

Internationalised Domain Names and Nominet - an Introduction

Clive Feather

Draft 2, 2004-08-02

Introduction

The “traditional” Internet was mostly designed by English speakers and therefore was based around the ASCII character set. When this was applied to domain names, this meant they were limited to the 26 “usual” unaccented Latin letters, the 10 “Arabic” digits, and hyphen (plus, of course, dot between components). In addition, it was decided that the case of letters would not be significant – **example**, **EXAMPLE**, and **eXaMPIE** are all equivalent.

Now that the Internet is truly international, it is unreasonable for domain names to be limited to these 37 characters (referred to in the rest of this paper as the “LDH” – letters, digits, and hyphen – character set). Even in other countries that use the Latin alphabet, people want to be able to spell their names properly, meaning that it is necessary to cope with accents like â and additional letters like ß. When we look further afield, there are many places where either non-Latin alphabets are used – such as Greek, Cyrillic, or Arabic – or other forms of writing – such as Chinese or Thai. It is unreasonable to expect them to use a completely different writing system for domain names.

It would be naïve to dismiss this as a “minority” or “foreign” problem – neither English nor Welsh can be properly written using only the LDH characters.

The IDNA approach

The approach chosen by IETF to solve this problem is called *Internationalizing Domain Names in Applications*, or *IDNA*; it is defined in RFC3490. In this approach, the DNS infrastructure is left completely untouched. Instead, individual applications are upgraded to recognise domain names in Unicode (see below). When they are given such a domain name (an *Internationalized Domain Name*, or *IDN*), they convert all labels (the components of domain names separated by dots) that are not written in the LDH set into *ACE* (*ASCII Compatible Encoding*), which is an encoding of Unicode using only the LDH characters. The resulting “IDN-unaware” domain name is then passed to the DNS in exactly the same way as normal domain names.

An ACE label can be easily identified because it begins with **XN--** (that is, X N followed by two hyphens). Nominet’s rules currently forbid registration of such names. The appendix contains more details, but basically an ACE label consists of the **XN--** prefix, the LDH characters from the original label, and then an encoding of the remaining characters. For example:

User enters:	www.nestlé.co.uk	www.êpô.org.uk
Application converts to:	www.xn--nestl-fsa.co.uk	www.xn--bdau3a.org.uk
Nominet registry entry:	xn--nestl-fsa (in co.uk)	xn--bdau3a (in org.uk)

Unicode

Unicode, otherwise known as ISO10646 (there are technical differences, but none that matter here) is an international standard character set which, it is intended, can encode *every* character in *every* writing system used anywhere in the world. It is described fully at the Unicode Consortium’s web site (<http://www.unicode.org>), but for these purposes all it is necessary to know is that it contains a large number of characters, each of which is given a code number (normally written in hexadecimal with a U+ prefix). For example:

H	Latin uppercase H	U+0048
ê	Latin lowercase e with circumflex	U+00EA
þ	Latin lowercase thorn	U+00FE
ج	Arabic letter jeem	U+062C
צ	Hebrew letter tsadi	U+05E6
€	Euro sign	U+20AC
♣	Black club suit	U+2663
И	Cyrillic small letter tse	U+0446
ケ	Katakana letter ke	U+30B1
𐀀	CJK unified ideograph 34AB ¹	U+34AB
𝄋	C clef	U+1D121

(note that the code number is always between 4 and 6 digits). Unicode also involves a number of subsidiary rules, some of which are made use of in IDNA.

¹ For anyone interested, this character is an antique form of another ideogram meaning “choked and unable to breathe”.

Issues for Nominet

The IDNA approach means that the core DNS is not affected by IDNA (except that any hashing or similar algorithms need to take account of the potentially large number of names with the same first four characters). Rather, any changes will be restricted to the support arrangements.

The first choice that will need to be made is whether to support IDNA at all. The options are:

1. Do not allow registrations beginning with **XN--**.
2. Allow registrations beginning with **XN--** but do not treat them specially in any way (additional support could be provided by tagholders, for example).
3. Allow IDNA registrations and provide some level of support infrastructure.

If Nominet are to provide support for IDNs, there are a number of possibilities:

- The automaton, whois web server, and so on could be limited to ACE labels, or they could accept some more readable representation of IDNs (for example, “[011D]”, “[g^]”, or even “[g circumflex]” for the character “ĝ”).
- The web site could provide a mechanism whereby an ACE label is entered and the corresponding IDN is displayed. Certificates, where provided, could also use both forms.

Secondly, while the IDNA specifications limit the characters that can be used in internationalized domain names, Nominet does not have to allow all of them. For example:

- Names could be limited to those characters described as “alphanumeric” by Unicode.
- Diacritical marks (accents and similar) could be forbidden².
- Characters could be limited to those in certain of the Unicode blocks (blocks are consecutive codes that all relate to one language or writing system; for example, Greek is covered by the blocks 0370-03FF (main) and 1F00-1FFF (polytonic)).

Finally, there are a number of issues that Nominet needs to consider:

- What are the implications of IDNA for the Dispute Resolution Service?
 - How will Experts determine whether two names in other character sets conflict?
 - What about names in one script that look like logos in other scripts?
- How will staff handle customer queries that involve IDNs, particular if the customer does not understand the concept of an ACE label?
- Does support for multiple character sets imply the need to provide support in other languages?
- Are there legal issues to be considered? Is support for Welsh, or for various EU languages, required? Is it unlawful discrimination to support some character sets but not others (e.g. Urdu but not Arabic)?

