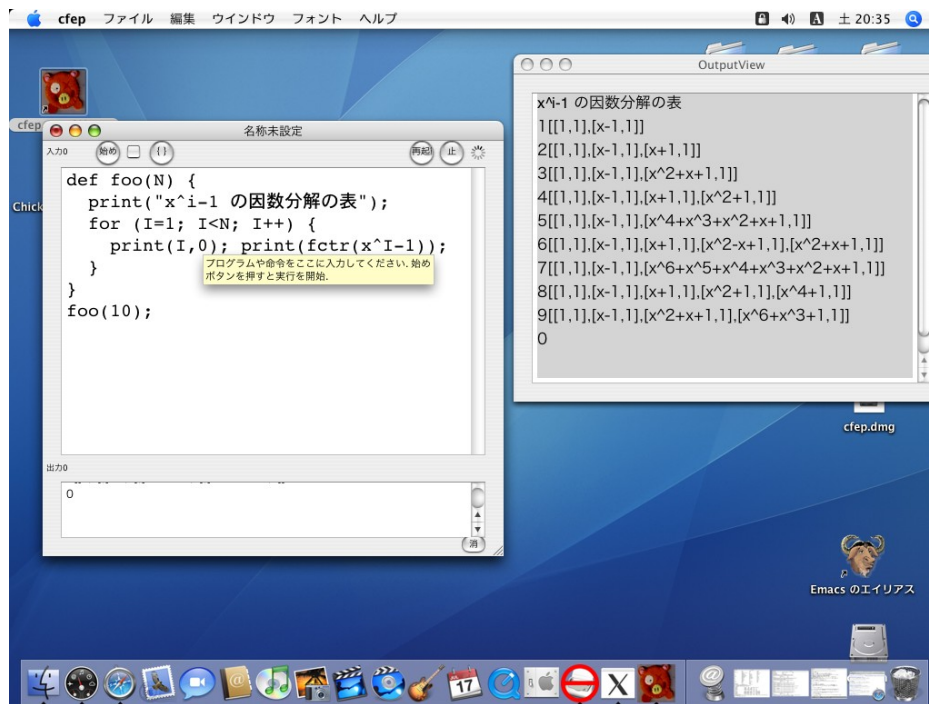


cfep/asir および texmacs インタ フェースの内部構造

- 高山信毅（神戸大学）
- 2008-03-18
- Risa/Asir Conference 2008



cfep/asirとは

- cfep/asir : 2006年度から神戸大学の教育システムがすべて Mac に. Mac用の asir gui
- 教養原論 (高山) 2006, 微分方程式と差分法 20名程度
<http://fe.math.kobe-u.ac.jp/Movies/cm/2006-genron-nt.html>
- 教養原論 (野呂) 2006, RSA暗号その他. 文字コード表. 50名程度
- 計算数学 I (高山) 2006, 30名程度.
<http://fe.math.kobe-u.ac.jp/Movies/cm/2006-keisan-1-nt.html>
- 教養原論 (野呂) 2007, RSA暗号その他 100名程度.
<http://www.math.kobe-u.ac.jp/~noro/g-07/index.html> 復号するのが課題.
- 計算数学 I (野呂) 2007, 30名程度.
- google で cfep/asir はたった6件. --> 神戸大学以外では利用実績なし.
- <http://www.math.kobe-u.ac.jp/~taka/2005/cfep/pdf/next2.pdf> 入門書

cfep/asir 超入門

- <http://www.math.kobe-u.ac.jp/~ta>
入門書

1.2 Cfep/Asir の起動法と電卓的な使い方

cfep のアイコン (いのふた君)

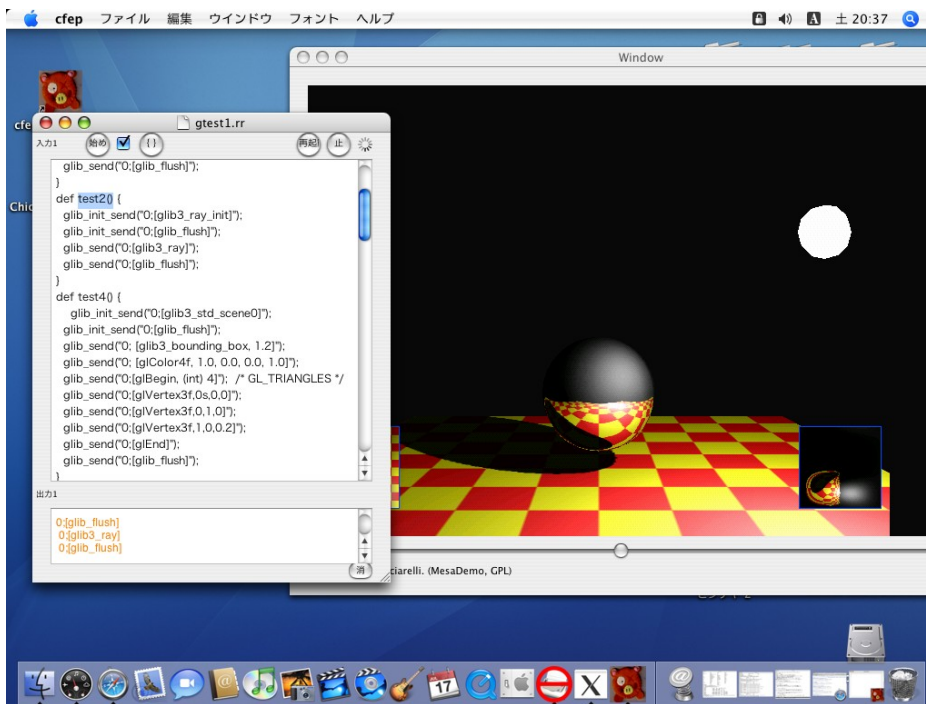


をダブルクリックすると図 1.1 のように cfep/asir が起動する. 以下 cfep/asi
図 1.1 の入力窓に計算したい式やプログラムを入力して“始め”ボタン



利用方法

- X11, xcode(cpp を asir が使うだけ) をインストールしておく.
- あとはいのぶた君をdrag&dropでインストールが終了
- 書いたものがそのまま実行される.
- 右は cfep で OpenGL を利用している例.
- くわしくはヘルプの入門文書を参照.



