33
1.8	Known problems	33
1.8.1	General information	34
1.8.1.1	Watch log files	34
1.8.1.2	Running the bot without the manage.sh script	34
1.8.2	MPD versions that cause problems	34
1.8.3	Bot starts and crashes and cannot connect to MPD	34
1.8.4	Unregistered users recognition	35
1.8.5	The bot does not start	35
1.8.6	Downloading videos with special characters does not work	35
1.8.7	Bot does not start completely and shows plugins named false	36
1.9	Status of the project	36
1.10	FAQ – Frequently Asked Questions	36
1.10.1	Is it free?	36
1.10.2	Where is the source code?	36
1.10.3	Does it run on Windows?	36
1.10.4	Is it complicated to use the bot?	36
1.10.5	There are so many commands, where to start?	36
2	About this documentation	37
2.1	Contribute to this documentation	37

Mumble-Ruby-Pluginbot is an extensible bot which can play audio, be fed by different sources, and much, really much more :)

It is free and open source.

You can find the source code on [GitHub](#).

This is the documentation of the current stable version 0.10.3 of the Mumble-Ruby-Pluginbot.

Features

- Supports client certificates and thus can be registered on a server by an admin or even can register itself if the server allows it.
- As of version 0.10.x it is available in English, German and Bavarian, see [here](#).
- Bot automatically adjusts its bitrate to fit the servers bandwidth limitation.
- Stream audio from an MPD.
- Support for *plugins*.
- Can download music, for example from Youtube or other websites.
- Can search for music on Youtube or other websites.
- Streaming of internet radio.
- Live changing of bitrate or volume.
- Audio ducking – the bot lowers the playback volume if a user speaks.
- Supports both CELT and Opus codec for maximum compatibility even on old Mumble servers <1.2.4.
- No need for a web interfaces to control the bot. Everything can be done with text commands on your Mumble server.
- Support for whitelist of users who are only able to control the bot. Superusers are treated as being on the whitelist automatically.

For a complete list of features you must try the bot yourself. Write `.help` to your own bot and have fun :)

See also:

If you hesitate whether setting up the Mumble-Ruby-Pluginbot is worth it you may connect to [Natenoms Mumble-Server](#) in order to test one of the bots there. Just ask someone for an admin and a temporary registration on the server.

Compatibility

As of 2017-01 the stable release 0.10.x of Mumble-Ruby-Pluginbot was successfully installed and tested on the following distributions with the official *Installation howto*:

- Debian ‘Wheezy’ 7 (oldstable)
- Debian ‘Jessie’ 8 (stable)
- Ubuntu 14.04 (Trusty) (thanks @robin)
- Ubuntu 15.10
- Ubuntu 15.10 Server and Desktop
- Ubuntu 16.04 LTS (thanks @robin)
- Ubuntu 16.10 Server and Desktop
- Arch Linux (thanks @Nascher)
- Ubuntu based smallest DigitalOcean.com virtual server

Note: If you installed the bot successfully on another platform or system please create an [issue on our project page](#) to inform us and we will update this information. Thank you.

You can also write an email to [Natenom](#).

Usage

Basic overview

The bot reacts to text commands, prefixed with a control string. The default control string is a dot .

A good start for learning to control the bot is:

```
.help
```

Note: If you are really excited and want to know all the commands available then try the commands:

```
.help all
```

and:

```
.internals
```

But first you need the bot to come into your channel, that is done with:

```
.ch
```

which is a command of the *Control plugin*, see:

```
.help control
```

The basic music related commands you need for volume control, skip forward/backward, select the title to play, etc. are located in the *MPD plugin*. If you don't want to download new music this is basically all that you need. To get the help of this plugin write to the bot:

```
.help mpd
```

If there is no music in the bot yet you need to download some, see for example *Example 4 – Download music from Youtube*. But there are other plugins, too, which are able to download new music, see *Plugins of Mumble-Ruby-Pluginbot*.

Usage examples

Example 1 – Volume control

The music is playing but you want to lower the volume, lets say the current volume is 65% and you want to lower it to 50%, you may do:

```
.v---  
.v 50
```

Every – after the `.v` command (note that there is no space character) means 5% less.

Write `.help mpd` to the bot for details.

Example 2 – Change state

The bot takes its music from a queue. To show the current queue use the command `.queue`.

Each song in the queue is prefixed with an index number. This number can be used to play a specific song in the queue.

Use the command `.play 5` to play the 5th song in the list. Note that the first song has the index number 0.

Write `.help mpd` to the bot for details.

Example 3 – Listen to a radio station

First you must tell the bot to download/update the list of known radion stations with `.radioupdate`.

Now you can get a list of available categories with `.radiocategories`.

Choose one of them, for example **Electro** and lets display all the streams within this category with the command `.radiocategory Electro`.

Now choose one of the available streams with `.radioselect Electro 0`, in this example the first stream in the category Electro.

Now it may happen that the stream offers more than one stream because of several quality ranges. The bot may write you to “choose” one of them.

In order to do so use the command `.choose` and the bot will show a list of them. Then use `.choose 1` to choose the second stream in this case.

Yeay, it is not very user friendly currently but you have the ability to choose between thousands of radio streams and thats fantastic :)

Example 4 – Download music from Youtube

Lets say you want to listen to music from Mozart...

