Independent Submission
Request for Comments: 5895
Category: Informational
ISSN: 2070-1721

P. Resnick
Qualcomm Incorporated
P. Hoffman
VPN Consortium

September 2010

Mapping Characters for Internationalized Domain Names in Applications (IDNA) 2008

Abstract

In the original version of the Internationalized Domain Names in Applications (IDNA) protocol, any Unicode code points taken from user input were mapped into a set of Unicode code points that "made sense", and then encoded and passed to the domain name system (DNS). The IDNA2008 protocol (described in RFCs 5890, 5891, 5892, and 5893) presumes that the input to the protocol comes from a set of "permitted" code points, which it then encodes and passes to the DNS, but does not specify what to do with the result of user input. This document describes the actions that can be taken by an implementation between receiving user input and passing permitted code points to the new IDNA protocol.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5895.

1. Introduction

This document describes the operations that can be applied to user input in order to get it into a form that is acceptable by the Internationalized Domain Names in Applications (IDNA) protocol [IDNA2008protocol]. It includes a general implementation procedure for mapping.

It should be noted that this document does not specify the behavior of a protocol that appears "on the wire". It describes an operation that is to be applied to user input in order to prepare that user input for use in an "on the network" protocol. As unusual as this may be for a document concerning Internet protocols, it is necessary to describe this operation for implementors who may have designed around the original IDNA protocol (herein referred to as IDNA2003), which conflates this user-input operation into the protocol.

It is very important to note that there are many potential valid mappings of characters from user input. The mapping described in this document is the basis for other mappings, and is not likely to be useful without modification. Any useful mapping will have features designed to reduce the surprise for users and is likely to be slightly (or sometimes radically) different depending on the locale of the user, the type of input being used (such as typing, copy-and-paste, voice, and so on), the type of application used, etc. Although most common mappings will probably produce similar results for the same input, there will be subtle differences between applications.

1.1. The Dividing Line between User Interface and Protocol

The user interface to applications is much more complicated than most network implementers think. When we say "the user enters an internationalized domain name in the application", we are talking about a very complex process that encompasses everything from the user formulating the name and deciding which symbols to use to

express that name, to the user entering the symbols into the computer using some input method (be it a keyboard, a stylus, or even a voice recognition program), to the computer interpreting that input (be it keyboard scan codes, a graphical representation, or digitized sounds) into some representation of those symbols, through finally normalizing those symbols into a particular character repertoire in an encoding recognizable to IDNA processes and the domain name system.

Considerations for a user interface for internationalized domain names involves taking into account culture, context, and locale for any given user. A simple and well-known example is the lowercasing of the letter LATIN CAPITAL LETTER I (U+0049) when it is used in the Turkish and other languages. A capital "I" in Turkish is properly lowercased to a LATIN SMALL LETTER DOTLESS I (U+0131), not to a LATIN SMALL LETTER I (U+0069). This lowercasing is clearly dependent on the locale of the system and/or the locale of the user. Using a single context-free mapping without considering the user interface properties has the potential of doing exactly the wrong thing for the user.

The original version of IDNA conflated user interface processing and protocol. It took whatever characters the user produced in whatever encoding the application used, assumed some conversion to Unicode code points, and then without regard to context, locale, or anything about the user's intentions, mapped them into a particular set of other characters, and then re-encoded them in Punycode, in order to have the entire operation be contained within the protocol. Ignoring context, locale, and user preference in the IDNA protocol made life significantly less complicated for the application developer, but at the expense of violating the principle of "least user surprise" for consumers and producers of domain names.

In IDNA2008, the dividing line between "user interface" and "protocol" is clear. The IDNA2008 specification defines the protocol part of IDNA: it explicitly does not deal with the user interface. Mappings such as the one described in this document explicitly deal with the user interface and not the protocol. That is, a mapping is only to be applied before a string of characters is treated as a domain name (in the "user interface") and is never to be applied during domain name processing (in the "protocol").

1.2. The Design of This Mapping

The user interface mapping in this document is a set of expansions to IDNA2008 that are meant to be sensible and friendly and mostly obvious to people throughout the world when using typical applications with domain names that are entered by hand. It is also

designed to let applications be mostly backwards compatible with IDNA2003. By definition, it cannot meet all of those design goals for all people, and in fact is known to fail on some of those goals for quite large populations of people.

A good mapping in the real world might use the "sensible and friendly and mostly obvious" design goal but come up with a different algorithm. Many algorithms will have results that are close to what is described here, but will differ in assumptions about the users' way of thinking or typing. Having said that, it is likely that some mappings will be significantly different. For example, a mapping might apply to a spoken user interface instead of a typed one. Another example is that a mapping might be different for users that are typing than for users that are copying-and-pasting from different applications. Yet another example is that a user interface that allows typed input that is transliterated from Latin characters could have very different mappings than one that applies to typing in other character sets; this would be typical in a Pinyin input method for Chinese characters.

2. The General Procedure

This section defines a general algorithm that applications ought to implement in order to produce Unicode code points that will be valid under the IDNA protocol. An application might implement the full mapping as described below, or it can choose a different mapping. This mapping is very general and was designed to be acceptable to the widest user community, but as stated above, it does not take into account any particular context, culture, or locale.

