Benben v0.5.0

Remilia Scarlet

Copyright © 2024 Remilia Scarlet

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Short Contents

Table of Contents

1 Overview of Benben

1.1 Introduction

Benben is a fast and efficient command line audio player, and audio->PCM converter, for Linux/Unix. It supports multiple input formats, and is especially suited to people who organize their music in folders, and for those who prefer to use terminals instead of GUIs.

Benben is written almost entirely in the Crystal (`https://crystal-lang.org/`) programming language.

1.2 Supported Formats

- VGM files (`https://vgmrips.net/`)
  - Uncompressed (.vgm)
  - GZip compressed (.vgz)
  - ZStandard compressed (.vgzst)
  - BZip2 compressed (.vgb)
- MPEG-1
  - MPEG-1 Layer III (.mp3)
  - MPEG-1 Layer II (.mp2)
  - MPEG-1 Layer I (.mp1)
- Ogg Vorbis
- Opus
- FLAC
- General MIDI (.mid, .rmi)
- All module formats supported by libxmp (`https://github.com/libxmp/libxmp/`)
- RIFF WAVE (.wav)
- Sun Au (.au)

1.3 History

Before Benben...

Before Benben ever existed, Remilia started work on a port of a SoundFont synthesis library called MeltySynth[1] from C# to Crystal. Her port of this software was called[2]. As part of her port, she included a simple command line player called midi123, which was very bare bones and was intended to be nothing more than an example/test player for the library. However, it quickly grew to be more than a simple example and eventually became a standalone command-line player for General MIDI files.

About a year later, Remilia started to work on creating a VGM playback library called YunoSynth[3] for native Crystal. This was partially a port of vgmplay[4] from C, and partially

[1] `https://github.com/sinshu/meltysynth/`

[2] Haematite (`https://chiselapp.com/user/MistressRemilia/repository/Haematite/`)

[3] `https://chiselapp.com/user/MistressRemilia/repository/yunosynth/`

[4] `https://github.com/vgmrips/vgmplay/`

an object-oriented rewrite. Similar to midi123, and using its code as a base, Remilia also created a companion command-line player for YunoSynth called Benben.

The first few version of Benben only played VGM files (or their compressed variants), and used raw ANSI console commands to draw its interface. Starting with v0.3.0, Benben moved to an interface built with S-Lang (more accurately, her own Crystal bindings to S-Lang). Remilia also found herself continually wishing that Benben could play other formats so that she wouldn't have to switch between audio players, and so beginning with v0.5.0, Benben became a general-purpose music player that supports multiple input formats.

2 Basic Usage

2.1 Playing Files

Benben is a command line program. To play a file, simply pass it as an argument to Benben:

```
benben coolsong.mp3
```

Multiple files can also be specified, one after another. They will be played in the order that you list them:

```
benben coolsong1.mp3 other-directory/coolsong2.flac coolsong3.vgm
```

If you specify a directory, then Benben will play all files in that directory in alphabetical order. It will not recurse into any subdirectories, however.

```
benben directory-with-music/
```

Playlists in XSPF[1] or JSPF[2] format can also be used. These can be passed in like any other file, and their contents will be queued up in the same order that the playlist specifies.

```
benben cool-mix.xspf
```

Benben supports loading VGM files from HTML or Gemini[3] links as well. As of v0.5.0, this is only supported for VGM files.

```
benben https://example.com/song.vgm
benben gemini://example.com/song.vgm
```

Finally, you can mix and match any of these methods as you see fit:

```
benben cool-mix.xspf coolsong1.flac /directory/with/music/ https://example.com/song.vg
```

2.2 User Interface

During playback, a progress bar will be displayed at the bottom. It will look something like this:

```
[EsCr-] 1/1, 1 of 2: |*****************---------------------|  45% [02:20/05:09]█
```

The characters to the left in brackets indicate the state of various effects and other pieces of information. These are, in order from left to right:

‘E’	EQ is on.
‘e’	EQ is off.
‘S’	Stereo enhancer is on.
‘s’	Stereo enhancer is off.
‘C’	Soft clipping is on.
‘c’	Soft clipping is off.
‘R’	Reverb is on.