- First lets search on youtube:

```
.yts mozart
```

- The bot responds with:

```
0 Mozart for Baby (3 Hours) - ...
1 The Best of Mozart | 3 HOURS Piano Sonatas ...
2 Mozart for Studying and ...
```

- Now you can either let the bot download all search results:

```
.yta all
```

- or just one or more specific song(s):

```
.yta 0 2
```

- In both cases the bot will inform you about the current download status:

```
[21:59:22] Music Bot 1: do 2 time(s)...
[21:59:22] Music Bot 1: fetch and convert
[21:59:23] Music Bot 1: fetch and convert
```

- Followed by a database update:

```
[21:59:48] Music Bot 1: Waiting for database update complete...
```

- Now lets show the current music queue:

```
.queue
```

- The bot responds with:

```
0 Mozart for Baby (3 Hours) - ...
1 Mozart for Studying and ...
```

- Now lets play the first song in the queue with:

```
.play first
```

Plugins

The bot comes with several plugins for different purposes. For example there are plugins to stream an internet radio or to download music from youtube or to download music from mixcloud.

General information

Every plugin has its own help. To get a list of activated plugins use the command:

```
.plugins
```

To get a plugin specific help, use:

```
.help pluginname
```

For example:

```
.help youtube
```

Note: If you are really excited and want to know all the commands of all plugins available then try the command:

```
.help all
```

See also:

See also *Usage*.

Plugin specific information

Bandcamp

This plugin can download music from Bandcamp.

Usage:

```
.help bandcamp
```

Control

This plugin provides some nice commands like automute, follow, ...

Usage:

```
.help control
```

Ektoplazm

This plugin can download music from Ektoplazm.

Usage:

```
.help ektoplazm
```

Idle

This plugin is used so that the bot can go to his home channel after a specified idle time.

Usage:

```
.help idle
```

Messages

This plugin enables you to get specific status messages from the bot.

Usage:

```
.help messages
```

Mixcloud

This plugin can download music from Mixcloud.

Usage:

```
.help mixcloud
```

Mpd

This is the main plugin of the bot and has all the capabilities of the MPD running in the background.

Usage:

```
.help mpd
```

Null

This plugin does simply nothing :)

Radiostream

This plugin can search for radiostreams and play them.

Usage:

```
.help radiostream
```

Soundcloud

This plugin can download music from Soundcloud.

Usage:

```
.help soundcloud
```

Timer

This plugin can create user based timers.

Usage:

```
.help timer
```

Version

This plugin shows the bots version.

Usage:

```
.help version
```

Youtube

This plugin can search on Youtube and download music from there.

Usage:

```
.help youtube
```

How to run the bot

There are several methods to run the bot.

Installation options

Option 1: Install it on your own – Installation Howto

Using this installation howto is basically a copy and paste task, even if you are unexperienced with Linux.

See *Installation HowTo*.

Option 2: Use a VirtualBox Virtual Appliance – Download a Fully set up Mumble Ruby Pluginbot

Instead of setting up the bot yourself you can download a fully set up Mumble-Ruby-Pluginbot as a virtual appliance for VirtualBox. All you need to do after importing it to VirtualBox is to change one configuration file and add your server address and bot name.

The howto can be here: [appliance-label](#).

Option 3: Use Docker

There is a [Dockerfile available](#) to automatically build a Docker container running MPD and the current stable branch of Mumble-Ruby-Pluginbot.

Option 4: Preconfigured images for different systems

See systemimages-label.

Maintenance

Automatic update

Login as botmaster and run:

```
~/src/mumble-ruby-pluginbot/scripts/updater.sh
```

Select the first entry and press enter when prompted :)

Then restart your bot(s). Thats it :)

Be aware that this only works if you installed the bot with the official installation howto, see [Installation HowTo](#).

Manual update (not recommended)

If you did install the bot yourself and did not use the official installation howto then please check the updater.sh script in the scripts directory in order to know what you need to update by hand.

Installation HowTo

In this tutorial/howto we install the *Mumble-Ruby-Pluginbot* into a users home directory. No system services are used.

See also:

- [Compatibility](#)
- [How to run the bot](#)
- [Explain the configs](#)

How much time will you need?

This tutorial works well using copy and paste if you want one bot only. If you want more bots, read the note boxes and warning boxes :)

After 10 to 20 minutes it should be finished, depending on your servers internet connection.

How will it look like?

This will be the directory structure when this howto is completed:

```
/home/botmaster/  
- mpdl  
|   - playlists  
|   - mpdl.log  
|   - mpd.conf  
|   - mpd.fifo  
|   - pid
```



```

|   - state
|   - sticker.sql
|   - tag_cache
- (optional) more mpd directories ... mpd2, mpd3, etc.
- logs
- music
|   - downloadedfromyt
|   - downloadedfromsc
|   - [...]
- src
|   - certs
|       - nameofyourbot_cert
|       - [...]
|   - mumble-ruby
|   - mumble-ruby-pluginbot
|       - scripts
|           - mumblerubypluginbot.service
|           - overwrite_conf.rb
|           - manage.sh
|           - updater.sh
|       - [...]
|   - bot1_conf.yml
|   - (optional) more bot<number>_conf.yml
|   - [...]
- temp
|   - youtubeplugin
|   - bandcampplugin
|   - [...]
[...]
```

