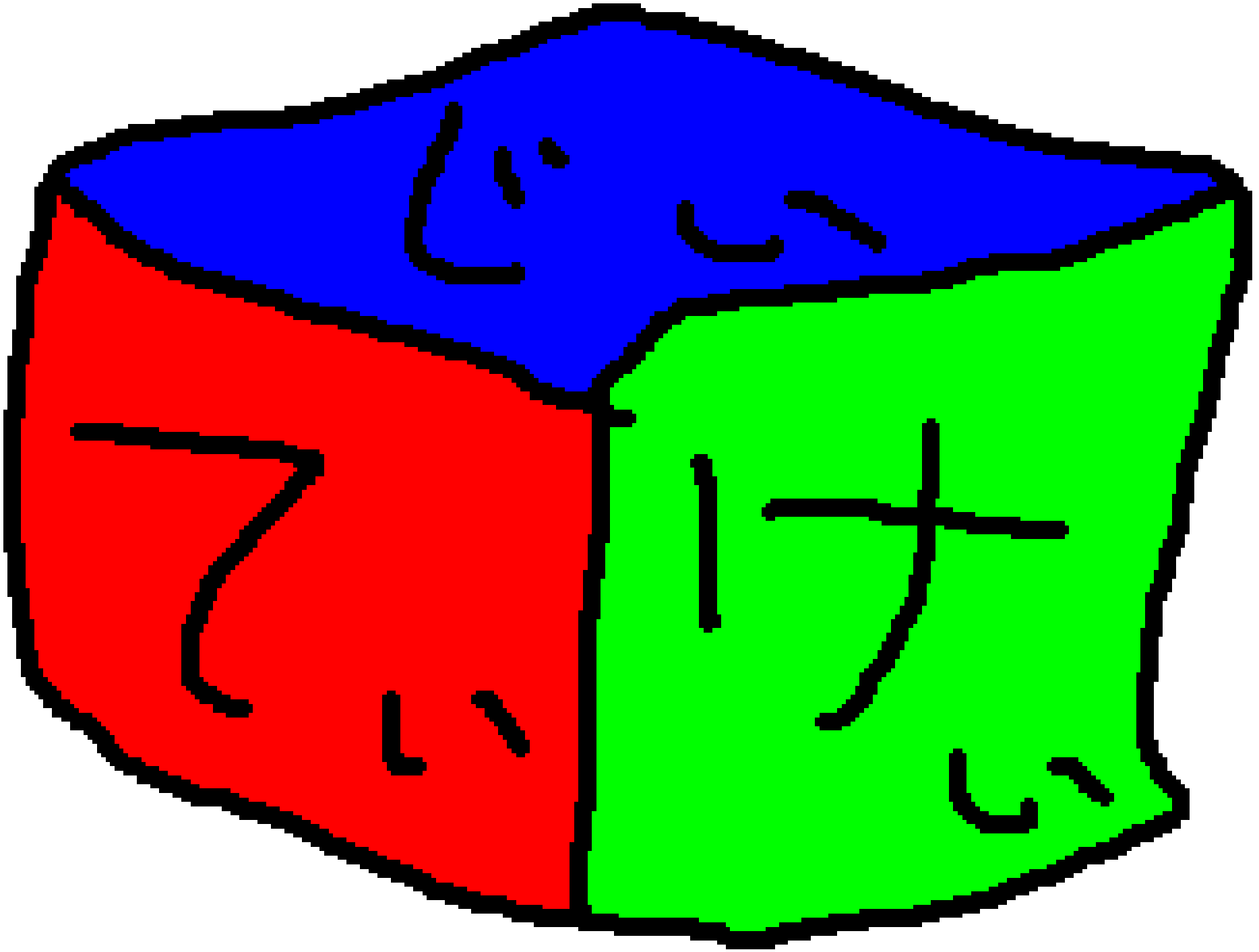
GTK+/GLibのファイル名エンコーディング

岩本 一樹




GTK+

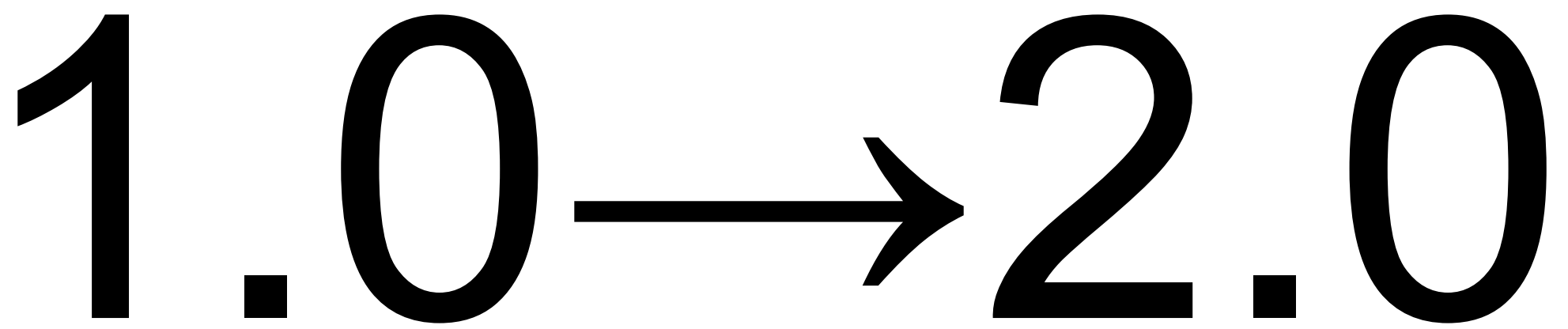
GUI
ツールキット


GLib

GTK+の下で働くライブラリ

2.8.17が最新版

1.0系と
2.0系に
分かれる


1.0⟶2.0

このとき、いくつかの特長がアピールされた

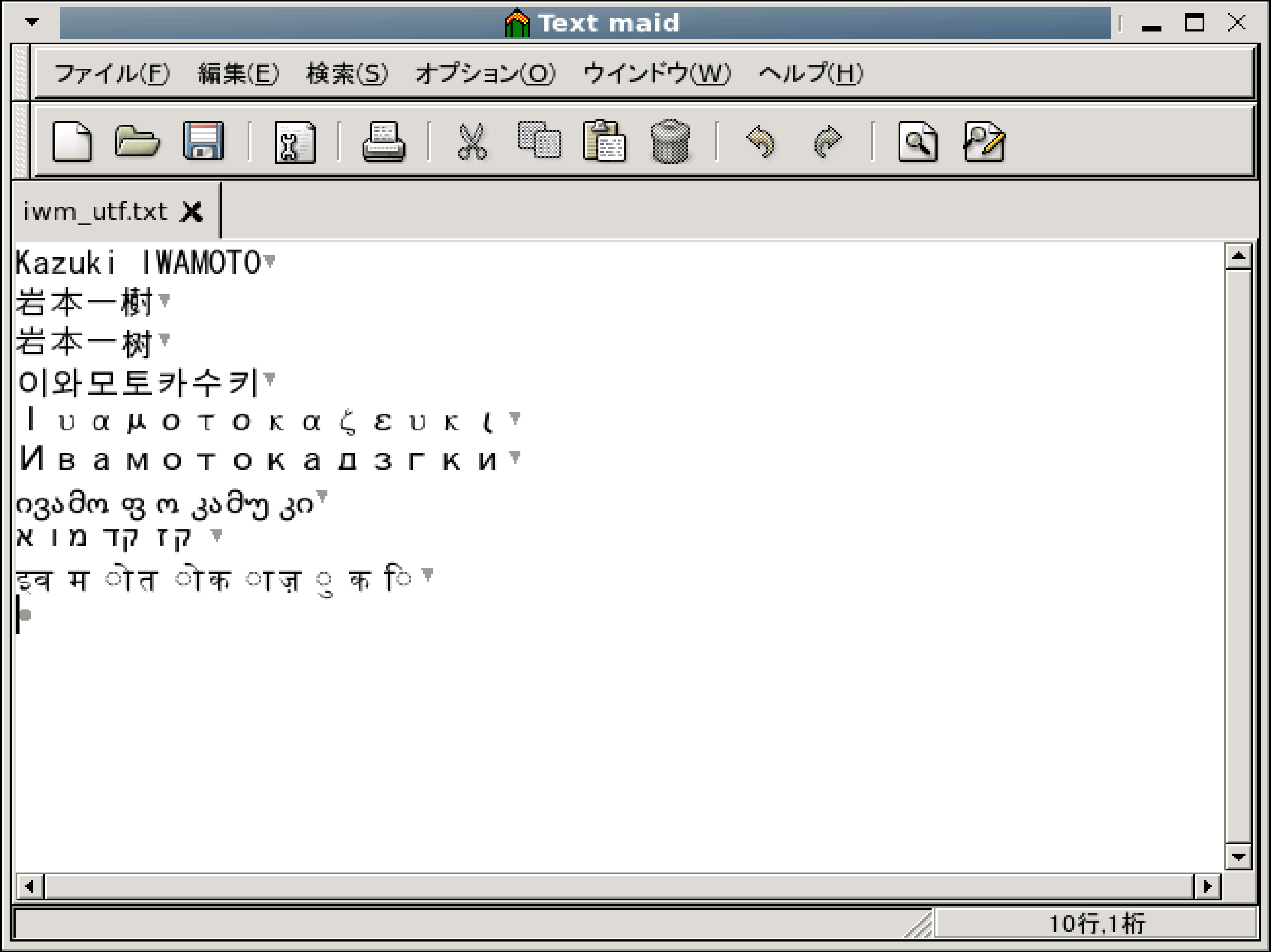
Pangoによる多言語テキスト出力

新テーマエンジン

ATKによるアクセシビリティサポートの向上

UTF-8

例えば
こんな
感じ


Text maid
ファイル(F) 編集(E) 検索(S) オプション(O) ウインドウ(W) ヘルプ(H)
iwm_utf.txt
Kazuki IWAMOTO
岩本一樹
岩本一树
이와모토카수키
Ι υ α μ ο τ ο κ α ζ ε υ κ ι
И в а м о т о к а д з г к и
10行,1桁

すべて
UTF-8に?

例えば…

GtkFileChooser

開くファイルを選択したり入力するGUI


開く
ホーム
デスクトップ
ファイルシステム
ホーム
test
名前
最終変更日
abc
2006年05月06日
xyz
2006年05月06日
岩本一樹
2006年05月06日
追加(A)
削除(R)
All Files
キャンセル(C)
開く(O)

ファイル名を取得するAPI
gtk_file_chooser_get_filename

ファイル名を設定するAPI
gtk_file_chooser_set_filename

これらのAPIの文字列
≠UTF-8

GTK+/GLib以外のAPI
例えば...

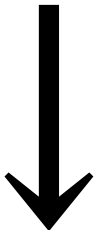
fopen

open

stat

readdir

gtk_file_chooser_get_filename
gtk_file_chooser_set_filename



これらのAPIで使える文字列

この文字列の
文字符号化方式を
file name encoding
と呼ぶ

file name encoding の実体は?

GLib-2.4以降
環境変数
G_FILENAME_ENCODING

G_FILENAME_ENCODING=ISO-8859-1

GLib-2.6以降ならば
複数指定可能

G_FILENAME_ENCODING=EUC-JP,CP932

GLib-2.4で
複数指定すると
先頭のみ有効

@localeという特殊な設定も

G_FILENAME_ENCODING=@locale,CP932

ロケールの文字符号化方式

その実体は？

優先順位が高い順に...

nl_langinfoの返値
setlocaleの返値
環境変数LC_ALL
環境変数LC_CTYPE
環境変数LANG

いずれか1つ

環境変数G_FILENAME_ENCODINGが未定義or GLib-2.2以前

環境変数
G_BROKEN_FILENAMES

g_get_charsetの返値

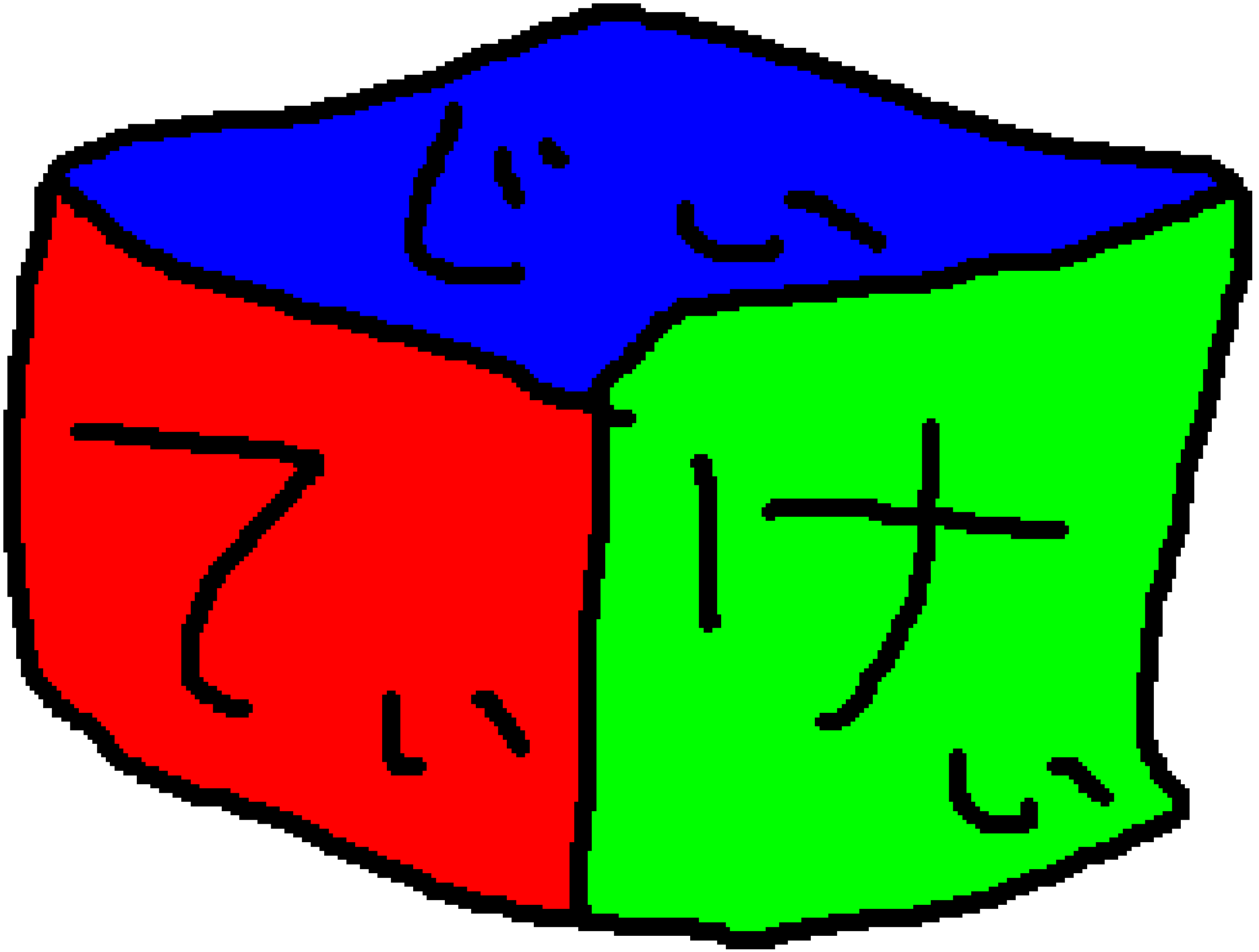
即ち、ロケールの文字符号化方式

環境変数G_FILENAME_ENCODING未定義
環境変数G_BROKEN_FILENAMES未定義


UTF-8

たなだし

Microsoft Windows では別



file name encodingの変換

file name ncoding からUTF-8に変換する g_filename_to_utf8

file name ncoding をUTF-8に変換する g_filename_from_utf8

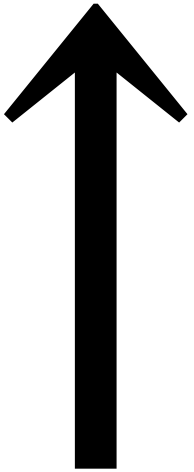
2.6以降

g_filename_display_name
g_filename_display_basename

g_filename_to_utf8
g_filename_display_name
の違い

2.6以降
g_get_filename_charsets

g_get_filename_charsets



環境変数
G_FILENAME_ENCODING

環境変数
G_FILENAME_ENCODING
で複数指定
↓
複数の文字符号化方式を返す

g_filename_to_utf8
最初だけ

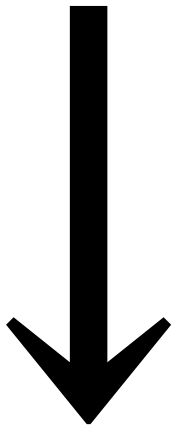
g_filename_display_name
順番に試みる

順番に試みる
=変換に失敗したら次の文字符号化方式を試す

g_filename_display_name
の方が変換できる
可能性が高い

g_filename_display_name
では元の文字符号
化方式が不明

file name encoding



UTF-8

一方通行

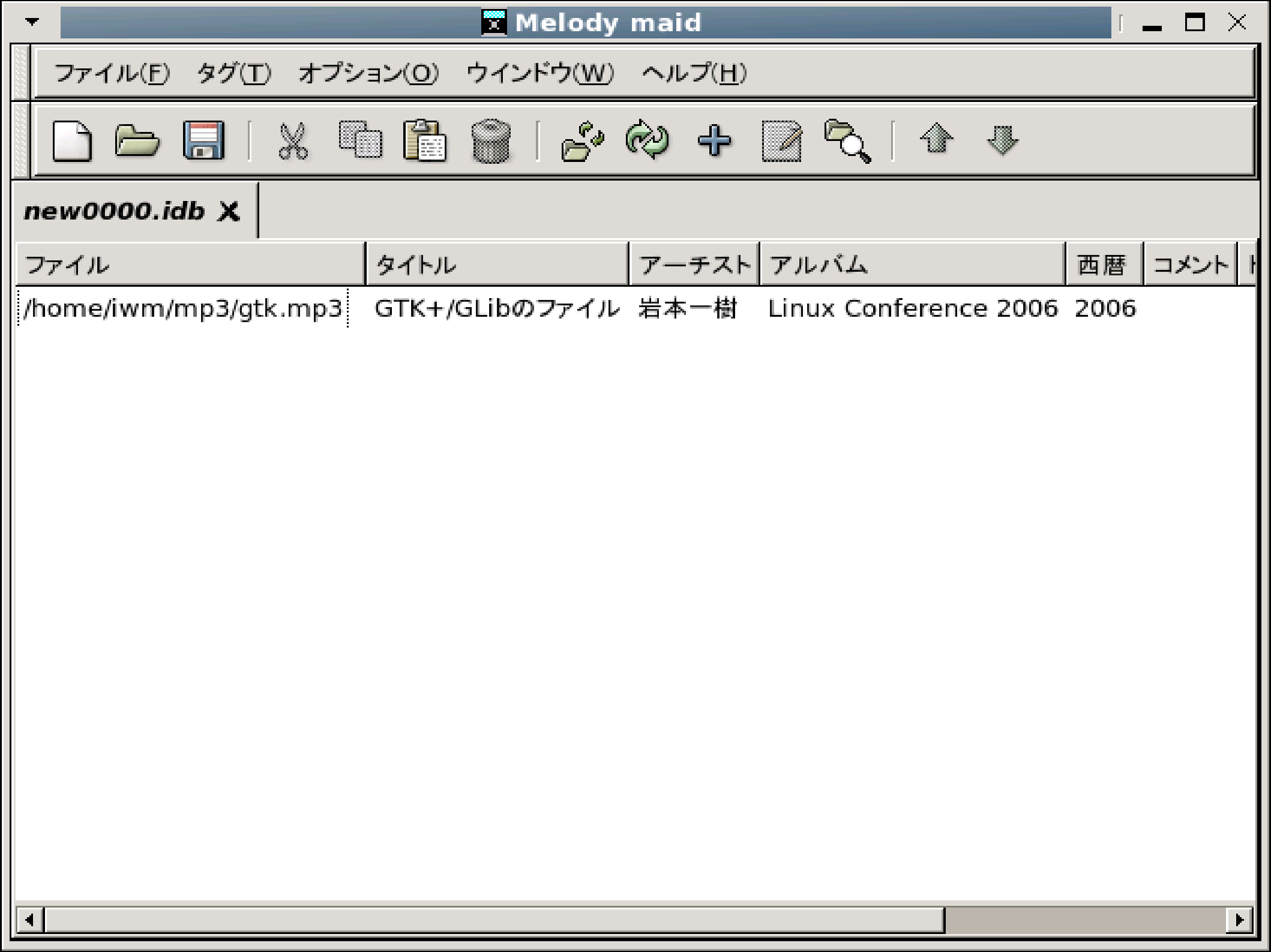
g_filename_display_name
表示のための変換

では
g_filename_to_utf8
と
g_filename_from_utf8
の使い所は?

