

m17nライブラリにおける 表示機能の実現

産業技術総合研究所

半田剣一、錦見美貴子、高橋直人、戸村哲

汎用多言語化ライブラリ

- ◆ ソフトウェアの多言語化を簡単にしたい！
- ◆ 個々のソフトで個別に処理 ⇒ 非効率
- ◆ アプリケーション内で
多言語機能を要する部分は切り出せる
- ◆ 多くのアプリケーションで
必要な多言語インタフェース機能はほぼ共通
- ◆ ということは...

多言語化機能をライブラリにすればみんな幸せ

テキスト表示機能の課題

◆フォントの選択

- ・言語によるフォントの切替
CJK フォント
- ・フォントのレパートリのチェック
Unicode フォント

◆複雑なスクリプトの表示 (CTL)

- ・文字の並べ替え
- ・グリフの選択
- ・リガチャ
- ・グリフの2次元配置
- ・...

CTL へのこれまでの対応

◆スクリプトに独立なコア

テキスト表示に関しては比較的low機能

+

◆スクリプト固有のモジュール

ハードコードされており高機能

◆問題点

- ・モジュール作成者の負荷が大きい
- ・アプリケーションの開発者が、
自分の必要な言語やスクリプトを
自力で追加することが困難

m17n-lib の表示機能1

◆ 汎用高機能コア

言語情報ベースの解釈

+

◆ 言語情報ベース

・ フォント情報

各フォントのエンコーディング/レパートリを決定

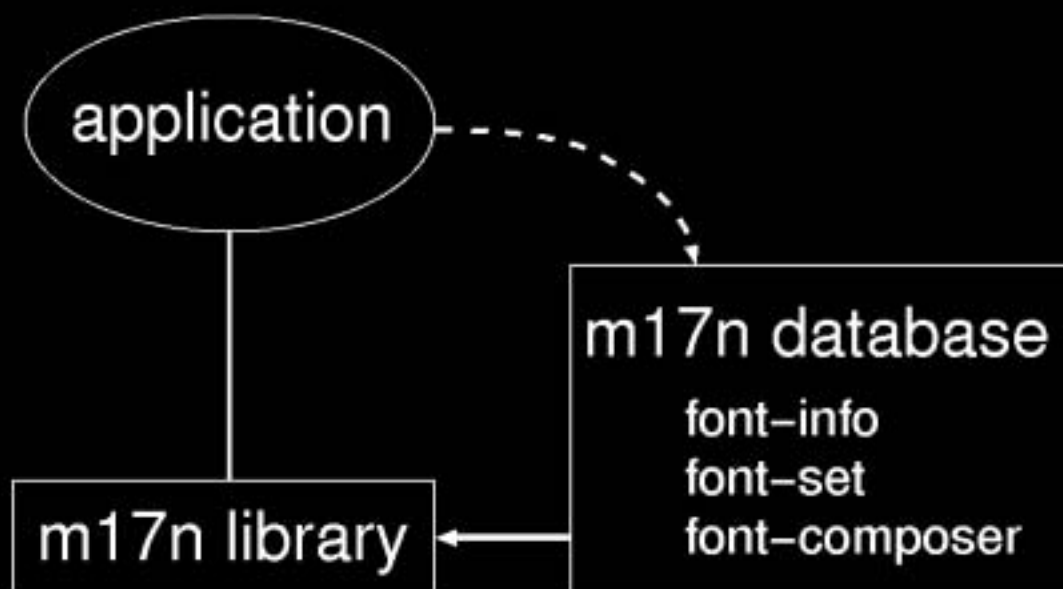
・ フォントセット

スクリプト/言語/文字セットからフォントを選択

・ フォントコンポーザ

文字列をグリフ列に変換 (CTL処理)

m17n-lib の表示機能2



フォントの情報

◆ <FONTINFO> データベース

フォントスペック

VS

エンコーディング&レパートリ

◆ データベースの形式

```
((FAMILY ADSTYLE REGISTRY      ENCODING REPERTORY)
...))
```

◆ 例

```
(nil nil "jisx0208.1983-0"  jisx0208.1983 t)
(nil nil "jisx0212.1990-0"  jisx0212      t)
(nil nil "iso10646-dev"     unicode unicode-dev)
(nil nil "iso10646-1"      unicode      nil)
("adobe-symbol" nil "adobe-fontspecific"
                           adobe-symbol t)
...
```

