\_rain

Generated by Doxygen 1.8.14

# **Contents**

1	_Rai	in			1
2	Clas	s Index			3
	2.1	Class I	_ist		3
3	File	Index			5
	3.1	File Lis	st		5
4	Clas	s Docu	mentatior	n	7
	4.1	Biquad	l Class Re	eference	7
		4.1.1	Detailed	Description	9
		4.1.2	Member	Function Documentation	9
			4.1.2.1	process()	9
	4.2	GrainP	layer Clas	ss Reference	9
		4.2.1	Detailed	Description	12
		4.2.2	Construc	ctor & Destructor Documentation	12
			4.2.2.1	GrainPlayer()	12
		4.2.3	Member	Function Documentation	12
			4.2.3.1	get_sample()	12
			4.2.3.2	set_funk()	13
			4.2.3.3	set_length()	13
			4.2.3.4	set_mode()	13
			4.2.3.5	set_pitch()	13
			4.2.3.6	set_relative_distance()	14
			4.2.3.7	set reset()	14

ii CONTENTS

		4.2.3.8	set_spread()	14
		4.2.3.9	set_start()	15
4.3	OscMa	an Class R	eference	15
	4.3.1	Detailed	Description	18
	4.3.2	Construc	etor & Destructor Documentation	18
		4.3.2.1	OscMan()	18
	4.3.3	Member	Function Documentation	19
		4.3.3.1	bit_callback()	19
		4.3.3.2	cutoff_callback()	19
		4.3.3.3	funky_callback()	20
		4.3.3.4	get_bit()	20
		4.3.3.5	get_cutoff()	21
		4.3.3.6	get_funky()	21
		4.3.3.7	get_length()	21
		4.3.3.8	get_mode()	21
		4.3.3.9	get_pitch()	22
		4.3.3.10	get_reset()	22
		4.3.3.11	get_spread()	22
		4.3.3.12	get_start()	22
		4.3.3.13	get_texture()	23
		4.3.3.14	length_callback()	23
		4.3.3.15	mode_callback()	23
		4.3.3.16	pitch_callback()	24
		4.3.3.17	q_callback()	24
		4.3.3.18	reset_callback()	26
		4.3.3.19	spread_callback()	26
		4.3.3.20	start_callback()	28
		4.3.3.21	texture_callback()	28
		4.3.3.22	win_callback()	30
4.4	Rain C	lass Refer	rence	30

CONTENTS

	4.4.1	Detailed	Description	33
	4.4.2	Construc	etor & Destructor Documentation	33
		4.4.2.1	Rain()	33
	4.4.3	Member	Function Documentation	33
		4.4.3.1	callback_process()	33
		4.4.3.2	crush()	34
		4.4.3.3	process()	34
		4.4.3.4	set_gain()	34
4.5	Single	Sample Cl	ass Reference	35
	4.5.1	Detailed	Description	36
	4.5.2	Construc	tor & Destructor Documentation	37
		4.5.2.1	SingleSample()	37
	4.5.3	Member	Function Documentation	38
		4.5.3.1	get_sample() [1/2]	38
		4.5.3.2	get_sample() [2/2]	38
		4.5.3.3	get_x()	38
		4.5.3.4	read_wavefile()	39
4.6	Triangu	ularWindov	v Class Reference	39
	4.6.1	Detailed	Description	41
	4.6.2	Member	Function Documentation	41
		4.6.2.1	get_playback_mode()	41
		4.6.2.2	get_relative_position()	41
		4.6.2.3	get_value()	41
		4.6.2.4	initialize()	41
		4.6.2.5	move_relative_position() [1/2]	42
		4.6.2.6	move_relative_position() [2/2]	42
		4.6.2.7	set_length()	42
		4.6.2.8	set_playback_mode()	43
		4.6.2.9	set_start()	43
		4.6.2.10	step_backward()	43
		4.6.2.11	step_forward()	44

iv CONTENTS

5	File I	Documentation	45
	5.1	/Users/marquis/Desktop/_rain/Biquad.cpp File Reference	45
		5.1.1 Detailed Description	45
	5.2	/Users/marquis/Desktop/_rain/Biquad.h File Reference	46
		5.2.1 Detailed Description	46
	5.3	/Users/marquis/Desktop/_rain/grainplayer.cpp File Reference	47
	5.4	/Users/marquis/Desktop/_rain/grainplayer.h File Reference	47
	5.5	/Users/marquis/Desktop/_rain/main.cpp File Reference	48
		5.5.1 Detailed Description	49
	5.6	/Users/marquis/Desktop/_rain/oscman.cpp File Reference	49
	5.7	/Users/marquis/Desktop/_rain/oscman.h File Reference	49
	5.8	/Users/marquis/Desktop/_rain/rain.cpp File Reference	50
	5.9	/Users/marquis/Desktop/_rain/rain.h File Reference	50
	5.10	/Users/marquis/Desktop/_rain/singlesample.cpp File Reference	51
	5.11	/Users/marquis/Desktop/_rain/singlesample.h File Reference	52
	5.12	/Users/marquis/Desktop/_rain/triangularwindow.cpp File Reference	53
	5.13	/Users/marquis/Desktop/_rain/triangularwindow.h File Reference	54
Inc	dex		55

# **Chapter 1**

# Rain

# **BRIEF**

# What is \_\_rain\_

- · Granular Synth with some extra features running on Rhaspberry PI
- · loop-based Grainplayer
- · Master section with filter and bitcrusher
- · Control via OSC

# **Features Grainplayer**

- · position: start sample of the section
- · Loop length: length of the section in samples
- · Loop mode infinite forward infinite backwards infinite forward / backward
- Playback speed (pitch)
- · Grain length: length of the window

