8 TrueType Open Tag Registry

The Tag Registry defines the TrueType Open tags that Microsoft supports. TrueType Open tags are 4-byte character strings that identify the scripts, language systems, baselines, and features in a TrueType Open font. The registry establishes conventions for naming and using these tags. Registered tags have a specific meaning and convey precise information to developers and text-processing clients of TrueType Open. Microsoft encourages font developers to use registered tags to assure compatibility and ease of use across fonts, applications, and operating systems.

This chapter contains sample sets of commonly used tags for scripts, language systems, and baselines. Microsoft will supply a list of additional tags upon request.

In addition, the Feature Tag section defines all the features and feature tags Microsoft has developed and registered to date. This section includes a description of the function of each feature.

Microsoft expects the list of registered tags and features to expand over time. The most recent version of the registry will be available on Microsoft's ftp and World Wide Web sites, as well as on the quarterly Microsoft Developers Network CD.

Microsoft welcomes nominations for new and useful features to register. For more information about feature and feature tag registration, see the Feature Tag section.

Script Tags

Script tags identify the scripts represented in a TrueType Open font. Script tags are defined by Microsoft Typography and correspond to the contiguous character code ranges in Unicode.

All tags are 4-byte character strings composed of a limited set of ASCII characters in the 0x20-0x7E range. Each script tag consists of four lowercase letters.

Some of most commonly used script tags are shown below. A full list of script tags is available from Microsoft.

Script Tag	Script
"arab"	Arabic
"cyrl"	Cyrillic
"grek"	Greek
"hani"	Han Ideographic
"hebr"	Hebrew
"kana"	Kana (Hiragana & Katakana)
"hang"	Korean Hangeul
"latn"	Latin
"thai"	Thai

Language System Tags

Language system tags identify the language systems supported in a TrueType Open font. Microsoft uses the standard language system tag names defined in the Windows Natural Language Support API document (called NLSAPI.doc), in Appendix A: Locales and Language ID's. This document is available on the Microsoft Developers Network CD released by Microsoft quarterly, or it can be acquired directly from Microsoft Typography.

All tags are 4-byte character strings composed of a limited set of ASCII characters in the 0x20-0x7E range. The following list shows sample language system tags from the NLSAPI document. Each language system tag consists of three uppercase letters, followed by a single space. This space character is a single byte, 0x20; it is not equal to NULL.

Language System Tag	Language System
“ARA ”	Arabic (Saudi Arabia)
“ZHT ”	Chinese (Taiwan)
“ZHS ”	Chinese (PRC)
“ENU ”	English (United States)
“ENG ”	English (British)
“FRA ”	French (Standard)
“DEU ”	German (Standard)
“IWR ”	Hebrew
“JAN ”	Japanese
“KOR ”	Korean
“ESP ”	Spanish (Traditional Sort)
“THA ”	Thai

Baseline Tags

This section defines the standard TrueType Open baseline tags that Microsoft supports. A registered baseline tag has a specific meaning and conveys information to font users about a baseline’s use. For example, the “romn” baseline tag is commonly used to identify the baseline for Latin text layout. For compatibility and ease of use, Microsoft encourages font developers to use registered baseline tags.

This version of the Tag Registry identifies the baselines that Microsoft has implemented to date. All tags are 4-byte character strings composed of a limited set of ASCII characters in the 0x20-0x7E range. Baseline tags consist of four lowercase letters.

Baseline Tag	Baseline
“ideo”	Ideographic
“hang”	Indic hanging
“math”	Mathematical (centered)
“romn”	Roman

Feature Tags

Features provide information about how to use the glyphs in a font to render a script or language. For example, an Arabic font might have a feature for substituting initial glyph forms, and a Kanji font might have a feature for positioning glyphs vertically. All TrueType Open features define data for glyph substitution, glyph positioning, or both.

Each TrueType Open feature has a feature tag that identifies its typographic function and effects. By examining a feature's tag, a text-processing client can determine what a feature does and decide whether to implement it. All tags are 4-byte character strings composed of a limited set of ASCII characters in the 0x20-0x7E range. Microsoft-registered feature tags use four lowercase letters. For instance, the "mark" feature manages the placement of diacritical marks, and the "swsh" feature renders swash glyphs.

This version of the Tag Registry describes all the TrueType Open features Microsoft has developed to date. It also includes details that identify the lookups that Microsoft uses to implement each feature. Lookup information is provided for reference purposes only; the set of lookups used to implement a feature will vary across system platforms, applications, fonts, and font developers.

A feature definition may not provide all the information required to properly implement glyph substitution or positioning actions. In many cases, a text-processing client may need to supply additional data. For example, the function of the "init" feature is to provide initial glyph forms. Nothing in the feature's lookup tables indicates when or where to apply this feature during text processing. To correctly use the "init" feature in Arabic text where initial glyph forms appear at the beginning of words, text-processing clients must be able to identify the first glyph position in each word before making the glyph substitution. In all cases, the text-processing client is responsible for applying, combining, and arbitrating among features and rendering the result.

The tag space defined by tags consisting of four uppercase letters (A-Z) with no punctuation, spaces, or numbers, is reserved as a vendor space. Font vendors may use such tags to identify private features. For example, the feature tag "PKRN" might designate a private feature that may be used to kern punctuation marks. Microsoft does not guarantee the compatibility or usability of private features, and it cannot ensure that two font vendors will not choose the same tag for a private feature. Microsoft does guarantee that it will not register or publish definitions or tags of private features.