現在までの問題点

```
nobuki-takayama-nokonpyuta:~ nobuki$ cd /Applications/Kobe
nobuki-takayama-nokonpyuta:/Applications/Kobe nobuki$ ls
Emacs.app          TeXShop.app
KEdit.app          VLC
Pixen.app          cfep.app
Remote Desktop Connection  orange2.command
nobuki-takayama-nokonpyuta:/Applications/Kobe nobuki$ cd cfep.app
nobuki-takayama-nokonpyuta:/Applications/Kobe/cfep.app nobuki$ ls
Contents          OpenXM
nobuki-takayama-nokonpyuta:/Applications/Kobe/cfep.app nobuki$ ls OpenXM
bin      doc      lib      rc
nobuki-takayama-nokonpyuta:/Applications/Kobe/cfep.app nobuki$ █
```

- 初心者向きには **おおむね順調**.
- 配布版の asir は PPC のみ. Intel CPU で遅いので cfep.app/OpenXM/bin 以下を Intel 版にとりかえる必要あり.
- UTF8 をはずれた部分は control channel.
`print(asciitostr(0xf8)+asciitostr(0xf0)+asciitostr(0xf1))` でトラブル. 以下の channel の部分を参照
- graphic 画面を印刷するコマンドがない => `apple+shift+4` で代用.
- debugger がない.

内部構造1



- CoCoA で書かれたユーザインタフェース部分 cfep + C で書かれた OpenXM server 群. NSTask() でパイプでつなぐ.
- インタフェースは xcode, Objective C で開発
- 教訓: CoCoA のプログラムで C の標準関数を混ぜては変なことがおきる. 動いてるように見えて, 不可解なエラーが.. Java をつかってみたりするも不可解なエラーが... あきらめかけて, 結局上の内部構造に.

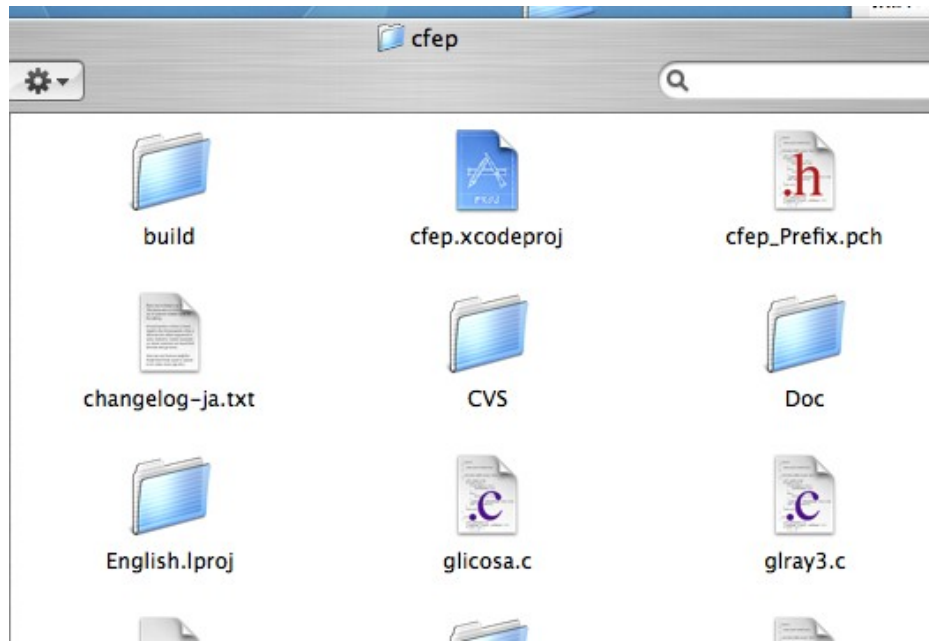
内部構造2



- デザインパターンと object 指向 MVC モデル (Model, View, Controller)
- ユーザインタフェースで自己主張 (新しい考えの導入)は一切しない.
- テンプレートとした実例
/Developer/Examples/AppKit/TextEdit
<http://developer.apple.com/jp/do>
- ソースの解読には

```
NSLog(@"count=%d\n", count);  
NSLog(@"obj=%@\n", myobj);
```
- <http://www.cocoabuilder.com/>

