iconv

Introduction

This module contains an interface to iconv character set conversion facility. With this module, you can turn a string represented by a local character set into the one represented by another character set, which may be the Unicode character set. Supported character sets depend on the iconv implementation of your system. Note that the iconv function on some systems may not work as you expect. In such case, it'd be a good idea to install the » GNU libiconv library. It will most likely end up with more consistent results.

Since PHP 5.0.0, this extension comes with various utility functions that help you to write multilingual scripts. Let's have a look at the following sections to explore the new features.

Installing/Configuring

Requirements

You will need nothing if the system you are using is one of the recent POSIX-compliant systems because standard C libraries that are supplied in them must provide iconv facility. Otherwise, you have to get the » libiconv library installed in your system.

Installation

To use functions provided by this module, the PHP binary must be built with the following configure line: *--with-iconv[=DIR]*.

Note
Note to Windows® Users In order to enable this module on a Windows® environment, you need to put a DLL file named *iconv.dll* or *iconv-1.3.dll* (prior to 4.2.1) which is bundled with the PHP/Win32 binary package into a directory specified by the *PATH* environment variable or one of the system directories of your Windows® installation. This module is part of PHP as of PHP 5 thus *iconv.dll* and *php_iconv.dll* is not needed anymore.

Runtime Configuration

The behaviour of these functions is affected by settings in *php.ini*.

Iconv configuration options

Name	Default	Changeable	Changelog
iconv.input_encoding	"ISO-8859-1"	PHP_INI_ALL	Available since PHP 4.0.5.
iconv.output_encoding	"ISO-8859-1"	PHP_INI_ALL	Available since PHP 4.0.5.
iconv.internal_encoding	"ISO-8859-1"	PHP_INI_ALL	Available since PHP 4.0.5.

For further details and definitions of the PHP_INI_* constants, see the php.ini directives.

Warning
Some systems (like IBM AIX) use "ISO8859-1" instead of "ISO-8859-1" so this value has to be used in configuration options and function parameters.

Note
Configuration option *`iconv.input_encoding`* is currently not used for anything.

Resource Types

This extension has no resource types defined.

Predefined Constants

Since PHP 4.3.0 it is possible to identify at runtime which *iconv* implementation is adopted by this extension.

iconv **constants**

Name	Type	Description
ICONV_IMPL	string	The implementation name
ICONV_VERSION	string	The implementation version

Note
Writing implementation-dependent scripts with these constants is strongly discouraged.

Since PHP 5.0.0, the following constants are also available:

iconv **constants available since PHP 5.0.0**

Name	Type	Description
ICONV_MIME_DECODE_STRICT	integer	A bitmask used for iconv_mime_decode()
ICONV_MIME_DECODE_CONTINUE_ON_ERROR	integer	A bitmask used for iconv_mime_decode()

iconv Functions

See Also

See also GNU Recode functions.

iconv_get_encoding

iconv_get_encoding -- Retrieve internal configuration variables of iconv extension

Description

mixed **iconv_get_encoding** ([string `$type`])

Retrieve internal configuration variables of iconv extension.

Parameters

`type`
The value of the optional *`type`* can be:

- all
- input_encoding
- output_encoding
- internal_encoding

Return Values

Returns the current value of the internal configuration variable if successful, or **FALSE** on failure.

If *`type`* is omitted or set to "all", iconv_get_encoding() returns an array that stores all these variables.

Examples

Example #1 - iconv_get_encoding() example

```
<pre>
<?php
iconv_set_encoding("internal_encoding", "UTF-8");
iconv_set_encoding("output_encoding", "ISO-8859-1");
var_dump(iconv_get_encoding('all'));
?>
</pre>
```

The above example will output:

```
Array
(
   [input_encoding] => ISO-8859-1
```

```
    [output_encoding] => ISO-8859-1
    [internal_encoding] => UTF-8
)
```

See Also

- iconv_set_encoding()
- ob_iconv_handler()

iconv_mime_decode_headers

iconv_mime_decode_headers -- Decodes multiple *MIME* header fields at once

Description

array **iconv_mime_decode_headers** (string `$encoded_headers` [, int `$mode` [, string `$charset`]])

Returns an associative array that holds a whole set of *MIME* header fields specified by *`encoded_headers`* on success, or **FALSE** if an error occurs during the decoding.

Each key of the return value represents an individual field name and the corresponding element represents a field value. If more than one field of the same name are present, iconv_mime_decode_headers() automatically incorporates them into a numerically indexed array in the order of occurrence.

Parameters

`encoded_headers`
The encoded headers, as a string.