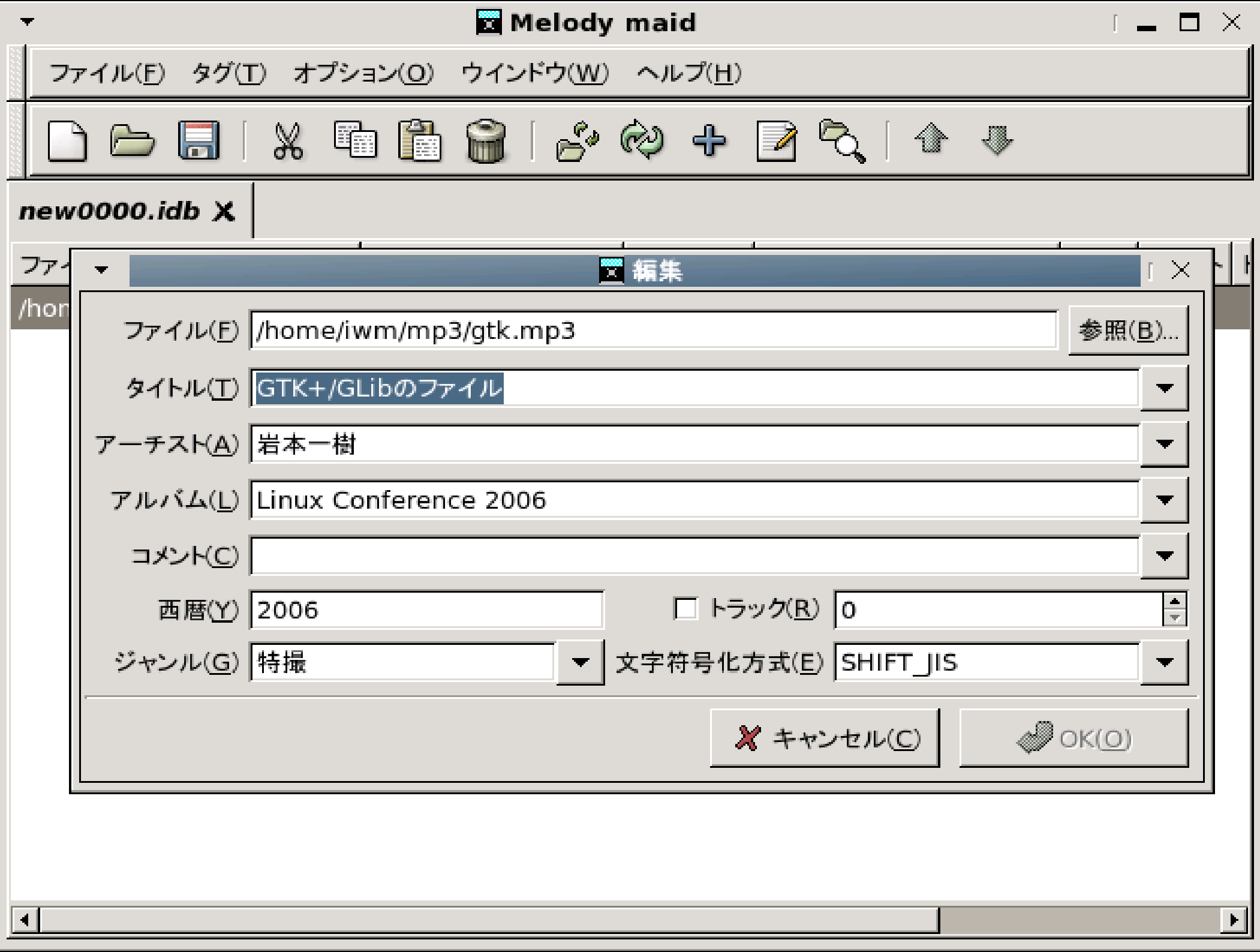
独自にファイル名を入力する場合

HDDオーディオプレイヤーのデータ管理ソフトの場合

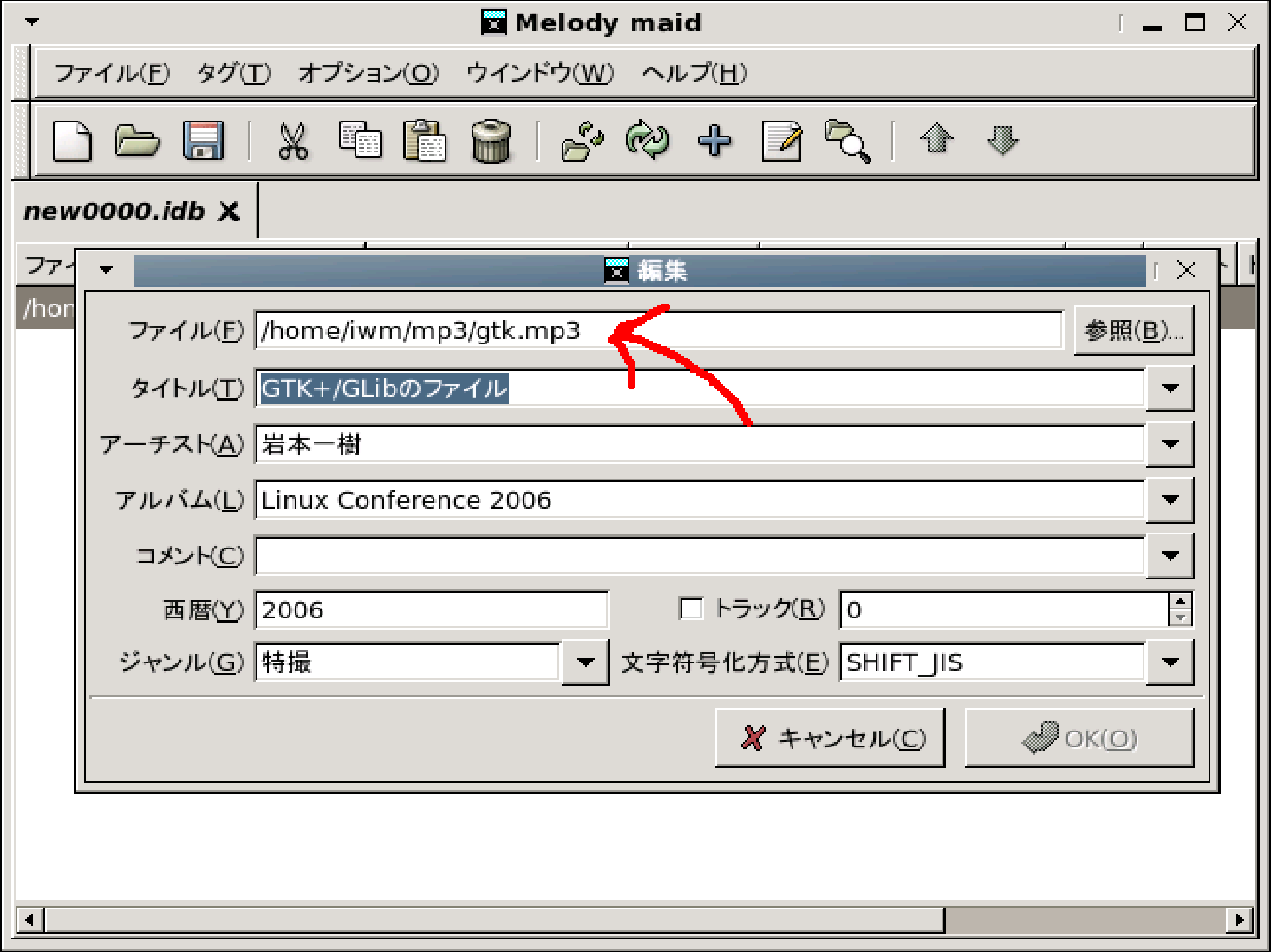
曲(ファイル)の
リストを作ります


Melody maid
ファイル(F) タグ(T) オプション(O) ウインドウ(W) ヘルプ(H)
new0000.idb
ファイル
タイトル
アーチスト
アルバム
西暦
コメント
/home/iwm/mp3/gtk.mp3
GTK+/GLibのファイル
岩本一樹
Linux Conference 2006
2006

リスト項目を編集するダイアログ


Melody maid
ファイル(F) タグ(T) オプション(O) ウインドウ(W) ヘルプ(H)
new0000.idb
編集
ファイル(F)
/home/iwm/mp3/gtk.mp3
参照(B)...
タイトル(T)
GTK+/GLibのファイル
アーチスト(A)
岩本一樹
アルバム(L)
Linux Conference 2006
コメント(C)
西暦(Y)
2006
トラック(R)
0
ジャンル(G)
特撮
文字符号化方式(E)
SHIFT_JIS
キャンセル(C)
OK(O)

このダイアログの
ファイル名を
入力する部分

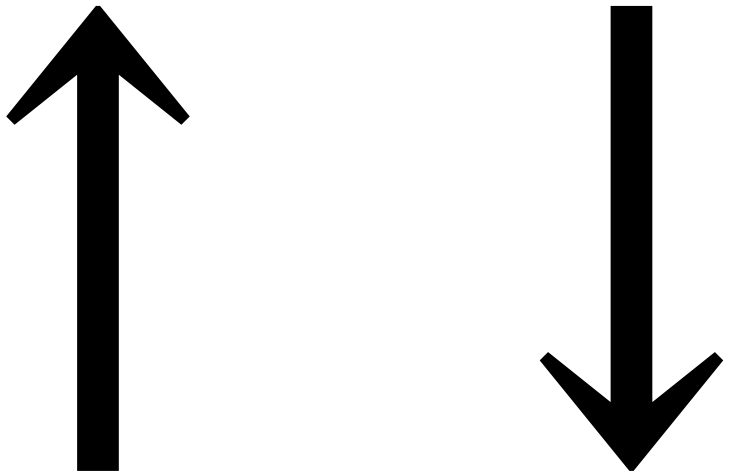

Melody maid
ファイル(F) タグ(T) オプション(O) ウインドウ(W) ヘルプ(H)
new0000.idb
編集
ファイル(F)
/home/iwm/mp3/gtk.mp3
参照(B)...
タイトル(T)
GTK+/GLibのファイル
アーチスト(A)
岩本一樹
アルバム(L)
Linux Conference 2006
コメント(C)
西暦(Y)
2006
トラック(R)
0
ジャンル(G)
特撮
文字符号化方式(E)
SHIFT_JIS
キャンセル(C)
OK(O)

1.file name encodingをUTF-8にする

2.エントリーウィジェットで編集

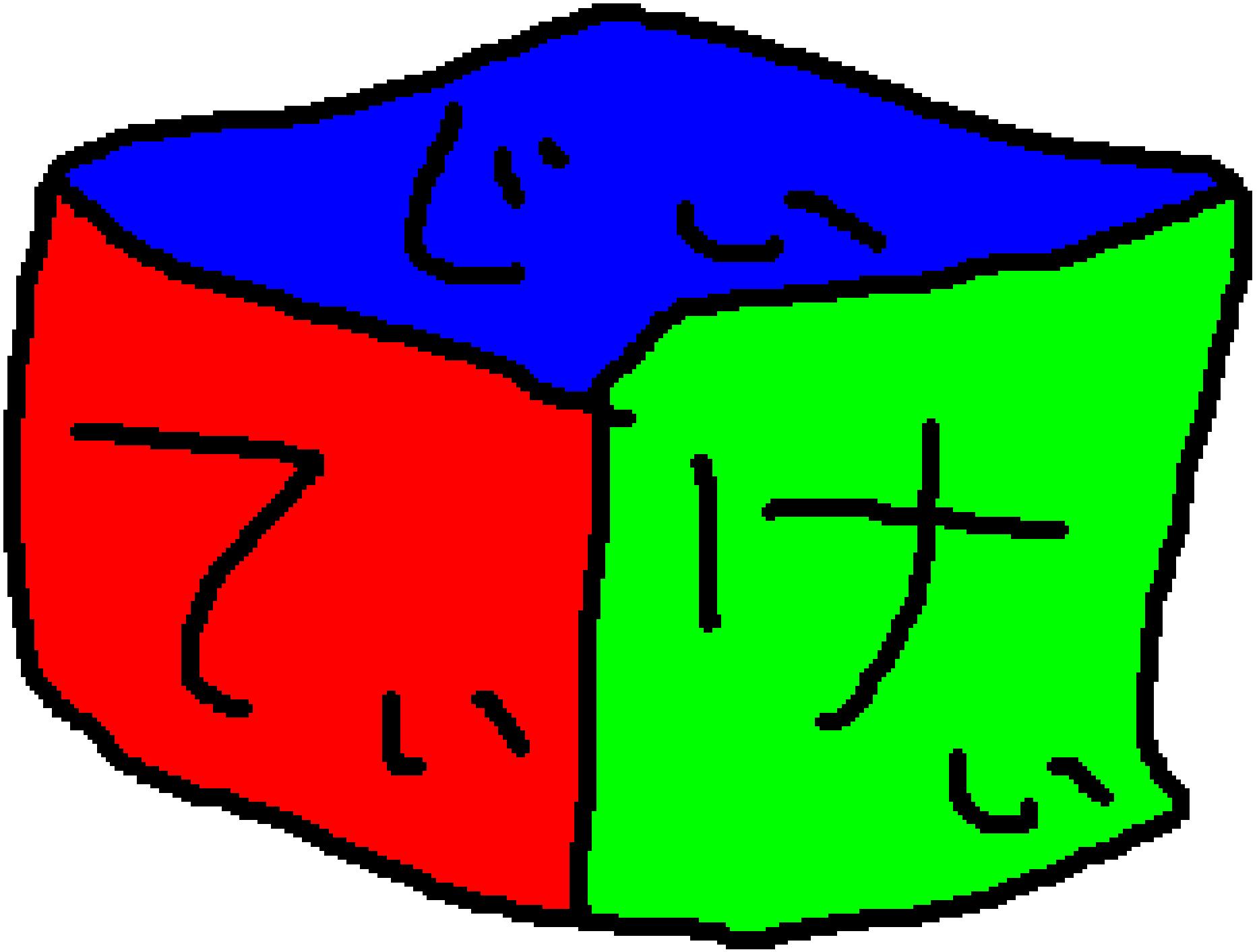
3.UTF-8をfile name encodingにする

g_filename_to_utf8



g_filename_from_utf8

可逆?



