IDN Self-Certification and instructions

Submitted to ICANN by: <Registry Operator>

String (A-label): <TLD>

Template Version: 1.1.0, Date: 2017-07-27

Registry Operator’s Revision Date: <day month year>

About this document

This document provides a template and instructions for the IDN self-certification for the Registry System Testing (“RST”). The document is provided in different formats, including pdf.

The IDN Self-Certification Document created by the Registry Operator shall follow the structure, headings and numbering given in this document, and must be submitted in pdf-format.

The document shall consist of the Registry Operator´s statements and certification that the requirements are fulfilled. Unless otherwise stated, the basis for the information can, in many cases, be decided by the Registry Operator.

The IDN Self-Certification Document shall comply with
- the requirements stated in the gTLD Applicant Guidebook, AGB.
- the requirements stated in the Registry Agreement including, but not limited to, Exhibit A and the specifications 2,4,6 and 10.

The Registry Operator shall provide the information described below.

1. Support for IDN

Instruction:
Does Exhibit A of the Registry Agreement for this TLD authorize support for IDN?

[ ] Yes

[ ] No

If you answered yes to this question please fill out the rest of this document. If you answered no to this question you do not need to fill out the rest of this document and you do not need to submit any other documents for the IDN portion of the RST.

Please note that you should submit this document in either case.

2. IDN support at start of “General Registration”

Instruction:
Do you intend to support IDN registration at the start of "General Registration" (as defined in the most recent version of the Rights Protection Mechanisms (RPM) Requirements Document at http://newgtlds.icann.org/en/about/trademark-clearinghouse/)?

[ ] Yes

[ ] No

If you answered yes to this question please identify which of the IDN tables listed in, or required to meet the terms of, Exhibit A of your Registry Agreement you will support at the start of General Registration.

The RST will verify that EPP requests for the registration of labels taken from each of the IDN tables you indicate here are processed consistently with the other documentation required for the IDN level of the RST. Please note that all IDN tables listed in or implied by Exhibit A of your Registry Agreement will be subject to the IDN level tests that do not have intrinsic EPP components.

Registry Operator self-certification:

IDN tables supported at the start of "General Registration":

IDN tables (languages/scripts) not supported at the start of "General Registration":

3. IDN tables

Each IDN table corresponding to the terms of Exhibit A in your Registry Agreement must be submitted to RST for validation. The Test Case IDNvalid00 states that all non-ASCII text(s) must be encoded in Unicode UTF-8. IDN table(s) are required to comply with the format specified in RFC 7940 (LGR format) or with text formats as specified in RFC 4290 and RFC 3743, respectively.

For detailed instructions, see appendix A of this document.

Registry Operator self-certification:

The table format of the IDN table(s) has been constructed using the guidelines in:

[ ] RFC 7940
[ ] RFC 4290
[ ] RFC 3743

Any comments on the IDN table formats:

Registry Operator checklist:

[ ] Each IDN table which includes non-ASCII text is encoded in UTF-8.

[ ] Each IDN table complies with the table format of RFC 7940, RFC 4290, or RFC 3743.

[ ] Each IDN table detailed in, or implicitly required by, Exhibit A of your Registry Agreement has been uploaded.

4. IDN EPP extensions

`The RST must have access to all EPP extensions needed for the registration of labels culled from an IDN table.`

For detailed instructions, see appendix B of this document.

Registry Operator self-certification (IDN EPP Extension):

For all IDN tables listed in Section 1 of the IDN Self-Certification Document and listed in or implied by Exhibit A of your Registry Agreement, list the corresponding EPPtags (if applicable) and make sure that there are no orphaned extensions or IDN tables.

See example below.

Registry Operator self-certification:

Example:

IDN table	**EPP tag**
French	**fr**
Italian	**it**
German	**de**
Portuguese	**pt**
Spanish	**es**
Cyrillic	**cyrl**

Registry Operator checklist:

[ ] All EPP tags are listed and correspond to submitted IDN tables.

[ ] All IDN EPP extensions needed to register an IDN label has been included/uploaded.

5. IDN policy

The RST requires the submission of a "policy statement", which is a summary of responses to questions asked in the Applicant Guidebook necessary for the understanding of a submitted IDN table.

