

---

# **unsilence**

**Tim-Luca Lagmöller**

**Jan 16, 2023**



# CONTENTS

|          |                            |           |
|----------|----------------------------|-----------|
| <b>1</b> | <b>Contents</b>            | <b>1</b>  |
| 1.1      | Getting started . . . . .  | 1         |
| 1.2      | API Reference . . . . .    | 2         |
| <b>2</b> | <b>Indices and tables</b>  | <b>15</b> |
| <b>3</b> | <b>References</b>          | <b>17</b> |
|          | <b>Python Module Index</b> | <b>19</b> |
|          | <b>Index</b>               | <b>21</b> |



## CONTENTS

## 1.1 Getting started

### 1.1.1 Installation

To install unsilence as a command line tool, you can use `pipx`.

```
# Installing pipx
$ pip install pipx

# Installing Unsilence as Command Line Software
$ pipx install unsilence

# If pipx asks you to, you also need to execute the following line
# as well as close and reopen your terminal window
$ pipx ensurepath
```

If you just want to use it as a python library, you could install it using `pip`.

```
# Installing Unsilence as Command Line Software
$ pip install unsilence
```

To install the command line tool directly from the GitHub source, you can use this:

```
# Clone the repository (stable branch)
$ git clone -b master https://github.com/lagmoellertim/unsilence.git unsilence

# Change Directory
$ cd unsilence

# Install pip packages
$ pip install -r requirements.txt
$ pip install pipx

# If pipx asks you to, you also need to execute the following line
# as well as close and reopen your terminal window
$ pipx ensurepath

# Install unsilence package
$ pipx install .
```

To install the library from the GitHub source, you can use this:

```
# Clone the repository (stable branch)
$ git clone -b master https://github.com/lagmoellertim/unsilence.git unsilence

#Change Directory
$ cd unsilence

# Install pip packages
$ pip install -r requirements.txt

# Install unsilence package
$ python3 setup.py install
```

## 1.2 API Reference

This page contains auto-generated API reference documentation<sup>1</sup>.

### 1.2.1 unsilence

#### Subpackages

`unsilence.command_line`

#### Submodules

`unsilence.command_line.ChoiceDialog`

#### Module Contents

#### Functions

---

|   |                                       |
|---|---------------------------------------|
| <code>choice_dialog(console, message[, default])</code> | A simple yes/no console dialog option |
|---|---------------------------------------|

---

`unsilence.command_line.ChoiceDialog.choice_dialog(console: rich.console.Console, message: str, default: bool = None)`

A simple yes/no console dialog option :param console: rich.Console Instance :param message: Message should be asked :param default: Default value when enter is pressed without an input (None, True, False) :return: Answer (True, False)

---

<sup>1</sup> Created with sphinx-autoapi

**unsilence.command\_line.EntryPoint****Module Contents****Functions**

|               |  |
|---------------|--|
| <i>main()</i> | Entry Point if this script is run as a script instead of a library |
| <i>run()</i>  | Run the Console Interface for Unsilence                            |

**unsilence.command\_line.EntryPoint.main()**

Entry Point if this script is run as a script instead of a library :return: None

**unsilence.command\_line.EntryPoint.run()**

Run the Console Interface for Unsilence :return: None

**unsilence.command\_line.ParseArguments****Module Contents****Functions**

|   |  |
|---|--|
| <i>convert_to_path</i> ([should_exist, should_parents_exist]) | Sets options for the nested class  |
| <i>number_bigger_than_zero</i> (s)                            | Returns the Number representation of s if it is bigger than zero, else an error occurs |
| <i>parse_arguments</i> ()                                     | Parses console arguments for the Unsilence Console Interface                           |

**unsilence.command\_line.ParseArguments.convert\_to\_path**(*should\_exist=True, should\_parents\_exist=True*)

Sets options for the nested class :param should\_exist: If the file needs to exists :param should\_parents\_exist: If the files parents need to exist :return: Nested Handle Function

**unsilence.command\_line.ParseArguments.number\_bigger\_than\_zero**(s)

Returns the Number representation of s if it is bigger than zero, else an error occurs :param s: Input string :return: Integer or None

**unsilence.command\_line.ParseArguments.parse\_arguments**()

Parses console arguments for the Unsilence Console Interface :return: List of Console Line Arguments