フォントセット

◆ <FONTSET> データベース

スクリプト (文字から決定)

言語 (テキストの属性から決定)



フォントスペックへのマッピング

◆ データベースの形式

```
((SCRIPT
  (LANG or nil
    (FONT-SPEC [COMPOSER])
    ...))
...))
...))
```

フォントセットの例

```
(greek
  (nil (nil nil "iso10646-1")
        (nil nil "iso8859-7")))
(han
  (ja (nil nil "jisx0208.1983-0")
       (nil nil "jisx0212.1990-0"))
  (zh (nil nil "gb2312.1980-0")
       (nil nil "big5.eten-0")))
(devanagari
  (nil (nil nil "devanagari-cdac" cdac-dev)
        (nil nil "iso10646-dev" pango-dev)))
```

フォントコンポーザ

◆タスク

CTL 処理: ह न् डी ⇒ हिन्डी

◆機能

- ・書記素(grapheme)クラスターの切り出し
- ・文字の並べ替え
- ・グリフの選択
- ・リガチャ処理
- ・グリフの2次元的配置

◆特徴

- ・多段パスで文字列をグリフ列に変換
- ・各パスの変換はルールベース
- ・正規表現による文脈依存処理

フォントコンポーザの構成

◆ 構成要素

- ・コードのカテゴリ定義
 - 入力コードをカテゴリ (A-Za-z) に分類
- ・生成ルール
 - 入力コード列から出力コード列を生成
(PATTERN COMMAND ...)

◆ 多段パス

- 第1ステージのカテゴリの定義
- 第1ステージの生成ルール (文字列→中間コード列)
- ...
- 最終ステージのカテゴリの定義 (必要ならば)
- 最終ステージの生成ルール (中間コード列→グリフ列)

コードカテゴリ

◆ デバナガリのカテゴリ定義例

```
(category
  ;; A: ANUSVARA or CANDRABINDU
  ;; C: CONSONANT (except for R)
  ;; R: LETTER RA
  ;; V: VOWEL INDEPENDENT
  ;; E: ELSE
  ...
  (0x0901 0x0902 ?A) ; CANDRABINDU, ANUSVARA
  (0x0903 ?E) ; SIGN VISARGA
  (0x0905 0x0914 ?V) ; A .. AU
  (0x0915 0x0939 ?C) ; KA .. HA
  (0x0930 ?R) ; RA
  ...)
```

生成ルール

◆ 構造

```
(generator
  (PATTERN COMMAND ...)
  (MACRO-NAME (PATTERN COMMAND ...)
  ...)
```

◆ PATTERN

◆ COMMAND

- ・出力コード（列）
- ・グリフ合成規則
- ・条件節
- ・生成ルール（再帰的に）
- ・組み込みコマンド
- ・マクロ名

生成ルールの例

ह न् डी ⇒ हिन्डी (ヒンディー)

◆ シラブルの切り出し

ह न् डी ⇒ ह ि | न् डी

◆ 並べ替え

ह ि ⇒ ि ह

◆ 文字列からグリフ列へ

ि ह ⇒ हि | न् ड ⇒ न्द

シラブルの切り出し

ह न् डी ⇒ ह ि | न् डी

```
(generator
  (0
    (cond
      ("VA?S?"
        ...独立母音の処理...)
      ("([RC]H)*[RC](H|[IM]?A?S?)?"
        ...子音列の処理...)
      (". "
        ...単独文字の処理...))
    *)
  ...
```

並べ替え

ह ि ⇒ ि ह

```
...
("([RC]H)*[RC](H|[IM]?A?S?)?"
  (cond
    ...
    ("([^\i]*) (I) (A?S?)"
      (2 vowel-sign-I) ;; 母音 I の処理
      (1 consonant *)  ;; 子音列の処理
      (3 post-modifier) ;; 後置修飾子の処理
    )
  )
  ...
```


文字列からグリフ列へ

̣ ̣ ̣ ⇒ ̣ ̣
 ̣ ̣ ̣ ⇒ ̣ ̣

(vowel-sign-I
 ((0x093F) 0xCA))
 (consonant
 (cond
 ...
 ((0x0921) 0x62 0xF7)
 ((0x0928 0x094D) 0x78)
 ((0x0939) 0xBD)
 ...

