PHPでバイナリ変換プログラミング

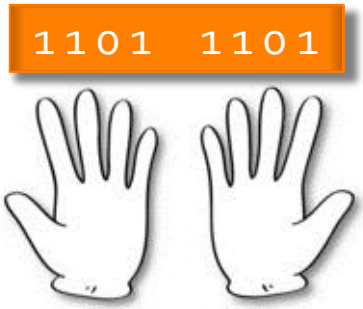
～ 前提知識から openpear/IO_Bit の紹介、応用事例まで ～

“よや” <yoya@awm.jp>

自己紹介

- 六本木の方で携帯**Web**の仕事をしてます

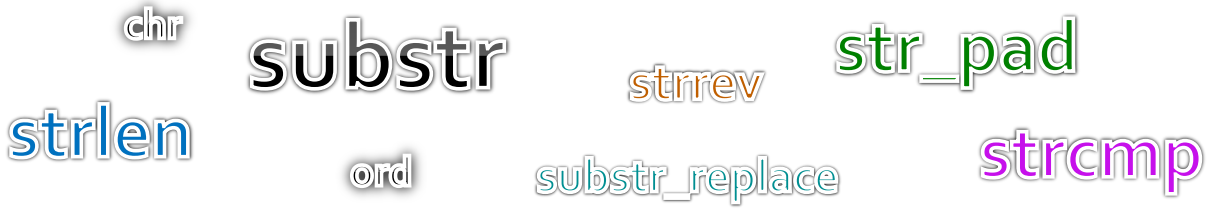
- **バイナリ**変換プログラミングが趣味です



神は人に
２進数を
与えたもう

発表題目

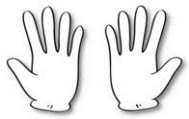
- (pure) PHP で**バイナリ**変換プログラミング
  - ここで云う pure は PHP の**標準関数**だけでという意味です



  - 資料は公開してるので、この場で分からなくなくても大丈夫です
    - http://d.hatena.ne.jp/yoya/20111112/php

対象者

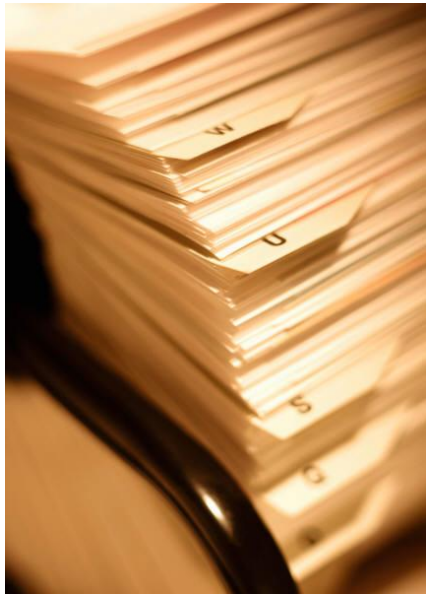
- Web開発に**飽きてきた**人向け
- ２進数を知っている。指を使ってよいので2進を16進に変換できる



- バイト(Byte)とかビット(Bit)という言葉を聞いたことがある
- PHP には(実は)型があって string, integer, flort で処理が違うという話を何処かで聞いたことがある

発表内容

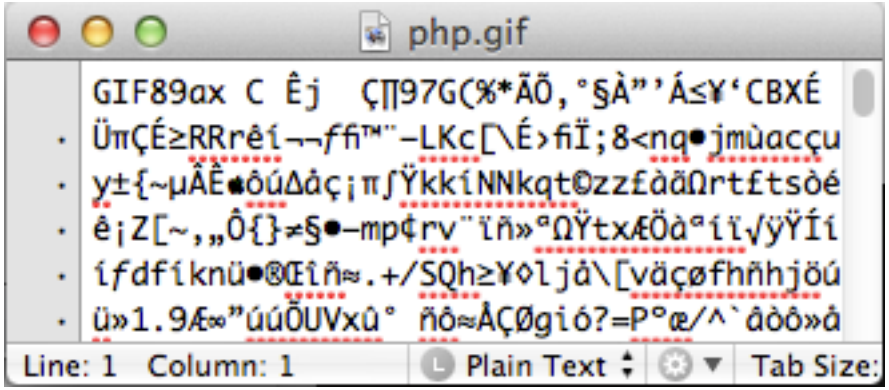
- バイナリについて
- ビット(Bit)とバイト(Byte)について
- PHP でバイト(Byte)処理
- PHP でビット(Bit)処理
- openpear/IO_Bit パッケージの紹介
- IO_Bit の応用事例 (IO_SWF, IO_Zlib)



バイナリについておさらい

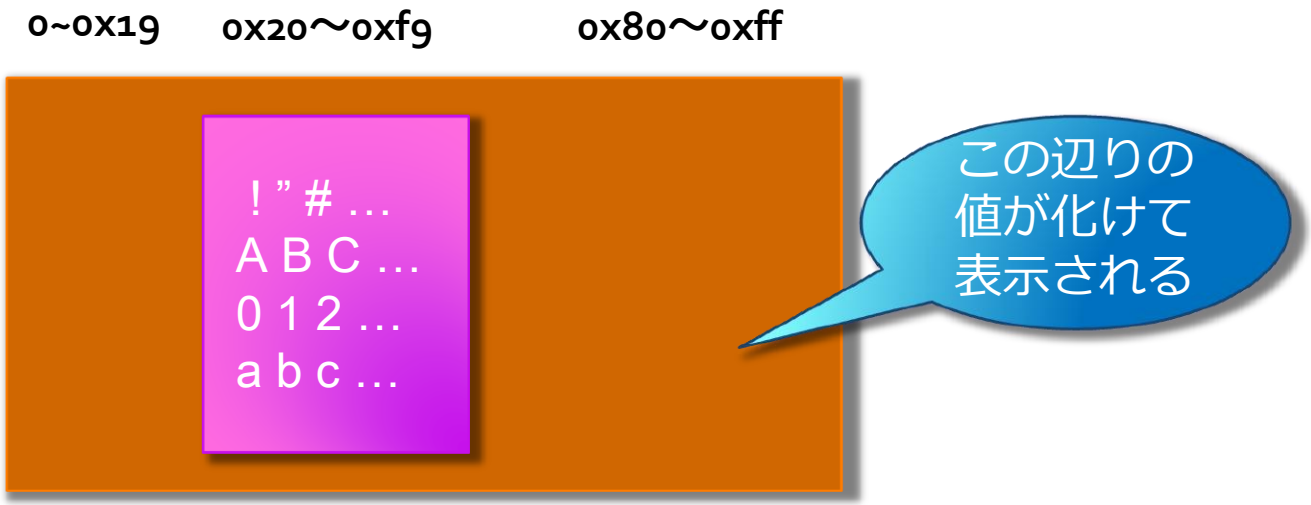
- バイナリって何？
  - 本来は、コンピュータが処理し易い 0,1 の２進値データ
  - 世間的には、**テキスト以外のデータ** (狭義のバイナリ)
    - エディタで開いて文字化けするデータ

GIFファイル
(php.gif)



バイナリとテキスト

- １バイトで0～255の値を表現できるけど、テキストはその一部しか使わない。(日本語の話は棚に置きます)



- バイナリの方がより多くの情報を詰められる

バイナリの実例

- バイナリの種類
  - マルチメディア系ファイル (JPEG, PNG, MPEG, AVI…) 
  - 実行ファイル (exe, a.out, jar, …) 
    (↑最近はバイナリというとこれを差す事が多い)
  - ネットワーク上の通信データ (TCP, ISDN, …) 
  - 暗号化されたデータ (zip, gzip, …) 

