



New Generic Top-Level  
**Domains**

# **IDN Self-Certification and instructions**

**Submitted to ICANN by: <Applicant>**

**String: <TLD>**

**Application ID: <ID>**

**Template Version: 1.0.7, Date: 20 December 2013**

**Applicants Revision Date: <day month year>**

## **About this document**

This document provides a template and instructions for the IDN self-certification for the Pre-Delegation Testing, ("PDT"). The document is provided in different formats, including pdf.

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

The document shall consist of the applicant'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 applicant.

If the applicant has previously submitted self-certification material to the PDT in a format differing from the one described here, future submissions may also be made in that format. Any changes to it, as well as all submissions from initial participants, should follow the instructions provided here.

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 applicant shall provide the information described below.

## 1. 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 PDT will verify that EPP requests for the registration of labels taken from each of the tables you indicate here are processed consistently with the other documentation required for the IDN level of the PDT. 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.

**Applicant self-certification:**

## 2. IDN tables

Each IDN table corresponding to the terms of Exhibit A in your Registry Agreement must be submitted to PDT 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 formats of RFC 4290, RFC 3743 or must be accompanied with justification as to why those formats could not be used.

**Instruction:** The IDN table(s) must conform to one of the two prescribed reference formats, or if it does not, the way in which the alternative is to be understood must be unambiguously apparent. If the RFC 3743 format is used, by general exception, code points may be listed with or without the Unicode "U+" prefix. 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 the guidelines in RFC 4290:

Each code point in the table appears in the Unicode "U+ nnnn" notation. Comment lines in the table are initiated by a NUMBER SIGN ("#", U+0023). Each non-comment line in the 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 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 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 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 table.

**Preferred variant:** In a language variant 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 table.

**Character variant:** In a language variant 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 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
```

**Applicant self-certification:**

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

- ☐ RFC 4290
- ☐ RFC 3743
- ☐ I have used neither and have included a justification below

Description of format and justification:

**Applicant checklist:**

- ☐ Each IDN table which includes non-ASCII text is encoded in UTF-8.
- ☐ Each IDN table complies with the table format of RFC 4290 or RFC 3743.
- ☐ Each IDN table not in RFC 4290 or RFC 3743 format is accompanied with a detailed format definition and a reasonable motivation why the standard formats cannot be used.
- ☐ Each IDN table detailed in, or implicitly required by, Exhibit A of your Registry Agreement has been uploaded.

### 3. IDN EPP extensions

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

**Instruction:** In order for the PDT 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:registrar>jdl234</domain:registrar>
        <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>
```

**Applicant 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 tables.

See example below.

**Applicant self-certification:**

**Example:**

| <b>IDN table</b>  | <b>EPP tag</b> |
|-------------------|----------------|
| <b>French</b>     | <b>fr</b>      |
| <b>Italian</b>    | <b>it</b>      |
| <b>German</b>     | <b>de</b>      |
| <b>Portuguese</b> | <b>pt</b>      |
| <b>Spanish</b>    | <b>es</b>      |
| <b>Cyrillic</b>   | <b>cyrl</b>    |

**Applicant 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.

## 4. IDN policy

The PDT 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.

### **Instruction:**

The reference documents for the PDT 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 PDT. 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 PDT 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 Applicant Checklist, below, lists the specific details that the PDT 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 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 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 table.) The

languages that a script table deliberately covers should all be listed in the policy statement. If the 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 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 table will be unavoidable. The PDT 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 PDT 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 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 contexts). 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.

### Contextual rules

Particular note should be made of the contextual rules given in Appendix A of RFC 5892 (which lists the IDNA status of all code points in the Unicode Chart). The PDT will verify that these rules are enforced for all 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 A 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.

***Applicant self-certification:***

Processing of registration requests for IDN labels

Use of multiple scripts in a single label

Variant management

Contextual rules

**Applicant 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

**This list includes all code points on which the IDNA protocol currently places contextual restrictions. It is, however, is non-normative, and is included here solely for illustrative purposes.**

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

© Internet Corporation For Assigned Names and Numbers.