OpenType Font

by

Harsha Wijayawardhana
UCSC

Introduction

- The OpenType font format is an extension of the TrueType font format, adding support for PostScript font data .
- The OpenType font format was developed jointly by Microsoft and Adobe.
- OpenType fonts and the operating system services which support OpenType fonts provide users with a simple way to install and use fonts, whether the fonts contain TrueType outlines or CFF (PostScript) outlines.

How do we define Open Type fonts

- Open Type fonts are important for non Western writing.
- In most western writing systems, a letter of the alphabet is individually mapped to a shape and so a one to one mapping exists between the characters in the text string and the glyphs in the display

Open Type fonts cont..

- In syllabic writing (Phonemes), the above changes.
- the displayed text in syllabic writing text is actually built up by applying the rules for shaping each syllable.
- These rules follow a fairly well established pattern for simple syllables but may associate special shapes with specific syllables

Open Type fonts cont..

- The text to be displayed could indeed be specified in terms of the consonants and vowel in a syllable which are the basic linguistic units in the language. But the desired shape for the syllable cannot be effected by simply placing the shapes for the consonants and the vowel in sequence.

Open Type fonts cont..

Input String (Unicode characters)		Written form	Glyphs used (In most fonts)	
2	क ि	कि	ि क	2
2	क ृ	कृ	क ृ	2
4	त ् र े	त्रे	त्र े	2
3	ज ् ञ	ज्ञ	ज्ञ	1
5	ष ् ट ् र	ष्ट्र	ष्ट ् र	2
4	प ् र ि	प्रि	ि प्र	2

The figures in Red indicate the number of Unicode characters in the syllable
The figures in Blue indicate the number of Glyphs used to display the syllable

The OpenType font format addresses the following goals:

- broader multi-platform support
- better support for international character sets
- better protection for font data
- smaller file sizes to make font distribution more efficient
- broader support for advanced typographic control

Advanced Typographic Extensions - OpenType Layout

- The Advanced Typographic tables (OpenType Layout tables) extend the functionality of fonts with either TrueType or CFF outlines.
- OpenType Layout fonts contain additional information that extends the capabilities of the fonts to support high-quality international typography:

Advanced Typographic Extensions – cont.

- OpenType Layout fonts allow a rich mapping between characters and glyphs, which supports ligatures, positional forms, alternates, and other substitutions.
- OpenType Layout fonts include information to support features for two-dimensional positioning and glyph attachment.
- OpenType Layout fonts contain explicit script and language information, so a text-processing application can adjust its behavior accordingly.
- OpenType Layout fonts have an open format that allows font developers to define their own typographical features.

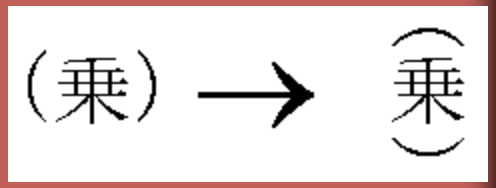
OpenType Layout at a Glance

- OpenType Layout addresses complex typographical issues that especially affect people using text-processing applications in multi-lingual and non-Latin environments.
- OpenType Layout fonts may contain alternative forms of characters and mechanisms for accessing them.

OpenType Layout at a Glance

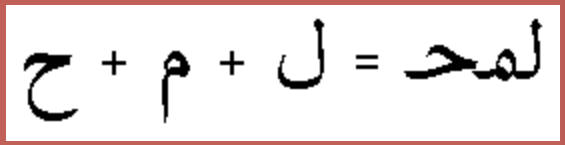
- OpenType Layout helps a text-processing application determine which variant to substitute when composing text.

- OpenType Layout helps an application use the correct forms of characters when text is positioned vertically instead of horizontally, such as with Kanji.



OpenType Layout at a Glance cont.

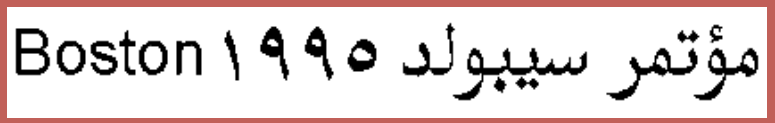
- The OpenType Layout font format also supports the composition and decomposition of ligatures.



- Glyph substitution is just one way OpenType Layout extends font capabilities. Using precise X and Y coordinates for positioning glyphs, OpenType Layout fonts also can identify points for attaching one glyph to another to create cursive text and glyphs that need diacritical or other special marks.

OpenType Layout at a Glance cont.

- OpenType Layout fonts also may contain baseline information that specifies how to position glyphs horizontally or vertically.



TrueType versus OpenType Layout

- A TrueType font is a collection of several tables that contain different types of data: glyph outlines, metrics, bitmaps, mapping information, and much more.
- Text-processing applications - referred to as "clients" of OpenType Layout - can retrieve and parse the information in OpenType Layout tables.
- The tables do not try to encode information that remains constant within the conventions of a particular language or the typography of a particular script.

