# ADSR-Envelope

Klassen: ADSREnvelope.h und ADSREnvelope.cpp

Funktionen: nextSample(), calculateMultiplier(), enterStage()

Der verwendete Algorithmus für den Multiplier ist von **Christian Schonebeck**(http://www.musicdsp.org/showone.php?id=189).

Zuerst muss der ADSR-Envelope wissen in welcher Phase er sich gerade befindet. Dafür gibt es die Variable "**currentStage"**.In der Funktion "**nextSample()**" wird geprüft ob der Generator sich nicht in der "Off" oder "Sustain" Phase befindet, da nur in der "Attack", "Decay" und "Release" Phase Veränderungen an der Kurve stattfinden. Nach einer vom Benutzer festgelegten Zeit verlässt der Generator eine Phase und geht in die nächste über, dazu werden "currentSampleIndex" und "nextStageSampleIndex" miteinander verglichen. Sind die beiden gleich, dann wird "newStage" die nächste Phase zugewiesen, anschließend wird diese der Funktion "**enterStage()**" übergeben.

*double ADSREnvelope::nextSample()*

*{*

*if (currentStage != ENVELOPE\_STAGE\_OFF &&*

*currentStage != ENVELOPE\_STAGE\_SUSTAIN)*

*{*

*if (currentSampleIndex == nextStageSampleIndex)*

*{*

*EnvelopeStage newStage = static\_cast<EnvelopeStage>((currentStage + 1) % kNumEnvelopeStages);*

*enterStage(newStage);*

*}*

*currentLevel \*= multiplier;*

*currentSampleIndex++;*

*}*

*return currentLevel;*

*}*

Die Funktion **"enterStage()"** greift auf die Funktion **"calculateMultiplier()"** zu, um in der "Attack", "Decay" und "Release" Phase zu berechnen, wann diese auslaufen. Der Variable "currentStage" wird die aktuelle Phase zugeweiesen und der aktuelle Index wieder auf 0 gesetzt, dann folgt eine Abfrage. Hier ist der else-Teil interessant, denn dort bekommt der Index für das nächste Sample einen Wert. Im Switch wird dann schließlich der Multiplier für die jeweilige Phase berechnet, welcher in **"nextSample()"** zu "currentLevel" hinzuaddiert wird. Der aktuelle Index wird ebenfalls hochgezählt um den Zeitablauf im Auge zu behalten. So wird die Funktion "**nextSample()**" für jedes neue Sample aufgerufen.

*void ADSREnvelope::enterStage(EnvelopeStage newStage) {*

*if (currentStage == newStage) return;*

*currentStage = newStage;*

*currentSampleIndex = 0;*

*if (currentStage == ENVELOPE\_STAGE\_OFF ||*

*currentStage == ENVELOPE\_STAGE\_SUSTAIN) {*

*nextStageSampleIndex = 0;*

*}*

*else {*

*nextStageSampleIndex = stageValue[currentStage] \* sampleRate;*

*}*

*switch (newStage) {*

*case ENVELOPE\_STAGE\_OFF:*

*currentLevel = 0.0;*

*multiplier = 1.0;*

*break;*

*[...]*