色々ありますネ！

バイナリ処理の目的

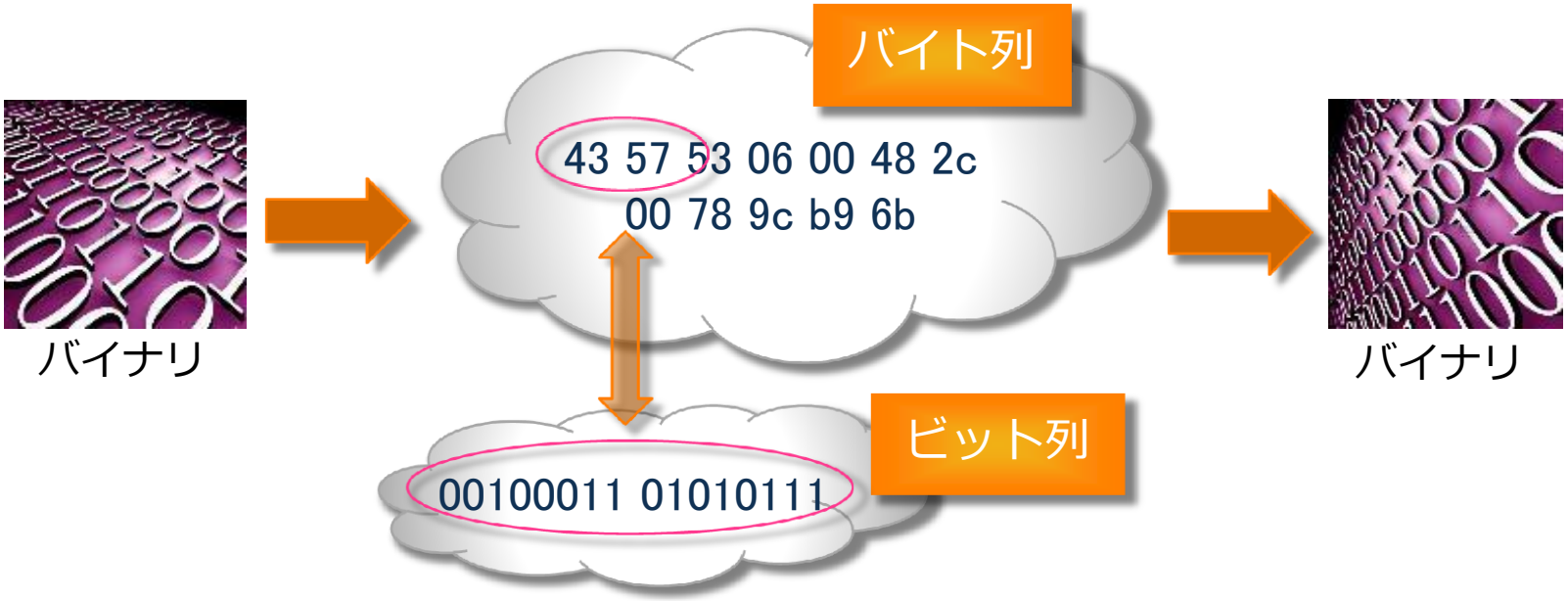
- Web サービスではテキスト以外に画像/動画データや、場合によると生の通信データを扱う事がある
- それらのデータを独自に変換する事で
  - より多くの種類のクライアント端末でサービスが受けられたり

  - 通信データ量を減らせたり

- エディタで開いて読めないから諦める。だと勿体ない

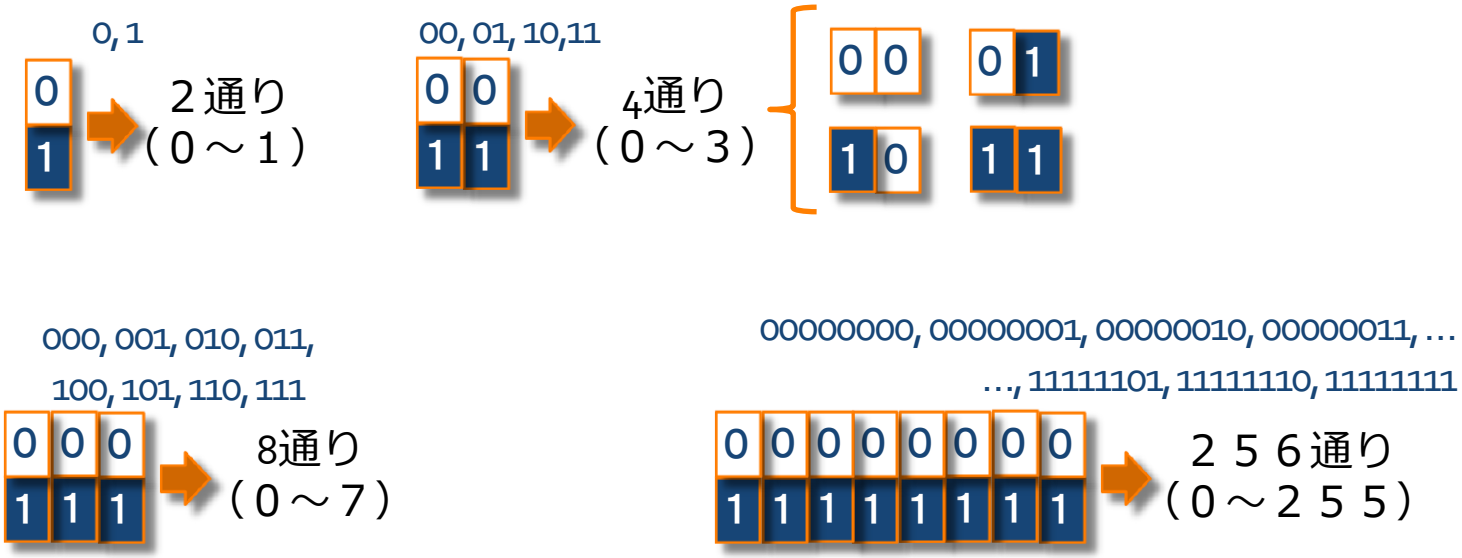
ビット(Bit)とバイト(Byte)

- コンピュータの処理(入出力や編集等)する単位
- バイナリ処理はこれらの単位でデータを操作する



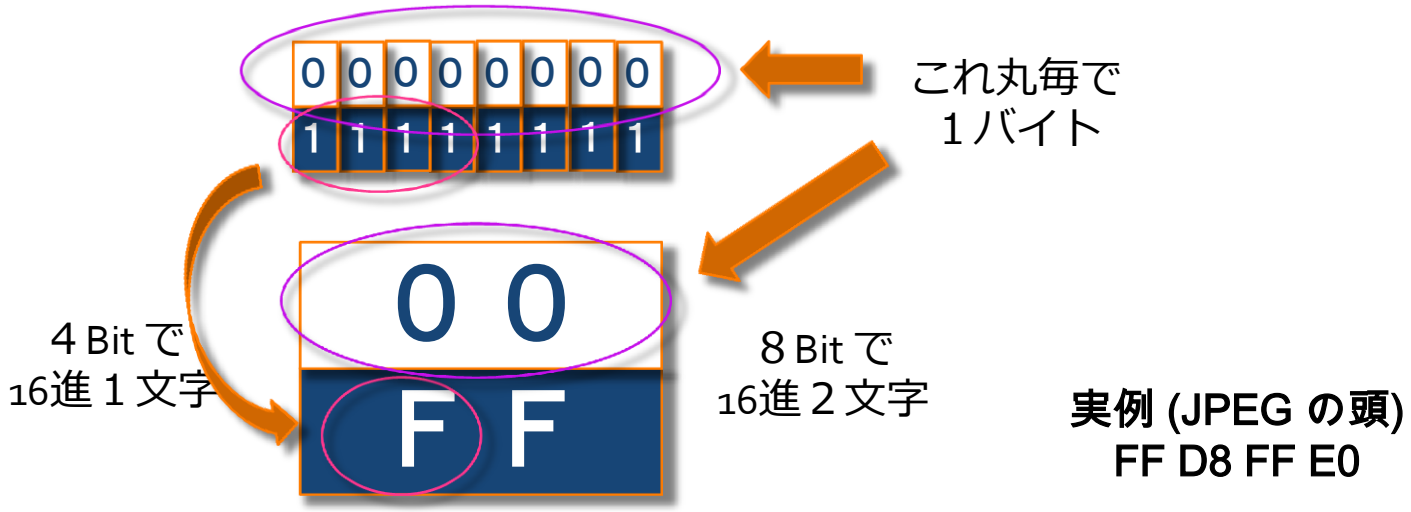
ビット(Bit) (おさらい)

- ビット(Bit)について
  - ０と１を用いて２通りの状態を表現したもの
  - ビットを並べると４通り８通りと表現の幅が広がる



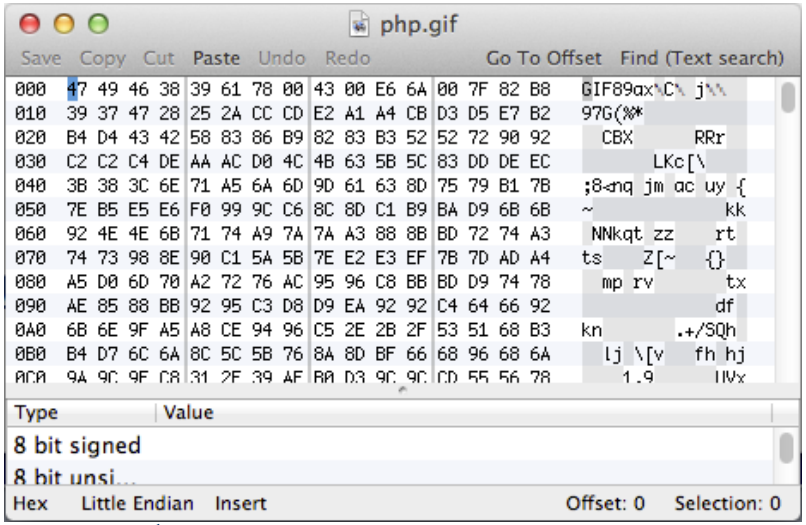
バイト(Byte) (おさらい)