OpenType Layout Terminology

- The OpenType Layout model is organized around glyphs, scripts, language systems, and features.
- *Characters versus glyphs*

Users don't view or print characters: a user views or prints *glyphs*. A glyph is a representation of a character. A font is a collection of glyphs. To retrieve glyphs, the client uses information in the "cmap" table of the font, which maps the client's character codes to glyph indices in the table. Glyphs can also represent combinations of characters and alternative forms of characters: glyphs and characters do not strictly correspond one-to-one.

OpenType Layout Terminology cont.

- *Scripts*
  - A script is composed of a group of related characters, which may be used by one or more languages. Latin, Arabic, and Thai are examples of scripts. A font may use a single script, or it may use many scripts. Within an OpenType Layout font, scripts are identified by unique 4-byte *tags*.

OpenType Layout Terminology cont.

- *Language systems*

  Scripts, in turn, may be divided into language systems. For example, the Latin script is used to write English, French, or German, but each language has its own special requirements for text processing. A font developer can choose to provide information that is tailored to the script, to the language system, or to both.

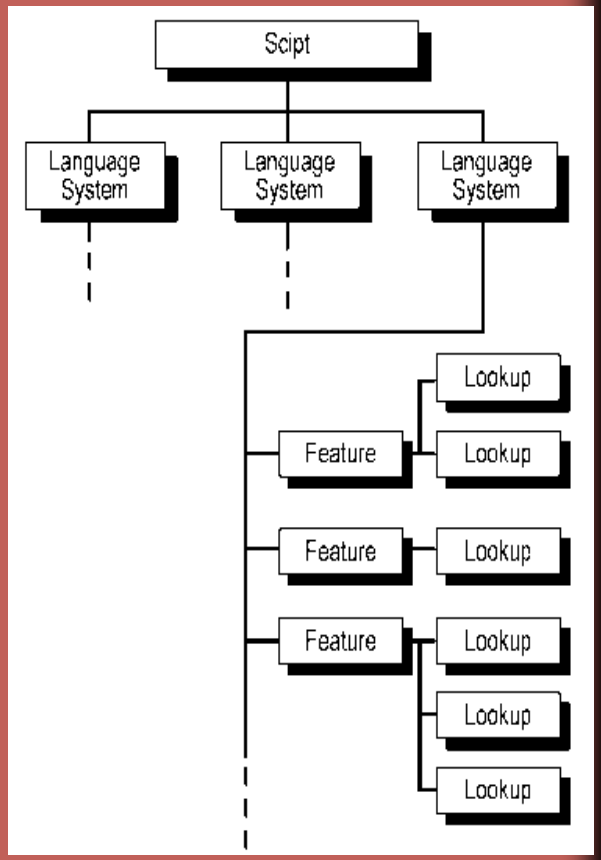
  Language systems, unlike scripts, are not necessarily evident when a text-processing client examines the characters being used .

OpenType Layout Terminology cont.

- *Features*

  Features define the basic functionality of the font. A font that contains tables to handle diacritical marks will have a "mark" feature. A font that supports substitution of vertical glyphs will have a "vert" feature.

  The relationship of scripts, language systems, features, and lookups for substitution and positioning tables.



OpenType Layout Terminology cont.

- *OpenType Layout tables*
  - **GSUB:** Contains information about glyph substitutions to handle single glyph substitution, one-to-many substitution (ligature decomposition), aesthetic alternatives, multiple glyph substitution (ligatures), and contextual glyph substitution.
  - **GPOS**: Contains information about X and Y positioning of glyphs to handle single glyph adjustment, adjustment of paired glyphs, cursive attachment, mark attachment, and contextual glyph positioning.
  - **BASE**: Contains information about baseline offsets on a script-by-script basis.
  - **JSTF:** Contains justification information, including whitespace and Kashida adjustments.
  - **GDEF**: Contains information about all individual glyphs in the font: type (simple glyph, ligature, or combining mark), attachment points (if any), and ligature caret (if a ligature glyph).

Text processing with OpenType Layout fonts

- A text-processing client follows a standard process to convert the string of characters entered by a user into positioned glyphs. To produce text with OpenType Layout fonts:
  1. Using the cmap table in the font, the client converts the character codes into a string of glyph indices.
  2. Using information in the GSUB table, the client modifies the resulting string, substituting positional or vertical glyphs, ligatures, or other alternatives as appropriate.
  3. Using positioning information in the GPOS table and baseline offset information in the BASE table, the client then positions the glyphs.

Text processing with OpenType Layout fonts cont.

