FAQ

F-BASIC に関してよくご質問いただく内容をまとめました。ご活用ください。

購入前／レベルアップ／アップデート

使用前

作成時（操作）

配布時

プログラミング（初級）

プログラミング（中級）

プログラミング（上級）

その他

FAQ

＜購入前／レベルアップ／アップデート＞

＜製品情報＞

Question

FB00001
対応しているパソコンは？

Answer

DOS/V 互換機、PC-9801 シリーズで動作します。

Question

FB00002
対応している OS は？

Answer

バージョンによって異なります。下記の表の通りとなります。

＼ OS バージョン ＼	Windows(R) 95	Windwos(R) 98	Windows(R)NT4.0 (R)	Windows(R)2000 Windows(R)Me	Windows(R) XP
F-BASIC V6.3	○	○	○	△ 修正モジュール提供	×
F-BASIC V6.0	○	○	○	×	×
F-BASIC 97 V5.0	○	×	○	×	×
F-BASIC V4.1	○	×	×	×	×

Question

FB00003
インタプリタはありますか？

Answer

ありません。

Question

FB00004
インタプリタがないのにデバッグができますか？

Answer

高機能なデバッガを搭載しているので、プログラムを 1 行ずつ実行したり、プログラム実行中の変数の内容を確認したりすることができます。

FAQ

Question
FB00005
最新版のバージョンは？

Answer
現在の最新版の F-BASIC は、「F-BASIC V6.3 L10」となります（2003 年 2 月 1 日現在）。

Question
FB00006
F-BASIC V6.0 を使用しているのですが、F-BASIC V6.3 にバージョンアップをしても以前のプログラムを使用できますか？

Answer
F-BASIC V6.3 は、F-BASIC V4.1～F-BASIC V6.0 に上位互換があるので、前のバージョンで作成されたプログラムソースをそのまま使用できます。
F-BASIC をバージョンアップ／アップデートした場合には、[翻訳]メニューの[すべて翻訳]コマンドを選択して実行ファイルを作成し直す必要があります。

Question
FB00007
レベルアップサービスについて

Answer
F-BASIC V4.1～F-BASIC V6.0 をお使いの方には、F-BASIC V6.3 へのレベルアップサービスが提供されています。

Question
FB00008
Windows システムによる制限について

Answer
F-BASIC で作成した実行形式ファイルは Windows 上で動作します。このため作成される実行ファイルはファイルサイズの定量制限など Windows システムの制限を受けます。

FAQ

＜アップデート情報＞

Question

FB01001
各バージョンの最新アップデート情報

Answer

F-BASIC のそれぞれのバージョンについて、下記の通り、障害修正用のアップデートパックが提供されています。FM WORLD（http://www.fmworld.net/）よりアップデート方法などを確認した後、アップデートパックをダウンロードしてください。ダウンロードされたファイルには障害修正一覧も添付されています。

※現在お使いのバージョンは、[ヘルプ]メニューの[バージョン情報]コマンドを選択して表示されたダイアログボックスにて確認することができます。

製品	最新版	説明
F-BASIC V6.3	L10 → L10 U003	Windows(R)2000／Windows(R)Me に対応しました。
F-BASIC V6.0	L10 → L10U002	
F-BASIC97 V5.0	L10A → L10A U001	
F-BASIC V4.1	L10 → L10A U004	

FAQ

＜使用前＞

＜インストール＞

Question

FB10001
インストール先のフォルダを指定する際に「ディレクトリの指定に誤りがあります」というメッセージボックスが表示されてしまいます。

Answer

インストール先のフォルダとして、空白を含むフォルダ名は指定できません。
また、インストール先のフォルダ名に、次の文字を含めることはできません。
?、"、/、<、>、*、|、+、,、;、=、[、]、!、#、$、%、&、'、(、)、@、、^、`、{、}、~、空白

Question

FB10002
インストーラ（SETUP.EXE）を起動したときに「レジストリにアクセスできません」というメッセージボックスが表示されてしまいます。

Answer

OS のユーザアカウントの設定によりレジストリへのアクセスが制限されていると、このメッセージが表示されます。この場合にはインストールが正しく行われませんので、ユーザアカウントの設定を変更してから再度インストールを行ってください。ユーザアカウントの設定についてはコンピュータの管理者に問い合わせてください。

＜作成時（操作）＞

＜プログラムの入力＞

Question

FB20001
プログラム編集中に改行したりカーソルキーで行を移動させると、[02]「文法が正しくありません」等のメッセージボックスが表示されてしまいます。

Answer

F-BASIC では、プログラムの入力ミスを見つけやすくするために、プログラムが 1 行入力されるたびに文法のチェックを行っています。
最後に入力した行のどこかに誤りがあるので、メッセージの内容を参考にして誤りを訂正してください。
エラーメッセージを表示したくない場合には、[表示]メニューの[オプション]コマンドを選択し、「エラーメッセージを表示する」のチェックマークを外してください。

Question

FB20002
プログラム編集中に入力した文字が勝手に大文字に（または小文字に）変換されてしまいます。

Answer

F-BASIC では、プログラムの入力ミスを見つけやすくするために、プログラムが 1 行入力されるたびに命令／関数を小文字に、変数名／ラベル等を大文字に変換しています。
大文字／小文字の変換を行いたくない場合には、[表示]メニューの[オプション]コマンドを選択し、「行解析を行う」のチェックマークを外してください。
※拡張命令は大文字に変換されます。

Question

FB20003
プログラム編集中、漢字を入力した時にカーソルがずれて表示されたり、文字化けすることがあります。

Answer

コンピュータにインストールされている RICHED32.DLL が古いものに置き換わっているとこの現象が発生する場合があります。RICHED32.DLL は Windows のコンポーネントなので、Windows の再インストール、もしくは Windows と Internet Explorer をアップデートしていただくことにより正常な RICHED32.DLL に置き換えることができます。

FAQ

＜プログラムのデバッグ＞

Question

FB21001
実行中のプログラムの変数の値を確認する方法は？

Answer

プログラムをデバッグモードで実行しているとき、プログラムファイル中の変数を選択してウォッチボタン（  ）を押下すると、変数の内容が表示されます。

＜プログラムの翻訳＞

Question

FB22001
フロッピーディスク上で翻訳できますか？

Answer

プログラムの翻訳は非常に多くのディスクスペースを必要とするため、フロッピーディスク上での翻訳は翻訳時エラーの原因となる可能性があります。お薦めできません。
ハードディスクにプログラムソースを移動して、ハードディスク上で翻訳してください。

Question

FB22002
ネットワークドライブ経由で翻訳できますか？

Answer

翻訳できます。ただし、ネットワークドライブ上でプログラムの翻訳を行う場合には、あらかじめ、OS で使用するネットワークドライブにドライブ名を割り当てておく必要があります。

＜移行支援機能＞

Question

FB23001
行番号は必要ですか？

Answer

行番号は必要ありません。N88-BASIC からのプログラム移行の場合のように、すでに行番号がついているプログラムに関しては、ラベルとして扱われます。従って、GOTO 文などの分岐先の指定に行番号を使っている場合はそのまま翻訳できます。

Question

FB23002
RENUM 命令はありますか？

Answer

ありません。F-BASIC では行番号を必要としません。

Question

FB23003
N88-BASIC のプログラムが保存されているディスクが認識されません。

Answer

ディスクが MS-DOS 以外のフォーマットになっている可能性があります。
N88(DISK)BASIC のプログラムファイルは MS-DOS フォーマットに変換する必要があります。NEC 製日本語 MS-DOS V6.2 の FILECONV.EXE を使用してください。

Question

FB23004
プログラムファイルを読みこんだときに、プログラムが表示されずに、でたらめな記号や文字列が表示されてしまいます。

Answer

プログラムがアスキーセーブされていない可能性があります。元の BASIC で開いてアスキーセーブしなおしてください。
また、プログラムファイル以外のファイルを開こうとしている可能性もあります。正しいファイルを指定しているか確認してください。たとえば、実行ファイル(*.EXE)などを開くことはできません。

Question

FB23005
アスキー形式での保存方法は？

Answer

アスキー形式で保存するには、そのプログラムを編集していた環境で次の手順に従って行ってください。
1）N-88BASIC を起動する。
2）LOAD "ファイル名"でプログラムファイルを読み込む。
3）SAVE "ファイル名",A で保存しなおす。

FAQ

Question

FB23006
数値データファイルを利用する方法は？

Answer

MS-DOS BASIC での数値データファイルへのデータ格納方法と、F-BASIC での数値データファイルへのデータ格納方法には違いがあります。
従って、次のいずれかの方法で移行作業を進めてください。

＜数値データ変換ツールを使用する＞
F-BASIC にはデータファイルを変換するためのコンバータ（数値変換ツール CNVF21.EXE）が添付されています。あらかじめデータを F-BASIC の形式に変換し、F-BASIC の形式でデータを読み取り、F-BASIC の形式でデータを書き込めるようにします。従来のプログラムからは新しいデータを参照することができなくなりますが、処理が高速になります。

＜数値変換拡張ライブラリを利用する＞
F-BASIC には従来の形式のデータを扱うための拡張ライブラリが添付されています。従来の形式でデータを読み取り、従来の形式でデータを書き込むようにプログラムを修正する必要がありますが、従来のプログラムとデータを共有することができます。元の BASIC に応じて処理を行います。

①N88BASIC で作成されたランダムファイルを読み書きする方法

N88-BASIC と F-BASIC V6.3 では、ランダムファイルへのデータの格納形式が異なるため、そのままでは N88-BASIC で作成されたランダムファイルのデータを利用できません。

＜N88-BASIC と F-BASIC V6.3 のデータの互換性＞

データ型	互換性
整数型データ	×
単精度実数型データ	×
倍精度実数型データ	×
文字列型データ	○

その代わりに、F-BASIC V6.3 には N88-BASIC で作成されたランダムファイルのデータを扱うための命令が用意されていますので、N88-BASIC で作成されたランダムファイルを利用する場合にはそれらの命令を使用するようにプログラムを書き直してください。

＜N88-BASIC で作成されたランダムファイルのデータを扱うための命令＞

データ型	データ操作命令		
	N88-BASIC		F-BASIC V6.3
整数型データ	CVI MKI$	→ →	CVI98 MKI98$
単精度実数型データ	CVS MKS$	→ →	CVS98 MKS98$
倍精度実数型データ	CVD MKD$	→ →	CVD98 MKD98$
文字列型データ	修正不要		

※N88-BASIC で作成されたランダムファイルのデータを扱うための命令は拡張命令であるため、利用の際にはプログラムの先頭に

```
#include "F1A0LWIC.BI"
```

を記述します。

＜N88-BASIC で作成されたランダムファイルのデータを扱うプログラムの例＞

N88-BASIC のプログラム

```
DD1%=CVI(ZX1$)
DD2#=CVD(ZX2$)
```

↓

F-BASIC V6.3 のプログラム

```
#include "F1A0LWIC.BI"
DD1%=CVI98(ZX1$)
DD2#=CVD98(ZX2$)
```

次に、N88-BASIC で作成されたランダムファイルの操作は、「④MS-DOS で作成されたランダムファイルを操作するプログラムを作成する手順」で紹介します。

②QuickBASIC／MSBASIC で作成されたランダムファイルを読み書きする方法

QuickBASIC／MSBASIC(以後 QuickBASIC)と F-BASIC V6.3 では、ランダムファイルへのデータの格納形式が異なるため、そのままでは QUICKBASIC で作成されたランダムファイルのデータを利用できません。

＜QUICKBASIC と F-BASIC V6.3 のデータの互換性＞

データ型	互換性
整数型データ	×
長整数型データ	×
単精度実数型データ	×
倍精度実数型データ	×
文字列型データ	○

その代わりに、F-BASIC V6.3 には QUICKBASIC で作成されたランダムファイルのデータを扱うための命令が用意されていますので、QUICKBASIC で作成されたランダムファイルを利用する場合にはそれらの命令を使用するようにプログラムを書き直してください。

＜QUICKBASIC で作成されたランダムファイルのデータを扱うための命令＞

データ型	データ操作命令		
	QUICKBASIC		F-BASIC V6.3
整数型データ	CVI MKI$	→ →	CVIV MKIV$
長整数型データ	CVL MKL$	→ →	CVLV MKLV$
単精度実数型データ	CVS MKS$	→ →	CVSV MKSV$
倍精度実数型データ	CVD MKD$	→ →	CVDV MKDV$
文字列型データ	修正不要		

※QUICKBASIC で作成されたランダムファイルのデータを扱うための命令は拡張命令であるため、利用の際にはプログラムの先頭に

```
#include "F1A0LWIC.BI"
```

を記述します。

＜QUICKBASIC で作成されたランダムファイルのデータを扱うプログラムの例＞

QUICKBASIC のプログラム
DD1%=CVI(ZX1$) DD2#=CVD(ZX2$)

↓

F-BASIC V6.3 のプログラム
#include "F1A0LWIC.BI" DD1%=CVIV(ZX1$) DD2#=CVDV(ZX2$)

次に、QUICKBASIC で作成されたランダムファイルの操作は、「④MS-DOS で作成されたランダムファイルを操作するプログラムを作成する手順」で紹介します。

③F-BASIC86HG で作成されたランダムファイルを読み書きする方法

F-BASIC86HG と F-BASIC V6.3 では、ランダムファイルへのデータの格納形式が異なるため、そのままでは F-BASIC86HG で作成されたランダムファイルのデータを利用できません。

＜F-BASIC86HG と F-BASIC V6.3 のデータの互換性＞

データ型	互換性
整数型データ	○
単精度実数型データ	×
倍精度実数型データ	×
文字列型データ	○

その代わりに、F-BASIC V6.3 には F-BASIC86HG で作成されたランダムファイルのデータを扱うための命令が用意されていますので、F-BASIC86HG で作成されたランダムファイルを利用する場合にはそれらの命令を使用するようにプログラムを書き直してください。

＜F-BASIC86HG で作成されたランダムファイルのデータを扱うための命令＞

データ型	データ操作命令		
	F-BASIC86HG		F-BASIC V6.3
整数型データ	修正不要		
単精度実数型データ	CVS MKS$	→ →	CVS2 MKS2$
倍精度実数型データ	CVD MKD$	→ →	CVD2 MKD2$
文字列型データ	修正不要		

※F-BASIC86HG で作成されたランダムファイルのデータを扱うための命令は拡張命令であるため、利用の際にはプログラムの先頭に

```
#include "F1A0LWIC.BI"
```

を記述します。

＜F-BASIC86HG で作成されたランダムファイルのデータを扱うプログラムの例＞

F-BASIC86HG のプログラム

```
DD1%=CVI(ZX1$)
DD2#=CVD(ZX2$)
```

↓

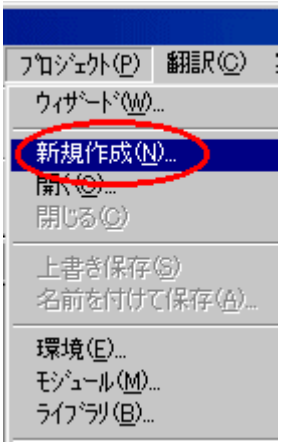
F-BASIC V6.3 のプログラム

```
#include "F1A0LWIC.BI"
DD1%=CVI2(ZX1$)
DD2#=CVD2(ZX2$)
```