# \_rain modes

- forward: Grainplayer is running forward
- · backward: Grainplayer is running backwards
- forward/backward: Grainplayer is running forward/backwards
- funk (optional): randomizes propagation

# texture / choppy

· controls how far the grains jump during playback

2 \_Rain

# color

• Lowpass (+ Bitcrusher): filters the overtones of the master signal with lowpass, distort the filtered signal with bitcrusher

# Libraries

The following libraries are used within this project:

- JACK
- · libsndfile
- liblo

# Usage

The binaray granular\_example expects three command line arguments:

- the wav file to read with the flag '-f'
- the grain size in samples '-l'

~/rain \$ ./rain -f ../../wav/dopesample.wav -l 1000

# Chapter 2

# **Class Index**

# 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Biquad		
	A biquad filter with variable Filter Type, Q, fc, Gain	7
GrainPla	yer	
	The granular player class, holding a SingleSound and an array of window functions, representing the grains	9
OscMan		
	Class which parses the incoming OSC messages	15
Rain		
	Read a wave file and play it back in a granular style, with modification through OSC	30
SingleSa	ample	
	Class which holds the waveform of a sample in an array	35
Triangula	arWindow	
	A single window (envelope) for granular synthesis	39

4 Class Index

# **Chapter 3**

# File Index

# 3.1 File List

Here is a list of all documented files with brief descriptions:

/Users/marquis/Desktop/_rain/Biquad.cpp
/Users/marquis/Desktop/_rain/Biquad.h
/Users/marquis/Desktop/_rain/grainplayer.cpp
/Users/marquis/Desktop/_rain/grainplayer.h
/Users/marquis/Desktop/_rain/main.cpp
Granular Synth with some extra features running on Rhaspberry PI
/Users/marquis/Desktop/_rain/oscman.cpp
/Users/marquis/Desktop/_rain/oscman.h
/Users/marquis/Desktop/_rain/rain.cpp
/Users/marquis/Desktop/_rain/rain.h
/Users/marquis/Desktop/_rain/singlesample.cpp
/Users/marquis/Desktop/_rain/singlesample.h
/Users/marquis/Desktop/_rain/triangularwindow.cpp
/Users/marquis/Desktop/_rain/triangularwindow.h

6 File Index

# **Chapter 4**

# **Class Documentation**

# 4.1 Biquad Class Reference

A biquad filter with variable Filter Type, Q, fc, Gain.

#include <Biquad.h>

Collaboration diagram for Biquad:

# Biquad + peakGain # type # a0 # a1 # a2 # b1 # b2 # Fc # Q # z1 # z2 + Biquad() + Biquad() + ~Biquad() + setType() + setFc() + setFc() + setPeakGain() + setBiquad() + process() # calcBiquad()

#### **Public Member Functions**

```
• Biquad (int type, double Fc, double Q, double peakGainDB)
```

void setType (int type)

sets the filter type

void setQ (double Q)

sets the filter quality

void setFc (double Fc)

sets the cutoff frequency

• void setFc (double Fc, int fs)

sets the cutoff frequency in relation to the sample rate

void setPeakGain (double peakGainDB)

sets the resonance gain value

· void setBiquad (int type, double Fc, double Q, double peakGain)

sets member variables necessary for calculation of the filter coefficients

float process (float in)

filters input signal

## **Public Attributes**

· double peakGain

peakGain resonance gain of the biquad filter

#### **Protected Member Functions**

void calcBiquad (void)

calculates filter coefficients from class member variables

## **Protected Attributes**

· int type

represents the filter type

• double a0

represents the filter coefficient a0

• double a1

represents the filter coefficient a1

• double a2

represents the filter coefficient a2

double b1

represents the filter coefficient b1

double b2

represents the filter coefficient b2

· double Fc

represents filter cutoff frequency

double Q

represents filter Q

double z1

buffer for previous calculation

double z2

buffer for previous calculation

# 4.1.1 Detailed Description

A biquad filter with variable Filter Type, Q, fc, Gain.

## 4.1.2 Member Function Documentation

# 4.1.2.1 process()

filters input signal

#### **Parameters**

in is input sample

## Returns

filtered output

The documentation for this class was generated from the following files:

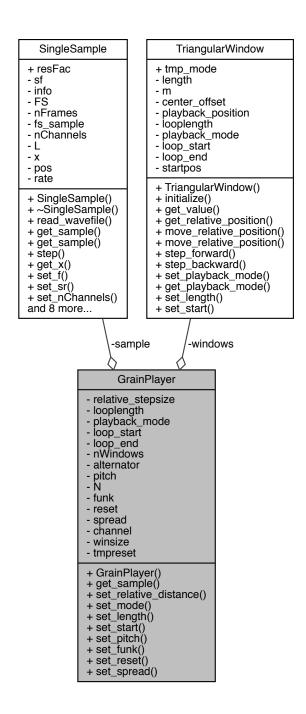
- /Users/marquis/Desktop/\_rain/Biquad.h
- /Users/marquis/Desktop/\_rain/Biquad.cpp

# 4.2 GrainPlayer Class Reference

The granular player class, holding a SingleSound and an array of window functions, representing the grains.

```
#include <grainplayer.h>
```

Collaboration diagram for GrainPlayer:



# **Public Member Functions**

- GrainPlayer (std::string filePath, int fs, int win\_size, int ch)
  - GrainPlayer the constructor.
- double get sample ()
  - get a single sample, using all active grains
- void set\_relative\_distance (double r)

is the step size of the moving playback pos.