- バイト(Byte)とは
  - 本来は、(欧米の) **１文字**を表すのに必要なビットの集まり
  - 狭義には**ビットを8つ**まとめた単位
  - １６進数で表現する事が多い (バイナリエディタの表示)



バイナリエディタを使う

- バイナリエディタ諸々
  - Macintosh なら 0xED、Windows なら Stirling, Bz Editor



- 手動で弄るのが**面倒**になったら PHP の出番です。

ここから本題

PHP

&

Byte

PHP とバイナリと String型

- PHP の String 型でバイナリ処理が出来る
  - PHP は String 型に対し文字としての特別な事をしない

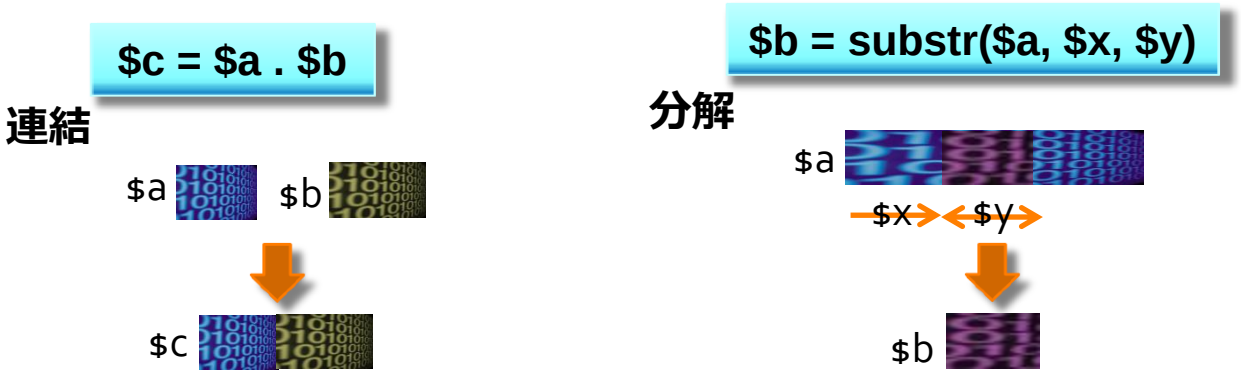
> PHP における文字列型は、**バイトの配列と整数値 (バッファ長) で実装**されています。バイト列を文字列に変換する方法については何の情報も持っておらず、完全にプログラマ任せとなっています。

    - http://www.php.net/manual/ja/language.types.string.php#language.types.string.details
  - 8bitスルー。 ¥0 終端もない ← この辺りの心配は無用
  - つまり、バイト(Byte)単位の処理は PHP でも簡単

String 型でバイト処理

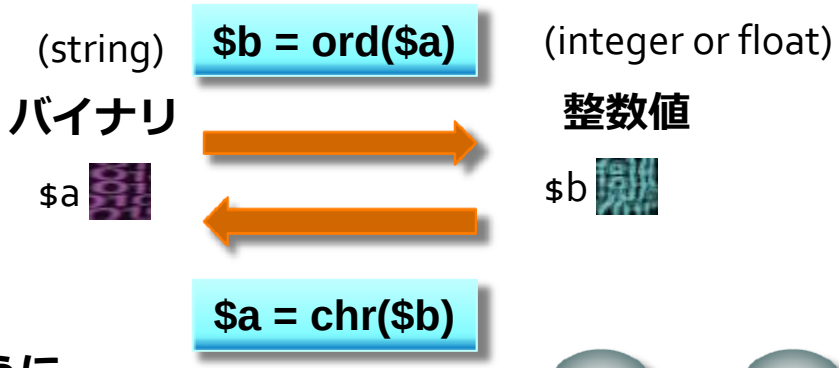
- ファイル入出力

- 連結と分解



String 型でバイト処理 ord,chr

+ バイナリと整数値との相互変換



実行例
(分かりやすいように
テキストで)

```
$a = 'Yoya';
$b = ord($a[0]);
echo $b;
```

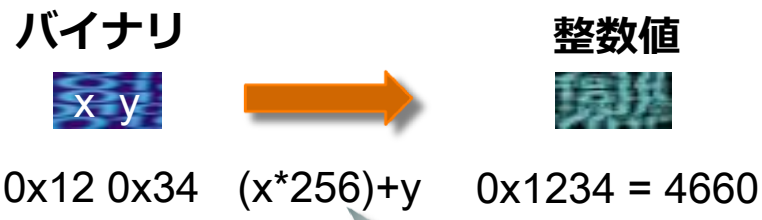
実行結果
⇒ 89
= (0x59)

```
$b = 89; $c = 111;
$a = chr($b).chr($c);
echo $a;
```

実行結果
⇒ Yo

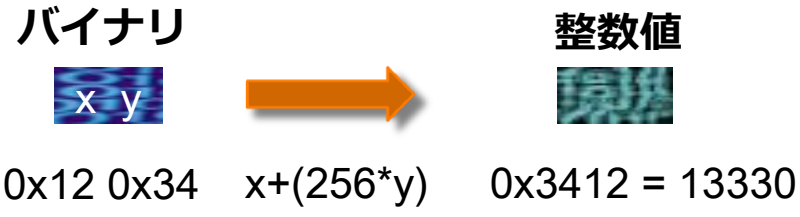
String 型でバイト処理 Endian

- ２バイト以上のバイナリと整数値の相互変換
  - Big Endian (MSB First)



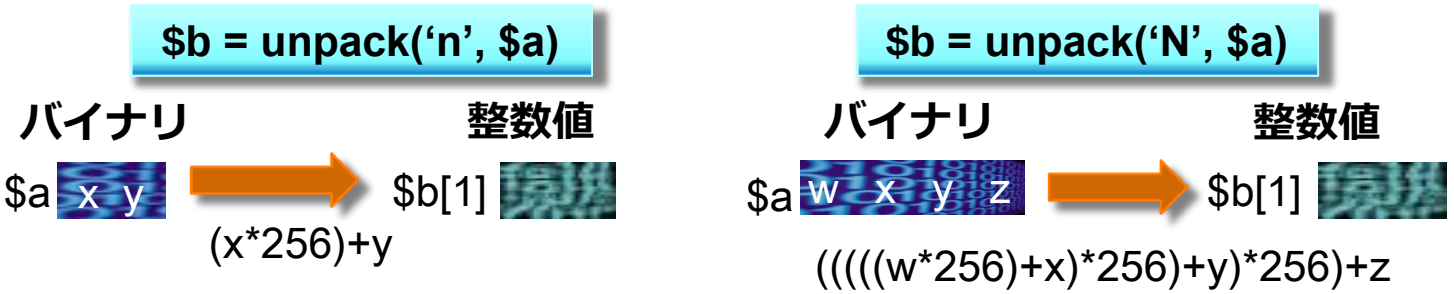
  - Little Endian (LSB first)

=0x100



String 型でバイト処理 BigEndian

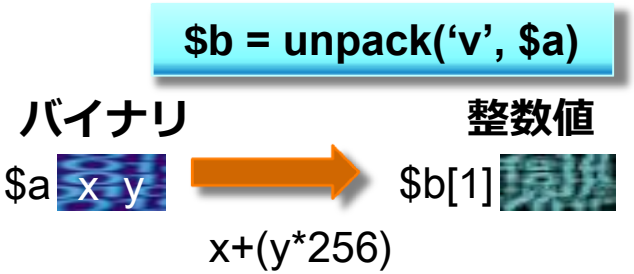
+ バイナリと整数値の相互変換 (Big Endian)

