Unicode, the Moving Target

Roozbeh Pournader

roozbeh@sharif.edu

TEX Users Group Meeting, September 2002

Trivandrum, India

The outline

- Introduction
- What is Unicode?
- What is OpenType?
- Other Friends
- New requirements for Omega

Introduction

- Unicode and friends, from OpenType to IDN
- Making global interchange possible, from Malayalam to APL to Runic
- Many tools available, from Microsoft Internet Explorer to IBM's International Classes (ICU) for Unicode to GNOME's Pango
- As complex as the scripts themselves, from Hangul to Arabic to Tibetan

Introduction (continued)

- Supporting modern text processing needs, from sorting and searching to character-level markup to accent positioning and contextual forms
- Widely deployed by Microsoft, from Word to SQL Server
- . . . but also backed by GNU and Linux communities, from Perl to `vim` to Trolltech's Qt to GNU readline to Linux console
- These are *alive*, *dynamic*, and *open*, and they encode everything that moves!

What is Unicode?

- $0 \leq$ character code $\leq \texttt{10FFFF}_{16} = 17 \times 2^{16} - 1 = 1{,}114{,}111$
- The minimum number of bits enough for encoding every character is 21, but that's almost nowhere used. (It's actually a 20.08746-bit character set!)
- U+20A8 is the RUPEE SIGN (Rs)
- Characters are arranged in blocks so one can find them easily (from Cyrillic to Mathematical Operators)

But what is a character?

- We don't know!

- . . . but we know some things that are not characters:
  - Glyphs: there is only one SYRIAC LETTER BETH (ܒ)
  - Ligatures: there is no LATIN LIGATURE CT (ct)
  - Markup: there is no START BOLDFACE (`\bfseries`)
  - Logos and emblems: there is no APPLE SIGN

That's a big lie!

- Glyphs: there are four different presentation forms of ARABIC LETTER GAF (ﮒ, ﮓ, ﮔ, ﮕ), in addition to one general one
- Ligatures: there is a LATIN SMALL LIGATURE FI (ﬁ), among many others
- Markup: there are control character everywhere, from a PARAGRAPH SEPARATOR to something named POP DIRECTIONAL FORMATTING
- Logos and Emblems: HAMMER AND SICKLE (☭) is there, as well as the ORTHODOX CROSS (☦)

Not just codes, names or shapes

- Several informative or normative properties and descriptions are available to disambiguify the characters:

  general category, combining class, bidirectional category, decomposition mapping, numeric value, mirroring property, case mappings, joining class and group, line breaking property, . . .

Some character properties

- **Decomposition, recomposition, reordering, equivalence, and normalization:** to make sure that me and you encode the same string the same way.

- **Bidirectional properties and behavior:** to make sure logically encoded bidirectional scripts are not displayed differently on my computer than yours.

What is decomposition?

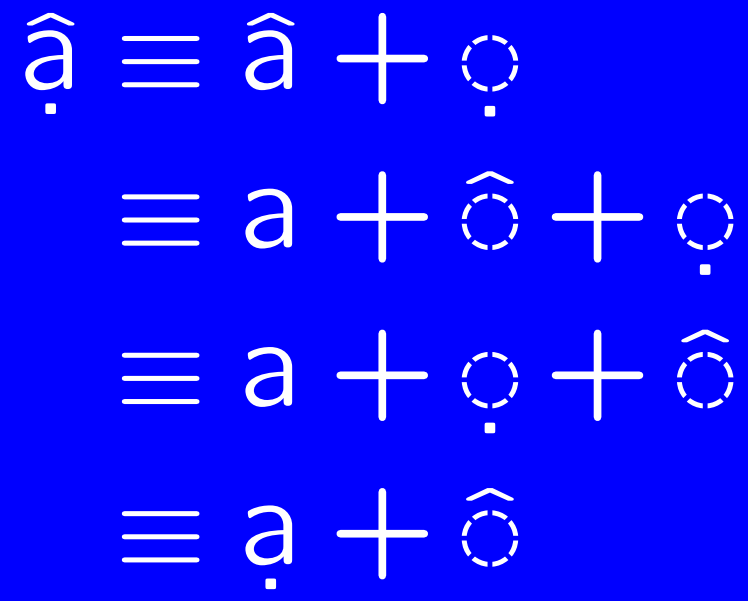
Decomposition is a Unicode normative property explaining the nature of some unnecessarily composite or complex character:

$$\tilde{\mathrm{a}} \rightarrow \mathrm{a} + \tilde{\circ}$$

$$\mathcal{H} \rightarrow \langle \mathit{font} \rangle (\mathrm{H})$$

What is equivalence? (continued)

Strings that are exactly the same in every respect, but the corresponding character sequence:



