Mathematica他言語リンクについて

Symbolic Systems (matsuda@symboclis.jp) 2010.3.24
http://www.symbolics.jp/lecturenotes.html

```
Off[General::"spell1"];

SetDirectory[NotebookDirectory[]];

FileNames[]
```

```
{a.c, addtwo, addtwo.c, addtwo.o, addtwo.tm, addtwotm40.c, addtwotm.c, addtwotm.o, a.out,
 bitops, bitops.c, bitops.o, bitops.tm, bitopstm.c, bitopstm.o, counter, counter.tm,
 countertm.c, countertm.o, factor, factor2, factor2.c, factor2.o, factor3, factor3.c,
 factor3.o, factor.c, factor.c.org, factor.o, list, list.c, list.o, makefile, MathKernel,
 mathlink, mathlink.c, mathlink.o, ._mathlink-sample.nb, mathlink-sample.nb,
 MyTimerClass.class, MyTimerClass.java, quotient, quotient.c, quotient.o, reverse,
 reverse.tm, reversetm.c, reversetm.o, sumalist, sumalist.tm, sumalisttm.c, sumalisttm.o}
```

---

from *Mathematica* to C(C++)

- C(C++)のmain付きプログラムを準備
- mainを除く関数部分のみを取り出し，includeファイル"mathlink.h"を追加.
- 関数プロトコルに関し tmファイルを準備．C関数とMathematica関数を結びつける.
- Mathematicaが提供するCコンパイラ mccを使い *.c ファイルならびに *.tmファイルをコンパイルする.
- Mathematicaから Install関数を使い上記で作られた実行形式を取り込む.
- あとはtmファイルで設定したプロトコルに従い自前で作った関数を実行する.

参照：tutorial/MathLinkAndExternalProgramCommunicationOverview

```
FilePrint["addtwo.c"]
```

```
#include "mathlink.h"
extern int addtwo( int i, int j);

int addtwo( int i, int j)
{
    return i+j;
}

int main(int argc, char* argv[])
{
    return MLMain(argc, argv);
}
```

```
FilePrint["addtwo.tm"]
```

```
int addtwo P(( int, int));

:Begin:
:Function:       addtwo
:Pattern:        AddTwoF[i_Integer, j_Integer]
:Arguments:      { i, j }
:ArgumentTypes:  { Integer, Integer }
:ReturnType:     Integer
:End:

:Evaluate: AddTwoF::usage = "AddTwoF[x, y] gives the sum of two machine integers x and y."
```

```
addtwolnk = Install["addtwo"];
```

```
? AddTwoF
```

AddTwoF[x, y] gives the sum of two machine integers x and y.

```
AddTwoF[1, 3]
```

```
4
```

```
Uninstall[addtwolnk];
```

```
AddTwoF[2, 3]
```

```
AddTwoF[2, 3]
```

at /Applications/Mathematica.app/SystemFiles/Links/MathLink/DeveloperKit/CompilerAdditions
sudo cp mprep /usr/local/bin/mprep

```
Run["mcc -o addtwo addtwo.tm addtwo.c"](* C compiler for C program for Mathematica *)
```

```
32 512
```