To Register Features

Microsoft encourages font developers to use registered feature tags when implementing registered features. However, font developers also may define and register their own features.

Microsoft welcomes nominations for new features and feature tags to register. To qualify for registration, a feature must have a single function that is clearly identified by its tag. The function of the feature should be defined at the lowest useful level and must be distinctly different from the functions of currently registered features. When font developers register feature tags and functions with Microsoft, they do not have to supply implementation details.

Microsoft reserves the right to officially assign feature tags in the Microsoft Tag Registry. Although Microsoft has reserved the feature and feature tag definitions listed here, Microsoft fonts do not contain all of the features.

Registered Features

Tag: **"init"**
Function: Provides initial glyph forms
Microsoft Implementation:
In Microsoft's Arabic fonts, the "init" feature replaces a default glyph located at the start of a word with an initial form of the glyph. Microsoft implements the feature as a single substitution lookup (GSUB LookupType = 1) for the Arabic script.
Tag: **"medi"**
Function: Provides medial glyph forms
Microsoft Implementation
In Microsoft's Arabic fonts, the "medi" feature replaces a default glyph located in the middle of a word with a medial glyph form of the glyph. Microsoft implements the feature as a single substitution lookup (GSUB LookupType = 1) for the Arabic script.
Tag: **"fina"**
Function: Provides final glyph forms
Microsoft Implementation
In Microsoft's Arabic fonts, the "fina" feature replaces a default glyph located at the end of a word with a final form of the glyph. Microsoft implements the feature as a single substitution lookup (GSUB LookupType = 1) for the Arabic script.
Tag: **"liga"**
Function: Provides ligature glyphs
Microsoft Implementation
The "liga" feature replaces a string of glyphs with a ligature glyph. Microsoft defines this feature with a ligature substitution lookup (GSUB LookupType = 4). The "liga" feature also might be implemented as a contextual substitution that replaces a sequence of glyph classes or coverage tables with a ligature glyph.
Tag: **"vert"**
Function: Substitutes vertical glyph variants for vertical text layout
Microsoft Implementation
The "vert" feature replaces default glyphs with vertically oriented glyph forms. Microsoft implements this feature as a single substitution lookup (GSUB LookupType = 1).

Tag: **"mark"**
Function: Positions mark glyphs with respect to base glyphs
Microsoft Implementation
The "mark" feature positions mark glyphs in relation to a base glyph, a ligature glyph, or another mark glyph. Microsoft implements this feature as a MarkToBase Attachment lookup (GPOS LookupType = 4), MarkToLigature Attachment lookup (GPOS LookupType = 5), or MarkToMark Attachment lookup (GPOS LookupType = 6).
Tag: **"mset"**
Function: Positions Arabic combining marks in fonts for Windows 95 using glyph substitution
Microsoft Implementation
In contrast to the "mark" feature, "mset" uses glyph substitution to combine marks and base glyphs. It replaces a default mark glyph with a correctly positioned mark glyph. The font designer specifies the position of the mark when describing the mark's contour in the font file. Microsoft's Arabic fonts, created for Windows 95, use a contextual substitution lookup (GSUB LookupType = 5) to implement the "mset" feature.
Tag: **"smcp"**
Function: Provides small capital/uppercase glyphs, commonly called "small caps"
Microsoft Implementation
The "smcp" feature replaces lowercase glyphs with small uppercase glyphs. Microsoft implements this feature with a single substitution lookup (GSUB LookupType = 1).
Tag: **"onum"**
Function: Provides old style numerals
Microsoft Implementation
The "onum" feature replaces default numeral glyphs with glyphs of old style numerals. Microsoft implements the feature as a single substitution lookup (GSUB LookupType = 1).
Tag: **"swsh"**
Function: Provides swash glyphs
Microsoft Implementation:
The "swsh" feature replaces default glyphs with swash versions of the glyph. Microsoft implements the feature as an alternate substitution lookup (GSUB LookupType = 3).
Tag: **"sups"**
Function: Provides superscript glyphs
Microsoft Implementation
The "sups" feature may replace a default glyph with a superscript glyph, or it may combine a glyph substitution with positioning adjustments for proper placement. First, a single or contextual substitution lookup implements the superscript glyph. Then, if the glyph needs repositioning, Microsoft applies a single adjustment, pair adjustment, or contextual adjustment positioning lookup.
Tag: **"subs"**
Function: Provides subscript glyphs
Microsoft Implementation
The "subs" feature may replace a default glyph with a subscript glyph, or it may combine

a glyph substitution with positioning adjustments for proper placement. First, a single or contextual substitution lookup implements the subscript glyph. . Then, if the glyph needs repositioning, Microsoft applies a single adjustment, pair adjustment, or contextual adjustment positioning lookup may then modify its position.

Tag: **"kern"**

Function: Adjusts amount of space between glyphs

Microsoft Implementation

The "kern" feature loosens or tightens glyph spacing. Microsoft implements a pair positioning lookup (GPOS LookupType = 2) when two glyphs are adjusted in relation to another or a contextual positioning lookup in GPOS when more than two glyphs are adjusted.

The TrueType Open "kern" feature is capable of providing more information than a TrueType 1.0 "kern" table. For instance, it supplies size dependent kerning data via device tables, "cross stream" kerning in the Y text direction, and independent adjustment of glyph placement and advance.