On Arch Linux only: Install and set up system package dependencies

Install the following dependencies as root or via sudo:

```
pacman -S libyaml opus zlib openssl git mpd mpc tmux automake \
autoconf libogg psmisc util-linux libtool curl base-devel wget aria2
```

On Debian/Ubuntu based Distributions only: Install and set up system package dependencies

Install the following dependencies as root or via sudo (please note that this is one command, copy and paste all four lines):

```
apt-get install curl libyaml-dev git libopus-dev \
build-essential zlib1g zlib1g-dev libssl-dev mpd mpc tmux \
automake autoconf libtool libogg-dev psmisc util-linux libgmp3-dev \
dialog unzip ca-certificates aria2
```

The message about no database `/var/lib/mpd/tag_cache` can be ignored. The database will be created as soon as you start the MPD server.

As we do not need the system wide MPD it can be disabled. To do this open the file `/etc/default/mpd` as root or via sudo and change `START_MPD` to `false`. At the next system start it will not be started any more.

Note: On newer distributions instead of editing the file you need to disable the MPD service by running the following command as root or with sudo:

```
systemctl disable mpd  
systemctl stop mpd
```

Create a user which should contain all the relevant bot structures

Note: It is crucial that you use the same username as in this tutorial. Otherwise you need to manually adapt most scripts and configuration files to another username.

As root or via sudo:

```
adduser botmaster
```

All relevant scripts will run within this user context.

Now it is the time to log in as your new user with:

```
su - botmaster
```

All the following steps are done as the user botmaster.

Create all needed directories and subdirectories for MPD and the bot(s)

Create a directory for the source code and scripts:

```
mkdir ~/src
```

Create a directory for log files:

```
mkdir ~/logs
```

Create a directory for the certificates:

```
mkdir ~/src/certs
```

Create a directory for the music:

```
mkdir ~/music
```

Create a directory for the temp files:

```
mkdir ~/temp
```

Create the directories for bot 1:

```
mkdir -p ~/mpdl/playlists
```

Note: Note that you can use more than one bots with this tutorial. Just create a new directory structure for every additional bot, for example with:

```
mkdir -p ~/mpd2/playlists
```

and so on.

Install and set up ruby and all needed libraries

We are using RVM (Ruby Version Manager) to install a local version of Ruby instead of using a system wide installed Ruby which may be too old.

First get and add the GPG key of RVM:

```
gpg --keyserver hkp://keys.gnupg.net --recv-keys   
↳ 409B6B1796C275462A1703113804BB82D39DC0E3
```

We need at least Ruby 1.9.x, here we use the latest stable version:

```
curl -L https://get.rvm.io | bash -s stable
```

Now we need to tell our current shell to use rvm:

```
source ~/.rvm/scripts/rvm
```

Disable autolibs so that rvm doesn't ask for root - we already installed all the dependencies earlier:

```
rvm autolibs disable
```

The next command should print nothing, if it does print something, login as root and install those:

```
rvm requirements
```

Now we install the latest stable version of Ruby:

```
rvm install ruby --latest
```

Set up a Ruby environment

Now setup the environment for Ruby:

```
rvm --create use @bots
```

Get and build mumble-ruby and ruby-mpd and other dependencies

Now we download the source code of Mumble-Ruby and build it:

```
cd src   
  
git clone https://github.com/dafoxia/mumble-ruby.git mumble-ruby   
  
cd mumble-ruby
```

```
rvm use @bots
gem build mumble-ruby.gemspec
rvm @bots do gem install mumble-ruby-*.gem
```

Install `ruby-mpd` so that the bot can control MPD:

```
rvm @bots do gem install ruby-mpd
rvm @bots do gem install crack
```

Download and set up celt-ruby and libcelt

For compatibility reasons the bot uses slightly modified versions of CELT which need to be built with the following steps:

```
cd ~/src
git clone https://github.com/dafoxia/celt-ruby.git
cd celt-ruby
rvm use @bots
gem build celt-ruby.gemspec
rvm @bots do gem install celt-ruby
cd ~/src
git clone https://github.com/mumble-voip/celt-0.7.0.git
cd celt-0.7.0
./autogen.sh
./configure --prefix=/home/botmaster/src/celt
make
make install
```

Download and set up opus-ruby

Do the following commands:

```
cd ~/src
git clone https://github.com/dafoxia/opus-ruby.git
cd opus-ruby
```

```
rvm use @bots  
  
gem build opus-ruby.gemspec  
  
rvm @bots do gem install opus-ruby
```

Download and set up mumble-ruby-pluginbot

Do the following commands:

```
cd ~/src  
  
git clone https://github.com/MusicGenerator/mumble-ruby-pluginbot.git  
  
cd mumble-ruby-pluginbot
```

We need a configuration file for the manage script:

```
cp templates/manage.conf ~/src/manage.conf
```

Now we create a copy of the config that we will use:

```
cp templates/override_config.yml ~/src/bot1_conf.yml
```

This approach has the advantage that this config file contains only the variables you want to change. All the other variables not set in your bot1_conf.yml are used from the ~/src/mumble-ruby-pluginbot/config/config.yml file and all plugins/*.yml files. See [here](#) for details.