## unsilence

---

`unsilence.command_line.PrettyTimeEstimate`

### Module Contents

#### Functions

---

|   |  |
|---|--|
| <code><i>format_timedelta</i>(seconds)</code>       | Generates a pretty time representation for seconds (format: hour:minute:second)  |
| <code><i>pretty_time_estimate</i>(time_data)</code> | Generates a <code>rich.table.Table</code> object from the <code>time_data</code> dict (from <code>lib.Intervals.TimeCalculations.calculate_time</code> ) |

---

`unsilence.command_line.PrettyTimeEstimate.format_timedelta(seconds)`

Generates a pretty time representation for seconds (format: hour:minute:second) :param seconds: Amount of seconds (can be negative) :return: String representation

`unsilence.command_line.PrettyTimeEstimate.pretty_time_estimate(time_data: dict)`

Generates a `rich.table.Table` object from the `time_data` dict (from `lib.Intervals.TimeCalculations.calculate_time`) :param time\_data: `time_data` dict (from `lib.Intervals.TimeCalculations.calculate_time`) :return: `rich.table.Table` object

`unsilence.command_line.TerminalSupport`

### Module Contents

#### Functions

---

|                                      |
|--------------------------------------|
| <code><i>repair_console</i>()</code> |
|--------------------------------------|

---

`unsilence.command_line.TerminalSupport.repair_console()`

`unsilence.lib`

### Subpackages

`unsilence.lib.detect_silence`

### Submodules

`unsilence.lib.detect_silence.DetectSilence`

### Module Contents



## Functions

---

|   |  |
|---|--|
| <code>detect_silence(input_file, **kwargs)</code> | Detects silence in a file and outputs the intervals (silent/not silent) as a <code>lib.Intervals.Intervals</code> object |
|---|--|

---

`unsilence.lib.detect_silence.DetectSilence.detect_silence(input_file: pathlib.Path, **kwargs)`

Detects silence in a file and outputs the intervals (silent/not silent) as a `lib.Intervals.Intervals` object :param input\_file: File where silence should be detected :param kwargs: Various Parameters, see below :return: `lib.Intervals.Intervals` object

**kwargs:**

silence\_level: Threshold of what should be classified as silent/audible (default -35) (in dB)  
 silence\_time\_threshold: Resolution of the ffmpeg detection algorithm (default 0.5) (in seconds)  
 short\_interval\_threshold : The shortest allowed interval length (default: 0.3) (in seconds)  
 stretch\_time: Time the interval should be enlarged/shrunk (default 0.25) (in seconds)  
 on\_silence\_detect\_progress\_update: Function that should be called on progress update

(called like: func(current, total))

`unsilence.lib.intervals`

## Submodules

`unsilence.lib.intervals.Interval`

## Module Contents

## Classes

---

|                       |   |
|-----------------------|---|
| <code>Interval</code> | Represents a section in time where the media file is either silent or audible |
|-----------------------|---|

---

**class** `unsilence.lib.intervals.Interval.Interval(start=0, end=0, is_silent=False)`

Represents a section in time where the media file is either silent or audible

Initializes an Interval object :param start: Start time of the interval in seconds :param end: End time of the interval in seconds :param is\_silent: Whether the interval is silent or not

**property start**

Get the start time :return: start time in seconds

**property end**

Get the end time :return: end time in seconds

**property duration**

Returns the duration of the interval :return: Duration of the interval

**enlarge\_audible\_interval** (*stretch\_time, is\_start\_interval=False, is\_end\_interval=False*)

Enlarges/Shrinks the audio interval, based on if it is silent or not :param stretch\_time: Time the interval should be enlarged/shrunk :param is\_start\_interval: Whether the current interval is at the start (should not enlarge/shrink) :param is\_end\_interval: Whether the current interval is at the end (should not enlarge/shrink) :return: None

**copy()**

Creates a deep copy of this Interval :return: Interval deepcopy

**serialize()**

Serializes the current interval into a dict format :return: serialized dict

**static deserialize(serialized\_obj: dict)**

Deserializes a previously serializes Interval and generates a new Interval with this data :param serialized\_obj: previously serializes Interval (type dict) :return: Interval

**\_\_repr\_\_()**

String representation :return: String representation

**unsilence.lib.intervals.Intervals****Module Contents****Classes**

---

|                  |                                      |
|------------------|--------------------------------------|
| <i>Intervals</i> | Collection of lib.Intervals.Interval |
|------------------|--------------------------------------|

