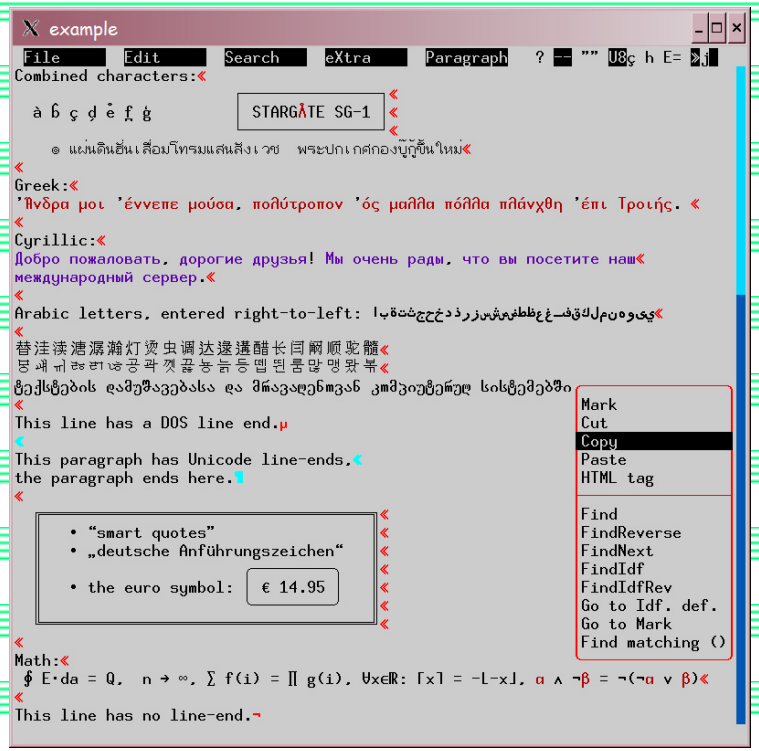
MINED: An Editor with Extensive Unicode and CJK Support for the Text-based Terminal Environment

IUC 27, Berlin, 2005-04-08

Thomas Wolff

MINED: introduction

- **A text editor**
  - suitable for editing text
  - suitable for editing programs
- **Editing environment**
  - text-mode terminal (xterm, mlterm, hanterm, cxterm, rxvt, linux console)
  - non-graphical UI: more light-weight interaction, seemless integration into command-line workflows
  - *MINED was the first editor that supported UTF-8 in this environment*



MINED: «IUC» support

- **Internationalisation support**
  - encoding support: UTF-8, CJK, and others
  - character and script information
  - character input support, input methods
- **Encoding environment**
  - text encoding: automatic detection, flexible handling
    - » configurable detection
    - » mixed-encoding editing, switching online
  - terminal encoding: automatic detection
    - » 8 bit vs. UTF-8 vs. CJK
    - » various sets of character width properties

MINED: overview

- **User interface**
  - simple and intuitive
  - affirmative toward modern interaction paradigms
    - » comprehensive menus
    - » mouse control, wheel scrolling
    - » scrollbar navigation
- **Program editing features**
  - program structure, auto-indent
  - identifier search functions (multi-file)
  - HTML highlighting and tag matching
  - multiple lines in search/replacement patterns
  - multi-file copy/paste
  - visual indications, binary transparency