void set\_mode (int m)

sets the playback mode: forward, backwards or forward-backward played loop

void set length (double I)

sets the looplength

void set\_start (double s)

sets the start of the loop

void set\_pitch (double p)

sets the playback speed of the loop

void set\_funk (int f)

randomized the relative position

void set\_reset (int r)

randomized the relative position

void set\_spread (double s)

randomized the relative position

# **Private Attributes**

• SingleSample \* sample

an instance of the single sample class

double relative\_stepsize

fraction of window size to shift the window position

• TriangularWindow \* windows

pointer to an array of windows

· int looplength

specifies the Length of the playbackloop

· int playback\_mode

specifies whether the window propagates in a forward or backward loop

· int loop\_start

specifies the start of the playback loop

· int loop\_end

specifies the end of the playback loop

· int nWindows

specifies how many windows contribute to the Grainplayer

· int alternator

is an auxiliary variable for forward-backward looping

· double pitch

is the playback speed

int N

is the amount of Samples corresponding to the Length of the loaded Wav File

int funk

initiates a random forward movement of the playback window

int reset

initalizes the windows of the GrainPlayer

int spread

modify the distances between the start positions of the loop in relation to each other

int channel

represents left and right channel of the output

· int winsize

represents window size

• int tmpreset = 1

auxilary variable for reset function

# 4.2.1 Detailed Description

The granular player class, holding a SingleSound and an array of window functions, representing the grains.

Author

Henrik von Coler modified by Marquis Fields and Malte Schneider

Date

2019/10/04

## 4.2.2 Constructor & Destructor Documentation

# 4.2.2.1 GrainPlayer()

GrainPlayer the constructor.

#### **Parameters**

filePath	path to the wav file to read
fs	sampling rate of the JACK server
win_size	the grain size (size of the window)
nWin	the number of grains to be used

# 4.2.3 Member Function Documentation

# 4.2.3.1 get\_sample()

```
double GrainPlayer::get_sample ( )
```

get a single sample, using all active grains

#### Returns

a sound sample for the output

```
4.2.3.2 set_funk()
```

randomized the relative position

**Parameters** 

f funk

## 4.2.3.3 set\_length()

```
void GrainPlayer::set_length ( \label{eq:condition} \mbox{double } \mbox{$I$} \mbox{ )}
```

sets the looplength

# **Parameters**

/ looplength

# 4.2.3.4 set\_mode()

sets the playback mode: forward, backwards or forward-backward played loop

## **Parameters**

m playback\_mode

# 4.2.3.5 set\_pitch()

```
void GrainPlayer::set_pitch ( \label{eq:condition} \mbox{double } p \mbox{ )}
```

sets the playback speed of the loop

Da					
ra	ra	m	eı	œ	rs



## 4.2.3.6 set\_relative\_distance()

```
void GrainPlayer::set\_relative\_distance ( double r )
```

is the step size of the moving playback pos.

#### **Parameters**

```
r relative_stepsize
```

# 4.2.3.7 set\_reset()

randomized the relative position

#### **Parameters**

r reset

# 4.2.3.8 set\_spread()

```
void GrainPlayer::set_spread ( \label{eq:condition} \mbox{double } s \mbox{ )}
```

randomized the relative position

#### **Parameters**

s spread

# 4.2.3.9 set\_start()

```
void GrainPlayer::set_start ( \label{eq:condition} \mbox{double } s \mbox{ )}
```

sets the start of the loop

## **Parameters**

s loop\_start

The documentation for this class was generated from the following files:

- /Users/marquis/Desktop/\_rain/grainplayer.h
- /Users/marquis/Desktop/\_rain/grainplayer.cpp

# 4.3 OscMan Class Reference

Class which parses the incoming OSC messages.

#include <oscman.h>

OscMan

Collaboration diagram for OscMan:

# - port - st - texture - cutoff - start\_loop - length\_loop - mode - pitch - bit - funk - reset - spread + OscMan() + get\_texture() + get\_cutoff() + get\_cutoff() + get\_start() + get\_length() + get\_mode() + get\_pitch() + get\_bit() + get\_funky() + get\_reset() + get\_spread() texture\_callback()cutoff\_callback()mode\_callback() - start\_callback() - length\_callback() - pitch\_callback() - bit\_callback() - q\_callback() - funky\_callback()

reset\_callback()
 win\_callback()
 spread\_callback()

#### **Public Member Functions**

```
    OscMan (int p)
        Constructor.
```

• double get\_texture ()

returns texture input from osc

• double get\_cutoff ()

returns filter cutoff input from osc

• double get\_start ()

returns loopstart input from osc

• double get length ()

returns looplength input from osc

int get\_mode ()

returns loopmode input from osc

• double get\_pitch ()

returns pitch input from osc

• int get bit ()

returns loopmode input from osc

· int get\_funky ()

returns the randomized relative playback position input from osc

· int get\_reset ()

toggles reset function

double get\_spread ()

returns the spread input

#### **Static Private Member Functions**

 static int texture\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user\_data)

OscMan::texture\_callback.

static int cutoff\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user
 \_\_data)

OscMan::cutoff callback.

static int mode\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user
 \_\_data)

OscMan::mode\_callback.

static int start\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user
 data)

OscMan::start\_callback.

• static int length\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user\_data)

OscMan::length\_callback.

static int pitch\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user←
 \_data)

OscMan::pitch\_callback

static int bit\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user\_
 data)

OscMan::bit\_callback.

- static int q\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user\_data)

  \*\*OscMan::q\_callback.
- static int funky\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user 
   data)

OscMan::funky\_callback.

static int reset\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user
 —data)

OscMan::reset\_callback.

static int win\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user
 \_\_data)

OscMan::win\_callback.

