

COLLABORATORS			
	TITLE : GNU Libidn API Refere	ence Manual	
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY		May 11, 2018	

REVISION HISTORY			
DATE	DESCRIPTION	NAME	

## **Contents**

1	GNU	U Libidn API Reference Manual	1
	1.1	idna	2
	1.2	stringprep	14
	1.3	punycode	24
	1.4	pr29	30
	1.5	tld	35
	1.6	idn-free	35
2	Inde	oy.	36

# **List of Figures**

1.1	Components of Libidn	 2

## **Chapter 1**

## **GNU Libidn API Reference Manual**

GNU Libidn is a fully documented implementation of the Stringprep, Punycode and IDNA specifications. Libidn's purpose is to encode and decode internationalized domain name strings. There are native C, C# and Java libraries.

The C library contains a generic Stringprep implementation. Profiles for Nameprep, iSCSI, SASL, XMPP and Kerberos V5 are included. Punycode and ASCII Compatible Encoding (ACE) via IDNA are supported. A mechanism to define Top-Level Domain (TLD) specific validation tables, and to compare strings against those tables, is included. Default tables for some TLDs are also included.

The Stringprep API consists of two main functions, one for converting data from the system's native representation into UTF-8, and one function to perform the Stringprep processing. Adding a new Stringprep profile for your application within the API is straightforward. The Punycode API consists of one encoding function and one decoding function. The IDNA API consists of the ToASCII and ToUnicode functions, as well as an high-level interface for converting entire domain names to and from the ACE encoded form. The TLD API consists of one set of functions to extract the TLD name from a domain string, one set of functions to locate the proper TLD table to use based on the TLD name, and core functions to validate a string against a TLD table, and some utility wrappers to perform all the steps in one call.

The library is used by, e.g., GNU SASL and Shishi to process user names and passwords. Libidn can be built into GNU Libc to enable a new system-wide getaddrinfo flag for IDN processing.

Libidn is developed for the GNU/Linux system, but runs on over 20 Unix platforms (including Solaris, IRIX, AIX, and Tru64) and Windows. The library is written in C and (parts of) the API is also accessible from C++, Emacs Lisp, Python and Java. A native Java and C# port is included.

Also included is a command line tool, several self tests, code examples, and more.

The internal layout of the library, and how your application interact with the various parts of the library, are shown in Figure 1.1.

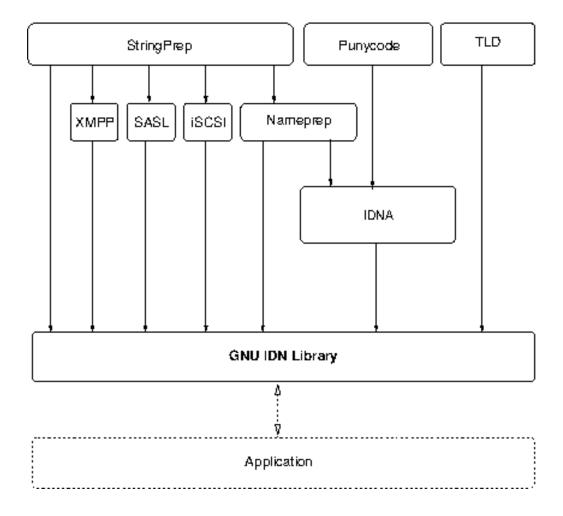


Figure 1.1: Components of Libidn

## 1.1 idna

idna —

## **Functions**

const char *	idna_strerror ()
int	idna_to_ascii_4i ()
int	idna_to_unicode_44i ()
int	idna_to_ascii_4z ()
int	idna_to_ascii_8z ()
int	idna_to_ascii_lz ()
int	idna_to_unicode_4z4z ()
int int	idna_to_unicode_8z4z ()
int	idna_to_unicode_8z8z ()
int	idna_to_unicode_8zlz ()
int	idna_to_unicode_lzlz ()

## **Types and Values**

#define	IDNAPI
enum	Idna_rc
enum	Idna_flags
#define	IDNA ACE PREFIX

## **Description**

#### **Functions**

#### idna strerror ()

```
const char~*
idna_strerror (Idna_rc rc);
```

Convert a return code integer to a text string. This string can be used to output a diagnostic message to the user.

IDNA\_SUCCESS: Successful operation. This value is guaranteed to always be zero, the remaining ones are only guaranteed to hold non-zero values, for logical comparison purposes. IDNA\_STRINGPREP\_ERROR: Error during string preparation. IDNA\_PUNYCODE\_ERROR: Error during punycode operation. IDNA\_CONTAINS\_NON\_LDH: For IDNA\_USE\_STD3\_ASCII\_RU indicate that the string contains non-LDH ASCII characters. IDNA\_CONTAINS\_MINUS: For IDNA\_USE\_STD3\_ASCII\_RULES, indicate that the string contains a leading or trailing hyphen-minus (U+002D). IDNA\_INVALID\_LENGTH: The final output string is not within the (inclusive) range 1 to 63 characters. IDNA\_NO\_ACE\_PREFIX: The string does not contain the ACE prefix (for ToUnicode). IDNA\_ROUNDTRIP\_VERIFY\_ERROR: The ToASCII operation on output string does not equal the input. IDNA\_CONTAINS\_ACE\_PREFIX: The input contains the ACE prefix (for ToASCII). IDNA\_ICONV\_ERROR: Could not convert string in locale encoding. IDNA\_MALLOC\_ERROR: Could not allocate buffer (this is typically a fatal error). IDNA\_DLOPEN\_ERROR: Could not dlopen the libcidn DSO (only used internally in libc).

#### **Parameters**

```
rc an Idna_rc return code.
```

#### Returns

Returns a pointer to a statically allocated string containing a description of the error with the return code rc.