Typographic editing support

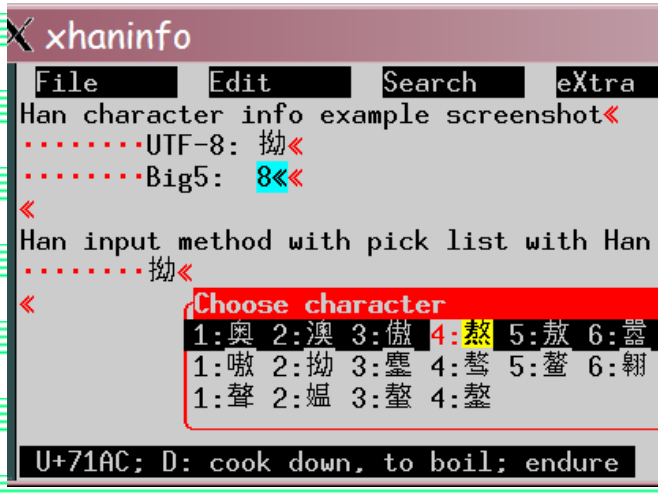
- **Smart quotes**
  - smart quotes mode with different styles
  - straight quotes (as from keyboard) are replaced with typographic quote marks for insertion
  - opening/closing quote mark
    - » depending on context
    - » nested handling for double/single quotes
    - » special heuristics for CJK quotes (without space context)
- **Smart dashes**
  - "--" → "–"
  - "---" → "—"
  - <Hebrew context>"-" → ־ (Maqaf)

Input support: character input

- **Mnemonic input**
  - RFC 1345 with completions
  - HTML mnemonics, TeX mnemonics
  - useful supplements
- **Numeric input**
  - hex, octal, decimal
  - native encoding, Unicode value
  - ISO 14755
- **Accented input**
  - accent prefix function keys
  - extensions for Vietnamese multiple accent combinations

Input support: input methods

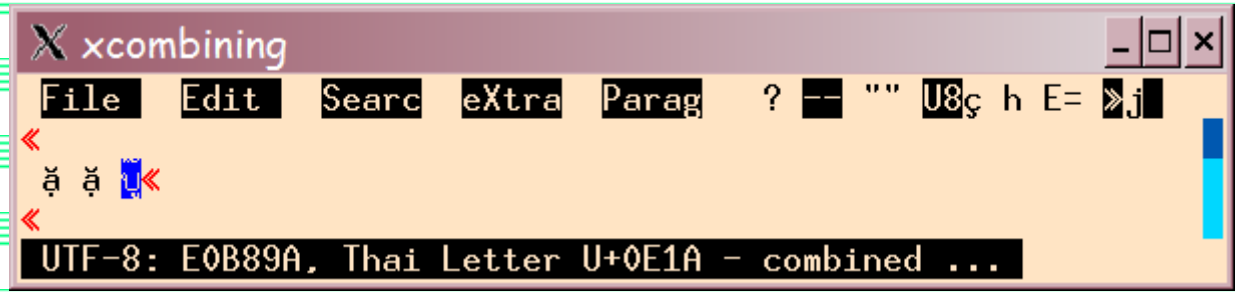
- **Simple keyboard mapping**
  - Greek, Cyrillic, Hebrew, Arabic, Thai
- **Input methods**
  - CJK
  - extended keyboard mapping
    - » multi-character input sequences
    - » ambiguous mappings
    - » resolution with selection menu (“pick-list”)
  - Radical/Stroke two-level selection menus



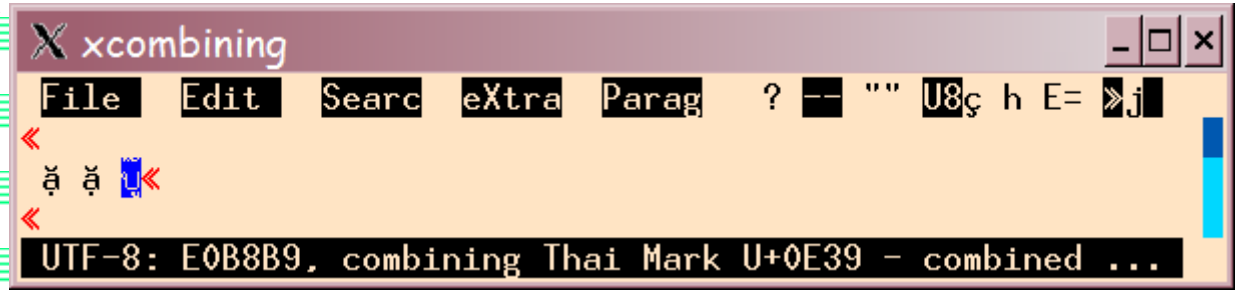
- **“Out-of-the-box” approach**
  - all mapping tables built-in
    - » all input methods are always available, even on legacy systems