次に、F-BASIC86HG で作成されたランダムファイルの操作は、「④MS-DOS で作成されたランダムファイルを操作するプログラムを作成する手順」で紹介します。

④MS-DOS で作成されたランダムファイルを操作するプログラムを作成する手順

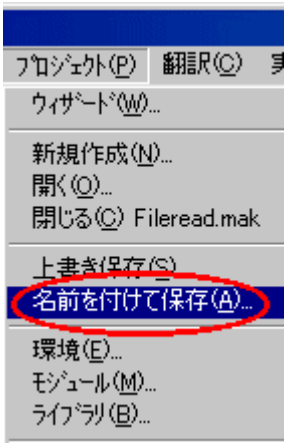
STEP 1 メニューから［プロジェクト(P)]-[新規作成(N)...]を選択して、プロジェクトを作成します。



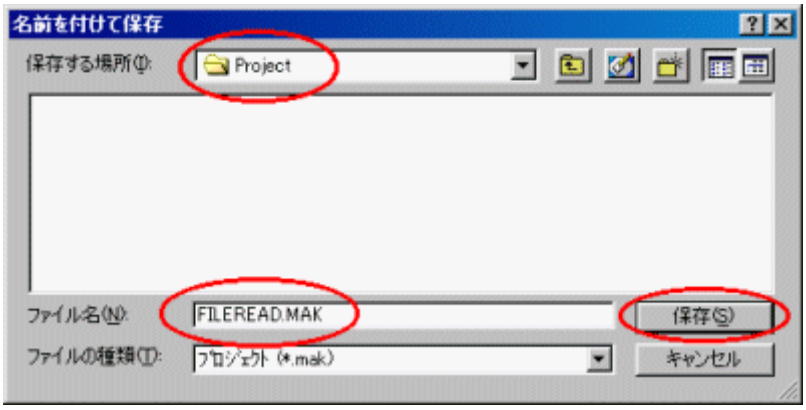
1-1 プロジェクトファイル名の入力を要求されるので、ここでは、"プログラム名.MAK"を入力して[OK]ボタンを押します。(例: FILEREAD.MAK)

1-2 空のプロジェクトファイルが表示されます。

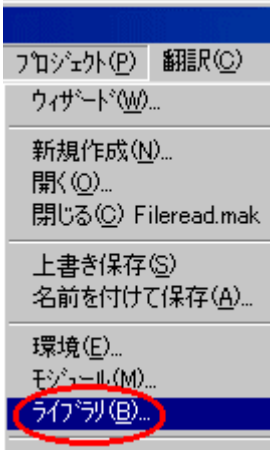
STEP 2 メニューから[プロジェクト(P)]-[名前を付けて保存(A)...]を選択して、プロジェクトを保存します。



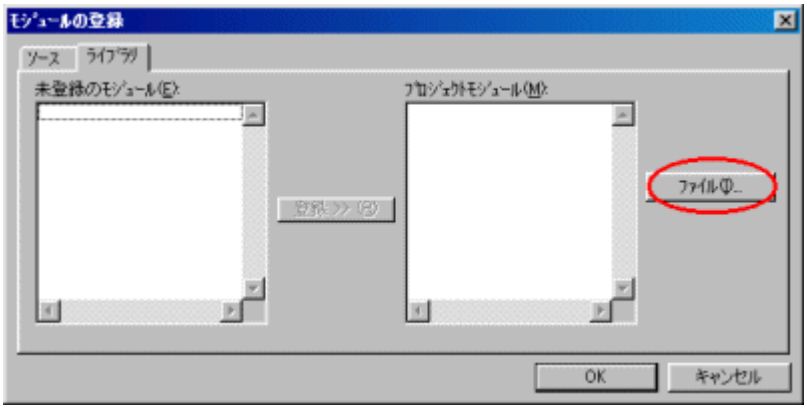
2-1 プログラムを保存するフォルダとプロジェクトファイル名を確認してから[OK]ボタンを押します。



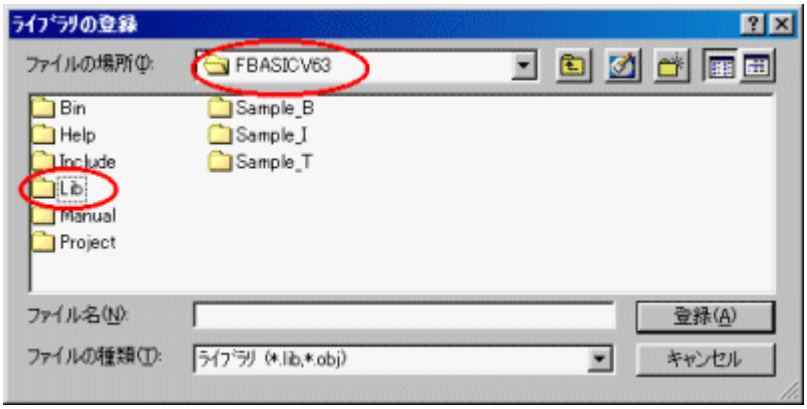
STEP 3 メニューから[プロジェクト(P)]-[ライブラリ(A)...]を選択して、プロジェクトに数値変換ライブラリを登録します。



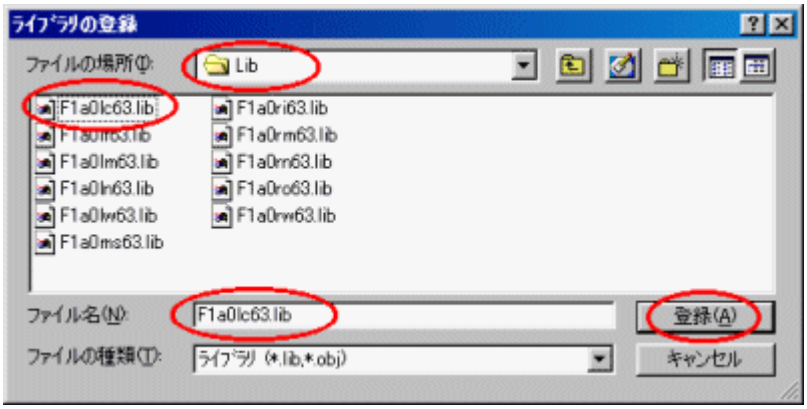
3-1 [ファイル]ボタンを押します。



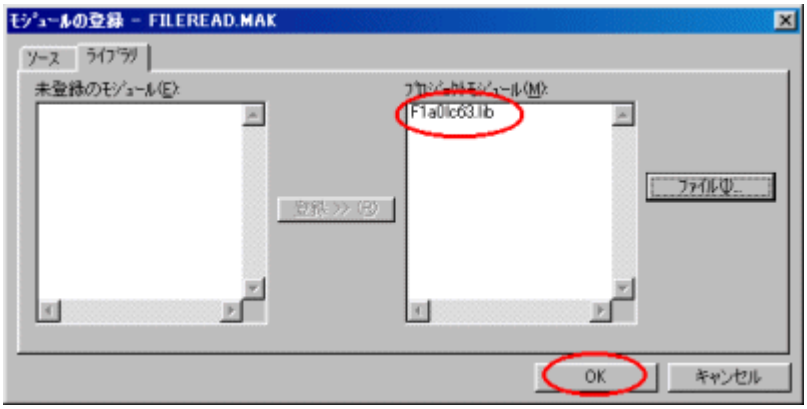
3-2 F-BASIC V6.3 の LIB フォルダを選択します。



3-3 "F1A0LC63.LIB"を選択して、[登録]ボタンを押します。

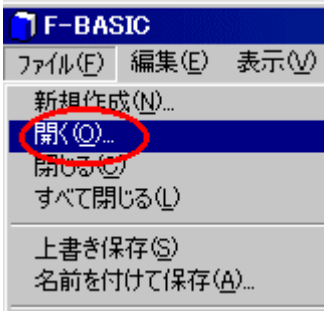


3-4 [プロジェクトモジュール]欄に"F1A0LC63.LIB"が表示されたことを確認して、[OK]ボタンを押します。

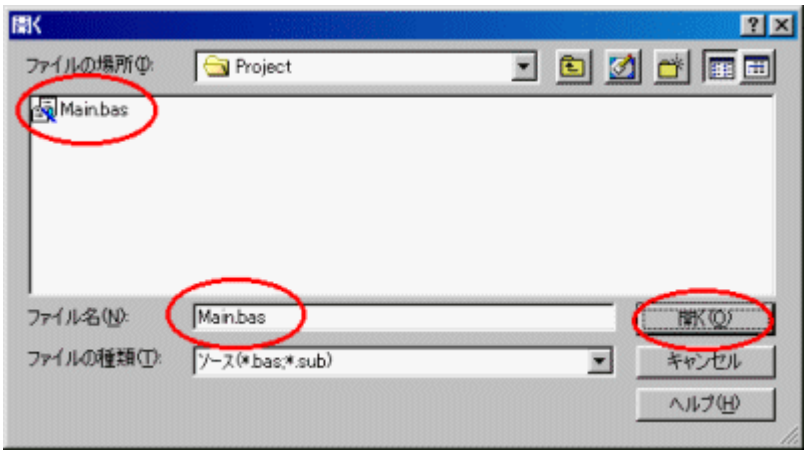


3-5 MAK ファイルの表示に"F1A0LC63.LIB"が追加されます。

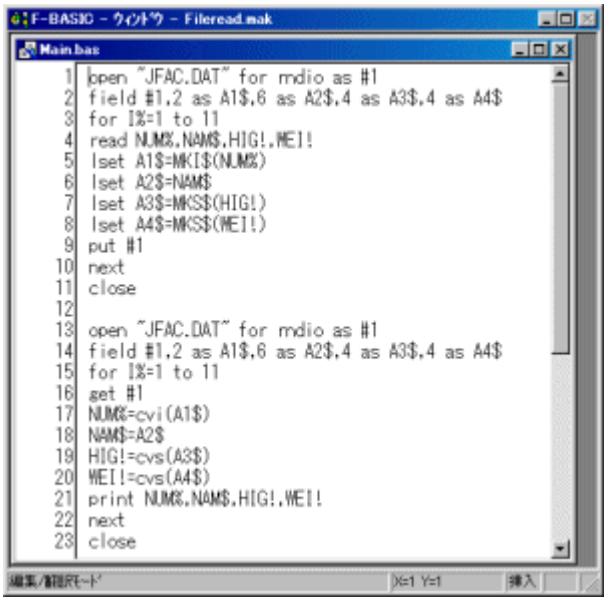
STEP 4 メニューから[ファイル]-[開く(O)...]を選択して、プログラムファイルを開きます。



4-1 ファイルを選択して[OK]ボタンを押します。(例: MAIN.BAS)



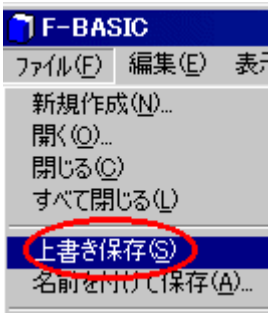
4-2 プログラムファイルが表示されます。



4-3 プログラムを入力します。
プログラムの先頭に #include "F1A0LWIC.BI" を記述します。このとき、CVI、CVS、CVL、CVD、MKI$、MKS$、MKL$、MKD$のかわりに、それぞれの元の BASIC にあった拡張命令を使用します。

元の BASIC	CVI	CVL	CVS	CVD	MKI$	MKL$	MKS$	MKD$
N88-BASIC	CVI98	-	CVS98	CVD98	MKI98$	-	MKS98$	MKD98$
QUICK BASIC	CVIV	CVLV	CVSV	CVDV	MKIV$	MKLV$	MKSV$	MKDV$
F-BASIC86 HG	修正不要	CVL2	CVS2	CVD2	修正不要	MKL2$	MKS2$	MKD2$

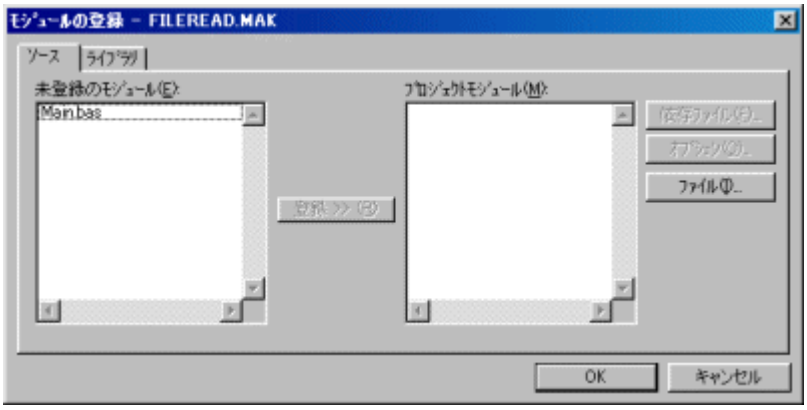
STEP 5 メニューから[ファイル(F)]-[上書き保存(S)...]を選択して、プログラムファイルを保存します。



STEP 6 メニューから[プロジェクト(P)]-[モジュール(M)...]を選択して、プログラムファイルを登録します。

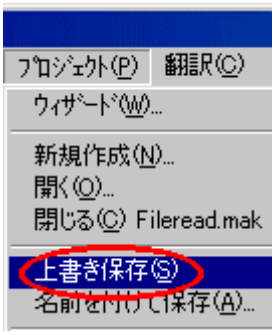
6-1 [未登録のモジュール]欄からプログラムファイルを選択して、[登録>>(R)]ボタンを押します。

6-2 プログラムファイルが[プロジェクトモジュール]欄に移動したのを確認して、[OK]ボタンを押します。



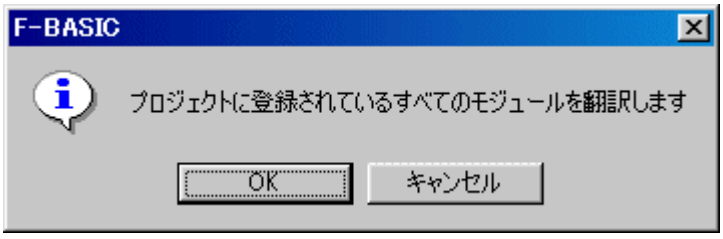
6-3 MAK ファイルの表示にプログラムファイルが追加されます。

STEP 7 メニューから[プロジェクト(P)]-[上書き保存(S)...]を選択して、プロジェクトを保存しなおします。

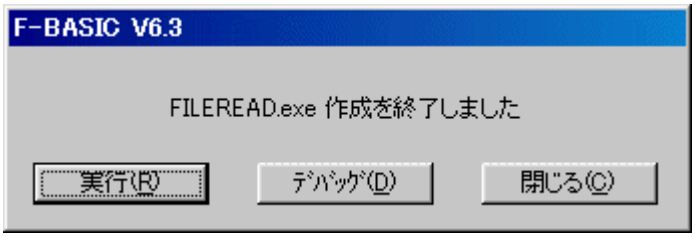


STEP 8 メニューから[翻訳(C)]-[すべて翻訳(A)]を選択して、プログラムを翻訳します。

8-1 [OK]ボタンを押します。



8-2 ダイアログが表示されて翻訳が完了します。



※F-BASIC V6.3 以外をお使いの場合には数値変換ライブラリのファイル名が異なります。ご確認のうえ、該当するファイルを登録して使用してください。

F-BASIC バージョン	数値変換ライブラリ
F-BASIC V6.3	F1A0LC63.LIB
F-BASIC V6.0	F1A0LC60.LIB
F-BASIC97 V5.0	F1A0LWIC.LIB
F-BASIC V4.1	F1A0LWIC.LIB