変換の実装は
libiconv

例えば

CP932→UTF-8→CP932

CP932
ED 61

UTF-8

E5 83 98

CP932
FA 7D

ED 61

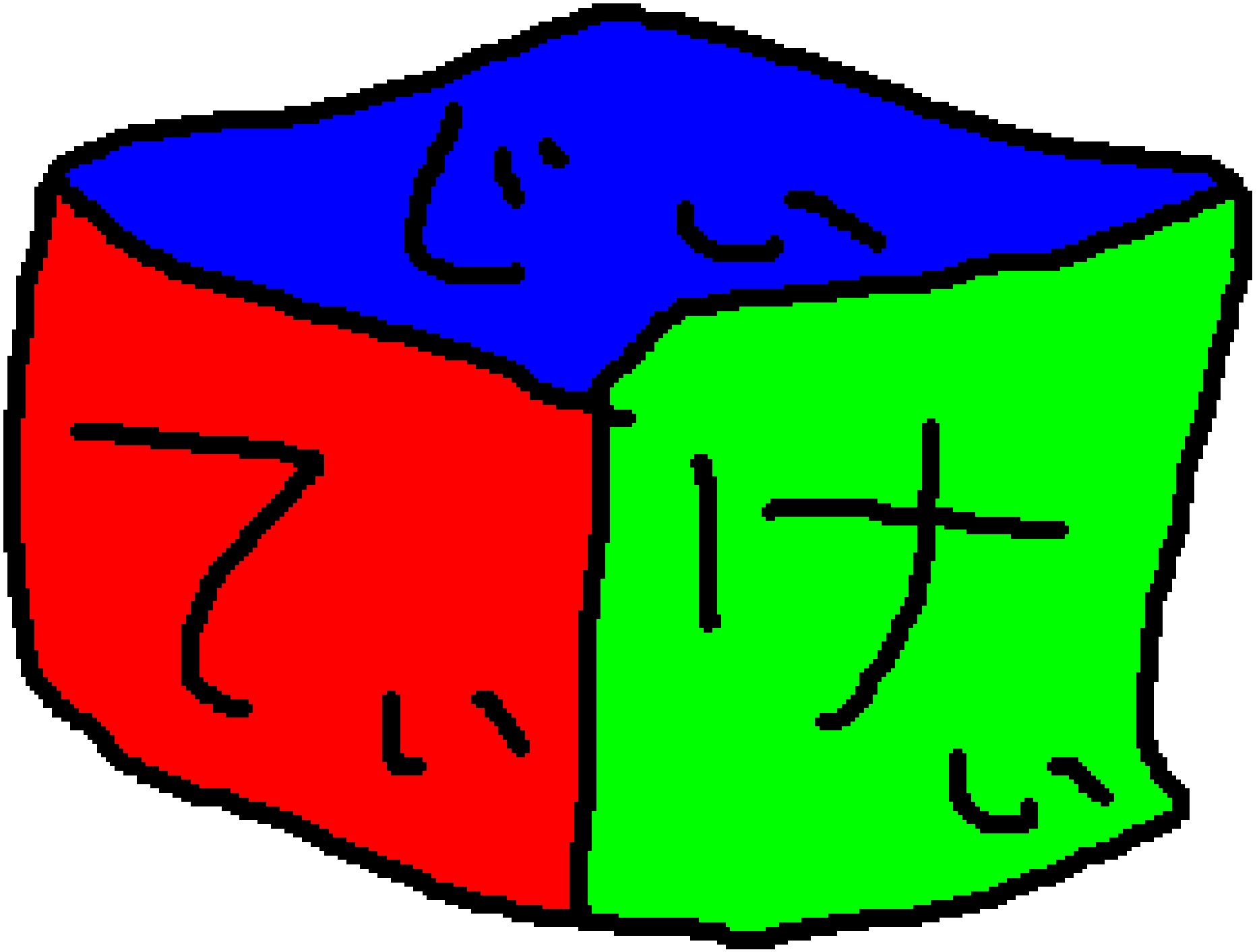
↓

FA 7D

CP932は
歴史的経緯により
同じ文字が重複し
て登録されている

NECのPC-9801とIBMのDOS/Vのそれぞれの文字を統合した

変換で元
に戻るとは
限らない



UTF-8と
file name encoding
が混在

混在＝面倒

gtk_file_chooser_get_filename
g_dir_read_name

↓

取得

取得

↓

即UTF-8に変換

&file name encoding破棄

fopen
open
stat
readdir
を呼び出す時

必要に応じて
その場でUTF-8から
file name encoding
へ変換

不要になったら
file name encoding
は即破棄

文字列は
すべて
UTF-8
で統一

統一＝楽

しかし

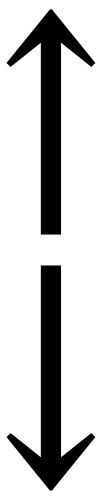
UTF-8に変換できない可能性

環境変数
G_FILENAME_ENCODING
や
G_BROKEN_FILENAME
が正しくない

libiconvが
file name encoding
と
UTF-8
の変換に未対応

対応する
文字が
ない

変換できない



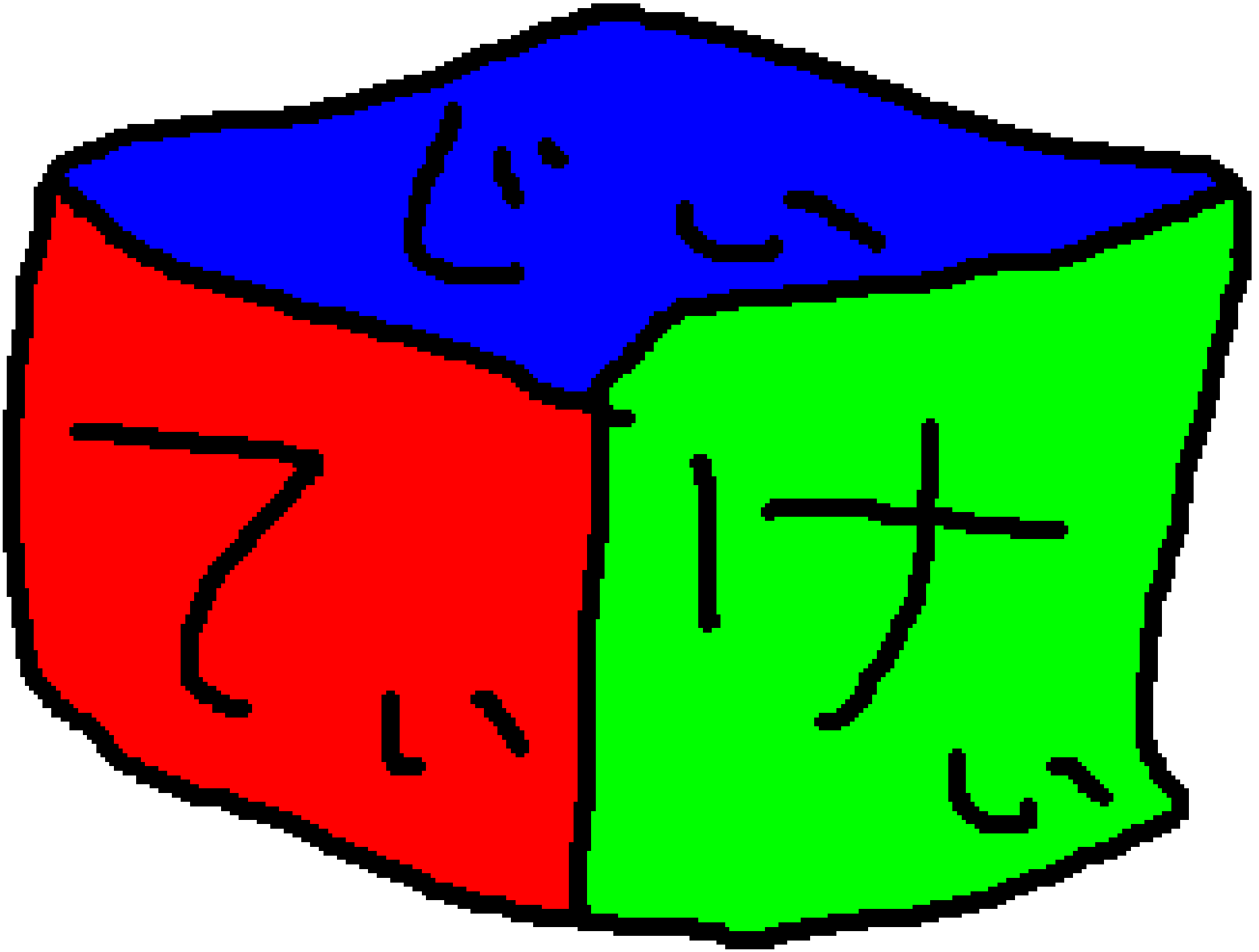
不正なファイル名？

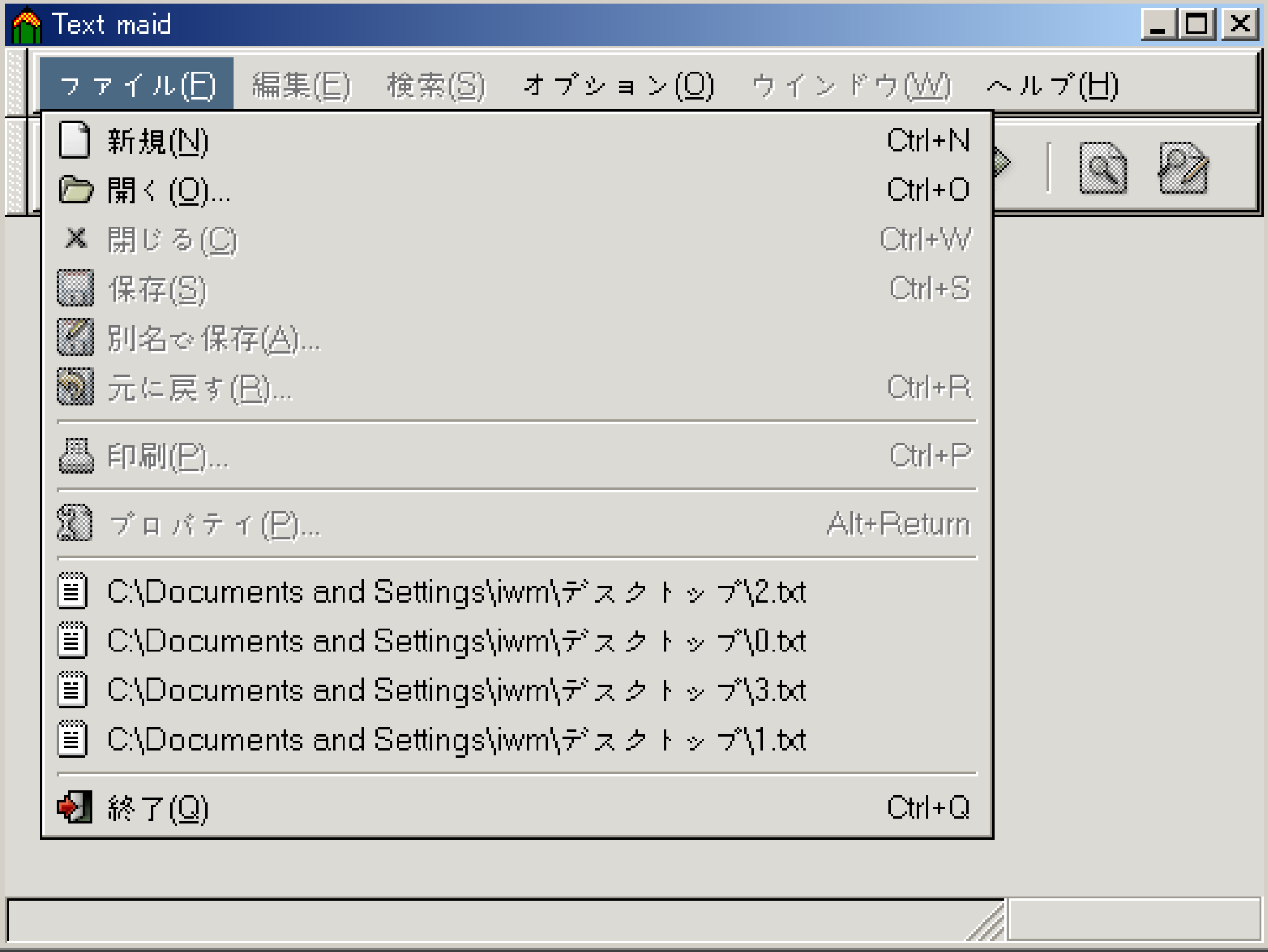
UTF-8に変換できなくてもファイル名として使える

ゆえに
file name encoding
のまま保持

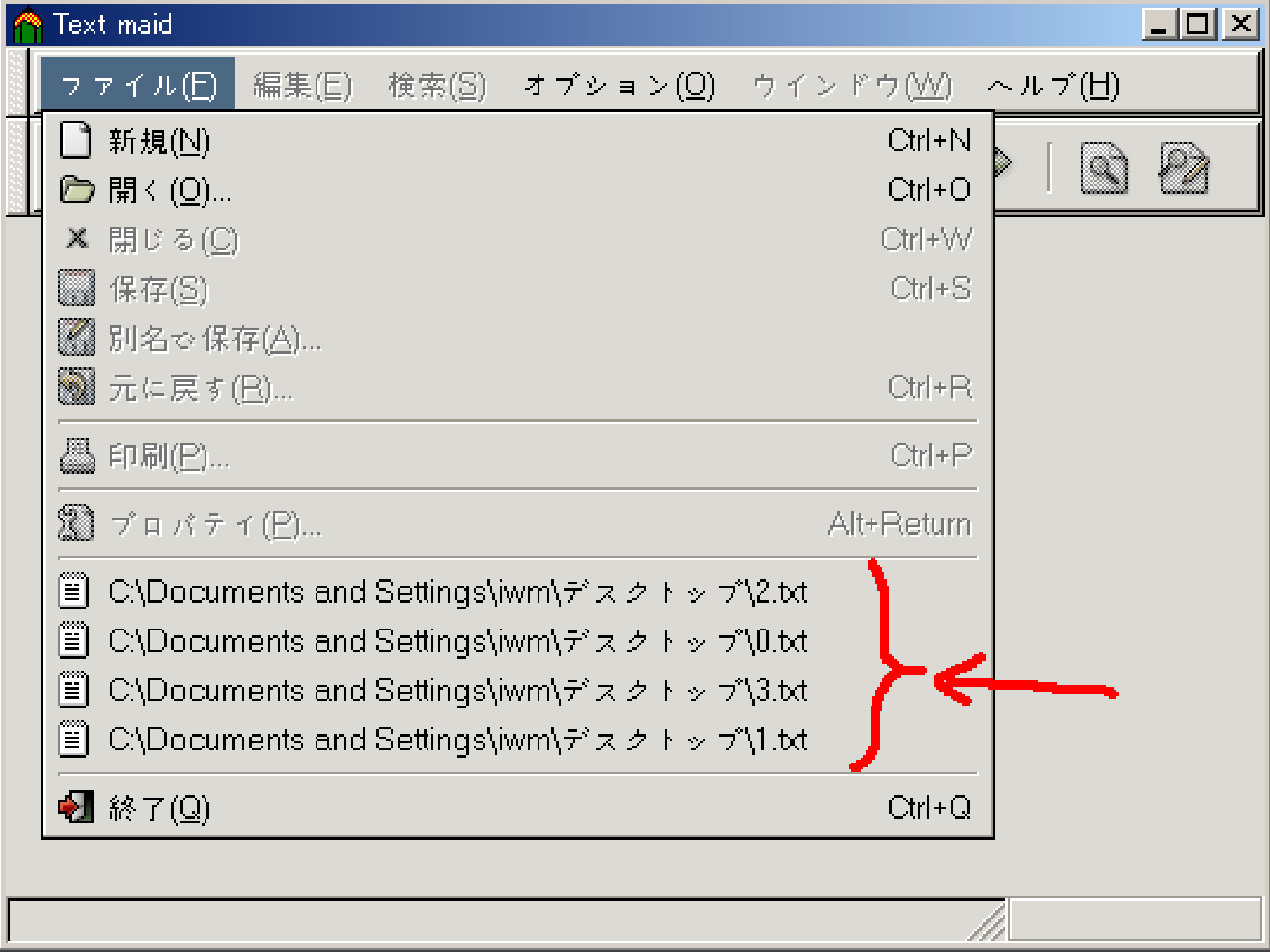
表示する時

必要に応じて
その場で
file name encoding
からUTF-8へ変換

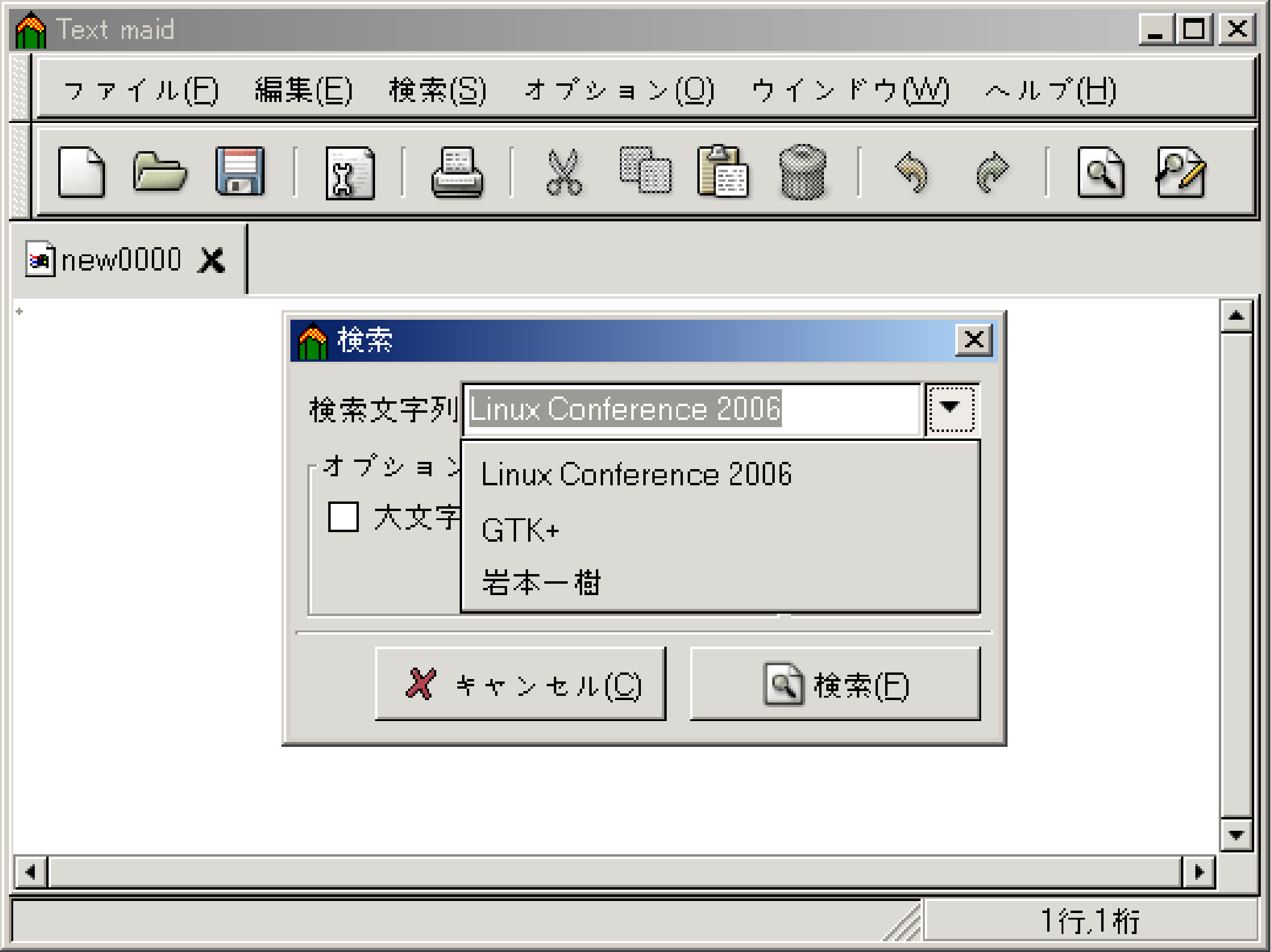



Text maid
ファイル(F)
編集(E)
検索(S)
オプション(O)
ウィンドウ(W)
ヘルプ(H)
新規(N) Ctrl+N
開く(O)... Ctrl+O
閉じる(C) Ctrl+W
保存(S) Ctrl+S
別名で保存(A)...
元に戻す(R)... Ctrl+R
印刷(P)... Ctrl+P
プロパティ(P)... Alt+Return
C:\Documents and Settings\iwm\デスクトップ\2.txt
C:\Documents and Settings\iwm\デスクトップ\0.txt
C:\Documents and Settings\iwm\デスクトップ\3.txt
C:\Documents and Settings\iwm\デスクトップ\1.txt
終了(Q) Ctrl+Q

ファイルの履歴


Text maid
ファイル(F)
編集(E)
検索(S)
オプション(O)
ウィンドウ(W)
ヘルプ(H)
新規(N) Ctrl+N
開く(O)... Ctrl+O
閉じる(C) Ctrl+W
保存(S) Ctrl+S
別名で保存(A)...
元に戻す(R)... Ctrl+R
印刷(P)... Ctrl+P
プロパティ(P)... Alt+Return
C:\Documents and Settings\iwm\デスクトップ\2.txt
C:\Documents and Settings\iwm\デスクトップ\0.txt
C:\Documents and Settings\iwm\デスクトップ\3.txt
C:\Documents and Settings\iwm\デスクトップ\1.txt
終了(Q) Ctrl+Q

検索の履歴


Text maid
ファイル(F)　編集(E)　検索(S)　オプション(O)　ウィンドウ(W)　ヘルプ(H)
new0000
検索
検索文字列
Linux Conference 2006
オプション
大文字
Linux Conference 2006
GTK+
岩本一樹
キャンセル(C)
検索(F)
1行,1桁

これらは
再起動しても
保存される

通常はファイルに保存

ホームディレクトリの
.tmaid
とか

.tmaidとかはテキスト形式

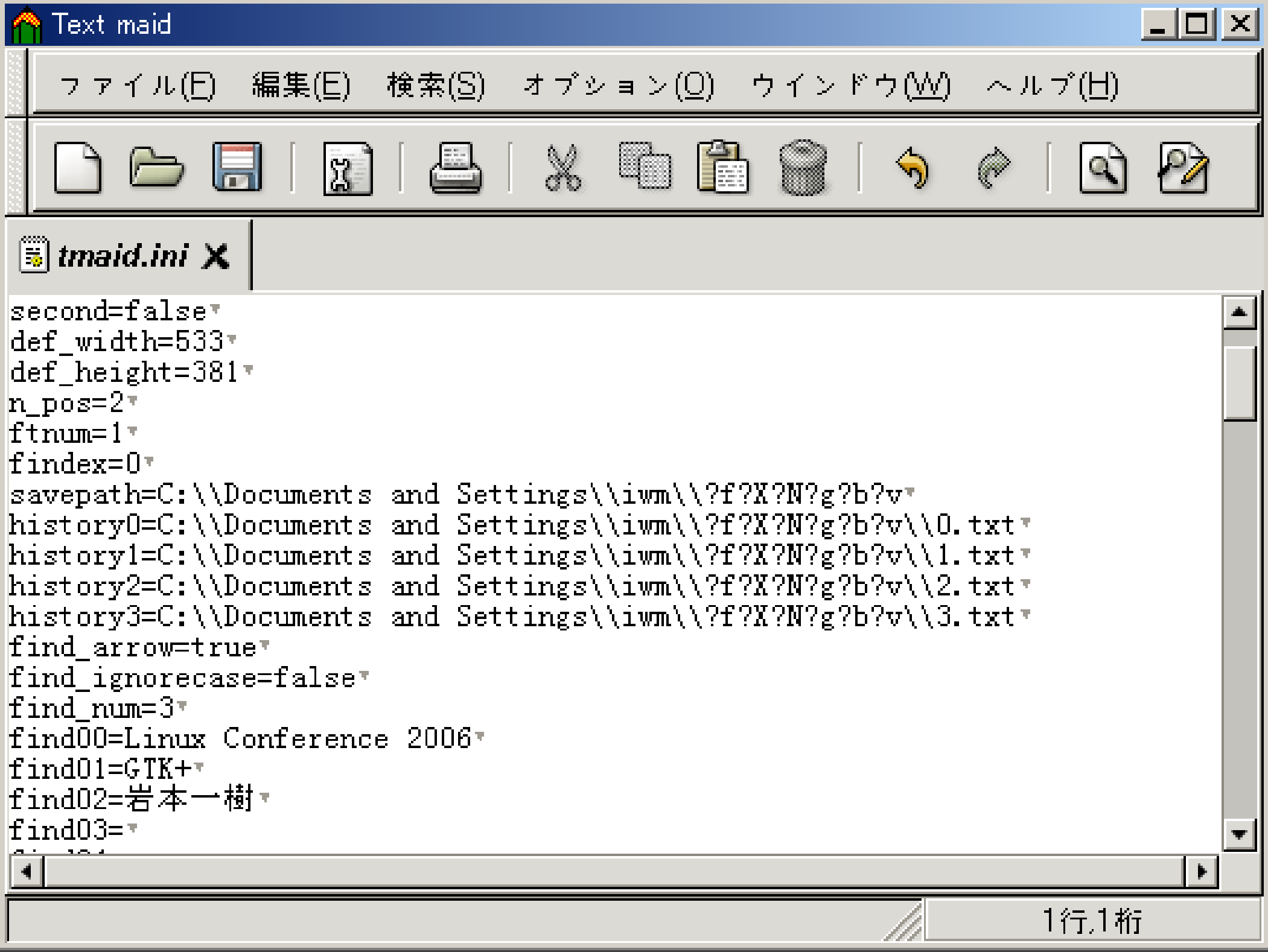
文字符号化方式は?