`mode`
`mode` determines the behaviour in the event iconv_mime_decode_headers() encounters a malformed *MIME* header field. You can specify any combination of the following bitmasks.

Bitmasks acceptable to iconv_mime_decode_headers()

Value	Constant	Description
1	ICONV_MIME_DECODE_STRICT	If set, the given header is decoded in full conformance with the standards defined in » RFC2047. This option is disabled by default because there are a lot of broken mail user agents that don't follow the specification and don't produce correct *MIME* headers.
2	ICONV_MIME_DECODE_CONTINUE_ON_ERROR	If set, iconv_mime_decode_headers() attempts to ignore any grammatical errors and continue to process a given header.

charset
: The optional *charset* parameter specifies the character set to represent the result by. If omitted, iconv.internal_encoding will be used.

Examples

Example #2 - iconv_mime_decode_headers() example

```
<?php
$headers_string = <<<EOF
Subject: =?UTF-8?B?UHLDvGZ1bmcgUHLDvGZ1bmc=?=
To: example@example.com
Date: Thu, 1 Jan 1970 00:00:00 +0000
Message-Id: <example@example.com>
Received: from localhost (localhost [127.0.0.1]) by localhost
   with SMTP id example for <example@example.com>;
   Thu, 1 Jan 1970 00:00:00 +0000 (UTC)
   (envelope-from example-return-0000-example=example.com@example.com)
Received: (qmail 0 invoked by uid 65534); 1 Thu 2003 00:00:00 +0000

EOF;

$headers =  iconv_mime_decode_headers($headers_string, 0, "ISO-8859-1");
print_r($headers);
?>
```

The above example will output:

```
Array
(
   [Subject] => Prüfung Prüfung
   [To] => example@example.com
   [Date] => Thu, 1 Jan 1970 00:00:00 +0000
   [Message-Id] => <example@example.com>
   [Received] => Array
       (
           [0] => from localhost (localhost [127.0.0.1]) by localhost with SMTP
id example for <example@example.com>; Thu, 1 Jan 1970 00:00:00 +0000 (UTC)
(envelope-from example-return-0000-example=example.com@example.com)
           [1] => (qmail 0 invoked by uid 65534); 1 Thu 2003 00:00:00 +0000
       )

)
```

See Also

- iconv_mime_decode()
- mb_decode_mimeheader()
- imap_mime_header_decode()

- imap_base64()
- imap_qprint()

iconv_mime_decode

iconv_mime_decode -- Decodes a *MIME* header field

Description

string **iconv_mime_decode** (string `$encoded_header` [, int `$mode` [, string `$charset`]])

Decodes a *MIME* header field.

Parameters

`encoded_header`
The encoded header, as a string.

`mode`
`mode` determines the behaviour in the event iconv_mime_decode() encounters a malformed *MIME* header field. You can specify any combination of the following bitmasks.

Bitmasks acceptable to iconv_mime_decode()

Value	Constant	Description
1	ICONV_MIME_DECODE_STRICT	If set, the given header is decoded in full conformance with the standards defined in » RFC2047. This option is disabled by default because there are a lot of broken mail user agents that don't follow the specification and don't produce correct *MIME* headers.
2	ICONV_MIME_DECODE_CONTINUE_ON_ERROR	If set, iconv_mime_decode_headers() attempts to ignore any grammatical errors and continue to process a given header.

`charset`
The optional *`charset`* parameter specifies the character set to represent the result by. If omitted, iconv.internal_encoding will be used.

Return Values

Returns a decoded *MIME* field on success, or **FALSE** if an error occurs during the decoding.

Examples

Example #3 - iconv_mime_decode() example

```
<?php
// This yields "Subject: Prüfung Prüfung"
echo iconv_mime_decode("Subject: =?UTF-8?B?UHLDvGZ1bmcgUHLDvGZ1bmc=?=",
                       0, "ISO-8859-1");
?>
```

See Also

- iconv_mime_decode_headers()
- mb_decode_mimeheader()
- imap_mime_header_decode()
- imap_base64()
- imap_qprint()

iconv_mime_encode

iconv_mime_encode -- Composes a *MIME* header field

Description

string **iconv_mime_encode** (string `$field_name`, string `$field_value` [, array `$preferences`])

Composes and returns a string that represents a valid *MIME* header field, which looks like the following:
`Subject: =?ISO-8859-1?Q?Pr=FCfung_f=FCr?= Entwerfen von einer MIME kopfzeile`
In the above example, "Subject" is the field name and the portion that begins with "=?ISO-8859-1?..." is the field value.

Parameters

`field_name`
The field name.

`field_value`
The field value.