4. Using *design coordinates* the client determines device-independent line breaks. Design coordinates are high-resolution and device-independent.
5. Using information in the JSTF table, the client justifies the lines, if the user has specified such alignment.
6. The operating system rasterizes the line of glyphs and renders the glyphs in *device coordinates* that correspond to the resolution of the output device.

Unicode script processor (Uniscribe)

The Unicode Script Processor (USP10.DLL) is a collection of API's that enable a text-layout client to format complex scripts.
supports the complex rules found in scripts such as Arabic, Indian, and Thai. Uniscribe also handles scripts written from right-to-left, such as Arabic or Hebrew, and supports the mixing of scripts.

Unicode Script Processor cont..

Uniscribe is composed of multiple "shaping engines." These shaping engines contain the layout knowledge for particular scripts (for example, Arabic, Hebrew, Thai, Hindi, Tamil). In addition, there is an *OpenType* Layout shaping engine for handling script features unknown to Uniscribe. Uniscribe provides character-to-glyph mapping; dx,dy positioning; line breaking at word boundaries; hit testing and cursor positioning.

How to set the rules in Sinhala Open type fonts

- Definitions
  - Syllables: Any consonants can form a syllable with a vowel.
  - Matra: Each vowel has a special shape associated with it for use with a combining consonant. This is known as a Matra. In Sinhala, Ka and Matra ii make kii. When matra is added to a basic consonant, the result is a syllable consisting of the consonant and the vowel.

Rules cont …

- Akhand Ligatures: consonant ligatures appear any part of syllable and may or may not involve base glyphs. As an example in Sinhala, Payana , Hal kirima with Sha by combining form Paksha. Akhand Ligatures have the highest priority.

Rules cont …

- Consonant: represents each consonant sound and consonants by itself have inherent vowel sound. Hal kiriyma (virama) removes the inherent vowel sound.
- Reph: In sinhala, Hal kirima and Ra as above base form is known as Reph form and, in Sinhala without repaya, words which have hal kirima and Ra can be written. There are exceptions to Reph forms in Sinhala.

Rules cont …

- Vattu (Rakar) below base form of letter Ra. In Sinhala, there are two vattu variants: rakaraynsya and Yansaya. Rakayansaya occurs below the base glyph, Yansaya on the other hand occurs at the right side of the base.

Rules cont …

- The following outlines how the uniscribe works for Indic scripts.

1. Uniscribe is a single binary with specific script engine for Indic script. As an example, <sinh> identifies Sinhala in the Indic Script engine.
2. Under each script, features are introduced to create the basic forms for the script. The order of the lookups under each feature is very important for proper rendering of the script. The order always has to be the longest lookups first and the next longest coming second and so forth.

Rules cont …

1. The standard order of applying features in the uniscribe is given below:

- i. nukt Nukta form
- ii. akhn Akhand form
- iii. rphf Reph form
- iv. blwf Below-base form
- v. half Half-base Form (Pre-base form)
- vi. pstf Post-base Form
- vii. vattu vattu variants

Rules cont …

- Conjunct & Typographical
- i. pres Pre-base substitution
- ii. blws Below base substitution
- iii. abvs Above base substitution
- iv. psts Post-base substitution
  Halant form
- v. haln Halant form substitution

Rules cont …

- Positioning features:
- vi. blwm below-base mark positioning
- vii. abvm Above-base mark positioning
- viii. dist Distances

Rules cont …

- In Sinhala, one does not require to have all the lookups mentioned above. The minimum requirement is to have the following and is enumerated below in the order of the application of the lookups by the Uniscribe:

1. Akahnd
2. Reph form
3. Below base form
4. Post –base form
5. Vattu variants

Rules cont …

1. Pre-base substitution (one can use without using Post base form)
2. Below base substitution
3. Above base substitution
4. Post base substitution

Rules cont …

- All of the above are Substitution features and they are handled by GSUB tables. Two main positioning features, which can be used for Sinhala, are:

1. Below-base mark positioning
2. Above-base mark positioning

Rules cont …

- One must heed to the priority order of the features that are mentioned above. Pre base, below base, above base and post base substitution primarily deal with Matras and are in the lowest priority order respectively

Rules cont …

- Under the below base substitution, all consonants with Matra Paapilla, such as sound Puu, can be defined. For instance, sound Paksu which is comprises of Unicode points of Payana + Kayana+Hal+ZWJ+Shaa+Paapilla, shuu sound of which will be defined in two lookups.

Rules cont …

- Initially, Since Akhand has the highest priority order, Ka+Hal+ZWJ+Shaa Kshaa ligature-Ksha will be defined under Akhand Feature, and in the lower order of the below base substitution, shaa+paapilla matra forms the shuu sound. Accordingly, certain sounds can have more than two lookup for the final glyphs. Above base deals with ispilla matra and post base deals with Paapilla of certain consonants such as Ka Gha and Ra.