## Putting the Cork back in the bottle

Improving Unicode

support in TeX

Mojca Miklavec & Arthur Reutenauer

#### **Genesis**

As a part of its *Summer of Code* programs, Google sponsors us to work on projects related to T<sub>E</sub>X.

I chose to work on "making TEX more Unicode-compliant".

Patterns
Intermission
More Unicode
Thanks

## The Name of the Game Genesis **Patterns** Intermission **More Unicode** Thanks Close

## The Name of the Game Genesis **Patterns** What is Unicode? Intermission **More Unicode** Thanks Close

What is Unicode?

A universal character set, suitable for representing any writing system.

Genesis

Patterns

Intermission

More Unicode

Thanks

What is Unicode?

A universal character set, suitable for representing any writing system.

What is TEX?

Genesis

Patterns

Intermission

More Unicode

Thanks

What is Unicode?

A universal character set, suitable for representing any writing system.

What is TEX?

'You kidding me?

Genesis

Patterns

Intermission

More Unicode

Thanks

What is Unicode?

A universal character set, suitable for representing any writing system.

What is TEX?

'You kidding me?

What does it mean for a TEX-based system to be Unicode-compliant?

Genesis

Patterns

Intermission

More Unicode

Thanks

What is Unicode?

A universal character set, suitable for representing any writing system.

What is TEX?

'You kidding me?

What does it mean for a TEX-based system to be Unicode-compliant?

Can you repeat the question?

Genesis

Patterns

Intermission

More Unicode

Thanks

#### A First Step

Supporting Unicode implies to support – at least – UTF-8 encoding. All of the TEX macro packages can accommodate it, but until very recent times, some parts of the core support files ignored it completely, in particular, hyphenation patterns. They used various "legacy" encodings known to TEX.

This was a problem when X<sub>3</sub>T<sub>E</sub>X was integrated in T<sub>E</sub>X Live in 2007.

Jonathan Kew then devised a way to convert the patterns to UTF-8 on the fly, if needed. He wrapped them in files called Xu-<hyphen>. tex (pronounced "zoo hyphen"). These detect if they are running XaTeX or some other TeX engine, and convert the patterns to UTF-8 in the former case: you make characters active, and define them to yield the corresponding UTF-8 byte sequence.

Genesis

Patterns

Intermission

More Unicode

Thanks

#### **Patterns**

For TEX Live 2008 we wanted to address the problem the other way round: the input files should be in UTF-8, and we should convert them to TEX's font encodings when using 8-bit engines.

We also wanted to make a clear distinction between the patterns and the TEX support code: \Catcode's, \lambda Code's (and other things you don't want to hear about).

Finally, we wished to adopt a clean naming scheme for the languages at stake, and we chose IETF language tags for that, a.k.a. RFC 4646. It was the only standard we found that could name all the language variants we needed to name. ISO codes simply weren't enough.

The next two slides give an overview of this strategy.

Patterns
Intermission
More Unicode
Thanks

#### Loading the patterns

The top-level file is called loadhyph-<language code>.tex, like here for Slovenian:

```
% Test whether we received one or two arguments
```

```
\def\testengine#1#2!{\def\secondarg{#2}}
```

```
% That's Tau (as in Taco or TEX, Tau-Epsilon-Chi),
```

```
% a 2-byte UTF-8 character
```

\testengine T!\relax

```
% Unicode-aware engines (such as XeTeX or LuaTeX)
```

```
% only see a single (2-byte) argument
```

```
\ifx\secondarg\empty
```

\message{UTF-8 Slovenian Hyphenation Patterns}

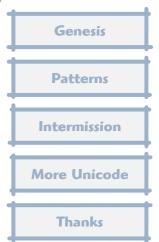
```
\else
```

\message{EC Slovenian Hyphenation Patterns}

```
\input conv-utf8-ec.tex
```

\fi

\input hyph-sl.tex



#### Converting the patterns

The converter files are called CONV-utf8-<font encoding>.tex.

Extract of CONV-utf8-ec.tex:

```
\catcode"C4=\active
\catcode"C5=\active
```

```
\def^^c4#1{%
\ifx#1^^8d^^a3\else % č - U+010D
\fi}
```

```
\def^^c5#1{%
\ifx#1^^a1^^b2\else % š - U+0161
\ifx#1^^be^^ba\else % ž - U+017E
\fi\fi}
```

```
%
% ensure all the chars above have valid lccode's
%
```

```
\lccode"A3="A3 % č - U+010D
\lccode"B2="B2 % š - U+0161
\lccode"BA="BA % ž - U+017E
```

Patterns
Intermission
More Unicode
Thanks

#### Taming the patterns

In many cases, things don't work as smoothly as they could be expected to.

Some languages use patterns that try to accommodate T1 and OT1 in the same file. This happens for German, French, Danish, Latin.

Some files can be customized to load completely different pattern sets (Russian, Ukrainian).

Sometimes, Unicode is inherently bad at representing the language at hand (Ancient / Polytonic Greek).

Sometimes Babel isn't on our side (Serbian).