or use the following makefile which exlicitly specifys the several environments

```
FilePrint["makefile"]
```

```
# This makefile can be used to build all or some of the sample
# programs.  To build all of them, use the command
# 'make all'.  To build one, say addtwo, use the command
# 'make addtwo'.

VERSION=7.0
MLINKDIR = /Applications/Mathematica.app/SystemFiles/Links/MathLink/DeveloperKit
SYS = MacOSX-x86-64
CADDSDIR = ${MLINKDIR}/CompilerAdditions
#EXTRA_CFLAGS=-Wno-long-double
# long double: floating point more precise than double precision.
EXTRA_CFLAGS=

INCDIR = ${CADDSDIR}
LIBDIR = ${CADDSDIR}

MPREP = ${CADDSDIR}/mprep
RM = rm

CC = /usr/bin/gcc
CXX = /usr/bin/c++

BINARIES = addtwo bitops counter factor factor2 factor3 quotient reverse sumalist

all : $(BINARIES)
```

```
addtwo : addtwotm.o addtwo.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} addtwotm.o addtwo.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

bitops : bitopstm.o bitops.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} bitopstm.o bitops.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

counter : countertm.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} countertm.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

factor : factor.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} factor.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

factor2 : factor2.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} factor2.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

factor3 : factor3.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} factor3.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

quotient : quotient.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} quotient.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

reverse : reversetm.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} reversetm.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

sumalist : sumalisttm.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} sumalisttm.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

mathlink : mathlink.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} mathlink.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@

list : list.o
    ${CXX} ${EXTRA_CFLAGS} -I${INCDIR} list.o -L${LIBDIR} -lMLi3 -lstdc++ -o $@
.c.o :
    ${CC} -c ${EXTRA_CFLAGS} -I${INCDIR} $<

addtwotm.c : addtwo.tm
    ${MPREP} $? -o $@

bitopstm.c : bitops.tm
    ${MPREP} $? -o $@

countertm.c : counter.tm
    ${MPREP} $? -o $@

reversetm.c : reverse.tm
    ${MPREP} $? -o $@

sumalisttm.c : sumalist.tm
    ${MPREP} $? -o $@

clean :
    @ ${RM} -rf *.o *tm.c $(BINARIES)
```

```
Run["head -40 addtwotm.c > addtwotm40.c"]; FilePrint["addtwotm40.c"]
```

```
/*
 * This file automatically produced by /Applications/Mathematica.app/SystemFiles/Links/MathLink/
 *  addtwo.tm
 * mprep Revision 14 Copyright (c) Wolfram Research, Inc. 1990-2008
 */

#define MPREP_REVISION 14

#if CARBON_MPREP
#include <Carbon/Carbon.h>
#include <mathlink/mathlink.h>
#else
#include "mathlink.h"
#endif

int MLAbort = 0;
int MLDone  = 0;
long MLSpecialCharacter = '\0';

MLINK stdlink = 0;
MLEnvironment stdenv = 0;
#if MLINTERFACE >= 3
MLYieldFunctionObject stdyielder = (MLYieldFunctionObject)0;
MLMessageHandlerObject stdhandler = (MLMessageHandlerObject)0;
#else
MLYieldFunctionObject stdyielder = 0;
MLMessageHandlerObject stdhandler = 0;
#endif /* MLINTERFACE >= 3 */

#if DARWIN_MATHLINK && CARBON_MPREP
#define rMenuBar    1128
#define rAboutBox   1128
#define rBadSIZE    1127
#define mApple      1128
#define iAbout      1
#define mFile       1129
#define mEdit       1130

AEEventHandlerUPP handle_core_ae_upp;
ModalFilterUPP about_filter_upp;
```

```
FilePrint["bitops.c"]
```

```
#include "mathlink.h"

int  bitand( int x, int y);

void complements( int *px, long nx);

int  bitand( int x, int y)
{
    return( x & y);
}

void complements( int *px, long nx)
{
    long    i;
    for( i = 0; i < nx; i++)
        px[i] = ~ px[i] ;
    MLPutIntegerList( stdlink, px, nx);
}

int main(int argc, char* argv[])
{
    return MLMain(argc, argv);
}
```

```
FilePrint["bitops.tm"]
```

```
int bitand P(( int, int));

:Begin:
:Function: bitand
:Pattern: bitAndF[x_Integer, y_Integer]