## idna\_to\_ascii\_4i ()

The ToASCII operation takes a sequence of Unicode code points that make up one domain label and transforms it into a sequence of code points in the ASCII range (0..7F). If ToASCII succeeds, the original sequence and the resulting sequence are equivalent labels.

It is important to note that the ToASCII operation can fail. ToASCII fails if any step of it fails. If any step of the ToASCII operation fails on any label in a domain name, that domain name MUST NOT be used as an internationalized domain name. The method for deadling with this failure is application-specific.

The inputs to ToASCII are a sequence of code points, the AllowUnassigned flag, and the UseSTD3ASCIIRules flag. The output of ToASCII is either a sequence of ASCII code points or a failure condition.

ToASCII never alters a sequence of code points that are all in the ASCII range to begin with (although it could fail). Applying the ToASCII operation multiple times has exactly the same effect as applying it just once.

#### **Parameters**

in	input array with unicode	
III	code points.	
inlen	length of input array with	
men	unicode code points.	
	output zero terminated	_
out	string that must have room	
out	for at least 63 characters	
	plus the terminating zero.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGNED	
nags	or	
	IDNA_USE_STD3_ASCII_RULES.	

#### **Returns**

Returns 0 on success, or an Idna\_rc error code.

#### idna\_to\_unicode\_44i ()

The ToUnicode operation takes a sequence of Unicode code points that make up one domain label and returns a sequence of Unicode code points. If the input sequence is a label in ACE form, then the result is an equivalent internationalized label that is not in ACE form, otherwise the original sequence is returned unaltered.

ToUnicode never fails. If any step fails, then the original input sequence is returned immediately in that step.

The Punycode decoder can never output more code points than it inputs, but Nameprep can, and therefore ToUnicode can. Note that the number of octets needed to represent a sequence of code points depends on the particular character encoding used.

The inputs to ToUnicode are a sequence of code points, the AllowUnassigned flag, and the UseSTD3ASCIIRules flag. The output of ToUnicode is always a sequence of Unicode code points.

in	input array with unicode
III	code points.
inlen	length of input array with
inien	unicode code points.
out	output array with unicode
out	code points.
	on input, maximum size of
	output array with unicode
outlen	code points, on exit, actual
	size of output array with
	unicode code points.
	an Idna_flags value, e.g.,
flags	IDNA_ALLOW_UNASSIGNED
nags	or
	IDNA_USE_STD3_ASCII_RULES.

Returns Idna\_rc error condition, but it must only be used for debugging purposes. The output buffer is always guaranteed to contain the correct data according to the specification (sans malloc induced errors). NB! This means that you normally ignore the return code from this function, as checking it means breaking the standard.

#### idna\_to\_ascii\_4z()

Convert UCS-4 domain name to ASCII string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

#### **Parameters**

innut	zero terminated input	
input	Unicode string.	
output	pointer to newly allocated	
Output	output string.	
	an Idna_flags value, e.g.,	
flogs	IDNA_ALLOW_UNASSIGN	ED
flags	or	
	IDNA_USE_STD3_ASCII_R	ULES.

#### **Returns**

Returns IDNA\_SUCCESS on success, or error code.

## idna\_to\_ascii\_8z ()

Convert UTF-8 domain name to ASCII string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

input	zero terminated input UTF-8 string.	
output	pointer to newly allocated	
output	output string.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
flags	or	
	IDNA_USE_STD3_ASCII_R	ULES.

Returns **IDNA\_SUCCESS** on success, or error code.

## idna\_to\_ascii\_lz ()

Convert domain name in the locale's encoding to ASCII string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

#### **Parameters**

	zero terminated input string	
input	encoded in the current	
	locale's character set.	
output	pointer to newly allocated	
σαιραι	output string.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
nags	or	
	IDNA_USE_STD3_ASCII_R	ULES.

#### Returns

Returns IDNA\_SUCCESS on success, or error code.

## idna\_to\_unicode\_4z4z ()

Convert possibly ACE encoded domain name in UCS-4 format into a UCS-4 string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

input	zero-terminated Unicode
mput	string.
output	pointer to newly allocated
output	output Unicode string.
	an Idna_flags value, e.g.,
flogs	IDNA_ALLOW_UNASSIGNED
flags	or
	IDNA_USE_STD3_ASCII_RULES.

Returns IDNA\_SUCCESS on success, or error code.

## idna\_to\_unicode\_8z4z ()

Convert possibly ACE encoded domain name in UTF-8 format into a UCS-4 string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

#### **Parameters**

innut	zero-terminated UTF-8	
input	string.	
output	pointer to newly allocated	
output	output Unicode string.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
	or	
	IDNA_USE_STD3_ASCII_R	ULES.

## Returns

Returns IDNA\_SUCCESS on success, or error code.

## idna\_to\_unicode\_8z8z ()

Convert possibly ACE encoded domain name in UTF-8 format into a UTF-8 string. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

#### **Parameters**

input	zero-terminated UTF-8	
input	string.	
output	pointer to newly allocated	
output	output UTF-8 string.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
	or	
	IDNA_USE_STD3_ASCII_R	ULES.

#### **Returns**

Returns IDNA\_SUCCESS on success, or error code.

#### idna\_to\_unicode\_8zlz ()

Convert possibly ACE encoded domain name in UTF-8 format into a string encoded in the current locale's character set. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

#### **Parameters**

input	zero-terminated UTF-8	
mput	string.	
	pointer to newly allocated	
output	output string encoded in the	
Output	current locale's character	
	set.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
	or	
	IDNA_USE_STD3_ASCII_R	ULES.

#### Returns

Returns **IDNA\_SUCCESS** on success, or error code.

## idna\_to\_unicode\_lzlz()

Convert possibly ACE encoded domain name in the locale's character set into a string encoded in the current locale's character set. The domain name may contain several labels, separated by dots. The output buffer must be deallocated by the caller.

