

Windows/Mac/Linux対応でI/Oもサクッ！

オープンソースのブロック型言語 Pure Dataではじめる サウンド信号処理

青木 直史，藍 圭介

最終回 第12回

C言語オリジナル・ブロックを使ったリアルタイム処理

● 今回の目標…リアルタイム処理用チルダ・ブロックを作る

前回（2016年10月号）に続いてPure Dataにおけるオリジナル・ブロックの作り方について解説します．

前回は，コントロール・レートで動作するブロックを作成しましたが，今回は音データをリアルタイムに処理することを目標として，オーディオ・レートで動作するオリジナル・ブロックを作成します．

Pure Dataのブロックは，

- コントロール・レートで動作するもの
- オーディオ・レートで動作するもの

の二つに分類できます．オーディオ・レートで動作するブロックは，名前に「~（チルダ）」が付くため，チルダ・ブロックとも呼ばれます

チルダ・ブロックの作り方

● コンパイル手順

コントロール・レートで動作するブロックと同様，チルダ・ブロックについてもオリジナル・ブロックを作成できます．まずは簡単な例として，**リスト1**のC言語のプログラムをコンパイルし，[through~]ブロックを作成してみましょう．コンパイルの手順は前回と同じです．このプログラムをコンパイルするには，Windowsの場合は**リスト2**(**a**)のmakefile，Macの場合は**リスト2**(**b**)のmakefileを使います．

● 動作確認…音データのスルー再生

コンパイルが完了したら，[through~]ブロックを使って，**図1**のPure Dataのプログラムを作ってみましょう．[through~]ブロックは，インレットから入力された音データをそのままアウトレットに出力するオリジナル・ブロックです．このプログラムを実行すると，ファイルから読み込んだ音データがそのままスルー再生されることになります．

● 作成ルール1…イベントに対応した処理をC言語の関数として記述する

リスト1に示すように，チルダ・ブロックについてもオリジナル・ブロックを作成するには，Pure Data

リスト1　音データをリアルタイムに処理するオリジナル・ブロックの基本形 [through~] ブロックのプログラム（through~.c）

```
#include "m_pd.h"

static t_class *through_class;

typedef struct _through
{
  t_object x_obj;
  float x_f;
} t_through;

static t_int *through_perform(t_int *w)
{
  t_float *s0 = (t_float*)(w[1]);
  t_float *s1 = (t_float*)(w[2]);
  int N = (int)(w[3]);

  int n;

  for (n = 0; n < N; n++)
  {
    s1[n] = s0[n];
  }

  return (w + 4);
}

void through_dsp(t_through *x, t_signal **sp)
{
  dsp_add(through_perform, 3, sp[0]->s_vec,
                              sp[1]->s_vec, sp[0]->s_n);
}

void *through_new(void)
{
  t_through *x = (t_through *)pd_new(through_class);
  outlet_new(&x->x_obj, gensym("signal"));
  return (void *)x;
}

void through_tilde_setup(void)
{
  through_class = class_new(gensym("through~"),
    (t_newmethod)through_new, 0, sizeof(t_through),
                                 CLASS_DEFAULT, 0);
  CLASS_MAINSIGNALIN(through_class, t_through, x_f);
  class_addmethod(through_class,
            (t_method)through_dsp, gensym("dsp"), 0);
}
```

[through~] ブロックの動作を記述
インレットから入力された音データ
アウトレットに出力する音データ
フレームのサイズ
インレットから入力された音データをそのままアウトレットに出力する音データとしてコピーする
Pdウィンドウの DSP をチェックしたときの処理
パッチ・ウィンドウにブロックを生成したときの処理
Pure Data を起動したときの処理

第1回　正弦波／ノコギリ波／矩形波…まずは基本音を鳴らす（2015年12月号）
第2回　サウンド処理の基本満載！レガシ・ピコピコ音BGM（2016年1月号）
第3回　リアルタイム音声処理の準備…データ・ファイルの保存＆再生（2016年2月号）