ファイルの履歴
=file name encoding

検索の履歴
=UTF-8

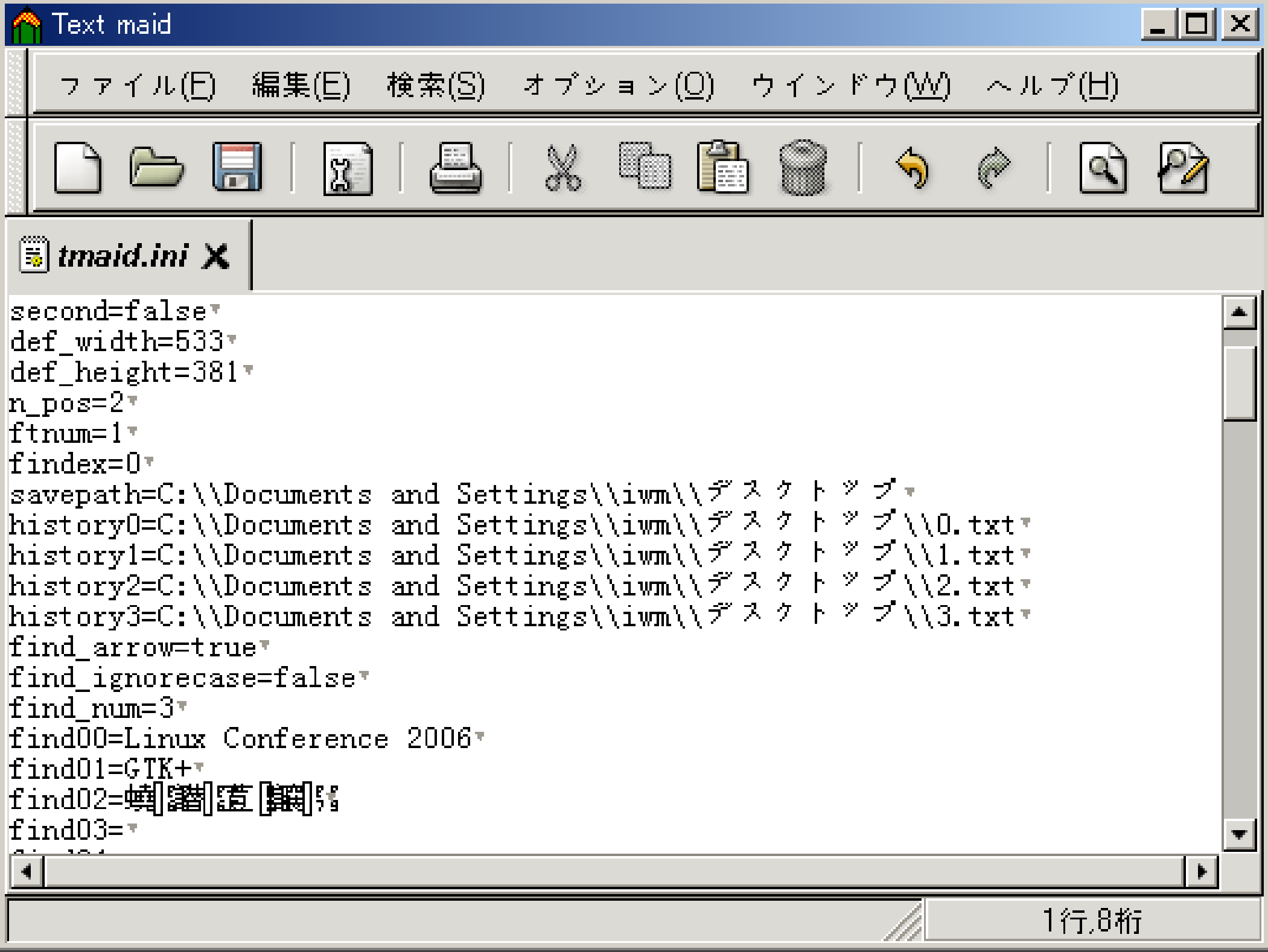
そのままファイルに書き出す

UTF-8で開く


Text maid
ファイル(F) 編集(E) 検索(S) オプション(O) ウィンドウ(W) ヘルプ(H)
tmaid.ini
second=false
def_width=533
def_height=381
n_pos=2
ftnum=1
findex=0
savepath=C:\\Documents and Settings\\iwm\\?f?X?N?g?b?v
history0=C:\\Documents and Settings\\iwm\\?f?X?N?g?b?v\\0.txt
history1=C:\\Documents and Settings\\iwm\\?f?X?N?g?b?v\\1.txt
history2=C:\\Documents and Settings\\iwm\\?f?X?N?g?b?v\\2.txt
history3=C:\\Documents and Settings\\iwm\\?f?X?N?g?b?v\\3.txt
find_arrow=true
find_ignorecase=false
find_num=3
find00=Linux Conference 2006
find01=GTK+
find02=岩本一樹
find03=
1行,1桁

file name encodingの文字列が文字化け

file name encoding
で開く

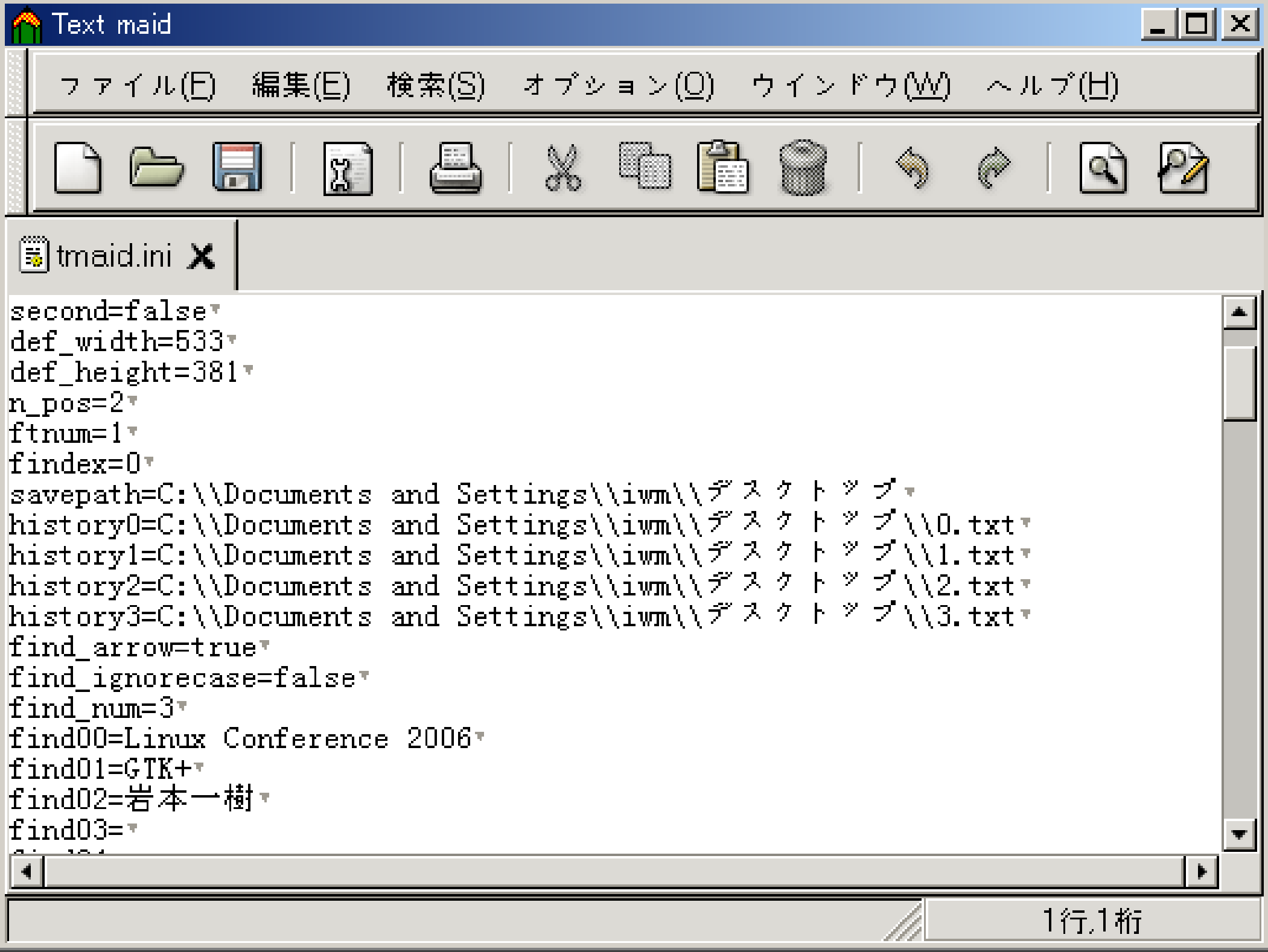

Text maid
ファイル(F) 編集(E) 検索(S) オプション(O) ウィンドウ(W) ヘルプ(H)
tmaid.ini
second=false
def_width=533
def_height=381
n_pos=2
ftnum=1
findex=0
savepath=C:\\Documents and Settings\\iwm\\デスクトップ
history0=C:\\Documents and Settings\\iwm\\デスクトップ\\0.txt
history1=C:\\Documents and Settings\\iwm\\デスクトップ\\1.txt
history2=C:\\Documents and Settings\\iwm\\デスクトップ\\2.txt
history3=C:\\Documents and Settings\\iwm\\デスクトップ\\3.txt
find_arrow=true
find_ignorecase=false
find_num=3
find00=Linux Conference 2006
find01=GTK+
find02=
find03=
1行,8桁

UTF-8の文字列が文字化け

閲覧困難

編集困難

file name encoding
をUTF-8に変換して
ファイルに書き出す


Text maid
ファイル(F) 編集(E) 検索(S) オプション(O) ウィンドウ(W) ヘルプ(H)
tmaid.ini
second=false
def_width=533
def_height=381
n_pos=2
ftnum=1
findex=0
savepath=C:\\Documents and Settings\\iwm\\デスクトップ
history0=C:\\Documents and Settings\\iwm\\デスクトップ\\0.txt
history1=C:\\Documents and Settings\\iwm\\デスクトップ\\1.txt
history2=C:\\Documents and Settings\\iwm\\デスクトップ\\2.txt
history3=C:\\Documents and Settings\\iwm\\デスクトップ\\3.txt
find_arrow=true
find_ignorecase=false
find_num=3
find00=Linux Conference 2006
find01=GTK+
find02=岩本一樹
find03=
1行,1桁

人にやさしい

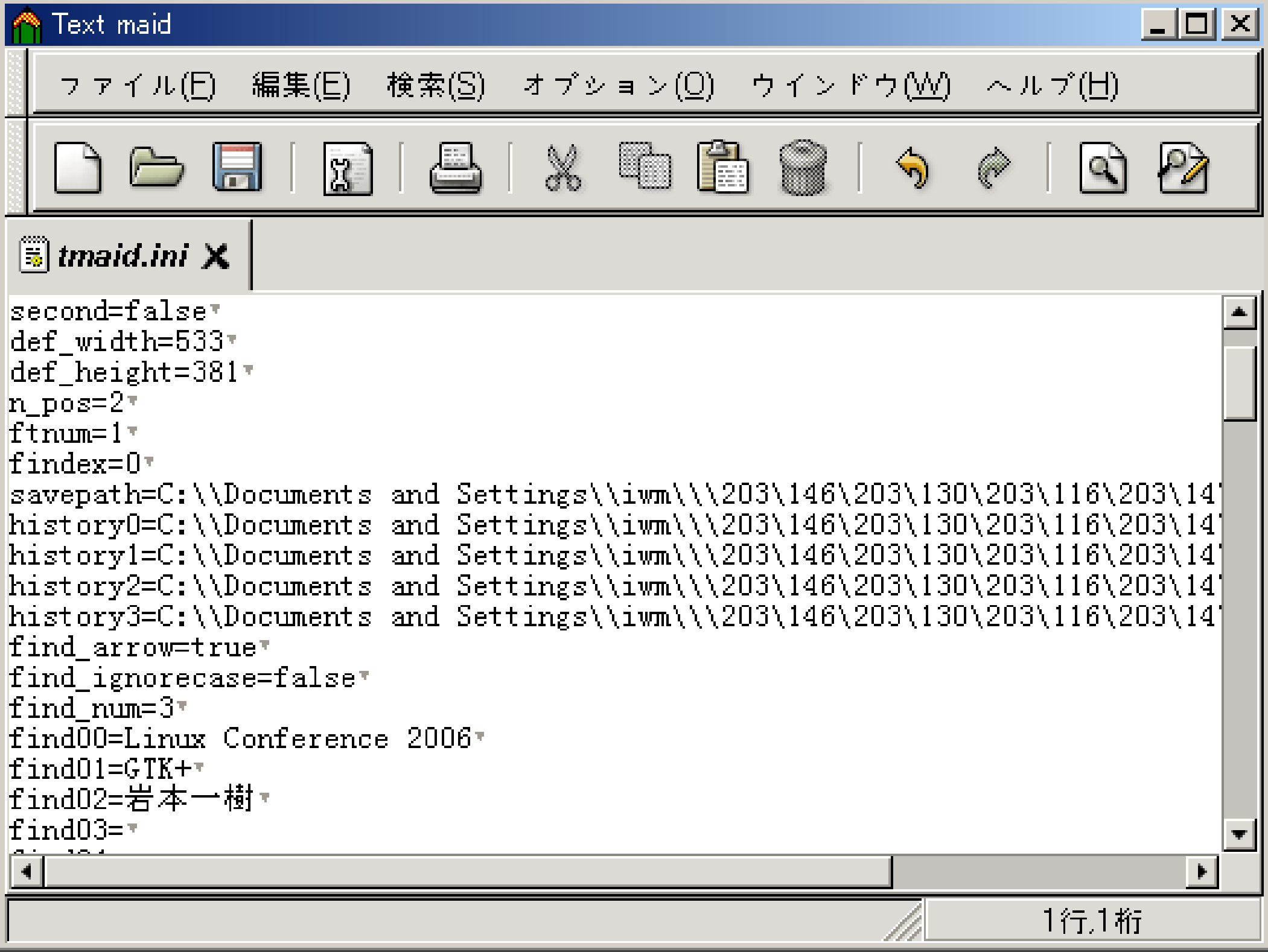
しかし前述の変換に関する問題が…

g_strescapeとg_strcompressを使う

g_strescapeは
7FhからFFhを
8進数表記に変換

08h \b
09h \t
0Ah \n
0Ch \f
0Dh \r
22h \"
5Ch \\
も変換

g_strcompressは
その反対の変換


Text maid
ファイル(F)　編集(E)　検索(S)　オプション(O)　ウィンドウ(W)　ヘルプ(H)
tmaid.ini
second=false
def_width=533
def_height=381
n_pos=2
ftnum=1
findex=0
savepath=C:\\Documents and Settings\\iwm\\\\203\146\203\130\203\116\203\14
history0=C:\\Documents and Settings\\iwm\\\\203\146\203\130\203\116\203\14
history1=C:\\Documents and Settings\\iwm\\\\203\146\203\130\203\116\203\14
history2=C:\\Documents and Settings\\iwm\\\\203\146\203\130\203\116\203\14
history3=C:\\Documents and Settings\\iwm\\\\203\146\203\130\203\116\203\14
find_arrow=true
find_ignorecase=false
find_num=3
find00=Linux Conference 2006
find01=GTK+
find02=岩本一樹
find03=
1行,1桁

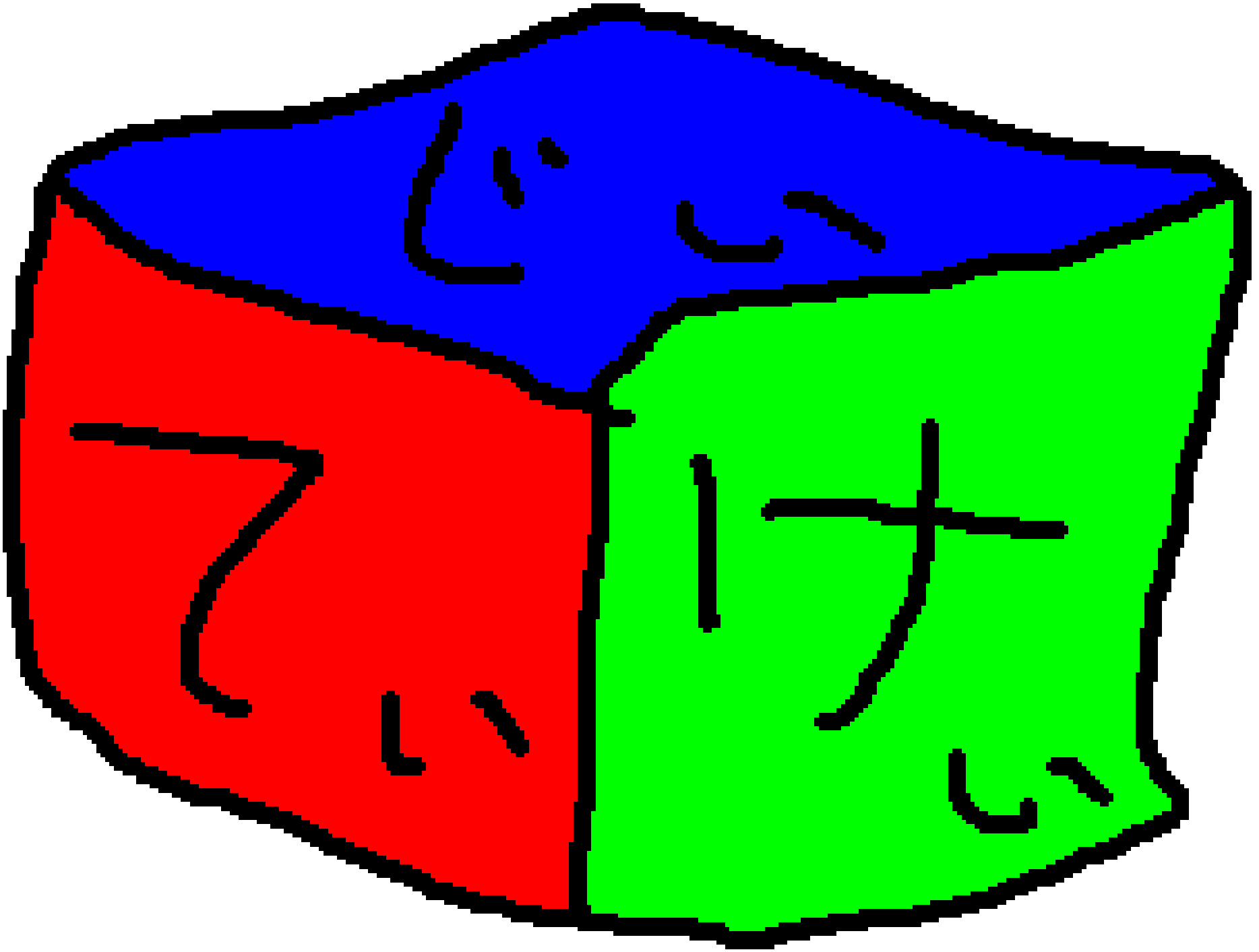
変換に関する問題は生じない

閲覧/編集はやや困難

1.そのまま
2.UTF-8に変換
3.g_strescape

どの方法が正しいとは言えない

状況に応じて
選択する必要あり



Microsoft Windows

文字列を扱うAPI

ANSIコードページとUNICODEの2種類

ソースコードレベルでは
CreateFile

実際には
CreateFileA
または
CreateFileW

ヘッダで決まる

```
#ifdef UNICODE
#define CreateFile CreateFileW
#else
#define CreateFile CreateFileA
#endif
```

