

CSV ファイルを読み込む

CSV ファイルを読み込む

Processing では XML、JSON や CSV などの形式のテキストファイルを読み込み、データとして扱うことができます。ここでは例として最も単純な CSV ファイル（comma separated value = カンマ区切り値）の読み込みを説明します。

```
Tori[] tori;          //Tori クラスのインスタンス配列を宣言
(中略)
Table csv = loadTable("data/birds.csv", "header");    //CSV ファイルを読み込む。
int num = csv.getRowCount(); // データの行数をカウント
tori = new Tori[num];      // 行数の分だけインスタンスを作る
for (int i = 0; i < num; i++) {    // 行数の分だけループ
    TableRow row = csv.getRow(i);    //csv から 1行分のデータを取り出し、row に代入する
    tori[i] = new Tori();
    tori[i].name = row.getString("name");    // "name" 列のデータを文字列として取り出す
    tori[i].seasonBegin = row.getFloat("season"); // "season" 列のデータを小数として取り出す
    tori[i].num = row.getInt("num");    // "num" 列のデータを整数として取り出す
}
```

CSV のファイルの作り方

- ・ Microsoft Excel で作ったリストを作ります（.xlsx 形式）。その際、1 行目を「ヘッダー」として扱い、以下の行のデータを取り出すための名前をつけます。例えば、「name」や「x」「y」「fileName」など。
- ・ .xlsx 形式のデータはマスター（原本）とするので大切にしておきます。
- ・ メニューから「名前をつけて保存」を選び、「フォーマット」のプルダウンメニューから「CSV（コンマ区切り）(.csv)」を選びます。
- ・ 書き出された CSV ファイルは「Shift_JIS」という文字コードになっています。一方で Processing が読むことのできる文字コードは「UTF-8」なので、このままでは読めません。そのため、文字コードを変換する必要があります。文字コードを変換するために「mi」というアプリケーションを使います（Mac の場合）。
- ・ 「mi」のウェブサイト（<https://www.mimikaki.net/>）で最新バージョンをダウンロードします。
- ・ 「mi」のインストールが終わったら、先ほどの CSV ファイルを「mi」で開きます。
- ・ 「mi」のウィンドウの上側に文字コードを選ぶプルダウンメニューがあり、現状では「Shift_JIS（Windows 拡張）」になっています。これをクリックし、メニューから「UTF-8」を選びます。これを保存します。
- ・ Processing のスケッチフォルダの中に「data」フォルダを作り、この中に先ほど保存し直した CSV ファイルを入れます。
- ・ これで Processing のスケッチから CSV ファイルを読み込む準備ができました。

Excel の書き方の例：

| name | height | age | fileName |
|------|--------|-----|------------|
| すずき | 184.5 | 53 | suzuki.jpg |
| たなか | 175.7 | 76 | tanaka.jpg |
| やまだ | 193.2 | 23 | yamada.jpg |

同じ Excel から書き出された CSV データの中身：

```
name,height,age,fileName
すずき,184.5,53,suzuki.jpg
たなか,175.7,76,tanaka.jpg
やまだ,193.2,23,yamada.jpg
```

Table クラスの主な取得系メソッド

```
Table table = loadTable("data/hoge.csv", "header"); //CSVを読み込み、先頭行をヘッダとして解釈する
Table table2 = loadTable("data/hoge.csv"); //CSVを読み込み、先頭行もデータとして解釈する
int rowNum = table.getRowCount(); //CSVデータの行数を取得
int columnNum = table.getColumnCount(); //CSVデータの列の数を取得
String st0 = table.getString(3, 0); //3行目 0列目の値を取得し、文字列として解釈
String st1 = table.getString(3, "name"); //3行目「name」列の値を取得し、文字列として解釈
int a0 = table.getInt(3, 2); //3行目 2列目の値を取得し、整数値として解釈
int a1 = table.getInt(3, "age"); //3行目「age」列の値を取得し、整数値として解釈
float b0 = table.getFloat(3, 1); //3行目 1列目の値を取得し、小数値として解釈
float b1 = table.getFloat(3, "height"); //3行目「height」列の値を取得し、小数値として解釈
```