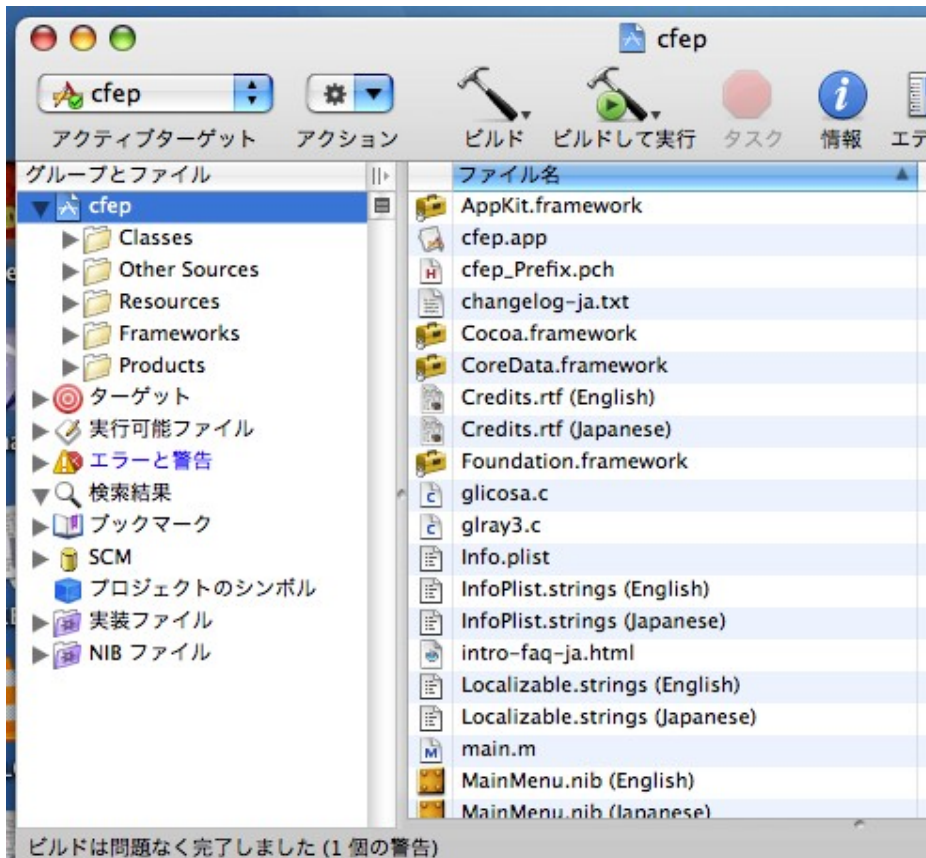
Rebuild1



- 以下 `cd OpenXM/src/cfep` として説明.
- `package/readme-ja.txt` にしたが
い `OpenXM` を `build/debug` の下へ
symbolic link.
- `cfep.xcodeproj` を `xcode` で開く.