[1] https://xspf.org/

[2] https://xspf.org/jspf

[3] https://geminiprotocol.net/

‘`r`’ Reverb is off.

‘`+`’ A song-specific config was found for the current song and loaded. See Section 4.3 [Song-specific Config Files], page 17,

‘`-`’ A song-specific config was not found for the current song. See Section 4.3 [Song-specific Config Files], page 17,

Next to these characters will be the current track, a slash, and then total number of tracks. Following this is the loop information (e.g. ‘`1 of 2`’ means “currently in the first loop out of two”). When the loop information displays as ‘`1 of *`’, then the song is looping indefinitely. Note that for VGM files, if the song does not have any loop information in it, then this will always read as ‘`1 of 1`’.

2.3 Keyboard Input

The program will respond to various keys during playback. These are listed below, and you can also press the *h* key to see them while Benben is running.

n Go to the next file in the playback queue. If the `--repeat` option is used and you are already on the last item in the queue, then Benben will loop back to the first item. See Chapter 3 [Command Line Options], page 6,

p Go to the previous file in the playback queue. If you are already on the first item, Benben will simply restart the playback of that item.

t Toggle the displayed GD3 tag language. Only works for VGM files.

i Toggle the interpolation mode. Only for module/tracker formats.

c Toggle soft clipping on and off.

e Toggle the EQ effect on and off.

s Toggle the stereo enhancer effect on and off.

r Toggle the reverb effect on and off.

a Raise the volume.

z Lower the volume.

] Increase the number of times the song loops. Not all formats support this.

[Increase the number of times the song loops. Not all formats support this.

q Quit the program.

R Redraws the entire screen.

S Toggles the “stop after current song” behavior. When stopping after the current song, Benben will exit once the current song finishes playing or you hit *n* to move to the next file.

C Reloads all song-specific configuration files. See Section 4.3 [Song-specific Config Files], page 17,

> Seeks forward a bit. Not all formats support this.

`<` Seeks backwards a bit. Not all formats support this.

`x` Toggles the chorus effect. This only works for General MIDI files.

`space` Pause/unpause playback.

3 Command Line Options

General Options

`--help, -h`

Displays help information, then exits.

`--version, -v`

Displays brief version information, then exits.

`--driver` *x*, `-d` *x*

Selects the audio driver to use. Use `--driver list` to see a list of available drivers. The default is `pulseaudio`.

`--list-themes`

Lists all available themes. See Chapter 5 [Themes], page 19,

`--theme` *x*, `-T` *x*

Use the specified theme. See Section 5.1 [Specifying a Theme], page 19,

`--dump-config`

Generates a fresh configuration file and prints it to standard output, then exits.

`--dump-song-config`

Generates a fresh song-specific configuration file and prints it to standard output, then exits.

`--chip-info`

Displays a list of supported VGM chips, then exits.

`--long-version`

Show verbose version information, then exits.

Network Options

`--gemini-no-cert-verify`

Don't require verification of TLS certificates when downloading over the Gemini protocol.

Sound Options

`--volume` *x*, `-v` *x*

Sets the output volume. The valid range is 0.0 to 3.0, and the default is 1.0.

`--sample-rate` *x*, `-S` *x*

Sets the output sample rate, in hertz. Valid range is 8000 to 48000, and the default is 44100.

`--no-eq, -e`

Start with the equalizer disabled. The equalizer can still be toggled during playback with the `e` key.

`--soft-clipping, -c`

Start with soft clipping enabled on the output. This can still be toggled during playback with the `c` key.

`--no-soft-clipping, -C`

Start with soft clipping disabled on the output. This can still be toggled during playback with the `c` key.

