Envelopes in ChucK 101
COS/MUS 314, Spring 2012

The Envelope object creates a linear "ramp" that can be used to smoothly change gain, frequency, or many other things.

Envelope has several important properties that you can modify (and read):

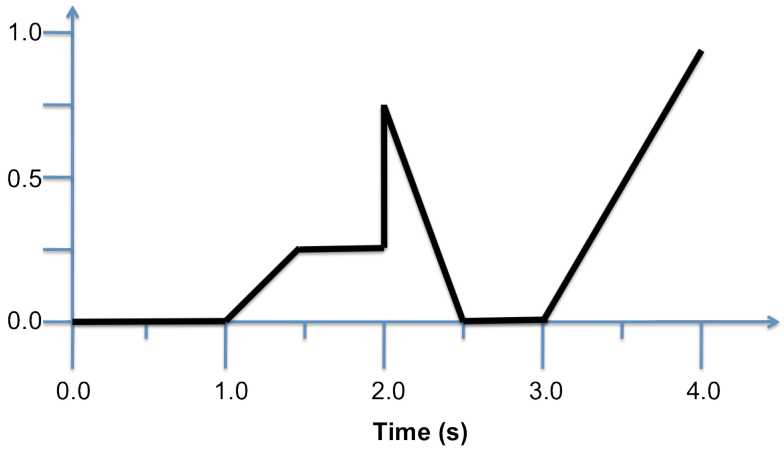
- **target:** What will the envelope "ramp to" ? (The target is of type float)
- **duration**: How long will it take to ramp to the target? (The duration is of type dur)
- **value:** What is the value of the envelope *right now*? (The value is of type float)

There are two basic ways to *control* an envelope.
Control Option 1. Set its **target** to a new value. Once **duration** has passed, the envelope value will stay at target until you set its target to something else. (Note: When you create a new Envelope, its initial **value** will be 0, so it will ramp from 0 to your target whenever you set a new target. If you want it to ramp from something else to your target, set the **value** to your preferred initial value.

For example, check out this code:

```
SinOsc s => Envelope e => dac;
.5::second => e.duration;

1::second => now;

.25 => e.target;
1::second => now;

.75 => e.value;
0 => e.target;
1::second => now;

1::second => e.duration;
1.0 => e.target;
1::second => now;
```

The code above will create an amplitude envelope that looks like this:

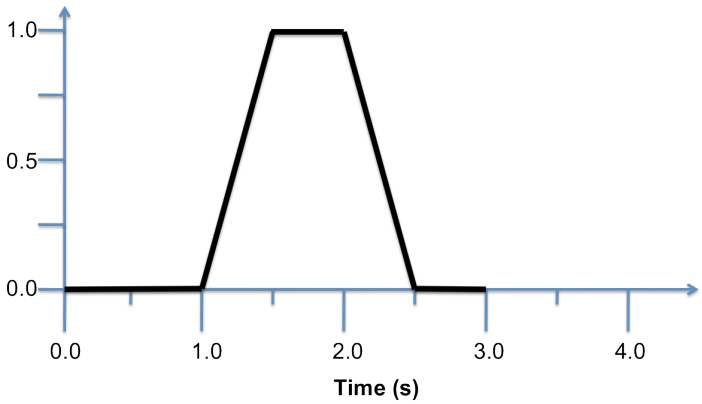


Control Option 2. Use the envelope's keyOn() and keyOff() functions. keyOn() is equivalent to setting the target to 1 and keyOff() is equivalent to setting the target to 0. These are used mostly when the Envelope is used for amplitude control (e.g., to impart a nice fade or non-clicky start and stop to notes).

Example:

```
Noise n => Envelope e => dac;
.5::second => e.duration;

1::second => now;
e.keyOn();
1::second => now;
e.keyOff();
1::second => now;
```

The code above will produce an amplitude envelope that looks like this:



There are two basic ways to *use* an envelope:

Option 1. Stick the envelope in your sound synthesis patch, and use it to control amplitude of some other unit generator (or combination thereof). Both examples above use an Envelope in this way.

Another slightly more complex example might use an Envelope as a master fader, controlling the overall volume of a patch containing several unit generators:

```
SinOsc s => Envelope masterFader => dac;
Noise n => masterFader;
Mandolin m => masterFader;

masterFader.keyOn();
...
```

Option 2. Use the envelope to control something other than amplitude (e.g., frequency). In this case, do NOT connect it directly to your sound-generating UGens using =>. Instead, connect it to a **blackhole** object, which will "suck" samples from the envelope and ensure that the envelope ramping actually happens.

Example:

```
SinOsc s => dac;
Envelope e => blackhole;

1::second => e.duration;

while (true) { //repeat swoop forever
    440 => e.value; //swoop from 440Hz
    880 => e.target; //  to 880Hz
    now + 1::second => time later; //swoop for 1 second

    //"manually" use changing envelope value to set freq
    while (now < later) {
        e.value() => s.freq;
        1::samp => now;
    }
}
```