L2/20-164

To: Unicode Technical Committee
From: Manish Goregaokar <manish@mozilla.com>
Re: `Identifier_Status` of `Limited_Use` scripts in UTS # 39
Date: July 7, 2020

The Unicode Technical Standard for Unicode Security Mechanisms [UTS 39] provides a categorization from `Identifier_Type` to `Identifier_Status` in [UTS 39, Table 1 *Identifier_Status and Identifier_Type*], in an attempt to suggest which code points to allow in identifiers being screened for security reasons.

Amongst this classification, `Identifier_Type = Limited_Use` code points are classified as `Identifier_Status = Restricted`, i.e. a default implementation of the [UTS 39, 3.1 *General Security Profile*] will not allow these code points.

The definition of `Limited_Use` comes from Unicode Identifier and Pattern Syntax [UAX 31], where they are defined as "Modern scripts that are in more limited use" [UAX 31, Table 7 *Limited_Use Scripts*]. They include scripts such as Javanese, Balinese, Syriac, and Cherokee.

The recommendations in [UTS 39] are not set in stone, rather, the standard mentions that implementations can make exceptions, and goes as far as to specifically suggest `Limited_Use` scripts as an example of an exception. So while the specification is not mandating anything, there is something to be said for reasonable defaults. It seems to me that it might be better to include `Limited_Use` as `Allowed` by default, with caveats saying that implementations may wish to exclude it, rather than the other way around.

1 Usage of UTS #39

[UTS 39] is used in the wild in many contexts.

One major context is determining when to display a URL as "punycode". To prevent spoofing, browsers choose to display non-ASCII URLs in their `ToAscii` form [RFC 3490] based on various heuristics. For example, Chrome [Chrome IDN] applies checks from [UTS 39, 3.1 *General Security Profile*] [UTS 39, 5.2 *Restriction-Level detection*], [UTS 39, 5.3 *Mixed-number detection*], [UTS 39, 4.2 *Mixed-script Confusables*], as well as some custom heuristics. Firefox does something similar.

Programming languages also use this. The Rust RFC for non-ASCII identifiers [Rust RFC 2457] uses a set of heuristics similar to that of Chrome to emit warnings about suspicious identifiers.

It's not clear to me that `Limited_Use` scripts need to be disadvantaged in either of these sets of use cases.

2 The digital presence of the `Limited_Use` scripts

Many of the `Limited_Use` scripts have a growing digital presence. There are Wikipedias in Aramaic (Syriac), Inuktitut (Canadian Syllabics), Cherokee, N'ko, and Santali (Ol Chiki).

The Javanese Wikipedia has some articles using both the Latin and Javanese scripts, and similarly the Balinese Wikipedia has similar dual pages in Balinese and Latin.

The official Indonesian registrar PANDI is currently registering Javanese domain names like http://ꦮꦶꦏꦶꦥꦼꦢꦶꦪ.id, and hopes to continue to do so as the script's digital foothold grows. This proposal arises in part due to PANDI's complaint about punycoded Javanese domain names. A comment from their representative Chika Hs on Bugzilla talks about this problem:

> currently, domain names in javanese is a bit of a novelty. However, a small yet robust and growing community of Javanese script user is seen today. One of the stumbling blocks they face is softwares and other digital interface that does not support Javanese adequately, and its making it a bit difficult to show new users on the applicability of this script. at the very least, the ability for the script to be displayed on screen is very much appreciated, and PANDI can use this to encourage other bodies that errors in digital implementation is a bit complicated and involve outside bodies, but entirely fixable and not something that "oh its the system, we cant fix it, we just have to let it be.

3 Proposal

While it is possible to handle this on a case by case basis (indeed, I have filed issues on Firefox and Chromium), I feel like it would be best if our defaults did not disadvantage scripts that are already facing many barriers to digital adoption.

Every entry under `Identifier_Status=Restricted` in [UTS 39, Table 1 *Identifier_Status and Identifier_Type*] has a straightforward reason why it should be `Restricted`, except for `Identifier_Type=Limited_Use`. The specification does mention "characters from these scripts have not been a priority for examination for confusables or to determine specialized, non-modern, or uncommon-use characters.", however this seems to be more of a reduction of attack surface than anything else, and to me this seems like a decision that the implementor is better suited to make.

Given that, I feel like `Identifier_Type=Limited_Use` should be classified as `Identifier_Status=Allowed`, with a prominent note suggesting that implementations worried about lack of confusable data may wish to exclude these characters.

