\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* \*

\* IDNA Convert (idna\_convert.class.php) \*

\* \*

\* http://idnaconv.phlymail.de mailto:phlymail@phlylabs.de \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* (c) 2004-2007 phlyLabs, Berlin \*

\* This file is encoded in UTF-8 \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Introduction

------------

The class idna\_convert allows to convert internationalized domain names

(see RFC 3490, 3491, 3492 and 3454 for detials) as they can be used with various

registries worldwide to be translated between their original (localized) form

and their encoded form as it will be used in the DNS (Domain Name System).

The class provides two public methods, encode() and decode(), which do exactly

what you would expect them to do. You are allowed to use complete domain names,

simple strings and complete email addresses as well. That means, that you might

use any of the following notations:

- www.nörgler.com

- xn--nrgler-wxa

- xn--brse-5qa.xn--knrz-1ra.info

Errors, incorrectly encoded or invalid strings will lead to either a FALSE

response (when in strict mode) or to only partially converted strings.

You can query the occured error by calling the method get\_last\_error().

Unicode strings are expected to be either UTF-8 strings, UCS-4 strings or UCS-4

arrays. The default format is UTF-8. For setting different encodings, you can

call the method setParams() - please see the inline documentation for details.

ACE strings (the Punycode form) are always 7bit ASCII strings.

ATTENTION: We no longer supply the PHP5 version of the class. It is not

necessary for achieving a successfull conversion, since the supplied PHP code is

compatible with both PHP4 and PHP5. We expect to see no compatibility issues

with the upcoming PHP6, too.

Files

-----

idna\_convert.class.php - The actual class

idna\_convert.create.npdata.php - Useful for (re)creating the NPData file

npdata.ser - Serialized data for NamePrep

example.php - An example web page for converting

ReadMe.txt - This file

LICENCE - The LGPL licence file

The class is contained in idna\_convert.class.php.

MAKE SURE to copy the npdata.ser file into the same folder as the class file

itself!

Examples

--------

1. Say we wish to encode the domain name nörgler.com:

// Include the class

include\_once('idna\_convert.class.php');

// Instantiate it \*

$IDN = new idna\_convert();

// The input string, if input is not UTF-8 or UCS-4, it must be converted before

$input = utf8\_encode('nörgler.com');

// Encode it to its punycode presentation

$output = $IDN->encode($input);

// Output, what we got now

echo $output; // This will read: xn--nrgler-wxa.com

2. We received an email from a punycoded domain and are willing to learn, how

the domain name reads originally

// Include the class

include\_once('idna\_convert.class.php');

// Instantiate it (depending on the version you are using) with

$IDN = new idna\_convert();

// The input string

$input = 'andre@xn--brse-5qa.xn--knrz-1ra.info';

// Encode it to its punycode presentation

$output = $IDN->decode($input);

// Output, what we got now, if output should be in a format different to UTF-8

// or UCS-4, you will have to convert it before outputting it

echo utf8\_decode($output); // This will read: andre@börse.knörz.info

3. The input is read from a UCS-4 coded file and encoded line by line. By

appending the optional second parameter we tell enode() about the input

format to be used

// Include the class

include\_once('idna\_convert.class.php');

// Instantiate it

$IDN = new dinca\_convert();

// Iterate through the input file line by line

foreach (file('ucs4-domains.txt') as $line) {

echo $IDN->encode(trim($line), 'ucs4\_string');

echo "\n";

}

NPData

------

Should you need to recreate the npdata.ser file, which holds all necessary translation

tables in a serialized format, you can run the file idna\_convert.create.npdata.php, which

creates the file for you and stores it in the same folder, where it is placed.

Should you need to do changes to the tables you can do so, but beware of the consequences.

Contact us

----------

In case of errors, bugs, questions, wishes, please don't hesitate to contact us

under the email address above.

The team of phlyLabs

http://phlylabs.de

mailto:phlymail@phlylabs.de