• static int spread\_callback (const char \*path, const char \*types, lo\_arg \*\*argv, int argc, void \*data, void \*user\_data)

OscMan::spread\_callback.

# **Private Attributes**

int port

the OSC port to be opened

lo::ServerThread \* st

OSC server thread

· double texture

texture Slider

double cutoff

cutoff Slider

double start\_loop

determines the start of the loop

double length\_loop

determines the end of the loop

• int mode

mode Slider

· double pitch

pitch Slider

int bit

determines the bit depth of the grainer

int funk

funk initiates a random forward/backward movement of the playback window

· int reset

initalize the windows of the GrainPlayer

double spread

modify the distances between the start positions of the loop in relation to each other

# 4.3.1 Detailed Description

Class which parses the incoming OSC messages.

**Author** 

Henrik von Coler

Date

2019/03/18

Author

Henrik von Coler

Date

2019/03/18\$

## 4.3.2 Constructor & Destructor Documentation

#### 4.3.2.1 OscMan()

```
OscMan::OscMan ( \inf p )
```

Constructor.

#### **Parameters**

```
p OSC port number
```

#### 4.3.3 Member Function Documentation

# 4.3.3.1 bit\_callback()

# OscMan::bit\_callback.

#### **Parameters**

path	
types	
argv	
argc	
data	
user_data	

## Returns

#### 4.3.3.2 cutoff\_callback()

## OscMan::cutoff\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.3 funky\_callback()

OscMan::funky\_callback.

# Parameters

path	
types	
argv	
argc	
data	
user_data	

Returns

```
4.3.3.4 get_bit()
```

```
int OscMan::get_bit ( )
```

returns loopmode input from osc

Returns

bit

```
4.3.3.5 get_cutoff()
double OscMan::get_cutoff ( )
returns filter cutoff input from osc
Returns
     filter cutoff
4.3.3.6 get_funky()
int OscMan::get_funky ( )
returns the randomized relative playback position input from osc
Returns
     funk
4.3.3.7 get_length()
double OscMan::get_length ( )
returns looplength input from osc
Returns
     length_loop
4.3.3.8 get_mode()
int OscMan::get_mode ( )
returns loopmode input from osc
Returns
     returns mode
```

```
4.3.3.9 get_pitch()
double OscMan::get_pitch ( )
returns pitch input from osc
Returns
     pitch
4.3.3.10 get_reset()
int OscMan::get_reset ( )
toggles reset function
Returns
     reset
4.3.3.11 get_spread()
double OscMan::get_spread ( )
returns the spread input
Returns
     spread
4.3.3.12 get_start()
double OscMan::get_start ( )
returns loopstart input from osc
Returns
     start_loop
```

```
4.3.3.13 get_texture()
```

```
double OscMan::get_texture ( )
```

returns texture input from osc

Returns

texture

# 4.3.3.14 length\_callback()

# OscMan::length\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.15 mode\_callback()

OscMan::mode\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

#### Returns

# 4.3.3.16 pitch\_callback()

# OscMan::pitch\_callback.

# Parameters

path	
types	
argv	
argc	
data	
user_data	

## Returns

# 4.3.3.17 q\_callback()

```
void * data,
void * user_data ) [static], [private]
```

OscMan::q\_callback.

## **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.18 reset\_callback()

## OscMan::reset\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.19 spread\_callback()

```
void * data,
void * user_data ) [static], [private]
```

OscMan::spread\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.20 start\_callback()

## OscMan::start\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.21 texture\_callback()

```
void * data,
void * user_data ) [static], [private]
```

OscMan::texture\_callback.

#### **Parameters**

path	
tunos	
types	
argv	
argc	
data	
user_data	

Returns

# 4.3.3.22 win\_callback()

OscMan::win\_callback.

# **Parameters**

path	
types	
argv	
argc	
data	
user_data	

Returns

The documentation for this class was generated from the following files:

- /Users/marquis/Desktop/\_rain/oscman.h
- /Users/marquis/Desktop/\_rain/oscman.cpp

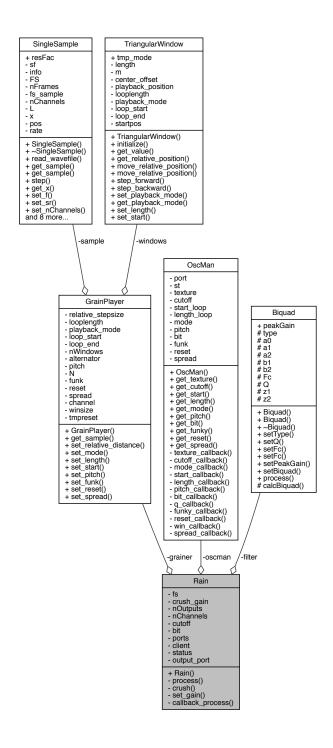
# 4.4 Rain Class Reference

Read a wave file and play it back in a granular style, with modification through OSC.

4.4 Rain Class Reference 31

#include <rain.h>

#### Collaboration diagram for Rain:



#### **Public Member Functions**

• Rain (std::string filename, int win\_size)

Constructor.