:Arguments: {x, y}
:ArgumentTypes: {Integer, Integer}
:ReturnType: Integer
:End:

:Evaluate: bitAndF::usage = "bitAndF[x, y] gives the bitwise conjunction
    of two integers x and y."


void complements P(( int *, long));

:Begin:
:Function: complements
:Pattern: bitComplementsF[x_List]
:Arguments: {x}
:ArgumentTypes: {IntegerList}
:ReturnType: Manual
:End:

:Evaluate: bitComplementsF::usage = "bitComplementsF[{x1,x2,...}] generates
    a list of the bitwise complements of the integers xi."
```

```
bitopslnk = Install["bitops"];
```

```
LinkObject[/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
   JLink(2010.3.24)/MathLinkExamples/bitops, 10, 10]
```

```
bitAndF[7, 3]
```

```
3
```

```
BitAnd[7, 3]
```

```
3
```

```
bitComplementsF[{1, 2, 3}]
```

```
{-2, -3, -4}
```

```
BitNot[1] (* 二の補数, BitNot[n]--> -1-n *)
```

```
-2
```

```
Uninstall[bitopslnk];
```

```
/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
  JLink(2010.3.24)/MathLinkExamples/bitops
```

```
counterlnk = Install["counter"];
```

```
LinkObject[/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
   JLink(2010.3.24)/MathLinkExamples/counter, 5967, 10]
```

```
FilePrint["counter.tm"]
```

```
#include "mathlink.h"

int AddToCounter P(( int));

:Evaluate:      BeginPackage[ "counter`"]

:Begin:
:Function:      AddToCounter
:Pattern:       AddToCounterF[$CurrentLink, n_Integer]
:Arguments:     {n}
:ArgumentTypes: {Integer}
:ReturnType:    Integer
:End:

:Evaluate:      AddToCounterF::usage = "AddToCounterF[ ck, n] adds n to the counter
        ck and returns the accumulated value."

:Evaluate:      EndPackage[ ]


int counter = 0;

int   AddToCounter( n) int n;
{
    counter += n;
    return counter;
}

int main(argc, argv)
    int argc; char* argv[];
{
    return MLMain(argc, argv);
}
```

```
AddToCounterF[counterlnk, 3]
```

```
3
```

```
AddToCounterF[counterlnk, 4]
```

```
7
```

```
AddToCounterF[counterlnk, 8]
```

```
15
```

```
Uninstall[counterlnk];
```

```
/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
  JLlink(2010.3.24)/MathLinkExamples/counter
```

```
reverselnk = Install["reverse"];
```

```
LinkObject[/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
   JLlink(2010.3.24)/MathLinkExamples/reverse, 5968, 10]
```

```
FilePrint["reverse.tm"]
```

```
#include "mathlink.h"

void reverse_string P((const unsigned short *s, int len));

:Begin:
:Function:      reverse_string
:Pattern:       reverseStringF[s_String]
:Arguments:     {s}
:ArgumentTypes: {UCS2String}
:ReturnType:    Manual
:End:

#define BUFCAP 64

void reverse_string(const unsigned short *s, int len)
{
    unsigned short buf[BUFCAP], *p = buf;
    long n = BUFCAP;

    MLPutNext( stdlink, MLTKSTR);
    while( len > 0){
        if( n-- == 0){
            MLPutUCS2Characters( stdlink, len, buf, (int)(p - buf));
            n = BUFCAP - 1;
            p = buf;
        }
        *p++ = s[--len];
    }
    MLPutUCS2Characters( stdlink, len, buf, (int)(p - buf));
}

int main(argc, argv)
    int argc; char* argv[];
{
    return MLMain(argc, argv);
}
```

```
reverseStringF["abcd123"]
```

```
321dcba
```

```
Uninstall[reverselnk];
```

```
/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
  JLlink(2010.3.24)/MathLinkExamples/reverse
```

```
sumalistlnk = Install["sumalist"];
```

```
LinkObject[/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
   JLlink(2010.3.24)/MathLinkExamples/sumalist, 5969, 10]
```

```
FilePrint["sumalist.tm"]
```

```
#include "mathlink.h"

int sumalist P(( int *, long));

:Begin:
:Function:      sumalist
:Pattern:       SumAListF[ list:{___Integer} ]
:Arguments:     { Reverse[list] }
:ArgumentTypes: { IntegerList }
:ReturnType:    Integer
:End:

:Evaluate:      SumAListF[ sequence___Integer]:= SumAListF[ {sequence} ]


int sumalist( int *list, long len)
{
        int res = 0;
        while(len--)
            res += *list++;
        return res;
}


int main(argc, argv)
    int argc; char* argv[];
{
    return MLMain(argc, argv);
}
```

```
SumAListF[{1, 2, 3, 4}]
```

```
10
```

```
Uninstall[sumalistlnk];
```

```
/Volumes/Untitled/講義/Mathematica講習会/Mathematica中級コース(2009)/MathLink,
  JLlink(2010.3.24)/MathLinkExamples/sumalist
```