---

**class unsilence.lib.intervals.Intervals(interval\_list: list = None)**

Collection of lib.Intervals.Interval

Initializes a new Interval Collection :param interval\_list: list of intervals, optional

**property intervals**

Returns the list of intervals :return:

**add\_interval(interval)**

Adds an interval to the collection :param interval: interval to be added :return: None

**optimize(short\_interval\_threshold=0.3, stretch\_time=0.25)**

Optimizes the Intervals to be a better fit for media cutting :param short\_interval\_threshold: The shortest allowed interval length (in seconds) :param stretch\_time: The time that should be added/removed from a audible/silent interval :return: None

**\_\_combine\_intervals(short\_interval\_threshold)**

Combines multiple intervals in order to remove intervals smaller than a threshold :param short\_interval\_threshold: Threshold for the shortest allowed interval :return: None

**\_\_enlarge\_audible\_intervals(stretch\_time)**

Enlarges/Shrinks intervals based on if they are silent or audible :param stretch\_time: Time the intervals should be enlarged/shrunk :return: None

**remove\_short\_intervals\_from\_start(audible\_speed=1, silent\_speed=2)**

Removes Intervals from start that are shorter than 0.5 seconds after speedup to avoid having a final output without an audio track :param audible\_speed: The speed at which the audible intervals get played back at (float) :param silent\_speed: The speed at which the silent intervals get played back at (float) :return: The new, possibly shorter, Intervals object

**copy()**

Creates a deep copy :return: Deep copy of Intervals

**serialize()**

Serializes this collection :return: Serialized list

**static deserialize(serialized\_obj)**

Deserializes a previously serialized object and creates a new Instance from it :param serialized\_obj: Serialized list :return: New instance of Intervals

**\_\_repr\_\_()**

String representation :return: String representation

**unsilence.lib.intervals.TimeCalculations****Module Contents****Functions**


---

|  |                             |  |
|--|-----------------------------|--|
| <code>calculate_time(intervals,</code> | <code>audible_speed,</code> | Generates a time estimate on the time saved if the current |
| <code>silent_speed)</code>             |                             | speed settings get applied                                 |

---

`unsilence.lib.intervals.TimeCalculations.calculate_time(intervals:`

`unsilence.lib.intervals.Intervals.Intervals,`  
`audible_speed: float, silent_speed: float)`

Generates a time estimate on the time saved if the current speed settings get applied :param intervals: Intervals which should be estimated (lib.Intervals.Intervals) :param audible\_speed: The speed at which audible intervals should be played back at :param silent\_speed: The speed at which silent intervals should be played back at :return: Time calculation dict

**unsilence.lib.render\_media****Submodules****unsilence.lib.render\_media.MediaRenderer****Module Contents****Classes**


---

|                            |  |
|----------------------------|--|
| <code>MediaRenderer</code> | The Media Renderer handles the rendering of Intervals objects, so it processes the complete video and concatenates |
|----------------------------|--|

---

**class** `unsilence.lib.render_media.MediaRenderer.MediaRenderer(temp_path: pathlib.Path)`

The Media Renderer handles the rendering of Intervals objects, so it processes the complete video and concatenates the different intervals at the end

Initializes a new MediaRenderer Object :param temp\_path: The temp path where all temporary files should be stored

**render**(*input\_file: pathlib.Path, output\_file: pathlib.Path, intervals: unsilence.lib.intervals.Intervals.Intervals, \*\*kwargs*)

Renders an input\_file and writes the final output to output\_file :param input\_file: The file that should be processed :param output\_file: Where the processed file should be saved :param intervals: The Intervals that should be processed :param kwargs: Keyword Args, see below :return: None

**kwargs:**

audio\_only: Whether the output should be audio only (bool) audible\_speed: The speed at which the audible intervals get played back at (float) silent\_speed: The speed at which the silent intervals get played back at (float) audible\_volume: The volume at which the audible intervals get played back at (float) silent\_volume: The volume at which the silent intervals get played back at (float) drop\_corrupted\_intervals: Whether corrupted video intervals should be discarded or tried to recover (bool) threads: Number of threads to render simultaneously (int > 0) on\_render\_progress\_update: Function that should be called on render progress update

(called like: func(current, total))