#### **Private Member Functions**

• int process (jack\_nframes\_t nframes)

get the next buffer from the sample and playback

• double crush (double sample)

reduce bit depth of input sample

void set\_gain (double fc)

sets gain of bitcrushed signale in dependence of filter cutoff frequency

#### **Static Private Member Functions**

• static int callback\_process (jack\_nframes\_t x, void \*object)

#### **Private Attributes**

• GrainPlayer \* grainer [2]

creates an GrainPlayer array with all necessary parameters

• Biquad \* filter

filter is a lowpass filter with variable cutoff frequency

• int fs = 0

sample rate of the jack server

• double crush\_gain = 0

gain of bit crushed output

• int nOutputs = 2

number off jack output ports

· int nChannels

number of channels in the sample

· double cutoff

cutoff represents the cutoff frequency of the lowpass fitler

• int bit

determines bit depth of bitcrushing

const char \*\* ports

the jack output ports

jack\_client\_t \* client

pointer to Jack client

jack\_status\_t status

Jack status

jack\_port\_t \*\* output\_port

pointer to Jack audio output port

• OscMan \* oscman

pointer to OSC manager object

4.4 Rain Class Reference 33

## 4.4.1 Detailed Description

Read a wave file and play it back in a granular style, with modification through OSC.

Author

Henrik von Coler modified by Marquis Fields and Malte Schneider

Date

2019/10/04

#### 4.4.2 Constructor & Destructor Documentation

### 4.4.2.1 Rain()

Constructor.

#### **Parameters**

filename	path to wav file
win_size	window size of triangular window

#### 4.4.3 Member Function Documentation

#### 4.4.3.1 callback\_process()

## **Parameters**

Χ	
object	

#### 4.4.3.2 crush()

reduce bit depth of input sample

**Parameters** 

sample current sample to be crushed

#### Returns

bitcrushed input sample

#### 4.4.3.3 process()

get the next buffer from the sample and playback

#### **Parameters**

```
nframes buffer size
```

### Returns

output of rain

#### 4.4.3.4 set\_gain()

```
void Rain::set_gain ( \label{eq:double_fc} \texttt{double}\ \textit{fc}\ \texttt{)}\quad \texttt{[private]}
```

sets gain of bitcrushed signale in dependence of filter cutoff frequency

### **Parameters**

fc filter cutoff frequency

The documentation for this class was generated from the following files:

- /Users/marquis/Desktop/\_rain/rain.h
- /Users/marquis/Desktop/\_rain/rain.cpp

# 4.5 SingleSample Class Reference

Class which holds the waveform of a sample in an array.

```
#include <singlesample.h>
```

Collaboration diagram for SingleSample:

# SingleSample + resFac - sf - info - FS - nFrames - fs\_sample - nChannels - L - x - pos - rate + SingleSample() + ~SingleSample() + read\_wavefile() + get\_sample() + get\_sample() + step() + get\_x() + set\_f() + set\_sr() + set\_nChannels() and 8 more...

#### **Public Member Functions**

```
• SingleSample (std::string filePath, int fs)
```

SingleSample the standard constructor.

virtual ∼SingleSample ()

 $\sim$  SingleSample ths should be the destructor

void read\_wavefile (std::string filePath)

read\_wavefile

• double get\_sample (int chan, int pos)

get\_sample get sample values at exact integer postions

• double get\_sample (int chan, double pos)

get\_sample return interpolated value this is an overloaded function!

• void step ()

step this function sets a new position, regarding the speed, and the resampleFactor

double \*\* get\_x ()

get\_x

- void set\_f (int in)
- · void set\_sr (int in)
- void set\_nChannels (int in)
- int get\_nChannels ()
- void set\_nFrames (int in)
- int get\_nFrames ()
- double get\_rate ()
- void **set\_rate** (double r)
- double get\_pos ()
- void set\_pos (double p)
- int getFS ()

#### **Public Attributes**

· double resFac

resampleFactor rescales the playback speed according to the sample rate of the wav file and the JACK server

#### **Private Attributes**

- · SNDFILE \* sf
- SF\_INFO info
- int FS
- · int nFrames

nFrames The number of frames in the wav file which is the length in samples

· int fs\_sample

fs\_sample sampling rate of the wav file

int nChannels

nChannels number of channels in this wav

int L

L length of the interleaved wav data (number of frames times number of channels)

double \*\* x

x The 2d-array with the sample data, arranged as a matrix.

double pos

pos the recent postion within the sample

• double rate

the playback rate

#### 4.5.1 Detailed Description

Class which holds the waveform of a sample in an array.

This class holds the waveform of a sample in an array and provides all necessary methods for reading and accessing it.

#### **Author**

Henrik von Coler

Version

Revision

0.52

Date

2016-11-22

This class holds the waveform of a sample in an array and provides all necessary methods for reading and accessing it. It serves as a state machine by holding a playback speed and position.

**Author** 

Henrik von Coler

Version

Revision

0.527

Date

2016-11-22

### 4.5.2 Constructor & Destructor Documentation

### 4.5.2.1 SingleSample()

SingleSample the standard constructor.

#### **Parameters**

filePath	
fs	the jack sample rate

### 4.5.3 Member Function Documentation

get\_sample get sample values at exact integer postions

#### **Parameters**

chan	
pos	

Returns

```
4.5.3.2 get_sample() [2/2]
```

```
double SingleSample::get_sample (
    int chan,
    double pos )
```

get\_sample return interpolated value this is an overloaded function!