² One way to forbid diacritical marks would be to apply Unicode “Normalization Form KD” to the name – among other things, this splits composite characters into pieces, so “é” is split into “e” and a floating acute – then forbid the name if it contains a “composing” character.

However any restriction is implemented, the important point for this paper is that it is possible to write unambiguous technical specifications for many restrictions, and only such should be allowed.

APPENDIX A

The structure of an IDN

A domain name is made up of labels, separated by dots. An IDN is thus made up of a mix of LDH labels and non-LDH labels. The latter are then converted to ACE labels for sending to the resolver, and converted back where necessary. Labels containing only LDH characters are *not* altered.

An ACE label has the following structure:

- If the original label contains no LDH characters, then the ACE label has two parts:
 1. the **XN--** prefix;
 2. an encoding of each non-LDH character in the label, in Unicode code number order.
- If the original label contains one or more LDH characters, then the ACE label has four parts:
 1. the **XN--** prefix;
 2. all the LDH characters in the label, in the same order as the original label;
 3. a hyphen
 4. an encoding of each non-LDH character in the label, in Unicode code number order.

The encoding of each non-LDH character is formed of one or more letters and digits (but not hyphens). The encoding depends on where the character fits within the label, how many LDH characters there are in the label (but not what they are), and on the encodings preceding it in the ACE label; the details are in RFC3492.

Note that not every string beginning with the **XN--** prefix is an ACE label; it might not decode correctly, the non-LDH part might encode LDH characters, spaces, or dots, or it might contain characters which IDNA does not accept in domain names.

Examples of IDN and ACE labels:

<u>IDN label</u>	<u>ACE label</u>
test	<i>no ACE label</i>
tést	xn--tst-bma
têst	xn--tst-fma
tèst	xn--tst-61a
tèss	xn--tss-61a
très	xn--trs-71a
tèrès	xn--trs-61ab
térès	xn--trs-61ag
térès	xn--trs-71ad
êpô	xn--bdau3a
êêpô	xn--bdaa4a1b
bøast	xn--bast-gra
bärast	xn--brast-gra
børast	xn--brast-vua
êpôs	xn--s-cga4a1b
sêpô	xn--s-dga4a1b

APPENDIX B

Restrictions and pre-processing of IDNs

RFCs 3454 and 3491 contain a number of restrictions on characters in IDNs and specify various bits of pre-processing that are carried out to increase the likelihood that IDNA operates in a way that would make sense for typical users. This appendix summarises these issues.

Characters in IDNA are limited to those in Unicode version 3.2 and not any later version. At the time of writing this paper, the current version is 4.0.1, which contains no new characters since 4.0. The latter contains 1226 characters not in version 3.2.

A number of Unicode formatting characters, such as zero width space and Mongolian free variation selectors, are deleted from the IDN label before conversion to an ACE label, and therefore will not appear when converted back.

Because DNS is usually treated as case-insensitive, characters that could be viewed as “uppercase” are converted to their lowercase versions. Other related mappings are also applied; for example, German “ß” is converted to “ss” and Greek letters with oxia are converted to those with tonos. The label then undergoes “normalization form KC”; this is a process that converts different characters with the same general appearance into a canonical form. For example, the “%” symbol is converted to the three characters “c/o” and the script “ℓ” is converted to the ordinary letter “l”. A combination of these processes also ensures that, for example, the Angstrom sign “Å” (which Unicode views as different from the Latin letter A with ring) is converted to “ä”.

IDNA forbids a number of characters from appearing in IDN labels; these can be summarised as:

- space characters;
- control characters and other characters with control functions (such as Arabic end of Ayah and “invisible times”, used to show an implicit multiplication);
- private-use characters;
- code points guaranteed never to be allocated (e.g. U+FFFF);
- Unicode “surrogates” used to represent codes above U+FFFF in 16 bits;
- codes that are inappropriate in a canonicalized text string, such as “object replacement”;
- codes used to alter the layout of a string, such as “inhibit Arabic form shaping”;
- codes used for invisible language tagging.

IDNA allows an application to decide whether or not to forbid the ASCII characters not in the LDH set, such as “/” and “\$” (known as the “STD3 test”).

If any forbidden code appears in a label, it is invalid for IDNA purposes. This test takes place after the various conversions; thus, for example, “%” is forbidden if the STD3 test is applied.

While most languages are written from left-to-right, some languages – such as Hebrew and Arabic – are normally written from right-to-left. This leads to some interesting situations where the two are mixed. IDNA avoids this by requiring that all labels contain only characters with one direction; furthermore, if a name contains right-to-left characters, it must begin and end with them (digits do not have an inherent direction, so domain names written in Hebrew can contain digits but must not begin or end with them, whereas names written in Cyrillic can do). Note that characters are encoded in “logical” order, so $\varkappa\psi\aleph$ is encoded as $\mathbf{xn---\langle encoded\ \varkappa\rangle\langle encoded\ \psi\rangle\langle encoded\ \aleph\rangle}$.