※既存のプログラムを流用する場合には、メニューの[編集(E)]-[置換(R)...]を使用しても命令の置換が行なえます。
※それぞれのメニューの詳しい働きを知りたい場合には、プログラミングガイドを参照してください。

Question

FB23007
INP 関数、OUT 命令は使えますか？

Answer

Windows の制限により使えません。

Question

FB23008
プログラムソースファイルがないのですが、実行ファイルだけでも移行は可能ですか？

Answer

プログラムソースファイルがないとプログラムを移行できません。

Question

FB23009
移行支援機能を使用すると行番号が削除されます。行番号が削除されても正常に動作しますか？

Answer

移行支援機能は、GOTO 命令などで参照されていない行から不要な行番号を削除しています。行番号を削除したくない場合には、「参照されていない行番号を削除する」のチェックマークを外してから移行支援機能を使用してください。

Question

FB23010
移行支援機能ダイアログボックスで「参照されていない行番号を削除する」をチェックして実行したのに行番号が削除されません。

Answer

変換元のプログラム中に文法エラー（変換後のプログラムには「CVx05@文法エラー'～～」のメッセージを挿入）が含まれていると、プログラム中に含まれている数値が行番号であるかどうかを正しく判断できないため、行番号の削除を中止します。

例)
GOTOP 100
　100 が行番号であるかどうかわからない。

＜配布時＞

＜プログラムの配布方法＞

Question

FB30001
作成したプログラムを配布する方法は？

Answer

作成した実行ファイル(*.EXE)とランタイムライブラリ(*.DLL)を一緒にコピーして配布してください。配布したランタイムライブラリは、実行ファイルと同じフォルダ内か、実行させたいマシンの SYSTEM フォルダ内に置いてください。

Question

FB30002
ランタイムライブラリはどこで入手できますか？

Answer

OS によって異なりますが、SYSTEM のフォルダの中にあります。
Windows95、Windows98、WindowsMe → C:¥WINDOWS¥SYSTEM
WindowsNT4.0、Windows2000 → C:¥WINNT¥SYSTEM32
※フォルダ表記 C:¥Windows¥system の C はドライブ名です。機種や OS のインストール先によって異なる場合があります。

Question

FB30003
ランタイムライブラリの配布は自由ですか？

Answer

ランタイムライブラリの配布は自由です。
詳細は、製品に添付されている「F-BASIC の使用に関する特約条項」に掲載されています。

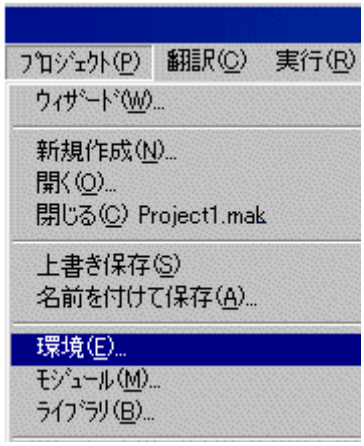
Question

FB30004
ランタイムライブラリなしでも実行できますか？

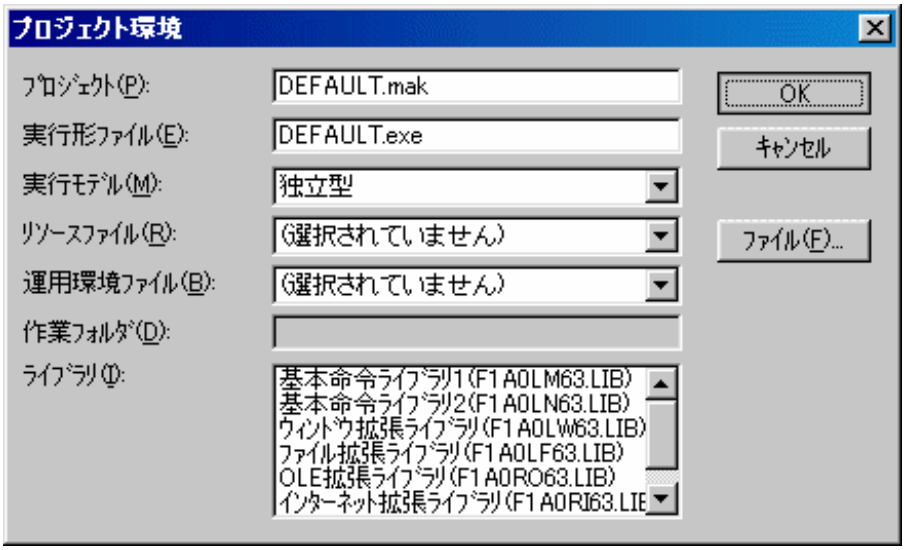
Answer

「独立型」という実行モデルを指定して翻訳することで、ランタイムライブラリなしでも実行可能なファイルが作成できます。

次の手順で作成します。
（1）　F-BASIC 上でプログラムソースファイルを開きます。プロジェクトにプログラムソースが登録されている場合は、プロジェクトファイルを開きます。
（2）　[プロジェクト]メニューの[環境]コマンドを選択します。



（3）　表示された[プロジェクト環境]ダイアログボックス上の「実行モデル」を「独立型」に設定し、OK ボタンを押下します。



（4）　翻訳をします。

上記の手順で作成した実行ファイルはランタイムライブラリを必要としません。

Question

FB30005
プログラム起動時に「プログラム開始エラー ***.exe - DLL がみつかりません。ダイナミックリンクライブラリ F1A0***.DLL が指定されたパス ***に見つかりません」というメッセージボックスが表示されてしまいます。

Answer

実行しようとしているマシンの SYSTEM フォルダに必要なランタイムライブラリがコピーされていません。「ランタイム型」で作成した実行プログラムは、実行するマシンにもランタイムライブラリが必要です。実行プログラムと同じフォルダか、実行するマシンの SYSTEM フォルダに次のファイルをコピーしてください。プログラムを配布する際にはこれらのファイルを一緒に配布してください。

バージョン	ランタイムファイル名	機能
F-BASIC V6.3	F1A0RN63.DLL	基本命令用
	F1A0RW63.DLL	拡張ライブラリ用
	F1A0RO63.DLL	OLE クライアント用
	F1A0RI63.DLL	インターネット機能用
F-BASIC V6.0	F1A0RN60.DLL	基本命令用
	F1A0RW60.DLL	拡張ライブラリ用
	F1A0RO60.DLL	OLE クライアント用
F-BASIC97 V5.0	F1A0RN50.DLL	基本命令用
	F1A0RW50.DLL	拡張ライブラリ用
	F1A0RO50.DLL	OLE クライアント用
F-BASIC V4.1	F1A0RWN4.DLL	基本命令用
	F1A0RWW4.DLL	拡張ライブラリ用
	F1A0RWO4.DLL	OLE クライアント用

Question

FB30006
DOS/V 機で作成したプログラムを PC-9801 シリーズで実行することはできますか？

Answer

使えます。PC-9801 シリーズで作ったプログラムを DOS/V 機で使うこともできます。

＜プログラミング（初級）＞

＜OLE＞

Question

FB40001

OLE 機能を使用すると、翻訳時エラー「LINK : fatal error LNK1181:入力ファイル "F1A0RO**.LIB を開けません」が発生してしまいます。

Answer

プログラムが独立型で作成されている可能性があります。OLE 拡張機能を利用する場合は、プログラムの実行モデルをランタイム型に設定して翻訳する必要があります。

Question

FB40002

OLE 機能を使用すると、実行時エラー「***.exe - DLL が見つかりません ダイナミックリンクライブラリ F1A0RO**.DLL が指定されたパス ***に見つかりません」が発生してしまいます。

Answer

実行しようとしているマシンに、作成したプログラムの実行に必要なランタイムライブラリがインストールされていない可能性があります。ランタイムライブラリについては「プログラムの配布」を参照してください。
また、プログラムが独立実行型で作成されている可能性があります。OLE 機能を利用する場合には、プログラムの実行モデルをランタイム型に設定して翻訳する必要があります。

＜インターネット＞

Question

FB41001

インターネット機能を使用すると、翻訳時エラー「LINK : fatal error LNK1181:入力ファイル "F1A0RI63.LIB を開けません」が発生してしまいます。

Answer

プログラムが独立型で作成されている可能性があります。インターネット拡張機能を利用する場合は、プログラムの実行モデルをランタイム型に設定して翻訳する必要があります。

Question

FB41002
インターネット機能を使用すると、実行時エラー「***.exe - DLL が見つかりません ダイナミックリンクライブラリ F1A0RI63.DLL が指定されたパス ***に見つかりません」が発生してしまいます。

Answer

実行しようとしているマシンに、作成したプログラムの実行に必要なランタイムライブラリがインストールされていない可能性があります。ランタイムライブラリについては「プログラムの配布」を参照してください。
また、プログラムが実行型で作成されている可能性があります。インターネット機能を利用する場合には、プログラムの実行モデルをランタイム型に設定して翻訳する必要があります。

Question

FB41003
プログラムの起動時に「プロシージャエントリーポイント InternetAutoDialHangUp がダイナミックリンクライブラリ WinInet.DLL から見つかりません」というメッセージボックスが表示されてしまいます。

Answer

インターネット機能やイメージ機能（ビットマップ画像と JPEG 画像とのデータ変換）を利用したプログラムを実行するマシンには、Internet Explorer ver.5.5 以上がインストールされている必要があります。Internet Explorer のブラウザを使わなくても同様に必要となります。

＜ファイル操作＞

Question

FB42001
OPEN 命令で実行時エラー[64]「指定のファイルは既に存在しています」が発生してしまいます。

Answer

ファイルを新規作成する際に、指定されたファイルが既に存在しているとエラーになります。同名のファイルを削除してからファイルをオープンしてください。

＜F-BASIC V6.0 以降の場合＞

F-BASIC V6.0 からは次の機能が追加されていますので、下記のように変更してください。

変更例 1)シーケンシャルファイルの場合=====

F-BASIC V6.0 からは、シーケンシャルファイルの OPEN モードに **FOR CREATE** が追加されています。FOR CREATE モードを使用することで、ファイルが既に存在していた場合は古いファイルを削除して新しいファイルを作成します。

```
open "FILE" for output as #1
                ↓
open "FILE" for create  as #1
```

変更例 2)バイナリファイルの場合=====

ファイルポインタ以降をカットしてファイルサイズを調整する **TRUNCATE 命令**が追加されています。FOR BINIO モードでファイルを開き、ファイルが既に存在する場合は、TRUNCATE 命令でサイズを 0 に調整してからデータを書き込みます。

```
open "FILE" for binout as #1
                ↓
open "FILE" for binio   as #1
truncate      #1
```

＜F-BASIC V4.1/F-BASIC97 V5.0 の場合＞

シーケンシャルファイルを新規作成する際には OPEN FOR OUTPUT 指定を使用しますが、そのファイル名が既に使用されていると実行時エラーになってしまいます。これは次のような方法で回避可能です。

例 1)KILL 命令を実行する。=====

```
open "FILE" for append as #1    'ファイルがなければ作成する。
close #1
kill "FILE"                          'ファイルを削除する。
open "FILE" for output as #1    'ファイルを新規作成する。
```

例 2)エラー処理を行う。=====

```
on error goto *E1
open "FILE" for output as #1    'ファイルを新規作成する。
on error goto *E1
goto *E2
*E1
kill "FILE"                     'ファイルを削除してやり直す。
resume
*E2
```

Question

FB42002
Microsoft Excel とデータを共有したり、Excel にデータを表示させたりすることはできますか？

Answer

CSV 形式のファイルを利用することによって可能です。

```
' Excel にデータを表示するプログラム
' CSV 形式を利用する。

open "data.csv" for create as #1
write #1, "DATA1", "660"
write #1, "DATA2", "670"
write #1, "DATA3", "520"
write #1, "DATA4", "430"
close #1

' EXCEL を実行する。パスは EXCEL のインストール先に合わせる必要がある。
shell "C:¥Program Files¥Microsoft Office¥Office¥Excel.exe", "test.csv"
stop
```

Excel のデータを利用する場合は、Excel でデータを保存する際にファイルの種類は「CSV（カンマ区切り）(*.CSV)」を選択してください。

Question

FB42003
ファイル名一覧の取得方法は？

Answer

F-BASIC には、N88-BASIC にあった FILES 命令のようにファイル名一覧を表示させる命令がありません。しかし、ファイルの拡張命令を利用することによって、ファイルに関する情報を獲得することができます。次のサンプルを参考にしてください。
また、赤で表示されているファイル拡張命令を利用する場合には、プログラムの先頭に「#include "FILE.BI"」を記述する必要があります。命令に関してはマニュアルやオンラインヘルプを参考にしてください。

＜フロッピーディスクのファイルの情報一覧を表示するサンプル＞

```
#include "FILE.BI"
DOPEN "A:*.bas",1
print "ファイル名","サイズ","更新日","更新時刻"
while 1
  GETFIN 1
  if EBUF(1) then exit
  print FINF$(1,0),FINF$(1,1);"byte",FINF$(1,2),FINF$(1,3)
wend
DCLOSE 1
stop
end
```

Question

FB42004
N88-BASIC で作成したランダムファイルを読み込めますか？

Answer

可能です。利用方法は、「FB23006　数値データを利用する方法は？」に掲載されています。

Question

FB42005
N88-BASIC で作成したシーケンシャルファイルを読み込めますか？

Answer

可能です。利用方法は、N88-BASIC と同様の手順です。

```
'ファイルをオープンしてデータを読みこみます。
open "kiroku.dat" for input as #1
input #1,A$,B$,C$       ' A$,B$,C$にデータを読みこみます。
close #1
```

※文字列の最後の文字が空白の場合、F-BASIC では INPUT 命令で読みこんだ場合、変数に格納しません。

Question

FB42006
FIELD の記述を 2 行以上に分けて記述する方法は？

Answer

次のように記述します。

＜FIELD の記述を 2 行以上に分けて記述するサンプル＞

```
open "(70)FILENAME" for rndinp as #1
field #1,10 as A1$,10 as A2$,10 as A3$
field #1,30 as DUMMY1$,10 as A4$,10 as A5$  'DUMMY1$の「30」は 1 行目の合計
field #1,50 as DUMMY2$,10 as A6$,10 as A7$  'DUMMY2$の「50」は 2 行目の合計
```

上記のプログラムで読みこまれるデータは次のように格納されます。

```
1          11          21          31          41          51          61        70
+----------+----------+----------+----------+----------+----------+----------+
|   DATA1  |   DATA2  |   DATA3  |   DATA4  |   DATA5  |   DATA6  |   DATA7  |
+----------+----------+----------+----------+----------+----------+----------+

+----------+----------+----------+----------+----------+----------+----------+
|XXX A1$ XXX|XXX A2$ XXX|XXX A3$ XXX|                                         |
+-----------------------------------+----------------------+                 |
|############ DUMMY1$ ############|XXX A4$ XXX|XXX A5$ XXX|                 |
+-------------------------------------------------------------+----------------------+
|############ DUMMY2$ ######################################|XXX A6$ XXX|XXX A7$ XXX|
+-----------------------------------------------------------------------------------+
```

