STDIO.PL リファレンス

STDIO.PL VERSION 9 by WEB POWER

はじめに

STDIO.PL(STandarD Input Output – Perl Library)　CGI　Perl
CGI　(　)

CGI
stdio.pl　CGI

CGI　sendmail
JIS
MIME
stdio.pl　stdio.pl
CGI-Perl
stdio.pl　stdio.pl

stdio.pl　CGI

2003　8　1
WEB POWER

CONTENTS

STDIO.PL の使い方

stdio.pl

リファレンスの見方

、[ ]
、( ) (Perl4)
、 0

使用上の注意

、

、

、 OS
Web

プログラミング上の注意

◆いきなりサーバー上で動作させない

Perl

◆セキュリティー対策はプログラマの責任で
stdio.pl (
HTTP)

◆メールの送信テストは必ず実在するアドレス宛に

ライブラリのロード

stdio.pl stdio.pl require

```
require "stdio.pl";
```

関数の呼び出し

stdio.pl stdio
jcode.pl
jcode.pl

```
      = stdio::     (    ); (Perl4       stdio'          )
```

) getImageSize

```
($type, $width, $height) = stdio::getImageSize($file);
```

値渡しと参照渡し

( )

(Perl4) ( ) ( )
( ) stdio.pl

▼例 1. 値渡しの例

```
stdio::getImageSize("image.gif");
```

▼例 2. 参照渡しの例（変数名の前に"¥"を付けるとその変数へのリファレンスを取得）

```
stdio::sendmail(¥%header, $body);
```

▼例 3. 参照渡しの例（$header には%header へのリファレンスが格納されている）

```
$header = ¥%header;
stdio::sendmail($header, $body);
```

▼例 4. 参照渡しの例（無名ハッシュ変数へのリファレンスを渡す）

```
stdio::sendmail({To=>'*',From=>'*',Subject=>'*'}, $body);
```

%hash = (key1=>val1, key2=>val2, ..) ( ) (
) ( )

■値渡しと参照渡しの両方に対応した関数

base64encode/base64decode
urlencode/urldecode trString setLink 6

( )

▼例 5. 参照渡しの例（$str の中身を書き換える）

```
stdio::urlencode(¥$str);
```

▼例 6. 値渡しの例（$str の中身はそのまま。処理された値は戻り値で取得する。）

```
$new_str = stdio::urlencode_($str);
```

パッケージ変数

stdio.pl

stdio.pl stdio

■version（stdio.pl のバージョン情報）

stdio.pl "stdio.pl/8.1x"

■max_byte（最大受信バイト数）

getFormData/getMultipartFormData/getUrlencodedFormData
128K

128K ( )

```
$stdio::max_byte = 1048576;
```

■ses_byte（セッションファイルのレコードサイズ）

setSession/getSession

```
$stdio::ses_byte = 1024;
```

■tmp_dir（テンポラリファイルを格納するディレクトリのパス）

sendmail
setSession/getSession getFormData/getMultipartFormData
lock/unlock/lockCheck /// ///

```
$stdio::tmp_dir = '/tmp/';
```

■sendmail（sendmail コマンドのパス）

sendmail sendmail

```
$stdio::sendmail = '/usr/lib/sendmail';
```

■inet（socket 関数呼び出し時のドメイン）

socket 2 1
Socket.pm PF_INET
openSocket

```
$stdio::inet = 2;
```

■stream（socket 関数呼び出し時のタイプ）

socket 3 1
Socket.pm SOCK_STREAM
openSocket

```
$stdio::stream = 1;
```

setCookie

クライアントにクッキーを渡す

◎ = **stdio::setCookie**(☆[,　[, □[, ■[, ○[, ●[, △]]]]]])

☆	ﾊｯｼｭ(参照)	(　)
	配列(参照)	(　)
★	スカラー	ID(　)
□	スカラー	(　-1　)
■	スカラー	(　)
○	スカラー	(　)
●	スカラー	(　)
△	スカラー	(　)
◎	スカラー	(　7　)

setCookie　HTTP　(Cookie)

Web

Web

ID(

)

クッキーの有効期限

3　(Fri, 31-Dec-1999 23:59:59 GMT　)

(　)　-1

クッキーの有効範囲

(　)　4　5　4

/　Web

5

hostname.ne.jp　hostname.ne.jp

.hostname.ne.jp　aaa.hostname.ne.jp　bbb.hostname.ne.jp　hostname.ne.jp

クッキーの ID と値

ID　2　ID

1　1　(

)

戻り値にクッキーの本体を返す

7 ( )
(STDOUT) HTML <meta http-equiv="Set-Cookie" content=" " /> ( )
JavaScript ( ) 7
HTML JavaScript

記 述 例

```
$cookie = stdio::setCookie(¥%cookie, $cookie_id, $expires, $path, $domain, $secure,
 $return_value);
```