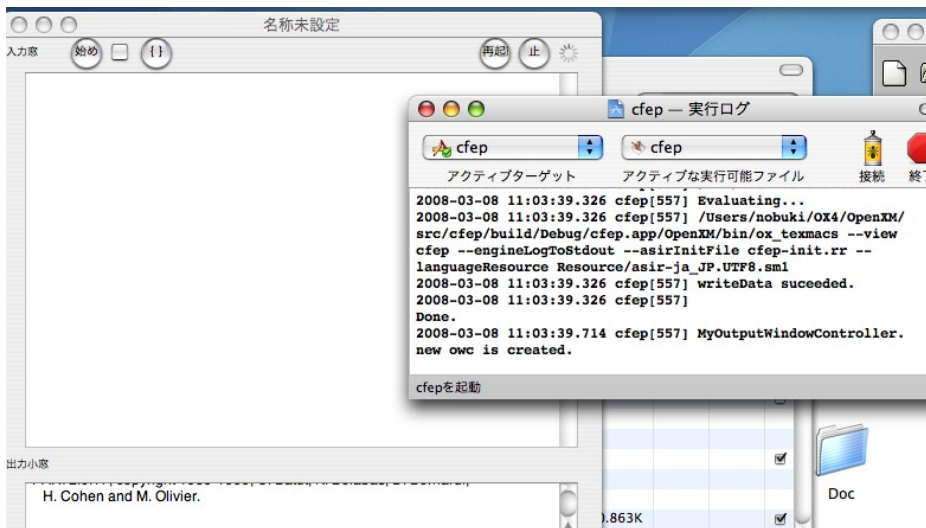
Rebuild2

- ビルドして実行をクリック



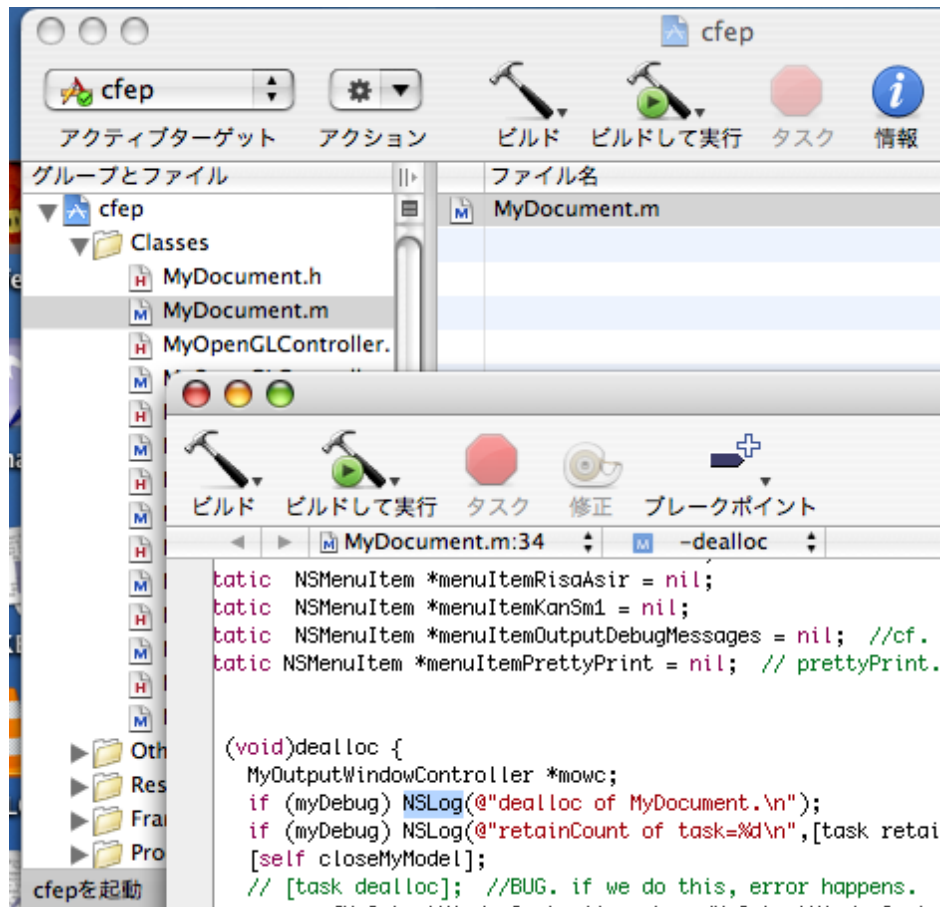
Rebuild3

- `NSLog(@"count=%d\n", count);`
`NSLog(@"obj=%@\n", myobj);`



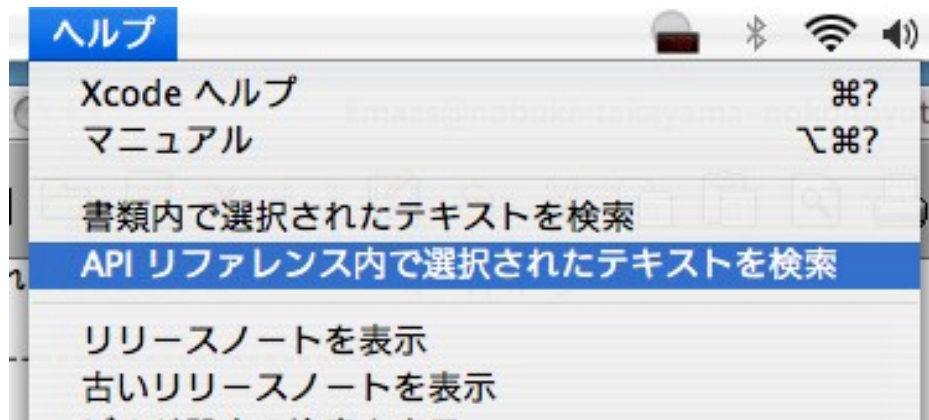
Rebuild4

- APIのヘルプの閲覧



Rebuild5

- APIヘルプの閲覧



Rebuild6

- 表示される解説

The screenshot shows the Xcode documentation browser for the 'Foundation Functions (Objective-C) - デベロッパマニュアル'. A search bar at the top right contains the text 'NSLog'. Below the search bar, a table lists search results for 'NSLog'. The table has columns for 'シンボル' (Symbol), 'タイプ' (Type), '言語' (Language), and '上' (Upper/Lower case). The results include several NSLog-related methods and functions, with 'NSLog' (function, C language) selected. Below the table, the documentation for 'NSLog' is displayed, including a description, a code snippet, a discussion, and a 'See Also' link.

シンボル	タイプ	言語	上
M NSLog.PrintStreamLogger()	メソッド	Java	co
M NSLog.PrintStreamLogger(java.io.PrintStream)	メソッド	Java	co
M NSLog.PrintStreamLogger(java.io.PrintStream)	メソッド	Java	co
F() NSLog	関数	C	
C NSLogicalTest	クラス	Obj-C	NS
M NSLogicalTest	メソッド	Java	co
F() NSLogPageSize	関数	C	
F() NSLogv	関数	C	

NSLog
Logs error messages to stderr.
`void NSLog(NSString *format, ...)`

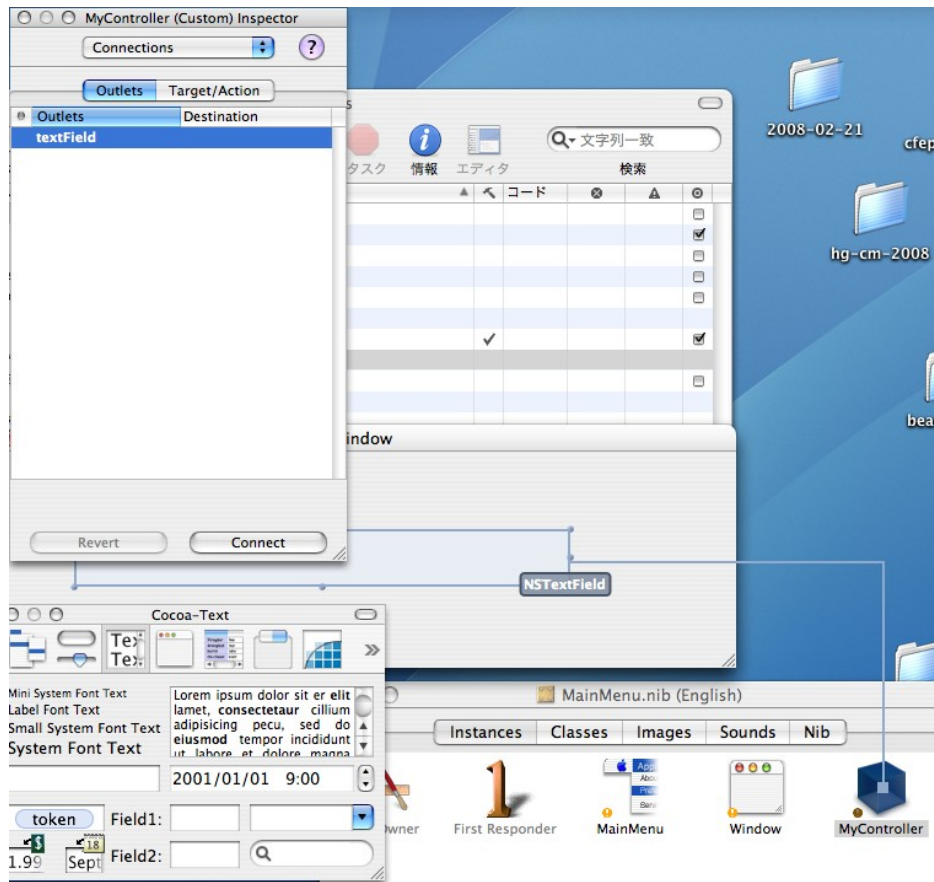
Discussion
Simply calls [NSLogv](#), passing it a variable number of arguments.