**on\_concat\_progress\_update: Function that should be called on concat progress update**

(called like: func(current, total))

**static \_\_concat\_intervals**(*file\_list: list, concat\_file: pathlib.Path, output\_file: pathlib.Path, update\_concat\_progress*)

Concatenates all interval files to create a finished file :param file\_list: List of interval files :param concat\_file: Where the ffmpeg concat filter file should be saved :param output\_file: Where the final output file should be saved :param update\_concat\_progress: A function that is called when a step is finished

(called like function(current, total))

**Returns**

None

## unsilence.lib.render\_media.RenderIntervalThread

### Module Contents

#### Classes

---

*RenderIntervalThread*

Worker thread that can render/process intervals based on defined options

---

```
class unsilence.lib.render_media.RenderIntervalThread.RenderIntervalThread(thread_id,
                                                                            input_file:
                                                                            pathlib.Path,
                                                                            render_options:
                                                                            types.SimpleNamespace,
                                                                            task_queue:
                                                                            queue.Queue,
                                                                            thread_lock:
                                                                            threading.Lock,
                                                                            **kwargs)
```

Bases: threading.Thread

Worker thread that can render/process intervals based on defined options

Initializes a new Worker (is run in daemon mode) :param thread\_id: ID of this thread :param input\_file: The file the worker should work on :param render\_options: The parameters on how the video should be processed, more details below :param task\_queue: A queue object where the worker can get more tasks :param thread\_lock: A thread lock object to acquire and release thread locks :param kwargs: Keyword Args, see below for more information

#### **run()**

Start the worker. Worker runs until stop() is called. It runs in a loop, takes a new task if available, and processes it :return: None

#### **stop()**

Stops the worker after its current task is finished :return:

**\_\_render\_interval**(*interval\_output\_file: pathlib.Path, interval: unsilence.lib.intervals.Interval.Interval, apply\_filter=True, drop\_corrupted\_intervals=False, minimum\_interval\_duration=0.25*)

Renders an interval with the given render options :param interval\_output\_file: Where the current output file should be saved :param interval: The current Interval that should be processed :param apply\_filter: Whether the AV-Filter should be applied or if the media interval should be left untouched :param drop\_corrupted\_intervals: Whether to remove corrupted frames from the video or keep them in unedited :return: Whether it is corrupted or not

**\_\_generate\_command**(*interval\_output\_file: pathlib.Path, interval: unsilence.lib.intervals.Interval.Interval, apply\_filter: bool, minimum\_interval\_duration: float*)

Generates the ffmpeg command to process the video :param interval\_output\_file: Where the media interval should be saved :param interval: The current interval :param apply\_filter: Whether a filter should be applied or not :return: ffmpeg console command

**static clamp\_speed**(*duration: float, speed: float, minimum\_interval\_duration=0.25*)

## **unsilence.lib.tools**

### **Submodules**

**unsilence.lib.tools.ffmpeg\_version**

### **Module Contents**

### **Functions**

---

*is\_ffmpeg\_usable()*

---

**unsilence.lib.tools.ffmpeg\_version.is\_ffmpeg\_usable()**

## Submodules

`unsilence.Unsilence`

## Module Contents

## Classes

---

|                  |   |
|------------------|---|
| <i>Unsilence</i> | Unsilence Class to remove (or isolate or many other use cases) silence from audible video parts |
|------------------|---|

---

**class** `unsilence.Unsilence.Unsilence`(*input\_file*: `pathlib.Path`, *temp\_dir*: `pathlib.Path` = `Path('.tmp')`)

Unsilence Class to remove (or isolate or many other use cases) silence from audible video parts

### Parameters

- **input\_file** (*Path*) – The file that should be processed
- **temp\_dir** (*Path*) – The temp dir where temporary files can be saved

**detect\_silence**(\*\**kwargs*)

Detects silence of the file (Options can be specified in kwargs)

### Parameters

**\*\*kwargs** – Remaining keyword arguments are passed to `detect_silence()`

### Returns

A generated Intervals object

### Return type

`Intervals`

**set\_intervals**(*intervals*: `unsilence.lib.intervals.Intervals.Intervals`)

Set the intervals so that they do not need to be re-detected

### Parameters

**intervals** (`Intervals`) – Intervals collection

### Returns

None

**get\_intervals**()

Get the current Intervals so they can be reused if wanted

### Returns

Intervals collection

### Return type

`Intervals`

**estimate\_time**(*audible\_speed*: `float` = 1, *silent\_speed*: `float` = 6)

Estimates the time (savings) when the current options are applied to the intervals