サンプル

```
#   Content                    (7200  =2              )
print "Content-Type: text/html¥n";
stdio::setCookie(¥%cookie, "MyCookie", 7200);
print "¥n";

#   HTML   META                (                          )
$cookie = stdio::setCookie(¥%cookie, "MyCookie", "", "", "", "", 1);
print qq|<meta http-equiv="Set-Cookie" content="$cookie" />¥n|;

#   JavaScript             (                          )
$cookie = stdio::setCookie(¥%cookie, "MyCookie", -1, "", "", "", 1);
print qq|<script type="text/javascript">¥n|;
print qq|<!--¥n|;
print qq|  document.cookie = "$cookie"; ¥n|;
print qq|// -->¥n|;
print qq|</script>¥n|;
```

Tips 携帯電話でクッキーは？

i-mode
utn ( )
i-mode i-mode utn HTML a form utn
( ) CGI utn ID stdio.pl
setSession/getSession

http://www.nttdocomo.co.jp/p_s/imode/tag/utn.html

getCookie

クライアントからクッキーを取得

◎ = **stdio::getCookie**(☆[, ★])

☆ : ﾊｯｼｭ(参照) ( )
配列(参照) ( )
★ : スカラー ID( )
◎ : リスト ( )

getCookie setCookie 1
setCookie

記 述 例

```
@keys = stdio::getCookie(¥%COOKIE, $cookie_id);
```

サンプル

```
#
%COOKIE = ();
@keys = stdio::getCookie("MyCookie", ¥%COOKIE);

#
foreach (@keys) {
    print "$_ : $COOKIE{$_}¥n";
}
```

Tips クッキーっておいしいの？

IE6 P3P (
)

setSession / getSession

セッション管理

◎ = **stdio::setSession**(☆[, ★[, □[, ■]]])

◎ = **stdio::getSession**(☆, ★[, □[, ○]])

☆ : スカラー ( )
/// /// $stdio::tmp_dir
★ : ﾊｯｼｭ(参照) ( )
setSession
□ : スカラー ID(IP)
■ : スカラー (3600 =1)
○ : スカラー (-1 1)
◎ : スカラー 1 ( )
0 (ID)

setSession/getSession

ID( )

セッションファイルのレコードサイズ

setSession/getSession
FTP
1

■レコードサイズ

1
stdio.pl $stdio::ses_byte stdio.pl

```
require 'stdio.pl';
$stdio::ses_size = 1024;
```

1 ID
ID URL
$ses_byte

記 述 例

```
$result = stdio::setSession($session_file, ¥%session, $session_id, $expires);
$result = stdio::getSession($session_file, ¥%session, $session_id, $expires);
```

サンプル

```
# IP          ID                (          )
%session = ();
stdio::getSession($ses_file, ¥%session);

# IP          ID
stdio::setSession($sef_file);

# IP          ID
stdio::setSession($ses_file, ¥%session);
```

getFormData / getUrlencodedFormData / getMultipartFormData

標準入力(クエリー文字列)からデータ(ファイル)取得

◎ = **stdio::getFormData**(☆[, ★[, □[, ■[, ○]]]])

◎ = **stdio::getUrlencodedFormData**(☆[, ★[, □[, ■]]])

◎ = **stdio::getMultipartFormData**(☆[, ★[, □[, ■[, ○]]]])

☆ : ﾊｯｼｭ(参照) ( )
★ : スカラー (< > & " → < > & ")
(1 2
)
□ : スカラー (sjis/jis/euc/SJIS/JIS/EUC)
■ : スカラー ( )
○ : スカラー

/// /// $stdio::tmp_dir
◎ : リスト ( )

getFormData/getUrlencodedFormData/getMultipartFormData ( )

CGI
getFormData (URL)
getMultipartFormData ( )
getUrlencoded-FormData
(jcode.pl)

戻り値

( )
each
1
( )

重複するキーの扱い

select multiple
4

"%PARAM"

```
name=Taro&name=Jiro&name=Saburo&mail=foo@hostname.jp
```

$PARAM{'name'} "Saburo" 4
4 4 "¥t"( ) $PARAM{'name'}

```
Taro>    Jiro>    Saburo  (  >          )
```

マルチパートフォームデータ

( )

5

128K ( )

(@stdio::file)

"->***"

▼ハッシュ変数に同時に格納されるファイル情報

```
*          サーバー側での保存されているパス (  file/1058337893-1.tmp)
*->name    クライアント側でのファイル名 (  image.gif)
*->path    クライアント側でのパス (  C:¥Document and Settings¥User¥image.gif)
*->type    ファイルの MIME タイプ (  image/gif)
*->size    ファイルサイズ (  38641)
```

送信データの上限

128K(131,072) stdio.pl

$stdio::max_byte

128K $max_byte

$max_byte stdio.pl

stdio.pl

```
require 'stdio.pl';
$stdio::max_byte = 1048576 * 10;
```

128KB

( ) 128K

$max_byte

その他

- "&" ";"
- 2 < > & " < > & " 2 2
- 3 (sjis/euc/jis) (SJIS/EUC/JIS)

jcode.pl

記 述 例

```
@keys = stdio::getFormData(¥%PARAM, 2, $jcode, $separator, $file_dir);
@keys = stdio::getMultipartFormData(¥%PARAM, 2, $jcode, $separator, $file_dir);
@keys = stdio::getUrlencodedFormData(¥%PARAM, 2, $jcode, $separator);
```