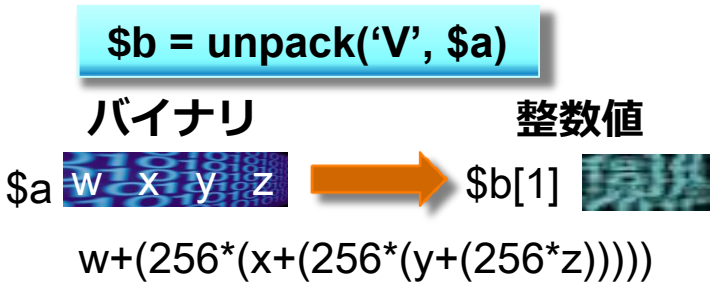


+ pack で逆変換

String 型でバイト処理 LittleEndian

+ バイナリと整数値の相互変換 (Little Endian)





+ pack で逆変換

String 型でバイト処理 その他

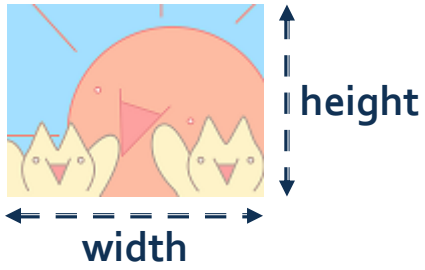
- strrev で前後リバース(逆順にする)
- substr_replace で一部入れ替え
- strcmp でバイナリ比較(一致するか否か)
- str_pad で同じバイトの繰り返し
- その他、str 系で色々な操作が可能。
  - (あくまで、Byteレベルで)

バイト処理の注意点

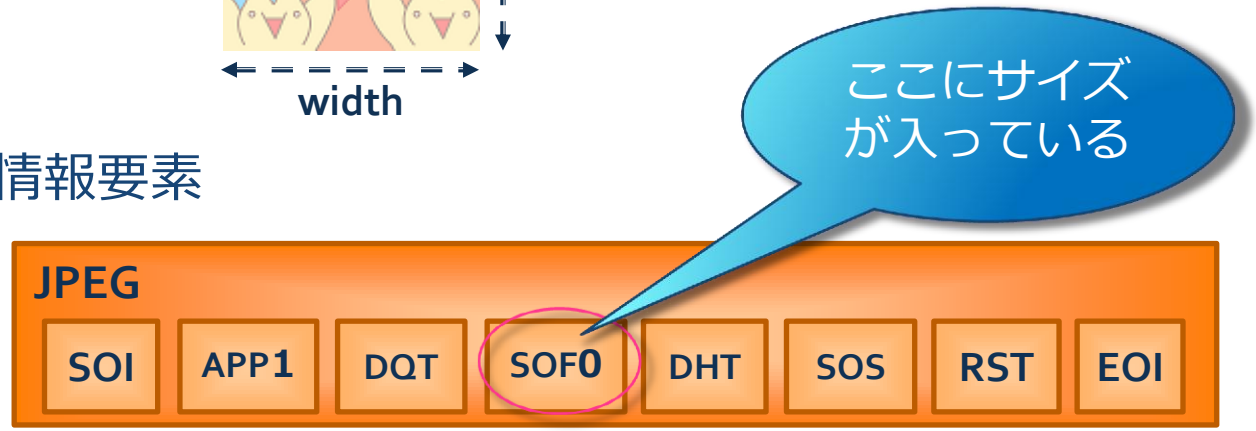
- $a[0]
  - 文字列を配列のように参照すると、($a の 0 番目の数値でなく)、**0番目の文字を切り出したもの**が取得できる。
  - $a[0] は substr($a, 0, 1) と同じ (C言語出身者は多分ココで躓く)
- unpack('N', ... と 'V'の PHP bug
  - N, V は unsigned long (32ビット値)のはずだが、**実際は signed long(符号付き)の値**が出てくる
  - 負の値が出てきたら 4294967296 を足して補正
    - 足すと (integer の範囲を超えて)型が float になる。要注意。
  - **pack は正でも負でも受理**してくれる。

バイト処理の実例 JPEG分解

- 例えば) JPEG 画像のサイズを抜き出す
  - http://www.w3.org/Graphics/JPEG/ ← 仕様はココ

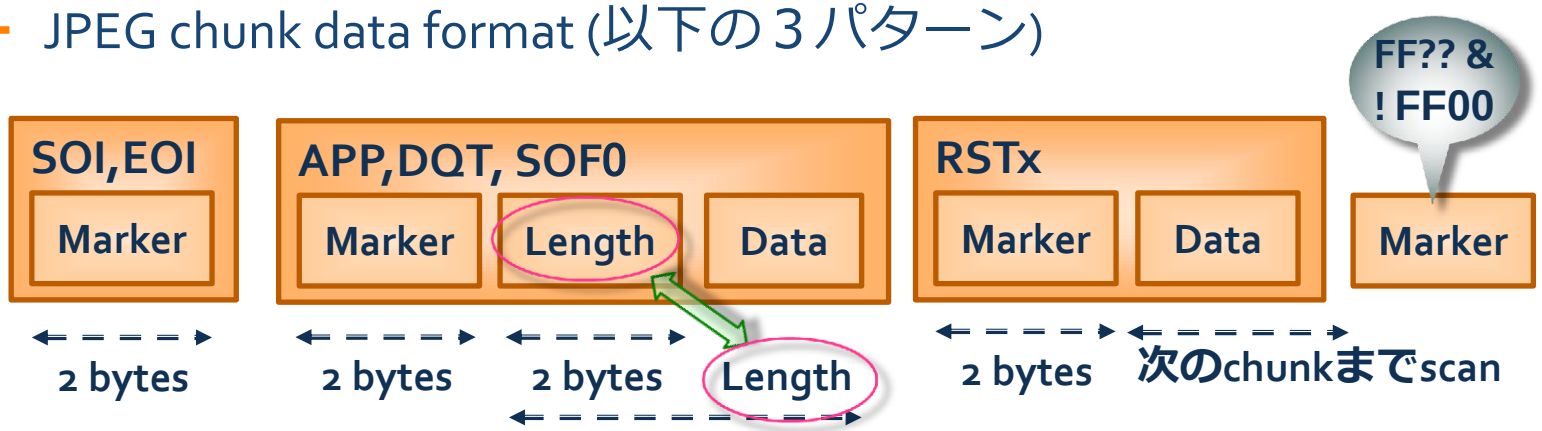


- JPEG 情報要素

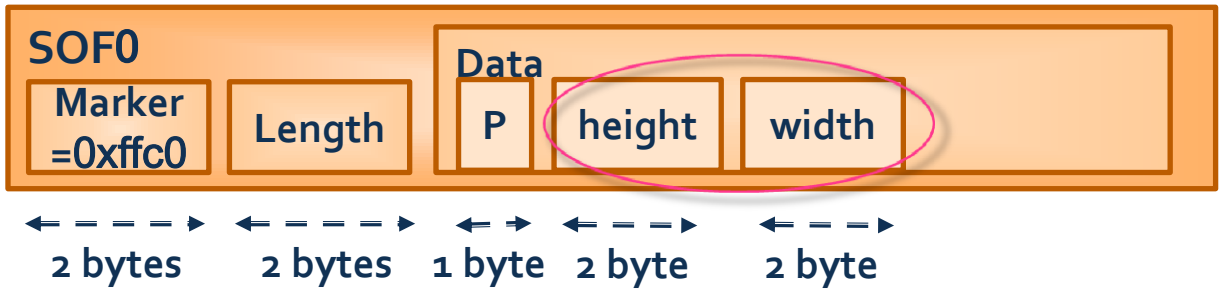


バイト処理の実例 JPEG形式

- JPEG chunk data format (以下の３パターン)



- SOF0 の中身



バイト処理の実例 JPEGサイズ

- Code

  Sample

- これで、

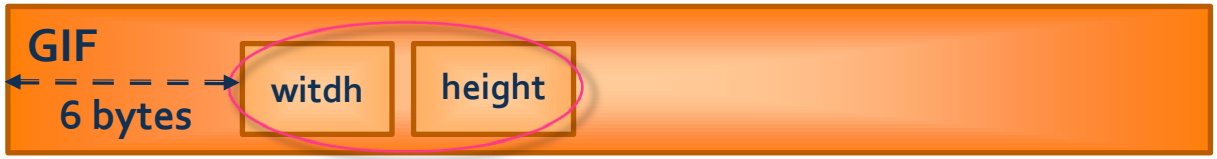
  width, height

  が抽出できる

```
<?php
$data = file_get_contents($argv[1]);  // JPEGfile input
for ($i = 1 ; $i < strlen($data) ; $i++) {
   switch(ord($data[$i++])) { // chunk marker
     case 0xD8: case 0xD9: // SOI (or EOI)
       break;                    // skip
     default:
       $len = unpack('n', substr($data, $i, 2));
       $i += $len[1];
       break;                    // skip
     case 0xC0: // SOF0
       $sof0 = unpack('CP/nH/nW', substr($data, $i + 2, 5));
       echo "width:{$sof0['W']} heigth:{$sof0['H']}¥n";
       exit (0);                 // OK
   }
}
```