マクロUNICODEが定義されていれば
CreateFileはCreateFileW

マクロUNICODEが定義されなければ
CreateFileはCreateFileA

ANSIコードページは
8ビット単位

1文字は1バイト
または2バイト

実際には
CP932など
言語環境によって
異なる

UNICODEは
16ビット単位

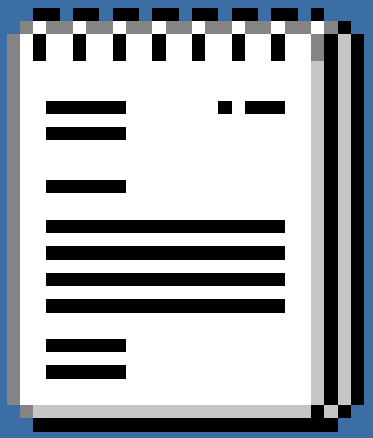
1文字は2バイト

言語環境による
違いはない

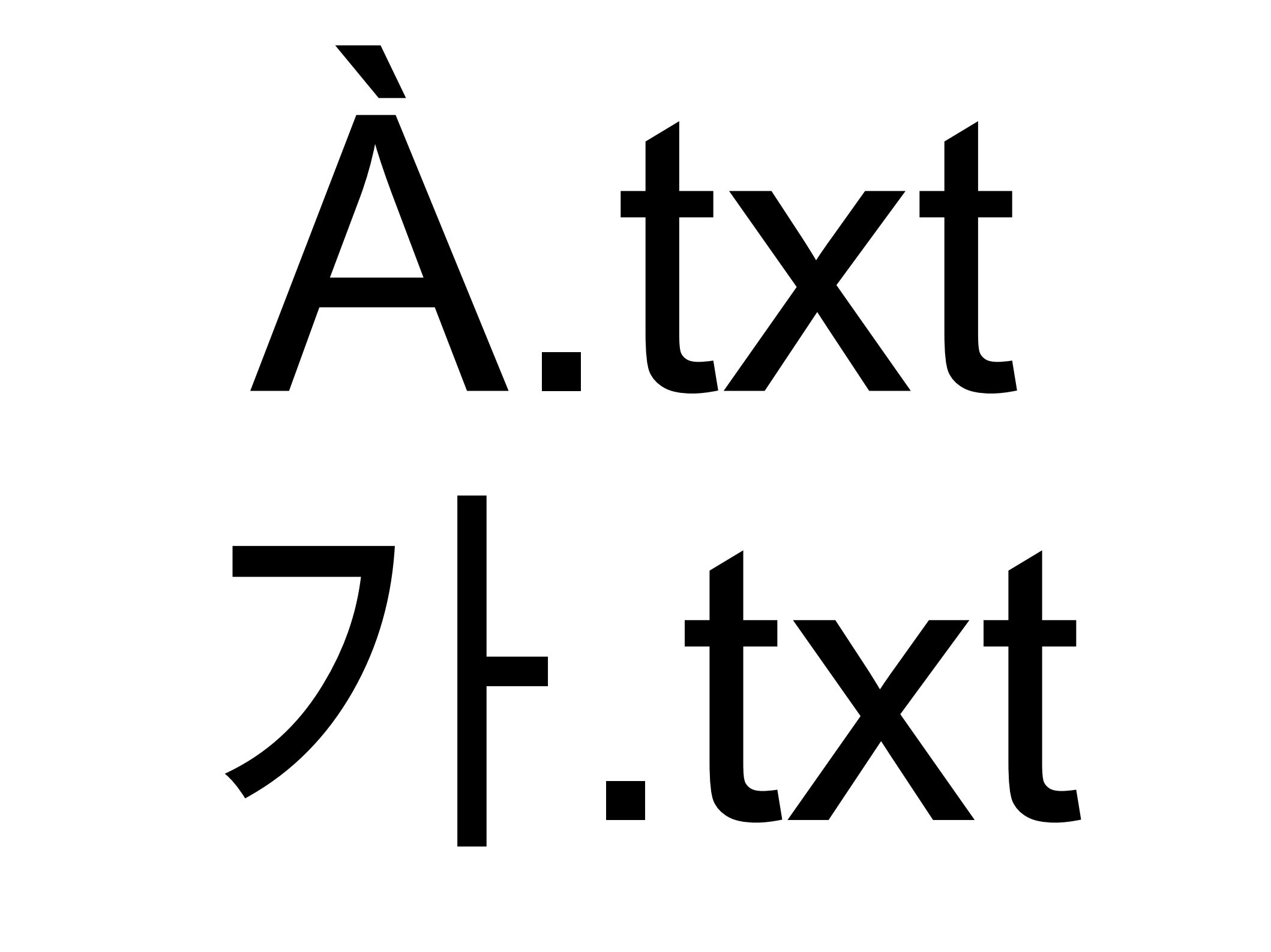
NT系では
UNICODEが
ネイティブ

ファイル名に
ANSIコードページに
ない文字も使える

ㅏ.txt



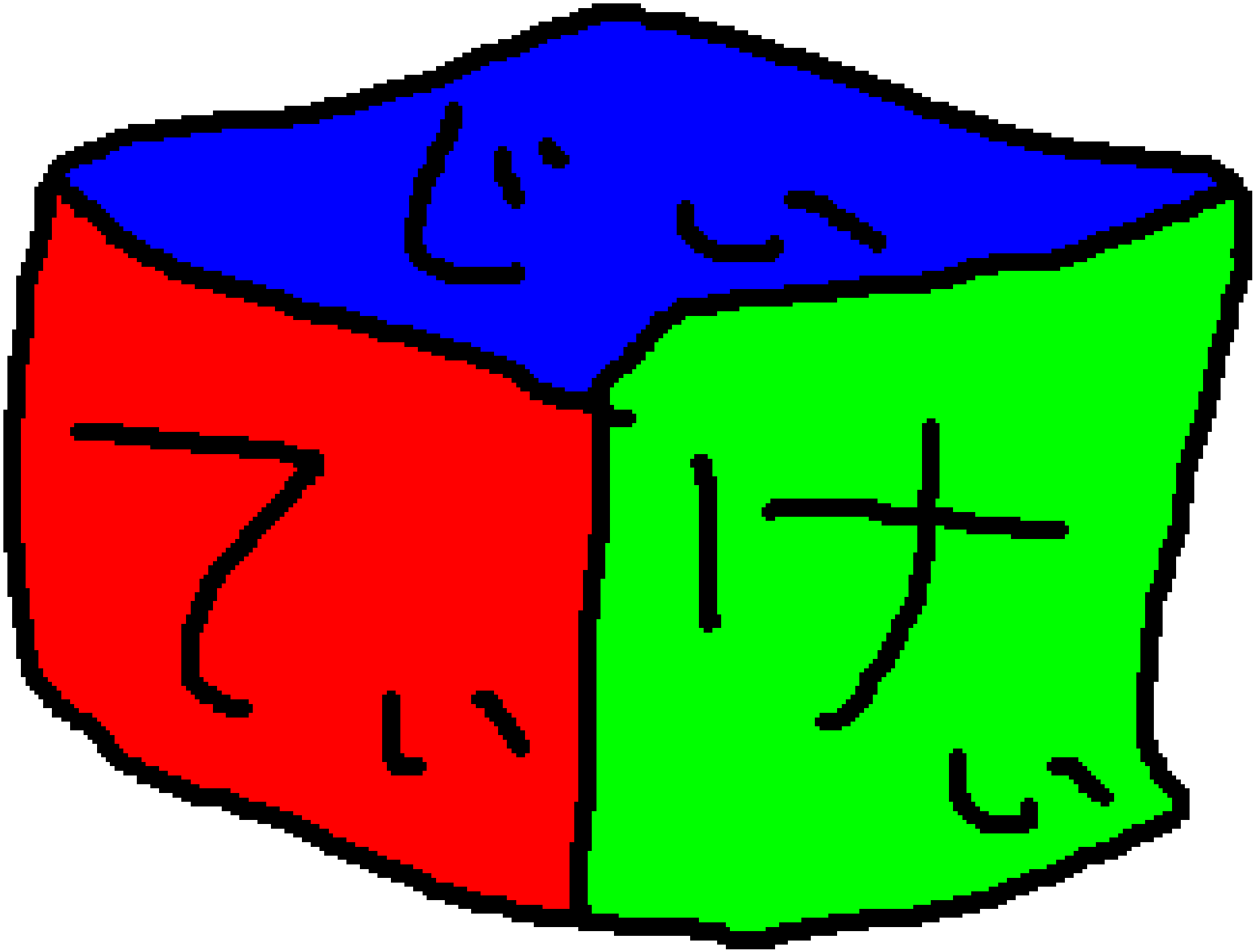
가.txt


À.txt
가.txt

これらのファイルに
アクセスできるのは
~WのAPIのみ

～AのAPIでは
ファイルを
正しく扱えない

同様にopenやstatなどに対応する
wopenやwstatがある



環境変数G_FILENAME_ENCODING
環境変数G_BROKEN_FILENAMES

Microsoft Windowsでは両方とも未対応

GLib-2.4以前では

file name encoding はANSIコードページに固定

ANSIコードページは
openやstat、readdir
などのAPIで使える

file name encodingは
openやstat、readdir
などのAPIで使える

この実装は
UNIX系OSでの
実装と矛盾しない

Microsoft Windowsではglib-2.4まで

2.6以降は煩雑

ANSIコードページだとファイルを正しく扱うことができない

UNICODE
であるべき

しかし
Microsoft Windowsの
16ビット単位の
UNICODE形式は
GTK+/GLibにそぐわない

だから
file name encoding
はUTF-8になった

GtkFileChooserや
g_dir_read_name

可能ならば UNICODEの APIを呼ぶ

可能ならば
=NT系OSならば

UNICODEから
UTF-8に変換

不可能ならば ANSIコードページのAPIを呼ぶ

不可能ならば
=95/98/ME

ANSIコードページからUTF-8に変換

UTF-8は
openやstat、readdir
などのAPIで使ない

file name encodingは
openやstat、readdir
などのAPIで使ない

GLib-2.6では
新しいAPIを追加

g_openや
g_statなど

機能はopenや
statなどと同じ

UNIX系OSでは
単に名前が
変わっただけ

UNIX系OS
では引数のファイル名は
file name encoding

Microsoft Windows
では引数のファイル名は
UTF-8

Microsoft Windows では引数のファイル名は file name encoding

即ち、g_openや
g_statなどは常に
file name encoding

ソースコードに
一貫性あり

UNIX系OSと Microsoft Windowsで 共通のソースコード

2.4以下との
互換性がなくなる

必要に応じて
2.4以下のために
g_openやg_stat
などをマクロ定義

```
#if ! GLIB_CHECK_VERSION(2,6,0)
#define g_open open
#endif
```

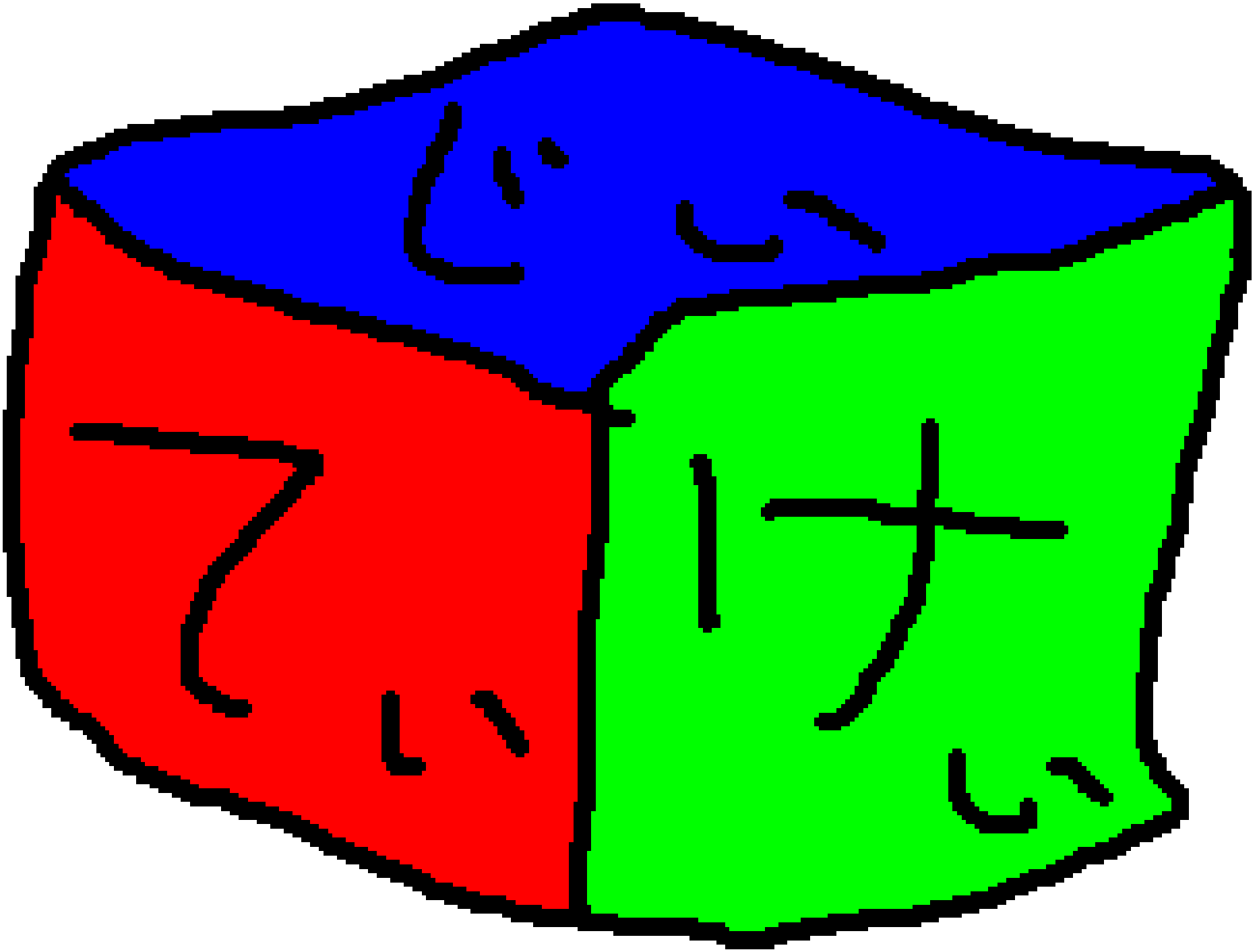
g_～を使わない
＝正しく動作しない

Microsoft Windowsでは
g_～はw～を使う

例えばg_openは
UTF-8をUNICODEに
変換してwopenを
呼び出す

ファイル名の
劣化なし

À.txtや
ヲト.txtも
OK



GLib-2.6にあるg_～なAPI

g_open、g_rename、
g_mkdir、g_stat、g_lstat、
g_unlink、g_remove、
g_rmdir、g_fopen、
g_freopen

GLib-2.8で追加された g_～なAPI

g_chmod、g_access、g_creat、g_chdir

ここにないAPIを使う時には?

2.6で chmod、access、creat、chdirを使うには?