See Also: [NSLogv](#)

ソースコードの構成

- `MyDocument.m` Model (うしろに `ox_texmacs` が控えている。NSTask method で呼び出す。) これが中心.
- `MyOutputWindowController.m` Controller の一つ。OutputWindow 用
- `Japanese.lproj/MyDocument.nib` View (interface builder で自動生成)
- `Japanese.lproj/MainMenu.nib`
- `English.lproj/MyOutputWinodw.nib` View (interface builder で自動生成)
- `English.lproj/MyOutputWinodw.nib/classes.nib` をみると、対応が書いてある.
- <http://fe.math.kobe-u.ac.jp/Movies/cm/2006-03-00-devel-cfep-1.html>

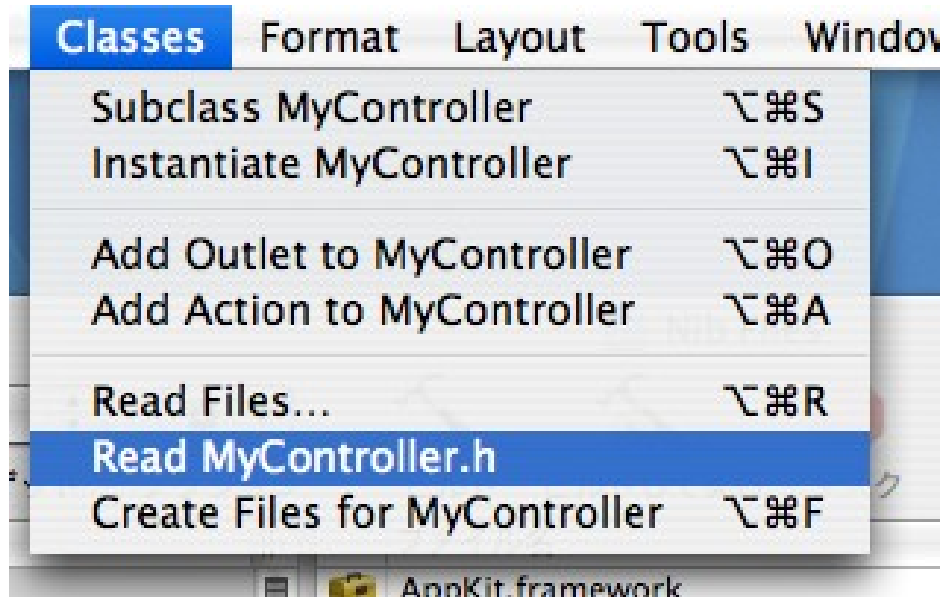
わき道: Interface builder



- CoCoA 入門の例題より
- outlet の接続 (ctrl-drag)

わき道: Interface builder

- 逆向きも可能.
- MyController.h を直接編集して outlet を追加. Interface builder へ反映させる.



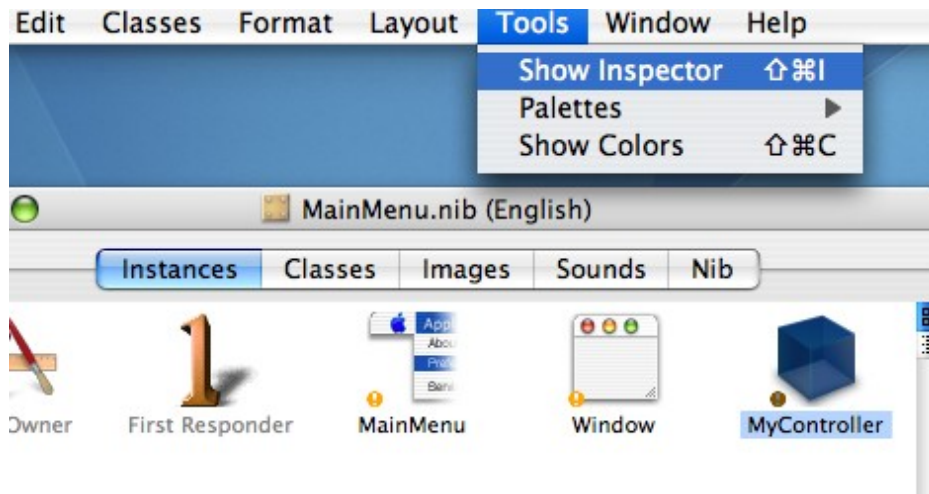
わき道: Interface builder

```
@interface MyController : NSObject
{
    IBOutlet id textField;
    IBOutlet id textField2;
}
@end
```

- id textField を NSTextField *textField に変更.
- これらは object MyController のインスタンス変数となる.
- これらの変数 (object) にメッセージを送るとwindowに反映される.