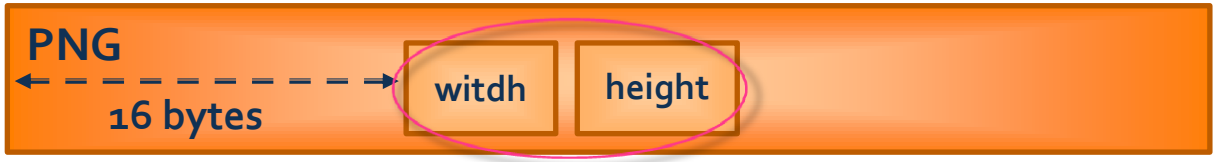
バイト処理の実例 GIF, PNG (簡単)

+ ついでに
+ GIF の
  画像サイズ




```
<?php
$data = file_get_contents($argv[1]);  // GIF file input
$size = unpack('vW/vH', substr($data, 6, 4));
echo "width:{$size['W']} heigth:{$size['H']}¥n";
```

+ PNG の
  画像サイズ




```
<?php
$data = file_get_contents($argv[1]);  // PNG file input
$size = unpack('NW/NH', substr($data, 16, 8));
echo "width:{$size['W']} heigth:{$size['H']}¥n";
```

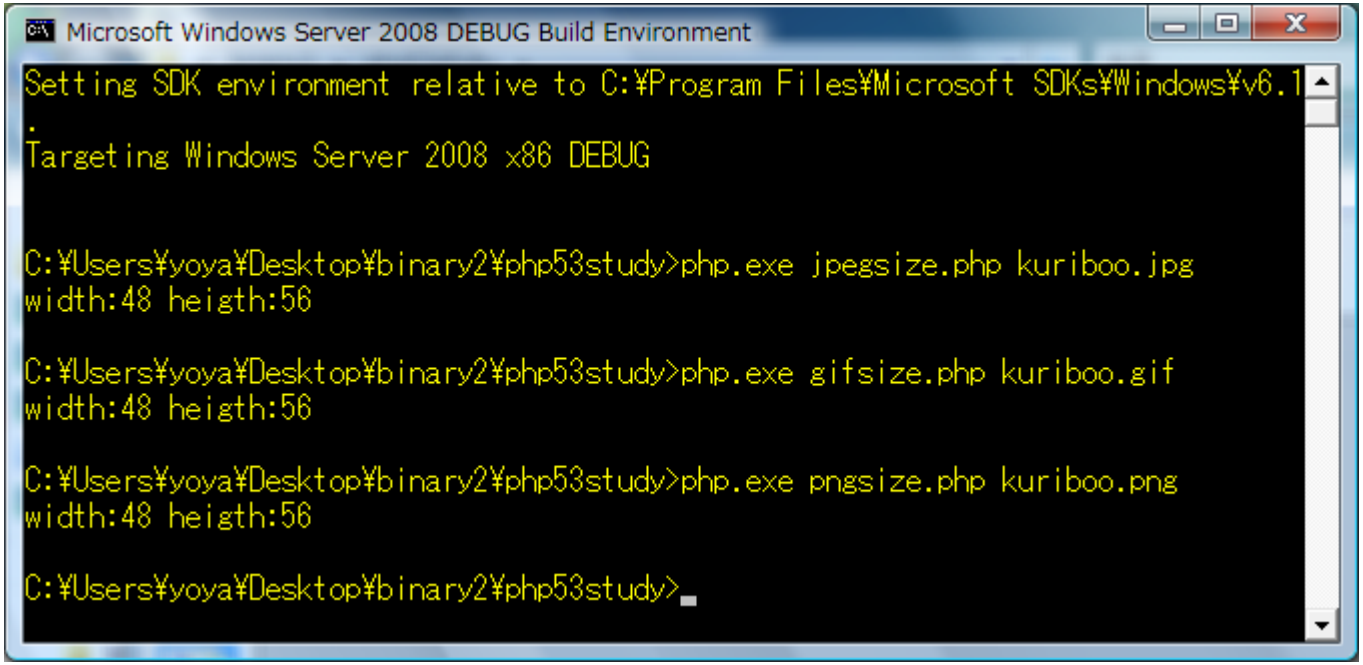
(Windows で PHP build)

- ネットワークが繋がらないので php.ext で動作デモ
- 最近は、IDE を操作せずに CLI だけで build 出来ます。
  - http://wiki.php.net/internals/windows/stepbystepbuild
  - 必要なファイルを揃えた後は、３つのコマンドだけ

```
buildconf
configure
nmake
```

JPEG/GIF/PNG サイズの実験デモ

- 端末上で動作デモ



ここからビットの話

PHP

&

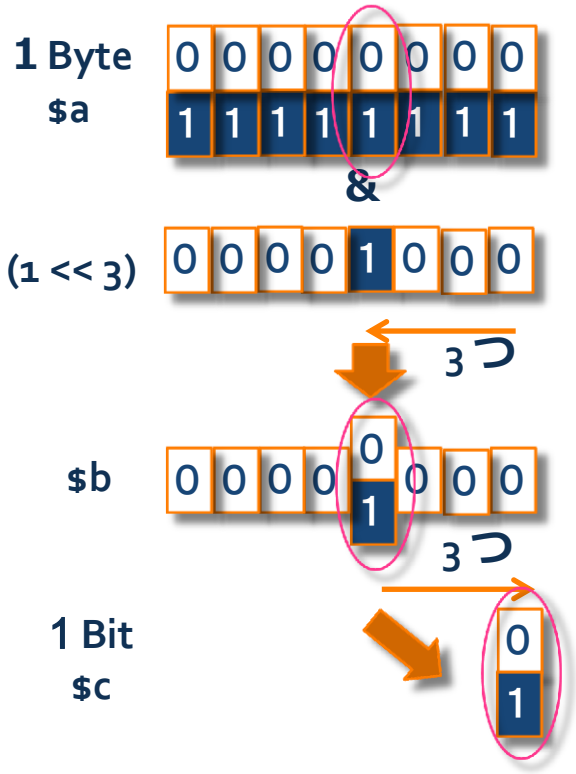
Bit

PHP でビット処理

- PHP にはビットを切り出すユーティリティはない
  - BitStream とか BitBuffer とかそういう感じの
  - ビット演算は出来るので、それで何とかする

ビット演算

- ビット演算(積とシフト)を使ったビット取り出し処理




```
$b = $a & (1 << 3);
```

```
$c = ($a & (1 << 3) >> 3;
```