`preferences`
You can control the behaviour of iconv_mime_encode() by specifying an associative array that contains configuration items to the optional third parameter *`preferences`*. The items supported by iconv_mime_encode() are listed below. Note that item names are treated case-sensitive.

Configuration items supported by iconv_mime_encode()

Item	Type	Description	Default value	Example
scheme	string	Specifies the method to encode a field value by. The value of this item may be either "B" or "Q", where "B" stands for *base64* encoding scheme and "Q" stands for *quoted-printable* encoding scheme.	B	B

input-charset	string	Specifies the character set in which the first parameter *field_name* and the second parameter *field_value* are presented. If not given, iconv_mime_encode() assumes those parameters are presented to it in the iconv.internal_encoding ini setting.	iconv.internal_encoding	ISO-8859-1
output-charset	string	Specifies the character set to use to compose the *MIME* header. If not given, the same value as *input-charset* will be used.	iconv.internal_encoding	UTF-8
line-length	integer	Specifies the maximum length of the header lines. The resulting header is "folded" to a set of multiple lines in case the resulting header field would be longer than the value of this parameter, according to » RFC2822 - Internet Message Format. If not given, the length will be limited to 76 characters.	76	996
line-break-chars	string	Specifies the sequence of	\r\n	\n

		characters to append to each line as an end-of-line sign when "folding" is performed on a long header field. If not given, this defaults to "\r\n" (*CR LF*). Note that this parameter is always treated as an ASCII string regardless of the value of *input-charset*.		

Return Values

Returns an encoded *MIME* field on success, or **FALSE** if an error occurs during the encoding.

Examples

Example #4 - iconv_mime_encode() **example**

```
<?php
$preferences = array(
   "input-charset" => "ISO-8859-1",
   "output-charset" => "UTF-8",
   "line-length" => 76,
   "line-break-chars" => "\n"
);
$preferences["scheme"] = "Q";
// This yields "Subject: =?UTF-8?Q?Pr=C3=BCfung_Pr=C3=BCfung?="
echo iconv_mime_encode("Subject", "Prüfung Prüfung", $preferences);

$preferences["scheme"] = "B";
// This yields "Subject: =?UTF-8?B?UHLDvGZ1bmcgUHLDvGZ1bmc=?="
echo iconv_mime_encode("Subject", "Prüfung Prüfung", $preferences);
?>
```

See Also

- imap_binary()
- mb_encode_mimeheader()
- imap_8bit()

iconv_set_encoding

iconv_set_encoding -- Set current setting for character encoding conversion

Description

bool **iconv_set_encoding** (string `$type`, string `$charset`)

Changes the value of the internal configuration variable specified by *`type`* to *`charset`*.

Parameters

`type`
: The value of *`type`* can be any one of those:

- input_encoding
- output_encoding
- internal_encoding

`charset`
: The character set.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #5 - iconv_set_encoding() example

```
<?php
iconv_set_encoding("internal_encoding", "UTF-8");
iconv_set_encoding("output_encoding", "ISO-8859-1");
?>
```

See Also

- iconv_get_encoding()
- ob_iconv_handler()

iconv_strlen

iconv_strlen -- Returns the character count of string

Description

int **iconv_strlen** (string `$str` [, string `$charset`])

In contrast to strlen(), iconv_strlen() counts the occurrences of characters in the given byte sequence *`str`* on the basis of the specified character set, the result of which is not necessarily identical to the length of the string in byte.

Parameters

`str`
The string.

`charset`
If *`charset`* parameter is omitted, *`str`* is assumed to be encoded in iconv.internal_encoding
.

Return Values

Returns the character count of *`str`*, as an integer.

See Also

- strlen()
- mb_strlen()

iconv_strpos

iconv_strpos -- Finds position of first occurrence of a needle within a haystack

Description

int **iconv_strpos** (string `$haystack`, string `$needle` [, int `$offset` [, string `$charset`]])

Finds position of first occurrence of a needle within a haystack.

In contrast to strpos(), the return value of iconv_strpos() is the number of characters that appear before the needle, rather than the offset in bytes to the position where the needle has been found. The characters are counted on the basis of the specified character set *`charset`*.

Parameters

`haystack`
: The entire string.

`needle`
: The searched substring.

`offset`
: The optional *`offset`* parameter specifies the position from which the search should be performed.

`charset`
: If *`charset`* parameter is omitted, *`string`* are assumed to be encoded in iconv.internal_encoding.

If *`haystack`* or *`needle`* is not a string, it is converted to a string and applied as the ordinal value of a character.

Return Values

Returns the numeric position of the first occurrence of *`needle`* in *`haystack`*.

If *`needle`* is not found, iconv_strpos() will return **FALSE**.