＜回線(RS-232C)＞

Question

FB43001
RS-232C 経由で通信した文字が化けてしまいます。

Answer

転送速度が送信側と受信側で一致していない可能性があります。BAUD 命令で転送速度を同じにしてください。
また、ファイルディスクリプタの設定も見なおしてください。

Question

FB43002
CHR$(&H11)と CHR$(&H13)が読みこまれません。

Answer

XON/OFF キャラクタフロー制御が「有り」に設定されている可能性があります。オプションの設定（OPEN 命令のファイルディスクリプタにて設定できる通信条）を見なおし、XON/OFF キャラクタフロー制御を「なし」に設定してください。ファイルディスクリプタのパラメタに関してはワンポイントレッスンの「第三回 モデムで電話をかけてみよう」で解説していますので、参考にしてください。

Question

FB43003
ブレーク信号の制御方法は？

Answer

次のようにして、RS-232C 信号線や DTR を制御することができます。参考にしてください。

```
' RS-232C 信号線を制御するプログラム
' FILEATTR 関数でハンドルを獲得し WindowsAPI へ引き渡します。

#define SETRTS   3 ' SET RTS HIGH
#define CLRRTS   4 ' SET RTS LOW
#define SETDTR   5 ' SET DTR HIGH
#define CLRDTR   6 ' SET DTR LOW
#define SETBREAK 8 ' SET THE DEVICE BREAK LINE.
#define CLRBREAK 9 ' CLEAR THE DEVICE BREAK LINE.

declare sub API_ESCAPECOMMFUNCTION lib "kernel32" alias "EscapeCommFunction" ( byval
HFILE&, byval FUNC& )

open "COM0:" for input  as #1
open "COM0:" for output as #2

' ブレーク信号を送出する。
API_ESCAPECOMMFUNCTION fileattr(#1), SETBREAK
wait 100 ' 1 秒
API_ESCAPECOMMFUNCTION fileattr(#1), CLRBREAK

' DTR を制御する。
API_ESCAPECOMMFUNCTION fileattr(#1), SETDTR
wait 100 ' 1 秒
API_ESCAPECOMMFUNCTION fileattr(#1), CLRDTR

close
stop
end
```

＜プログラムの開始／実行／終了＞

Question

FB44001
プログラムの起動時に「プログラム開始エラー ***.exe - DLL が見つかりません。ダイナミックリンクライブラリ F1A0***.DLL が指定されたパス ***に見つかりません」というメッセージボックスが表示されてしまいます。

Answer

実行しようとしているマシンの SYSTEM フォルダに必要なランタイムライブラリがコピーされていません。「ランタイム型」で作成した実行プログラムは、実行するマシンにもランタイムライブラリが必要です。実行プログラムと同じフォルダか、実行するマシンの SYSTEM フォルダに次のファイルをコピーしてください。プログラムを配布する際にはこれらのファイルを一緒に配布してください。詳しくは、「FB30005　ランタイムライブラリのファイル名等」の詳細をご覧ください。
また、実行ファイルが正しく翻訳されていない可能性があります。F-BASIC の[翻訳]メニューから[すべて翻訳]コマンドを選択してプログラムを翻訳しなおしてください。
※F-BASIC をバージョンアップ／アップデートした場合には、[すべて翻訳]コマンドを選択してプログラムを翻訳しなおす必要があります。

Question

FB44002
F-BASIC で作成した実行ファイルが実行後すぐにウィンドウが閉じてしまい結果を確認することができません。

Answer

F-BASIC では、プログラムを内部から終了させることができるようにするために、END 命令を実行すると直ちにすべてのウィンドウを閉じてプログラムを終了するつくりになっています。プログラムを終了する前に表示内容を確認したい場合には、INPUT 命令／STOP 命令などを使ってプログラムを一時停止するようにしてください。

修正前)

```
print "1+2=";1+2
end
```

修正後)

```
print "1+2=";1+2
stop                    '←この命令を追加する。
End
```

Question

FB44003
SHELL 命令／SHELLEXECUTE 命令で、名前が ftp や www で始まるファイルを実行しようとすると、指定したプログラムが実行されずに、代わりにウェブブラウザが起動されてしまいます。

Answer

Windows NT(R)4.0 ／ Windows(R)98 の一部において、SHELL 命令／SHELLEXECUTE 命令を使用して名前が ftp や www で始まるファイルを実行しようとした場合に、Internet Explorer が起動される場合があります。現在原因を調査中です。ファイル名をフルパスで指定することにより回避してください。

修正前）
```
shell "ftp.exe"     '←ウェブブラウザが起動してしまう場合がある。
```

修正後）
```
shell ".¥ftp.exe"
```

メモ）
なお、この動作は OS のバージョンによって異なるため、ftp や http のオープンを目的として SHELL 命令／SHELLEXECUTE 命令を使用する場合には、アドレスの前に"ftp:"、"http:"を付けるようにしてください。

修正前）
```
shell "www.fujitsu.com/"    '←ウェブブラウザが起動しない場合がある。
```

修正後）
```
shell "http://www.fujitsu.com/"
```

Question

FB44004
RTN 関数で 16bit アプリケーションの復帰値を取得できますか？

Answer

Windows95／Windows98 において、SHELL 命令／SHELLEXECUTE 命令を使用して 16bit アプリケーションを実行した場合に、復帰値として常に 0 が返されます。回避方法はありません。

Question

FB44005
SHELL 命令／SHELLEXECUTE 命令で同期実行を指定しているのに、アプリケーションが終了するまでプログラムの実行が止まりません。

Answer

一部のアプリケーションは 2 回目以降の起動時にプロセスの情報を返さないため、非同期実行されます。回避方法はありません。

例)
Microsoft Word で 2 つめの文書を開いたときや、Internet Explorer で 2 つめのページを開いたときには、アプリケーションがプロセスの情報を返さないため非同期実行されます。

Question

FB44006
SHELL 命令で[63]「指定のファイルが見つかりません」というエラーが発生してしまいます。

Answer

SHELL 命令でファイル名とコマンドライン引数を繋げて記述しているとファイルが正しく実行されません。ファイル名とコマンドライン引数は分けて記述するようにしてください。

誤) shell "notepad.exe readme.txt"
正) shell "notepad.exe","readme.txt"

Question

FB44007
SHELL 命令で起動するアプリケーションを表示しないようにできますか？

Answer

SHELL 命令の表示スイッチに 0 を指定すると、新しいアプリケーションを画面に表示せずに実行します。
例) shell "test.exe",,0

※強制的にウィンドウを表示するアプリケーションや、常にウィンドウを表示しないアプリケーションの動作を変えることはできません。
※本機能は F-BASIC V6.3 以降のみの機能です。

Question

FB44008
SHELL 命令で同期実行させる方法は？

Answer

F-BASIC V6.0 以前のバージョンでは同期実行させることはできません。
F-BASIC V6.3 からは、同期実行させることが可能となりました。「同期実行スイッチ」を指定してください。

例） shell "test.exe",,,1

Question

FB44009
SHELL 命令で DOS アプリケーションを呼び出すと、MS-DOS プロンプト画面上に「パラメータの値が許容範囲を超えています」「定義ファイルが開けません」と表示されてしまいます。

Answer

該当する DOS アプリケーションのプロパティで環境変数領域のサイズが不正な値に設定されているとこの現象が発生する場合があります。
エクスプローラを使用して該当する DOS アプリケーションのプロパティを編集し、環境変数領域のサイズを修正してください。

※Windows(R)95 の障害のため、DOS アプリケーションのプロパティが勝手に不正な値に設定されてしまう場合があります。

Question

FB44010
SHELL 命令でバッチファイルを実行したときに「環境変数領域が足りません」と表示されてしまいます。

Answer

Windows の問題です。
エクスプローラーでバッチファイルのプロパティを選択して、環境変数領域のサイズを変更してください。

※Windows(R)95／Windows(R)98 のバッチファイルではあらかじめ指定しておかないと必要最小限の環境変数の領域しか確保されないため、環境変数を追加したり変更したりしたときにこのエラーが発生する可能性があります。

Question

FB44011
作成したプログラムを RUN 命令で呼び出したときに、[55]「ファイル記述に誤りがあります」というエラーが発生してしまいます。

Answer

RUN 命令で BASIC で作成したプログラムを実行する場合には、プログラムファイル名(*.bas)ではなく、実行ファイル名(*.exe)を指定してください。

誤) run "test.bas"
正) run "test.exe"

Question

FB44012
F-BASIC で作成した実行ファイルを実行中に中断させるキーはありますか？

Answer

Ctrl キーと C キーを同時に押下することで、実行中のプログラムを中断させることができます。

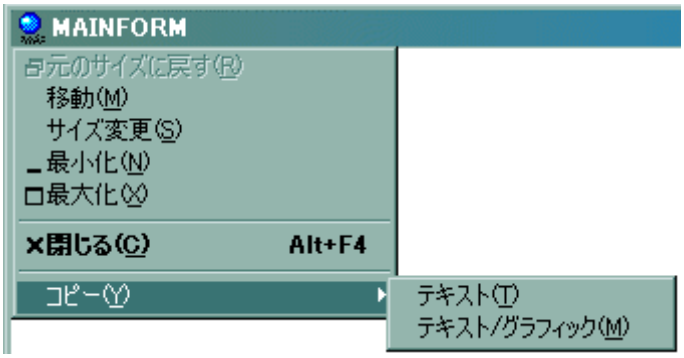
Question

FB44013
実行画面をクリップボードにコピーする方法は？

Answer

次の方法で実行画面をクリップボードにコピーすることができます。＜クライアントエリア内のみをコピーする方法＞では、実行ウィンドウの枠の内側だけをコピーします。それに対し、＜実行ウィンドウ全体をコピーする方法＞では、実行ウィンドウの枠まで含めてコピーします。

＜クライアントエリア内のみをコピーする方法＞
（1）作成したプログラムの実行中のウィンドウ左上隅のアイコンをクリックします。
（2）表示されたメニューのうち、[コピー]コマンドを選択し、その右側に表示された[テキスト]コマンドか[テキスト／グラフィック]コマンドを選択します。



[テキスト]コマンドを選択した場合は、実行ウィンドウに表示されているテキスト部分のみをクリップボードにコピーします。

＜実行ウィンドウ全体をコピーする方法＞
作成したプログラムが実行中に Alt キーと PrintScreen キーを同時に押下してください。この方法では、テキストのみのクリップボードへのコピーは行えません。

Question

FB44014
プログラムの起動時に「プログラム開始エラー ***.exe - DLL が見つかりません。ダイナミックリンクライブラリ SHLWAPI.DLL が指定されたパス ***に見つかりません」というメッセージボックスが表示されてしまいます。

Answer

F-BASIC V6.0 L10 初期バージョンの障害です。Windows(R)95 で、Windows の DLL のバージョンが古いとこの現象が発生する場合があります。Internet Explorer3.0 以降をインストールすることにより回避可能です。
また、F-BASIC V6.0 L10 U001 アップデートパックを適用することで回避することも可能です。

＜印刷＞

Question

FB45001
LPRINT 命令で何も印字されません。

Answer

プリンタが Windows の「通常使用するプリンタ」に設定されていない可能性があります。[スタート]-[設定]-[プリンタ]でお使いのプリンタを通常使用するプリンタに設定してください。

F-BASIC V5.0 以降のバージョンで、LPRINT TYPE で指定された制御コード系がプリンタの制御コード系と異なっている可能性があります。制御コード系を確認してください。制御コードを使用しない場合には LPRINT TYPE を"DC"に設定してください。

また、F-BASIC V4.1 では、Windows 専用プリンタからの LPRINT 命令による印刷はできません。

Question

FB45002
LPRINT 命令で漢字が文字化けしてしまいます。

Answer

Windows 専用プリンタの場合、LPRINT による漢字文字は出力されません。
MS-DOS 対応プリンタを使用しても漢字が文字化けする場合は、LPRINT TYPE で指定された制御コード系がプリンタの制御コード系と異なっている可能性があります。制御コード系を確認してください。
Windows 専用プリンタしかない場合およびプログラムからプリンタ制御コードを使用しない場合には LPRINT TYPE 命令を"DC"に設定しプログラムの先頭に記述してください。

F-BASIC V4.1 につきましては、LPRINT TYPE を"DC"を使用できません。製品に添付されている「ソフトウェア説明書(README.TXT)」に対応方法が掲載されています。

Question

FB45003
LPRINT 命令でプリンタの制御コードが正しく処理されません。

Answer

F-BASIC V5.0 以上のバージョンで、LPRINT TYPE で指定された制御コード系がプリンタの制御コード系と異なっている可能性があります。制御コード系を確認してください。

プリンタが該当する制御コードをサポートしていない可能性があります。各プリンタの対応状況に付いてはプリンタのメーカーへお問い合わせください。

メモ:富士通製のプリンタは Windows 環境下では FM シークエンスを使用できません。

LPRINT に改行位置が設定されていると、制御コード中に改行コードを挿入されてしまいます。WIDTH LPRINT または、WIDTH "LPT3:"で改行幅を 0 に設定してください。

LPRINT TYPE が"TEXT"／"ESC/P"／"PC-PR"に設定されているとき、制御コード中に&h80～&hFF のコードが含まれていると漢字コードに変換されてしまい、正しく処理されません。制御コードを送る前に LPRINT TYPE "RAW"に設定して漢字コードの変換を禁止してください。

修正前)
```
lprint chr$(&h1B,&h21,&h80);
lprint "ABC";
lprint chr$(&h1B,&h21,&h0);
```

修正後)
```
lprint type "RAW"
lprint chr$(&h1B,&h21,&h80);
lprint type "ESC/P"
lprint "ABC";
lprint chr$(&h1B,&h21,&h0);
```

Question

FB45004
LPRINT TYPE "ESC/P"等で漢字を出力した直後に LPRINT TYPE "RAW"にして 1 バイト特殊文字を出力すると文字化けしてしまいます。

Answer

LPRINT TYPE "RAW"を実行したときにプリンタの漢字出力は解除されません。必要ならば、LPRINT TYPE "RAW"を実行した後に 漢字出力を解除する制御コードを直接送出してください。

Question

FB45005
プリンタの制御コードを知りたい場合は？

Answer

プリンタの制御コードはメーカーや機種によって異なるため、こちらでは答えていません。プリンタに添付されているマニュアルを参考にしていただくか、プリンタのメーカーに問い合わせてください。

Question

FB45006
OPEN 命令／OPENPRINTER 命令でネットワーク上のプリンタをオープンしようとするとエラーが発生してしまいます。

Answer

プリンタにポート名が正しく割り当てられていない可能性があります。

＜Windows(R)98,Windows(R)95 の場合＞
（1） スタートメニューから[スタート]-[設定]-[プリンタ]を選択して、プリンタの設定ダイアログを開く。
（2） プリンタを選択する。
（3） [ファイル]-[プロパティ]を選択して、プリンタのプロパティダイアログを開く。
（4） [詳細]タブを開く。

（5）　[ポートの割り当て]を選択して、ポートの割り当てダイアログを開く。
（6）　ネットワークプリンタのアドレス(¥¥SERVER1¥PRINTER1)とポート名(LPT2:)を指定して、[OK]ボタンを押す。
（7）　ポートの割り当てダイアログの[出力先のポート]をポート名(LPT2:)に代えて、[OK]ボタンを押す。
（8）　OPEN 命令のポート名を"LPT2:"に代える。