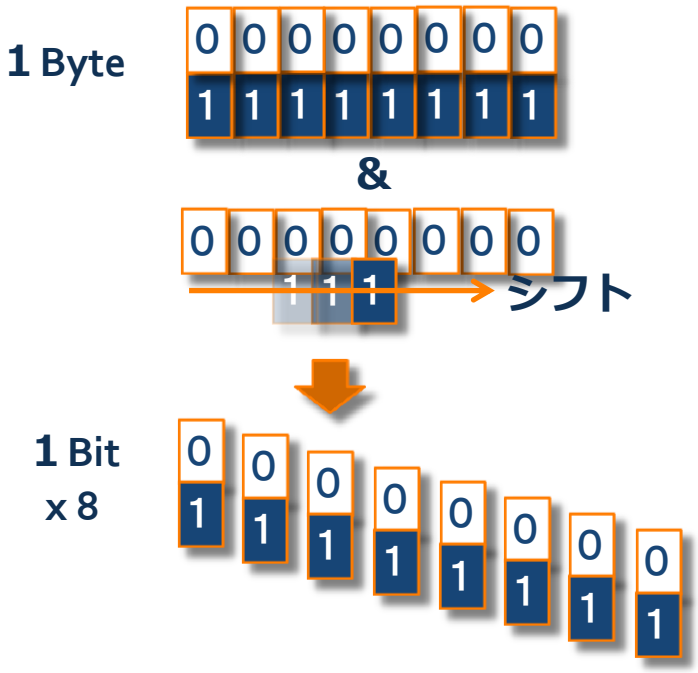
```
$c = $b >> 3;
```

3 を一般化して $n に

```
$c = ($a & (1 << $n) >> $n;
```

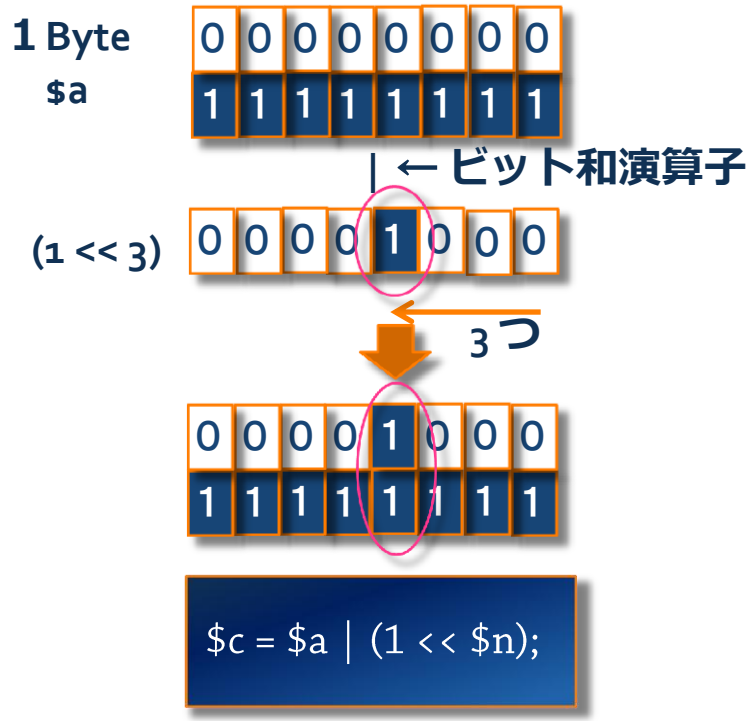
PHP でビット読み出し (Read)

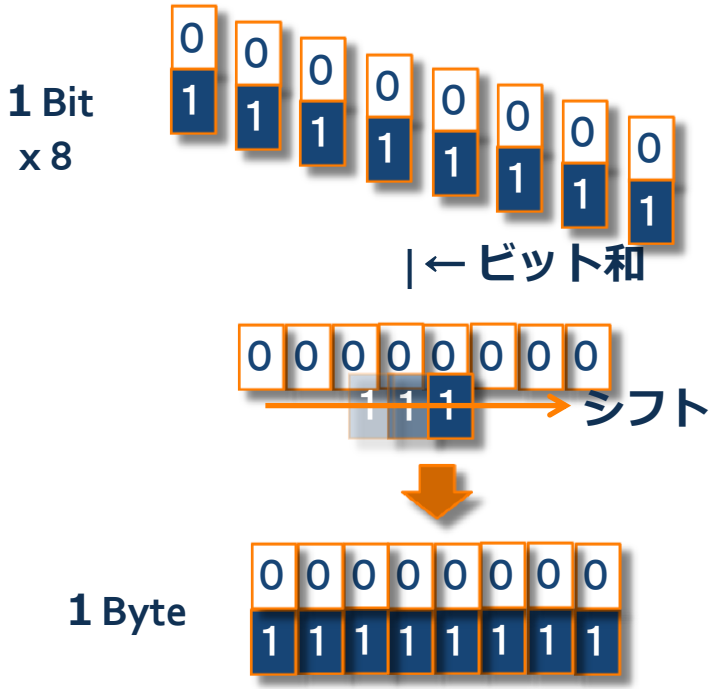
- 頭から 1 Bit 毎に読み出し



PHP でビット書き込み (Write)

- Bit を連結して Byte を生成





IO_Bit の紹介

- **Openpear ～ IO_Bit**

  - http://openpear.org/package/IO_Bit

    ビット処理のユーティリティです。いちいち、pack v したり、incremental に offset を処理するのが面倒だという人向けです。利用に制限はかけません。コピーも改変もご自由にどうぞ。MIT ライセンスにしました。

  - http://pwiki.awm.jp/~yoya/?IO_Bit

IO_Bit の使い方

- バイト入出力

```
$binary = file_get_contents($argv[1]);
$bit = new IO_Bit();
$bit->input($binary);
echo $bit->getUI8(); // get unsigned integer 8bit (=1 byte)
// $bit->putUI8(0x37);
// echo $bit-<output();
```

- ビット入出力

```
$binary = file_get_contents($argv[1]);
$bit = new IO_Bit();
$bit->input($binary);
echo $bit->getUIBit(); // get unsigned integer bit (=1 bit)
// $bit->putUIBit(1);
// echo $bit-<output();
```

IO_Bit の応用例

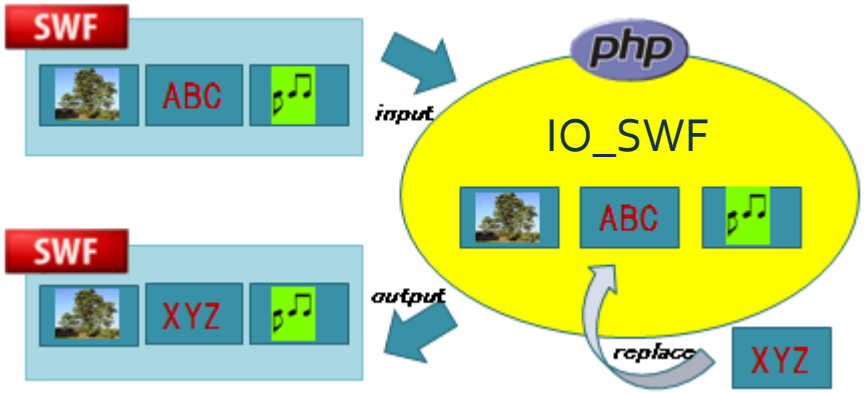
- openpear/IO_SWF (この後で説明)
  - Flash の実行ファイル(SWF)を編集

- openpear/IO_Zlib (時間があったら説明)
  - Zlib 圧縮されたデータを伸長する

IO_SWF の紹介

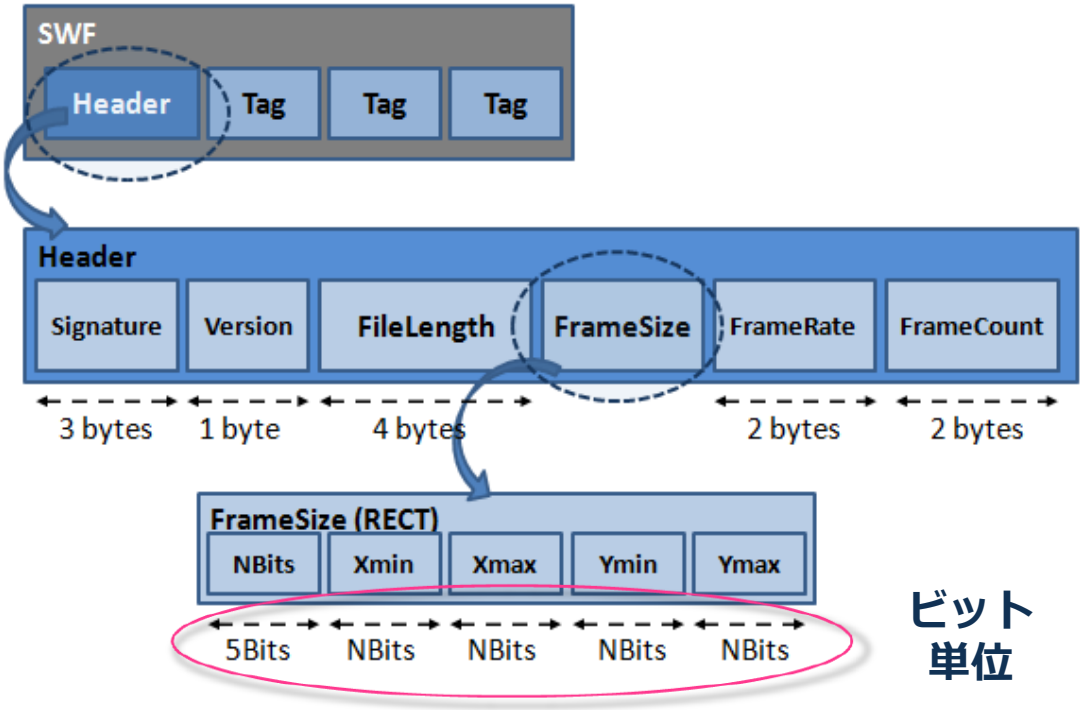
- **Openpear ～ IO_SWF**
  - http://openpear.org/package/IO_SWF

