277	**Reconciling Script and Script_Extensions Character Properties**	**Closing Date: 2014.07.28**
Status:	Open	
Originator:	UTC	
Informal Discussion:	Unicode Mail List (Join)	
Formal Feedback:	Contact Form	

Description of Issue:

There are currently a small number of characters whose Script value is explicit (neither Common nor Inherited) and whose Script_Extensions value set has more than one value (a “diverse” value set). An example is U+FDF2 ARABIC LIGATURE ALLAH ISOLATED FORM, which has Script=Arabic, and Script_Extensions={Arabic Thaana}. These characters are not typical; most characters with a diverse Script_Extensions value set have a Script value of either Common or Inherited.

The Unicode Standard provides no principle for this: the character’s Script value may be an explicit script although there is a diverse Script_Extensions value set, but it is not documented what that means, and why it is not Common or Inherited. There is a cost to this anomaly in terms of usability and understandability. By giving users of Unicode data no clue as to when or why this is done, there is no value provided for that cost.

The Unicode Technical Committee would like to eliminate the ambiguity, and move to one of the first two following policies, **A** or **B**. It would appreciate feedback as to the preferred approach.

Where a character’s Script_Extensions value set has more than one element:

Policy A. The character’s Script value must *not* be explicit.

- This would require a change for 30+ characters.
- **Advantage:** Slightly easier for API usage, since implementations need only lookup extra Script_Extensions info for Common or Inherited characters, not for every single character.
- **Disadvantage:** Implementations that don’t use Script_Extensions may do worse in resolving script runs for highly-used scripts. For example, BENGALI DIGIT characters would resolve as Common, when they are most likely in the Bengali script.

There are exactly 2 states:

1. Script_Extensions = {Script}
2. Script_Extensions ⊅ {Script} & Script is !explicit

Examples:

1. scx={Common} & sc=Common; scx={Arabic} & sc=Arabic
2. scx={Arabic, Syriac} & sc=Common

Policy B. The character's Script value must not be explicit, *except* where that script is a reasonable default value.

- An example of such a default is where that default script accounts for the vast majority of usage. This would not require changes to the data, but the committee could consider changes.
- **Advantage:** For implementations that don't use Script Extensions, better results could obtain for the majority of cases. For example, a string containing U+0660 (٠) ARABIC-INDIC DIGIT ZERO and some Common symbols would be presumed to be Arabic by such an implementation.
- **Disadvantage:** Implementations that don't use Script_Extensions may break up script runs in lesser-used scripts. For example, a sequence of Chakma characters containing a BENGALI DIGIT ZERO (which is used for digits in Chakma) would be broken into 3 runs. The UTC would also need to be in the business of selecting the "best" default, which may not be the best for any particular implementation.

There is one more state (3):

1. Script_Extensions = {Script}
2. Script_Extensions ⊅ {Script} & Script is !explicit
3. **Script_Extensions ⊋ {Script} & Script is explicit**

Example:

1. scx={Common} & sc=Common; scx={Arabic} & sc=Arabic
2. scx={Arabic, Syriac} & sc=Common
3. **scx={Arabic, Syriac} & sc=Arabic**

Note: If the committee adopts Policy A, any implementations could support the effect of Policy B with its own data, such as processing Script_Extensions to choose the Recommended Script from that value set, if there is exactly one; or picking among the scripts of the implementation-supported languages.

How to Provide Feedback: For information about how to discuss this Public Review Issue and how to supply formal feedback, please see the feedback and discussion instructions. The accumulated feedback received so far on this issue is shown below, or you can look at a full page view. Feedback is reviewed by the relevant committee according to their meeting schedule.