## **Parameters**

	zero-terminated string	
input	encoded in the current	
	locale's character set.	
	pointer to newly allocated	
output	output string encoded in the	
output	current locale's character	
	set.	
	an Idna_flags value, e.g.,	
flags	IDNA_ALLOW_UNASSIGN	ED
	or	
	IDNA_USE_STD3_ASCII_R	ULES.

#### **Returns**

Returns IDNA\_SUCCESS on success, or error code.

## **Types and Values**

## **IDNAPI**

#define IDNAPI

## enum Idna\_rc

Enumerated return codes of idna\_to\_ascii\_4i(), idna\_to\_unicode\_44i() functions (and functions derived from those functions). The value 0 is guaranteed to always correspond to success.

## Members

IDNA\_SUCCESS

era tion. This value is guaranteed to ways be zero, the remaining ones are only guaranteed to hold nonzero values, for logical comparison purposes.

Successful

op-

	Error
	dur-
	ing
IDNA_STRINGPREP_ERROR	string
IDIVI_DIMINOTREE_EMMOR	prepa-
	ra-
	tion.
	Error
	dur-
	ing
	pun-
IDMA DUMINGODE EDDOD	y+
IDNA_PUNYCODE_ERROR	code
	op-
	er-
	tion.
	For
	IDNA_USE_STD3_ASCII_RULES,
	in-
	di-
	cate
	that
	the
IDNA_CONTAINS_NON_LDH	string
	con-
	tains
	non-
	LDH
	ASCII
	char-
	aç-
	ters.
	Same
	as
	IDNA_CONTAINS_NON_LDH
	,
	for
	com-
	pat-
	i-
IDNA_CONTAINS_LDH	bil-
	ity
	with
	typo
	in
	ear-
	lier
	ver-
	sions.
	1 -1

IDNA_CONTAINS_MINUS	For IDNA_USE_STD3_ASCII_RULES, in- di- cate that the string con- tains a lead- ing or trail- ing hyphen- minus (U+002D).
IDNA_INVALID_LENGTH	The fi- nal out- put string is not within the (in- clu- sive) range 1 to 63 char- ac- ters.
IDNA_NO_ACE_PREFIX	The string does not contain the ACE prefix (for ToUnicode).