You should now edit the bots configuration file named “bot1_conf.yml” with your favorite editor:

```
nano ~/src/bot1_conf.yml
```

... and adapt at least the following settings to your needs:

- mumble -> host
- mumble -> port
- mumble -> username
- mumble -> password
 - Note: This password is optional only and not needed if you want to register the bot as an admin on your server.
- mumble -> channel
 - Note: This channel name is the one your bots tries to enter when getting a .gotobed
- mumble -> bitrate
 - Note: If you set a higher bandwidth than your server can handle the bot automatically reduces its bandwidth to fit the servers needs.

The rest of the configuration file should be fine.

Note: For every additional bot you need to copy the original config file and edit it, for example for bot 2 do:

```
cd ~/src/mumble-ruby-pluginbot
cp templates/override_config.yml ~/src/bot2_conf.yml
```

Now edit `~/src/bot2_conf.yml` and change at least the following variables:

- main -> logfile to `"/home/botmaster/logs/bot2.log"`, and so forth
- main -> fifo to `"/home/botmaster/mpd2/mpd.fifo"`, and so forth
- mumble -> name to `"anything different then for bot1"`, and so forth
- plugin -> mpd -> port to 7702, and so forth

Set up MPD (Music Player Daemon)

Copy the configuration file for your local MPD:

```
cp ~/src/mumble-ruby-pluginbot/templates/mpd.conf ~/mpd1/mpd.conf
```

Note: For every additional bot you must increase the number of `~/mpd1/...` by one. For a second bot use:

```
cp ~/src/mumble-ruby-pluginbot/templates/mpd.conf ~/mpd2/mpd.conf
```

Then open the downloaded `mpd.conf` file and substitute every occurrence of `mpd1` by `mpd2`. Also increase the port from 7701 to 7702, 7703, etc.

Note: For experts only: Explanations about the configuration file and additional settings you may want to have...

- You can see the configuration file at [here](#)..
- Instead of the default sample rate of 44100 this config uses 48000 which is the sample rate Mumble clients use. And we need a mono signal.
- The mixer type is set to software so that the volume in MPD can be changed without the need of a real soundcard.
- You can enable [volume normalization](#) by adding the following line to the mpd config file:

```
volume_normalization "yes"
```

Set up the script to start your bot(s) and MPD instance(s)

Change into the `mumble-ruby-pluginbot` directory:

```
cd ~/src/mumble-ruby-pluginbot
```

The Bash script named `manage.sh` in the `scripts` directory is used to start all your bots and your MPD instance(s).

Make it executable:

```
chmod u+x ~/src/mumble-ruby-pluginbot/scripts/manage.sh
```

Also make the update script executable:

```
chmod u+x ~/src/mumble-ruby-pluginbot/scripts/updater.sh
```

Note: If you created more than one bot in this tutorial open the file and uncomment the needed lines to start your additional mpd instances.

You can see the script [here](#).

Without modification the scripts starts only bot 1, for every additional bot modify the script.

Install a custom version of youtube-dl

You don't want to rely on the distributions version of an old youtube-dl so you must setup an own.

Only on Arch Linux: Install dependencies for youtube-dl

As root or with sudo install:

```
pacman -S imagemagick ffmpeg python
```

Only on Debian/Ubuntu based distributions: Install the dependencies (if ffmpeg is available for your distribution)

First install as root or via sudo the following system packages:

```
apt-get install imagemagick ffmpeg python
```

Only on Debian/Ubuntu based distributions (OPTIONAL): Install the dependencies (if ffmpeg IS NOT available for your distribution)

On some distributions the package ffmpeg was replaced by libav-tools; install this if ffmpeg is not available.

Install as root or via sudo the following system packages:

```
apt-get install imagemagick libav-tools python
```

Also create the symlink so that the plugin can find it:

```
ln -s /usr/bin/avconv /usr/bin/ffmpeg
```

Login as botmaster

If not already logged in as botmaster, then do:

```
su - botmaster
```

Install the youtube-dl script

The youtube plugin needs youtube-dl; download it and make it executable:

```
curl -L https://yt-dl.org/downloads/latest/youtube-dl -o ~/src/youtube-dl
chmod u+x ~/src/youtube-dl
```

Almost done, start your bot(s) for the first time

You almost finished; now you can run your bot(s):

```
~/src/mumble-ruby-pluginbot/scripts/manage.sh start
```

When the bot(s) appear on your server, register it/them and start working with it/them. Try `.help` as the first command.

See also:

If the bot does not connect refer to *Known problems*.

Set up bot to start automatically on system startup

Start everything automatically – if your system is NOT systemd

Add the following lines to `/etc/rc.local` before the `exit . . .` line to start your bot(s) when your system starts:

```
su - botmaster -c "/home/botmaster/src/mumble-ruby-pluginbot/scripts/manage.sh start" &
```

The bot will start automatically on the next system start.

Start everything automatically – if your system is systemd

Run the following command as root:

```
cp /home/botmaster/src/mumble-ruby-pluginbot/templates/mumblerubypluginbot.service /
etc/systemd/system/
systemctl enable mumblerubypluginbot
```

The bot will start automatically on the next reboot.