Character handling: combining characters

- **Combined display mode**
  - "normal" handling of combining characters



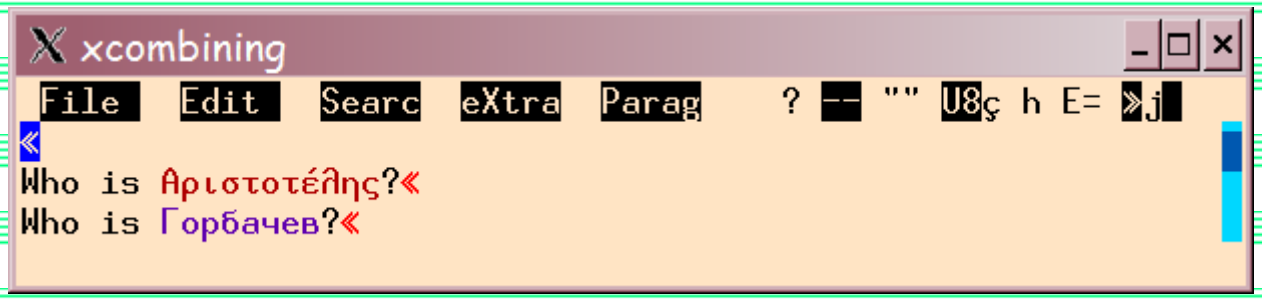
- **Combined editing**
  - move cursor "into" a combined character, edit parts



- **Separated display mode**
  - provides clear view of combining characters in text