The   ThASCII   OP   OP   OP   OP   OP   OP   OP		
IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAIN		The
Opt   er   a   tion   Oh   Oh   Oh   Oh   Oh   Oh   Oh   O		The
Opt   er   a   tion   Oh   Oh   Oh   Oh   Oh   Oh   Oh   O		ToASCII
Contains		
IDNA_ROUNDTRIP_VERIFY_ERROR		
IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_		et-
IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_ROUNDTRIP_VERIFY_ERROR  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_IDNA_		
IDNA_ROUNDTRIP_VERIFY_ERROR    Oil-   pirt   string   does   does   not   cqual   the   in-   pirt   cqual   the   in-   pirt   contains   cont		
IDNA_ROUNDTRIP_VERIFY_ERROR  pit string does not equal the in- plut.  The in- plut con- tains liDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_CONTAIN		tion
IDNA_ROUNDTRIP_VERIFY_ERROR  pit string does not equal the in- plut.  The in- plut con- tains liDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_CONTAIN		on
IDNA_KOUND TRIP_VERIF 1_ERROR  put string does not equal the in- put.  The in- put con- tuins the ACE pre- lix (for ToASCII).  Could not con- vert string in lo- cale cale cale cale cale cale cale cale		
string does not cqual the cqual the ir- plt.  The in- put contains Lains the ACE pree- ik (for TpASCII).  Could not con- vert string in lo- cale en- cod- ing.  Could not cale cale cale cale cale cale cale cale	IDMA DOLINDTDID VEDIEV EDDOD	out-
string does not equal the input.  IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTA	IDNA_ROUNDTRIP_VERIFT_ERROR	nut
does   not   equal   the   in   in   in   in   in   in   in   i		Ρμ.
does   not   equal   the   in   in   in   in   in   in   in   i		string
not equal the in- plut.  The in- plut con- tains the ACE plet lik (for ThASCII).  Could not con- vert string in lo- cale en- cod- ing.  Could not color ing.  IDNA_ICONV_ERROR  IDNA_ICONV_ERROR		
equal the   in   ppt.		
the in-path		not
the in-path		emial
in-   plt.     The   in-   plt     in-   plt     con-   tains     the   ACE     pre-   fik     (for   TPASCII).     Could   not     con-   vert     string     IDNA_ICONV_ERROR     In   lo-     cale   eh-     cod-     ing     Could     not     al-     lo-     cate     buffer ((this     is     typ-     i-     cally     a     fa-     fa-     tal     et-		
put.   The   in-   put   con-   con		
put.   The   in-   put   con-   con		in-
IDNA_CONTAINS_ACE_PREFIX  IDNA_CONTAINS_ACE_		
In- put   con- tains     the     ACE     pre- fix     (for     TbASCII).    Could     not     con- vert     string     in     lc- calle     en- cod- ing.     Could     not     lc- calle     en- cod- ing.     Could     Interval     Could     Con- vert     Could     Con- vert     Could     Con- vert     Could     C		pμt.
In- put   con- tains     the     ACE     pre- fix     (for     TbASCII).    Could     not     con- vert     string     in     lc- calle     en- cod- ing.     Could     not     lc- calle     en- cod- ing.     Could     Interval     Could     Con- vert     Could     Con- vert     Could     Con- vert     Interval		The
DNA_CONTAINS_ACE_PREFIX   times		
Contains		1 <b>n</b> -
Contains		put
tains   the   ACE   ACE   pre-fix   (for   TpASCII).		
IDNA_CONTAINS_ACE_PREFIX   the   ACE   Prefix   (for   ToASCII).		con-
IDNA_CONTAINS_ACE_PREFIX   the   ACE		tains
ACE   ppe-   fik   (for   ToASCII).     Could   not   convert   string   in   lo-   cate   en-   cod-   ing.     Could   not   al-   lo-   cate   buffer   (this   is   typ-   i-   call   ca	IDNA CONTAINS ACE PREFIX	
pre-   fik   (for   TDASCII).	IDNA_CONTAINS_ACE_PREFIX	
pre-   fix     (for     ToASCII).    Could     not     con-   vert     string     in     lo-   cale     en-   cod-   ing.    Could     not     al-   lo-   cate     buffer     (this     is     is     typ-   i-   cally     a     fa-   tall     er-		ACE
fix (for   TpASCII).     Could   not   con-vert   string   in   lo-cale   eh-cod-ing.   Could   not   cold   lo-cale   eh-cod-ing.   Could   not   al-lo-cate   buffer   this   is   typ-i-cally   a   fa-tal   fa-tal   tal		
(for TpASCII).		pre-
(for TpASCII).		
ToASCII).		
IDNA_ICONV_ERROR  IDNA_ICONV_E		
IDNA_ICONV_ERROR  IDNA_ICONV_E		ThASCII)
IDNA_ICONV_ERROR  IDNA_ICONV_E		
Convert   String   IDNA_ICONV_ERROR   In   Io-   Cale   en-   Cod-   ing.     Could   not   al-   Io-   Cate   buffer   (this   IDNA_MALLOC_ERROR   Is   Io-   Cally   a   fa-   fa-   tall   er-   Io-   Cally   a   fa-   tall   er-   Io-		Could
Convert   String   IDNA_ICONV_ERROR   In   Io-   Cale   en-   Cod-   ing.     Could   not   al-   Io-   Cate   buffer   (this   IDNA_MALLOC_ERROR   Is   Io-   Cally   a   fa-   fa-   tall   er-   Io-   Cally   a   fa-   tall   er-   Io-		not
Vert   String   in   lo-   cale   en-   cod-   ing.		
IDNA_ICONV_ERROR  in lo- cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-		con-
IDNA_ICONV_ERROR  in lo- cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-		vert
IDNA_ICONV_ERROR  in lo- cale en- cod- ing.  Could not al- lo- cate buffer (this  is typ- i- cally a fa- tal er-		
IDNA_ICONV_ERROR  in lo- cale en- cod- ing.  Could not al- lo- cate buffer (this  is typ- i- cally a fa- tal er-		string
lo- cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-	IDNA ICONY FRROR	
cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-	IDIVI_ICONV_DARKOR	
en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-		Id-
en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-		19
Could   not   al-   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa- tal   er-		
not   al-   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa-   tal   er-		cale
not   a -   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa-   tal   er-		cale en-
not   a -   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa-   tal   er-		cale en-
not   al-   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa-   tal   er-		cale en-
not   al-   lo-   cate   buffer   (this   is   typ-   i-   cally   a   fa-   tal   er-		cale en-
al-   lo-   cate     buffer     (this     is     typ-   i-   cally     a     fa-   tal     er-		cale en-
al-   lo-   cate     buffer     (this     is     typ-   i-   cally     a     fa-   tal     er-		cale en- cod- ing. Could
IDNA_MALLOC_ERROR  IDNA_MALLOC_ERROR  IDNA_MALLOC_ERROR  IDNA_MALLOC_ERROR  IS  typ-  i-  cally  a  fa-  tal  er-		cale en- cod- ing.  Could not
cate buffer (this is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not
buffer (this is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al-
buffer (this is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al-
IDNA_MALLOC_ERROR  (this is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al- lo-
IDNA_MALLOC_ERROR  is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al- lo- cate
IDNA_MALLOC_ERROR  is typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al- lo- cate buffer
typ- i- cally a fa- tal er-		cale en- cod- ing.  Could not al- lo- cate buffer
i- cally a fa- tal er-		cale en- cod- ing.  Could not al- lo- cate buffer (this
i- cally a fa- tal er-	IDNA MALLOC ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this
cally a fa- tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is
a fa- tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ-
a fa- tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i-
fa- tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i-
tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally
tal er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a
er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a
er-	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa-
	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal
	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal
	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-
	IDNA_MALLOC_ERROR	cale en- cod- ing.  Could not al- lo- cate buffer (this is typ- i- cally a fa- tal er-

	Could
	not
	dlopen
	the
	lib-
	cidn
IDMA DI ODENI EDDOD	DSO
IDNA_DLOPEN_ERROR	(only
	used
	in-
	ter-
	nally
	in
	libc).

## enum Idna\_flags

Flags to pass to idna\_to\_ascii\_4i(), idna\_to\_unicode\_44i() etc.

## **Members**

Te-   ject   strings   con-   tain-   ing   unassigned   Unicode   code   cod		Don't
IDNA_ALLOW_UNASSIGNED  ing uhas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		re-
IDNA_ALLOW_UNASSIGNED  ing uhas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		ject
IDNA_ALLOW_UNASSIGNED  ing unas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		strings
IDNA_ALLOW_UNASSIGNED  ing unas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES		
IDNA_ALLOW_UNASSIGNED  ing unas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES		tain-
unas- signed Uni- code code points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name	IDNA ALLOW UNASSIGNED	
signed Uni- code code points.  Validate strings ac- cord- ing to STD3 TIDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES  signed Uni- code code points.  Validate strings ac- cord- ing to sort- ing		
Uni- code code points.  Validate strings ac- cord- ing to STD3 TURS  IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES		
code code points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		
code points.  Validate strings ac- cord- ing to  STD3 rules (i.e., nor- mal host name		
points.  Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		
Validate strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		
strings ac- cord- ing to STD3 rules (i.e., nor- mal host name		
ac- cord- ing to STD3 rules (i.e., nor- mal host name		
IDNA_USE_STD3_ASCII_RULES  IDNA_USE_STD3_ASCII_RULES  STD3 rules (i.e., nor- mal host name		
ing to STD3 rules (i.e., nor- mal host name		
IDNA_USE_STD3_ASCII_RULES  to STD3 rules (i.e., nor- mal host name		
IDNA_USE_STD3_ASCII_RULES  STD3 rules (i.e., nor- mal host name		
rules (i.e., nor- mal host name		
(i.e., nor-mal host name	IDNA_USE_STD3_ASCII_RULES	
nor- mal host name		
mal host name		
host name		
name		
rules).		
		rules).