What is equivalence? (continued)

$$\begin{aligned} \tilde{\acute{u}} &\equiv \tilde{u} + \acute{o} \\ &\equiv u + \tilde{o} + \acute{o} \\ &\not\equiv u + \acute{o} + \tilde{o} \equiv \tilde{\acute{u}} \end{aligned}$$

Combining classes

- To help applications determine the general position of the accent, and provide a canonical ordering of the combining marks. Same position gets same number.
  - Cedilla and Ogonek: 202 (Below Attached)
  - Dot Below and Arrow Below: 220 (Below)
  - Grave, Acute, and Tilde: 230 (Above)

Normalization Forms

- Normal forms help applications do binary equivalence checking instead of applying the heavy equivalence determination algorithm

- **Normalization Form D (NFD):** Decomposing all precomposed letters, and then ordering the combining characters based on their combining classes

- Mobile phones can detect equivalent strings, C programmers can use `strcmp`, UNIX users can `grep`

Normalization Form C (NFC)

Recomposing letters back after doing decomposition and reordering:

- Maximum compatiblity with old character sets like ISO-8859-1
- Very simple to render for embedded systems
- The standard required for all World Wide Web content
- Recomended way for encoding text files and file names in UNIX (except Mac OS X, which uses NFD)

Bidirectional Algorithm

Providing an *exact* and *implicit* mechanism for converting a logically stored stream of characters including some characters of a right-to-left script, to a visually ordered one suitable for display.

This is needed for Arabic (incl. Persian, Urdu, Sindhi, . . .) Hebrew (incl. Yiddish), Syriac, and Thanaa.

A car is called "ΤΗƎ ƆΑЯ" in Hebrew

⇓

A car is called "ЯΑƆ ƎΗΤ" in Hebrew

Bidirectional Algorithm (continued)

Many implicit and explicit *bidirectional categories*:

left-to-right, right-to-left, right-to-left Arabic, European number, Arabic number, European number separator, European number terminator, common number separator, non-spacing mark, boundary neutral, paragraph separator, segment separator, whitespace, other neutrals, *left-to-right embedding, right-to-left embedding, left-to-right override, right-to-left override, pop directional format*

Interesting features

- Different characters for letters that look the same but have different semantics:

code	00D0	0110	0189
name	Eth	D with stroke	African D
uppercase shape	Ð	Đ	Ɖ
lowercase shape	ð	đ	ɖ
name	Eth	D with stroke	D with tail
code	00F0	0111	0256

Interesting features (continued)

- Ligation, digraph shaping, and digraph breaking control characters
- Line breaking properties
- Mathematically semantical characters
- Mirroring characters
- All characters and symbols needed for mathematical typesetting (thanks to AMS)

Compliance!

Compliance

You may not interpret a character against what is written in the standard. Breaking any of the rules makes your application non-compliant:

- You MUST NOT render two canonically equivalent strings differently
- You MUST NOT use unassigned code points
- You MUST NOT enhance the bidirectional algorithm

Compliance (continued)

You don't want a user to see something with a different meaning when she opens a file created by someone else's application, do you?

Font standards

- A font standard is needed for providing the character to glyph conversion info

- Three are available:
  - Microsoft and Adobe's OpenType
  - Apple's Advanced Typography (AAT)
  - SIL's Graphite

OpenType

- A proper superset of TrueType
- Various implementations: FreeType project, GNOME project, IBM
- Many high quality fonts
- The only choice if you want e.g. Devanagari or Khmer

Just some data tables

- It doesn't encode the visual semantics of the scripts. It's just some data tables for contextual forms, kerning, positioning accents and marks, ligating, . . .

- The script logic is in the layout engine: An OpenType font won't say if an ARABIC LETTER ALEF MAKSURA (ى) is a right-joining letter or a dual-joining one, but only provides certain contextual forms if asked by the layout engine.

Other friends

PDF 1.4: Provides a mechanism for storing Unicode values of the original character streams

CSS 2 and 3: Specify guidelines for how to render a character stream using available fonts

MathML 2: With almost all its symbols now in Unicode, will let you cut and paste mathematics between applications

Where do we stand?

- METAFONT, PS Type 1, TFM and PK instead of OpenType
- DVI instead of PDF
- Loose syntax instead of XML
- It's not those good old days anymore. You cannot just invent something and hope it will become *de facto*

Is Omega the saviour?

- Pros: already implemented 16-bit fonts, post- and pre-processing filters, some XML and MathML
- Cons: non-stable, lacks an active team, too academic, *developed in the Cathedral model*

New Omega requirements

- Opening Omega development: Help! We need volunteers!
- Accepting the international standards as they are: if there are bugs in specifications, instead of fixing them locally, they should be fixed in the standard
- PDF output, supporting Unicode character streams

New Omega requirements (continued)

- Native OpenType support: Making Omega fonts Unicode and OpenType compliant as the first milestone, OpenType to ΩTP and ΩFM as the second, . . .
- XML and MathML as both input and output formats
- Following Unicode closely

Compliance!

Questions?