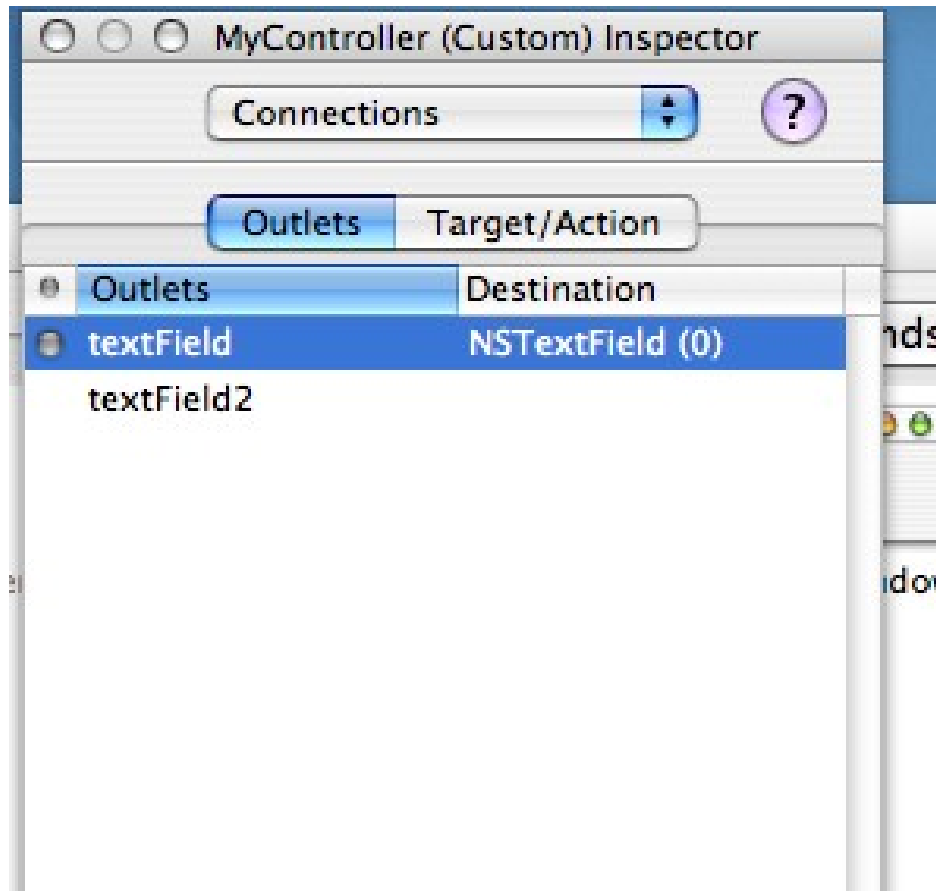
わき道: Interface builder

- Inspector を起動するとどのような変数が window のどの部品と対応しているかわかる。



わき道: Interface builder

- double click すると
図示される。



わき道: Interface builder

- awakeFromNib のプロトタイプ宣言.

```
/* MyController */  
  
#import <Cocoa/Cocoa.h>  
  
@interface MyController : NSObject  
{  
    IBOutlet NSTextField *textField;  
    IBOutlet id textField2;  
}  
- (void) awakeFromNib;  
@end
```

わき道: Interface builder

- インスタンスmethod
awakeFromNibの実体.
- [object メッセージ:
引数1 キー1 :
引数2 ...] なる形式.

```
#import "MyController.h"

@implementation MyController
- (void) awakeFromNib
{
    [textField setValue:[NSDate date]];
}
@end
```

ソースコードの解説メモビデオ

- <http://fe.math.kobe-u.ac.jp/Movies/cm/2006-03-00-devel-cfep-1.html>
- xcode の使いかた.
- objective C の使い方.
- release 版の作り方.
- OpenGL の新しいコマンドを加える例.
- 新しい error メッセージを加える例.
- その他注意事項.

チャンネルの仕組み

MyDocument.m の instance method

```
-(void) readInboundData: (NSNotification*)aNotification
/* 1 openGL, 2 openGLInit, 10 pngAction (TeX の画像) */
    if ((c >> 5) == UTF8_2) { u0=c ; state=1; }
    else if ((c >> 4) == UTF8_3) { u0 =c ; state = 2; }
    else if ((c >> 2) == MYDECODER) { u0=c; state = 4; }
/* UTF8_2 == 0x6    110  utf8 2 byte data の header.
   UTF8_3 == 0x5    1110 utf8 3 byte data の header.
   111110 channel data の header. (3 bytes で 1 byte) */
```

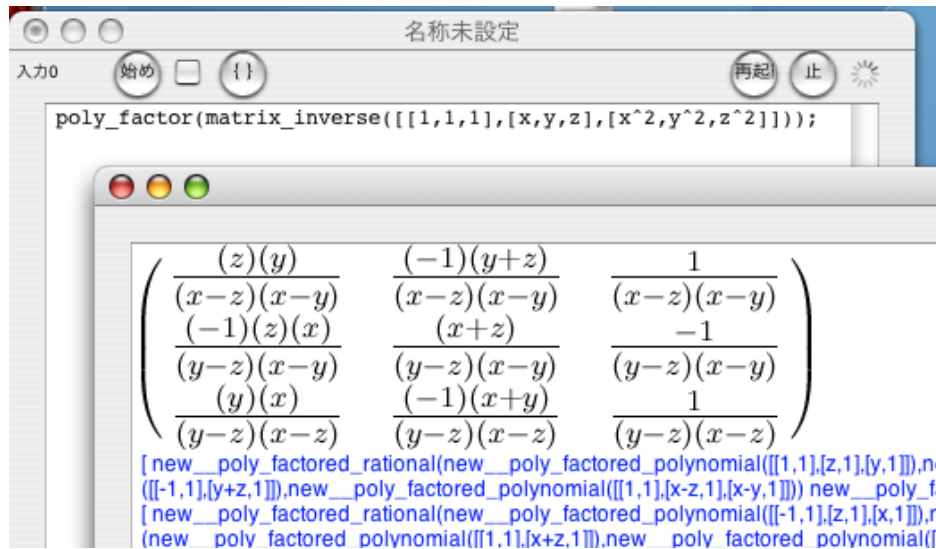
チャンネルの仕組み2

```
OpenXM/src/cfep/Samples/simple-gl-1.rr
```

```
N=length(strtoascii(S))+1;  
C=t_encode("{1<"+rtostr(N)+" "+S+" >}");  
ctrl("hex",1);  
output(); ctrl("hex",0);  
print(C); output();
```