サンプル

```
####                    ####

#
%PARAM = ();
@keys = stdio::getFormData(¥%PARAM);

#
foreach (@keys) {
    print "$_ = $PARAM{$_}¥n";
}
```

```
####                    ####

#
%PARAM = ();
@keys = stdio::getMultipartFormData(¥%PARAM, 2, "EUC", "", $file_dir);

#
foreach (@keys) {

    #                 ($PARAM{$_}                                )
    if ($PARAM{"$_->name"}) {
        print qq|$key =           ($PARAM{"$_->path"} $PARAM{"$_->size"}     )¥n|;

    #
    } else {
        print "$key = $PARAM{$key}¥n";
    }
}
```

▼ ①の実行結果（引数：name=%C2%C0%CF%BA&mail=address@hostname.co.jp&uri=http%3A%2F%2Fwww.hostname.co.jp%2F）

```
name =
mail = address@hostname.co.jp
uri = http://www.hostname.co.jp/
```

▼ ②の実行結果

```
name =
mail = address@hostname.co.jp
file =           (C:¥Document and Settings¥User¥image.gif 342      )
```

setQueryString / setHiddenForm

ハッシュからクエリー文字列・HIDDEN フィールドを作成

◎ = **stdio::setQueryString**(☆[, ★[, □[, ■]]])

◎ = **stdio::setHiddenForm**(☆[, ★[, □[, ■]]])

☆ : ﾊｯｼｭ(参照) ( )
★ : 配列(参照) ( ) ( )
□ : スカラー (: setQueryString ";" setHiddenForm "¥n")
■ : スカラー (1 0)
◎ : スカラー HIDDEN

setQueryString setHiddenForm
(<input type="hidden" … />)

記 述 例

```
$query_string = stdio::setQueryString(¥%hash, ¥@array, $separator, $cut_novalue_key);
$hidden = stdio::setHiddenForm(¥%hash, ¥@array, $separator, $cut_novalue_key);
```

サンプル

```
#
%PARAM = (
    'keyword' => 'CGI         ',
    'max'     => '20'
);

#
$query_string = stdio::setQueryString(¥%PARAM);
print qq|<a href="search.cgi?$query_string">    </a>¥n|;

#
$hidden = stdio::setHiddenForm(¥%PARAM);
print $hidden;
```

▼ ①の実行結果

```
<a href="search.cgi?keyword=CGI+%a5%d7%a5%ed%a5%b0%a5%e9%a5%e0;max=20">    </a>
```

▼ ②の実行結果

```
<input type="hidden" name="keyword" value="CGI         " />
<input type="hidden" name="max" value="20" />
```

getTime

フォーマットを指定して時間を取得

◎ = **stdio::getTime**([☆[, ★[, □]]])

☆ : スカラー (gmtime)
★ : スカラー GMT(=) (3600*9)
□ : スカラー (=time)
◎ : スカラー

getTime 1 (
) 2 (=)
+9 3600*9 3 UTC( )
(time)

日付書式に使える記号と意味

% %
%%

記号	意味	例	記号	意味	例
yyyy	4	2002	ap	/	
yyy	( )	14	ap1	AM/PM	AM
yy	2 (1 2 0)	02	ap2	am/pm	am
y	1	2	hh	24 2 (1 2 0)	19
mm	2 (1 2 0)	12	h	24 1	19
m	1	12	HH	12 2 (1 2 0)	07
MM		Jan	H	12 1	7
MM2		January	nn	2 (1 2 0)	12
dd	2 (1 2 0)	05	n	1	12
d	1	5	ss	2 (1 2 0)	05
ww			s	1	5
ww2		Sun			
ww3		Sunday			

日付書式指定の例

```
%yyyy/%mm/%dd %hh:%nn:%ss      2002/03/13 12:34:21
   %yyy  %m  %d                   14  3  13
%ap%HH  %nn  %ss                  12  34  21
%ww3 %MM %d, %yyyy             Thursday Jun 6, 2002
%yyyy%mm%dd%hh%nn%cc           20020313123421
%H:%nn%ap2, %ww2 %MM d         3:52pm, Thu Jul 6
```

記 述 例

```
$time_now = stdio::getTime($time_format, $time_difference, $base_time);
```

サンプル

```
#
print stdio::getTime("%yyyy/%mm/%dd (%ww) %hh:%nn", 3600*9);

#    $filename                GMT
$atime = (stat($filename))[9];
print stdio::getTime("%ww2, %mm3 d, yyyy  HH:nn ap3", 0, $atime);
```

▼ ①の実行結果（以下の形式で現在時間を表示）

```
2002/06/07 (  ) 02:14
```

▼ ②の実行結果（以下の形式で$filename の UTC での最終アクセス時間を表示）

```
Friday, June 7, 2002 02:14 pm
```

