CWave

a basic Synthesizer made using C.

Made by Pranav Prabhakar

Contributors: Prema P Kotur, Pranav Chandrasekar

What is CWave?

What is CWave?

CWave is a simple synthesizer, made in C. It lets you play and record sounds using your computer keyboard or a MIDI device.

Different waveforms

Different waveforms

Volume and octave control

Different waveforms

Volume and octave control

Using a keyboard to control

Different waveforms

Volume and octave control

Using a keyboard to control

Recording sessions

includes and definitions

includes and definitions

```
#include <stdio.h>
#include <stdlib.h>
 #include <math.h>
#include <Windows.h>
#include <mmsystem.h>
#include "portaudio.h"
#pragma comment(lib, "winmm.lib")
```

data structures

data structures

```
typedef enum { WAVE_SINE, WAVE_SQUARE, WAVE_SAW } Waveform;
```

```
typedef struct {
    float frequency;
    float phase;
    int active;
} Note;
```

data structures

Synth structure for defining waveforms (kinda important)

```
typedef struct {
    Note notes[MAX_NOTES];
    float volume;
    Waveform waveform;
    int octaveShift;
    int recording;
    float *recordBuffer;
    size_t recordPos;
    size_t recordMaxSamples;
} SynthData;
```

note handling

note handling

"equal temperament tuning"

note handling

`midiNoteToFreq` turns a **MIDI note number** into a real sound **frequency.**

`note_on(freq)` – starts playing a note at that frequency, sets `active = 1`

`note_off(freq)` – stops playing it, sets `active = 0`

```
float midiNoteToFreq(int midiNote);
```

```
void note_on(float freq);
void note_off(float freq);
```

MIDI callback

MIDI callback

- Checks the message to see if it's a **Note On** or **Note Off**
- Then it calls:
    - `note_on()` – if the note should start playing
    - `note_off()` – if the note should stop playing.

```
void CALLBACK midi_callback(...)
```

waveform generator

waveform generator

```
float generateWaveSample(float frequency, float time, Waveform waveform);
```

frequency

time

waveform

sine

square

sawtooth

Recording to WAV

```
void writeWavFile(const char *filename, float *data, size_t numSamples);
```

saves files in this format:

- `recording1.wav`
- `recording2.wav`
- `Recording3.wav`

and so on.

PortAudio Callback

```
void writeWavFile(const char *filename, float *data, size_t numSamples);
```

keyboard interaction

```
void updateNotesFromKeyboard(SynthData *data)
```

- *Monitors key states using `GetAsyncKeyState()`.*
- *Plays notes based on keys A–K and W–U.*
- *Adjusts volume (+/-).*
- *Changes waveform (Z, X, C).*
- *Shifts octaves (↑, ↓).*
- *Toggles recording (R).*

GetFrequencyForKey

how your computer keyboard becomes a piano

GetFrequencyForKey

how your computer keyboard becomes a piano

```
float getFrequencyForKey(int vkey, int octaveShift)
```

main function - int main()

Main Function - int main()

What It Does:

1. **Starts things up:**
    - Initializes **PortAudio** (for sound output)
    - Sets up **MIDI input** (to listen to your keyboard/MIDI controller)
2. **Begins audio playback:**
    - Starts the **audio stream**
    - Enters a loop to keep the synth running
3. **Main loop:**
    - **Watches for keyboard input** (like computer keys)
    - If you press **ESC**, it stops the loop and shuts everything down
4. **When you're done:**
    - Closes MIDI + audio connections
    - Shuts down PortAudio
    - Saves any **recording** you were making to a WAV file

thank you