＜WindowsNT(R)4.0 の場合＞
ネットワークプリンタをインストールすると、自動的にポート名"Ne01:"～"Ne99:"が割り当てられます。ポート名を変更することはできません。

Question
FB45007
F-BASIC の[ファイル]メニューの[ページレイアウトの設定]コマンドで用紙やページの余白の大きさを変更しても LPRINT 命令に反映されません。

Answer
F-BASIC の[ファイル(F)]-[ページレイアウトの設定(U)...]はプログラムのリストを印刷する際の用紙の設定です。LPRINT 命令などには反映されません。
LPRINT 命令での印字方法を設定するには、Windows のスタートメニュー[スタート]-[設定(S)]-[プリンタ(P)]から[通常使用するプリンタ]を開き、プリンタのプロパティで設定してください。

Question
FB45008
LPRINT CHR$(12)を記述してもプログラムの実行途中で排紙されません。

Answer
次のようにセミコロンを記述してください。

```
lprint chr$(12);
```

Question
FB45009
WIDTH 命令で改行幅を指定したのにそのとおりに改行されません。

Answer
WIDTH 命令で改行幅を指定する場合のデバイス名は"LPT3:"を指定してください。

例）
```
' LPRINT の桁折り返しを禁止します
  width "LPT3:",0
  lprint “ABCDEFGHIJKLMNOPQRSTUVWXYZ・・・”
```

Question

FB45010
B4 の用紙に印字したいのですが、用紙の途中で改行が入ってしまいます。用紙に合わせて改行幅を指定する方法は？

Answer

WIDTH 命令もしくは、WIDTH LPRINT 命令で改行幅を指定することができます。

例 1）

```
width "LPT3:",120      ' 120 桁で改行します
lprint "ABCDEFGHIJKLMNOPQRSTUVWXYZ・・・"
```

例 2）

```
width lprint 120      ' 120 桁で改行します
lprint "ABCDEFGHIJKLMNOPQRSTUVWXYZ・・・"
```

また[通常使用するプリンタ]のプロパティで用紙のサイズを設定しておく必要があります。

Question

FB45011
罫線は印刷できますか？

Answer

F-BASIC V6.3 より、直線や矩形を描画する命令が追加されました。

例 1）罫線を描画します。

```
#include "WINDOWS.BI"
LPRINT TYPE "DC"     ' 枠を描画
LPRINTDRAWLINE (1*1440),(1*1440),(3*1440),(1*1440)
LPRINTDRAWLINE (3*1440),(1*1440),(3*1440),(3*1440)
LPRINTDRAWLINE (3*1440),(3*1440),(1*1440),(3*1440)
LPRINTDRAWLINE (1*1440),(3*1440),(1*1440),(1*1440)
LPRINTSETPOS (1*1440),(1*1440)   ' 枠内に印字
LPRINT "枠"
lprint chr$(12);
stop
end
```

例 2）矩形を描画します。

```
#include "WINDOWS.BI"
LPRINT TYPE "DC"
LPRINTSETCOLOR RGB(255,0,0)    ' 塗りつぶされた矩形(赤)を描画
LPRINTFILLRECT (1*1440),(1*1440),(3*1440),(3*1440)
LPRINTSETCOLOR RGB(255,255,255)    ' 矩形内に印字(白ぬき)
LPRINTSETPOS (1*1440),(1*1440)
LPRINT "枠"
lprint chr$(12);
stop
end
```

Question

FB45012
ハードコピーを採ることはできますか？

Answer

WINHARDC 命令で実行画面のハードコピーをプリンタへ出力することができます。WINHARDC 命令は Windows 拡張命令なので、プログラムの先頭に「#include "WINDOWS.BI"」と記述してください。

例）

```
#include "WINDOWS.BI"
circle(100,100),100
WINHARDC
stop
end
```

Question

FB45013
実行画面を反転させたハードコピーを採ることはできますか？

Answer

WINHARDC 命令の「反転スイッチ」を指定することにより、実行画面を反転させたハードコピーを採ることが可能です。

例）

```
#include "WINDOWS.BI"
circle(100,100),100
WINHARDC ,1      ' 反転スイッチ「1」を指定します
stop
end
```

Question

FB45014
ネットワーク上に接続されている複数台のプリンタに振り分けて出力できますか？

Answer

プリンタにポート名が割り当ててあれば可能です。ネットワーク上のプリンタへのポート名を割り当てる方法は、FB45006 を参考にしてください。

＜ネットワーク上の複数台のプリンタに振り分けて印字するサンプル＞

```
#include "windows.bi"
var shared PRN as object
PRINTEROBJECT PRN

'LPT5 に割り当てられたプリンタへの出力
PRN.OPENPRINTER "LPT4:"
PRN.STARTDOC "sample1"
PRN.STARTPAGE
PRN.print "これは 1 台目のネットワークプリンタからの出力です"
PRN.ENDPAGE
PRN.ENDDOC
PRN.CLOSEPRINTER

'LPT6 に割り当てられたプリンタへの出力
PRN.OPENPRINTER "LPT5:"
PRN.STARTDOC "sample2"
PRN.STARTPAGE
PRN.print "これは 2 台目のネットワークプリンタからの出力です"
PRN.ENDPAGE
PRN.ENDDOC
PRN.CLOSEPRINTER

'LPT7 に割り当てられたプリンタへの出力
PRN.OPENPRINTER "LPT6:"
PRN.STARTDOC "sample3"
PRN.STARTPAGE
PRN.print "これは 3 台目のネットワークプリンタからの出力です"
PRN.ENDPAGE
PRN.ENDDOC
PRN.CLOSEPRINTER

stop
end
```

Question

FB45015
プリンタオブジェクトを使用した印刷で文字が小さく印字されてしまいます。

Answer

SETFONTSIZE 命令を使用して文字の大きさを指定してください。

例)プリンタに印字する文字の大きさを指定します。

```
#include "WINDOWS.BI"
var PRN as object
var PPRN as object
PRINTEROBJECT PRN
if PRN.PRINTDLG(PPRN) then
  PRN.STARTDOC ""
  PRN.STARTPAGE
  PRN.SETMAPMODE 2
  PRN.SETFONTNAME "MS 明朝"    'MS は全角英字で指定します
  PRN.SETFONTSIZE 10           'フォントのサイズを指定します
  PRN.print "10 ポイントの印字"
  PRN.ENDPAGE
  PRN.ENDDOC
  PRN.CLOSEPRINTER
endif
stop
end
```

Question

FB45016
指定と異なるフォントで印字されてしまいます。

Answer

SETFONTNAME 命令を使用して文字のフォントを指定してください。"明朝"、"ゴシック"、"SYSTEM"などのフォントはプリンタによって名前が異なったりインストールされていなかったりする場合があるため、まったく違うフォントで印刷されてしまう可能性があります。"MS 明朝"か"MS ゴシック"はすべてのプリンタで同じように使用できますので、こちらの使用を推奨します。なお、このときフォント名の「M」と「S」は全角で、「MS」と「明朝」(「ゴシック」)の間に半角のスペースが 1 つ入ります。全角／半角等を間違えるとフォントが正しく選択されませんのでご注意ください。

例)プリンタに印字する文字のフォントを指定します。

```
#include "WINDOWS.BI"
var PRN as object
var PPRN as object
PRINTEROBJECT PRN
if PRN.PRINTDLG(PPRN) then
  PRN.STARTDOC ""
  PRN.STARTPAGE
  PRN.SETMAPMODE 2
  PRN.SETFONTNAME "MS 明朝"  'MSは全角英字で指定します
  PRN.SETFONTSIZE 10
  PRN.print "明朝体の印字"
  PRN.ENDPAGE
  PRN.ENDDOC
  PRN.CLOSEPRINTER
endif
stop
end
```

※プリンタにインストールされているフォントは次のプログラムで確認できます。

```
#include "WINDOWS.BI"
var PRN as object
var PPRN as PRINTPARAM
print "ディスプレイにインストールされているフォントを一覧表示します。"
input "次の処理へ進みます";A$
F$=""
do
  F$=ENUMFONTNAMES(F$)
  print F$,
loop while F$<>""
print "プリンタにインストールされているフォントを一覧表示します。"
input "次の処理へ進みます";A$
PRINTEROBJECT PRN
if PRN.PRINTDLG(PPRN) then
  F$=""
  do
    F$=PRN.ENUMFONTNAMES(F$)
    print F$,
  loop while F$<>""
  PRN.CLOSEPRINTER
endif
stop
end
```

Question

FB45017
SETUPPRINTER 関数のダイアログでプリンタの設定をしても、その後の OPENPRINTER 命令に反映されません。

Answer

SETUPPRINTER 関数の設定は、OPENPRINTER 命令で同じオブジェクト変数を使用して、同じポートを開いた場合にのみ有効になります。OPENPRINTER 命令のパラメータで別のプリンタを指定した場合には、SETUPPRINTER 関数の設定は無視されます。

例)

```
A=PRINTER1.SETUPPRINTER
PRINTER1.OPENPRINTER "LPT2:"
```

↓

SETUPPRINTER 関数の「プリンタの設定」ダイアログで LPT1:に接続されたプリンタを選択していた場合には「プリンタの設定」ダイアログの内容は無視される。

F-BASIC V6.3 では、OPENPRINTER 命令に、SETUPPRINTER 関数の「プリンタの設定」ダイアログで選択されたプリンタをオープンするという指定が追加されています。

例)

```
A=PRINTER1.SETUPPRINTER
PRINTER1.OPENPRINTER "$SETUPPRINTER"
```

↓

必ず、SETUPPRINTER 関数の「プリンタの設定」ダイアログで選択されたプリンタをオープンする。

Question

FB45018
フォントの種類や大きさを指定して印字することができますか？

Answer

LPRINT TYPE "DC"を使った印刷では、次の拡張命令を使用して印字に効果を付けることが可能です。

命令	効果
LPRINTSETCOLOR	文字色の変更
LPRINTSETFONT	印字フォント／文字サイズの変更
LPRINTSETMARGIN	マージンの指定
LPRINTSETPITCH	文字間隔の変更
LPRINTSETPOS	印字位置の指定
LPRINTDRAWLINE	直線の描画
LPRINTFILLRECT	矩形の描画

※これらの拡張命令を利用する場合には、プログラムの先頭に「#include "WINDOWS.BI"」を記述する必要があります。
※これらの拡張命令は F-BASIC V6.3 の新機能です。

＜文字色を変更するサンプル＞

```
#include "WINDOWS.BI"
lprint type "DC"
LPRINTSETCOLOR RGB(255,0,0)
lprint "赤"
lprint chr$(12);
stop
end
```

＜印字フォントを変更するサンプル＞

```
#include "WINDOWS.BI"
lprint type "DC"
LPRINTSETFONT "MS ゴシック",0,0,0,0  'MSは全角英字
lprint "ゴシック"
lprint chr$(12);
stop
end
```

＜文字サイズを変更するサンプル＞

```
#include "WINDOWS.BI"
lprint type "DC"
LPRINTSETFONT "MS 明朝",420,0,0,0  'MS は全角英字
LPRINTSETPITCH 288,480
lprint "4 倍角"
lprint chr$(12);
stop
end
```

＜文字間隔を変更するサンプル(1)＞

```
#include "WINDOWS.BI"
lprint type "DC"
LPRINTSETPITCH 96,240
lprint "15CPI"
lprint chr$(12);
stop
end
```

＜文字間隔を変更するサンプル(2)＞

```
#include "WINDOWS.BI"
lprint type "DC"
LPRINTSETFONT "MS P 明朝",0,0,0,0  'MS,P は全角英字
LPRINTSETPITCH -1,240
lprint "Proportional Font"
lprint chr$(12);
stop
end
```

＜印字位置を指定するサンプル＞

```
#include "WINDOWS.BI"
lprint type "DC"
X=LPRINTGETPAPERSIZE(0)/2
Y=LPRINTGETPAPERSIZE(1)/2
LPRINTSETPOS X,Y
lprint "紙の真ん中"
lprint chr$(12);
stop
end
```

LPRINT TYPE "RAW"／"ESC/P"／"PC-PR"等を使った印刷では、プリンタの制御コードを使用した制御が可能です。ただし、プリンタによって制御コードが利用できなかったり、制御コードが異なったりする場合があります。制御コードについてはプリンタのメーカーにお問い合わせください。

＜演算＞

Question

FB46001
1～2 桁程度の小数を含む演算を行なったときに、演算結果に小数点以下 6 位～15 位くらいの端数が生じることがあります。

Answer

数値のまるめ誤差によるものです。
数値のまるめ誤差の影響のない演算が必要な場合には BCD 演算を利用してください。
表示を整えたい場合には、PRINT USING 命令や FORMAT$関数を利用してください。

＜BCD 演算を使ったプログラムの例＞

```
def bcd "2.2" as A$
def bcd "2.2" as S$
bcd A$="0.01"
bcd S$="0"
for I=1 to 100
  bcd S$=S$+A$
next
print S$
stop
end
```

＜print using 命令を使ったプログラムの例＞

```
=0.01
S=0
for I=1 to 100
  S=S+A
next
print using "##.##";S
stop
end
```

解説:
コンピュータ内部の演算は 2 進数で行なわれます。
しかし、10 進数で 1/3 が正確に表現できないように、小数の 0.1 は 2 進数で正確に表現できない数値となっているため、例えばあなたが「0.1」と入力すると、これは実際には「0.1 にもっとも近い数値」に変換されてしまいます。
このため、倍精度実数／単精度実数を用いて小数を含む演算を行なうと、0.01 を 100 回足しても 1 ちょうどにならないといったことが起こることがあります。

10 進数の場合に置き換えてみると分かりやすいと思います。

A=1/3
B=A*3

という演算処理があったとします。演算の誤差がなければ B の値は 1 ちょうどになるはずです。
しかし、精度が 3 桁の 10 進数の計算では、A の値は通常 0.333 になります。B は A の 3 倍であるため 0.999 となり、1 にはなりません。
精度が 6 桁の 10 進数の計算では、A の値は通常 0.333333 になります。B は A の 3 倍であるため 0.999999 となり、やはり 1 にはなりません。
上の例で結果がいずれも 1 にならないのは 1/3 が 10 進数で記述できない値であり、10 進数で書き下した時点で既に本来の数値と異なる値になっているためで、演算の精度を上げるだけでは解消されません。このように数値の記述方法の違いによって生じる誤差を数値のまるめ誤差といいます。

10 進数の小数には 2 進数で記述できない値が多数存在します。
倍精度実数／単精度実数は 2 進数による演算であるため数値を 2 進数で記述する必要があり、この時点でまるめ誤差が生じます。
まるめ誤差は演算精度の範囲内に収まるため、科学技術計算において問題になることはあまりありませんが、定数との比較を含む計算を行なう場合には演算精度の範囲内であっても結果が変わってしまう場合があるため、小数を含む金額の計算には向きません。