`--stereo-enhancer, -t`

Start with the stereo enhancer effect enabled. This can still be toggled during playback with the `s` key.

`--stereo-enhancement` *x*, `-E` *x*

Sets the stereo enhancement amount when the stereo enhancer is enabled. The valid range is 0.0 (nearly monaural) to 1.5 (very wide). The default is 0.5, which is equivalent to not having the effect enabled. If the stereo enhancer is not enabled, this does nothing.

`--reverb, -r`

Start with the reverb effect enabled. This can still be toggled during playback with the `r` key.

`--no-reverb, -R`

Start with the reverb effect disabled. This can still be toggled during playback with the `r` key.

`--reverb-type` *x*

Set the type of reverb unit that is used. You can use `--reverb-type list` to see a listing of valid types. The default is `mverb`.

`--reverb-amount` *x*, `-r` *x*

The amount of reverb to apply to the output. Valid range is 0.0 to 1.0 default is 0.5. This does nothing if the reverb is disabled.

`--reverb-preset` *x*

Changes the reverb unit's parameters to an alternate preset. You can use `--reverb-preset list` to see a listing of valid types for the selected reverb unit. The default is `GmDefault`.

`--replay-gain` *x*

Set the ReplayGain mode. This only applies to MP3 files. Use `--replay-gain list` to see a list of values; The default is `disabled`.

Playback Options

`--loop` *x*, `-l` *x*

The number of times to loop during playback and rendering; a value of 0 means to loop indefinitely. A value of 0 cannot be used when rendering to a file. Only certain formats support looping.

`--repeat, -p`

Go back to the first song after the last one has finished playing.

`--no-repeat, -P`

Do not go back to the first song after the last one has finished playing. This is mainly meant for overriding the config file setting.

`--shuffle`

Randomizes the playback queue before starting playback.

Rendering Options

`--render, -n`
Tells Benben to render files to WAV (or Au) rather than play them back. Each input file is rendered to its own destination file, where the output name matches the original filename. This renders to WAV by default.

`--quiet, -Q`
Don't print any messages or progress, except errors.

`--normalize, -N`
Normalize each rendered file so that it's peak is zero dBFS.

`--cue x` Write a CUE file for the rendered files.

`--au, -A` Render to Au instead of WAV.

`--float, -F`
Output files will contain IEEE Floating Point data instead of integer PCM data.

`--bit-depth` *x*`, -b` *x*
Set the bit depth for the output files. Valid bit depths are 8, 16, 24, 32, and 64. The default is 16 when rendering integer PCM data, and 32 when rendering IEEE floating point data.

`--outdir` *x*`, -o` *x*
Where to save the rendered files.

`--overwrite`
Overwrite existing files when rendering.

`--jobs x` The number of parallel rendering jobs to use. This is only meaningful when rendering more than one input file. This must be at least 1, and the default is equal to the number of logical CPU cores.

4 Configuration

Benben's configuration file lets you specify many of its options, saving you from having to type them in on the command line each time, as well as a few that aren't available via the command line. The file uses the YAML[1] format for ease of use. The first time you run Benben, a default configuration file will be created. If you wish to create a fresh one at a later date, you can run `benben --dump-config`, which will print a fresh configuration file with the default values to standard output.

The location of the configuration file depends on your platform. On Linux/Unix, the file is located at `$XDG_CONFIG_HOME/benben/benben.yaml` (this usually means `~/.config/benben/benben.yaml`).

Note that command line options can still override the configuration file. So you can have your configuration file set to loop each song 3 times, you still use the `--loop` parameter to temporarily change to another value. See Chapter 3 [Command Line Options], page 6,

Important Note

The configuration format is not considered stable, and won't be until Benben v1.0 is released. Until then, you *may* have to make small adjustments when upgrading between versions. These changes will be noted in the release notes for each version of Benben.