チャンネルのしくみ3

- 例：チャンネル10に dvipng で生成したデータを送る.



```
poly_factor(matrix_inverse([[1,1,1],[x,y,z],[x^2,y^2,z^2]]));
```

$$\begin{pmatrix} \frac{(z)(y)}{(x-z)(x-y)} & \frac{(-1)(y+z)}{(x-z)(x-y)} & \frac{1}{(x-z)(x-y)} \\ \frac{(-1)(z)(x)}{(y-z)(x-y)} & \frac{(x+z)}{(y-z)(x-y)} & \frac{-1}{(y-z)(x-y)} \\ \frac{(y)(x)}{(y-z)(x-z)} & \frac{(-1)(x+y)}{(y-z)(x-z)} & \frac{1}{(y-z)(x-z)} \end{pmatrix}$$

[new__poly_factored_rational(new__poly_factored_polynomial([[1,1],[z,1],[y,1]]),ne
[[-1,1],[y+z,1]]).new__poly_factored_polynomial([[1,1],[x-z,1],[x-y,1]]) new__poly_fa
[new__poly_factored_rational(new__poly_factored_polynomial([[1,1],[z,1],[x,1]])n
(new__poly_factored_polynomial([[1,1],[x+z,1]]).new__poly_factored_polynomial([

エラーメッセージはどのようにだしてるの？

エラー: この代入はできません.

原因例: X=1 はできるが x=1 はできない. 大文!

エラー行: 4

エラー行: 2

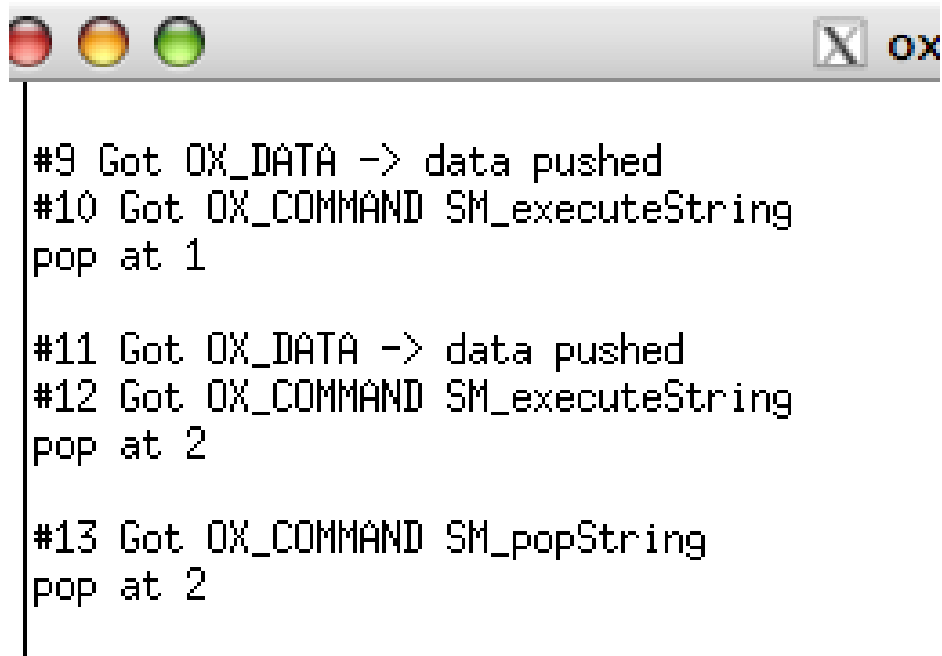
エラー行: 2

エラー行: 2

エラー行: 2

- ```
def foo(N) { if (N<1) { x=N;
return x; } return N*foo(N-1);}
```
- ```
foo(3);
```
- ```
error([41, 4294967295, eval :
invalid
assignment, [asir_where, [[toplevel, 5], [string, foo, 3], [string, foo, 3],
[string, foo, 3], [string, foo, 2]]])
```
- 日本語メッセージについては  
[OpenXM/src/kan96xx/Doc/Resource/asir-ja\\_JP.UTF8.sm1](http://OpenXM/src/kan96xx/Doc/Resource/asir-ja_JP.UTF8.sm1)

# asir の print 文をどのように出力しているの? ox\_asir の初期化

A terminal window with a title bar containing three colored buttons (red, yellow, green) and a close button labeled 'ox'. The terminal displays the following text:

```
#9 Got OX_DATA -> data pushed
#10 Got OX_COMMAND SM_executeString
pop at 1

#11 Got OX_DATA -> data pushed
#12 Got OX_COMMAND SM_executeString
pop at 2

#13 Got OX_COMMAND SM_popString
pop at 2
```

- OpenXM/src/asir-contrib/packages/src/cfep-init.rr
- `ctrl("debug_window",0);`
- `ctrl("double_output",1);`
- `flush(NULL);`
- `ox_texmacs.c`
- `ctrl("message",0);` push pop のメッセージを出力しない.

