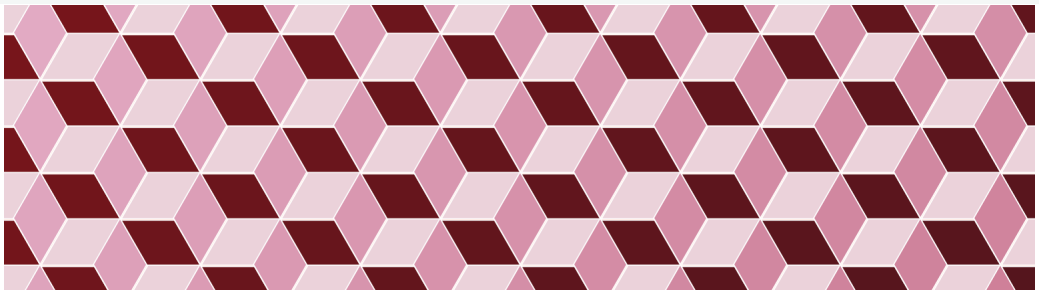


15-104 Introduction to Computing for Creative Practice

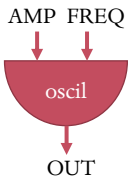
Fall 2021

24 More Sound

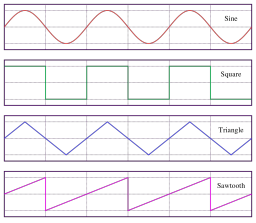
Instructor: Tom Cortina, tcortina@cs.cmu.edu, GHC 4117, 412-268-3514

1

p5.Oscillator



- Creates a signal that oscillates between -1.0 and 1.0.
  - By default, the oscillation takes the form of a sinusoidal shape ('sine').
  - The frequency defaults to 440 oscillations per second (440Hz).
- Some methods:
  - `start()` Start an oscillator.
  - `stop()` Stop an oscillator.
  - `amp()` Set the amplitude between 0 and 1.0.
  - `freq()` Set frequency of an oscillator to a value.
  - `setType()` Set type to 'sine', 'square', 'triangle', or 'sawtooth' .
  - `disconnect()` Do not send output of this oscillator to the speakers.



2

Example (review)

```
function draw() {
   background(200);
   fill(0);
   ellipse(mouseX, mouseY, 20, 20);
   myTone.amp(constrain(mouseX / width, 0, 1));
   myTone.freq(constrain(200 + 1000*(mouseY / height), 200, 1200));
   if (mouseX > 2*width) {
       myTone.stop();
       noLoop();
   }
}
```

Volume ↔

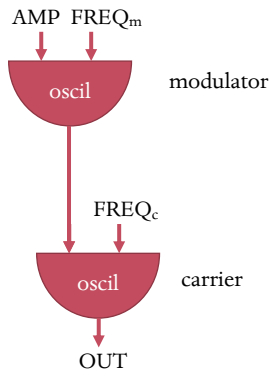
Pitch ↕

A fraction between 0 and 1 when the mouse is in the canvas bounds.

3

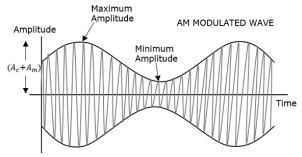
Amplitude Modulation

- When the amplitude of an oscillator is controlled by another oscillator, you have amplitude modulation.
- The sound producing oscillator is called the carrier and the oscillator controlling the amplitude is called the modulator.
- If the frequency of modulation is small, we get the musical effect we know as tremolo.
- If the frequency of modulation is large, we get a spectrally-rich sound instead.



4

Tremolo




```
var myTone;
var tremolo;
function setup() {    SAME AS BEFORE    }

function soundSetup() { // setup for audio generation
    myTone = new p5.Oscillator();
    myTone.setType('sine');
    myTone.start();
    tremolo = new p5.Oscillator();
    tremolo.setType('sine');
    tremolo.disconnect();     // don't send this oscil to speakers
    tremolo.start();
}
```

5

Tremolo (cont'd)

```
function draw() {
    background(200);
    fill(0);
    ellipse(mouseX, mouseY, 20, 20);
    tremolo.amp(0.75);
    tremolo.freq(constrain(2 + 6 * (mouseX / width), 2, 8));
    myTone.amp(tremolo);
    myTone.freq(constrain(200 + 1000 *(mouseY / height), 200, 1200));
    if (mouseX > 2*width) {
       myTone.stop(); noLoop();
    }
}
```

The amplitude will increase to 0.75 from 2 to 8 times per second depending on the horizontal position of the mouse.

The `amp()` function can also have an oscillator as its argument.

6

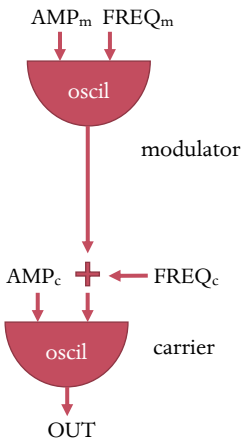
Amplitude Modulation

```
function draw() {
    background(200);
    fill(0);
    ellipse(mouseX, mouseY, 20, 20);
    tremolo.amp(0.75);
    tremolo.freq(constrain(100 + 100 * (mouseX / width), 100, 200));
    myTone.amp(tremolo);
    myTone.freq(constrain(200 + 1000 *(mouseY / height), 200, 1200));
    if (mouseX > 2*width) {
       myTone.stop(); noLoop();
    }
}
```

7

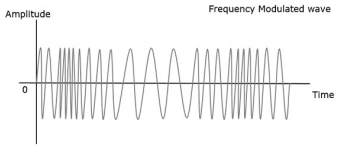
Frequency Modulation

- When the frequency of an oscillator is controlled by another oscillator, you have **frequency modulation**.
- The sound producing oscillator is called the carrier and the oscillator controlling the amplitude is called the modulator.
- If the frequency of modulation is small, we get the musical effect we know as vibrato.
- If the frequency of modulation is large, we get a spectrally-rich sound instead.