▼ ③の実行結果（以下の形式で UTC での現在時間を表示）

```
20020810141039
```

Tips 正午は午前 12 時、それとも午後 12 時？

12 12 getTime
12 0 (5) 12
( ) 1
12 12
getTime 12 (12) 12
Windows Yahoo!
( ) / 0 12 0

getSerialTime

時間からUTCシリアル値を取得

◎ = **stdio::getSerialTime**(☆, ★[, □[, ■[, ○[, ●[, △]]]]])

☆	:	スカラー	GMT(=) (3600*9)
★	:	スカラー	(4)
□	:	スカラー	(1)
■	:	スカラー	(1)
○	:	スカラー	(0)
●	:	スカラー	(0)
△	:	スカラー	(0)
◎	:	スカラー	UTC

getSerialTime ( ) UTC (1970 1 1 0 0 0)

記 述 例

```
$utc_serial = stdio::getSerialTime($time_difference, $year, $mon, $day, $hour, $min, $sec);
```

サンプル

```
# 2020  10  31  14  15  32   UTC
$utc_serial = stdio::getSerialTime(9*3600, 2020, 10, 31, 14, 15, 32);
print "2020/10/31 14:15:32  UTC        : $serial_time¥n";
$time = scalar(localtime $serial_time);
print "                    : $time¥n";

# 1999  7  10     2002  6  7
$day1 = stdio::getSerialTime(9*3600, 1999, 7, 10);
$day2 = stdio::getSerialTime(9*3600, 2002, 6, 7);
$days = int(($day2 - $day1) / 86400);
print "1999/7/10    2002/6/7        : $days  ¥n";
```

▼ 実行結果

```
2020/10/31 14:15:32  UTC         : 1604121332
                    : Sat Oct 31 14:15:32 2020
1999/7/10    2002/6/7         : 1063
```

base64encode / base64decode

base64 エンコード・デコード

stdio::base64encode(☆[, ★]) , ◎ = **base64encode_**(■[, ★])

stdio::base64decode(□) , ◎ = **stdio::base64decode_**(○)

☆ : ｽｶﾗｰ(参照) Base64 ( )
★ : スカラー 1 ( )
□ : ｽｶﾗｰ(参照) Base64 ( )

※ 関数名が stdio::base64encode_() と stdio::base64decode_() の場合は値渡しになります
■ : スカラー Base64 ( )
○ : スカラー Base64 ( )
◎ : スカラー Base64 ( ) ( )

base64encode/base64decode Base64 /

Base64 エンコードとは？

Base64 (24)
1 (8) (2^6 64
52 10 + / 64
)

サンプル

```
$str = 'Base64                    ';
print "          : $str¥n";

# Base64
stdio::base64encode(¥$str);
print "base64encode : $str¥n";

# Base64       (    )
stdio::base64decode(¥$str);
print "base64decode : $str¥n";
```

▼ 実行結果

```
          : Base64
Base64encode : QmFzZTY0g0eDk4NSgVuDaIK3gumVto6al/GBQg==
Base64decode : Base64
```

urlencode / urldecode

URL エンコード・デコード

stdio::urlencode(☆) , ◎ = stdio::urlencode_(□)

stdio::urldecode(★) , ◎ = stdio::urldecode_(■)

☆ : ｽｶﾗｰ(参照) URL ( )
★ : ｽｶﾗｰ(参照) URL ( )

※ 関数名が stdio::urlencode_() と stdio::urldecode_() の場合は値渡しになります
□ : スカラー URL ( )
■ : スカラー URL ( )
◎ : スカラー URL ( ) ( )

urlencode/urldecode URL /

URL エンコードとは？

URL(URI) +
16 (00 ff) 16 %

サンプル

```
#                URL                          (                    )
foreach (split /&/, $ENV{'QUERY_STRING'}) {
    my($key, $val) = split /=/, $_, 2;
    stdio::urldecode(¥$val);
    $QUERY_STRING{$key} = $val;
}

#   $str   URL        (              JIS)
$str = 'URL              Test';
print "         : $str¥n";
stdio::urlencode(¥$str);
print "URL          : $str¥n";
```

▼ ②の実行結果

```
         : URL              Test
URL          : URL%83G%83%93%83R%81%5B%83h+%83e%83X%83g+Test
```

trString

文字列を置換する

stdio::trString(☆[, ★[, □[, ■[, ○[, ●]]]]])

◎ = **stdio::trString_**(△[, ★[, □[, ■[, ○[, ●]]]]])

☆ : ｽｶﾗｰ(参照) ( )
★ : スカラー 1 HTML 2 HTML
□ : スカラー 1 2
■ : スカラー 1 2
○ : スカラー 1 2
● : スカラー

※ 関数名が stdio::trString_() の場合は値渡しになります
△ : スカラー ( )
◎ : スカラー ( )

trString

4

jcode.pl

記 述 例

```
stdio::trString(¥$string, $conv_tags, $lc, $z2h, $k2h, $rmstr);
```

サンプル

```
#
$str = "                                          ";
stdio::trString(¥$str, 0, 1, 1);
print $str;

#
$str = "                          ";
stdio::trString(¥$str, 0, 0, 0, "     ");
print $str;
```

▼ ①の実行結果

```
abcdefghijklmnopqrstuvwxyz
```

▼ ②の実行結果

searchString

マルチ文字列検索

◎ = **stdio::searchString**(☆, ★[, □])

☆ : スカラー ( )
★ : スカラー ( )
□ : スカラー (AND)
◎ : スカラー 1 0

searchString AND OR BOOLEAN( )

キーワードの指定とワイルドカード

1 2 ( )
* 0
32 * **

◆例. *
...