The general algorithm that an application (or the input method provided by an operating system) ought to use is relatively straightforward:

1. Uppercase characters are mapped to their lowercase equivalents by using the algorithm for mapping case in Unicode characters. This step was chosen because the output will behave more like ASCII host names behave.

2. Fullwidth and halfwidth characters (those defined with Decomposition Types <wide> and <narrow>) are mapped to their decomposition mappings as shown in the Unicode character database. This step was chosen because many input mechanisms, particularly in Asia, do not allow you to easily enter characters in the form used by IDNA2008. Even if they do allow the correct character form, the user might not know which form they are entering.

3. All characters are mapped using Unicode Normalization Form C (NFC). This step was chosen because it maps combinations of combining characters into canonical composed form. As with the fullwidth/halfwidth mapping, users are not generally aware of the particular form of characters that they are entering, and IDNA2008 requires that only the canonical composed forms from NFC be used.

4. [IDNA2008protocol] is specified such that the protocol acts on the individual labels of the domain name. If an implementation of this mapping is also performing the step of separation of the parts of a domain name into labels by using the FULL STOP character (U+002E), the IDEOGRAPHIC FULL STOP character (U+3002) can be mapped to the FULL STOP before label separation occurs. There are other characters that are used as "full stops" that one could consider mapping as label separators, but their use as such has not been investigated thoroughly. This step was chosen because some input mechanisms do not allow the user to easily enter proper label separators. Only the IDEOGRAPHIC FULL STOP character (U+3002) is added in this mapping because the authors have not fully investigated the applicability of other characters and the environments where they should and should not be considered domain name label separators.

Note that the steps above are ordered.

Definitions for the rules in this algorithm can be found in [Unicode52]. Specifically:

- Unicode Normalization Form C can be found in Annex #15 of [Unicode-UAX15].

- In order to map uppercase characters to their lowercase equivalents (defined in Section 3.13 of [Unicode52]), first map characters to the "Lowercase_Mapping" property (the "<lower>" entry in the second column) in <http://www.unicode.org/Public/UNIDATA/SpecialCasing.txt>, if any. Then, map characters to the "Simple_Lowercase_Mapping" property (the fourteenth column) in <http://www.unicode.org/Public/UNIDATA/UnicodeData.txt>, if any.

- In order to map fullwidth and halfwidth characters to their decomposition mappings, map any character whose "Decomposition_Type" (contained in the first part of the sixth column) in <http://www.unicode.org/Public/UNIDATA/UnicodeData.txt> is either "<wide>" or "<narrow>" to the "Decomposition_Mapping" of that character (contained in the second part of the sixth column) in <http://www.unicode.org/Public/UNIDATA/UnicodeData.txt>.

- The Unicode Character Database [TR44] has useful descriptions of the contents of these files.

If the mappings in this document are applied to versions of Unicode later than Unicode 5.2, the later versions of the Unicode Standard should be consulted.

These form a minimal set of mappings that an application should strongly consider doing. Of course, there are many others that might be done.

3. Implementing This Mapping

If you are implementing a mapping for an application or operating system by using exactly the four steps in Section 2, the authors of this document have a request: please don't. We mean it. Section 2 does not describe a universal mapping algorithm because, as we said, there is no universally-applicable mapping algorithm.

If you read the material in Section 2 without reading Section 1, go back and carefully read all of Section 1; in many ways, Section 1 is more important than Section 2. Further, you can probably think of user interface considerations that we did not list in Section 1. If you did read Section 1 but somehow decided that the algorithm in Section 2 is completely correct for the intended users of your application or operating system, you are probably not thinking hard enough about your intended users.

4. Security Considerations

This document suggests creating mappings that might cause confusion for some users while alleviating confusion in other users. Such confusion is not covered in any depth in this document (nor in the other IDNA-related documents).

5. Acknowledgements

This document is the product of many contributions from numerous people in the IETF.

6. Normative References

[IDNA2008protocol] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, August 2010.

[TR44] The Unicode Consortium, "Unicode Technical Report #44: Unicode Character Database", September 2009, <http://www.unicode.org/reports/tr44/tr44-4.html>.

[Unicode-UAX15] The Unicode Consortium, "Unicode Standard Annex #15: Unicode Normalization Forms, Revision 31", September 2009, <http://www.unicode.org/reports/tr15/tr15-31.html>.

[Unicode52] The Unicode Consortium. The Unicode Standard, Version 5.2.0, defined by: "The Unicode Standard, Version 5.2.0", (Mountain View, CA: The Unicode Consortium, 2009. ISBN 978-1-936213-00-9). <http://www.unicode.org/versions/Unicode5.2.0/>.

Authors' Addresses

Peter W. Resnick
Qualcomm Incorporated
5775 Morehouse Drive
San Diego, CA 92121-1714
US

Phone: +1 858 651 4478
EMail: presnick@qualcomm.com
URI: http://www.qualcomm.com/~presnick/

Paul Hoffman
VPN Consortium
127 Segre Place
Santa Cruz, CA 95060
US

Phone: 1-831-426-9827
EMail: paul.hoffman@vpnc.org