Mathematica specification		C specification
Integer	integer	int
Real	floating-point number	double
IntegerList	list of integers	int *, long
RealList	list of floating-point numbers	double *, long
String	character string	char *
Symbol	symbol name	char *
Manual	call *MathLink* routines directly	void

from C(C++) to *Mathematica*

- Mathematicaカーネルへのリンク(MathLink使用)を確立する.
- Mathematicaにやってほしい計算のために必要な関数，データをMathLinkに送り込む．この際，単純な数値だけでなくリストなどを送る場合にはC側でリスト構造を作ってあげる必要がある（専用関数が用意）.
- Mathematicaから受け取った答え（たいていはリスト）をC側で分解処理する（このためにも専用関数用意）.

```
FilePrint["factor.c"]
```

```
/* To run this program use the command-line below:
 *  Unix:           factor -linkname "math -mathlink"
 *  Mac or Windows: factor -linkmode launch
 */


#include <stdio.h>
#include <stdlib.h>

#include "mathlink.h"

static void init_and_openlink( int argc, char* argv[]);
static void error( MLINK lp);


MLENV ep = (MLENV)0;
MLINK lp = (MLINK)0;


int main(int argc, char* argv[])
{
    int pkt, n, prime, expt;
    long len, lenp, k;

    init_and_openlink( argc, argv);

    printf( "Integer to factor: ");

    scanf( "%d", &n);

    MLPutFunction( lp, "EvaluatePacket", 1);
    MLPutFunction( lp, "FactorInteger", 1);
    MLPutInteger( lp, n);
    MLEndPacket( lp);
    while( (pkt = MLNextPacket( lp), pkt) && pkt != RETURNPKT) {
        MLNewPacket( lp);
        if (MLError( lp)) error( lp);
    }
    if ( ! MLCheckFunction( lp, "List", &len)) error(lp);
    for (k = 1; k <= len; k++) {
        if (MLCheckFunction( lp, "List", &lenp)
        &&  lenp == 2
        &&  MLGetInteger( lp, &prime)
        &&  MLGetInteger( lp, &expt)
        ){
            printf( "%d ^ %d\n", prime, expt);
        }else{
            error( lp);
        }
    }

    MLPutFunction( lp, "Exit", 0);

    return 0;
}
```

```
static void error( MLINK lp)
{
    if( MLError( lp)){
        fprintf( stderr, "Error detected by MathLink: %s.\n",
            MLErrorMessage(lp));
    }else{
        fprintf( stderr, "Error detected by this program.\n");
    }
    exit(3);
}


static void deinit( void)
{
    if( ep) MLDeinitialize( ep);
}


static void closelink( void)
{
    if( lp) MLClose( lp);
}


static void init_and_openlink( int argc, char* argv[])
{
    int err;

    ep =  MLInitialize( (MLParametersPointer)0);
    if( ep == (MLENV)0) exit(1);
    atexit( deinit);
    lp = MLOpenArgv( ep, argv, argv + argc, &err);
    if(lp == (MLINK)0) exit(2);
    atexit( closelink);
}
```

./factor -linkmode launch -linkname "/Applications/Mathematica.app/Contents/MacOS/MathKernel"

Integer to factor: 10

2 ^ 1

5 ^ 1

■ **二次元リスト作成**

```
int a[in][jn]={{1,2,3},{3,4,5},{6,7,8}};
init_and_openlink( argc, argv);
MLPutFunction( lp, "EvaluatePacket", 1);
MLPutFunction(lp,"List",in);
for (i=0; i<in; i++){
      MLPutFunction(lp,"List",jn);
      for(j=0; j<jn; j++){
            MLPutInteger32List(lp,a[i][j],1);
      }
}
MLEndPacket( lp);
```

from *Mathematica* to Java built-in class

- Javaクラスおよび各種MethodがMathematicaの要素として準備されている．したがってただこれを利用するだけ．ただし，一部，JavaとMathematicaを繋ぐものがあり，これらはMathematicaの関数として用意されている．

参照：JLink/tutorial/Overview

```
Needs["JLink`"]
```

■ Beep関数

```
Beep2[] :=
 (
  LoadJavaClass["java.awt.Toolkit"];
  Toolkit`getDefaultToolkit[]@beep[]
 )
```

getDefaultToolkitメソッドがTookkitオブジェクトを一つ生成．そのメソッドbeepを実行．
クラス@メソッド

```
Beep2[]
```

```
BetterBeep[]:=
    JavaBlock[
        InstallJava[];
        LoadJavaClass["java.awt.Toolkit"];
        Toolkit`getDefaultToolkit[]@beep[];
    ]
```