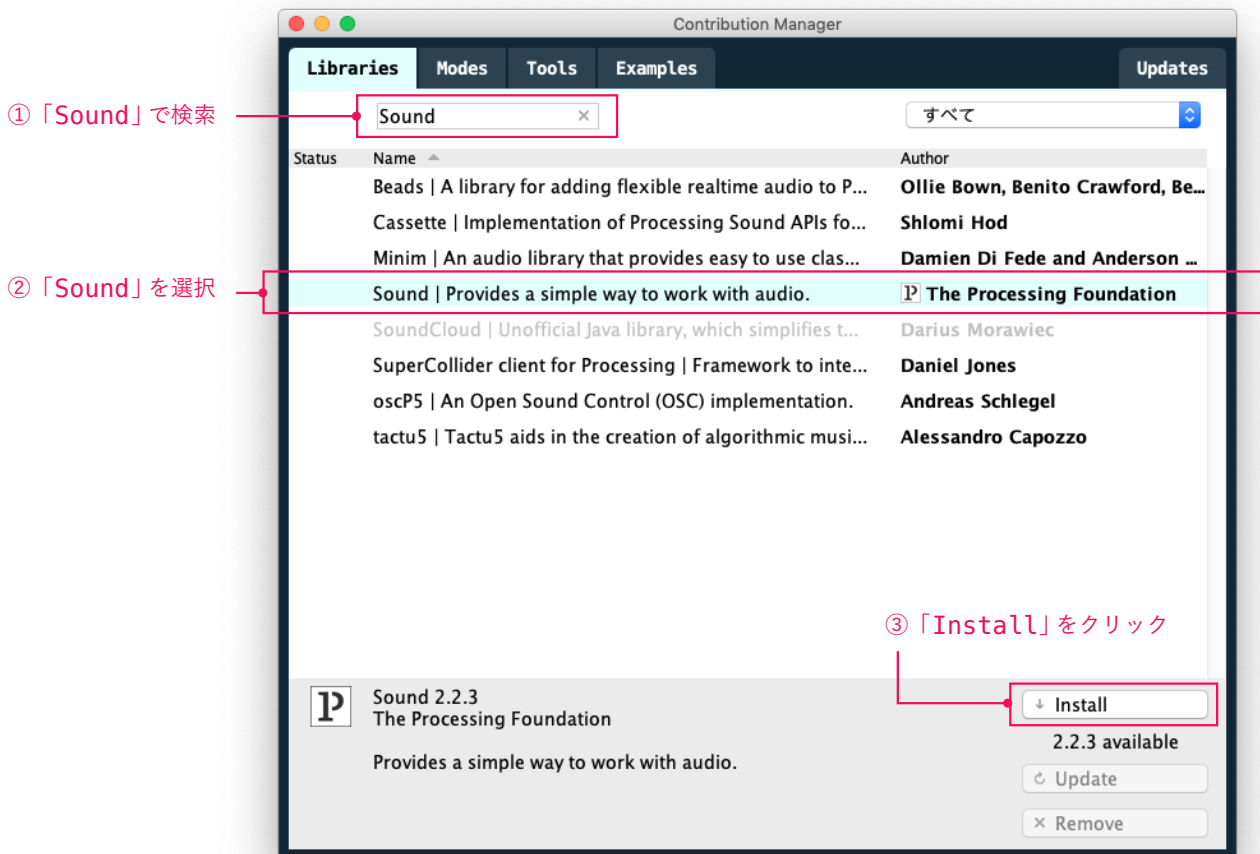
TableRow クラスの主な取得系メソッド

```
TableRow row = table.getRow(10); //CSVの10行目のデータを1行ぶん取得
String st0 = row.getString("name"); //rowから「name」列のデータを取得し、文字列として解釈
String st1 = row.getString(0); //rowから 0列目のデータを取得し、文字列として解釈
int a0 = row.getInt("age"); //rowから「age」列のデータを取得し、整数値として解釈
int a1 = row.getInt(2); //rowから 2列目のデータを取得し、整数値として解釈
float b0 = row.getFloat("height"); //rowから「height」列のデータを取得し、小数値として解釈
float b1 = row.getFloat(1); //rowから 1列目のデータを取得し、小数値として解釈
int columnNum = row.getColumnCount(); //rowの列数を取得
String columnName = row洗.getColumnName(2); //rowの2列目の列の名前を取得
```