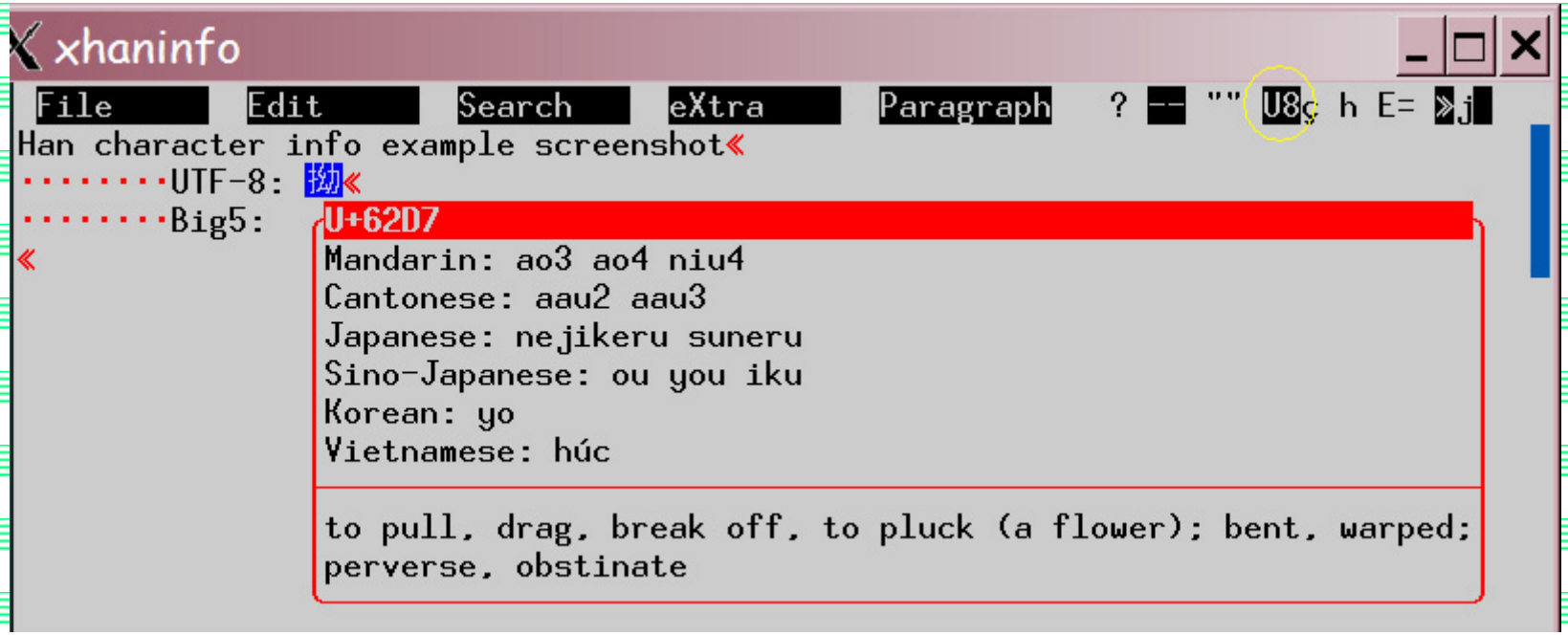
Character handling: script highlighting

- **Distinguish characters with similar-looking glyphs**
  - coloured display of certain scripts



Character handling: Han character information

- **Information from Unihan database**
  - pronunciation entries (configurable selection)
    » Mandarin, Cantonese, (Sino-)Japanese, Korean, Vietnamese
  - semantic description of character (in English)
    » automatic typographic fixed to descriptions



Character handling: interactive conversion

- **Case toggle**
  - handles Greek final sigma, Turkish i
- **Hiragana/Katakana toggle**
  - using case toggle function key
- **Latin-1 / UTF-8 toggle**
  - search for non-matching character codes
  - conversion function
- **Mnemonic conversion**
  - cursor on “oe”
    - » ESC ä → “ö”
    - » ESC ç → “œ”
    - » ESC å → “ø”

Terminal feature detection: locale trouble

- **Why the locale mechanism fails**
  - » export LC_CTYPE=my_country.UTF-8
  - » *installed locale:* vendor_country.utf8
  - user has trouble to find a suitable locale on each machine
    - » if proper encoding is found, language/country may not match
  - want to install fr_FR.UTF-8? → system admin doesn't care!
  - only some tools / terminals use system locale data
    - » esp. xterm maintains its own width data
  - **heterogeneous network:** rlogin / telnet
    - » terminal and application run on different machines
    - » they see different system locale data
    - » → it is in principle not possible to make sure that the locale data you get from the system matches the behaviour of your terminal

Terminal property auto-detection

Auto-detection mechanism

- send test strings to terminal, request cursor position report
  - » determine width of test strings, conclude width properties

String	Width	Conclusion
åلاษษ刈墢	6	UTF-8 terminal, no double-width support, with Arabic LAM/ALEF ligature joining
	7	UTF-8, no double-width, no LAM/ALEF ligature joining
	8	UTF-8, double-width, with LAM/ALEF ligature joining (probably mlterm)
	9	UTF-8, double-width, no LAM/ALEF ligature joining
	10, 11, 14, 16, 17	CJK encoded terminal
	15, 18	8 bit terminal or CJK terminal
a U+0321	1	terminal supports combining characters
	2	terminal does not support combining characters
《》 U+301A U+301B U+FF60	< 8	xterm version 157–166, corresponding Unicode version 3.2
‘’“”…—-	> 8	xterm with option -cjk_width
0x8130A132 («Ğ» in GB18030)	> 2	not a GB18030 terminal
0xA1A4A1B1EAA5A6A1	< 10	non-native CJK terminal, Unicode width data (luit)

MINED: conclusion

- **Character encoding supported "out-of-the-box"**
  - including input methods
    » users may quickly edit international text without config. trouble
- **Terminal interoperability**
  » xterm, mlterm, hanterm, cxterm, rxvt, linux console
- **Good range of text and program editing features**
- **Intuitive user interface**
- **Robust text and file handling engine**
- **Portability**
  » Unix (Linux, Sun, HP, BSD, Mac, ...), Windows (cygwin), DOS
- **Small-footprint behaviour**