#### **Parameters**

chan	the cannnel to access
pos	the floating point position

#### Returns

the sample's interpolated value

```
4.5.3.3 get_x()
double ** SingleSample::get_x ( )
get_x
```

#### Returns

returns a pointer to the sample data arrays

#### 4.5.3.4 read\_wavefile()

```
void SingleSample::read_wavefile (
            std::string filePath )
```

read\_wavefile

**Parameters** 

filePath

The documentation for this class was generated from the following files:

- /Users/marquis/Desktop/\_rain/singlesample.h
- /Users/marquis/Desktop/\_rain/singlesample.cpp

#### 4.6 **TriangularWindow Class Reference**

A single window (envelope) for granular synthesis.

```
#include <triangularwindow.h>
```

Collaboration diagram for TriangularWindow:

# TriangularWindow

- + tmp\_mode
- length
- center\_offset
- playback\_position
- looplength
- playback\_mode
- loop\_start
- loop\_end
- startpos
- + TriangularWindow()
- + initialize()
- + get\_value()
- + get\_relative\_position()
- + move\_relative\_position()
- + move\_relative\_position()
  + move\_relative\_position()
  + step\_forward()
  + step\_backward()

- + set\_playback\_mode()
- + get\_playback\_mode() + set\_length()
- + set\_start()

#### **Public Member Functions**

• TriangularWindow ()

Constructor

· void initialize (int L, int start, int looplength, int direction, int startpos)

initializes window parameters

double get\_value ()

get the value of the window at the playback\_position

• int get\_relative\_position ()

get the relativ playback position of window in wav file

• void move\_relative\_position (double p, int maxLength)

move window position in the wav file

void move\_relative\_position (double p, int maxLength, int win)

move window position in the wav file

int step\_forward ()

increments playback position

• int step\_backward ()

decrements playback position

void set\_playback\_mode (int mode)

sets playback loop mode

int get\_playback\_mode ()

get the playback loop mode

void set\_length (int I, int n)

sets playback loop length

void set\_start (int s, int n)

sets playback loop start position

#### **Public Attributes**

· int tmp\_mode

auxiliary variable to control the playbackmode

#### **Private Attributes**

· int length

the length of the window

double m

the slope of the triangular window

int center\_offset

the position of the window in the wav file

• int playback\_position

is the playback position within the window

· int looplength

specifies the Length of the playbackloop

· int playback\_mode

specifies whether the window propagates in a forward or backward loop

· int loop\_start

represets playback loop start position

· int loop\_end

represets playback loop end position

· int startpos

represets start position of window in the wav file

### 4.6.1 Detailed Description

A single window (envelope) for granular synthesis.

**Author** 

Henrik von Coler modified by Marquis Fields and Malte Schneider

Date

2019/10/04

#### 4.6.2 Member Function Documentation

```
4.6.2.1 get_playback_mode()
int TriangularWindow::get_playback_mode ( )
get the playback loop mode
Returns
     playback_mode
4.6.2.2 get_relative_position()
int TriangularWindow::get_relative_position ( )
get the relativ playback position of window in wav file
Returns
     relative positon
4.6.2.3 get_value()
double TriangularWindow::get_value ( )
get the value of the window at the playback_position
Returns
     window value
4.6.2.4 initialize()
void TriangularWindow::initialize (
              int L,
              int start,
              int looplength,
```

int direction,
int startpos )

initializes window parameters

#### **Parameters**

L	window length
start	start position of loop
looplength	length of loop
direction	direction of playback
startpos	relative position of center_offset in wav file

#### 4.6.2.5 move\_relative\_position() [1/2]

```
void TriangularWindow::move_relative_position ( \label{eq:constraint} \mbox{double } p, \\ \mbox{int } \mbox{\it maxLength } \mbox{)}
```

move window position in the wav file

#### **Parameters**

p	the fraction of the window size to shift the position
maxLength	the maximum length of the wav file

#### 4.6.2.6 move\_relative\_position() [2/2]

move window position in the wav file

#### **Parameters**

p	the fraction of the window size to shift the position	
maxLength the maximum length of the wav file		
win	window size in samples	

## 4.6.2.7 set\_length()

sets playback loop length

#### **Parameters**

1	desired playback loop length
n	length of wav file in samples

## 4.6.2.8 set\_playback\_mode()

sets playback loop mode

#### **Parameters**

```
mode the desired playback loop mode
```

#### 4.6.2.9 set\_start()

sets playback loop start position

### **Parameters**

1	desired playback loop start position
n	length of wav file in samples

### 4.6.2.10 step\_backward()

```
int TriangularWindow::step_backward ( )
```

decrements playback position

#### Returns

returns 3 if end of window is reached

## 4.6.2.11 step\_forward()

```
int TriangularWindow::step_forward ( )
```

increments playback position

#### Returns

returns 1 if end of window is reached

The documentation for this class was generated from the following files:

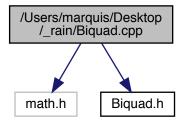
- /Users/marquis/Desktop/\_rain/triangularwindow.h
- /Users/marquis/Desktop/\_rain/triangularwindow.cpp

# **Chapter 5**

# **File Documentation**

# 5.1 /Users/marquis/Desktop/\_rain/Biquad.cpp File Reference

#include <math.h>
#include "Biquad.h"
Include dependency graph for Biquad.cpp:



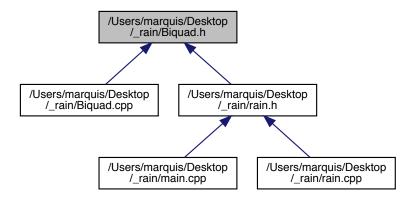
# 5.1.1 Detailed Description

Henrik von Coler modified by Marquis Fields and Malte Schneider 2019/10/04

46 File Documentation

# 5.2 /Users/marquis/Desktop/\_rain/Biquad.h File Reference

This graph shows which files directly or indirectly include this file:



#### Classes

· class Biquad

A biquad filter with variable Filter Type, Q, fc, Gain.