検索条件の指定

1=AND

▼指定可能な検索条件

```
0 = AND
1 = OR
2 = BOOLEAN = BLN
```

■AND 検索と OR 検索の使い方

AND OR
("!") "-" NOT

■検索式の使い方

3 "BOOLEAN"

▼使用可能な検索演算子

```
AND     &
OR      |
NOT     !                    (                    )
(  )
```

AND　OR　NOT
AND NOT　　NOT

◆例 1.　Java AND NOT JavaScript　(　AND NOT　　NOT　　)
…Java　　JavaScript
◆例 2.　(　　or　　) and (　　or　　)
…　　(　　)　　(　　)
◆例 3.　(　　|　　) & (　!　　& !　　)
…　　(　　)　　(　　)

記 述 例

```
$result = stdio::searchString($string, $keyword, $match_mode);
```

サンプル

```
#
@record = (
    '1.          ',
    '2.             ',
    '3.       ',
    '4.                 ',
    '5.           ',
    '6.             ',
    '7.             '
);

#   "      "  "      "
foreach (@record) {
    if (stdio::searchString($_, "          ")) {
        print "$_¥n";
    }
}

#   "  "      "  "
foreach (@record) {
    if (stdio::searchString($_, "   -  ")) {
        print "$_¥n";
    }
}

#   "  "  "  "      "  "                (          )
foreach (@record) {
    if (stdio::searchString($_, "(    OR    ) AND NOT   ", "Boolean")) {
        print "$_¥n";
    }
}
```

```
#    "  "  "  "                    (              )
foreach (@record) {
   if (stdio::searchString($_, "NOT   AND NOT   ", "Boolean")) {
      print "$_¥n";
   }
}
```

▼ ①の実行結果

```
5.
```

▼ ②の実行結果

```
1.
2.
3.
5.
```

▼ ③の実行結果

```
1.
3.
5.
6.
```

▼ ④の実行結果

```
4.
7.
```

getRandomString

ランダムな文字列を作成・取得

◎ = **stdio::getRandomString**([☆[, ★]])

☆ : スカラー (8)
★ : スカラー (62)
◎ : スカラー

getRandomString

crypt

記 述 例

```
$random_string = stdio::getRandomString($byte, $string);
```

サンプル

```
#    $password   6
$password = stdio::getRandomString(6);

# crypt      $password
$crypted_password = crypt($password, stdio::getRandomString(2));

print "¥$password = $password (getRandomString                    )¥n";
print "¥$crypted_password = $crypted_password (crypt              )¥n";

# crypt
if (crypt($password, $crypted_password)) {
    print "                    ¥n";
}
```

▼ 実行結果（結果は毎回異なる）

```
$password = mN7ZsY (getRandomString                    )
$crypted_password = PR4VmYTEpEzCQ (crypt              )
```

setLink

クリッカブルURI(URL)を設定する

stdio::setLink(☆[, ★[, □[, ■]]])

◎ = **stdio::setLink_**(○[, ★[, □[, ■]]])

☆ : ｽｶﾗｰ(参照) URI ( )
★ : スカラー target ( )
□ : スカラー a (URI)
■ : スカラー a ( )

※ 関数名がstdio::setLink_()の場合は値渡しになります

○ : スカラー URI ( )
◎ : スカラー URI ( )

URI a URI
http https ftp telnet wais gopher news nntp rstp mailto URI ;
"mailto:"

記 述 例

```
stdio::setLink(¥$string, $target_attribute, $link_string, $mail_string);
```

サンプル

```
# $str      URI
$str = <<'_EOF_';
http://www.google.com/
mailto:address@hostname.com
ftp://www.webpower.jp/
<a href="http://www.yahoo.co.jp/">http://www.yahoo.co.jp/</a>
_EOF_

# $str      URI          (                        [E-MAIL])
stdio::setLink(¥$string, 'target="_top"', undef, '[E-MAIL]');
print $str;
```

▼ 実行結果

```
<a href="http://www.google.com/" target="_top">http://www.google.com/</a>
<a href="mailto:address@hostname.ne.jp">[E-MAIL]</a>
<a href="ftp://www.webpower.jp/" target="_top">ftp://www.webpower.jp/</a>
<a href="http://www.yahoo.co.jp/">http://www.yahoo.co.jp/</a>
```

setComma

3 桁毎にカンマを打つ

◎ = **stdio::setComma**(☆)

☆ : スカラー
◎ : スカラー

setComma 3

記 述 例

```
$digit = stdio::setComma($digit);
```

サンプル

```
# 3
print stdio::setComma(5629800), "¥n";
print stdio::setComma(9800), "¥n";
print stdio::setComma(345), "¥n";
print stdio::setComma("12A0F"), "¥n";  # 16
```