## IDNA\_ACE\_PREFIX

```
# define IDNA_ACE_PREFIX "xn--"
```

The IANA allocated prefix to use for IDNA. "xn--"

## 1.2 stringprep

stringprep —

## **Types and Values**

#define	IDNAPI
#define	STRINGPREP_VERSION
enum	Stringprep_rc
enum	Stringprep_profile_flags
enum	Stringprep_profile_steps
#define	STRINGPREP_MAX_MAP_CHARS

## **Description**

## **Functions**

## **Types and Values**

## **IDNAPI**

#define IDNAPI

## STRINGPREP\_VERSION

```
# define STRINGPREP_VERSION "1.35"
```

String defined via CPP denoting the header file version number. Used together with <a href="stringprep\_check\_version">stringprep\_check\_version</a>() to verify header file and run-time library consistency.

## enum Stringprep\_rc

Enumerated return codes of stringprep(), stringprep\_profile() functions (and macros using those functions). The value 0 is guaranteed to always correspond to success.

## Members

	Successful
	op-
	er-
	tion.
	This
	value
	is
	guar- an-
	teed
	to
	al-
	ways
	be
	zero,
	the
	re-
	main-
	ing
	ones
STRINGPREP_OK	are
	only
	guar-
	an-
	teed
	to
	hold
	non-
	zero
	val-
	ues,
	for
	log-
	log- i-
	cal
	com-
	par-
	par- i-
	sφn
	pur-
	noses
	poses.

STRINGPREP_CONTAINS_UNASSIGNED	String con- tain unas- signed Uni- code code points, which is for- bid- den by the pro- file. String
STRINGPREP_CONTAINS_PROHIBITED	contain code points prohib- ited by the profile.
STRINGPREP_BIDI_BOTH_L_AND_RAL	String con- tain code points with con- flict- ing bidi- rec- tion cat- e- gory.