file name encoding をANSIコードページに変換

UTF-8
をANSIコードページ
に変換

Microsoft Windowsでは
ロケールの文字符号化方式
がANSIコードページ

g_locale_from_utf8

通常
file name encoding≠UTF-8

ロケールの文字符号化方式
=ANSIコードページ
はmicrosoft Windowsのみ

file name encodingを
g_locale_to_utf8で
変換してファイル名として
使う

Microsoft Windowsの特別な処理

GLib-2.8以降

g_win32_locale_filename_from_utf8

新設

UTF-8

↓

ロケールの文字符号化方式

専用API

file name encoding

↓

ANSIコードページ

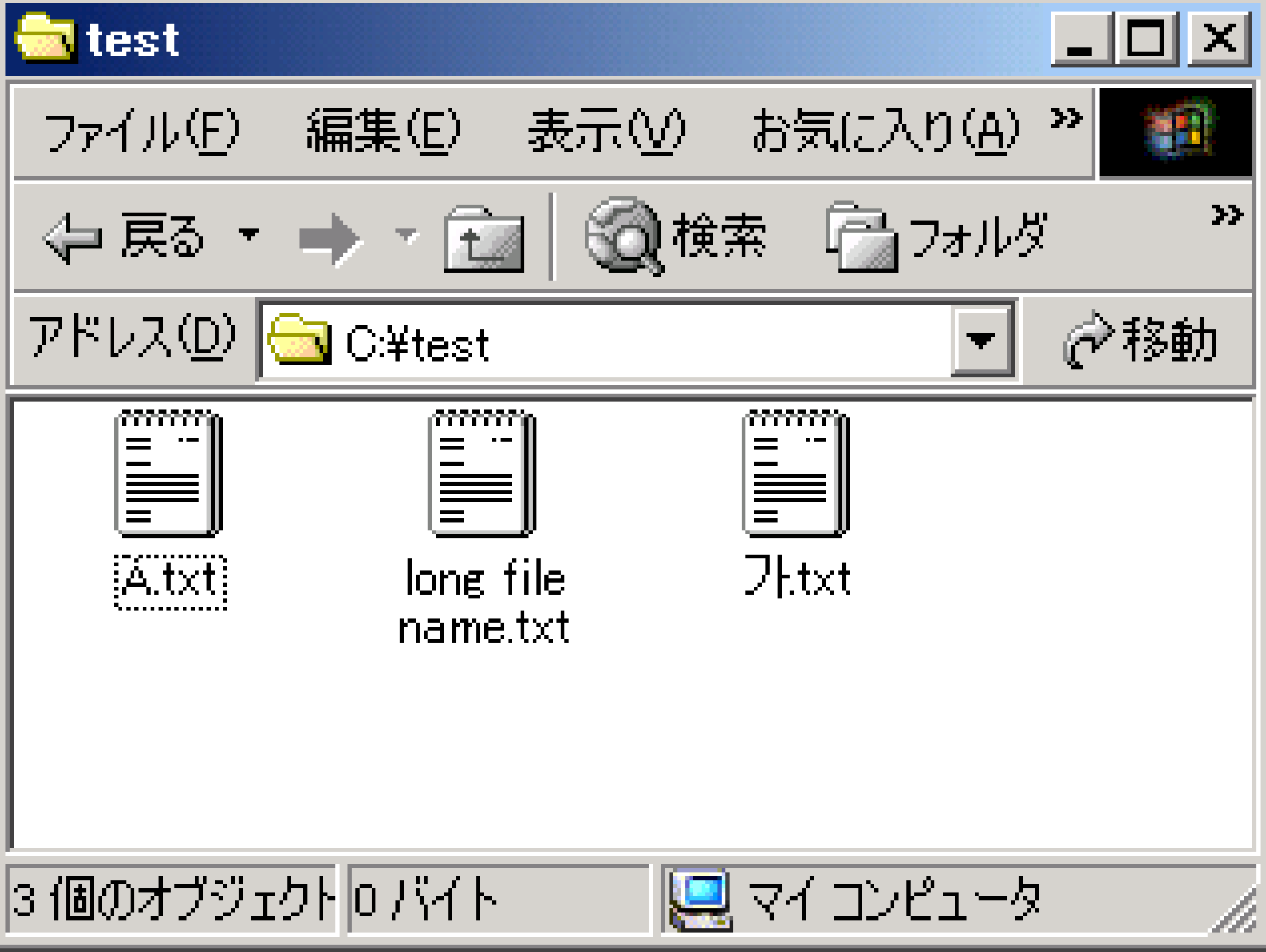
専用API

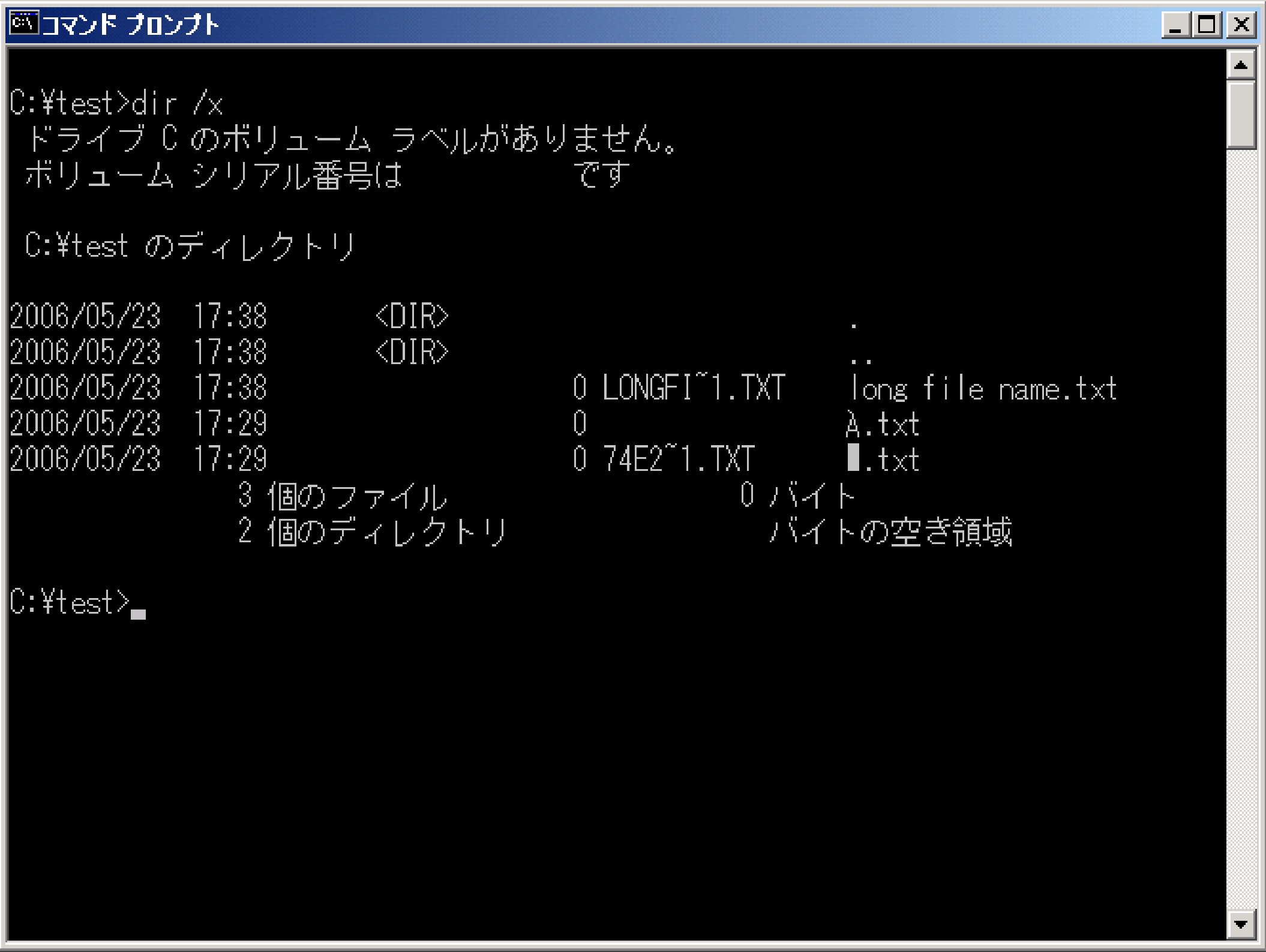
g_locale_from_utf8 変換を試みるだけ

変換は
失敗するかも

例えばCP932で
À.txtやｱﾄ.txtは
変換できない

g_win32_locale_filename_from_utf8は失敗したらGetShortPathNameW


test
ファイル(F)
編集(E)
表示(V)
お気に入り(A)
戻る
検索
フォルダ
アドレス(D)
C:¥test
移動
A.txt
long file name.txt
가.txt
3 個のオブジェクト
0 バイト
マイ コンピュータ


コマンド プロンプト
C:¥test>dir /x
ドライブ C のボリューム ラベルがありません。
ボリューム シリアル番号は です
C:¥test のディレクトリ
2006/05/23 17:38 <DIR> .
2006/05/23 17:38 <DIR> ..
2006/05/23 17:38 0 LONGFI~1.TXT long file name.txt
2006/05/23 17:29 0 Â.txt
2006/05/23 17:29 0 74E2~1.TXT ■.txt
3 個のファイル 0 バイト
2 個のディレクトリ バイトの空き領域
C:¥test>

가.txt

↓

74E2~1.TXT

74E2~1.TXT
ならばANSIコード
ページに変換可

À.txt
短いファイル名に
変換できていない

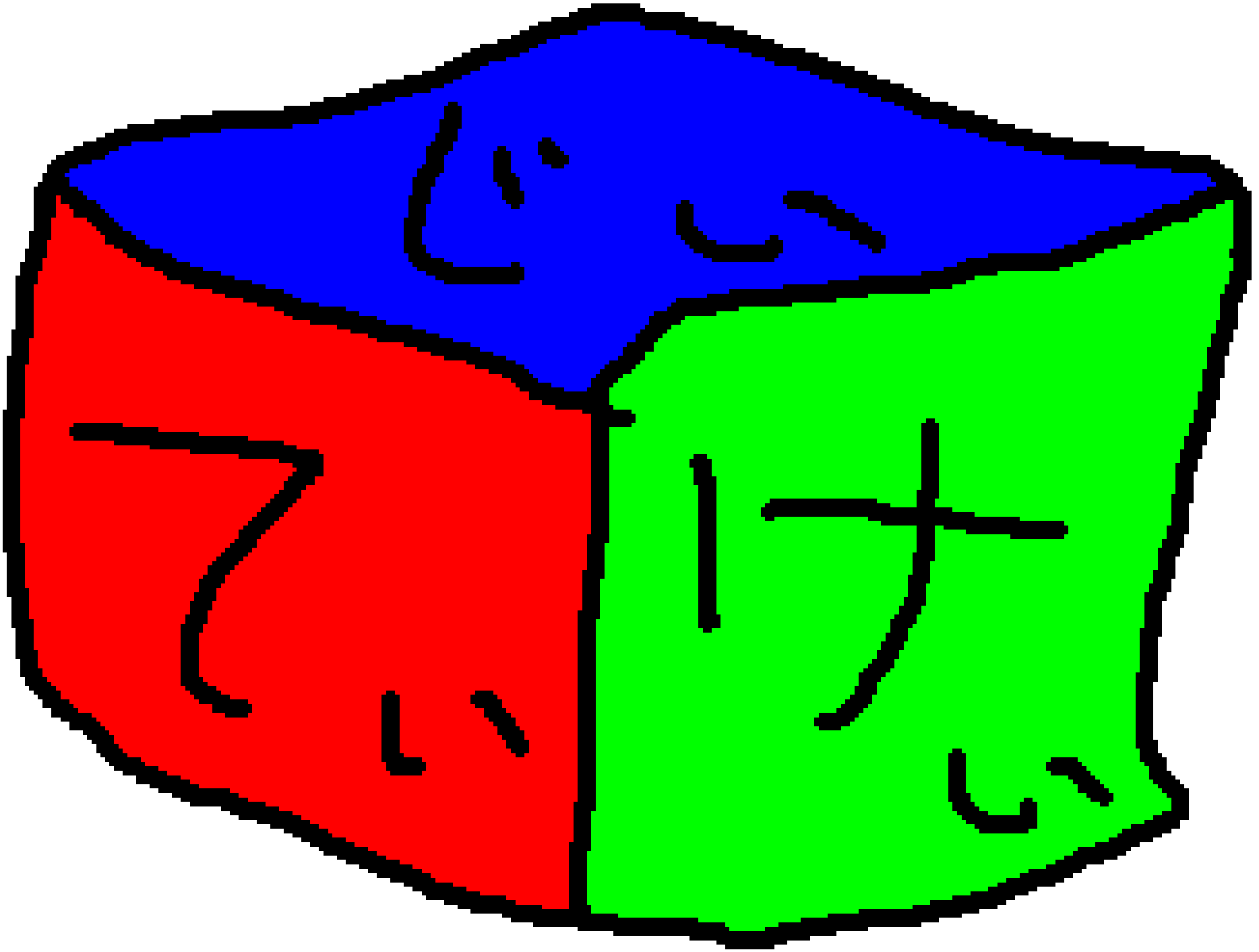
変換に成功する
可能性は高まる

でも失敗する
可能性もある

Microsoft Windowsならば
g_locale_from_utf8や
g_win32_locale_filename_from_utf8
で変換するよりも
g_utf8_to_utf16

UNICODEに変換してUNICODEのAPIを呼ぶべき

CreateFileWとか
wopenなど



パスの区切り

UNIX系OS
/
スラッシュ
(2Fh)

Microsoft Windows
\
バックスラッシュ
(5Ch)

マクロ
G_DIR_SEPARATOR
G_DIR_SEPARATOR_S

G_DIR_SEPARATOR は文字

G_DIR_SEPARATOR_Sは文字列

```
#define G_DIR_SEPARATOR '/'
#define G_DIR_SEPARATOR_S "/"
```

GLib-2.6以降

マクロ

G_IS_DIR_SEPARATOR (c)

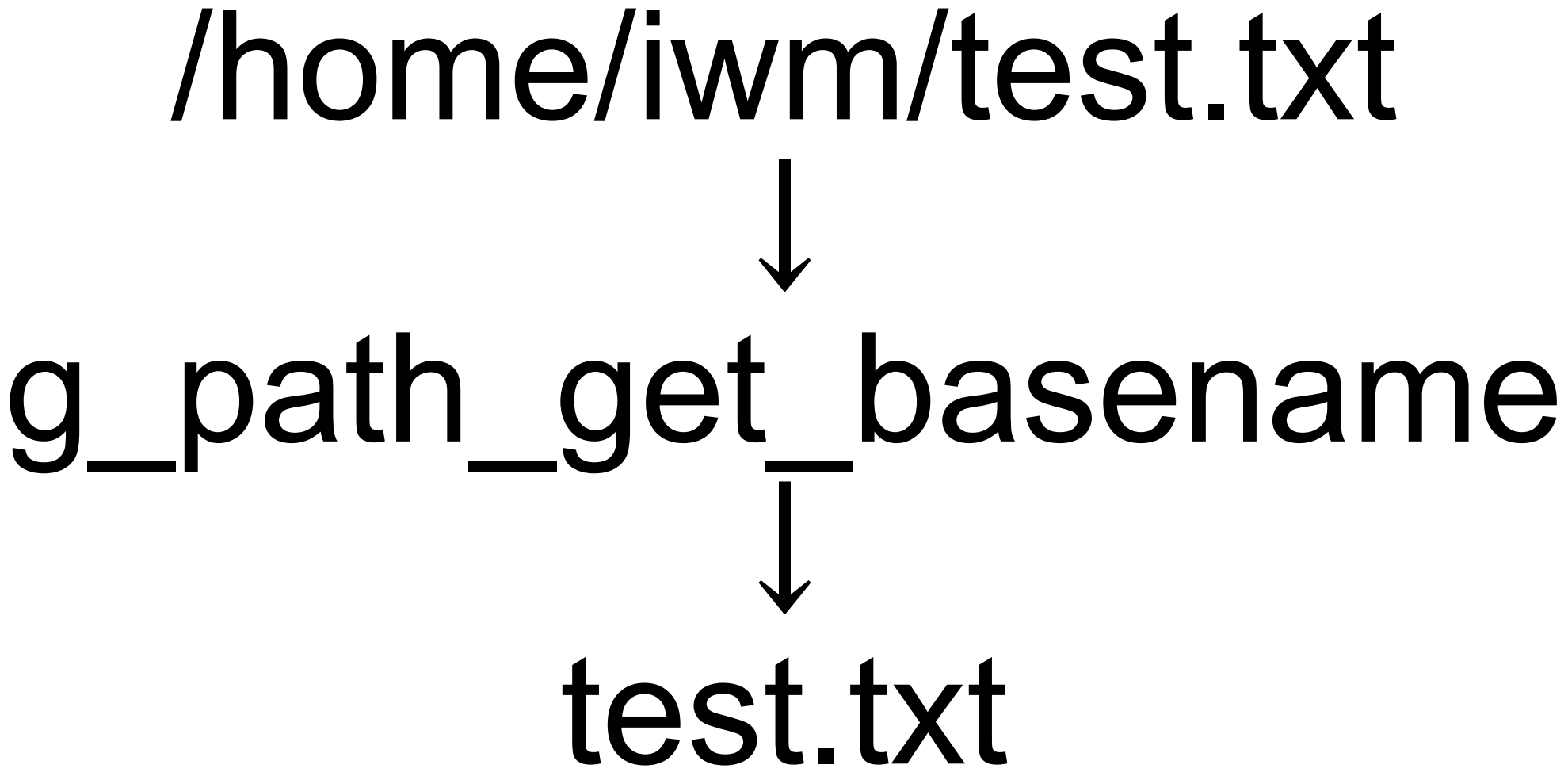
文字がパスの
区切りならば真

Microsoft Windowsでは特別な実装

「/」と「\」
の両方で真

GLibのパスを扱うAPI

g_path_get_basename
g_path_get_dirname


/home/iwm/test.txt
g_path_get_basename
test.txt

/home/iwm/test.txt

↓

g_path_get_dirname

↓

/home/iwm

Microsoft Windows
GLib-2.2.3以降
g_path_get_basename
g_path_get_dirname
「/」と「\」の両方を区切り

マクロ
G_IS_DIR_SEPARATOR (c)
の導入よりも前

複数のパスを区切る

UNIX系OS

：

：

コロン

(3Ah)

Microsoft Windows

;

セミコロン

(3Bh)

マクロ
G_SEARCHPATH_SEPARATOR
G_SEARCHPATH_SEPARATOR_S

G_SEARCHPATH_SEPARATOR
は文字

G_SEARCHPATH_SEPARATOR_S
は文字列

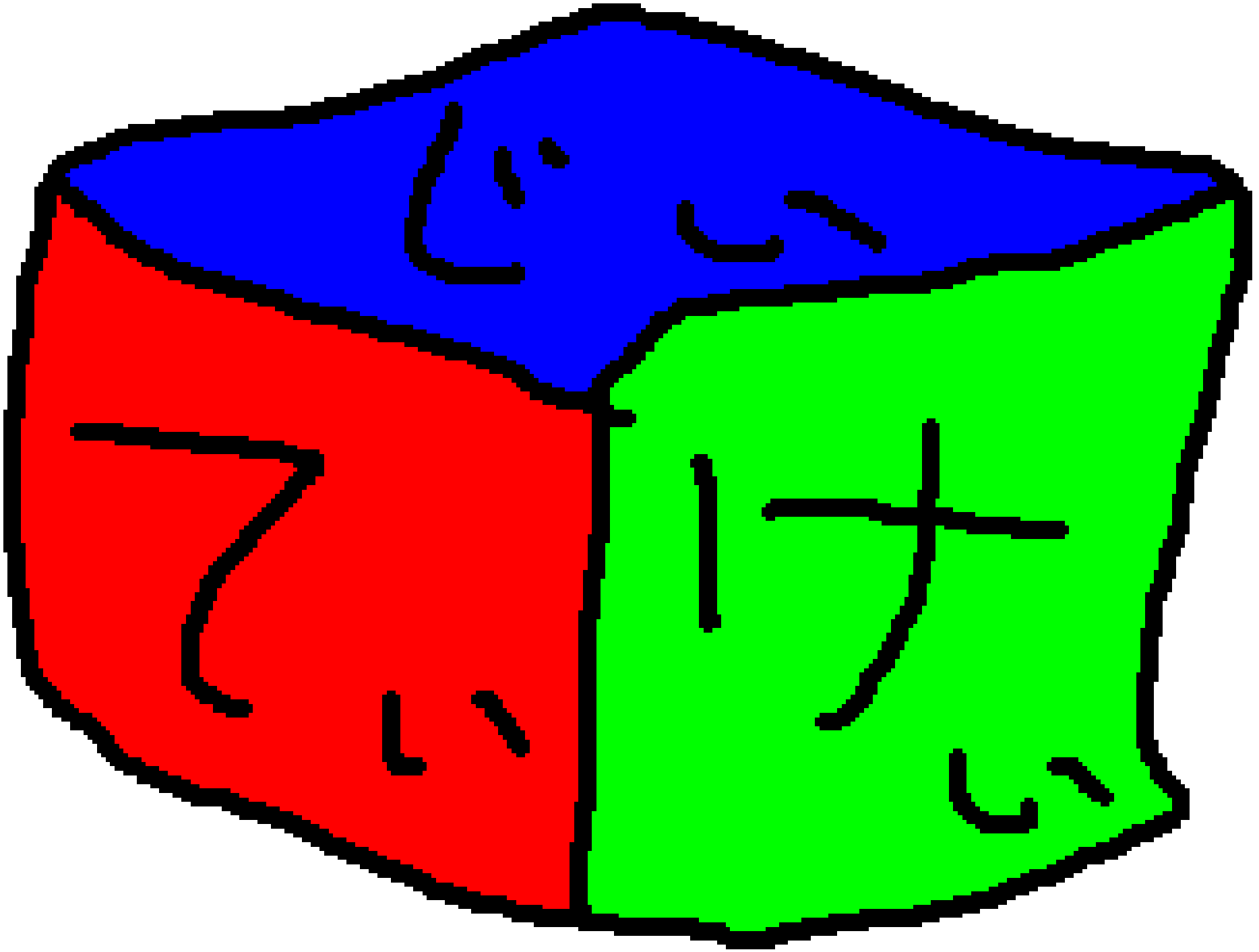
```
#define G_SEARCHPATH_SEPARATOR ':'
#define G_SEARCHPATH_SEPARATOR_S ":"
```

マクロを使わない

↓

ハードコード

移植性に問題が…



5Ch問題

5Ch=パスの区切り

5Ch=2バイト目

単純に5Chを検索

strchr、strcspn、strpbrk、strrchr、strspn、strstr、strtok

2バイトの文字が分断される可能性

CP932

95h 5Ch

C:\doc\表1.txt

C	:	\	d	o	c	\	表		l	.	t	x	t
43h	3Ah	5Ch	64h	6Fh	63h	5Ch	95h	5Ch	31h	2Eh	74h	78h	74h