4.1 Example Configuration File

```
audio-driver: any
buffer-size: 1024
sample-rate: 44100
max-loops: 1
repeat-playlist: false
render-jobs: 20
fadeout-seconds: 5
no-soft-clipping: false
enable-stereo-enhancer: false
stereo-enhancement-amount: 1.0
main-volume: 1.0
preferred-gd3-lang: japanese
ui:
  animations-enabled: true
emulators:
  dmg-boost-wave-chan: true
  huc6280-core: ootake
  ym2151-core: mame
vu-meter:
  bar-character:
  tip-character:
  colors:
  - 2
```

[1] https://en.wikipedia.org/wiki/YAML

```
  - 3
  - 1
  clipped-channel-time: 2
equalizer-enabled: false
equalizer-disabled-during-rendering: false
equalizer:
  low-shelf:
    frequency: 80.0
    gain: 0.0
    bandwidth: 0.707
  bands:
  - frequency: 105.0
    gain: 0.0
    bandwidth: 1.0
  - frequency: 800.0
    gain: 0.0
    bandwidth: 1.0
  - frequency: 4600.0
    gain: 0.0
    bandwidth: 1.0
  high-shelf:
    frequency: 12000.0
    gain: 0.0
    bandwidth: 0.707
reverb-enabled: false
reverb-type: mverb
reverb-amount: 0.5
mverb-reverb-preset: GmDefault
zita-reverb-preset: GmDefault
```

4.2 Configuration Options

4.2.1 Main Options

Key Name	**Type**	**Description**
`audio-driver`	One of: `pulseaudio`, `portaudio`, `ao`, or `any`	The output driver used to produce sound during playback. Use `benben --driver list` to see a list of valid options. Default: ‘`pulseaudio`’. See Chapter 3 [Command Line Options], page 6,

`buffer-size`	An integer that is at least 256, and evenly divisible by 256.	The size of the audio buffer. Note that this also affects the smoothness of the VU meter. In most cases, this doesn't need changing. Default: '`1024`'.
`sample-rate`	An integer between 8000 and 48000.	The output sample rate, in hertz. Default: '`44100`'.
`resampler`	One of: `Linear`, `ZeroOrderHold`, `SincFastest`, `SincMedium`, or `SincBest`	The resampler mode to use when playing a file that does not match the `sample-rate` setting. Default: '`SincFastest`'.
`replay-gain`	One of: `Disabled`, `Mix`, or `Album`	How to apply ReplayGain. Default: '`Disabled`'. Note that this currently only applies to MPEG-1 files (.mp3/.mp2/.mp1).
`stderr-logging`	One of: `full`, `crystal_only`, or `none`	When this is '`full`', then all messages that would normally be printed to standard error will instead appear in `~/.local/share/benben/stderr.log` (or wherever the `XDG_DATA_DIRS` environment variable is pointing). This includes messages from the audio drivers. When this is set to '`crystal_only`', then only messages that originate from the Crystal side of the code will be logged to the file (i.e., audio driver messages won't be logged to that file). If set to '`none`', then no stderr messages will be logged, and all of them will instead appear in the terminal like normal. This setting should only be changed by a user who understands what they're doing, and the default ('`full`') is fine for nearly all cases.

`max-loops`	An integer between 0 and 4294967295	The maximum number of times to loop if the song has loop information. If the song has no loop information, this is ignored. A value of ‘`0`’ means “loop forever”. Default: ‘`1`’ (meaning “play once, then loop once”).
`repeat-playlist`	`true` or `false`	When ‘`true`’, then the player will loop back to the first song once the final song is finished playing. This only has an effect during playback, not during rendering. Default: ‘`false`’.
`render-jobs`	Integer >= 1	The number of workers (threads) to use when rendering in parallel. This must be at least 1, and defaults to the number of logical CPU cores you have.
`fadeout-seconds`	An integer between 0 and 255	The number of seconds a looping song will fade out when finished. Default: ‘`5`’. This only applies to certain formats.
`main-volume`	Float between 0.0 and 2.0	The main output volume. Default: ‘`1.0`’.
`no-soft-clipping`	`true` or `false`	When ‘`true`’, soft clipping is disabled by default. Default: ‘`false`’.
`enable-stereo-enhancer`	`true` or `false`	When ‘`true`’, this enables a stereo enhancement effect. Default: ‘`false`’.
`stereo-enhancement-amount`	Float between 0.0 and 1.5.	The amount of stereo enhancement to apply to the output signal. This requires `enable-stereo-enhancer` to be ‘`true`’. This can go from ‘`0.0`’ (nearly monaural) to ‘`1.5`’ (very wide). Default: ‘`0.5`’, which is the same as not enabling this effect.