Leading and trail- ing   char- ac- ter   cir   in		
stringprep_bidi_leadtrail_NOT_ral  STRINGPREP_Bidi_		Leading
STRINGPREP_BIDI_LEADTRAII_NOT_RAI.  STRINGPREP_BIDI_LEADTRAII_NOT_RAI.  STRINGPREP_BIDI_LEADTRAII_NOT_RAI.  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BID		
ing character ch		
char- ac- ter in string string not of proper bidi- rec- fibral cal- cl- gory.  Contains pro- hib- iiied code points de- tected by bidi- rec- tional cal- cl- gory.  STRINGPREP BIDI CONTAINS PROHIBITED  Buffer handed to finc- fijon was too schall, This usu- ally ii- cal- cide cal- cide cal- cide cal- cide cal- cijon was too schall, This usu- ally ii- di- cal- cal- cal- iiig ap- pib- cal-		trail-
char- ac- ter in string string not of proper bidi- rec- fibral cal- cl- gory.  Contains pro- hib- iiied code points de- tected by bidi- rec- tional cal- cl- gory.  STRINGPREP BIDI CONTAINS PROHIBITED  Buffer handed to finc- fijon was too schall, This usu- ally ii- cal- cide cal- cide cal- cide cal- cide cal- cijon was too schall, This usu- ally ii- di- cal- cal- cal- iiig ap- pib- cal-		ing
STRINGPREP_BIDL_LEADTRAIL_NOT_RAL  String met of proper bidirections cate cate depoints deceded by bidirected by string s		char
STRINGPREP_BIDL_LEADTRAIL_NOT_RAL.  String not of proper bidic rectional cate e gbry.  Contains pro-lib-lib-lib-lib-lib-lib-lib-lib-lib-lib		
STRINGPREP_BIDI_LEADTRAIL_NOT_RAI.  siring mot of proper bidi-rec-tipnal cal-cal-elemin in the call-inig app-pi-lead siring in the call-inig app-pi-lead siring app-lead siring app-pi-lead siring app-lead s		
STRINGPREP_BIDL_LEADTRAIL_NOT_RAIL  string npt of proper bidi- rec- tional cat- e- gnry.  Contains pro- hib- nitd code points de- tected by bidi- rec- tional code.  Buffer handed to func- tion tion was too small. This usu- ally in- in- tin- tin- tin- tin- tin- tin- t		ter
STRINGPREP_BIDL_LEADTRAIL_NOT_RAIL    String   not of   proper   bidi-   rectional   cait   c-   gory.		in
STRINGPREP_BIDL_CONTAINS_PROHIBITED  STRINGPREP_		
of proper bidinger bidinger proper bidinger proper bidinger proper bidinger proper bidinger proper bidinger pr	STRINGPREP_BIDI_LEADTRAIL_NOT_RAL	
pioper bidi- rec- tional cat- e- gryy.  Contains pro- hib- ited code points de- teteted by by bidi- rec- tional code.  Buffer hinded to func- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pii- cate		not
bidirectional cate characteristics of the cate characteris		of
bidirectional cate epgory. Contains prohibited code points detected by bidirectional code.  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  Example 1		proper
rectional cate e   e   gory.  Contains prohib- ited code points de- tected by by bidi- rectional code.  Buffer hinded to fine- tion was too shall. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  STRINGPREP_TOO_SMALL_BUFFER  This de- cate a   prob- lem in the call- ing ap- pli- cate cate call- ing ap- pli- cate		bidi-
tional cat- cat- cy gory.  Contains pro- hib- ited code points  de- tetred by bidi- rec- tional code.  Buffer handed to fine- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in tile call- ing ap- ppli- ca-		
cat- e   gory.  Contains pro- hib- ited code points  STRINGPREP_BIDL_CONTAINS_PROHIBITED  STRINGPREP_BIDL_CONTAINS_PROHIBITED  Buffer handed to func- tional code.  Buffer handed to func- tion was too shaall. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  Cate a prob- lem in the cate cate cate cate a prob- lem in the cate cate cate cate cate a prob- lem in the cate cate cate cate cate cate cate cat		
c- gbry.  Contains  Pro- hib- itged code points  de- tected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pii- cal-		
gbry. Contains pro- hib- itted code points de- tected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- Cate a prob- lem in the catl- in the catl- in the catl- in gap- pli- ca-		cat-
gery. Contains pro- hib- ited code points de- tected by bidi- rec- tibnal code.  Buffer handed to func- tion was too small. This usu- ally in- cate a prob- lem in the call- ing ap- pli- ca-		e+
Contains pro- hib- ited code points de- tected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER   Cate a prob- lem if the call- ing ap- pli- ca-		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_		gory.
STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  SUBJECT  SUBJECT  STRINGPREP_TOO_SMALL_BUFFER		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  STRINGPREP_BIDI_CONTAINS_PROHIBITED  SUBJECT  SUBJECT  STRINGPREP_TOO_SMALL_BUFFER		pro-
STRINGPREP_BIDL_CONTAINS_PROHIBITED  aby bidirect tional code.  Buffer hinded to fine- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  approblem in the call- ing ap- pli- ca- care ap- pli- ca- care ap- pli- ca- care		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  detected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally ir- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  points de- tected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pji- ca-		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  detected by bidi- rec- tional code.  Buffer handed to func- tion was too shall, This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
STRINGPREP_BIDI_CONTAINS_PROHIBITED  detected by bidi- rec- tional code.  Buffer handed to func- tion was too shall, This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		points
tected by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem if the call- ing ap- pii- ca-	STRINGPREP BIDI CONTAINS PROHIBITED	de-
by bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
bidi- rec- tional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
rectional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
tional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		bidi-
tional code.  Buffer handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
code.  Buffer handed to func- tion was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
Buffer handed to function was too small. This usu-ally in-di-cate a problem in the call-ing applicate of the call-ing applicate.		
handed to func- tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
to function was too small. This usu-ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER   Cate a prob- lem in the call- ing ap- pli- ca-		Buffer
to function was too small. This usu-ally in- STRINGPREP_TOO_SMALL_BUFFER   STRINGPREP_TOO_SMALL_BUFFER   Cate a prob- lem in the call- ing ap- pli- ca-		handed
function was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
tion was too small. This usu- ally in- di- cate a prob- lem in the call- ing ap- pli- ca-		
was too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
too small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		tion
too small. This usu- ally in- Gate a prob- lem in the call- ing ap- pli- ca-		was
small. This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
This usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- Ca-		small.
usu- ally in- STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- Ca-		This
STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
STRINGPREP_TOO_SMALL_BUFFER  di- cate a prob- lem in the call- ing ap- pli- ca-		
STRINGPREP_TOO_SMALL_BUFFER  dicate a prob- lem in the call- ing ap- pli- ca-		
cate a prob- lem in the call- ing ap- pli- ca-		
cate a prob- lem in the call- ing ap- pli- ca-	STRINGPREP TOO SMALL BUFFER	di-
a problem in the calling applicate the plicate calling applicate calling applicate.		
prob- lem in the call- ing ap- pli- ca-		
lem in the call- ing ap- pli- ca-		
lem in the call- ing ap- pli- ca-		
in the call- ing ap- pli- ca-		
the call- ing ap- pli- ca-		
call- ing ap- pli- ca-		
ing ap- pli- ca-		
ap- pli- ca-		
ap- pli- ca-		
pli- ca-		
ca-		ap-   - ].
ca-		
tion.		ca-
upit.		tion.
		**

	The
	string-
	prep
	pro-
	file
	was
	in-
	con-
	sis-
	tent.
	This
	usu-
CTRINCIPED DROEILE EDROR	ally
STRINGPREP_PROFILE_ERROR	in-
	di-
	cate
	an
	in-
	ter-
	nal
	er-
	ror
	in
	the
	li-
	brary.
	The
	sup-
	plied
	flag
	con-
	flicted
	with
	pro-
	pio-
	file.
	This
	usu-
	ally
	in-
STRINGPREP_FLAG_ERROR	
	di-
	cate
	a
	prob-
	lem
	in
	the
	call-
	ing
	ap-
	pli-
	ca-
	tion.

	The
	sup-
	plied
	pro-
	file
	name
STRINGPREP_UNKNOWN_PROFILE	was
	not
	known
	to
	the
	li-
	brary.
	Could
	not
	con-
	vert
	string
STRINGPREP_ICONV_ERROR	in
STAIN OF THE TENTON	10-
	cale
	en-
	cod-
	ing.
	The
	Uni-
	code
	NFKC
	op-
	er-
	a-
	(a)
	tion
	failed.
	This
	usu-
	ally
STRINGPREP_NFKC_FAILED	in- di-
	di-
	cate
	an
	an in-
	ter-
	mol
	nal
	er-
	ror
	in
	in the
	tne
	li-
	li- brary.
	1 -1>.

STRINGPREP\_MALLOC\_ERROR

The malloc() was out of memory. This is usually a fatal error.

## enum Stringprep\_profile\_flags

Stringprep profile flags.

## Members

Disable the NFKC normalization, as well as se-lect-ing the non-NFKC case folding tables. Usually the profile STRINGPREP\_NO\_NFKC specifies BIDI and NFKC settings, and applications should not override it unless in speclal situ a tions.