### Parameters

- **audible\_speed** (*float*) – The speed at which the audible intervals get played back at
- **silent\_speed** (*float*) – The speed at which the silent intervals get played back at

**Raises**

**ValueError** – If silence detection was never run

**Returns**

Dictionary of time information

**Return type**

dict

**render\_media**(*output\_file*: *pathlib.Path*, *\*\*kwargs*)

Renders the current intervals with options specified in the kwargs

**Parameters**

- **output\_file** (*Path*) – Where the final file should be saved at
- **\*\*kwargs** – Remaining keyword arguments are passed to [render\(\)](#)

**Returns**

None

**cleanup()**

Cleans up the temporary directories, called automatically when the program ends

**Returns**

None

**unsilence.\_\_main\_\_**

## Package Contents

### Classes

|                                  |   |
|----------------------------------|---|
| <a href="#"><i>Unsilence</i></a> | Unsilence Class to remove (or isolate or many other use cases) silence from audible video parts |
| <a href="#"><i>Interval</i></a>  | Represents a section in time where the media file is either silent or audible                   |
| <a href="#"><i>Intervals</i></a> | Collection of <code>lib.Intervals.Interval</code>   |

**class** `unsilence.Unsilence`(*input\_file*: *pathlib.Path*, *temp\_dir*: *pathlib.Path* = *Path('.tmp')*)

Unsilence Class to remove (or isolate or many other use cases) silence from audible video parts

**Parameters**

- **input\_file** (*Path*) – The file that should be processed
- **temp\_dir** (*Path*) – The temp dir where temporary files can be saved

**detect\_silence**(*\*\*kwargs*)

Detects silence of the file (Options can be specified in kwargs)

**Parameters**

**\*\*kwargs** – Remaining keyword arguments are passed to [detect\\_silence\(\)](#)

**Returns**

A generated Intervals object

**Return type**

[\*Intervals\*](#)

**set\_intervals**(*intervals*: [unsilence.lib.intervals.Intervals.Intervals](#))

Set the intervals so that they do not need to be re-detected

**Parameters**

**intervals** ([Intervals](#)) – Intervals collection

**Returns**

None

**get\_intervals**()

Get the current Intervals so they can be reused if wanted

**Returns**

Intervals collection

**Return type**

[Intervals](#)

**estimate\_time**(*audible\_speed*: *float* = 1, *silent\_speed*: *float* = 6)

Estimates the time (savings) when the current options are applied to the intervals

**Parameters**

- **audible\_speed** (*float*) – The speed at which the audible intervals get played back at
- **silent\_speed** (*float*) – The speed at which the silent intervals get played back at

**Raises**

**ValueError** – If silence detection was never run

**Returns**

Dictionary of time information

**Return type**

dict

**render\_media**(*output\_file*: [pathlib.Path](#), *\*\*kwargs*)

Renders the current intervals with options specified in the kwargs

**Parameters**

- **output\_file** (*Path*) – Where the final file should be saved at
- **\*\*kwargs** – Remaining keyword arguments are passed to [render\(\)](#)

**Returns**

None

**cleanup**()

Cleans up the temporary directories, called automatically when the program ends

**Returns**

None

**class** [unsilence.Interval](#)(*start*=0, *end*=0, *is\_silent*=False)

Represents a section in time where the media file is either silent or audible

Initializes an Interval object :param start: Start time of the interval in seconds :param end: End time of the interval in seconds :param is\_silent: Whether the interval is silent or not

**property start**

Get the start time :return: start time in seconds



**property end**

Get the end time :return: end time in seconds

**property duration**

Returns the duration of the interval :return: Duration of the interval

**enlarge\_audible\_interval**(*stretch\_time*, *is\_start\_interval=False*, *is\_end\_interval=False*)

Enlarges/Shrinks the audio interval, based on if it is silent or not :param stretch\_time: Time the interval should be enlarged/shrunk :param is\_start\_interval: Whether the current interval is at the start (should not enlarge/shrink) :param is\_end\_interval: Whether the current interval is at the end (should not enlarge/shrink) :return: None