For IDN tables in LGR format (RFC 7940) all information that can be encode (variant rules and contextual rules) must be encoded.

For detailed instructions, see appendix C of this document.

<u>Registry Operator self-certification:</u>

<u>Processing of registration requests for IDN labels</u>

<u>Use of multiple scripts in a single label</u>

<u>Variant management</u>

Contextual rules

Registry Operator checklist:

The policy information included in the IDN Self-Certification Document section 4 unambiguously indicates:

[ ] How requests for registration of IDN labels will be processed.
[ ] How the Registry handles comingling of scripts.
[ ] How the Registry handles variants.
[ ] How the Registry handles contextual rules.

Appendix A: IDN Table formats

Instruction:
The IDN table(s) must conform to one of the three prescribed IDN table formats. Below are table format information and examples that will be helpful in producing and submitting the IDN table(s).

Information about IDN table format following RFC 7940 (LGR):

The LGR format is a well-defined and feature rich format. It is an XML format, and its basic compliance with the RFC can be tested against the XML schema that is found with other RST XML schemas. Please note that a test against the schema will not capture all potential errors.

Information about IDN table format following the guidelines in RFC 4290:

Each code point in the IDN table appears in the Unicode "U+ nnnn" notation. Comment lines in the IDN table are initiated by a NUMBER SIGN ("#", U+0023). Each non-comment line in the IDN table starts with the code point that is allowed in the registry and expected to be used in registrations, which is also called the "base character". If the base character has any variants, the indication of its code point is followed by a VERTICAL LINE ("|", U+007C) and the variant character. If the base character has more than one variant, the code points for the variants are separated by a COLON (":", U+003A). If a base character has a variant that is composed of a sequence of characters (strings) they are indicated with a HYPHEN-MINUS ("-", U+002D) between each code point.

Example of an IDN table in RFC 4290 format:

```
# Comment                       # this is a comment line
U+2200                          # valid code point
U+2201|U+0043                   # valid code point|variant
U+2202|U+0064:U+03B4            # valid code point|two variants for the same character
U+2237|U+003A-U+003A            # valid code point|variant is a sequence of characters
```

Information about IDN table format following the guidelines in RFC 3743:

Each IDN table must have a version number and its release date. This is tagged with the word "Version" followed by an integer then followed by the date in the format YYYYMMDD, where YYYY is the 4-digit year, MM is the 2-digit month, and DD is the 2-digit day of the publication date of the IDN table.

The format consist of three columns: valid code point, preferred variant, and character variant. The columns are separated by a SEMICOLON (";", U+003B). If the preferred variant or character variant is composed of a sequence of characters (string), the code points for the variants are separated by a SPACE (" ", U+0020). If there are multiple preferred variants or character variants they are separated by a COMMA (",", U+002C). Each code point optionally has a reference number listed in PARENTHESIS ("()", U+0028 U+0029) directly after it. This references the authoritative source for the code point entry, as listed in the beginning of the IDN table(s) file(s).

Each entry in a column consists of one or more code points, expressed numerically as given in the Unicode Code Chart and optionally followed by a parenthetical reference. The first column, or valid code point, may have only one code point specified in a given row. The other columns may have more than one. Any row may be terminated with an optional comment, starting with a NUMBER SIGN ("#", U+0023).

Valid code point: In a language variant IDN table, this is a list of code points that are permitted for that language. Any other code points, or any string containing them, will be rejected by this specification. The valid code point list appears as the first column of the language variant IDN table.

Preferred variant: In a language variant IDN table, this is a list of code points corresponding to each valid code point, providing possible substitutions for it. These substitutions are "preferred" in the sense that the variant labels generated using them are normally registered in the zone file, or "activated". The preferred code points appear in column 2 of the language variant IDN table.

Character variant: In a language variant IDN table, this is a second list of code points corresponding to each valid code point, providing possible substitutions for it. Unlike the preferred variants, substitutions based on character variants are normally reserved but not actually registered (or "activated"). Character variants appear in column 3 of the language variant IDN table.