	Disable
	the
	BIDI
	step.
	Usu-
	ally
	the
	pro-
	file
	spec-
	i-
	1-   6-pg
	fies
	BIDI
	and
	NFKC
	set-
	tings,
CERTIFICATION NO. DIDI	and
STRINGPREP_NO_BIDI	ap-
	pli-
	ca-
	tions
	should
	not
	over-
	ride
	it
	un-
	less
	in
	spe-
	cial
	sit-
	u <del> </del>
	a <del> </del>
	tions.

Make the library return with an error if string STRINGPREP\_NO\_UNASSIGNED contains unassigned chara¢ters a¢cording to profile.

## enum Stringprep\_profile\_steps

Various steps in the stringprep algorithm. You really want to study the source code to understand this one. Only useful if you want to add another profile.

## **Members**

STRINGPREP_NFKC	The NFKC step.
STRINGPREP_BIDI	The BIDI step.
STRINGPREP_MAP_TABLE	The MAP step.
STRINGPREP_UNASSIGNED_TABLE	The Unas-signed step.
STRINGPREP_PROHIBIT_TABLE	The Pro- hib- ited step.
STRINGPREP_BIDI_PROHIBIT_TABLE	The BIDI- Prohibited step.

STRINGPREP_BIDI_RAL_TABLE	The BIDI- RAL step.
STRINGPREP_BIDI_L_TABLE	The BIDI- L step.

## STRINGPREP\_MAX\_MAP\_CHARS

```
# define STRINGPREP_MAX_MAP_CHARS 4
```

Maximum number of code points that can replace a single code point, during stringprep mapping.

## 1.3 punycode

punycode —

#### **Functions**

const char *	punycode_strerror ()
int	punycode_encode ()
int	punycode_decode ()

## **Types and Values**

#define	IDNAPI
enum	Punycode_status
typedef	punycode_uint

## **Description**

## **Functions**

## punycode\_strerror ()

```
const char~*
punycode_strerror (Punycode_status rc);
```

Convert a return code integer to a text string. This string can be used to output a diagnostic message to the user.

PUNYCODE\_SUCCESS: Successful operation. This value is guaranteed to always be zero, the remaining ones are only guaranteed to hold non-zero values, for logical comparison purposes. PUNYCODE\_BAD\_INPUT: Input is invalid. PUNYCODE\_BIG\_OUTPUT: Output would exceed the space provided. PUNYCODE\_OVERFLOW: Input needs wider integers to process.

rc	an Punycode_status return
ic	code.

Returns a pointer to a statically allocated string containing a description of the error with the return code rc.

## punycode\_encode ()

Converts a sequence of code points (presumed to be Unicode code points) to Punycode.

	The number of code points
innut langth	in the input array and the
input_length	number of flags in the
	case_flags array.
	An array of code points.
	They are presumed to be
	Unicode code points, but
	that is not strictly
	REQUIRED. The array
	contains code points, not
	code units. UTF-16 uses
	code units D800 through
immut	DFFF to refer to code
input	points 1000010FFFF. The
	code points D800DFFF do
	not occur in any valid
	Unicode string. The code
	points that can occur in
	Unicode strings (0D7FF
	and E00010FFFF) are also
	called Unicode scalar
	values.

case_flags	A NULL pointer or an array of boolean values parallel to the <i>input</i> array. Nonzero (true, flagged) suggests that the corresponding Unicode character be forced to uppercase after being decoded (if possible), and zero (false, unflagged) suggests that it be forced to lowercase (if possible).  ASCII code points (07F) are encoded literally, except that ASCII letters are forced to uppercase or	
	lowercase according to the corresponding case flags. If case_flags is a NULL pointer then ASCII letters are left as they are, and other code points are treated as unflagged.	
output_length	The caller passes in the maximum number of ASCII code points that it can receive. On successful return it will contain the number of ASCII code points actually output.	
output	An array of ASCII code points. It is *not* null-terminated; it will contain zeros if and only if the input contains zeros. (Of course the caller can leave room for a terminator and add one if needed.)	

The return value can be any of the Punycode\_status values defined above except PUNYCODE\_BAD\_INPUT. If not PUNYCODE\_SUCCESS, then <code>output\_size</code> and <code>output</code> might contain garbage.

## punycode\_decode ()

Converts Punycode to a sequence of code points (presumed to be Unicode code points).

input_length	The number of ASCII code	
mput_iengui	points in the input array.	
innut	An array of ASCII code	
input	points (07F).	
	The caller passes in the	
	maximum number of code	
	points that it can receive	
	into the output array	
	(which is also the	
	maximum number of flags	
	that it can receive into the	
	case_flags array, if	
	case_flags is not a	
	NULL pointer). On	
	successful return it will	
	contain the number of code	
	points actually output	
output_length	(which is also the number	
. – 0	of flags actually output, if	
	case_flags is not a null	
	pointer). The decoder will	
	never need to output more	
	code points than the number	
	of ASCII code points in the	
	input, because of the way	
	the encoding is defined.	
	The number of code points	
	output cannot exceed the	
	maximum possible value of	
	a punycode_uint, even if the	
	supplied output_length	
	is greater than that.	
	An array of code points like	
	the input argument of	
output	punycode_encode() (see	
	above).	
	A NULL pointer (if the	
	flags are not needed by the	
	caller) or an array of	
	boolean values parallel to	
	the output array. Nonzero	
	(true, flagged) suggests that	
	the corresponding Unicode	
	character be forced to	
	uppercase by the caller (if	
case_flags	possible), and zero (false,	
	unflagged) suggests that it	
	be forced to lowercase (if	
	possible). ASCII code	
	points (07F) are output	
	already in the proper case,	
	but their flags will be set	
	appropriately so that	
	applying the flags would be	
	harmless.	
	marinicos.	

The return value can be any of the Punycode\_status values defined above. If not PUNYCODE\_SUCCESS, then  $output\_length$ , output, and  $case\_flags$  might contain garbage.