JavaBlockで囲まれた部分は全体の実行が終わった後解放される．
InstallJava Javaランタイムを起動．Beep2の例ではなくても動作したが．．．

```
(* Alternative version *)
BetterBeep2[]:=
    Module[{toolkit},
        InstallJava[];
        LoadJavaClass["java.awt.Toolkit"];
        toolkit = Toolkit`getDefaultToolkit[];
        toolkit@beep[];
        ReleaseJavaObject[toolkit]
    ]
```

ReleaseJavaObject 明示的にメモリ解放．ただしJavaBlockの方が利点あり．上記のように解放対象を変数(toolkit)として指定する必要ない．

■ 日付

```
date = JavaNew["java.util.Date"]
```

```
« JavaObject[java.util.Date] »
```

JavaNew 該当クラスのオブジェクトを作成

```
LoadJavaClass["java.text.DateFormat"];
dateFormatter = DateFormat`getInstance[]
```

```
« JavaObject[java.text.SimpleDateFormat] »
```

```
dateFormatter@format[date]
```

```
10/03/23 21:17
```

dateFormatter自身もインスタンス(オブジェクト).

■ プログレスバー

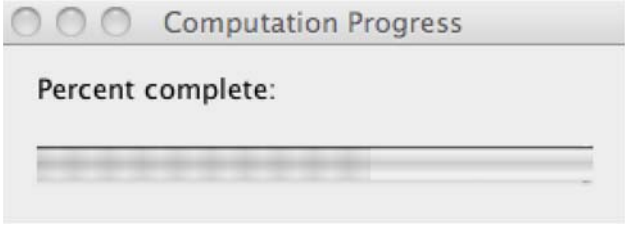
```
ShowProgressBar[title_String:"Computation Progress",
                caption_String:"Percent complete:",
                percent_Integer:0
                  ] :=
    JavaBlock[
        Module[{frame, panel, label, bar},
            InstallJava[];
            bar = JavaNew["javax.swing.JProgressBar"];
            frame = JavaNew["javax.swing.JFrame", title];
            frame@setSize[300, 110];
            frame@setResizable[False];
            frame@setLocation[400, 400];
            panel = JavaNew["javax.swing.JPanel"];
            panel@setLayout[Null];
            frame@getContentPane[]@add[panel];
            label = JavaNew["javax.swing.JLabel", caption];
            label@setBounds[20, 10, 260, 20];
            panel@add[label];
            bar@setBounds[20, 40, 260, 30];
            bar@setMinimum[0];
            bar@setMaximum[100];
            bar@setValue[percent];
            panel@add[bar];
            JavaShow[frame];
            bar
        ]
    ]
```

JProgressBar オブジェクト bar を一つ作成.
フレームオブジェクト frame (サイズ[300,110], リサイズ不可, 座標[400,400])作成.
パネルオブジェクト panel 作成. frame にはめ込む
panel, bar は frameに.

```
DestroyProgressBar[bar_?JavaObjectQ] :=
    JavaBlock[
        LoadJavaClass["javax.swing.SwingUtilities"];
        SwingUtilities`windowForComponent[bar]@dispose[];
        ReleaseJavaObject[bar]
    ]
```

dispose[] 後始末.

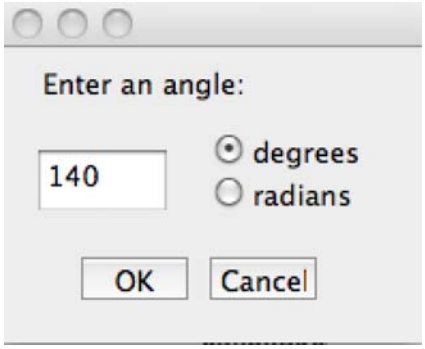
```
bar = ShowProgressBar[];
n = 0;
While[n <= 5,
    bar@setValue[n/5 * 100];
    Pause[1]; (* This simulates the time-consuming computation. *)
    n++
];
DestroyProgressBar[bar];
```