BCD 演算では数値を 10 進数として扱うため、数値の変換誤差を生じません。
10 進数のまるめ誤差を防ぐためには、BCD 演算を使用してください。

Question

FB46002
計算した結果、小数点以下の桁がたくさん表示されてしまいます。

Answer

PRINT USING や FORMAT$などを使用して桁数を整えてください。

例）

プログラム例	実行例
input A input B print using "##.##";A/B stop end	? 10 ? 3 3.33

＜割り込み＞

Question

FB48001
イベントプロシージャ内で割り込み制御命令を記述すると、翻訳時エラー[76]「ここでは宣言または実行できません」が発生してしまいます。

Answer

イベントプロシージャ内では割り込み制御命令を使用できません。

＜入力＞

Question

FB49001
INKEY$関数／INPUT 命令でキー入力を読み取れません。

Answer

キー入力フォーカスがフォーム以外のボタンなどのコントロールにあると、INKEY$関数／INPUT 命令ではキー入力を読み取れません。マウスでフォームをクリックしてからキー入力を行ってください。
なお、イベントプロシージャ内では INKEY$関数／INPUT 命令を使用できません。

Question

FB49002
漢字入力／英数字入力を自動的に切り替えられますか？

Answer

SETIMEMODE 命令を使用してください。

例)

```
#include "WINDOWS.BI"
SETIMEMODE 4 '漢字入力 ON
input "名前(漢字表記)=",K1$
SETIMEMODE 2 '漢字入力 OFF
input "名前(ローマ字表記)=",K2$
stop
end
```

Question

FB49003
PC-9801 のキーボードの仮想キーコード表はありますか？

Answer

109 キーボードの対応するキーの仮想キーコードを参照してください。

109 キーボードに対応するキーがないキーはパソコンによって仮想キーコードが異なる可能性があるため動作を保証しませんが、次のプログラムで確認可能できます。

＜仮想キーコード 0～255 に対応する各キーの状態をリアルタイムに表示するサンプル＞

```
#include "WINDOWS.BI"
dim K(256) as long
dim L(256) as long
locate 4,1
print "仮想キーコード"
for I=0 to 15
  locate 4,I+4
  print hex$(I)+"0";
next
for J=0 to 15
  locate J*3+8,2
  print "+"+hex$(J);
next
do
  F=0
  for I=0 to 15
    for J=0 to 15
      K(16*I+J)=GETASYNCKEYSTATE(16*I+J)
      if L(16*I+J)<>K(16*I+J) then
        F=1
      endif
    next
  next
  if F<>0 then
    for I=0 to 15
      for J=0 to 15
        locate J*3+8,I+4
        if L(16*I+J)<>K(16*I+J) then
          color 2
        else
          color 7
        endif
        print K(16*I+J);
        L(16*I+J)=K(16*I+J)
      next
    next
  endif
loop
```

※このプログラムは F-BASIC V6.3 の新機能を使用しています。
※このプログラムを終了させるには、実行ウィンドウの右上の「×」ボタンをクリックするか、Ctrl キーと C キーを同時に押下してください。

＜表示（グラフィック）＞

Question

FB50001
テキストの桁数／行数を 80 桁／25 行にする方法は？

Answer

「画面デザイン」を参照してください。

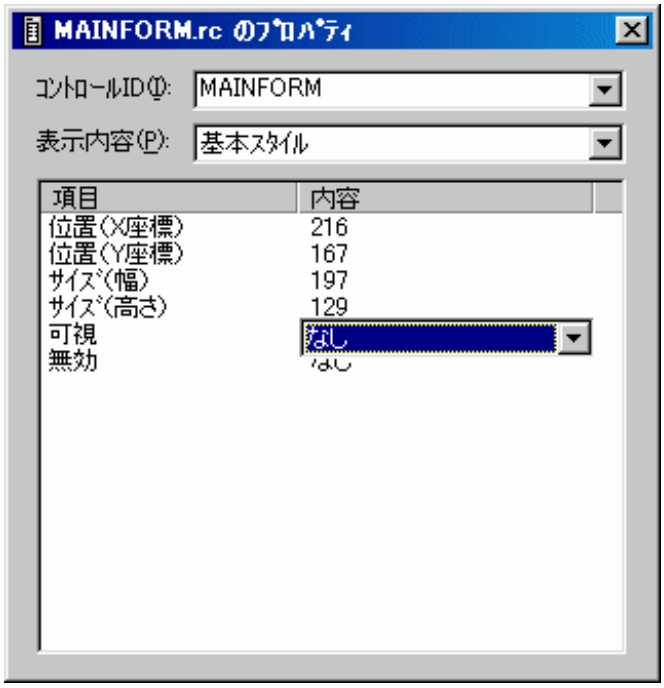
Question

FB50002
実行ウィンドウを表示させないで実行させることはできますか？

Answer

リソースファイルで実行ウィンドウの表示・非表示を設定できます。
次の手順で行います。また、リソースの利用方法につきましては、「FB61001 リソースを利用する方法は？」に掲載されています。

（1） MAINFORM のプロパティウィンドウを表示させます。
（2） [表示内容]を「基本スタイル」に変更します。
（3） [可視]の項目の内容を「なし」に設定します。
（4） 翻訳して実行ファイルを作成します。



また、リソースを利用しない場合でも、拡張命令の SHOWWINDOW の表示スイッチを「0」にすることで可能です。

＜ファイル"HIDUKE.TXT"に現在の日付と時刻を書き込むサンプル＞

```
#include "windows.bi"
SHOWWINDOW 0      '表示スイッチ「0」を指定
open "HIDUKE.txt" for create as #1
print#1,today$
print#1,time$
close #1
M$="プログラムを終了します"+chr$(13,10)+"HIDUKE.TXT に日付と時刻を書きこみました"
A=MESSAGEBOX("メッセージ",M$,0,3)
end
```

Question

FB50003
グラフィックの座標を 640×400 ドット相当にする方法は？

Answer

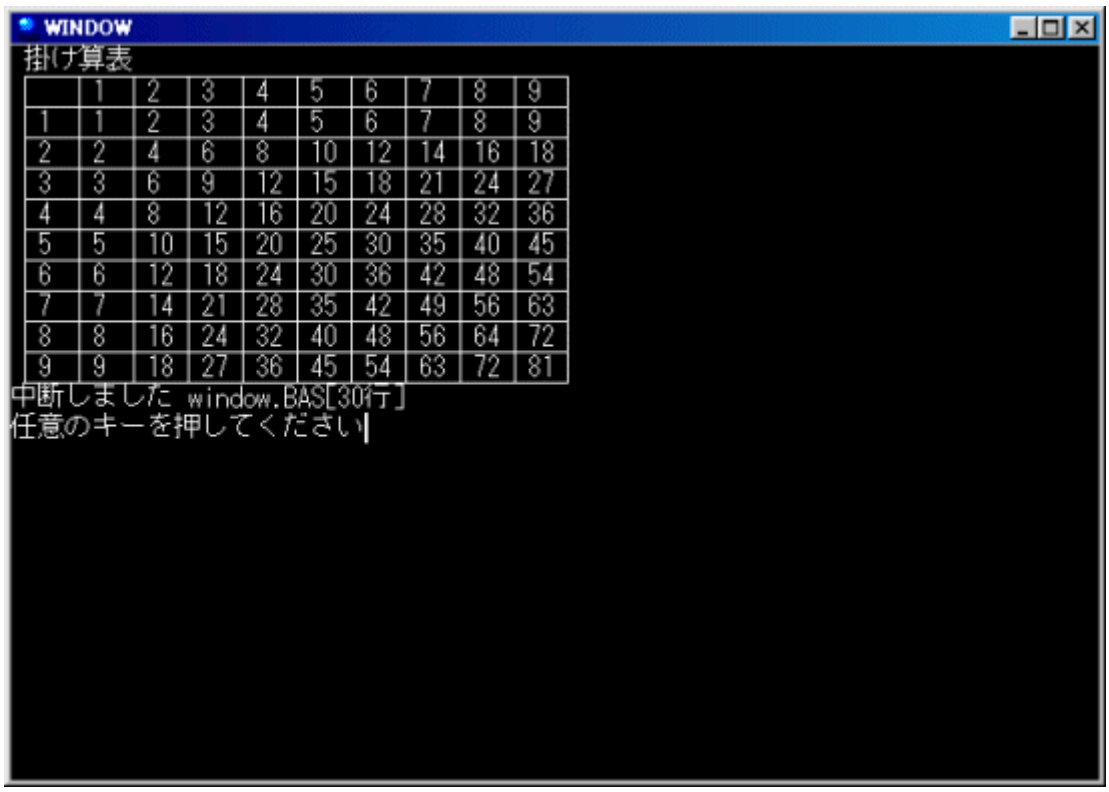
WINDOW 命令を使用することによりグラフィックの座標を 640x400 ドット相当にすることができます。

＜80 桁×25 行のテキストと 640×400 ドット相当のグラフィックを重ねて表示するサンプル＞

```
'表と罫線の表示
'テキストとグラフィックの座標を設定する
width 80,25
window (0,0)-(640,400)
'グラフィックで罫線を描画する
for I=0 to 10
  line ((8+(I*32)),16)-((8+(I*32)),(16+(10*16))),pset
next
for J=0 to 10
  line (8,(16+(J*16)))-((8+(10*32)),(16+(J*16))),pset
next
locate 1,0
print "掛け算表"
'テキストで掛ける数と掛けられる数を表示する
for I=1 to 9
  locate (1+(I*4)),1
  print I;
next
for J=1 to 9
  locate 1,(1+J)
  print J;
next
'テキストで掛け算の結果を表示する
for I=1 to 9
  for J=1 to 9
    locate (1+(I*4)),(1+J)
    print (I*J);
  next
```

```
next
stop
end
```

＜実行画面＞



	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

Question

FB50004
プログラムをフルスクリーンで実行させることはできますか？

Answer

できません。しかし、リソースファイルなどを利用することによって、実行画面を大きくすることができます。実行画面を大きくする方法は、「画面デザイン」を参照してください。

Question

FB50005
GET@命令で指定する配列変数の要素数の算出方法は？

Answer

GET@命令で図形情報を読み込む際に、データを格納する配列変数の要素数が必要数より少ないとエラーが発生します。配列の要素数は、バッファサイズを変数のバイトサイズを変数型による 1 要素のサイズで割り求めます。算出方法の例を次に示します。

（1）次の式でバッファサイズを求めます。

バッファサイズ＝40+カラーテーブル数*4+((ビット数*図形幅+31)¥32)*4*図形高さ

色数	カラーテーブル数	ビット数
2	2	1
16	16	4
256	256	8
65536	3	16
161777216	0	24
4294967296	3	32

※色数は、ご使用になっているディスプレイのプロパティで設定している色数を選択してください。

（2）算出したバッファサイズを配列変数の変数型による 1 要素のサイズで割ります。

配列の要素数＝バッファサイズ/配列 1 要素のバイトサイズ

配列要素の型	配列 1 要素のバイトサイズ
整数型	2
ロング整数型	4
単精度実数型	4
倍制度実数型	8

例）配列変数の変数型が整数型
　　画像サイズが縦 40 ピクセル、横 40 ピクセル
　　ディスプレイ 256 色の場合
バッファサイズ=40+256*4+((8*40+31)¥32)*4*40=2664
配列要素数=2664/2=1332

Question

FB50006
実行画面上の図をビットマップファイルとして保存する方法は？

Answer

SAVEFILE 命令を使用してください。

例）描画した画像をビットマップファイルで保存する

```
#include "WINDOWS.BI"
var BMP as object
var FORM as object

BITMAPOBJECT BMP
```

```
FORM.ATTACH GETHWND

SETMAPMODE 1
SETREDRAWMODE 1
line(0,0)-(100,100)
BMP.CREATEBITMAP 100,100
BMP.BITBLT 0,0,100,100,FORM,0,0
BMP.SAVEFILE "sample.bmp"
stop
end
```

Question

FB50007
フォントの種類や大きさを指定して表示することができますか？

Answer

Windows 機能を利用することによって、マルチフォントを使用して表示させることができます。次に例を示します。

```
'フォントのサイズを変更する
#include "WINDOWS.BI"
SETMAPMODE 2
SETFONTNAME "Times New Roman"
SETFONTSIZE 12
print "12point"
SETFONTSIZE 24
print "24point"
stop
end
```

```
'FONT 構造体によるフォントの変更
#include "WINDOWS.BI"
var FFONT as FONT
FFONT.SIZE=24
FFONT.BOLD=-1
FFONT.ITALIC=-1
FFONT.FFNAME="Times New Roman"
SETMAPMODE 2
SETFONT FFONT
print "ABCDEFG"
stop
end
```

＜プログラミング（中級）＞

＜マルチモジュール＞

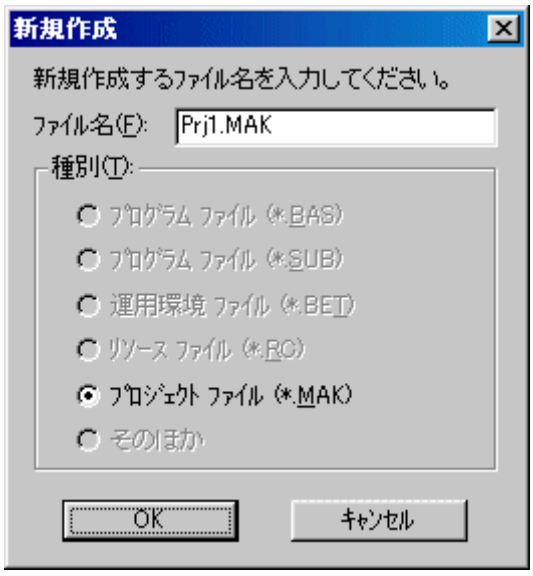
Question

FB60001
サブプログラムの利用方法は？

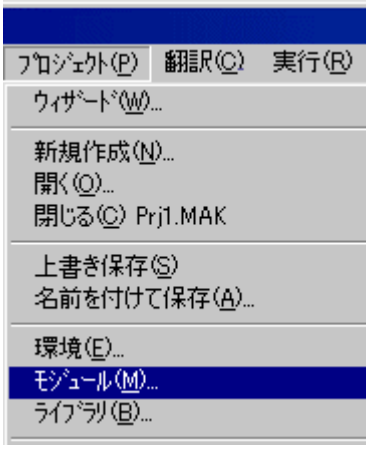
Answer

F-BASIC でサブプログラムを利用するためには、プロジェクトにそのプログラムを登録する必要があります。次に手順を示します。

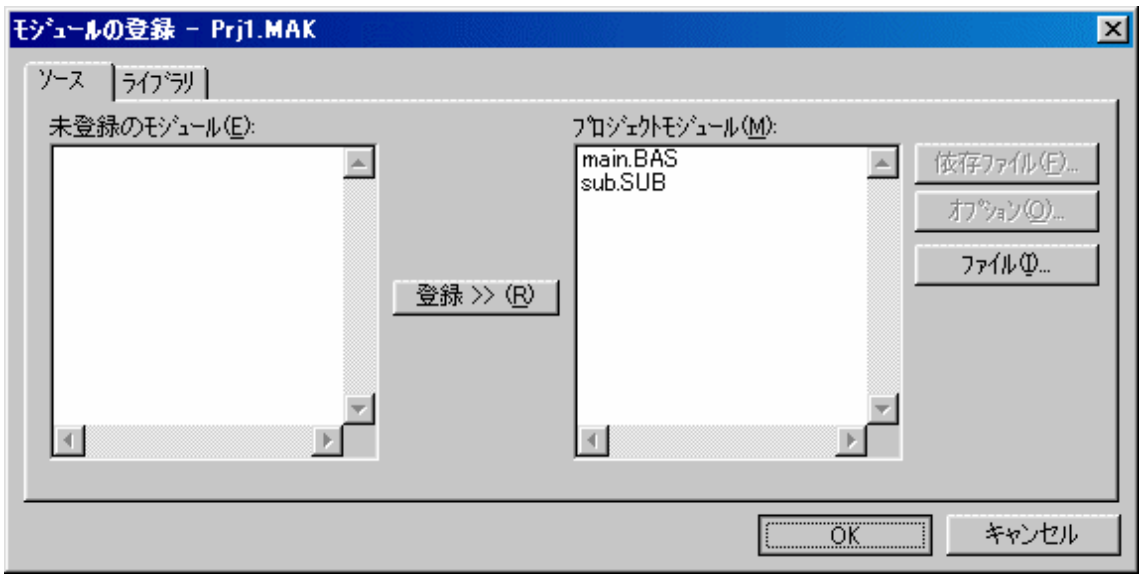
（1）F-BASIC 上でメインプログラムと利用したいサブプログラムを開きます。
（2）[プロジェクト]メニューの[新規作成]コマンドを選択し、表示されたダイアログボックスに任意の名前を記入して OK ボタンを押下します。→プロジェクトファイルが作成され、プロジェクトウィンドウが表示されます。