音声ファイルを扱う（3.x 系列）

Sound ライブラリを使う

Processing では音の入出力ができます。サイン波やホワイトノイズのような電子音もできますが、ここでは WAV などの音声ファイルを扱います。Processing には標準で音を扱う機能がついていないので、「Sound」というライブラリをインストールします。Processing のメニューの「スケッチ」から「ライブラリをインポート」>「ライブラリを追加」を選択します。すると「Contribution Manager」という名前のウィンドウが開き、追加可能なライブラリの一覧が現れるので、その中から「Sound」を選び、右下の「Install」ボタンをクリックします。



音声ファイルを扱ってみる

実際に音声ファイルを扱うには、メインスケッチの先頭に音声ライブラリを使う旨を記す必要があります。

```
import processing.sound.*;    // 音声ライブラリをインポートする
```

つぎに、音声ファイルを準備します。公称では WAV、AIF/AIFF、MP3 が使えることになっていますが、MP3 はデコードが遅く、しかもエラーが起きやすいので WAV か AIF/AIFF が推奨されます。音声ファイルを用意したらスケッチのウィンドウにドラッグ&ドロップするか、スケッチフォルダの「data」フォルダの中に入れます。

音声ファイルをスケッチフォルダに入れたら、実際のコードを書いていきます。

例：画面の中心の x,y 座標とマウスの座標との関係で音量とパンを行うサンプル

```
import processing.sound.*; //Sound ライブラリのインポート
SoundFile file;           // 音声ファイルを入れる箱

void setup() {
    size(1024, 768);
    file = new SoundFile(this, "data/mahiwa.wav"); // 音声ファイルを読み込み
    file.loop();           // ループ再生
    // file.play();        // 一度だけ再生
    file.amp(0.001);       // 音量。0～1.0
}

void draw() {
    background(0);
    float x = width/2;      // パンと音量を決めるための基準点の座標
    float y = height/2;
    float angle = atan2(y - mouseY, x - mouseX); // マウスと x, y との角度を測る
    float distance = dist(x, y, mouseX, mouseY); // マウスと x, y の距離を測る
    //x, y とマウスの距離が、音がなり始める半径より小さくなったら音量とパン度合いを計算する
    float r = 300;          // 音がなり始める半径
    if (distance < r) {
        float volume = (r - distance) / r; // ボリュームを、2点が近くなるに連れて大きくなるよう計算。
        float pan = cos(angle); // パン度合い
        file.amp(volume);       // 音量を設定
        file.pan(pan);          // パンを設定 (-1.0～1.0)
        noStroke();
        fill(255);
        text("amp: " + volume + ", pan: " + pan, mouseX, mouseY);
    } else {
        file.amp(0.001);       // 音量を下げる
    }
    noStroke();
    fill(0, 255, 255, 50);
    ellipse(x, y, r*2, r*2);
    stroke(255);
    noFill();
    line(x, y, mouseX, mouseY);
}
```

SoundFile クラスの主なメソッド

```
SoundFile file = new SoundFile(this, "data/sound.wav"); // 音声ファイルを読み込む
file.play();           // 1度だけ再生
file.loop();           // ループ再生
file.cue(3.5);         // 再生ヘッドを特定の秒数に飛ばす（再生はされない）
file.jump(3.5);        // 再生ヘッドを特定の秒数に飛ばす（再生されていなければ再生開始）
file.amp(0.3);         // 音量を設定する。0 ～ 1.0
file.pan(-0.5);        // 音を左右に振る（パンする）。-1.0（左）～ 1.0（右）。ステレオ音源には無効
file.pause();          // 一時停止
file.stop();           // 停止
float sec = file.duration(); // 音声ファイルの秒数を取得
boolean isPlay = file.isPlaying(); // 再生中かどうかを取得する
int channelNum = file.channels(); // 音源のチャンネル数を取得。1 がモノラル、2 がステレオ
```