Genesis

Patterns

Intermission

More Unicode

Thanks

#### **Breeding the patterns**

The "new" patterns have been available for a few weeks on CTAN under the name hyph-utf8, and have been imported into TEX Live for inclusion in the 2008 DVD. They are the basis for language support in plain TEX and LATEX through Babel, and have been integrated in the ConTEXt multilingual infracstructure.

Genesis

Patterns

Intermission

More Unicode

Thanks

#### Intermission

What we have learned in the process:

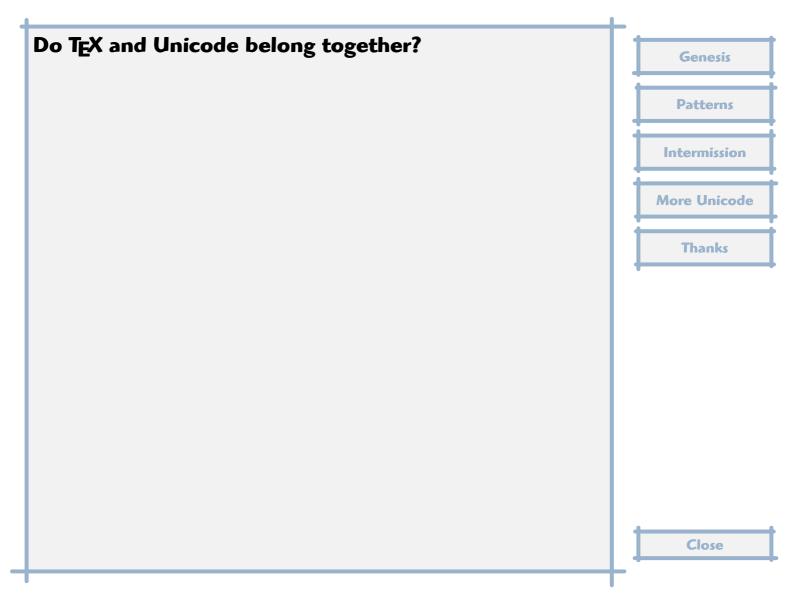
Genesis

**Patterns** 

Intermission

**More Unicode** 

Thanks



### Do TEX and Unicode belong together? Genesis **Patterns** All the data is easy to represent in Unicode. Intermission **More Unicode** Thanks

#### Do TEX and Unicode belong together?

All the data is easy to represent in Unicode.

The real problem was to integrate the patterns in the general TEX land-scape, and to preserve Holy Backward Compatibility.

Genesis

Patterns

Intermission

More Unicode

Thanks

#### How many languages?

ar Arabic fa Farsi eu Basque bg Bulgarian cop Coptic hr Croatian CS Czech da Danish nl Dutch eo Esperanto et Estonian fi Finnish fr French de-1901 German, "old" spelling de-1996 German, "new" spelling el-monoton Modern Greek, monotonic spelling el-polyton Modern Greek, polytonic spelling grc Ancient Greek grc-x-ibycus Ancient Greek in Ibycus encoding hu Hungarian is Icelandic id Indonesian ia Interlingua ga Irish it Italian la Latin Mongolian, Cyrillic script mn-cyrl-x-2a Mongolian, Cyrillic script (new patterns) Norwegian nb Norwegian Bokmål nn Norwegian Nynorsk zh-latn Chinese Pinyin pl Polish pt Portuguese ro Romanian Russian ru sr-latn Serbian in the Latin script sr-cyrl Serbian in the Cyrillic script sh-latn Serbo-Croatian in the Latin script sh-cyrl Serbo-Croatian in the Cyrillic script sl Slovene es Spanish Swedish SV tr Turkish en-qb British English en-us American English uk Ukrainian hsb **Upper Sorbian** су Welsh

Genesis

**Patterns** 

Intermission

**More Unicode** 

**Thanks** 

#### More Unicode

Actually, supporting Unicode implies much more than that: my original proposal for Google Summer of Code included, in particular, to better handle combining characters in X<sub>3</sub>T<sub>E</sub>X and luaT<sub>E</sub>X.

Combining characters are Unicode's diacritical marks: you put them after a base character to add an accent to the latter.

For example, Unicode character U+017E LATIN SMALL LETTER Z WITH CARON  $(\check{z})$  can also be represented as the sequences of two characters U+007A LATIN SMALL LETTER Z (z) followed by U+030C COMBINING CARON  $(\check{z})$ .

Unicode specifies algorithms, known as normalization, to transform character sequences in fully decomposed or fully composed form.

Patterns
Intermission
More Unicode
Thanks

## Handling normalization natively Genesis **Patterns** Intermission **More Unicode** Thanks Close

# Handling normalization natively XaTeX has been recently extended to support this at the engine level. Intermission More Unicode Thanks

#### Handling normalization natively

XaTeX has been recently extended to support this at the engine level.

In luaTEX, it can be handled in the macro package thanks to appropriate hooks, which it was ConTEXt Mark IV already does.

Genesis

Patterns

Intermission

More Unicode

Thanks

#### Handling normalization natively

XaTeX has been recently extended to support this at the engine level.

In luaTEX, it can be handled in the macro package thanks to appropriate hooks, which it was ConTEXt Mark IV already does.

Problem solved!

Genesis

Patterns

Intermission

More Unicode

Thanks