Example of an IDN table in RFC 3743 format:

```
Reference 1                 # justification for code points 2200, 2201, and 2202
Reference 2                 # justification for code points 0043,2237, and 003A
Reference 3                 # justification for code point 03B4

Version 1 20020701    # July 2002 Version 1

2200(1)                     # valid code point
2201(1);0043(2)             # valid code point; preferred variant
2237(2);003A(2);03B4(3)     # valid code point; preferred variant; character variant
2202(1); 003A(2) 003A(2)    # valid code point; preferred variant is a sequence of characters
```

Example of an IDN table in RFC 3743 format without the optional references:

```
Version 1 20020701    # July 2002 Version 1

2200                        # valid code point
2201;0043                   # valid code point; preferred variant
2237;003A;03B4              # valid code point; preferred variant; character variant
2202; 003A 003A             # valid code point; preferred variant is a sequence of characters
```

By general exception, code points may be listed with or without the Unicode "U+" prefix.

Appendix B: IDN EPP Extension

Instruction:
In order for the RST to perform IDN variant code point validation (if applicable) and IDN online registry response verification, a list is needed of all IDN EPP extensions, such as language and script tags, required to submit a request for the registration of an IDN label.

Below is an example of an EPP create command with an IDN language extension tag that will be helpful in producing and submitting the EPP extensions.

```
Example <create> command:

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
 <command>
   <create>
     <domain:create
       xmlns:domain="urn:ietf:params:xml:ns:domain-1.0">
     <domain:name>xn--espaol-zwa.example.com</domain:name>
     <domain:registrant>jd1234</domain:registrant>
     <domain:authInfo>
       <domain:pw>2fooBAR</domain:pw>
     </domain:authInfo>
     </domain:create>
   </create>
   <extension>
   <idn:data xmlns:idn="urn:ietf:params:xml:ns:idn-1.0">
     <idn:table>es</idn:table>
     <idn:uname>español.example.com</idn:uname>
   </idn:data>
   </extension>
   <clTRID>123456</clTRID>
 </command>
</epp>
```

Appendix C: IDN Policy

Instruction:
The reference documents for the RST do not go into rigorous detail about everything that a TLD registry must consider when establishing policies for the registration of IDN labels. The need for local decision on key issues is recognized by the IDNA protocol, which makes it mandatory for a registry to determine a subset of the full repertoire of valid code points that is appropriate to its target community. The ICANN IDN Guidelines place a number of constraints on that selection (most notably by prohibiting the mixing of scripts in an individual label without clear verifiable linguistic justification). Additional concerns are noted with the management of variant relationships among code points in an IDN table. These are also subject to broad general constraints with additional need for local judgment.

The IDN policy statement describes the outcome of these processes for a given TLD and is a fundamental requisite for the IDN level of the RST. Stated informally, the policy statement together with the IDN table(s) indicates the code points a registered label may include, and how a registrant can expect it to be processed. The RST verifies that a submitted IDN table and the associated policy statement conform to the reference specifications, that due consideration has been given to varying local needs, and that the registry behaves consistently with the expectations arising from the documentation.

The Registry Operator Checklist, below, lists the specific details that the RST will scrutinize. Each point corresponds to a heading in the following sequence of general remarks:

Processing of registration requests for IDN labels

The policy statement must include a complete list of the supported IDN tables, each identified by a language or script. When deciding which of these alternatives is preferable, it may be useful to note that the Unicode Standard, the IDNA protocol, and key elements of ICANN's IDN Guidelines and variant management requirements are script based. The IDNA protocol determines the status of a code point by the Unicode script properties assigned to it and not the languages that include it in their writing systems; there are no Unicode language properties.

Basing an IDN table on script may therefore be more straightforward than basing it on language. The languages that a registry wishes to accommodate obviously must be taken into account when preparing a script IDN table. Emphasis on those languages will again be necessary in awareness and marketing efforts. There may be less reason to highlight language in technical contexts.

A single script IDN table can be explicitly collated to support a number of languages and may coincidentally support others. (The latter factor can also apply to a language IDN table.) The languages that a script IDN table deliberately covers should all be listed in the policy statement. If the IDN table is intended to cover a broader aggregate of languages, a collective designator for them may be given instead.

If no reference to language is made in a script IDN table, the undifferentiated listing of all IDNA-valid code points associated with the designated script (i.e. with the same “explicit Unicode script property value”) may require special justification, nonetheless. The blanket inclusion of all code points that Unicode does not uniquely associate with a single script (i.e. with the “special Unicode script property values” COMMON or INHERITED) will not be accepted.