SWF バイナリを解釈/編集する為のライブラリです。 IO_Bit が必要です。主に Flash Lite 1.x/2.x を対象にしています。利用に制限はかけません。コピーも改変もご自由にどうぞ。MIT ライセンスにしました。



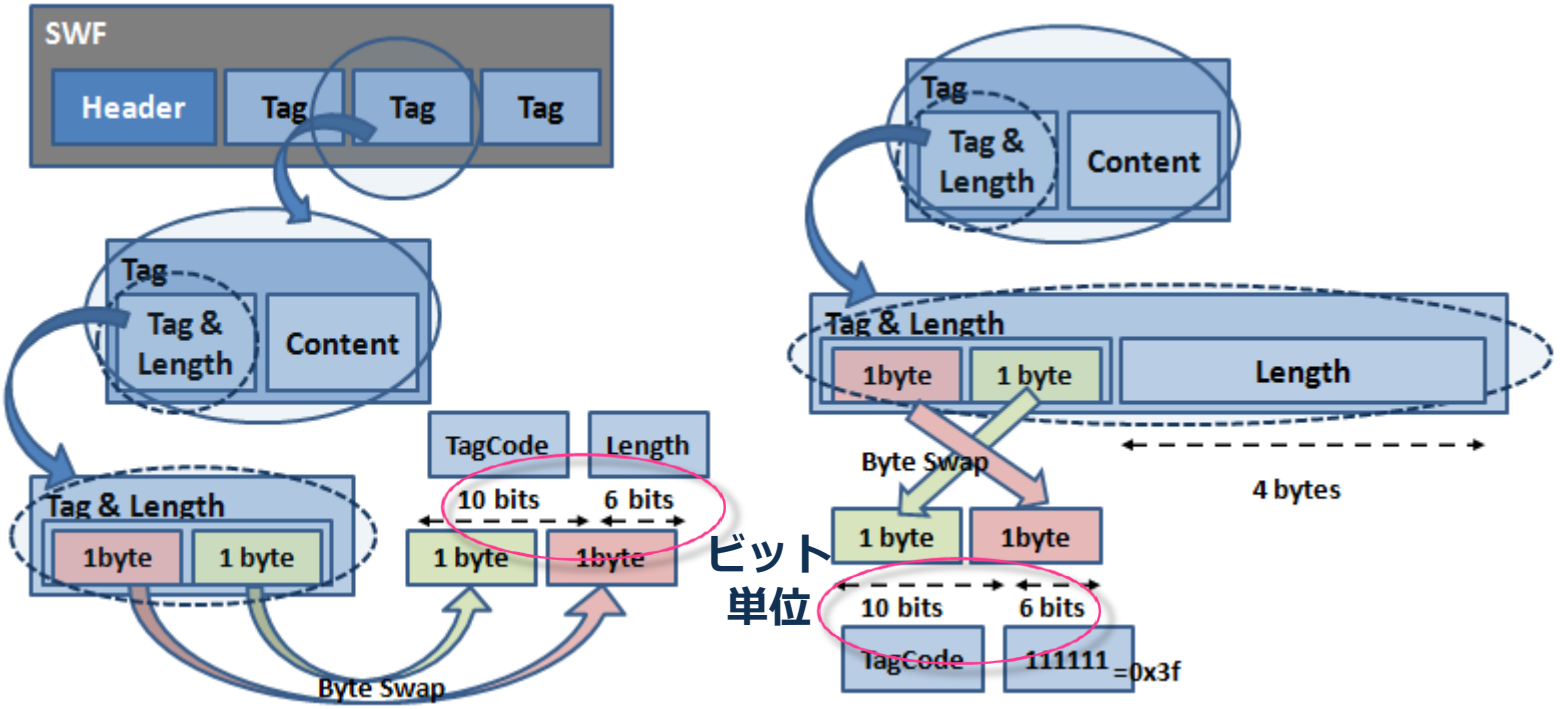
SWF のバイナリ構造 (ヘッダ)

- ビット処理が必要
  - http://labs.gree.jp/blog/2010/08/631/ から抜粋



SWF のバイナリ構造 (タグ)

- Short Tag と Long Tag



IO_Bit で SWF を解釈 (ヘッダ)

- IO/SWF.php から抜粋 (バイト処理)

```
function parse($swfdata) {
    $reader = new IO_Bit();
    $reader->input($swfdata);
    $this->_swfdata  = $swfdata;
    /* SWF Header */
    $this->_headers['Signature'] = $reader->getData(3);
    $this->_headers['Version'] = $reader->getUI8();
    $this->_headers['FileLength'] = $reader->getUI32LE();
```

IO_Bit で SWF を解釈 (RECT)

- IO/SWF/Type/RECT.php から引用 (ビット処理)

```
class IO_SWF_Type_RECT extends IO_SWF_Type {
   static function parse(&$reader, $opts = array()) {
      $frameSize = array();
      $reader->byteAlign();
      $nBits = $reader->getUIBits(5);
      $frameSize['Xmin'] = $reader->getSIBits($nBits);
      $frameSize['Xmax'] = $reader->getSIBits($nBits);
      $frameSize['Ymin'] = $reader->getSIBits($nBits);
      $frameSize['Ymax'] = $reader->getSIBits($nBits) ;
      return $frameSize;
   }
```

IO_SWF の使い方

- 使い方

```
$swfed = new IO_SWF_Editor(); // インスタンス生成

$swfed->parse($swfdata); // SWFバイナリ読み込み

// 何らかの編集するメソッドを呼ぶ

echo $swfed->build(); // 編集結果の SWF バイナリを出力
```

- http://pwiki.awm.jp/~yoya/?IO_SWF

IO_SWF の利用例

- SWF ファイルの解析

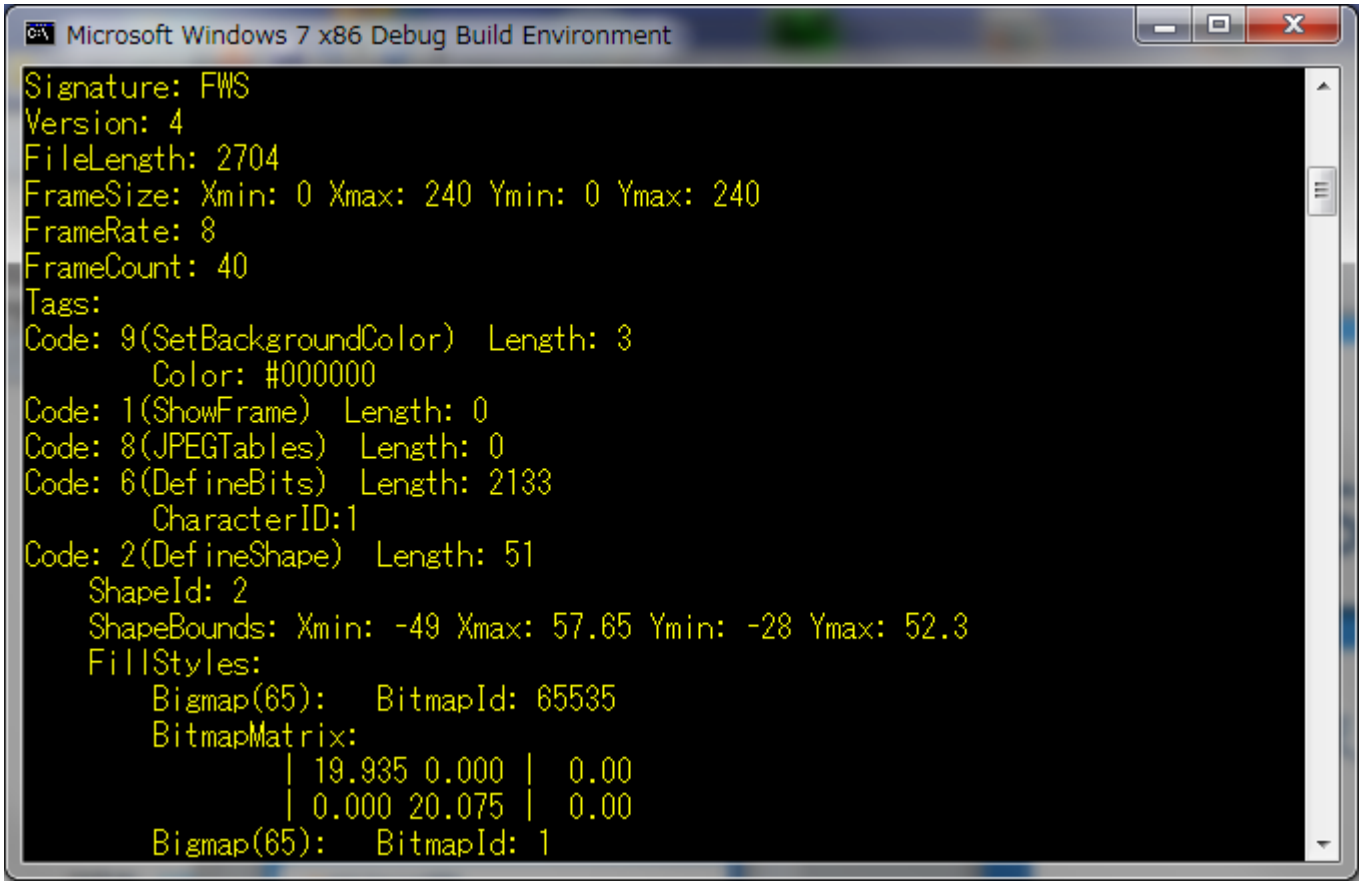
```
$swfdata = file_get_contents($argv[1]);
$swfed = new IO_SWF_Editor();
$swfed->parse($binary);
$swfed->dump(array('hexdump' => true));
```