Thats it, you are done :)

Controlling the bot(s) from your shell

To restart your bot(s) run:

```
~/src/mumble-ruby-pluginbot/scripts/manage.sh restart
```

To stop your bot(s) run:


```
~/src/mumble-ruby-pluginbot/scripts/manage.sh stop
```

To watch log files in real time run:

```
~/src/mumble-ruby-pluginbot/scripts/manage.sh log
```

To get the current status of your bot(s) run:

```
~/src/mumble-ruby-pluginbot/scripts/manage.sh status
```

Configuration settings of your bot

See also:

See *Explain the configs*.

Known problems

See also:

See *Known problems*.

Explain the configs

Configuration files

There is a main configuration file named `config/config.yml`.

Also every plugin does have its own configuration file, see `plugins/*.yml`.

Default configuration and override configuration

The bot always reads the default configuration files from `config/config.yml` and all plugin configuration files from `plugins/*.yml`.

If you want to use an own configuration file you don't need to write one with all configuration settings but only with those you want to change. A small sample override configuration file is available in `templates/override_config.yml`. All settings that you set there will overwrite those from the default configuration file `config/config.yml` and also those from every single plugin configuration file `plugins/*.yml`.

Syntax within this help

If we refer to a configuration option in this help text we write for example `main:tempdir` if we mean:

```
main:
  tempdir: "/home/botmaster/temp/"
```

Notes for editing the configuration

The documentation uses the YAML syntax, that means that it is based on indentation by space characters.

When you write a **string** you can enclose it in double quotes:

```
"--external-downloader aria2c"
```

Enclose the string into single quotes if you want to use double quotes within a string:

```
'--external-downloader aria2c --external-downloader-args "-j 6 -k 1M -x 10"'
```

Default config/config.yml

The following config shows all available configuration options of the `config/config.yml` as of version 0.10.x:

```
---
config:
  version: 2.2

debug: true
language: en

main:
  tempdir: "/home/botmaster/temp/"
  logfile: "/home/botmaster/logs/bot1.log"
  ducking: false
  automute_if_alone: true
  stop_on_unregistered: true
  channel_notify: 0
  controllable: true
  whitelist_enabled: false
  control:
    string: "."
    message:
      private_only: false
      registered_only: true
    historysize: 20
  display:
    comment:
      set: true
  user:
    whitelisted: # See http://mumble-ruby-pluginbot.rtf.d.io/en/master/explain\_the\_
↪config.html#main-user-whitelisted
    superuser: # See http://mumble-ruby-pluginbot.rtf.d.io/en/master/explain\_the\_
↪config.html#main-user-superuser
    banned: # See http://mumble-ruby-pluginbot.rtf.d.io/en/master/explain\_the\_config.
↪html#main-user-banned
    bound: # See http://mumble-ruby-pluginbot.rtf.d.io/en/master/explain\_the\_config.
↪html#main-user-bound
  certfolder: "/home/botmaster/certs/"
  fifo: "/home/botmaster/mpd1/mpd.fifo"
  logo: "../config/logo/logo.html"
  timer:
    ticks: 3600
  blacklisted_commands: ""
```

```
mumble:
  use_vbr: true
  bitrate: 72000
  host: m.natenom.com
  port: 64738
  name: "MumbleRubyPluginbot"
  password: ""
  channel: Bottest
```

Main settings

config:version

- Type: float
- Default: 2.2

This is for internal reasons only and is not meant to be changed.

debug

- Type: boolean
- Default: true
- Possible values: true, false

Set this to false to disable debug output in the logfile.

language

- Type: string
- Default: en
- Possible values: en, de, bar

Set this to the preferred language. “bar” is Bavarian.

main:tempdir

- Type: string
- Default: “/home/botmaster/temp/”

This is the base path where the bot downloads new music to, but every plugin that downloads music adds an own subdirectory to this path.

After the download the bots copies the downloaded files into the final directory defined in `plugin:mpd:musicfolder`, also into a plugin specific folder.

For example these are the resulting directories for the Youtube plugin:

- **temp:** /home/botmaster/temp/youtubepugin/
- **final:** /home/botmaster/music/downloadedfromyt/

main:logfile

- Type: string
- Default: “/home/botmaster/logs/bot1.log”

The path to the bots log file.

main:ducking

- Type: boolean
- Default: false
- Possible values: true, false

If true the bot automatically reduces its volume while users in the channel are talking.

main:automute_if_alone

- Type: boolean
- Default: true
- Possible values: true, false

The bot automatically mutes itself if he is alone in a channel.

main:stop_on_unregistered

- Type: boolean
- Default: true
- Possible values: true, false

The bot pauses the music or stops a radiostream if an unregistered user enters the channel.

main:channel_notify

- Type: int
- Default: 0

Calculating value for ‘channel_notify’:

Add all values for the desired channel notification

- 1 send message when volume change
- 2 send message when database update
- 4 send message when random mode changed
- 8 send message when single mode changed
- 16 send message when crossfading changed
- 32 send message when consume-mode changed
- 64 send message when repeat-mode changed

- 128 send message when state changes

Sum up all you need and use it as the configuration value.

main:controllable

- Type: boolean
- Default: true
- Possible values: true, false

Bot is only controllable if this is set to true. If false it will ignore all text commands.

main:whitelist_enabled

- Type: boolean
- Default: false
- Possible values: true, false

If true then only *whitelisted* can control the bot.