▼ 実行結果

```
5,629,800
9,800
345
12,A0F
```

Tips ヒアドキュメントの中で関数呼び出し

```
@{[stdio::setComma(1200)]}
```

scalar

```
@{[scalar locatime]}
```

lock / unlock / lockCheck

排他制御

◎ = **stdio::lock(☆[,★[,□]])**

◎ = **stdio::unlock(☆)**

◎ = **stdio::lockCheck(☆[,★[,□]])**

☆ ： スカラー　ロックフラグ用ディレクトリのパス
※ パスの指定が「///」で始まる場合は、「///」は$stdio::tmp_dir で指定した値に置換。
★ ： スカラー　ロック中の場合の再試行回数 (省略時は 3)
□ ： スカラー　次の試行までの待機秒数[整数] (省略時は 1)
◎ ： スカラー　1：非ロック中／0：ロック中

lock/unlock 関数は、ファイルのロック/アンロックをします。lockCheck 関数は、ロックされているかどうかをチェックします。(チェックするだけでロックはしません。) lock、unlock 関数を用いないファイル操作はロックの対象となりません。この関数では、ロックフラグにディレクトリを使っています。

記 述 例

```
$result = stdio::lockCheck($lock_dir);
$result = stdio::lock($lock_dir);
stdio::unlock($lock_dir);
```

サンプル

```
# ロック
if (!stdio::lock($lockdir)) {
    print "Busy!¥n";
    exit;
}

# ファイル入出力
open IO, "+<count.txt" || die "Can't open file.";
my($count) = <IO>;
$count ++;
seek IO, 0, 0;
print IO $count;

# アンロック
stdio::unlock($lockdir);
```

コ ラ ム

●ファイルロックが必要な理由

CGI

```
1 open IN, "data.txt";      #
2 @record = <IN>;           #
3 close IN;                 #
4 unshift @record, $record; #
5 open OUT, ">data.txt";    #                    (                    )
6 print OUT @record;        #                    (                )
7 close OUT;                #              (                          )
```

5 open data.txt 6

7

A 5 7 3

B 2 A 5

B 1 2

( ) stdio.pl

mkdir

( )

(=)

Windows OS

① ロックフラグにファイルを使う

```
if (!-f $lock_file) {
  open OUT, ">$lock_file" || die "Can't open lock file."
```

2

1

② ロックフラグにディレクトリを使う

```
if (mkdir $lock_dir) {
```

openSocket / closeSocket

ソケット確立・解放

◎ = **stdio::oepnSocket**(☆, ★[, □[, ■[, ○]]])

◎ = **stdio::closeSocket**(☆)

☆ : スカラー　ソケットと結びつけるファイルハンドル
★ : スカラー　(http URI 3)
□ : スカラー　(GET)
■ : ﾊｯｼｭ(参照)
○ : スカラー　(GET)

◎ : スカラー　1 0

openSocket/closeSocket / 2
http URI Web HTTP
3 http
SMTP FTP

ドメインとタイプ

(socket) stdio.pl $stdio::inet
$stdio::stream OS
Socket.pm PF_INET SOCK_STREAM

Web サーバーに HTTP 接続する

2 URI http (80)
Web CGI Web Web

```
openSocket(SOCK, "http://          :      (      80)      URI");
```

proxy.hostname.jp:80 http://www.google.co.jp/

```
openSocket(SOCK, "http://proxy.hostname.jp:80 http://www.google.co.jp");
```

リクエストメソッドとヘッダーを指定する

HTTP 3 4
"GET" 4
Referer( )

標準入力と Content-Type/Content-Length

HTTP POST 5
Content-Type application/x-www-form-urlencoded
Content-Length 4

4 Content-Type

記 述 例

```
$result = stdio::oepnSocket(SOCK, $uri, $method, ¥%header, $stdin);
$result = stdio::closeSocket(SOCK);
```

サンプル

```
#
%header = (
    "User-Agent"  => "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)",
    "Referer"     => "http://www.hostname.jp/dir/file.html"
);

#       (          )
$SIG{'ARM'} = ¥&TimeoutError;

#       (          )  30
alarm 30;

# URI(http://www.hostname.jp/)
if (stdio::openSocket(SOCK, "http://www.hostname.jp/", "GET", ¥%header)) {
    print while(<SOCK>);
    stdio::closeSocket(SOCK);
}

#
alarm 0;

#
sub TimeoutError
{
    print "                          ¥n";
    exit;
}
```

sendmail

sendmail を使用してメール送信

◎ = **stdio::sendmail**(☆, ★[, □[, ■[, ○]]])

☆	:	ﾊｯｼｭ(参照) ( )
★	:	スカラー
□	:	スカラー HTML (HTML)
■	:	スカラー (base64encode uuencode) base64
○	:	リスト ([; MIME [; [;]]])
◎	:	スカラー 1 0

sendmail sendmail
sendmail(sendmail)
jcode.pl
sendmail stdio.pl $stdio::sendmail
/usr/lib/sendmail