**copy()**

Creates a deep copy of this Interval :return: Interval deepcopy

**serialize()**

Serializes the current interval into a dict format :return: serialized dict

**static deserialize**(*serialized\_obj: dict*)

Deserializes a previously serializes Interval and generates a new Interval with this data :param serialized\_obj: previously serializes Interval (type dict) :return: Interval

**\_\_repr\_\_()**

String representation :return: String representation

**class unsilence.Intervals**(*interval\_list: list = None*)

Collection of lib.Intervals.Interval

Initializes a new Interval Collection :param interval\_list: list of intervals, optional

**property intervals**

Returns the list of intervals :return:

**add\_interval**(*interval*)

Adds an interval to the collection :param interval: interval to be added :return: None

**optimize**(*short\_interval\_threshold=0.3*, *stretch\_time=0.25*)

Optimizes the Intervals to be a better fit for media cutting :param short\_interval\_threshold: The shortest allowed interval length (in seconds) :param stretch\_time: The time that should be added/removed from a audible/silent interval :return: None

**\_\_combine\_intervals**(*short\_interval\_threshold*)

Combines multiple intervals in order to remove intervals smaller than a threshold :param short\_interval\_threshold: Threshold for the shortest allowed interval :return: None

**\_\_enlarge\_audible\_intervals**(*stretch\_time*)

Enlarges/Shrinks intervals based on if they are silent or audible :param stretch\_time: Time the intervals should be enlarged/shrunk :return: None

**remove\_short\_intervals\_from\_start**(*audible\_speed=1*, *silent\_speed=2*)

Removes Intervals from start that are shorter than 0.5 seconds after speedup to avoid having a final output without an audio track :param audible\_speed: The speed at which the audible intervals get played back at (float) :param silent\_speed: The speed at which the silent intervals get played back at (float) :return: The new, possibly shorter, Intervals object

**copy()**

Creates a deep copy :return: Deep copy of Intervals

**serialize()**

Serializes this collection :return: Serialized list

**static deserialize(*serialized\_obj*)**

Deserializes a previously serialized object and creates a new Instance from it :param *serialized\_obj*: Serialized list :return: New instance of Intervals

**\_\_repr\_\_()**

String representation :return: String representation

## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



**REFERENCES**



## PYTHON MODULE INDEX

### U

- [unsilence](#), 2
- [unsilence.\\_\\_main\\_\\_](#), 11
- [unsilence.command\\_line](#), 2
  - [unsilence.command\\_line.ChoiceDialog](#), 2
  - [unsilence.command\\_line.EntryPoint](#), 3
  - [unsilence.command\\_line.ParseArguments](#), 3
  - [unsilence.command\\_line.PrettyTimeEstimate](#), 4
  - [unsilence.command\\_line.TerminalSupport](#), 4
- [unsilence.lib](#), 4
  - [unsilence.lib.detect\\_silence](#), 4
    - [unsilence.lib.detect\\_silence.DetectSilence](#), 4
  - [unsilence.lib.intervals](#), 5
    - [unsilence.lib.intervals.Interval](#), 5
    - [unsilence.lib.intervals.Intervals](#), 6
    - [unsilence.lib.intervals.TimeCalculations](#), 7
  - [unsilence.lib.render\\_media](#), 7
    - [unsilence.lib.render\\_media.MediaRenderer](#), 7
    - [unsilence.lib.render\\_media.RenderIntervalThread](#), 8
  - [unsilence.lib.tools](#), 9
    - [unsilence.lib.tools.ffmpeg\\_version](#), 9
  - [unsilence.Unsilence](#), 10





## Symbols

[\\_\\_combine\\_intervals\(\)](#) (*unsilence.Intervals method*), [13](#)  
[\\_\\_combine\\_intervals\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals method*), [6](#)  
[\\_\\_concat\\_intervals\(\)](#) (*unsilence.lib.render\_media.MediaRenderer.MediaRenderer static method*), [8](#)  
[\\_\\_enlarge\\_audible\\_intervals\(\)](#) (*unsilence.Intervals method*), [13](#)  
[\\_\\_enlarge\\_audible\\_intervals\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals method*), [6](#)  
[\\_\\_generate\\_command\(\)](#) (*unsilence.lib.render\_media.RenderIntervalThread.RenderIntervalThread method*), [9](#)  
[\\_\\_render\\_interval\(\)](#) (*unsilence.lib.render\_media.RenderIntervalThread.RenderIntervalThread method*), [9](#)  
[\\_\\_repr\\_\\_\(\)](#) (*unsilence.Interval method*), [13](#)  
[\\_\\_repr\\_\\_\(\)](#) (*unsilence.Intervals method*), [14](#)  
[\\_\\_repr\\_\\_\(\)](#) (*unsilence.lib.intervals.Interval.Interval method*), [6](#)  
[\\_\\_repr\\_\\_\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals method*), [7](#)