`reverb-enabled`	`true` or `false`	When ‘`true`’, then a reverb effect will be enabled by default. Default: ‘`false`’.
`reverb-type`	`mverb` or `zita`	The type of reverb to apply when reverb is enabled. Default: ‘`mverb`’.
`reverb-amount`	Float between 0.0 and 1.0	The amount of reverb to apply when reverb is enabled. Default: ‘`0.5`’.
`mverb-reverb-preset`	String	The preset to use for the reverb when reverb is enabled and `reverb-type` is set to ‘`mverb`’. Default: ‘`GmDefault`’. Use the command line option `--reverb-preset list` together with `--reverb-type` to see a list of presets for that type.
`zita-reverb-preset`	String	The preset to use for the reverb when reverb is enabled and `reverb-type` is set to ‘`zita`’. Default: ‘`GmDefault`’. Use the command line option `--reverb-preset list` together with `--reverb-type` to see a list of presets for that type.
‘preferred-gd3-lang‘	One of: `english`, `japanese`, `toggle_prefer_english`, or `toggle_prefer_japanese`	The default language in which to display GD3 tag info. Default: ‘`japanese`’. This can be toggled during playback with the *`t`* key. Only applicable to VGM files.
`ui`	Sub-block. See Section 4.2.2 [UI Config Block], page 14,	The settings for the user interface.
`emulators`	Sub-block. See Section 4.2.3 [Emulator Config Block], page 14,	Sets emulator-specific settings for VGM files.
`vu-meter`	Sub-block. See Section 4.2.4 [VU Meter Config Block], page 15,	Configuration section for the VU meter.

`equalizer-enabled`	`true` or `false`	When ‘`true`’, an equalizer effect will be applied to the output. You can still turn on and off the equalizer by pressing the `e` key during playback.
`equalizer-disabled-during-rendering`	`true` or `false`	When ‘`true`’, no equalizer effect will be applied to the output when rendering a VGM to WAV/Au. This only takes effect when rendering to files.
`equalizer`	Sub-block. See Section 4.2.5 [Equalizer Config Block], page 15,	Configuration section for the equalizer.

4.2.2 UI Config Block

Key Name	Type	Description
`animations-enabled`	`true` or `false`	When true, then various animations will be displayed by the user interface. This does not affect the VU meters. Default: ‘`true`’.
`banner`	One of: `graffiti`, `cyber`, `rounded`, `chunky`, `soft`, `doomed`, `stronger`; or an array containing one or more of these values.	The banner to show at the top of the interface. If this is an array, a random banner from the array will be chosen. This cannot be an empty array. Default: an array with all of the possible values.
`theme`	String	The name of the theme to use. Default: ‘`default`’. See Chapter 5 [Themes], page 19,

4.2.3 Emulator Config Block

Key Name	Type	Description
`dmg-boost-wave-chan`	`true` or `false`	Doubles the volume of the wave channel when playing GameBoy music. Default: ‘`true`’.

`huc6280-core`	`mame` or `ootake`	Determines the emulation core used for music that uses the HuC6280 chip (e.g. PC Engine/TurboGrafx-16). Default: ‘`ootake`’
`ym2151-core`	`mame` or `nuked`	Determines the emulation core used for music that uses the YM2151 chip. Default: ‘`mame`’.