- SWF 内コンテンツの入れ替え

```
list($prog_name, $swf_file, $bitmap_id, $bitmap_file) = $argv;
$swf_data = file_get_contents($swf_file);
$bitmap_data = file_get_contents($bitmap_file);
$swfed = new IO_SWF_Editor();
$swfed->parse($swf_data);
$swfed->replaceBitmapData($bitmap_id, $bitmap_data);
echo $swfed->build(); // 入れ替え後のSWF バイナリ出力
```

IO_SWF の実験デモ

- 端末上で動作デモ



IO_Zlib の紹介

- **Openpear ～ IO_Zlib**

  - http://openpear.org/package/IO_Zlib

Zlib フォーマットの分解ルーチンです。
Inflate(伸張)は動作しますが、deflate(圧縮)は btype:0 (=無圧縮)のみ対応します。

Zlib って何？

- **データ圧縮アルゴリズムに Deflate というモノがあり、そのコンテナ形式**
  - **Deflate の入れ物として有名なものに Gzip と Zlib がある**
  - 詳しくはここにリンクまとめ
    - → http://pwiki.awm.jp/~yoya/?Deflate
- **Gzip は gzip コマンドで生成されるファイル形式**
- **Gzip はファイル名やタイムスタンプが入れられるが、純粋に圧縮したい場合は、より簡略な Zlib 形式が用いられる。**

Zlib について

- ハフマン符号とLZ77を組み合わせた圧縮。
  - ハフマン符号は符号の出現頻度に応じて、頻出する符号に短いビット列、稀な符号に長いビット列を割り当てる事でデータ量を減らす手法
  - LZ77は同じパターンがある時にはその繰り返しの長さを指定する事で、データ量を減らす手法
  - 真面目に話すと**丸一日かかる**ので、説明はココまで。
- ハフマン符号はビット単位の処理が必要な符号化方式
- IO_Bitの出番！

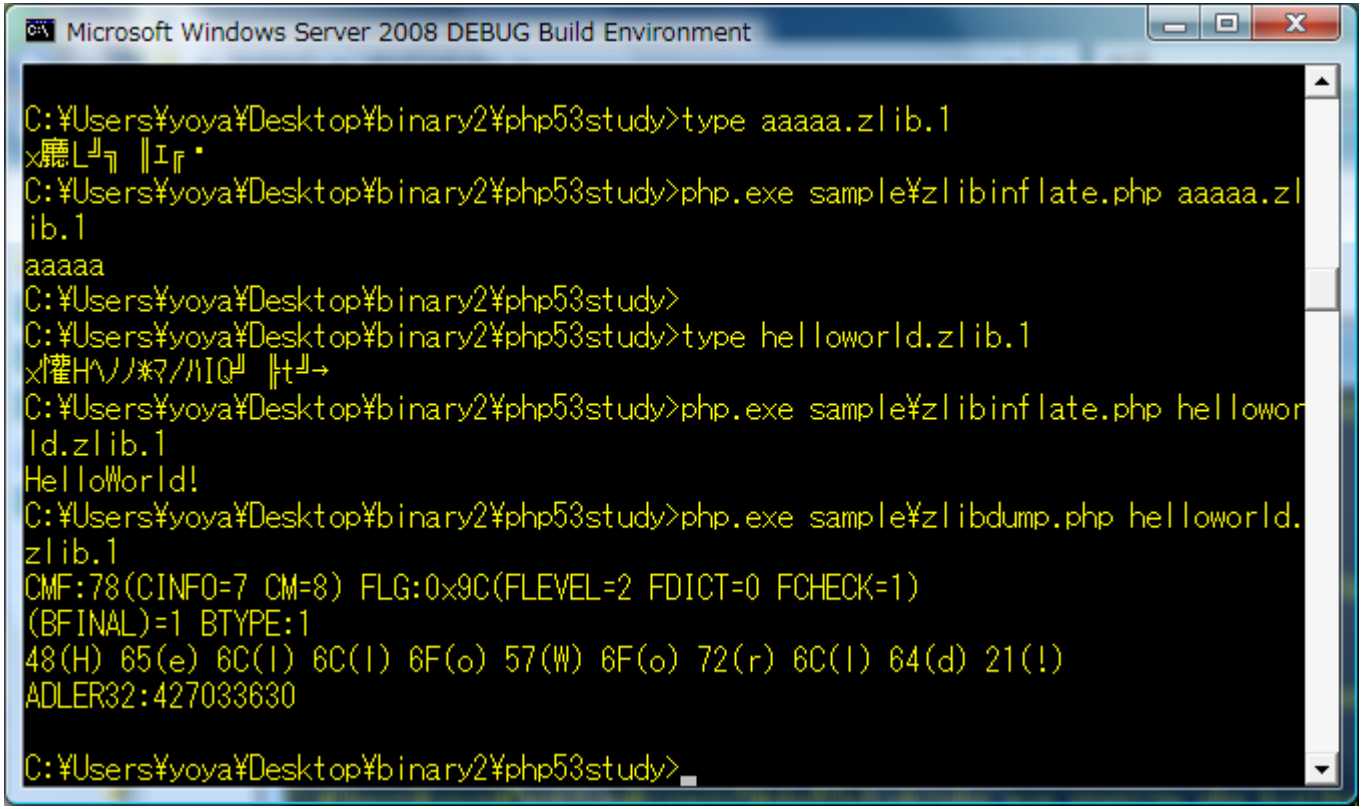
IO_Zlib の使い方

- 使い方

```
$zlib = new IO_Zlib(); // インスタンス生成

$zlib->parse($zlibdata); // Zlib 圧縮データ読み込み

// 何らかの編集するメソッドを呼ぶ

echo $zlib->build(); // 伸長結果のデータを出力
// $zlib->dump(); // フォーマット解析用
```

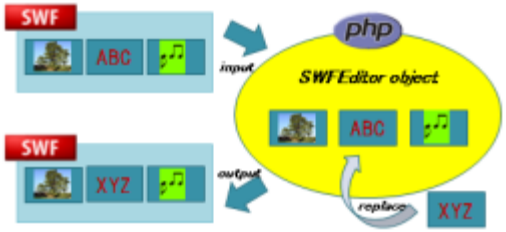
IO_Zlib の動作デモ

- 端末で動作デモ



エクスキューズ

- 実はSWFEditorというPHP拡張で同じ事出来ます。
  - http://sourceforge.jp/projects/swfed/
  - 実サービスで使うならこっちです。



- 実は標準関数に gzuncompress があります。
  - http://php.net/manual/ja/function.gzuncompress.php
  - IO_Zlib は実装サンプル、又はフォーマットの解析用で。。

要望

- 他の言語で、これに似た発表があれば教えて下さい。

質問

- getimagesize の方がよくないですか？
  - 普通はそっちを使いますが、getimagesize はサイズ以外の余計な物も返すので、自前で処理した方が軽いかもしれない。
- 拡張子でファイルの中身を判断するだけでなく、頭のバイナリってみたりします？
  - magick (/usr/share/mime/magick 等)ファイルを見ると、典型的なファイル形式の頭４バイトが列挙されてて便利です。
- ZIP パスワードの入力を自動化できません？
  - Deflate(RFC1951),Zlib(RFC1950), Gzip(RFC1952) みたいには仕様が公開されていないので、調べていません。

以上

- ご清聴ありがとうございました