## A

[add\\_interval\(\)](#) (*unsilence.Intervals method*), [13](#)  
[add\\_interval\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals method*), [6](#)

## C

[calculate\\_time\(\)](#) (*in module unsilence.lib.intervals.TimeCalculations*), [7](#)  
[choice\\_dialog\(\)](#) (*in module unsilence.command\_line.ChoiceDialog*), [2](#)  
[clamp\\_speed\(\)](#) (*unsilence.lib.render\_media.RenderIntervalThread.RenderIntervalThread static method*), [9](#)  
[cleanup\(\)](#) (*unsilence.Unsilence method*), [12](#)

[cleanup\(\)](#) (*unsilence.Unsilence.Unsilence method*), [11](#)  
[convert\\_to\\_path\(\)](#) (*in module unsilence.command\_line.ParseArguments*), [3](#)  
[copy\(\)](#) (*unsilence.Interval method*), [13](#)  
[copy\(\)](#) (*unsilence.Intervals method*), [13](#)  
[copy\(\)](#) (*unsilence.lib.intervals.Interval.Interval method*), [6](#)  
[copy\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals method*), [6](#)

## D

[deserialize\(\)](#) (*unsilence.Interval static method*), [13](#)  
[deserialize\(\)](#) (*unsilence.Intervals static method*), [14](#)  
[deserialize\(\)](#) (*unsilence.lib.intervals.Interval.Interval static method*), [6](#)  
[deserialize\(\)](#) (*unsilence.lib.intervals.Intervals.Intervals static method*), [7](#)  
[detect\\_silence\(\)](#) (*in module unsilence.lib.detect\_silence.DetectSilence*), [5](#)  
[detect\\_silence\(\)](#) (*unsilence.Unsilence method*), [11](#)  
[detect\\_silence\(\)](#) (*unsilence.Unsilence.Unsilence method*), [10](#)  
[duration](#) (*unsilence.Interval property*), [13](#)  
[duration](#) (*unsilence.lib.intervals.Interval.Interval property*), [5](#)

## E

[end](#) (*unsilence.Interval property*), [12](#)  
[end](#) (*unsilence.lib.intervals.Interval.Interval property*), [5](#)  
[enlarge\\_audible\\_interval\(\)](#) (*unsilence.Interval method*), [13](#)  
[enlarge\\_audible\\_interval\(\)](#) (*unsilence.lib.intervals.Interval.Interval method*), [5](#)  
[estimate\\_time\(\)](#) (*unsilence.Unsilence method*), [12](#)  
[estimate\\_time\(\)](#) (*unsilence.Unsilence.Unsilence method*), [10](#)

## F

[format\\_timedelta\(\)](#) (*in module unsilence.command\_line.PrettyTimeEstimate*), [12](#)

4

## G

`get_intervals()` (*unsilence.Unsilence* method), 12  
`get_intervals()` (*unsilence.Unsilence.Unsilence* method), 10

## I

`Interval` (class in *unsilence*), 12  
`Interval` (class in *unsilence.lib.intervals.Interval*), 5  
`Intervals` (class in *unsilence*), 13  
`Intervals` (class in *unsilence.lib.intervals.Intervals*), 6  
`intervals` (*unsilence.Intervals* property), 13  
`intervals` (*unsilence.lib.intervals.Intervals.Intervals* property), 6  
`is_ffmpeg_usable()` (in module *unsilence.lib.tools.ffmpeg\_version*), 9