メールヘッダー

Base64
1

メール本文

2 3 2 3
HTML
2 2 3
HTML 3
HTML
HTML

ファイル添付

5
Base64encode Uuencode
BinHex Base64encode UUencode Base64encode
Base64encode

Tips 巨大ファイルの添付は御法度

(
1.5)
1M
P2P

ファイルを添付する際の詳細設定

5 MIME

```
ファイルパス [; MIME タイプ [; 名前 [; エンコード形式 ]]]  * [ ] は省略可能
```

MIME タイプ "image/gif" "audio/midi" MIME
"application/octet-stream"
名前
エンコード形式 "base64encode" "uuencode"

添付ファイルのキャッシュオプション

2 4 [
] cache

■どのような場合に有効？

sendmail
1.3MB 2
16 8 (
)

```
16 wallclock secs (15.59 usr 0.09 sys + 0.12 cusr 0.06 csys = 15.86 CPU)
9 wallclock secs ( 8.02 usr 0.03 sys + 0.06 cusr 0.10 csys = 8.21 CPU)
```

1

■キャッシュファイルはどこに作られる？

$stdio::tmp_dir base64 (+
/ - _) .b64(uu).cache

■キャッシュファイルを削除するには？

stdio::removeCacheFiles

```
stdio::removeCacheFiles(@files); # @files
```

■関数側でエンコードせずに添付するには？

cache encoded

記 述 例

```
$result = stdio::sendmail($sendmail, ¥%header, $body, $html_body, $encode, @attachments);
```

サンプル

```
#                                    (                              )
%header = (
    'To'       => 'address@hostname.co.jp',
    'Cc'       => 'cc_addr@hostname.co.jp',
    'From'     => '    <taro@hostname.co.jp>',
    'Subject'  => '          '
);

#
@attachments = (
    'imagefile.gif; image/gif',
    'zipfile.zip; application/zip; archive.zip; uuencode cache'
);

#          (       )
$body = <<'_EOF_';


_EOF_

#          (HTML)
$html_body = <<'_EOF_';
<html>
<head>
</head>
<body>
<p>               <br>
                          </p>
</body>
</html>
_EOF_

#
$result = stdio::sendmail(¥%header, $body, $html_body, "base64", @attachments);

#
if (!$result) {
    print "                      ¥n";
    exit;
}
```

shuffleArray

配列をシャッフルする

◎ = **stdio::shuffleArray**(☆)

☆ : リスト
◎ : リスト

shuffleArray ( )

記 述 例

```
@array = stdio::shuffleArray(@array);
```

サンプル

```
#
@array = ('A'..'Z','a'..'z','0'..'9');

print stdio::shuffleArray(@array);
```

▼ 実行結果(結果は毎回異なる)

```
3f5GAmba8ruBCHQvOFoEV4kPNJw76Yld1RcexDLqS2KjTtpZiIs9UWXhgzMyn0
```

getImageSize

GIF/JPEG/PING 画像のピクセルサイズを取得

(□, ■, △) = **stdio::getImageSize**(☆)

☆ : スカラー
□ : スカラー (GIF JPG PNG)
(0 GIF/JPEG/PNG)
■ : スカラー [width] (0)
△ : スカラー [height] ( )

getImageSize GIF JPEG PING
HTML img
width/height

記 述 例

```
($type, $width, $height) = stdio::getImageSize($file_name);
```

サンプル

```
# image.gif
($type, $width, $height) = stdio::getImageSize("image.gif");

#
if ($type eq "GIF" || $type eq "JPG" || $type eq "PNG") {
    print "    : $type¥n";
    print "    : $width¥n";
    print "    : $height¥n";
} else {
    print "                                    ¥n";
}

# img      width      height
if ($type eq "GIF" || $type eq "JPG" || $type eq "PNG") {
    print qq|<img src="image.gif" width="$width" height="$height" alt="    " />¥n|;
}
```

getMimeType

ファイル名から適切なMIMEタイプを取得する

◎ = **stdio::getMimeType**(☆)

☆ : スカラー

◎ : スカラー MIME (MIME " application/octet-stream")

getMimeType MIME
CGI Content-Type
MIME MIME application/octet-stream

記 述 例

```
$mime_type = stdio::getMimeType($file_extention);
```

サンプル

```
# $file   MIME
$mime_type = stdio::getMimeType($file);

# Content-Type         MIME
print "Content-Type: $mime_type¥n";
print "¥n";
open IN, $file || die "Can't open file.";
print while (read IN, $_, 1024);
close IN;
```