Superusers are treated as if they were on the whitelist.

main:control:string

- Type: string
- Default: ""

This is the character/string a user must prepend to text commands. The bot ignores commands not starting with that character/string.

main:control:message:private_only

- Type: boolean
- Default: false
- Possible values: true, false

If true the bot reacts only to private messages and not to channel messages. If false, the bot reacts to channel and private messages.

main:control:message:registered_only

- Type: boolean
- Default: true
- Possible values: true, false

If true the bot reacts only to registered users. If false also unregistered users can control the bot.

main:control:historysize

- Type: int
- Default: 20

Store this many entries in the command history of the bot.

main:display:comment:set

- Type: boolean
- Default: true
- Possible values: true, false

If true the bot sets its comment to display the current music that is being played.

main:user:whitelisted

You can define several whitelisted users here. To get a users hash use the command `.showhash`, see `.internals`.

Note that `main:whitelist_enabled` must be set to true in order for this to work.

Example:

```
whitelisted:
  72x60721xx216x4xx017f3x1x476d4358x48x648: dafoxia
```

main:user:superuser

You can define several superusers here. To get a users hash use the command `.showhash`, see `.internals`.

The commands `.reset`, `.set` and `.settings` can only be used by the defined superusers.

Example:

```
superuser:
  72x60721xx216x4xx017f3x1x476d4358x48x648: dafoxia
```

main:user:banned

You can define several banned users here. To get a users hash use the command `.showhash`, see `.internals`.

The bot will ignore the defined users completely.

Example:

```
banned:
  123452342348234782937ckjfv032ckj20938473: user3
```

main:user:bound

Only ONE user hash as a string. If defined nobody will be able to use the bind command anymore but the defined user. The blacklist command can only be used after being bound.

Example:

```
bound: "72x60721xx216x4xx017f3x1x476d4358x48x648"
```

main:certfolder

- Type: string
- Default: “/home/botmaster/certs/”

In this folder the bot automatically creates an openssl certificate per *Mumble username* you set up.

main:fifo

- Type: string
- Default: “/home/botmaster/mpd1/mpd.fifo”

This fifo must also be used by the MPD the bot connects to.

main:logo

- Type: string
- Default: “../config/logo/logo.html”

A relative path to the logo the bot uses.

main:timer:ticks

- Type: int
- Default: 3600

FIXME

main:blacklisted_commands

- Type: string
- Default: “”

Here you can disable specific commands. Be aware that currently the bot checks this very stupid. That means that it checks only the beginning of a word. For example if you blacklist set then settings is also blacklisted.

mumble:use_vbr

- Type: boolean
- Default: true
- Possible values: true, false

If true the bot encodes with a variable bitrate. If false it encodes with a constant bitrate.

mumble:bitrate

- Type: int
- Default: 72000

The overall bandwidth in bits per second the bot is allowed to use. Please note that the bot is able to ask the server for its maximum bandwidth and so can reduce its bitrate if you set it higher than possible.

mumble:host

- Type: string
- Default: m.natenom.com

The hostname or IP address of the Mumble server your bot connects to.

mumble:port

- Type: int
- Default: 64738

The port of the Mumble server your bot connects to.

mumble:name

- Type: string
- Default: "MumbleRubyPluginbot"

The name of your bot. Be aware that on most Mumble servers you are not allowed to use white spaces or other special characters.

mumble:password

- Type: string
- Default: ""

If your user is registered via a password, set it here or if the server uses a password, use this, too.

mumble:channel

- Type: string
- Default: “Bottest”

The channel the bot connects to. This is also the channel the bot tries to enter if you command it to `.gotobed`.

Bandcamp plugin settings

plugin:bandcamp:folder:download

- Type: string
- Default: “downloadedfrombc/”

The subdirectory the bot copies downloaded audio files into. The full path is built from `plugin:mpd:musicfolder+plugin:bandcamp:folder:download`.

plugin:bandcamp:folder:temp

- Type: string
- Default: “bandcampplugin/”

The subdirectory the bot downloads new audio files into. The full path is built from `main:tempdir+plugin:bandcamp:folder:temp`.

plugin:bandcamp:youtube_dl:path

- Type: string
- Default: “/home/botmaster/src/youtube-dl”

plugin:bandcamp:youtube_dl:options

- Type: string
- Default: “”

With this string you can add options/parameters to the executed `youtube_dl` shell command.

If `youtube_dl` command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:options` is `--restrict-filenames` then the full command results in:

```
youtube-dl --restrict-filenames foo bar
```

plugin:bandcamp:youtube_dl:prefixes

- Type: string
- Default: “”

This is a string that will prefix the `youtube_dl` shell command. You can set anything that would work on a shell. This can for example be used to renice the `youtube_dl` command.

If `youtube_dl` command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:prefixes` is `nice -n 19` then the full command results in:

```
nice -n19 youtube-dl foo bar
```

plugin:bandcamp:to_mp3

- Type: boolean
- Default: false

By default the bot tries to download OPUS encoded audio files or whatever is available.

Set this to true in order to convert audio files into MP3.

Ektoplazm plugin settings

plugin:ektoplazm:prefixes

- Type: string
- Default: ""

plugin:ektoplazm:folder:download

- Type: string
- Default: "ektoplazm/"

The subdirectory the bot copies downloaded audio files into. The full path is built from `plugin:mpd:musicfolder+plugin:ektoplazm:folder:download`.

plugin:ektoplazm:folder:temp

- Type: string
- Default: "ektoplazmplugin/"

The subdirectory the bot downloads new audio files into. The full path is built from `main:tempdir+plugin:ektoplazm:folder:temp`.