デモ

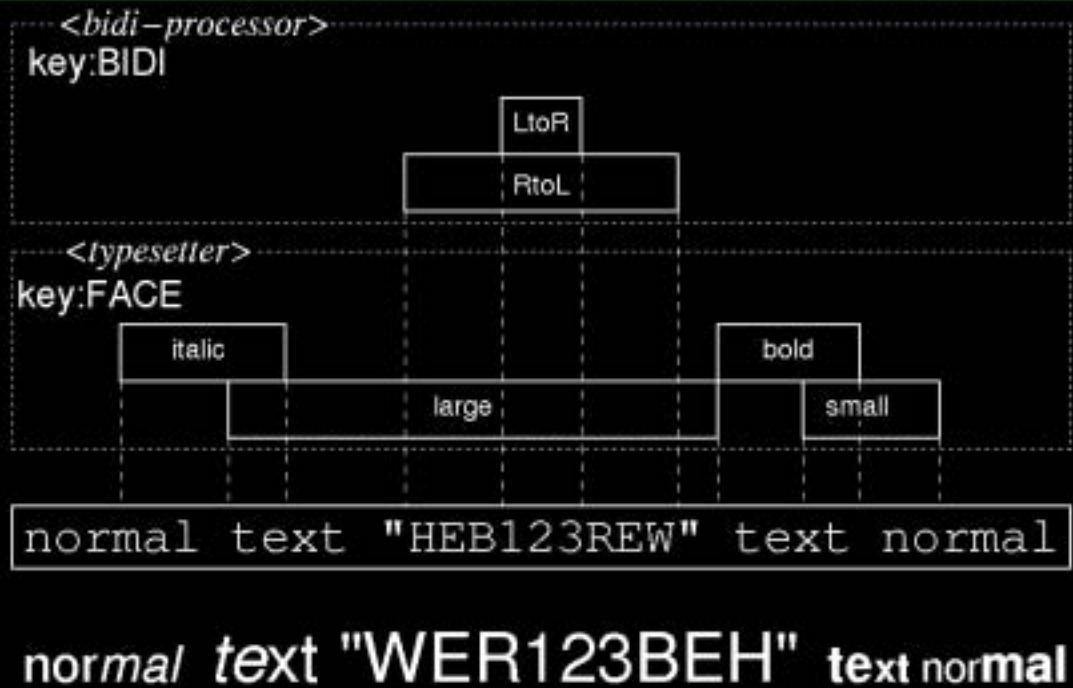
◆ 種々のスクリプトの表示例

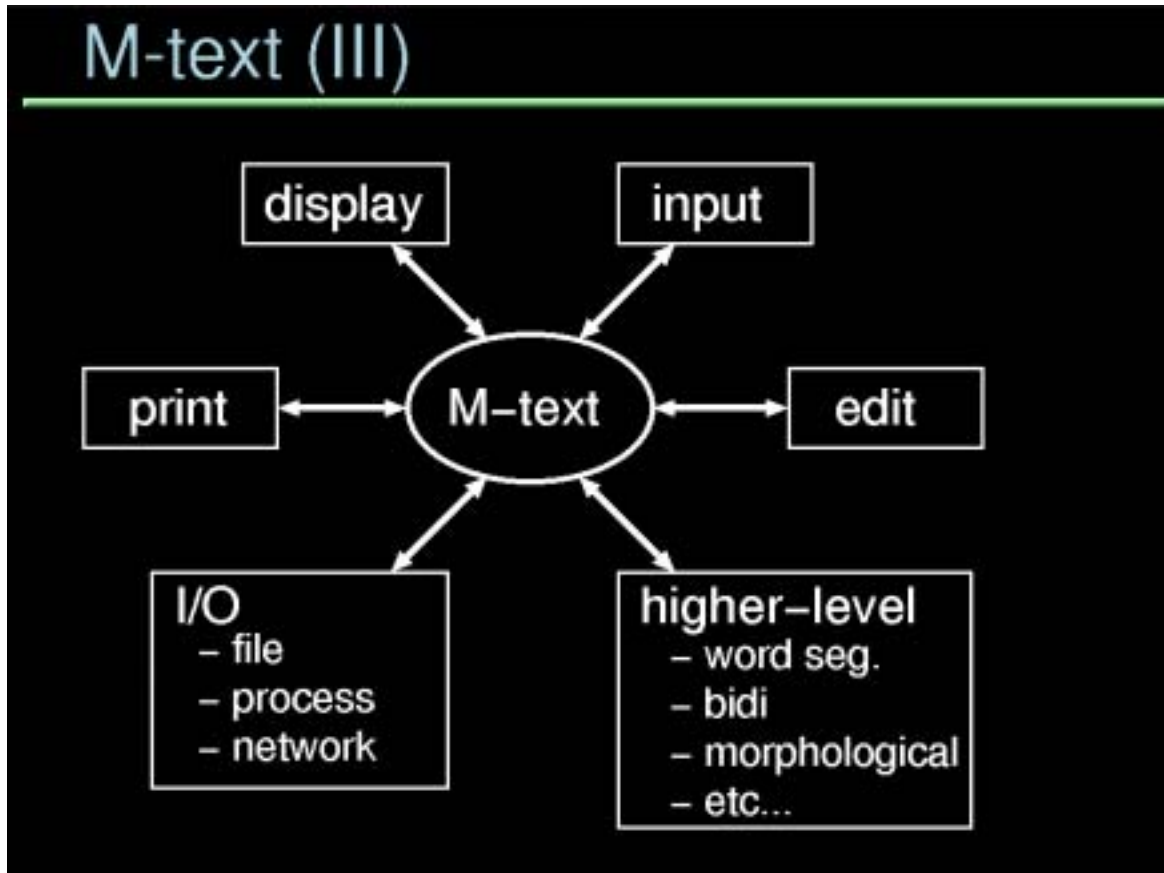
ヨーロッパ		中近東・アフリカ	
フランス語 (Français)	Bonjour, Salut	トルコ語 (Türkçe)	Merhaba
ドイツ語 (Deutsch Süd)	Grüß Gott	アラビア語 (العربية)	السلام عليكم
スロバキア語 (slovensky)	Dobrý deň	ペルシャ語 (فارسی)	سلام عليكم
ロシア語 (Русский)	Здравствуйте!	ヘブライ語 (עברית)	שלום
ギリシャ語 (Ελληνικά)	Γειάσας	アムハラ語 (አማርኛ)	ሰላም
南・東南アジア		東アジア	
ヒンズー語 (हिन्दी)	नमस्ते, नमस्कार ।	日本語	こんにちは
タミール語 (தமிழ்)	வணக்கம்	中国語 (汉语, 普通话)	你好
チベット語 (བོད་སྐད་)	བུ་མཉམ་ལཱེ་ལེགས།	韓国語 (한국)	안녕하세요
クメール語 (ខ្មែរ)	ជំរាបសួរ (ex: ជំរាប)		
ベトナム語 (Tiếng Việt)	Chào bạn		