「表」という文字の
途中で分断

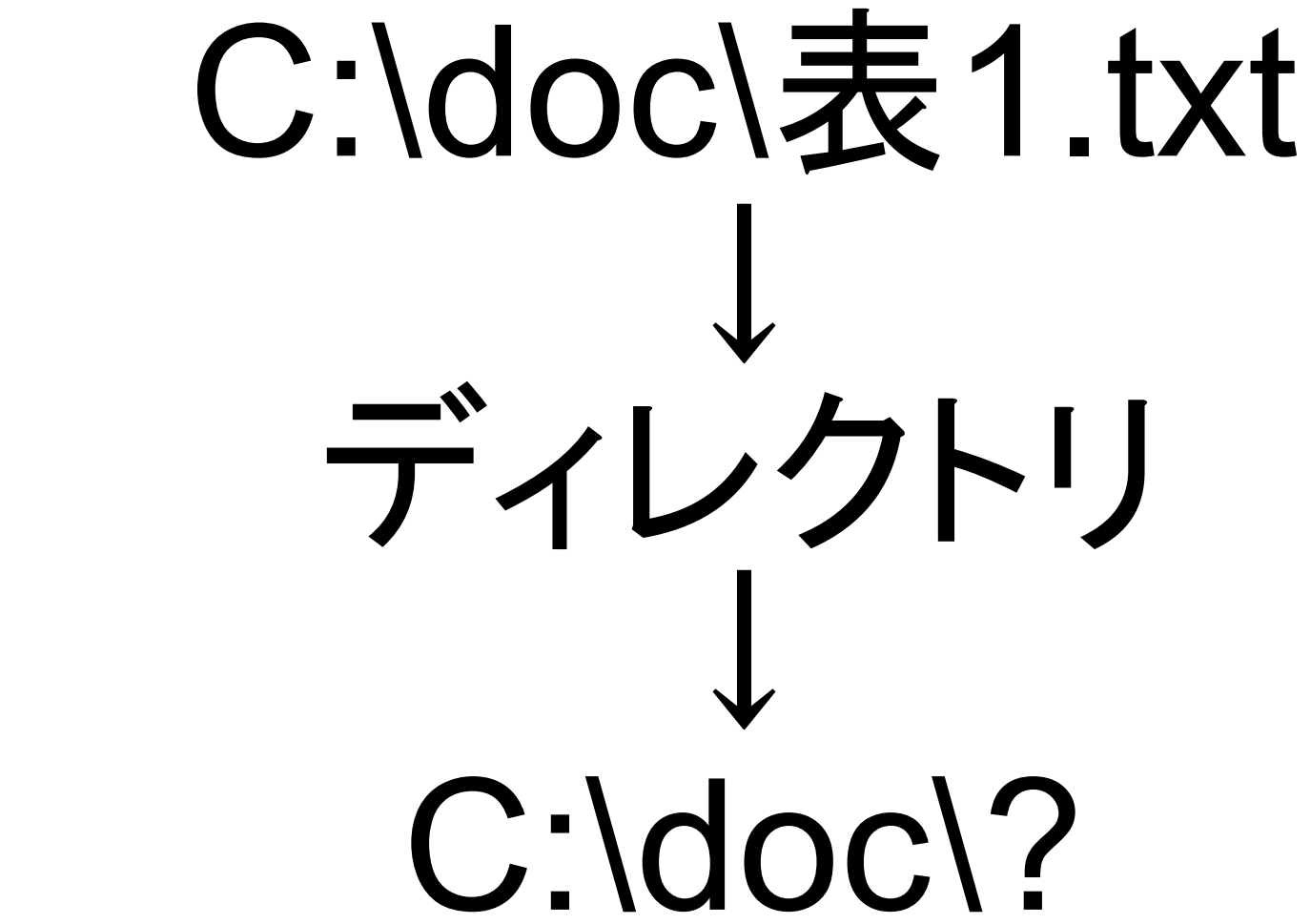
C:\doc\表1.txt

↓

ファイル名

↓

1.txt

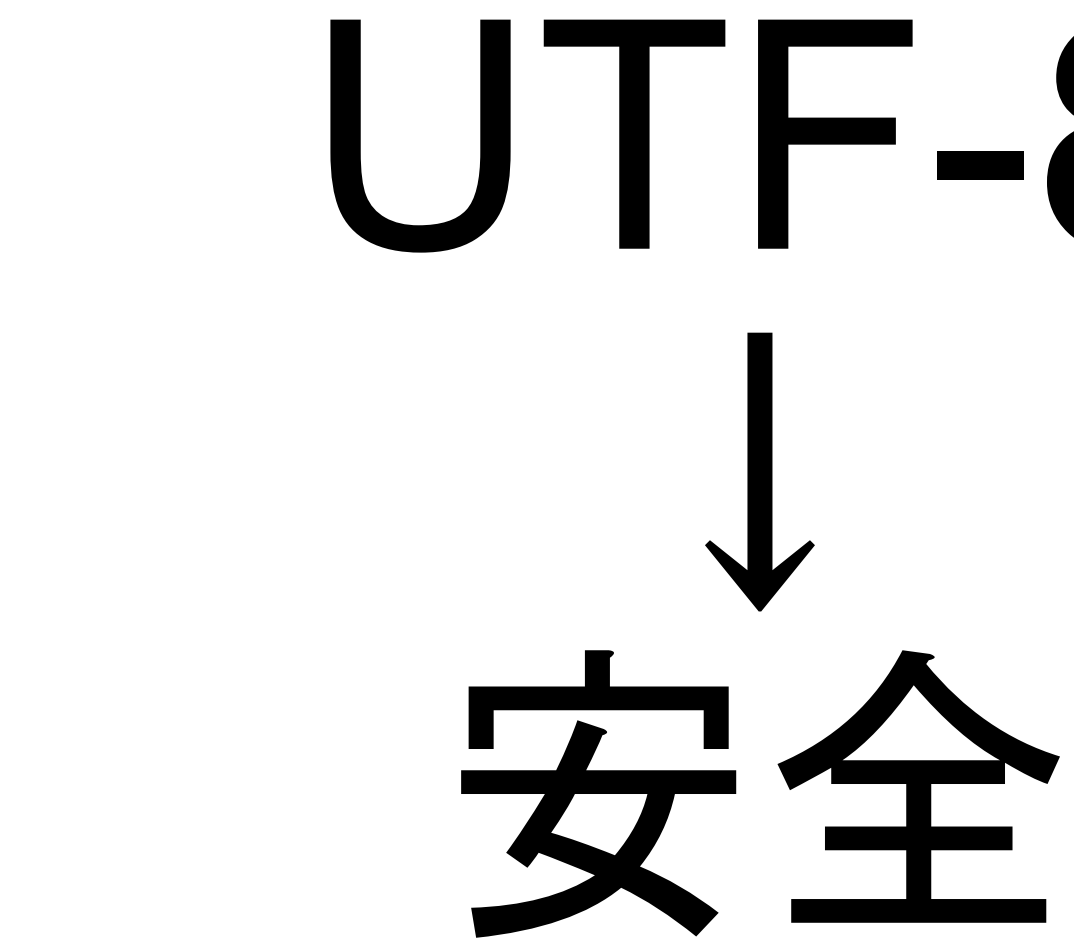

C:\doc\表1.txt
ディレクトリ
C:\doc\?

先頭から見て
2バイト文字の
先頭かどうか
判別しながら探す

GLibの実装は?

GLib-2.6以降
file name encoding
↓
UTF-8

UTF-8は
2バイト目以降に
5Chはない


UTF-8
安全

GLib-2.6以降

↓

安全

GLib-2.6以降
file name encoding
↓
ANSIコードページ

ANSIコードページは
2バイト目以降に5Ch
があるかも

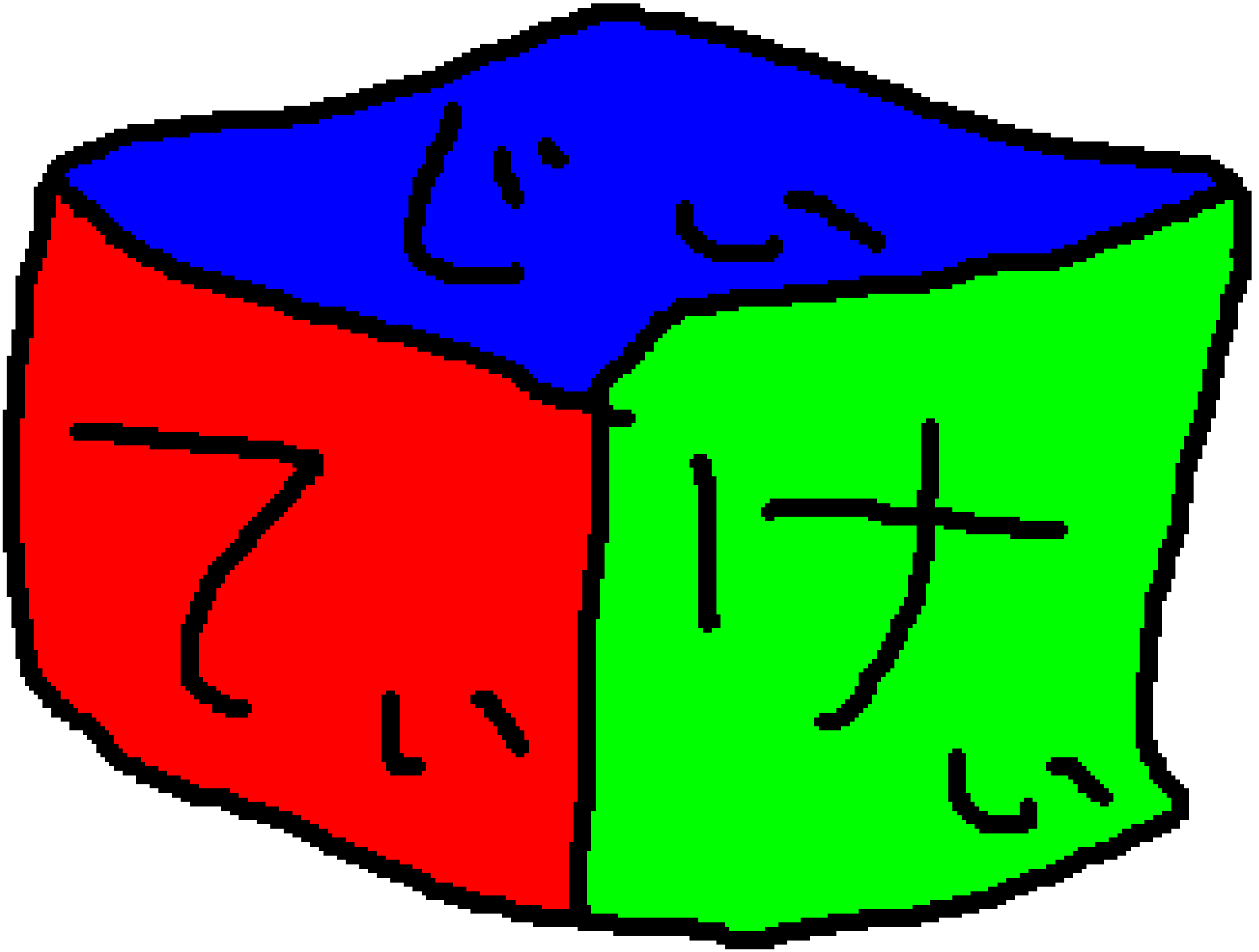
GLib-2.0.0~GLib-2.4.8
すべての実装で
単純に5Chで分割

GLib-2.4以前

↓

危険

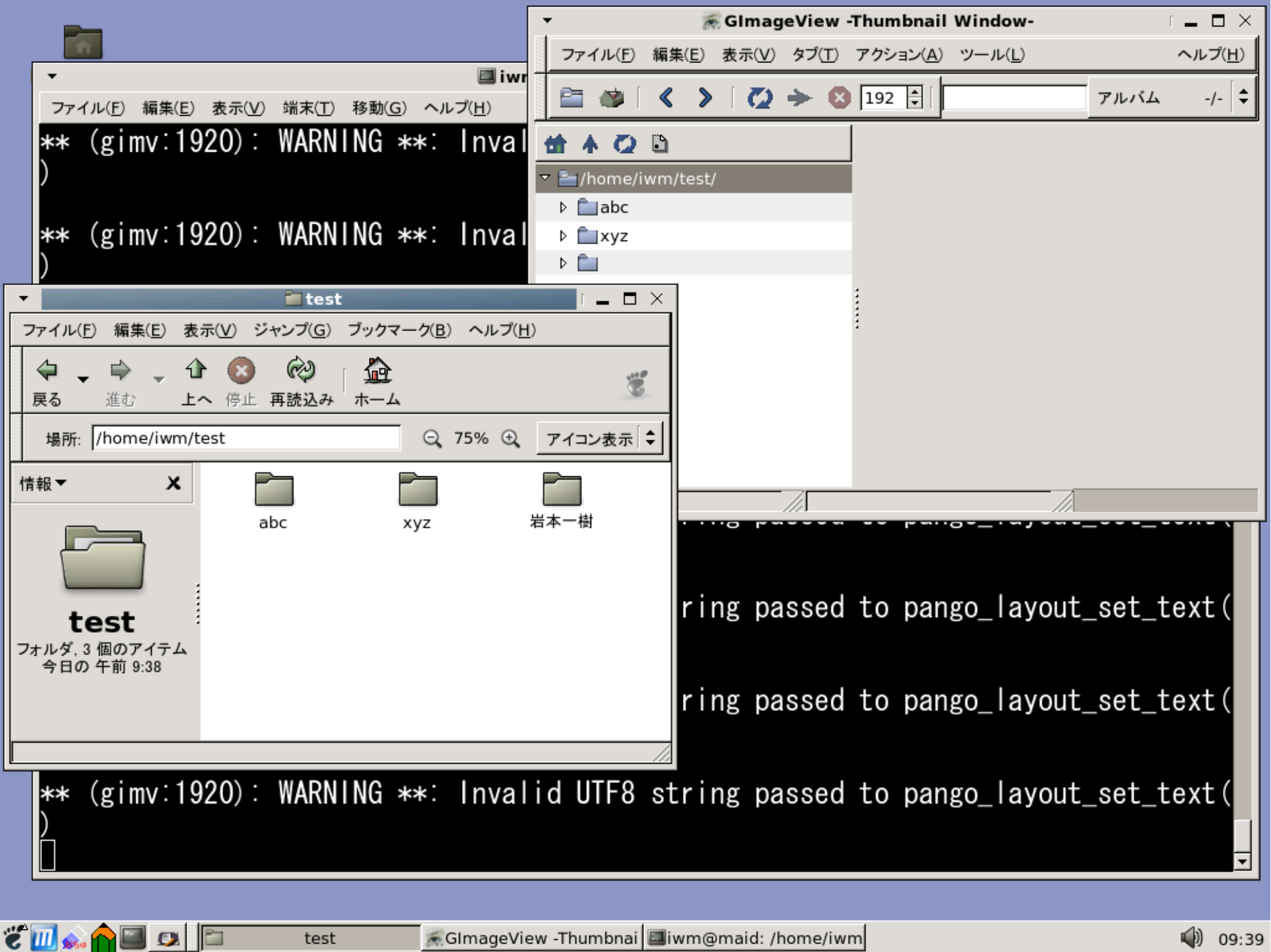
Microsoft Windowsでは
GLib-2.4以前の
g_path_get_basename
g_path_get_dirname
は使えない

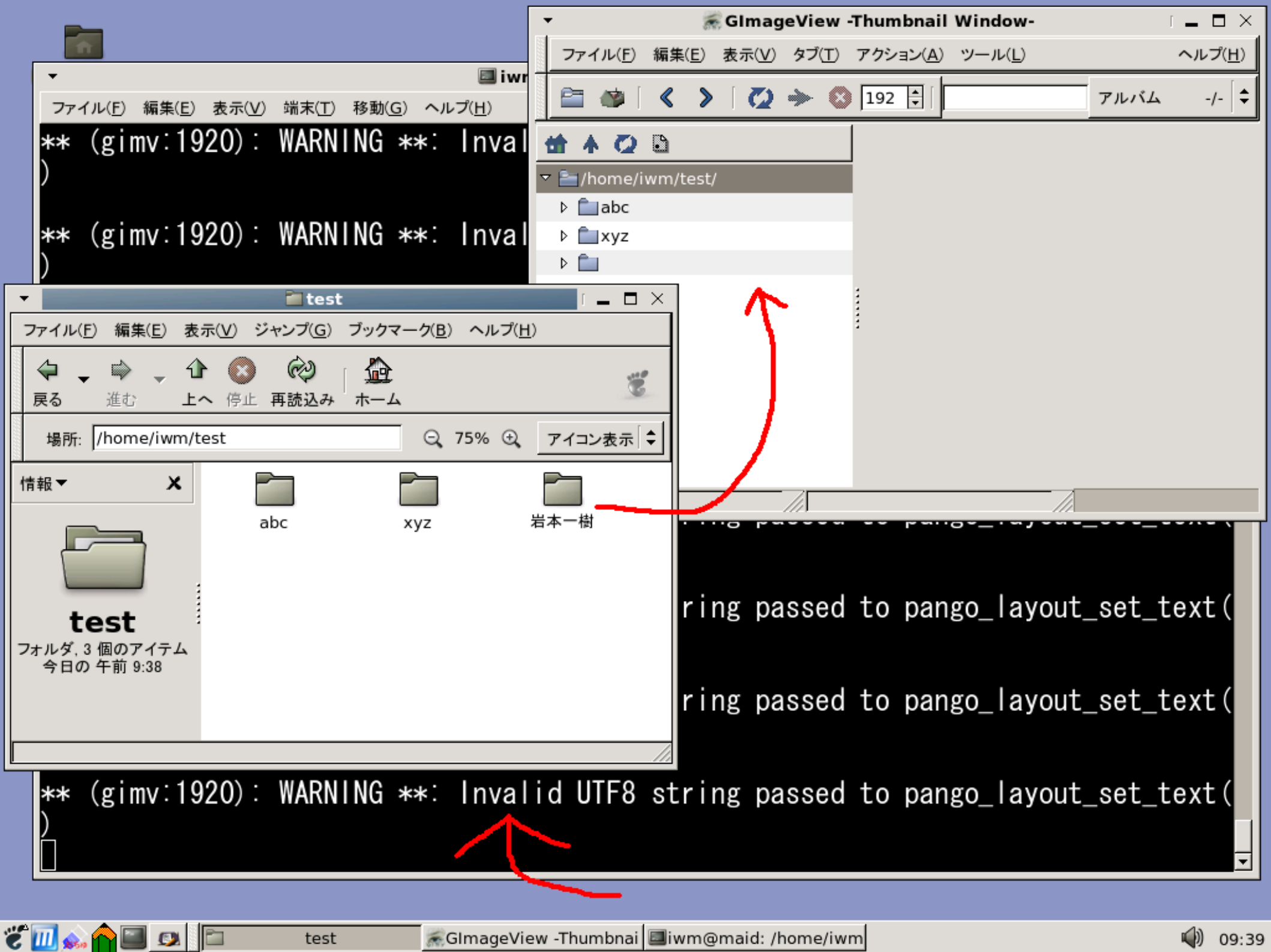


実際のアプリケーションでは?