メニュー

```
GetAngle[] :=
    JavaBlock[
        Module[{frm, inputField, cbGroup, degBox, radBox,
                label, okButton, cancelButton, wasOKButton, angle},

            InstallJava[]; (* In case the user has not called it already. *)

            frm = JavaNew["com.wolfram.jlink.MathFrame"];
            label = JavaNew["java.awt.Label", "Enter an angle:"];
            inputField = JavaNew["java.awt.TextField"];
            cbGroup = JavaNew["java.awt.CheckboxGroup"];
            degBox = JavaNew["java.awt.Checkbox", "degrees", cbGroup, True];
            radBox = JavaNew["java.awt.Checkbox", "radians", cbGroup, False];
            okButton = JavaNew["java.awt.Button", "OK"];
            cancelButton = JavaNew["java.awt.Button", "Cancel"];

            frm@setLayout[Null];
            frm@add[label];
            frm@add[inputField];
            frm@add[degBox];
            frm@add[radBox];
            frm@add[okButton];
            frm@add[cancelButton];

            frm@setBounds[200, 200, 200, 160];
            label@setBounds[20, 30, 150, 20];
            inputField@setBounds[20, 70, 60, 28];
            degBox@setBounds[100, 60, 80, 20];
            radBox@setBounds[100, 80, 80, 20];
            okButton@setBounds[40, 120, 50, 20];
            cancelButton@setBounds[100, 120, 50, 20];
            frm@setResizable[False];

            okButton@addActionListener[
                JavaNew["com.wolfram.jlink.MathActionListener",
                        "(EndModal[]; True)&"]
            ];
            cancelButton@addActionListener[
                JavaNew["com.wolfram.jlink.MathActionListener",
                        "(EndModal[]; False)&"]
            ];

            (* Now make the window visible and bring it to the foreground. *)
            JavaShow[frm];

            frm@setModal[];
            wasOKButton = DoModal[];
            (* Even though the window may have been closed, it is perfectly
               OK to extract values from the controls in the window.
            *)
```

```
            If[TrueQ[wasOKButton],
                angle = ToExpression[inputField@getText[]];
                If[angle =!= Null && degBox@getState[], angle *= Pi/180],
            (* else *)
                (* We will get here if the Cancel button was clicked
                   (wasOKButton will be False), or the dialog was closed
                   by clicking in its close box (wasOKButton will be Null).
                *)
                angle = $Failed
            ];
            (* If the cancel or OK buttons were clicked, frm is still
               visible, so we dispose it here.
            *)
            frm@dispose[];
            angle
        ]
    ]
```



フレームオブジェクト(frm)その他，ラベル，テキスト入力，ラジオボタン等を用意
すべて組み合わせる
配置を決める
アクション関数を決定 okButton@addActionListener, cancelButton@addActionListener.
setModalメソッド カーネルとの接続を維持
DoModal Java側が`EvaluatePacket`[`EndModal`[*args*]]の形で値を返すまで待つ.

```
GetAngle[]
```

$$\frac{7\pi}{9}$$

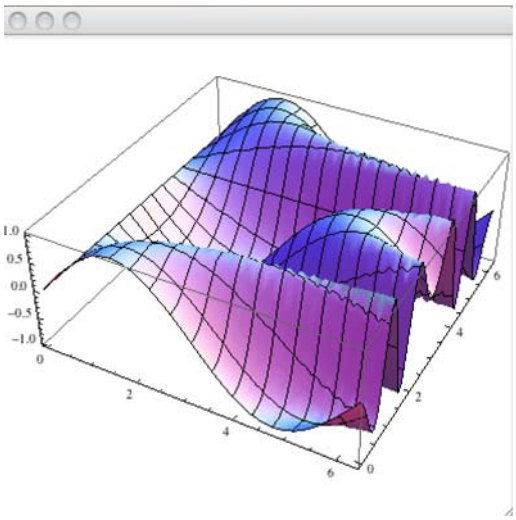
■ グラフィックス

```
frame = JavaNew["com.wolfram.jlink.MathFrame"];
frame@setLayout[JavaNew["java.awt.BorderLayout"]];
mathCanvas = JavaNew["com.wolfram.jlink.MathCanvas"];
frame@add["Center", mathCanvas];
frame@setSize[400, 400];
frame@layout[];
mathCanvas@setMathCommand["Plot[x, {x,0,1}]"];
JavaShow[frame];
```

フレームを用意．その上に描画用のキャンバス作成．絵はキャンバス上に描く.

```
mathCanvas@setMathCommand["Plot3D[Sin[x Cos[y]], {x,0,2Pi}, {y,0,2Pi}]"];
```