#### **Enumerations**

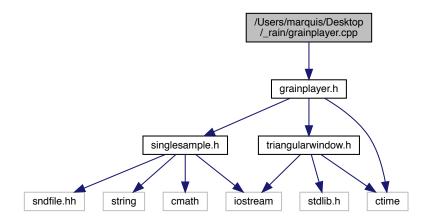
enum {
 bq\_type\_lowpass = 0, bq\_type\_highpass, bq\_type\_bandpass, bq\_type\_notch,
 bq\_type\_peak, bq\_type\_lowshelf, bq\_type\_highshelf }

### 5.2.1 Detailed Description

Henrik von Coler modified by Marquis Fields and Malte Schneider 2019/10/04

# 5.3 /Users/marquis/Desktop/\_rain/grainplayer.cpp File Reference

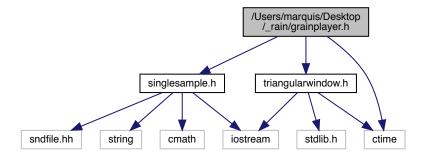
#include "grainplayer.h"
Include dependency graph for grainplayer.cpp:



# 5.4 /Users/marquis/Desktop/\_rain/grainplayer.h File Reference

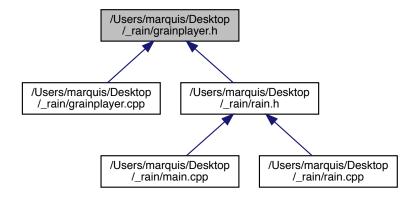
```
#include "singlesample.h"
#include "triangularwindow.h"
#include <ctime>
```

Include dependency graph for grainplayer.h:



48 File Documentation

This graph shows which files directly or indirectly include this file:



#### **Classes**

· class GrainPlayer

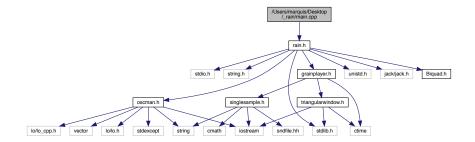
The granular player class, holding a SingleSound and an array of window functions, representing the grains.

# 5.5 /Users/marquis/Desktop/\_rain/main.cpp File Reference

Granular Synth with some extra features running on Rhaspberry PI.

#include "rain.h"

Include dependency graph for main.cpp:



#### **Functions**

• int main (int argc, char \*argv[])

### 5.5.1 Detailed Description

Granular Synth with some extra features running on Rhaspberry PI.

**Author** 

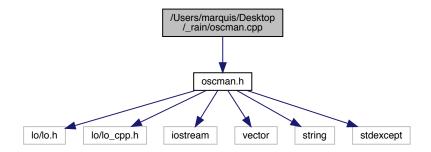
Henrik von Coler modified by Marquis Fields and Malte Schneider

Date

2019/10/04

# 5.6 /Users/marquis/Desktop/\_rain/oscman.cpp File Reference

```
#include "oscman.h"
Include dependency graph for oscman.cpp:
```



# 5.7 /Users/marquis/Desktop/\_rain/oscman.h File Reference

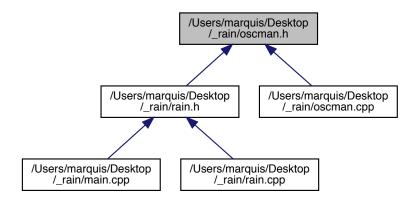
```
#include <lo/lo.h>
#include <lo/lo_cpp.h>
#include <iostream>
#include <vector>
#include <string>
#include <stdexcept>
Include dependency graph for oscman.h:
```

/Users/marquis/Desktop
/\_rain/oscman.h

lo/lo\_cpp.h iostream vector string stdexcept

50 File Documentation

This graph shows which files directly or indirectly include this file:



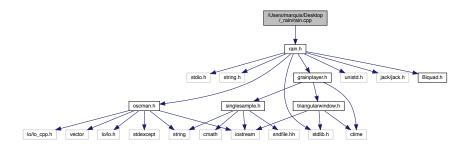
#### **Classes**

• class OscMan

Class which parses the incoming OSC messages.

# 5.8 /Users/marquis/Desktop/\_rain/rain.cpp File Reference

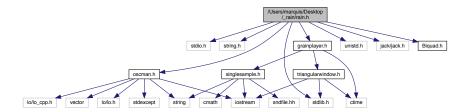
#include "rain.h"
Include dependency graph for rain.cpp:



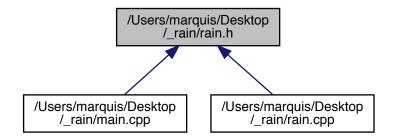
# 5.9 /Users/marquis/Desktop/\_rain/rain.h File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <jack/jack.h>
#include "oscman.h"
```

```
#include "grainplayer.h"
#include "Biquad.h"
Include dependency graph for rain.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

• class Rain

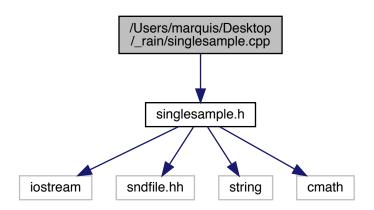
Read a wave file and play it back in a granular style, with modification through OSC.

# 5.10 /Users/marquis/Desktop/\_rain/singlesample.cpp File Reference

#include "singlesample.h"

52 File Documentation

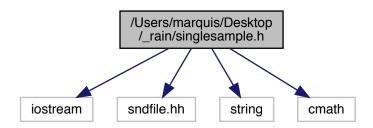
Include dependency graph for singlesample.cpp:



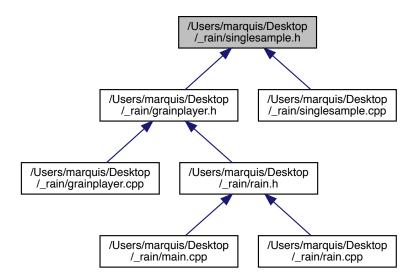
# 5.11 /Users/marquis/Desktop/\_rain/singlesample.h File Reference

```
#include <iostream>
#include <sndfile.hh>
#include <string>
#include <cmath>
```

Include dependency graph for singlesample.h:



This graph shows which files directly or indirectly include this file:



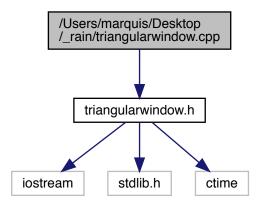
#### **Classes**

· class SingleSample

Class which holds the waveform of a sample in an array.