## **Types and Values**

## **IDNAPI**

#define

IDNAPI

## enum Punycode\_status

Enumerated return codes of punycode\_encode() and punycode\_decode(). The value 0 is guaranteed to always correspond to success.

#### **Members**

	Successful
	op-
	er-
	a
	tion.
	This
	value
	value
	is
	guar-
	an-
	teed
	to
	al-
	ways
	be
	zero,
	the
	re-
	main-
	ing
	ones
PUNYCODE_SUCCESS	are
	only
	guar-
	an-
	teed
	to
	hold
	non-
	zero
	val-
	ues,
	for
	log-
	i-
	cal
	com-
	par-   i-
	søn
	pur-
	poses.
	Input
DIDIVOODE DAD DIDIVE	is
PUNYCODE_BAD_INPUT	in-
	valid.
	Output
	would
	ex-
PUNYCODE_BIG_OUTPUT	ceed
TOM TOODE_DIO_OUTFUL	the
	space
	pro-
	vided.
	viaca.

PUNYCODE\_OVERFLOW

## punycode\_uint

```
typedef uint32_t punycode_uint;
```

Input needs wider in-

tegers to process.

Unicode code point data type, this is always a 32 bit unsigned integer.

## 1.4 pr29

pr29 —

## **Functions**

const char *	pr29_strerror ()
int	pr29_4 ()
int	pr29_4z ()
int	pr29_8z ()

## **Types and Values**

#define	IDNAPI
enum	Pr29_rc

## **Description**

## **Functions**

#### pr29\_strerror()

```
const char~*
pr29_strerror (Pr29_rc rc);
```

Convert a return code integer to a text string. This string can be used to output a diagnostic message to the user.

PR29\_SUCCESS: Successful operation. This value is guaranteed to always be zero, the remaining ones are only guaranteed to hold non-zero values, for logical comparison purposes. PR29\_PROBLEM: A problem sequence was encountered. PR29\_STRINGPREP\_ERROR: The character set conversion failed (only for pr29\_8z()).

#### **Parameters**

rc an Pr29\_rc return code.

Returns a pointer to a statically allocated string containing a description of the error with the return code rc.

## pr29\_4 ()

Check the input to see if it may be normalized into different strings by different NFKC implementations, due to an anomaly in the NFKC specifications.

#### **Parameters**

in	input array with unicode code points.	
len	length of input array with	
	unicode code points.	

#### Returns

Returns the Pr29\_rc value PR29\_SUCCESS on success, and PR29\_PROBLEM if the input sequence is a "problem sequence" (i.e., may be normalized into different strings by different implementations).

#### pr29\_4z()

```
int
pr29_4z (const uint32_t *in);
```

Check the input to see if it may be normalized into different strings by different NFKC implementations, due to an anomaly in the NFKC specifications.

#### **Parameters**

in	zero terminated array of
	Unicode code points.

## Returns

Returns the Pr29\_rc value PR29\_SUCCESS on success, and PR29\_PROBLEM if the input sequence is a "problem sequence" (i.e., may be normalized into different strings by different implementations).

## pr29\_8z ()

```
int
pr29_8z (const char *in);
```

Check the input to see if it may be normalized into different strings by different NFKC implementations, due to an anomaly in the NFKC specifications.

in	zero terminated input
111	UTF-8 string.

Returns the Pr29\_rc value PR29\_SUCCESS on success, and PR29\_PROBLEM if the input sequence is a "problem sequence" (i.e., may be normalized into different strings by different implementations), or PR29\_STRINGPREP\_ERROR if there was a problem converting the string from UTF-8 to UCS-4.

## **Types and Values**

#### **IDNAPI**

|--|--|

## enum Pr29\_rc

Enumerated return codes for pr29\_4(), pr29\_4z(), pr29\_8z(). The value 0 is guaranteed to always correspond to success.

#### **Members**

	Successful
	op-
	er-
	a+
	tion.
	This
	value
	is
	guar-
	an-
	teed
	to
	al-
	ways
	be
	zero,
	the
	re-
	main-
	ing
	ones
PR29_SUCCESS	are
	only
	guar-
	an-
	teed
	to
	hold
	non-
	zero
	val-
	ues,
	for
	log-
	i-
	cal
	com-
	par-
	i- i-
	son
	pur-
	poses.
	A A
	prob- lem
DD20 DDODLEM	se-
PR29_PROBLEM	quence
	was
	en-
	coun-
	tered.

PR29\_STRINGPREP\_ERROR

The character set conversion failed (only for pr29\_8z()).

## 1.5 tld

tld —

## **Types and Values**

#define | IDNAPI

**Description** 

**Functions** 

**Types and Values** 

**IDNAPI** 

#define IDNAPI

## 1.6 idn-free

idn-free —

## **Types and Values**

#define IDNAPI

**Description** 

**Functions** 

**Types and Values** 

**IDNAPI** 

#define IDNAPI

## **Chapter 2**

## Index

```
IDNA_ACE_PREFIX, 13
Idna_flags, 13
Idna_rc, 9
idna_strerror, 3
idna_to_ascii_4i, 3
idna_to_ascii_4z, 5
idna_to_ascii_8z, 5
idna_to_ascii_lz, 6
idna_to_unicode_44i, 4
idna_to_unicode_4z4z, 6
idna_to_unicode_8z4z, 7
idna_to_unicode_8z8z, 7
idna_to_unicode_8zlz, 8
idna_to_unicode_lzlz, 8
IDNAPI, 9, 14, 28, 33, 35
P
pr29_4, 31
pr29_4z, 31
pr29_8z, 31
Pr29_rc, 33
pr29_strerror, 30
punycode_decode, 26
punycode_encode, 25
Punycode_status, 28
punycode_strerror, 24
punycode_uint, 30
STRINGPREP_MAX_MAP_CHARS, 24
Stringprep_profile_flags, 20
Stringprep_profile_steps, 23
Stringprep_rc, 14
STRINGPREP_VERSION, 14
```