同じキャンバス上に違う絵を描画.




```
(* スローアニメーション *)

n = 1.0;
mathCanvas@setMathCommand["Plot3D[Sin[n x Cos[y]], {x,0,2Pi}, {y,0,2Pi}]"];
Do[n += 0.1; mathCanvas@recompute[]; Pause[1], {10}]
```

■ オブジェクトとインヘリタンス(inheritance 継承)

```
urlClass = LoadJavaClass["java.net.URL"]
```

```
JavaClass[java.net.URL, <>]
```

```
urlObject = JavaNew["java.net.URL", "http://www.wolfram.com"]
```

```
« JavaObject[java.net.URL] »
```

```
Methods[urlClass]
```

```
boolean equals(Object)
String getAuthority()
Class getClass()
Object getContent(Class[]) throws java.io.IOException
Object getContent() throws java.io.IOException
int getDefaultPort()
String getFile()
String getHost()
String getPath()
int getPort()
String getProtocol()
String getQuery()
String getRef()
String getUserInfo()
int hashCode()
void notify()
void notifyAll()
java.net.URLConnection openConnection(java.net.Proxy) throws java.io.IOException
java.net.URLConnection openConnection() throws java.io.IOException
java.io.InputStream openStream() throws java.io.IOException
boolean sameFile(java.net.URL)
static void setURLStreamHandlerFactory(java.net.URLStreamHandlerFactory)
String toExternalForm()
String toString()
java.net.URI toURI() throws java.net.URISyntaxException
void wait(long, int) throws InterruptedException
void wait(long) throws InterruptedException
void wait() throws InterruptedException
```

■ *J/Link* Class Loader

```
LoadJavaClass["java.lang.Class"];
cls = Class`forName["some.class.that.only.JLink.can.find"]
```

Java::excptn :

A Java exception occurred: java.lang.ClassNotFoundException: some.class.that.only.JLink.can.find

at java.net.URLClassLoader$1.run(URLClassLoader.java:200)

at java.security.AccessController.doPrivileged(Native Method)

at java.net.URLClassLoader.findClass(URLClassLoader.java:188)

at java.lang.ClassLoader.loadClass(ClassLoader.java:319)

at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:330)

at java.lang.ClassLoader.loadClass(ClassLoader.java:254)

at java.lang.ClassLoader.loadClassInternal(ClassLoader.java:399)

at java.lang.Class.forName0(Native Method)

at java.lang.Class.forName(Class.java:169).

```
$Failed
```

```
LoadJavaClass["com.wolfram.jlink.JLinkClassLoader"];
cls = Class`forName["some.class.that.only.JLink.can.find",
  True, JLinkClassLoader`getInstance[]]
```

```
$Failed
```

***Mathematica*Java 型の対応**

Java type	Mathematica type
byte , char , short , int , long	Integer
Byte , Character , Short , Integer , Long , BigInteger	Integer
float , double	Real
Float , Double , BigDecimal	Real
boolean	True or False
String	String
array	List
controlled by user (see "Complex Numbers")	Complex
Object	JavaObject
Expr	any expression
null	Null

from *Mathematica* to Java own class

javac MyTimerClass.java

```
Needs["JLink`"]

AddToClassPath[Directory[]];

InstallJava[]

LinkObject[
 '/Applications/Mathematica.app/SystemFiles/Links/JLink/JLink.app/Contents/MacOS/
   JavaApplicationStub' -init "/tmp/m00000113931", 4, 4]

FilePrint["MyTimerClass.java"]
```

```
import java.sql.Date;
import java.util.Timer;
import java.util.TimerTask;
import java.awt.Toolkit;
public class MyTimerClass {
   public static void myTimer (int numberOfMillisecondsInTheFuture)
throws Exception {
     Date timeToRun =
      new Date (System.currentTimeMillis() +
numberOfMillisecondsInTheFuture);
     Timer timer = new Timer(true);
     timer.schedule (new TimerTask() {
         public void run() {
           Toolkit.getDefaultToolkit().beep();
           }
         }, timeToRun);
         System.out.println("????????");
     }
   }
```

```
LoadJavaClass["MyTimerClass"]

JavaClass[MyTimerClass, <>]

MyTimerClass`myTimer[2000]
```