## M

`main()` (in module *unsilence.command\_line.EntryPoint*), 3  
`MediaRenderer` (class in *unsilence.lib.render\_media.MediaRenderer*), 7

module

*unsilence*, 2  
`unsilence.__main__`, 11  
`unsilence.command_line`, 2  
`unsilence.command_line.ChoiceDialog`, 2  
`unsilence.command_line.EntryPoint`, 3  
`unsilence.command_line.ParseArguments`, 3  
`unsilence.command_line.PrettyTimeEstimate`, 4  
`unsilence.command_line.TerminalSupport`, 4  
`unsilence.lib`, 4  
`unsilence.lib.detect_silence`, 4  
`unsilence.lib.detect_silence.DetectSilence`, 4  
`unsilence.lib.intervals`, 5  
`unsilence.lib.intervals.Interval`, 5  
`unsilence.lib.intervals.Intervals`, 6  
`unsilence.lib.intervals.TimeCalculations`, 7  
`unsilence.lib.render_media`, 7  
`unsilence.lib.render_media.MediaRenderer`, 7  
`unsilence.lib.render_media.RenderIntervalThread`, 8  
`unsilence.lib.tools`, 9  
`unsilence.lib.tools.ffmpeg_version`, 9  
`unsilence.Unsilence`, 10

## N

`number_bigger_than_zero()` (in module *unsilence.command\_line.ParseArguments*), 3

## O

`optimize()` (*unsilence.Intervals* method), 13  
`optimize()` (*unsilence.lib.intervals.Intervals.Intervals* method), 6

## P

`parse_arguments()` (in module *unsilence.command\_line.ParseArguments*), 3  
`pretty_time_estimate()` (in module *unsilence.command\_line.PrettyTimeEstimate*), 4

## R

`remove_short_intervals_from_start()` (*unsilence.Intervals* method), 13  
`remove_short_intervals_from_start()` (*unsilence.lib.intervals.Intervals.Intervals* method), 6  
`render()` (*unsilence.lib.render\_media.MediaRenderer.MediaRenderer* method), 7  
`render_media()` (*unsilence.Unsilence* method), 12  
`render_media()` (*unsilence.Unsilence.Unsilence* method), 11  
`RenderIntervalThread` (class in *unsilence.lib.render\_media.RenderIntervalThread*), 8  
`repair_console()` (in module *unsilence.command\_line.TerminalSupport*), 4  
`run()` (in module *unsilence.command\_line.EntryPoint*), 3  
`run()` (*unsilence.lib.render\_media.RenderIntervalThread.RenderIntervalThread* method), 9

## S

`serialize()` (*unsilence.Interval* method), 13  
`serialize()` (*unsilence.Intervals* method), 13  
`serialize()` (*unsilence.lib.intervals.Interval.Interval* method), 6  
`serialize()` (*unsilence.lib.intervals.Intervals.Intervals* method), 6  
`set_intervals()` (*unsilence.Unsilence* method), 11  
`set_intervals()` (*unsilence.Unsilence.Unsilence* method), 10  
`start` (*unsilence.Interval* property), 12  
`start` (*unsilence.lib.intervals.Interval.Interval* property), 5  
`stop()` (*unsilence.lib.render\_media.RenderIntervalThread.RenderIntervalThread* method), 9

## U

- unsilence
  - module, 2
- Unsilence (*class in unsilence*), 11
- Unsilence (*class in unsilence.Unsilence*), 10
- unsilence.\_\_main\_\_
  - module, 11
- unsilence.command\_line
  - module, 2
- unsilence.command\_line.ChoiceDialog
  - module, 2
- unsilence.command\_line.EntryPoint
  - module, 3
- unsilence.command\_line.ParseArguments
  - module, 3
- unsilence.command\_line.PrettyTimeEstimate
  - module, 4
- unsilence.command\_line.TerminalSupport
  - module, 4
- unsilence.lib
  - module, 4
- unsilence.lib.detect\_silence
  - module, 4
- unsilence.lib.detect\_silence.DetectSilence
  - module, 4
- unsilence.lib.intervals
  - module, 5
- unsilence.lib.intervals.Interval
  - module, 5
- unsilence.lib.intervals.Intervals
  - module, 6
- unsilence.lib.intervals.TimeCalculations
  - module, 7
- unsilence.lib.render\_media
  - module, 7
- unsilence.lib.render\_media.MediaRenderer
  - module, 7
- unsilence.lib.render\_media.RenderIntervalThread
  - module, 8
- unsilence.lib.tools
  - module, 9
- unsilence.lib.tools.ffmpeg\_version
  - module, 9
- unsilence.Unsilence
  - module, 10