8

Vibrato




```
var myTone;
var vibrato;
function setup() {    SAME AS BEFORE    }

function soundSetup() { // setup for audio generation
    myTone = new p5.Oscillator();          // default freq: 440Hz
    myTone.setType('sine');
    myTone.start();
    vibrato = new p5.Oscillator();
    vibrato.setType('sine');
    vibrato.disconnect();      // don't send this oscil to speakers
    vibrato.start();
}
```

9

Vibrato (cont'd)

The pitch will vary above and below 440 Hz by 0 to 100 Hz between 1-4 times a second.

```
function draw() {
    background(200);
    fill(0);
    ellipse(mouseX, mouseY, 20, 20);
    vibrato.amp(constrain(100 * (mouseY / height), 0, 100));
    vibrato.freq(constrain(1 + 3 * (mouseX / width), 1, 4));
    myTone.amp(0.5);
    myTone.freq(vibrato);
    if (mouseX > 2*width) {
        myTone.stop(); noLoop();
    }
}
```

The `freq()` function can also have an oscillator as its argument.

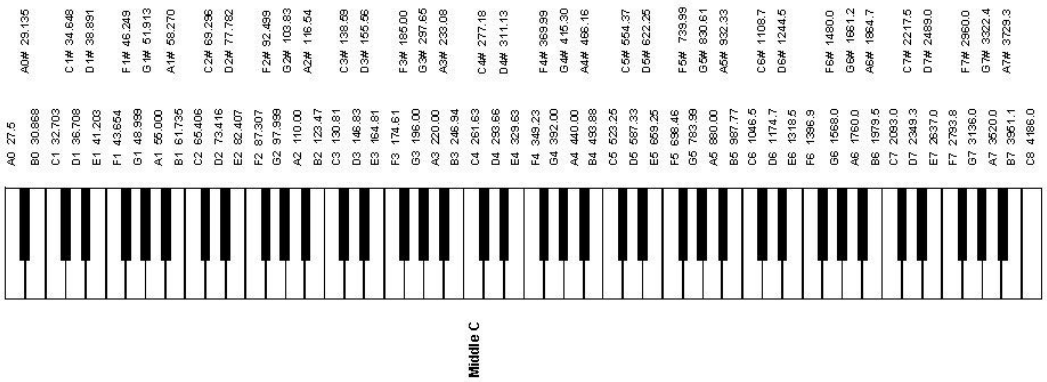
10

Frequency Modulation

```
function draw() {
    background(200);
    fill(0);
    ellipse(mouseX, mouseY, 20, 20);
    vibrato.amp(constrain(440 * (mouseY / height), 0, 440));
    vibrato.freq(constrain(440 * (mouseX / width), 0, 440));
    myTone.amp(0.5);
    myTone.freq(vibrato);
    if (mouseX > 2*width) {
        myTone.stop(); noLoop();
    }
}
```

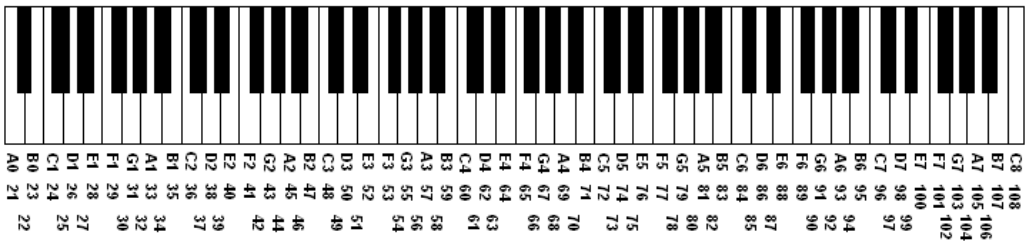
11

Common Pitches



12

MIDI (Musical Instrument Digital Interface)



13

Example: Play major scale

```
var note = 60;    // starting note in MIDI notation
var steps = [2, 2, 1, 2, 2, 2, 1];  // 2=whole step, 1=half step
var stepIndex = 0;
var osc;
function setup() {
    createCanvas(200, 100);
    frameRate(4); // draw repeats 4 times per second
    useSound();
}
function soundSetup() {
    osc = new p5.Oscillator();
    osc.amp(0.25); osc.freq(midiToFreq(note)); osc.start(); }
```

`midiToFreq` returns the frequency that corresponds to the MIDI value given as its argument.

14

Example: Play major scale (cont'd)

```
function draw() {
    print(frameCount);
    background(200);
    if (frameCount % 4 == 0) {
        if (stepIndex == steps.length) {
            osc.stop(); noLoop();
        }
        note += steps[stepIndex];
        osc.freq(midiToFreq(note));
        stepIndex++;
    }
}
```

Every 4th frame (`frameCount` is a multiple of 4) we advance to the next note of the scale by adding the next value in the `steps` array to `note`, stopping once we reach the end of the array.

When you listen to this program's sound, why is the first note shorter in duration than the others? How would you fix this?

15

New instruction: `switch` statement

```
if (x == 0) {
    freq += 100;
} else if (x == 1) {
    freq -= 100;
} else if (x == 2) {
    freq *= 2;
} else if (x == 3) {
    freq /= 2;
} else {
    freq = 440;
}
```

```
switch (x) {
    case 0: freq += 100; break;
    case 1: freq -= 100; break;
    case 2: freq *= 2; break;
    case 3: freq /= 2; break;
    default: freq = 440;
}
```

The break statements are important here! They break out of the switch statement. Without them, case 0 would execute all 5 cases! (case only indicates where to start)

16