タイ語 (ภาษาไทย)	สวัสดีครับ, สวัสดีค่ะ		
ラオ語 (ລາວ)	ສະບາຍດີ, ຊື່ສິດຊາດີ		

M-Text (I)

- ◆ 多言語のテキストを表わす構造体
属性付き文字列オブジェクト
- ◆ テキストプロパティ
属性-属性値のペア
M-text の任意の部分に
任意の属性を
幾層にも付加できる
ex. 言語、書字方向、タイプフェース

M-text (II)





- ## 今後：OpenType フォント
- ◆ Unicode エンコーディング
 - ◆ フォント自身が種々の情報を持つ
 - ・グリフ置換
 - ・代替グリフ
 - ・リガチャ
 - ・グリフの2次元配置
 - ◆ 表示エンジンの仕事
 - ・シラブルの切り出し
 - ・並べ替え
 - ・上記情報の利用
 - ◆ フォントコンポーザの単純化

今後：GNU Emacs 等との関連

- ◆ m17nlib は Emacs とは独立したライブラリ
 - ・ Emacs 開発の経験に基づく
 - ・ 表示機能は全く新規に開発
- ◆ 現在 Unicode ベースの Emacs も開発中
 - ・ m17nlib と同等の表示エンジンを搭載予定
 - ・ 言語情報ベースも共有

今後：GNOME

- ◆ 現状
- ◆ 将来