STDIO.PL QUICK REFERENCE I

STDIO.PL クイックリファレンス

setCookie	動作	HTTP
	引数	%@ [, ID [, [, [, [, ? [, ?]]]]]]
	戻値	(7)
getCookie	動作	HTTP
	引数	%@ ( )[, ID]
	戻値	( )( )
setSession	動作	(ID)
	引数	, %@ [, ID [, (3600)]]
	戻値	1: 0: ( )
getSession	動作	(setSession)
	引数	, %@ ( )[, ID]
	戻値	1: 0: (ID)
getFormData getUrlencodedFormData getMultipartFormData	動作	( ) getUrlencodeFormData URL getMultipartFormData
	引数	% [, ?HTML (2) [, (euc/sjis/jis/EUC/JIS/SJIS) [, ( ) [, ( )]]]]
	戻値	( )( )
setQueryString	意味	
	引数	% [, @ ( ) [, (";") [, ?]]]
	戻値	
setHiddenForm	動作	
	引数	% [, @ ( ) [, ("¥n") [, ?]]]
	戻値	
getTime	動作	
	引数	[[, GMT (3600*9) [, (UTC)]]]
	戻値	
getSerialTime	動作	UTC
	引数	GMT (3600*9), (4) [, [, [, [, [,]]]]]
	戻値	UTC
base64encode base64decode	動作	Base64
	引数	( ) [, 1 ( )]
	戻値	(( ))
urlencode urldecode	動作	URL
	引数	( )
	戻値	(( ))

( )% @ ? ( ) []
( )

STDIO.PL QUICK REFERENCE I

STDIO.PL クイックリファレンス

setCookie	**動作**	HTTP クッキーを作成する
	引数	*%@クッキー本体* [, クッキーID [, 有効期限 [, 有効パス [, 有効ドメイン [, ?セキュア時のみクッキー発行 [, ?クッキー本体を戻り値に返す]]]]]
	戻値	クッキーの値(第 7 引数にクッキー本体を戻り値に返すようにした場合)
getCookie	**動作**	ブラウザから HTTP クッキーを取得する
	引数	*%@クッキー本体を格納するハッシュ(配列)*[, クッキーID]
	戻値	取得したクッキーのキーのリスト(処理順)(取得できなかった場合は無し)
setSession	**動作**	セッション管理(ハッシュに有効期限と ID を付けて保存)
	引数	ファイルのパス, *%@セッション情報* [, セッション ID [, 有効秒数(省略時は 3600 秒)]]
	戻値	1:成功 0:失敗(ファイル読み書き不可)
getSession	**動作**	セッション管理(setSession 関数で保存したセッション情報を読み出す)
	引数	ファイルのパス, *%@セッション情報を格納するハッシュ(配列)*[, セッション ID]
	戻値	1:成功 0:失敗(ファイル読み書き不可かセッション ID が存在しない)
getFormData **getUrlencodedFormData** **getMultipartFormData**	**動作**	フォームデータ(クエリー文字列か標準入力)を取得する。getUrlencodeFormData は URL エンコードフォームデータのみ、getMultipartFormData はマルチパートフォームデータのみを取得する
	引数	*%フォームデータを格納するハッシュ* [, ?HTML 構成文字を変換(※2 を指定した場合改行は にする) [, 変換文字コード(euc/sjis/jis/EUC/JIS/SJIS ※大文字の場合は半角カタカナを全角置換) [, 重複キーのセパレータ(未定義の場合は上書き) [, 添付ファイルを格納するディレクトリのパス(省略時はハッシュに格納)]]]]
	戻値	取得したフォームデータのキーのリスト(処理順)(取得できなかった場合は無し)
setQueryString	**意味**	ハッシュからクエリー文字列を作成する
	引数	*%ハッシュ* [, *@配列(順番を指定する場合)* [, セパレータ(省略時は";") [, ?値が空の場合はキー情報を省略する]]]
	戻値	作成された文字列
setHiddenForm	**動作**	ハッシュからフォームの隠しフィールドを作成する
	引数	*%ハッシュ* [, *@配列(順番を指定する場合)* [, セパレータ(省略時は"¥n") [, ?値が空の場合はキー情報を省略する]]]
	戻値	作成された文字列
getTime	**動作**	書式を指定して時間を取得する
	引数	[書式 [, GMT との秒差(日本は 3600*9) [, 基準時間(UTC シリアル値。省略時は現在時間)]]]
	戻値	整形済み時間
getSerialTime	**動作**	時間から UTC シリアル値を取得する
	引数	GMT との秒差(日本は 3600*9), 年(西暦 4 桁) [, 月 [, 日 [, 時 [, 分 [, 秒]]]]
	戻値	UTC シリアル値
base64encode **base64decode**	**動作**	引数で指定したデータを Base64 エンコード・デコードする
	引数	*エンコード(デコード)するデータ* [, 1 行あたりのバイト数(エンコード時のみ)]
	戻値	なし (値渡しの場合はエンコード(デコード)されたデータ)
urlencode **urldecode**	**動作**	引数で指定したデータを URL エンコード・デコードする
	引数	*エンコード(デコード)するデータ*
	戻値	なし (値渡しの場合はエンコード(デコード)されたデータ)

(表記の記号の意味) %：ハッシュ／@：リスト／?：真か偽で指定／なし：スカラー変数／*イタリック*は参照(リファレンス) ／[ ]は省略可能

(引数の色分け) 黒：第 1 引数／赤：第 2 引数／青：第 3 引数／緑：第 4 引数／紫：第 5 引数／茶：第 6 引数／紺：第 7 引数

※ 各関数の詳細は詳細リファレンスを参照してください。