Making TeX support Unicode

The Quest for the Holy Grail

The Holy Grail

An obvious statement

Since luaTeX, XƎTeX, and other TeX engine can read UTF-8 files and handle 16-bit input correctly, they fully support Unicode, right?

Really?

Really?

NO!

Unicode support is not equivalent to UTF-8 input; Unicode is not a pile of characters without relations between each others. And it needs more than 16 bits (21, approximately).

What do me miss, then?

Unicode defines a number of properties for each character, and even processes to transforms character streams into others.

Many of them are still missing from any TeX engines.

Combining characters

Informal definition: A combining character is a character that puts an accent on the character it follows.

This is well known to TeX users, except that it *follows* the character it applies to.

Combining demo

Canonical equivalence & normalization

We have several ways to input characters like ž: ⟨ž⟩ and ⟨z, ˇ⟩.

What is the difference, then?

Unicode says: none!

More precisely, it defines such sequences as *canonically equivalent*, and says:

A process shall not assume that the interpretations of two canonical-equivalent character sequences are distinct.

Unicode defines transformations of the character sequences that preserve canonical equivalence. This is called *normalization*.

There are four normalization forms; two are of interest here: Normalization Form C (NFC) – maximum composition – and Normalization Form D (NFD) – maximum decomposition.

Normalization demo

Trivia: Normalization is especially relevant for “European” alphabetic scripts … and for Korean.

Unicode also contains all sort of characters with special properties: unbreakable space, zero-width-non-joiner, soft hyphen, etc.

No math …

Unfortunately, I have little knowledge about Unicode math encoding, but this is also a very important aspect for TEX especially in connexion with the Gyre math project.