4.2.4 VU Meter Config Block

Key Name	**Type**	**Description**
`bar-character`	Any Unicode character	This is the character that is used to draw the VU meter bar. Default: ‘’.
`tip-character`	Any Unicode character	This is the character that is used to draw the tip of the VU meter bar. Default: ‘’.
`clipped-channel-time`	An integer between 1 and 255, inclusive	The amount of time in seconds that the VU meter will show the channel name in red if clipping occurs. Default: ‘`2`’.
`colors`	An array of integers between 0 and 255, inclusive	The ANSI terminal colors to use for the VU meter segments. The VU meter will be divided up into as many segments as there are colors, with the first color being the left-most (most quiet) segment. There must be at least one color, and there can not be more than 70. You can use any ANSI 8-bit color index[2]. Default: ‘`[1, 3, 2]`’ (red, yellow, green).

4.2.5 Equalizer Config Block

Key Name	**Type**	**Description**
`post-gain`	Float	The amount of gain to apply to the signal *after* the EQ has processed it, in decibels. Default: ‘`0.0`’ (no change to the signal).

[2] `https://en.wikipedia.org/wiki/ANSI_escape_code#8-bit`

`low-shelf`	Sub-block. See Section 4.2.5.1 [Low-shelf Config Block], page 16,	The settings for the low shelf filter.
`bands`	An array of sub-blocks. See Section 4.2.5.2 [EQ Band Config Block], page 16,	An array of peaking EQ bands for the equalizer. There can be as many or as few as you wish.
`high-shelf`	Sub-block. See Section 4.2.5.3 [High-shelf Config Block], page 16,	The settings for the high shelf filter.

4.2.5.1 Low-shelf Config Block

Key Name	**Type**	**Description**
`frequency`	Float between 20.0 and 22050.0	The cutoff frequency for the low shelf in hertz. Default: ‘`80.0`’.
`gain`	Float	How much the volume of the signal under the low shelf is adjusted, in decibels. Default: ‘`0.0`’.
`bandwidth`	Float	How wide the transition band of the filter is, in octaves. Default: ‘`0.707`’.

4.2.5.2 EQ Band Config Block

Key Name	**Type**	**Description**
`frequency`	Float between 20.0 and 22050.0	The center frequency for the band in hertz.
`gain`	Float	How much the volume of the signal around the center frequency is adjusted, in decibels.
`bandwidth`	Float	The width of the band, in octaves.

4.2.5.3 High-shelf Config Block

Key Name	**Type**	**Description**
`frequency`	Float between 20.0 and 22050.0	The cutoff frequency for the high shelf in hertz. Default: ‘`80.0`’.

gain	Float	How much the volume of the signal under the high shelf is adjusted, in decibels. Default: '0.0'.
bandwidth	Float	How wide the transition band of the filter is, in octaves. Default: '0.707'.

4.3 Song-specific Config Files

In addition to the main config file, you can specify alternative configurations for specific song files or directories. These are matched using globbing[3]. When Benben notices you are playing a song file that matches one of these song-specific configurations, it will use the settings in that file to temporarily override your main configuration settings (the command line can still override everything, however). This lets you easily create settings specific to entire groups of songs. For example, I normally have the equalizer enabled, but I have specific settings for one album that disables the equalizer.

Song-specific config files reside in:

On Linux/Unix
: `$XDG_CONFIG_HOME/benben/song-configs/` (this usually means `~/.config/benben/song-configs/`).

The filename for every song-specific configuration file must start with `song-config-` and end in `.yaml`. For example, `song-config-x68k-eq.yaml`.