Idle plugin settings

plugin:idle:maxidletime

- Type: int
- Default: 600

Time in seconds the bot can idle before doing an action, see `plugin:idle:action`.

plugin:idle:action

- Type: string
- Default: “channel”
- Possible values: “channel”, “deafen”

If “channel” the bot enters its home channel when being idle longer than `plugin:idle:maxidletime`.

Mixcloud plugin settings

plugin:mixcloud:folder:download

- Type: string
- Default: “downloadedfrommc/”

The subdirectory the bot copies downloaded audio files into. The full path is built from `plugin:mpd:musicfolder+plugin:mixcloud:folder:download`.

plugin:mixcloud:folder:temp

- Type: string
- Default: “mixcloudplugin/”

The subdirectory the bot downloads new audio files into. The full path is built from `main:tempdir+plugin:mixcloud:folder:temp`.

plugin:mixcloud:youtube_dl:path

- Type: string
- Default: “/home/botmaster/src/youtube-dl”

plugin:mixcloud:youtube_dl:options

- Type: string
- Default: “-external-downloader aria2c -external-downloader-args “-j 6 -k 1M -x 10””

With this string you can add options/parameters to the executed `youtube_dl` shell command.

If `youtube_dl` command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:options` is `--restrict-filenames` then the full command results in:

```
youtube-dl --restrict-filenames foo bar
```

plugin:mixcloud:youtube_dl:prefixes

- Type: string
- Default: “”

This is a string that will prefix the `youtube_dl` shell command. You can set anything that would work on a shell. This can for example be used to renice the `youtube_dl` command.

If `youtube_dl` command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:prefixes` is `nice -n 19` then the full command results in:

```
nice -n19 youtube-dl foo bar
```

plugin:mixcloud:to_mp3

- Type: boolean
- Default: false

By default the bot tries to download OPUS encoded audio files or whatever is available.

Set this to true in order to convert audio files into MP3.

MPD plugin settings

plugin:mpd:testpipe

- Type: boolean
- Default: true
- Possible values: true, false

When this is set to true the bot will test whether MPD is running before it continues to start fully.

Note: On some systems this may fail and cause the bot to show several plugins named “false” when you use the command `.plugins`; set this value to `false` in such a case.

plugin:mpd:volume

- Type: int
- Default: 65
- Value range: 0 to 100

The default volume in % when the bot starts.

plugin:mpd:host

- Type: string
- Default: localhost

The host where your MPD server is running on.

plugin:mpd:port

- Type: int
- Default: 65

The port your MPD server is reachable at.

plugin:mpd:musicfolder

- Type: string
- Default: “/home/botmaster/music/”

plugin:mpd:template:comment:disabled

- Type: string
- Default: “Artist: DISABLED
Title: DISABLED
Album: DISABLED

Write %shelp to me, to get a list of my commands!”

plugin:mpd:template:comment:enabled

- Type: string
- Default: “Artist: %s
Title: %s
Album: %s

Write %shelp to me, to get a list of my commands!”

Soundcloud plugin settings

plugin:soundcloud:folder:download

- Type: string
- Default: “downloadedfromsc/”

The subdirectory the bot copies downloaded audio files into. The full path is built from `plugin:mpd:musicfolder+plugin:soundcloud:folder:download`.

plugin:soundcloud:folder:temp

- Type: string
- Default: “soundcloudplugin/”

The subdirectory the bot downloads new audio files into. The full path is built from `main:tempdir+plugin:soundcloud:folder:temp`.

plugin:soundcloud:youtube_dl:path

- Type: string
- Default: “/home/botmaster/src/youtube-dl”

plugin:soundcloud:youtube_dl:options

- Type: string
- Default: ""

With this string you can add options/parameters to the executed youtube_dl shell command.

If youtube_dl command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:options` is `--restrict-filenames` then the full command results in:

```
youtube-dl --restrict-filenames foo bar
```

plugin:soundcloud:youtube_dl:prefixes

- Type: string
- Default: ""

This is a string that will prefix the youtube_dl shell command. You can set anything that would work on a shell. This can for example be used to renice the youtube_dl command.

If youtube_dl command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:prefixes` is `nice -n 19` then the full command results in:

```
nice -n19 youtube-dl foo bar
```

plugin:soundcloud:to_mp3

- Type: boolean
- Default: false

By default the bot tries to download OPUS encoded audio files or whatever is available.

Set this to true in order to convert audio files into MP3.

Youtube plugin settings

plugin:youtube:folder:download

- Type: string
- Default: "downloadedfromyt"

The subdirectory the bot copies downloaded audio files into. The full path is built from `plugin:mpd:musicfolder+plugin:youtube:folder:download`.

plugin:youtube:folder:temp

- Type: string
- Default: "youtubeplugin/"

The subdirectory the bot downloads new audio files into. The full path is built from `main:tempdir+plugin:youtube:folder:temp`.

plugin:youtube:youtube_dl:path

- Type: string
- Default: “/home/botmaster/src/youtube-dl”

plugin:youtube:youtube_dl:options

- Type: string
- Default: “”

With this string you can add options/parameters to the executed youtube_dl shell command.