The proposal is to make the following changes to [UTS 39, 3.1 *General Security Profile*]:

The Restricted characters are characters not in common use, and they can be blocked to further reduce the possibilities for visual confusion. They include the following:

- characters not in modern use
- characters only used in specialized fields, such as liturgical characters, phonetic letters, and mathematical letter-like symbols
- ~~characters in limited use by very small communities~~

The principle has been to be more conservative initially, allowing for the set to be modified in the future as requirements for characters are refined. For information on handling modifications over time, see *Section 2.9.1, Backward Compatibility* in *Unicode Technical Report #36, "Unicode Security Considerations"* [UTR36].

An implementation following the General Security Profile does not permit any characters in \p{Identifier_Status=Restricted}, unless it documents the additional characters that it does allow. Such documentation can specify characters via properties, such as \p{Identifier_Status=Technical}, or by explicit lists, or by combinations of these. Implementations may also specify that fewer characters are allowed than implied by \p{Identifier_Status=Restricted}; for example, they can restrict characters to only those permitted by [IDNA2008].

~~Common candidates for such additions include characters for scripts listed in *Table 7, Limited Use Scripts* of [UAX31]. However, characters from these scripts have not been a priority for examination for confusables or to determine specialized, non-modern, or uncommon-use characters.~~

A potential candidate for such additional restrictions include characters for scripts listed in *Table 7, Limited Use Scripts* of [UAX31], since characters from these scripts have not been a priority for examination for confusables or to determine specialized, non-modern, or uncommon-use characters.

Canonical equivalence is applied when testing candidate identifiers for inclusion of *Allowed* characters. For example, suppose the candidate string is the sequence

<u, *combining-diaeresis*>

The target string would be Allowed in *either* of the following 2 situations:

1. u is Allowed and ¨ is Allowed, or
2. ü is Allowed

For details of the format for the [idmod] files, see *Section 7, Data Files*.

Table 1. Identifier_Status and Identifier_Type

Identifier_Status	Identifier_Type	Description
Restricted	Not_Character	Unassigned characters, private use characters, surrogates, non-whitespace control characters.
	Deprecated	Characters with the Unicode property *Deprecated=Yes*.
	Default_Ignorable	Characters with the Unicode property *Default_Ignorable_Code_Point=Yes*, except U+200C ZWNJ and U+200D ZWJ.
	Not_NFKC	Characters that cannot occur in strings normalized to NFKC.
	Not_XID	Characters that do not qualify as default Unicode identifiers; that is, they do not have the Unicode property *XID_Continue=True*.
	Exclusion	Characters with Script_Extensions values containing a script in *Table 4, Candidate Characters for Exclusion from Identifiers* from [UAX31], and no explicit script from *Table 7, Limited Use Scripts* or *Table 5, Recommended Scripts*.
	Obsolete	Characters that are no longer in modern use.
	Technical	Specialized usage: technical, liturgical, etc.
	Uncommon_Use	Characters whose status is uncertain, or that are not commonly used in modern text.
	~~Limited_Use~~	~~Characters from scripts that are in limited use: with Script_Extensions values containing a script in *Table 7, Limited Use Scripts* in [UAX31], and no explicit script from *Table 5, Recommended Scripts*.~~
Allowed	**Inclusion**	Exceptionally allowed characters, including *Table 3a, Optional Characters for Medial* and *Table 3b, Optional Characters for Continue* in [UAX31], and some characters for [IDNA2008], except for certain characters that are Restricted above.
	Limited_Use	Characters from scripts that are in limited use: with Script_Extensions values containing a script in *Table 7, Limited Use Scripts* in [UAX31], and no explicit script from *Table 5, Recommended Scripts*. Note that these may not have been thoroughly vetted for confusables or to determine specialized, non-modern, or uncommon-use characters, and implementations may wish to exclude these from the **Recommended** category if this is not ideal for their use case.
	Recommended	Characters with Script_Extensions values containing an explicit script in *Table 5, Recommended Scripts* in [UAX31], except for those characters that are Restricted above.

References

[Chrome IDN] The Chromium Project Developers. *IDN in Google Chrome*. Tech. rep. The Chromium Project.

[RFC 3490] P Faltstrom, P Hoffman, and A Costello. "RFC 3490". In: *Internationalizing Domain Names in Applications (IDNA)* (2003).

[Rust RFC 2457] Pyfisch and Manish Goregaokar. *Rust RFC 2457: Non-Ascii Idents*. Tech. rep. The Rust Project.

[UAX 31] Mark Davis. *Unicode Identifier and Pattern Syntax. Unicode Standard Annex 31*. Tech. rep. Unicode Consortium, 2020.

[UTS 39] Mark Davis and Michel Suignard. *Unicode Security Mechanisms. Unicode Technical Standard 39*. Tech. rep. Technical report, Unicode Consortium, 2020.