GImageView

画像閲覧ソフト

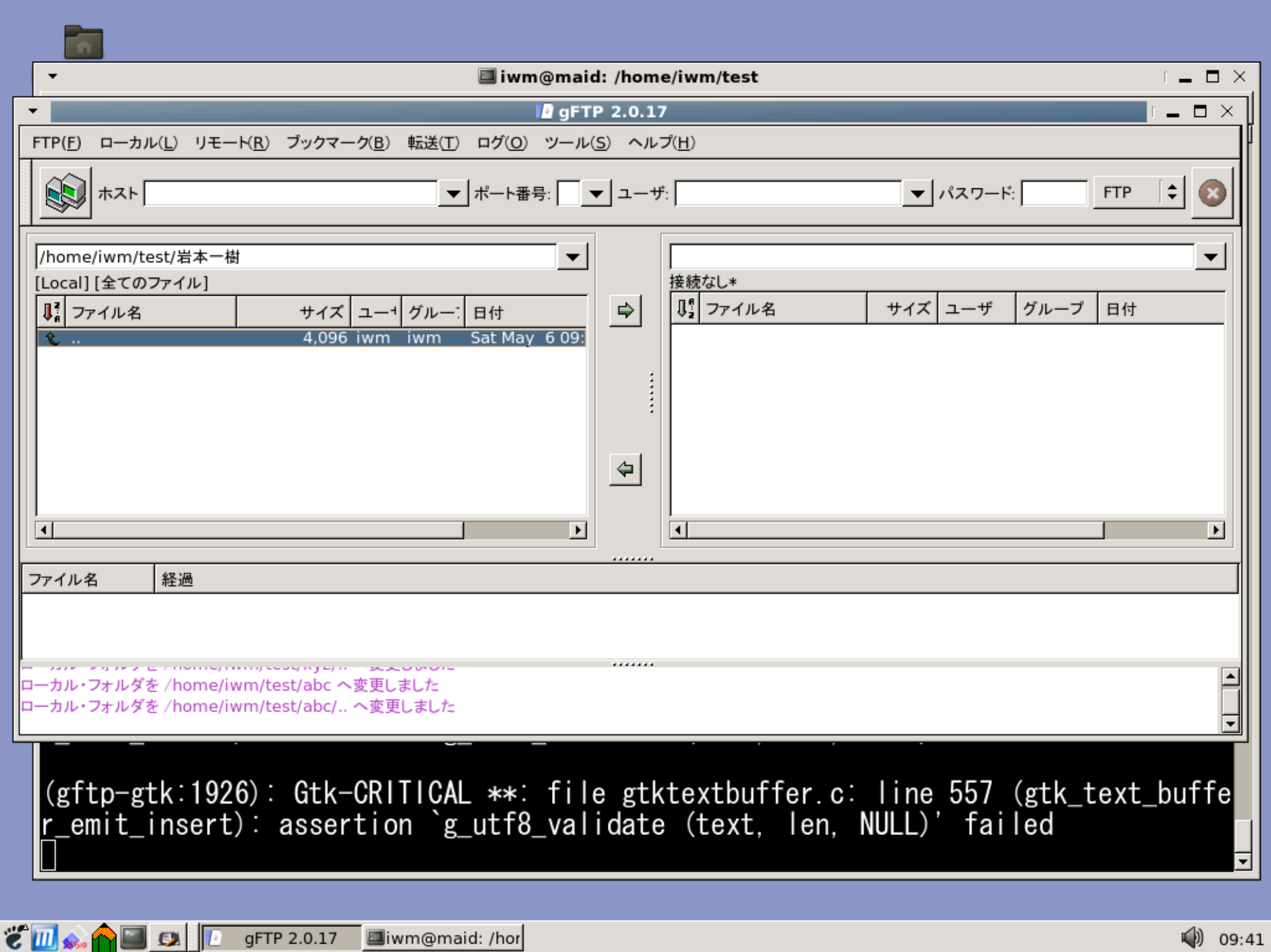

GImageView -Thumbnail Window-
ファイル(F) 編集(E) 表示(V) タブ(T) アクション(A) ツール(L) ヘルプ(H)
192
アルバム
/home/iwm/test/
abc
xyz
test
ファイル(F) 編集(E) 表示(V) ジャンプ(G) ブックマーク(B) ヘルプ(H)
戻る 進む 上へ 停止 再読込み ホーム
場所: /home/iwm/test
75%
アイコン表示
情報
abc
xyz
岩本一樹
test
フォルダ, 3 個のアイテム
今日の 午前 9:38
ファイル(F) 編集(E) 表示(V) 端末(T) 移動(G) ヘルプ(H)
** (gimv:1920): WARNING **: Inval
)
** (gimv:1920): WARNING **: Inval
)
ring passed to pango_layout_set_text(
ring passed to pango_layout_set_text(
** (gimv:1920): WARNING **: Invalid UTF8 string passed to pango_layout_set_text(
)
test
GImageView -Thumbnai
iwm@maid: /home/iwm
09:39

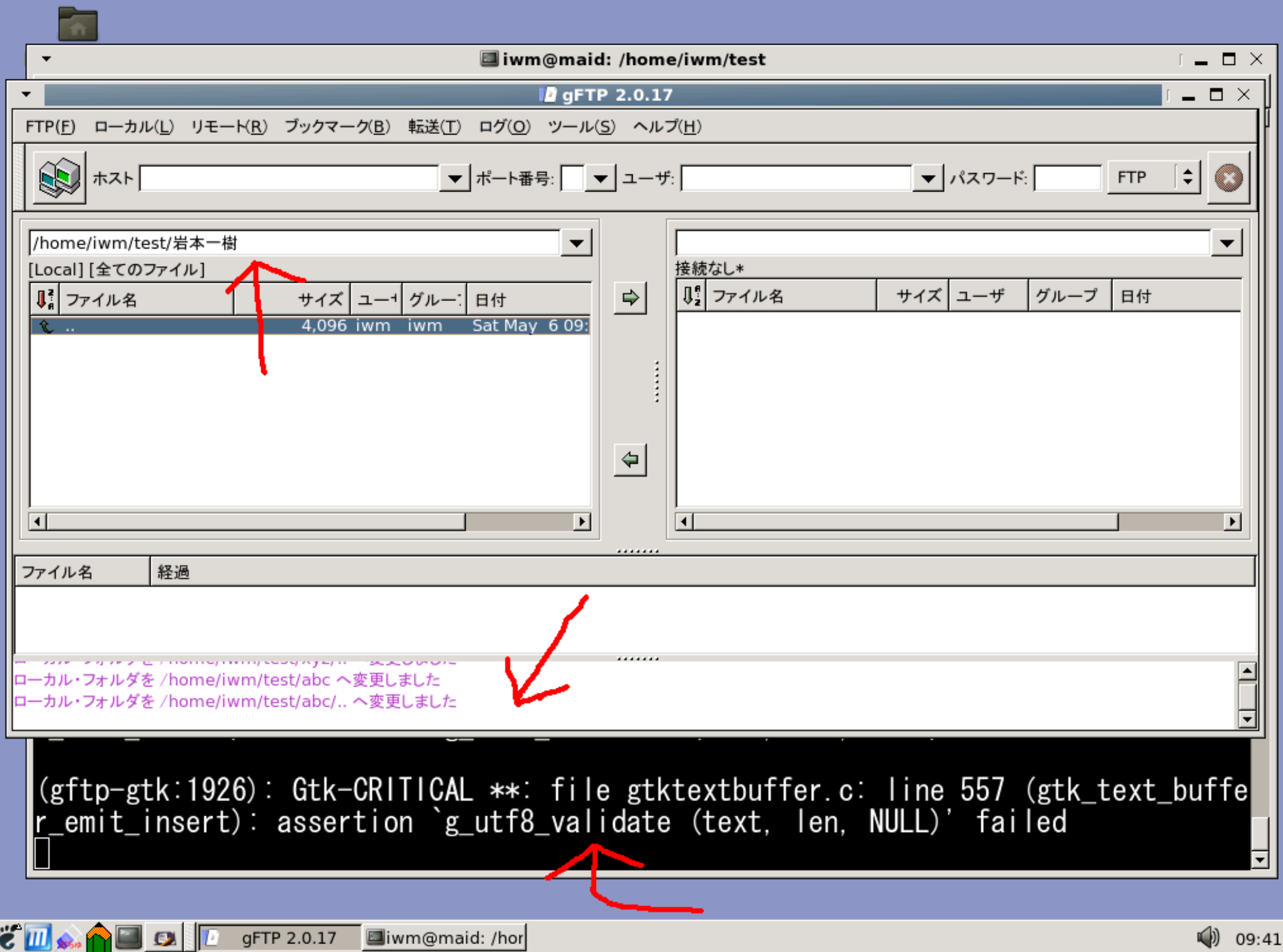

GImageView -Thumbnail Window-
ファイル(F) 編集(E) 表示(V) タブ(T) アクション(A) ツール(L) ヘルプ(H)
192
アルバム -/-
/home/iwm/test/
abc
xyz
test
ファイル(F) 編集(E) 表示(V) ジャンプ(G) ブックマーク(B) ヘルプ(H)
戻る 進む 上へ 停止 再読込み ホーム
場所: /home/iwm/test
75%
アイコン表示
情報
abc
xyz
岩本一樹
test
フォルダ, 3 個のアイテム
今日の 午前 9:38
** (gimv:1920): WARNING **: Invalid UTF8 string passed to pango_layout_set_text(
)
test
GImageView -Thumbnai
iwm@maid: /home/iwm
09:39

まったく変換を行っていない

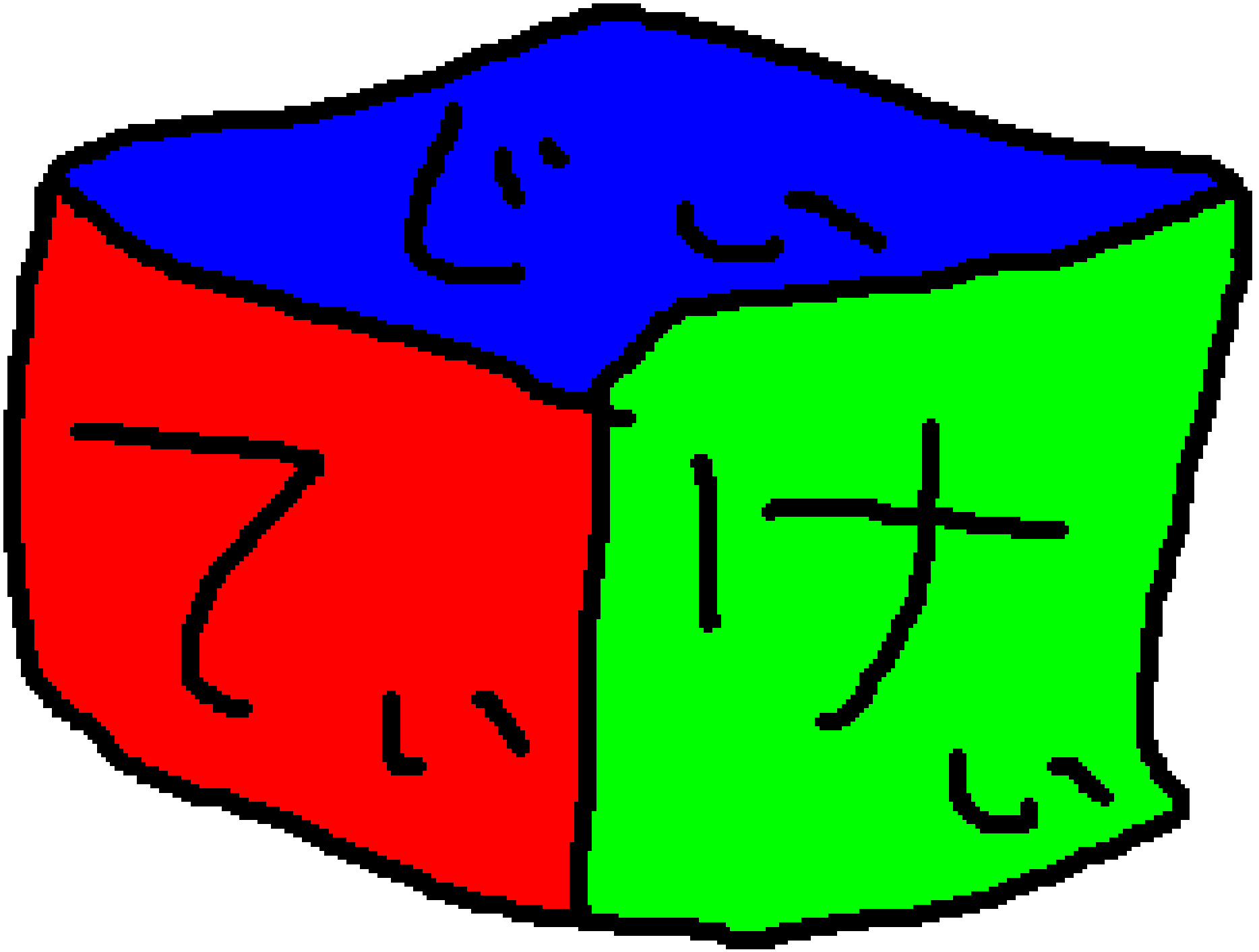

gFTP

FTPクライアント


iwm@maid: /home/iwm/test
gFTP 2.0.17
FTP(F) ローカル(L) リモート(R) ブックマーク(B) 転送(T) ログ(O) ツール(S) ヘルプ(H)
ホスト
ポート番号:
ユーザ:
パスワード:
FTP
/home/iwm/test/岩本一樹
[Local] [全てのファイル]
ファイル名 サイズ ユー グルー 日付
.. 4,096 iwm iwm Sat May 6 09:
接続なし*
ファイル名 サイズ ユーザ グループ 日付
ファイル名 経過
ローカル・フォルダを /home/iwm/test/abc へ変更しました
ローカル・フォルダを /home/iwm/test/abc/.. へ変更しました
(gftp-gtk:1926): Gtk-CRITICAL **: file gtktextbuffer.c: line 557 (gtk_text_buffe
r_emit_insert): assertion `g_utf8_validate (text, len, NULL)' failed
gFTP 2.0.17
iwm@maid: /hor
09:41


iwm@maid: /home/iwm/test
gFTP 2.0.17
FTP(F) ローカル(L) リモート(R) ブックマーク(B) 転送(T) ログ(O) ツール(S) ヘルプ(H)
ホスト
ポート番号:
ユーザ:
パスワード:
FTP
/home/iwm/test/岩本一樹
[Local] [全てのファイル]
ファイル名 サイズ ユーザ グループ 日付
.. 4,096 iwm iwm Sat May 6 09:
接続なし*
ファイル名 サイズ ユーザ グループ 日付
ファイル名 経過
ローカル・フォルダを /home/iwm/test/abc へ変更しました
ローカル・フォルダを /home/iwm/test/abc/.. へ変更しました
(gftp-gtk:1926): Gtk-CRITICAL **: file gtktextbuffer.c: line 557 (gtk_text_buffer_emit_insert): assertion `g_utf8_validate (text, len, NULL)' failed
gFTP 2.0.17
iwm@maid: /hor
09:41

変換を忘れている?



結局
file name encoding とは...

file name encoding はネイティブなAPIのファイル名としては使えない

file name encoding はGTK+のウィジェットで表示できない

file name encoding は文字列としての体裁を整えてはいるが単なるデータでしかない

中途半端に
文字列だから
混乱を生じる

誤った使い方をしてもコンパイル時に警告は出ない

ASCIIな環境だと
問題が発覚しにくい

プログラマ側が
きっちり管理する
必要がある

プログラマの負担が大きい

変換し忘れることも…

隠せ

例えば、構造体 GFileName

メンバはgchar型の配列かもしれない

```
struct  _GFileName {
  gchar *file;
};
typedef struct _GFileName GFileName;
```

Microsoft Windows ならばメンバは gunichar2型の配列かもしれない

```
struct  _GFileName {
  gunichar2 *file;
};
typedef struct _GFileName GFileName;
```

メンバは16ビットの数値かもしれない

```
struct  _GFileName {
  guint16 id;
};
typedef struct _GFileName GFileName;
```

gtk_file_chooser_get_filename

↓

GFileName

gtk_file_chooser_set_filename

↑

GFileName

g_open、g_fopen、g_dir_open...

↑

GFileName

GFileName *g_filename_get_basename (GFileName *fn)
GFileName *g_filename_get_dirname (GFileName *fn)
gchar *g_filename_get_display_name (GFileName *fn)
GFileName *g_filename_copy (GFileName *fn)
void g_filename_free (GFileName *fn)
gboolean g_filename_compare (GFileName *fn1, GFileName *fn2)
gchar *g_filename_get_native_name (GFileName *fn)
gboolean g_filename_set_native_name (GFileName *fn, const gchar *file)
gunichar2 *g_filename_get_win32_name (GFileName *fn)
gboolean g_filename_set_win32_name (GFileName *fn, const gunichar2 *file)

GFileName構造体を
ネイティブなAPIに使う
↓
エラー

GFileName構造体を
GTK+のウィジェットで
表示
↓
エラー

コンパイル時に
ミスが発覚する

プログラマの
負担が軽くなる

実は既に近いものがある


開く
ホーム
デスクトップ
ファイルシステム
ホーム
test
名前
最終変更日
abc
2006年05月06日
xyz
2006年05月06日
岩本一樹
2006年05月06日
追加(A)
削除(R)
All Files
キャンセル(C)
開く(O)

GtkFileChooser

↑

gtkfilesystem.[ch]

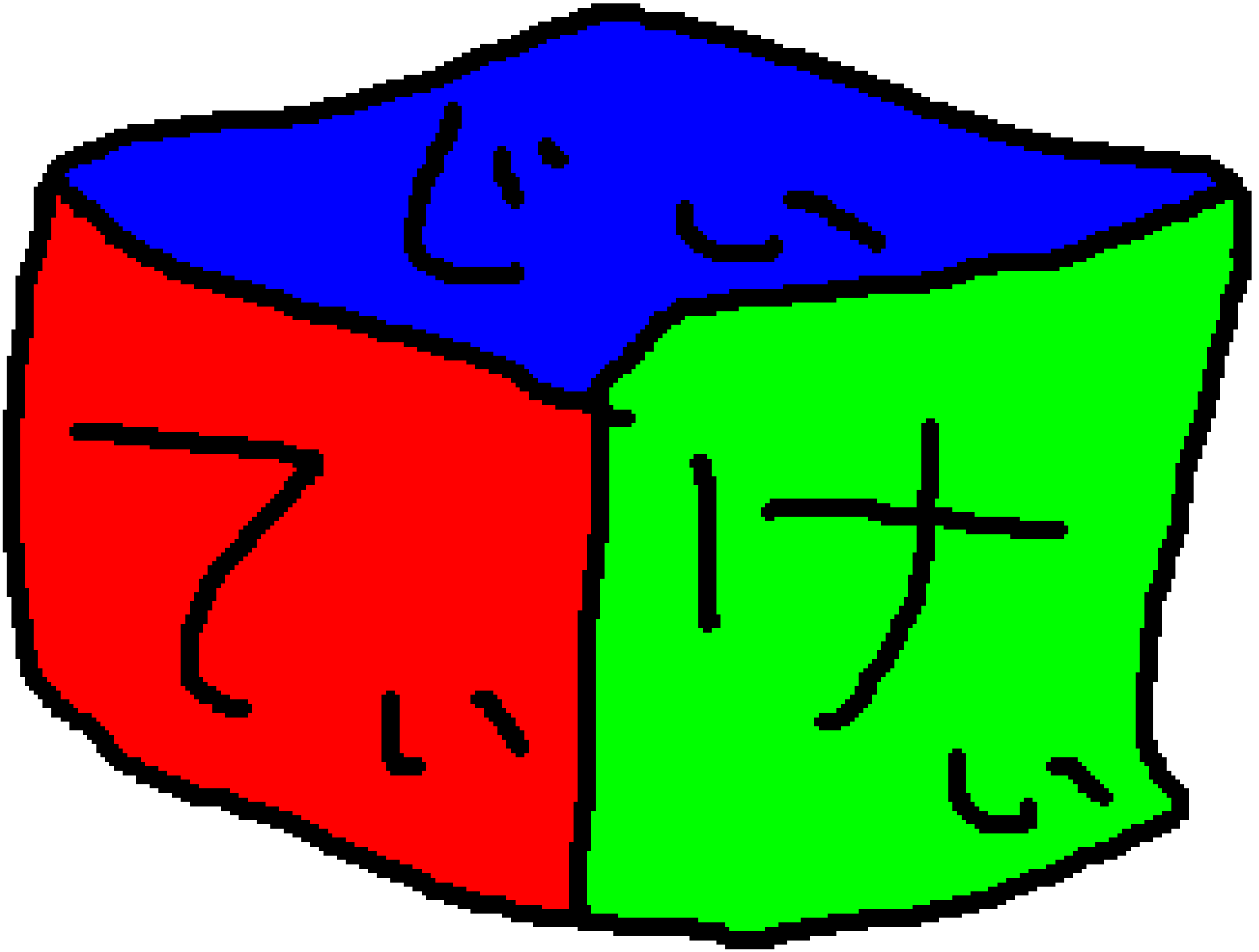
gtkfilesystemunix.[ch]
や
gtkfilesystemwin32.[ch]
がある

GtkFilePathという
構造体を定義している

GtkFilePathを使うAPIもある

しかしこれは
GtkFileChooser
向け

手直ししてGLib側に移したら…



UNIX系OSが一般に普及

↓

ファイル名に非ASCII文字を使う

GTK+はマルチプラットホーム

file name encoding
への対応が不十分
↓
UNIX系OSでしか
動作しない

がっかり

日本語が母国語

↓

問題を理解しやすい