（3） [プロジェクト]メニューの[モジュール]を選択します。



(4) 表示されたダイアログボックスの左側のリストにあるメインプログラムのファイル名を選択して、真中の[登録]ボタンを押下します。→メインプログラムが右側のリストに移動します。
同様に左側のリストにあるサブプログラムのファイル名も選択して、[登録]ボタンを押下し、右側のリストに移動させます。



(5) モジュールを登録しおわったら[OK]ボタンを押下してダイアログボックスを閉じます。プロジェクト内容が次のように表示されます。

(6) [翻訳]メニューの[すべて翻訳]を選択します。この時、プロジェクトファイルやプログラムファイルを保存するかどうか聞いてきますので、すべてのファイルを同じフォルダに保存するようにしてください。
(7) 翻訳が成功すると、プロジェクトファイル名で実行ファイルが作成されます。

Question

FB60002
サブプログラム内で拡張命令を使用すると、翻訳時エラー[76]「ここでは宣言または実行できません」等のエラーが発生してしまいます。

Answer

必要な#include 文が記述されていない可能性があります。#include 文はメインプログラムとサブプログラムの両方に記述する必要があります。

誤)

＜main.bas＞

```
#include "windows.bi"
SETWINDOWTEXT "MAIN"
calls "DRAW"
stop
end
```

＜draw.sub＞

```
subpgm
SETWINDOWTEXT "DRAW"
returns
```

↓

正)

main.bas

```
#include "windows.bi"
SETWINDOWTEXT "MAIN"
calls "DRAW"
stop
end
```

draw.sub

```
#include "windows.bi"
subpgm
SETWINDOWTEXT "DRAW"
returns
```

#include 文が誤った場所に記述されている可能性があります。#include 文は subpgm 文／sub～end sub 文／function～end function 文よりも前に記述する必要があります。

誤)

```
subpgm
#include "windows.bi"
print "A"
returns
```

↓

正)

```
#include "windows.bi"
subpgm
print "A"
returns
```

同じファイル内に subpgm 命令と sub～end sub／function～end function を混在させることはできません。別のプログラムファイルに分けてください。

＜画面デザイン＞

Question

FB61001
リソースを利用する方法は？

Answer

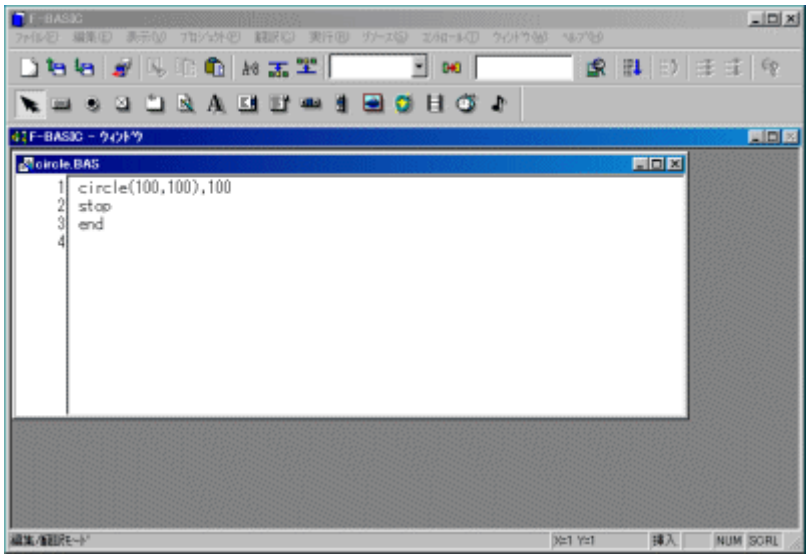
F-BASIC では、リソースファイルを利用し、実行画面のサイズや見た目、コントロールの配置などを設定することができます。

＜新規にプログラムを作成する場合＞
新規にリソースファイルを利用したプログラムを作成する場合は、ウィザード機能を利用すると簡単に作成することができます。ウィザード機能では、プロジェクトファイルへのモジュールやリソースファイルの登録などを自動的に行います。

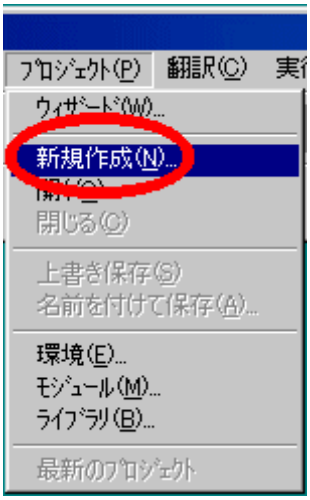
＜既存のプログラムファイルにリソースファイルを利用する場合
今まで作成したプログラムの実行画面を効率的に設定したり、新たな機能を追加する場合には、次の手順でリソースファイルを利用したプログラムが作成できます。

（1）　今まで作成したプログラムファイル(*.BAS)を開きます。(例：circle.bas)
＜プログラム内容例＞

```
circle(100,100),100
stop
end
```

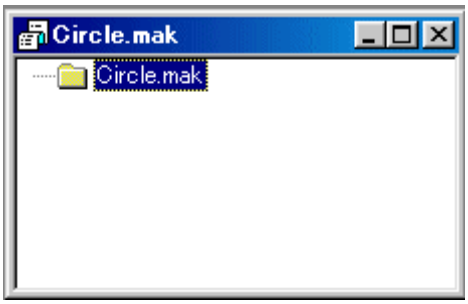


（2）　プロジェクトファイルを作成します。(例：CIRCLE.MAK)
1. [プロジェクト]メニューの[新規作成]コマンドを選択します。



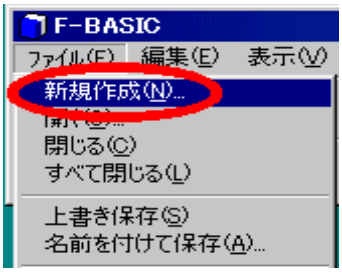
2. 表示されたダイアログボックスに任意のプロジェクトファイル名を入力し、[OK]ボタンをクリックします。

3. プロジェクトファイルが画面に表示されます。

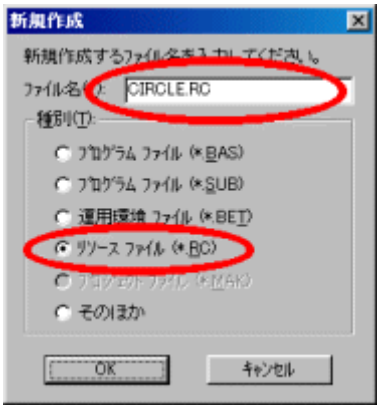


(3)　リソースファイルを作成します。(例：CIRCLE.RC)

1. [ファイル]メニューの[新規作成]コマンドを選択します。

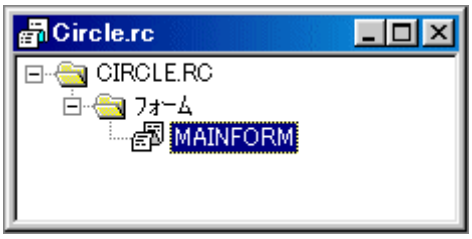


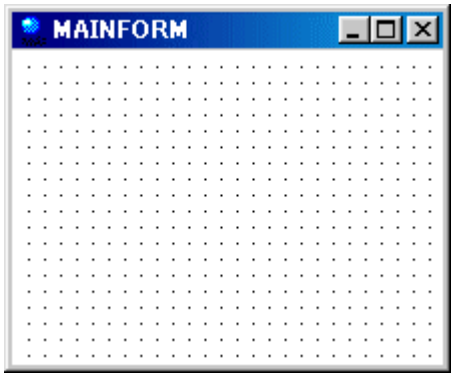
2. 表示されたダイアログボックスに任意のリソースファイル名を入力し、[ファイルの種類]で「リソースファイル(*.RC)」のラジオボタンを選択し、[OK]ボタンをクリックします。



3. リソースファイルと MAINFORM が表示されます。

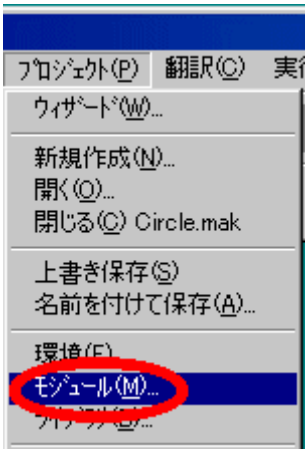
リソースファイル　　MAINFORM



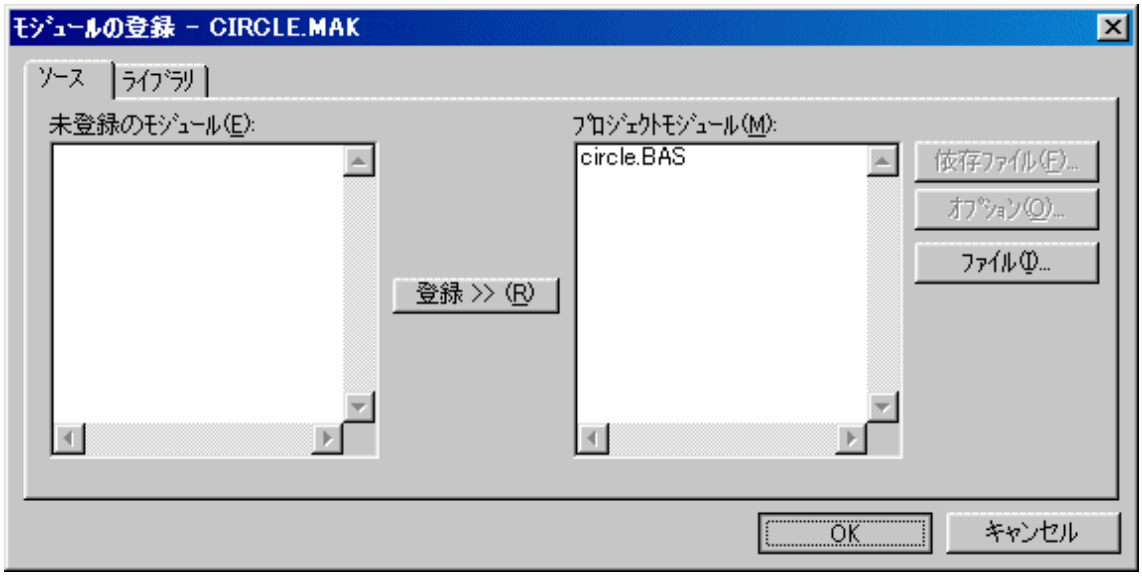


(4)　プロジェクトファイルにプログラムファイルとリソースファイルを登録します。

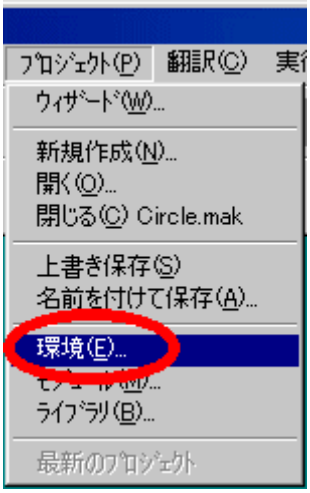
1. [プロジェクト]メニューの[モジュール]コマンドを選択します。



2. 表示されたダイアログボックスの左側のリストにあるプログラムファイル名を選択し、真中の[登録]ボタンをクリックします。すると右側のリストにプログラムファイル名が移動します。

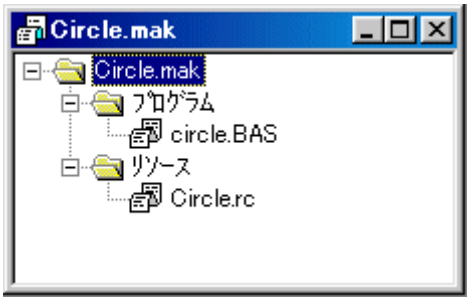


3. [プロジェクト]メニューの[環境]コマンドを選択します。



4. 表示されたダイアログボックスの「リソースファイル」の項目で、作成したリソースファイル名を選択し、[OK]ボタンをクリックします。

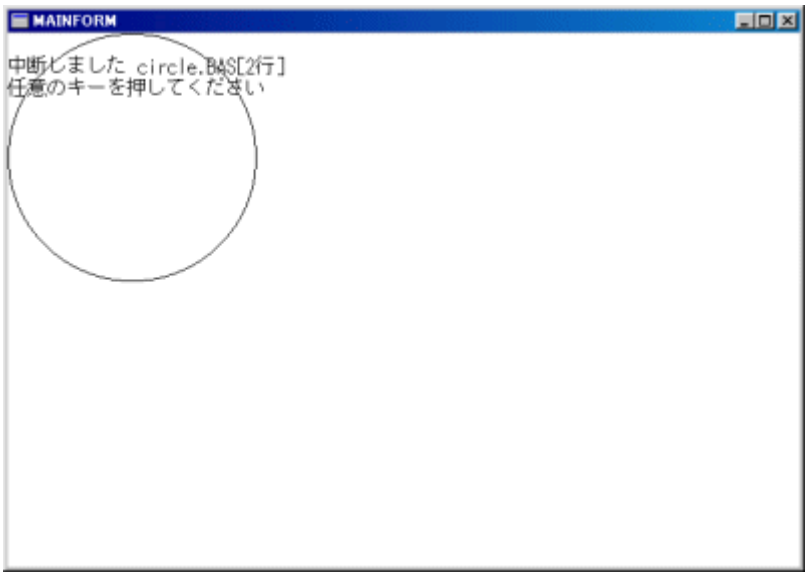
5. モジュールとリソースファイルの登録が終了したプロジェクトファイルの内容は次のようになります。



（5）プロジェクトファイルとリソースファイルを保存します。
作成したプロジェクトファイルとリソースファイルを保存します。プログラムファイルと同じフォルダに保存することをお薦めします。