# 5.12 /Users/marquis/Desktop/\_rain/triangularwindow.cpp File Reference

#include "triangularwindow.h"
Include dependency graph for triangularwindow.cpp:

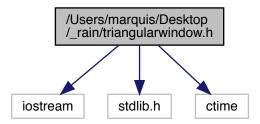


54 File Documentation

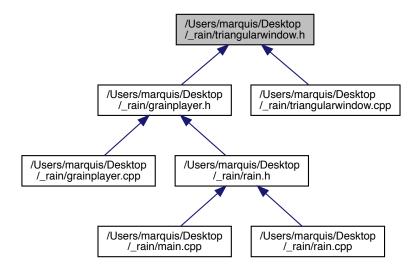
# 5.13 /Users/marquis/Desktop/\_rain/triangularwindow.h File Reference

#include <iostream>
#include <stdlib.h>
#include <ctime>

Include dependency graph for triangularwindow.h:



This graph shows which files directly or indirectly include this file:



#### Classes

• class TriangularWindow

A single window (envelope) for granular synthesis.

# Index

/Users/marquis/Desktop/_rain/Biquad.cpp, 45 /Users/marquis/Desktop/_rain/Biquad.h, 46 /Users/marquis/Desktop/_rain/grainplayer.cpp, 47 /Users/marquis/Desktop/_rain/grainplayer.h, 47 /Users/marquis/Desktop/_rain/main.cpp, 48 /Users/marquis/Desktop/_rain/oscman.cpp, 49 /Users/marquis/Desktop/_rain/oscman.h, 49 /Users/marquis/Desktop/_rain/rain.cpp, 50 /Users/marquis/Desktop/_rain/rain.h, 50 /Users/marquis/Desktop/_rain/singlesample.cpp, 51 /Users/marquis/Desktop/_rain/singlesample.h, 52 /Users/marquis/Desktop/_rain/triangularwindow.cpp, 53 /Users/marquis/Desktop/_rain/triangularwindow.h, 54	get_spread OscMan, 22 get_start OscMan, 22 get_texture OscMan, 22 get_value TriangularWindow, 41 get_x SingleSample, 38 GrainPlayer, 9 get_sample, 12 GrainPlayer, 12
Biquad, 7 process, 9 bit_callback OscMan, 19  callback_process Rain, 33 crush	set_funk, 12 set_length, 13 set_mode, 13 set_pitch, 13 set_relative_distance, 1 set_reset, 14 set_spread, 14 set_start, 14
Rain, 33 cutoff_callback OscMan, 19	initialize TriangularWindow, 41
funky_callback OscMan, 20	length_callback OscMan, 23
get_bit OscMan, 20 get_cutoff	mode_callback OscMan, 23 move_relative_position TriangularWindow, 42
OscMan, 20 get_funky OscMan, 21	OscMan, 15 bit_callback, 19
get_length OscMan, 21 get_mode	cutoff_callback, 19 funky_callback, 20 get_bit, 20
OscMan, 21 get_pitch OscMan, 21	get_cutoff, 20 get_funky, 21 get_length, 21
get_playback_mode     TriangularWindow, 41 get_relative_position	get_mode, 21 get_pitch, 21 get_reset, 22
TriangularWindow, 41 get_reset OscMan, 22	get_spread, 22 get_start, 22 get_texture, 22
get_sample GrainPlayer, 12 SingleSample, 38	length_callback, 23 mode_callback, 23 OscMan, 18

56 INDEX

pitch_callback, 24	step_backward
q_callback, 24	TriangularWindow, 43
reset_callback, 26	step_forward
spread_callback, 26	TriangularWindow, 43
start_callback, 28	
texture_callback, 28	texture_callback
win_callback, 30	OscMan, 28
	TriangularWindow, 39
pitch_callback	get_playback_mode, 41
OscMan, 24	get_relative_position, 41
process	get_value, 41
Biquad, 9	initialize, 41
Rain, 34	move_relative_position, 42
	set_length, 42
q_callback	set_playback_mode, 43
OscMan, 24	set_start, 43
Dain 00	step_backward, 43
Rain, 30	step_forward, 43
callback_process, 33	win callback
crush, 33	Win_caliback OscMan, 30
process, 34	Oscivian, 30
Rain, 33	
set_gain, 34	
read_wavefile	
SingleSample, 38	
reset_callback	
OscMan, 26	
set_funk	
GrainPlayer, 12	
set_gain	
Rain, 34	
set_length	
GrainPlayer, 13	
TriangularWindow, 42	
set_mode	
GrainPlayer, 13	
set_pitch	
GrainPlayer, 13	
set_playback_mode	
TriangularWindow, 43	
set_relative_distance	
GrainPlayer, 14	
set reset	
GrainPlayer, 14	
set_spread	
GrainPlayer, 14	
set start	
GrainPlayer, 14	
TriangularWindow, 43	
SingleSample, 35	
get_sample, 38	
get_x, 38	
read_wavefile, 38	
SingleSample, 37	
spread_callback	
OscMan, 26	
start_callback	
OscMan, 28	
300man, <u>20</u>	