Warning
This function may return Boolean **FALSE**, but may also return a non-Boolean value which evaluates to **FALSE**, such as *0* or "". Please read the section on Booleans for more information. Use the === operator for testing the return value of this function.

See Also

- strpos()
- iconv_strrpos()
- mb_strpos()

iconv_strrpos

iconv_strrpos -- Finds the last occurrence of a needle within a haystack

Description

int **iconv_strrpos** (string `$haystack`, string `$needle` [, string `$charset`])

In contrast to strpos(), the return value of iconv_strrpos() is the number of characters that appear before the needle, rather than the offset in bytes to the position where the needle has been found.

Parameters

`haystack`
The entire string.

`needle`
The searched substring.

`charset`
If *`charset`* parameter is omitted, *`string`* are assumed to be encoded in iconv.internal_encoding.

If *`haystack`* or *`needle`* is not a string, it is converted to a string and applied as the ordinal value of a character.

Return Values

Returns the numeric position of the last occurrence of *`needle`* in *`haystack`*. The characters are counted on the basis of the specified character set *`charset`*.

If *`needle`* is not found, iconv_strrpos() will return **FALSE**.

Warning
This function may return Boolean **FALSE**, but may also return a non-Boolean value which evaluates to **FALSE**, such as *0* or "". Please read the section on Booleans for more information. Use the === operator for testing the return value of this function.

See Also

- strrpos()
- iconv_strpos()
- mb_strrpos()

iconv_substr

iconv_substr -- Cut out part of a string

Description

string **iconv_substr** (string `$str`, int `$offset`)

string **iconv_substr** (string `$str`, int `$offset`, int `$length`, string `$charset`)

Cuts a portion of *`str`* specified by the *`offset`* and *`length`* parameters.

Parameters

`str`
The original string.

`offset`
If *`offset`* is non-negative, iconv_substr() cuts the portion out of *`str`* beginning at *`offset`* 'th character, counting from zero. If *`offset`* is negative, iconv_substr() cuts out the portion beginning at the position, *`offset`* characters away from the end of *`str`*.

`length`
If *`length`* is given and is positive, the return value will contain at most *`length`* characters of the portion that begins at *`offset`* (depending on the length of *`string`*). If negative *`length`* is passed, iconv_substr() cuts the portion out of *`str`* from the *`offset`* 'th character up to the character that is *`length`* characters away from the end of the string. In case *`offset`* is also negative, the start position is calculated beforehand according to the rule explained above.

`charset`
If *`charset`* parameter is omitted, *`string`* are assumed to be encoded in iconv.internal_encoding. Note that *`offset`* and *`length`* parameters are always deemed to represent offsets that are calculated on the basis of the character set determined by *`charset`*, whilst the counterpart substr() always takes these for byte offsets.

Return Values

Returns the portion of *`str`* specified by the *`offset`* and *`length`* parameters.

If *`str`* is shorter than *`offset`* characters long, **FALSE** will be returned.

See Also

- substr()

- mb_substr()
- mb_strcut()

iconv

iconv -- Convert string to requested character encoding

Description

string **iconv** (string `$in_charset`, string `$out_charset`, string `$str`)

Performs a character set conversion on the string *`str`* from *`in_charset`* to *`out_charset`*.

Parameters

`in_charset`
: The input charset.

`out_charset`
: The output charset. If you append the string *//TRANSLIT* to *`out_charset`* transliteration is activated. This means that when a character can't be represented in the target charset, it can be approximated through one or several similarly looking characters. If you append the string *//IGNORE*, characters that cannot be represented in the target charset are silently discarded. Otherwise, *`str`* is cut from the first illegal character.

`str`
: The string to be converted.

Return Values

Returns the converted string or **FALSE** on failure.

Examples

Example #6 - iconv() example

```
<?php
echo iconv("ISO-8859-1", "UTF-8", "This is a test.");
?>
```

ob_iconv_handler

ob_iconv_handler -- Convert character encoding as output buffer handler

Description

string **ob_iconv_handler** (string `$contents`, int `$status`)

Converts the string encoded in *`internal_encoding`* to *`output_encoding`*.

`internal_encoding` and *`output_encoding`* should be defined in the *php.ini* file or in iconv_set_encoding().

Parameters

See ob_start() for information about this handler parameters.

Return Values

See ob_start() for information about this handler return values.

Examples

Example #7 - ob_iconv_handler() example:

```
<?php
iconv_set_encoding("internal_encoding", "UTF-8");
iconv_set_encoding("output_encoding", "ISO-8859-1");
ob_start("ob_iconv_handler"); // start output buffering
?>
```

See Also

- iconv_get_encoding()
- iconv_set_encoding()
- output-control functions