If youtube_dl command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:options` is `--restrict-filenames` then the full command results in:

```
youtube-dl --restrict-filenames foo bar
```

plugin:youtube:youtube_dl:prefixes

- Type: string
- Default: “”

This is a string that will prefix the youtube_dl shell command. You can set anything that would work on a shell. This can for example be used to renice the youtube_dl command.

If youtube_dl command is `youtube-dl foo bar` and `plugin:name_of_your_plugin:youtube_dl:prefixes` is `nice -n 19` then the full command results in:

```
nice -n19 youtube-dl foo bar
```

plugin:youtube:youtube_dl:maxresults

- Type: int
- Default: 200

plugin:youtube:to_mp3

- Type: boolean
- Default: false

By default the bot tries to download OPUS encoded audio files or whatever is available.

Set this to true in order to convert audio files into MP3.

Known problems

See also:

Please check also the [issues on the project page](#).

General information

Watch log files

Note: By default the bot does a very detailed *logging* into `~/logs/*log`. You may use any tool to view those logs. If debug logging is disabled you can enable it via the debug setting, see [here](#).

You can also watch the log files with `manage.sh`:

```
~/src/mumble-ruby-pluginbot/scripts/manage.sh log
```

Press `Ctrl + c` to stop the log output.

Running the bot without the `manage.sh` script

Only for experts: If you want to start the bot without the `manage.sh` script you must manually run the following commands after every login in order to use the correct bot environment:

```
source ~/.rvm/scripts/rvm
rvm use @bots
```

Now you can try to start the bot from the shell.

MPD versions that cause problems

The following MPD versions cause problems and should be avoided:

- 0.19.1 in Debian 8 crashes on AAC format radio streams.
- 0.20.1 causes loud noise/slow playback after songchange, see [here](#).

Note: If you send the command `.version` to your bot and if it uses a known buggy version of MPD then you get a note about that:

This MPD version (0.x.x) is known to cause problems, please refer to...

Bot starts and crashes and cannot connect to MPD

If the log contains the line:

```
[...]
pluginbot is starting...
start
/home/botmaster/.rvm/gems/ruby-2.2.1@bots/gems/ruby-mpd-0.3.3/lib/ruby-mpd.rb:106:in
↳ `connect': Unable to connect (possibly too many connections open)
↳ (MPD::ConnectionError)
    from /home/botmaster/src/mumble-ruby-pluginbot/plugins/mpd.rb:84:in `init'
    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:233:in `block in
↳ mumble_start'
    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:232:in `each'
```



```

    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:232:in `mumble_start
↪ '
    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:565:in `block in
↪ <main>'
    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:560:in `loop'
    from /home/botmaster/src/mumble-ruby-pluginbot/pluginbot.rb:560:in `<main>'
[...]
```

You can also check whether connecting via mpc works with:

```
mpc -p 7701 status
```

If it says:

```
error: Connection closed by the server
```

Or if you find the mentioned lines in your log file then add the following line to the file `/etc/hosts.allow`:

```
mpd: LOCAL
```

And restart the bot.

Unregistered users recognition

Be aware that there is a bug in the recognition whether an unregistered user is in the channel. This is the reason why we disabled this feature (`main:stop_on_unregistered`) earlier in this tutorial, because the bot recognizes itself as unregistered and doesn't start the music. If you didn't change the variable to false you should register your bot as an admin before starting music with the `.play` command.

The bot does not start

- Make sure that the desired bot name is not already registered on your server.
- Make sure you that you changed the `mumbleserver_host` in the bot's configuration file.

Downloading videos with special characters does not work

When you get something like this in your debug console:

```
UnicodeEncodeError: 'ascii' codec can't encode character u'\u2605' in position 49:
↪ ordinal not in range(128)
(END)
```

Make sure that your system has an appropriate locale available and add the following line to the second line of your `manage.sh`:

```
export LANG="en_US.UTF-8"
```

Replace this by your locale or use the above one. Make sure it is activated in `/etc/locale.gen`.

Bot does not start completely and shows plugins named false

If the `.plugin` command shows one or more plugins named `false` then you must change the configuration setting `plugin:mpd:testpipe` to `false`.

Status of the project

- Releases can be found at <https://github.com/MusicGenerator/mumble-ruby-pluginbot/releases>.
- News about the bot can be found in [Natenoms Blog](#).

FAQ – Frequently Asked Questions

Is it free?

Yes, completely. It is not only free to use, it is “Open Source” and it is “Free Software”.

Where is the source code?

You can find the source code on [GitHub](#).

Does it run on Windows?

No. But you can download a VirtualBox Virtual Appliance, see `appliance-label` with a Linux inside, and run this virtual machine on your Windows desktop.

Is it complicated to use the bot?

No, there are many commands but most users need only few of them, see next Question :)

There are so many commands, where to start?

See *Usage*.

About this documentation

This documentation is written in ReStructuredText.

- [ReStructuredText](#) (on Wikipedia)
- <http://www.sphinx-doc.org/en/1.5.1/rest.html>
- <http://www.sphinx-doc.org/en/1.5.1/markup/inline.html>
- <http://docutils.sourceforge.net/docs/ref/rst/restructuredtext.html>

Contribute to this documentation

You can contribute to this documentation on Github, see <https://github.com/MusicGenerator/mumble-ruby-pluginbot-docs>.