Song-specific config files follow the same basic format as the main configuration file, with a few changes. The most important change is that song-specific config files have an additional key, `match`, that lets you specify one or more patterns that will be used to match files. For example, the file `song-config-x68k-eq.yaml` on my system has this line at the very top:

```
match:
  - /mnt/nanako/vgms-and-mods/VGMs/X68000/**/*.vg?
```

This specifies that any VGM file in any subdirectory under `/mnt/nanako/vgms-and-mods/VGMs/X68000/` gets this song-specific config applied to it during playback. Any songs that don't match this pattern (or any pattern in any other song-specific config files I may have) use the settings in the main config file.

Any keys within a song-specific configuration file will override the main configuration settings (command line arguments still override everything). Keys that are missing from the song configuration file will be filled in with the values from the main configuration file, or the defaults.

Song configuration files are almost identical to the normal configuration file format, except they ignore the following keys:

- `audio-driver`
- `buffer-size`
- `render-jobs`

[3] https://en.wikipedia.org/wiki/Glob_(programming)

- `sample-rate`
- `ui`
- `repeat-playlist`
- `resampler`
- `seek-time`
- `equalizer`

All other keys are valid. See Section 4.2.1 [Main Configuration Options], page 10,

You can use `benben --dump-song-config` to print a new config to standard output that you can then modify as you wish.

4.4 Example Song-Specific Config File

```
match:
  - /mnt/nanako/vgms-and-mods/VGMs/X68000/**/*.vg?

equalizer:
  post-gain: 0.0
  low-shelf:
    frequency: 80.0
    gain: 2.7
    bandwidth: 1.8
  bands:
    - frequency: 105.0
      gain: 2.98
      bandwidth: 0.9
    - frequency: 275.0
      gain: -6.0
      bandwidth: 0.2
    - frequency: 6000.0
      gain: 1.5
      bandwidth: 1.0
  high-shelf:
    frequency: 9001.0
    gain: 1.4
    bandwidth: 0.9
```

5 Themes

Benben stores its themes in YAML format[1] files within a special `themes` folder in its configuration directory. The exact location of this folder depends on your platform:

On Linux/Unix
: `$XDG_CONFIG_HOME/benben/themes/` (this usually means `~/.config/benben/themes/`).

Each theme is stored in its own file, and the filename of each theme must use the format `theme-<theme name>.yaml`. So for example, a theme named "dark-custom" must be in the `themes` directory and have the filename `theme-dark-custom.yaml`.

If this directory does not exist, Benben will create it at startup. If no theme is specified, Benben uses its default built-in theme.

5.1 Specifying a Theme

The theme to use is specified in your main configuration file in the `ui-config` section (See Chapter 4 [Configuration], page 9):

```
ui-config:
  theme: default
```

You can also change the theme on the command line:

```
benben --theme smooth-neon coolfile.mp3
```

The name of the theme depends on its filename. As mentioned previously, the filename of each theme must use the format `theme-<theme name>.yaml`. So for example, a theme stored in a file named `theme-dark-custom.yaml` is named "dark-custom" and can be specified using `benben --theme dark-custom`.

To see a list of available themes, use the `--list-themes` argument. See Chapter 3 [Command Line Options], page 6,

5.2 Theme File Format

Note that color values can be expressed multiple ways. See Section 5.2.1 [Theme Color Values], page 22,

Key Name	**Type**	**Description**
`version`	The integer value `1`	The version of the Theme Format specification this file conforms to. In all cases, this should be set to '`1`'. The default is '`1`' if this key is not specified.
`bg-color`	Color value	The color of the background.

[1] `https://en.wikipedia.org/wiki/YAML`

`fg-color`	Color value	The color of all text that isn't covered by another key. In other words, the default foreground color.
`banner-color`	Color value	The color of the banner text.
`banner-fade-down-bright`	Color value	The color of bright lines when the banner text is doing its fade-down animation.
`banner-fade-down-dim`	Color value	The color of dim lines when the banner text is doing its fade-down animation.
`header-color`	Color value	The color of the field headers.
`err-color`	Color value	The color of the "Error:" text when displaying an error.
`cur-song-color`	Color value	The color of the currently playing song in the playback queue.
`prev-song-color`	Color value	The color of the previous song in the playback queue.
`next-song-color-1`	Color value	The color of the song that is next in the playback queue.
`next-song-color-2`	Color value	The color of the song that is two spots away in the playback queue.
`next-song-color-3`	Color value	The color of all songs three spots away and further in the playback queue.
`song-queue-box-color`	Color value	The color of the border of the playback queue box.
`song-queue-header-color`	Color value	The color of the "Song Queue" header text of the playback queue box.