# asir の print 文をどのように出力しているの? ox\_asir の初期化2

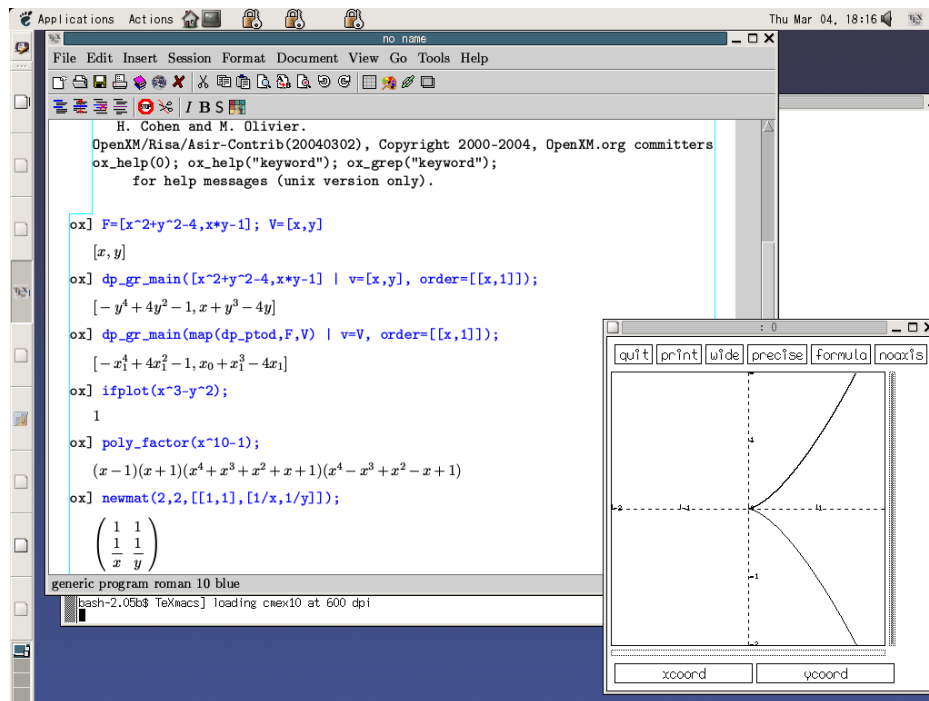
```
OpenXM/Risa/Asir-Contrib(20040302), Copyright
rs
ox_help(0); ox_help("keyword"); ox_grep("keywo
 for help messages (unix version only).
#14 Got OX_DATA -> data pushed
#15 Got OX_COMMAND SM_executeString
pop at 2

#16 Got OX_COMMAND SM_popCMD
pop at 2

#17 Got OX_DATA -> data pushed
#18 Got OX_COMMAND SM_executeString
pop at 2
----- Messages from asir -----
</S></S>print("Hello");
1+2;
^E
Hello
<S type=verbatim><S type=verbatim>3</S></S>
```

- OpenXM/src/kxx/ox\_texmacs.c
- `ox_texmacs --view cfep --engineLogToStdout`
- OpenXM/src/kan96xx/Doc/ox.sm1  
Xm\_engineLogToStdout
- OpenXM/src/kxx/ox100start.c  
EngineLogToStdout
- `ox_texmacs --view debug --engineLogToStdout`

# TeXmacs, cfep --> ox\_texmacs-->ox servers



- ox\_texmacs から OpenXM-RFC 100, 103 プロトコルを介して ox\_asir サーバ.
- DATA\_BEGIN 2
- DATA\_END 5
- verbatim:
- latex:
- TeXmacs manual. Part A User Guide.
- Chapter 9. Using GNU TeXmacs as an interface

```

ox_texmacs.c
while (1) {
 if (SETJMP(EnvOfStackMachine)) {
 /* 中断 (ctrl-C) の時の処理 */ irt=1;
 } else { }
 if (!irt) {
 printf("%s",Data_end[View]); fflush(stdout);
 }
 irt=0;
 s=readString(stdin, "if (1) { ", " ; }else{ }");
 if (s == NULL) { irt = 1; continue; }
 printf("%s",Data_begin_v[View]);
 /* ox サーバで入力を評価 */
 KSexecuteString(" ox.engine oxclearstack ");
 KSexecuteString(" ox.engine ");
 ob = KpoString(s); KSpush(ob);
 KSexecuteString(" oxsubmit ");
 KSexecuteString(" ox.engine oxgeterrors ");
 ob = KSpop();
 if (ob.tag == Sarray) { /*エラーがあるときの処理*/ continue; }
 KSexecuteString(" ox.engine oxpopstring ");
 r = KSpopString(); printv(r);
}

```

cvsweb を見るなら 2006年1月—3月ごろ.

<http://www.math.sci.kobe-u.ac.jp/cgi/cvsweb.cgi/>

## kan/sm1 の設計思想.

C のライブラリ.

ライブラリを安全にコントロールするための小さい言語(sm1).  
(1991, 1994)

## ox\_texmacs との関係

kanlib.a が ox\_texmacs.c にリンクされる.

## 主なファイル一覧

OpenXM/src/kan96xx/Doc/changelog-ja.tex

OpenXM/src/kan96xx/Kan/stackm.h データ型

OpenXM/src/kan96xx/Kan/scanner\*.c PostScript風インタプリタ.

OpenXM/src/kan96xx/Kan/kanExport\*.c operator の処理.

OpenXM/src/kan96xx/plugin 雑務. ネットワーク関連を含む.

OpenXM/src/kan96xx/Kan/parser\*.c 多項式のパーサ.

OpenXM/src/kan96xx/Kan/poly\*.c 多項式の演算.

OpenXM/src/kan96xx/Kan/red\*.c, ecart.c 割り算.

OpenXM/src/kan96xx/Kan/gb\*.c Grobner basis

# kan/sm1 ソース解読のヒント

errorPacket (body) dc 3 get がリスト (キーワード, 値) となる. 場所 (where) や理由 (reason), 処理系によっては, line や parse error の場所. 関連事項: misc-2005/A1/cfep. oxclearstack.

ox\_textmacs で tunnel channel 0 へ cfep 用のエラー命令を送る. (outputStringToTunnel())

```
ox_textmacs --view debug
```

```
!sm1;^E
```

```
1 shell ^E
```

ソースの変更.

1. OpenXM/src/kan96xx/Kan/stackmachine.c 1.34
2. OpenXM/src/kan96xx/Doc/ox.sm1 1.48
3. OpenXM/src/kxx/smistackmachine.c 1.6
4. OpenXM/src/kxx/ox\_textmacs.c 1.27
5. OpenXM/doc/OpenXM-specs/OX-RFC-103.oxw 1.6

- OpenXM/src/kan96xx/Doc/changelog-ja.tex

- object, タグがついている.

- printObject() 関数で表示.

- まとめ. 詳しくはスライドから参照したビデオ資料 (commit video) および cvsweb.