If language tags or script tags are used in the registration process, the corresponding EPP extensions must be documented with an exhaustive list of the available tags and the syntax of their use.

Use of multiple scripts in a single label

In cases where a language's writing system requires the use of multiple scripts, the preparation of a language IDN table will be unavoidable. The RST will verify that such need exists by reference to established orthographic authorities for the indicated language. A registry may be asked to justify the inclusion of code points that cannot be validated in that manner, by citing orthographic authorities that the RST may not have used.

Variant management

IDN tables that indicate variant relationships among listed code points are subject to special terms in Exhibit A of the registry agreement with ICANN. The corresponding variant management policies should be detailed in the policy statement.

With reference to the alternate IDN table formats discussed in Section 2, above, it may be worth noting that RFC 3743 includes an approach to variant management that is primarily intended for use with ideographic labels (notwithstanding an initial expectation that it might also be applied in alphabetic contextsl). Individual ideographs are semantically laden, and the basic concept of variant relationships among them is tied to those semantics. That aspect of variance is irrelevant in the alphabetic realm, where individual characters rarely have any such semantic value.

When using the format in RFC 3743 without intent to invoke its approach to variant management, care should be taken to indicate this in the policy statement. Additional detail will include the local approach to reserving, bundling, and blocking clusters of labels that differ in terms of variant code points.

When using LGR format (RFC 7940) then all possible variant rules can be encoded.

Contextual rules

Particular note should be made of the contextual rules given in Appendix D of RFC 5892 (which lists the IDNA status of all code points in the Unicode Chart), and Section 2 of RFC 5893 (dealing with bidirectional domain names and labels in scripts that are written from right to left). The RST will verify that these rules are enforced for tabulated code points to which they apply. Any similar contextual restrictions imposed directly by the registry should be fully documented, together with all other local processing detail relevant to the registration of an IDN label. Appendix D of this document lists code points associated with a requirement for contextual information including an example of minimum content description to be included in a policy statement.

Appendix D: Contextual rules

This list includes the code points on which the IDNA protocol (RFCs) currently explicitly places contextual restrictions.

Please note that this is not a complete listing of all code points that require contextual rules. There are additional restrictions imposed by the Unicode database.

The normative references for the code points listed here are the RFCs. The listing here is given solely for illustrative purposes.

From RFC 5891:

- HYPHEN-MINUS (U+002D). Must not be label initial or label final. Must not be in posisions three *and* four.
- Code points belonging to general category "mark" (nonspacing mark, spacing mark or enclosing mark) in the Unicode database. Must not be in label initial position.

From RFC 5892:

- ZERO WIDTH NON-JOINER (U+200C). This may occur in a formally cursive script (such as Arabic) in a context where it breaks a cursive connection as required for orthographic rules as, for example, in the Persian language. It also may occur in Indic scripts in a consonant-conjunct context (immediately following a virama), to control required display of such conjuncts.

- ZERO WIDTH JOINER (U+200D). This may occur in Indic scripts in a consonant-conjunct context (immediately following a virama), to control required display of such conjuncts.

- MIDDLE DOT (U+00B7) is used to permit the Catalan character ela geminada to be expressed. Must be preceded and followed by “l” (U+006C).

- GREEK LOWER NUMERAL SIGN, KERAIA (U+0375) is only permitted in Greek script.

- HEBREW PUNCTUATION GERESH (U+05F3) must be preceded by a Hebrew script character.

- HEBREW PUNCTUATION GERSHAYIM (U+05F4) must be preceded by a Hebrew script character.

- KATAKANA MIDDLE DOT (U+30FB) is only permitted in a label if at least one other character in the label is a code point from Hiragana, Katakana or Han script.

- ARABIC-INDIC DIGITS (0660..0669) can not be mixed with EXTENDED ARABIC-INDIC DIGITS (06F0..06F9).

From RFC 5893:

- ARABIC-INDIC DIGITS (0660..0669) cannot be mixed with European DIGITS (0030..0039) in RTL labels.

- ARABIC-INDIC DIGITS (0660..0669), EXTENDED ARABIC-INDIC DIGITS (06F0..06F9), or European DIGITS (0030..0039), cannot start an RTL label.