`progress-bar-char`	A single character, or a string containing a single character.	The character used to draw the progress part (left-hand side) of the progress bar. This can be any UTF-8 encoded character as long as it's equivalent to a single Unicode code point (a single "character", essentially).
`progress-bar-space-char`	A single character, or a string containing a single character.	The character used to draw the right-hand side of the progress bar. This can be any UTF-8 encoded character as long as it's equivalent to a single Unicode code point (a single "character", essentially).
`progress-bar-colors`	An array of one or more color values	The colors for the progress bar. There must be at least one color, and there can be up to 38 different colors.
`vu-clip-color`	Color value	The color of the words "Left" and "Right" displayed next to the VU meters when a song clips.
`vu-clipped-channel-time`	An integer between 0 and 255, inclusive	How long in seconds the words "Left" and "Right" remain in their `vu-clip-color` when Benben detects that clipping has occurred.
`vu-bar-character`	A single character, or a string containing a single character.	The character used to draw the bar segments of the VU meter. This can be any UTF-8 encoded character as long as it's equivalent to a single Unicode code point (a single "character", essentially).
`vu-bar-character`	A single character, or a string containing a single character.	The character used to draw the tip of the VU meter. This can be any UTF-8 encoded character as long as it's equivalent to a single Unicode code point (a single "character", essentially).

`vu-colors`	An array of one or more color values	The colors for each VU meter segment. There must be at least one color, and there can be up to 70 different colors.

5.2.1 Color Values

Colors can be expressed in one of three ways:

- An integer between 0 and 255, inclusive. This acts as an index into the 8-bit ANSI color table[2], so for example a value of ‘`33`’ would be a light blue color.
- An array of three integer values, each between 0 and 255, inclusive. This defines a 24-bit RGB color, where the first element is the red value, the second is the green, and the third is the blue.
- A string in the format `#RRGGBB`, which also defines a 24-bit RGB color. `RR` segment is a hex value between ‘`00`’ and ‘`FF`’, inclusive, that defines the red value, `GG` is a hex value for green, and `BB` is a hex value for blue. Note that the leading `#` is **required**, and so you’ll need to enclose the entire value in double quotes.

Note that not all terminals support 24-bit RGB colors (or even ANSI colors). See Section 5.2.2 [Theme Troubleshooting], page 22,

5.2.2 My Theme Isn’t Working!

Are you using 24-bit color values? If you are, and you’re sure your terminal supports 24-bit ANSI colors, then it may be that the underlying S-Lang library just isn’t detecting support for it properly. Try doing this before launching Benben:

```
export COLORTERM=truecolor
```

This will force it to treat the terminal as support 24-bit colors.

5.3 Example Theme File

```
bg-color: 0
fg-color: 250
header-color: 255
err-color: 196
cur-song-color: 201
prev-song-color: 242
next-song-color-1: 244
next-song-color-2: 239
next-song-color-3: 236
song-queue-box-color: 250
song-queue-header-color: 99

progress-bar-char: "*"
progress-bar-space-char: "-"
progress-bar-colors: [[170, 170, 170]]
```

[2] `https://en.wikipedia.org/wiki/ANSI_escape_code#8-bit`

```
vu-clip-color: 196
vu-clipped-channel-time: 1
vu-bar-character: "\u25A0"
vu-tip-character: "\u25B6"
vu-colors:
  - 105
  - 117
  - 119
  - 190
  - 206
  - 161
```

Index