（6）実行ファイルを作成します。
[翻訳]メニューの[すべて翻訳]コマンドを選択し、実行ファイルを作成します。

＜実行結果＞



Question

FB61002
実行画面を大きくする方法は？

Answer

F-BASIC の実行ウィンドウの初期の大きさは、80 桁 24 行です。実行ウィンド

ウの大きさを変更する場合にはいくつかの方法が用意されています。プログラムの構成や用途に応じて、方法を選択してください。

＜WIDTH 命令で調節する方法＞
WIDTH 命令で実行ウィンドウの桁数と行数を指定し、MAXIMIZEWINDOW 命令で最大表示することにより、実行ウィンドウの大きさを変更することができます。次の例は、桁数 132 桁、行数 40 行に設定した例です。この方法では、桁数は 132 桁まで、行数は 124 行まで指定することができます。

＜実行画面を 132 桁 40 行にします＞

```
#include "WINDOWS.BI"
width 132,40        '桁数、行数を指定します
MAXIMIZEWINDOW      '最大表示を指定します

stop
end
```

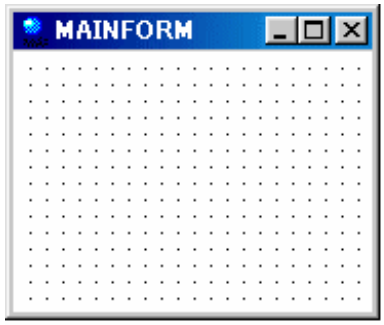
※WIDTH 命令での実行画面の桁数・行数を指定は、F-BASIC V6.0 以降の機能です。

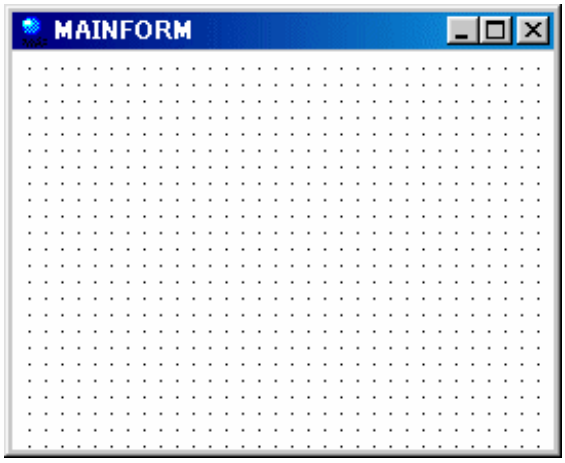
＜リソースを利用する方法（1）＞
リソースファイルを利用し、MAINFORM のプロパティの詳細を設定することによって、実行ウィンドウの大きさを変更することができます。詳細は、ワンポイントレッスンの「実行画面を大きくしよう」に掲載されています。この方法では、MAINFORM に使用するフォントの大きさを大きくすることで 80 桁 24 行の表示領域を大きくしています。

＜リソースを利用する方法（2）＞
上記の方法はフォントの大きさで調節しましたが、プロパティで桁数と行数も変更することができます。リソースの新規作成方法やプロジェクトの環境の設定方法などは、ワンポイントレッスンの「実行画面を大きくしよう」と同様ですので、プロパティの設定部分だけを紹介します。

（1）プロパティウィンドウの[コントロール ID]を「MAINFORM」にし、[表示内容]を「クライアントエリア」にします。
（2）[テキスト行数]に任意の数値を入力します。
　[テキスト桁数]に任意の数値を入力します。

（3）MAINFORM をマウスでドラッグして大きくします。





（4） 翻訳します。

Question

FB61003
自分で作成したアイコンを実行ファイルで使用する方法は？

Answer

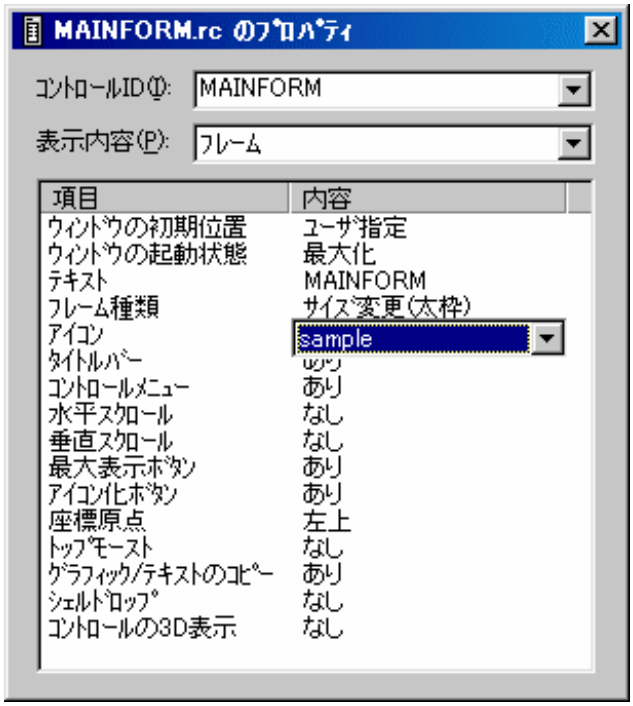
次の手順で自分で作成したアイコンを使用したプログラムが作成できます。

（1）F-BASIC の統合環境上でアイコンを使用したいプログラムのリソースファイルを開きます。

（2）[リソース]メニューの[インポート]コマンドを選択し、使用したいアイコン(*.ico)を開きます。

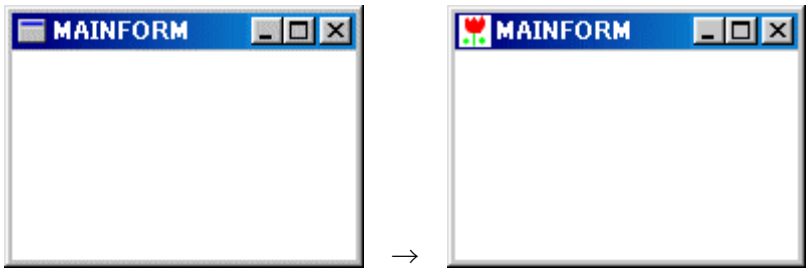
（3）プロパティウィンドウの[コントロール ID]を「MAINFORM」、[表示内容]

を「フレーム」にし、[アイコン]項目でインポートしたアイコン名を選択します。



（4） 翻訳します。

＜アイコン使用前＞　　＜アイコン使用＞



※例で使用したアイコンは、F-BAASIC V6.3 に添付されているサンプルプログラム内(C:¥FBASICV63¥Sample_B¥TTICON)にあります。（C ドライブに F-BASIC をインストールした場合のパスです。）

Question

FB61004

テキストの桁数／行数を 80 桁／25 行にする方法は？

Answer

＜WIDTH 命令で調節する方法＞

WIDTH 命令で実行ウィンドウの桁数と行数を指定し、MAXIMIZEWINDOW 命令で最大表示することにより、実行ウィンドウの大きさを変更することができます。この方法では、桁数は 132 桁まで、行数は 124 行まで指定することができます。

＜80 桁 25 行に設定するサンプル＞

```
#include "WINDOWS.BI"
width 80,25          '80 桁、25 行を指定します
MAXIMIZEWINDOW       '最大表示を指定します

stop
end
```

※WIDTH 命令での実行画面の桁数・行数を指定は、F-BASIC V6.0 以降の機能です。

＜リソースを利用する方法＞
プロパティウィンドウでは、ウィンドウ内の表示行数と桁数を設定することができます。リソースの作成方法やプロジェクトの環境の設定方法などは、ワンポイントレッスンの「実行画面を大きくしよう」を参考にしてください。行数、桁数の設定は次の手順で行います。

（1） プロパティウィンドウの[コントロール ID]を「MAINFORM」にし、[表示内容]を「クライアントエリア」にします。
（2） [テキスト行数]に「25」、[テキスト桁数]に「80」を入力します。
（3） MAINFORM をマウスでドラッグして大きくします。
（4） 翻訳します。

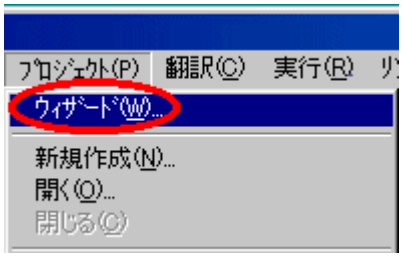
＜マルチウィンドウ＞

Question

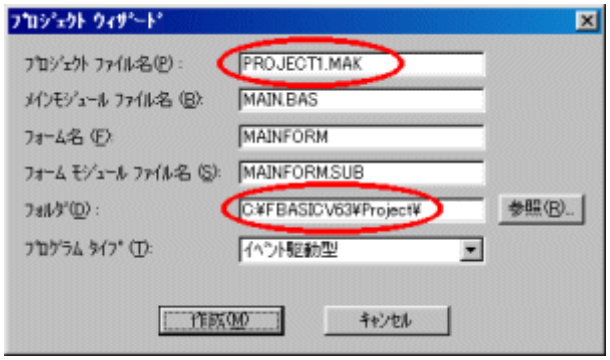
FB62001
複数フォームを使ったプログラムの作成方法は？

Answer

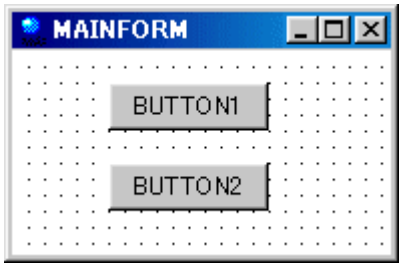
1 枚目のウィンドウはリソースファイルを作成する際に定義できますが、2 枚目以降は、[リソース]メニューの[新規作成]コマンドを選択し、新しいフォームを作成していきます。また、複数のウィンドウを制御するためには、それぞれのフォームにオブジェクト変数を割り当てて使用することになります。
次に 3 枚のフォームを利用した簡単なプログラムを紹介します。参考にしてください。
MAINFORM 上のコマンドボタン「BUTTON1」「BUTTON2」を押下すると FORM2、FORM3 がそれぞれ表示されるプログラムを作成します。既に表示されている場合はなにも処理を行いません。
（1）[プロジェクト]メニューの[ウィザード]コマンドを選択します。



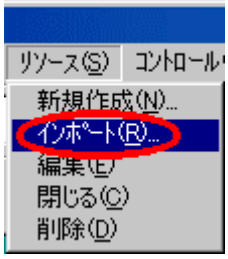
(2) 表示されたダイアログボックスで任意の[プロジェクトファイル名]と[フォルダ]を指定して、[作成]ボタンを押下します。



(3) MAINFORM にコマンドボタン「BUTTON1」と「BUTTON2」を配置します。



(4)　[リソース]メニューの[新規作成]コマンドを選択します。



(5) 表示されたダイアログボックスのリストからフォームを選択し、[OK]ボタンを押下すると、FROM2 というフォームが追加され表示されます。この作業（4）～（5）をもう一度繰り返すと、FORM3 というフォームが追加されます。

(6) MAINFORM に配置したコマンドボタンをダブルクリックすると、MAINFORM.sub にそれぞれのボタンに対するイベントプロシージャが自動的に生成されます。

＜MAINFORM.sub＞

```
'=============================================================
'                                                           =
'=============================================================
declare sub BUTTON1_ON edecl ()
sub BUTTON1_ON()

end sub
'=============================================================
'                                                           =
'=============================================================
declare sub BUTTON2_ON edecl ()
sub BUTTON2_ON()

end sub
```

（7）MAIN.bas、MAINFORM.sub それぞれのプログラムに太字の部分を追加します。各命令や関数については、お手元の文法書やリファレンス、オンラインヘルプを参考にしてください。

＜MAIN.bas＞

```
#include "windows.bi"
common shared FORM2 as object
common shared FORM3 as object
FORM2.CREATEWINDOW "FORM2"
FORM3.CREATEWINDOW "FORM3"
'
while 1
    WAITEVENT
wend
stop
end
```

＜MAINFORM.sub＞

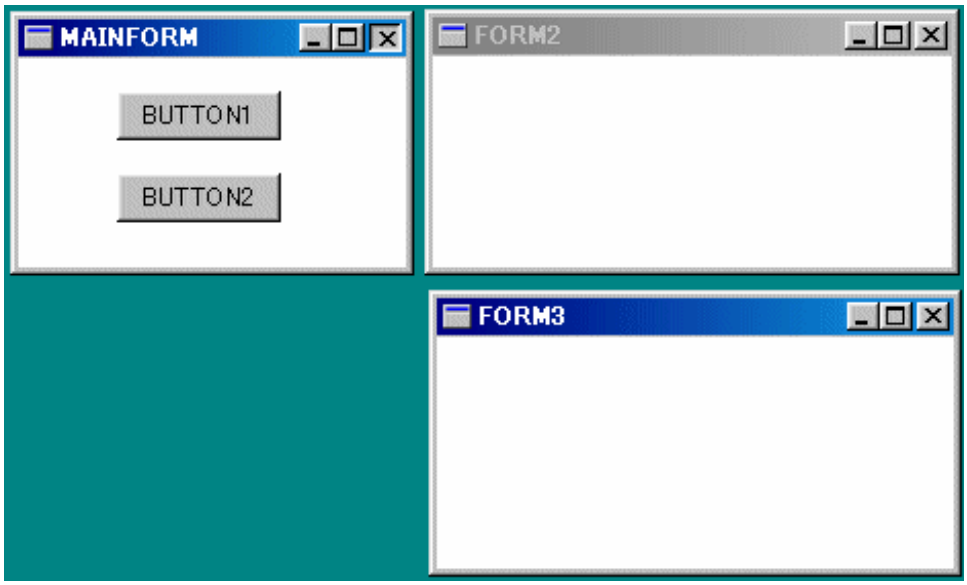
```
#include "windows.bi"
common shared FORM2 as object
common shared FORM3 as object

'=============================================================
'BUTTON1 のイベントプロシージャ
'=============================================================
declare sub BUTTON1_ON edecl ()
sub BUTTON1_ON()
    if not(CHECKOBJECT(FORM2)) then
        FORM2.CREATEWINDOW "FORM2"
    endif
    FORM2.SHOWWINDOW -1
    FORM2.SETFOCUS
```

```
end sub

'=============================================================
'BUTTON2 のイベントプロシージャ
'=============================================================
declare sub BUTTON2_ON edecl ()
sub BUTTON2_ON()
    if not(CHECKOBJECT(FORM3)) then
        FORM3.CREATEWINDOW "FORM3"
    endif
    FORM3.SHOWWINDOW -1
    FORM3.SETFOCUS
end sub
```

（8）翻訳し、実行させます。MAINFORM が表示され、BUTTON1 を押下すると FORM2 が、BUTTON2 を押下すると FORM3 が表示されます。



＜コントロール＞

Question

FB63001
エディットコントロールに複数行の文字列を設定する方法は？

Answer

複数行入力可能なエディットコントロールに複数行の文字列を設定するには、改行したい個所に CHR$(13,10)を挿入してください。次に記述例と実行結果を示します。

記述例

```
'複数行入力可能なエディットコントロールに
'複数行の文字列を設定します
#include "WINDOWS.BI"
var shared EDIT1 as object
EDIT1.ATTACH GETDLGITEM("EDIT1")

EDIT1.SETWINDOWTEXT "1 行目"+chr$(13,10)+"2 行目"+chr$(13,10)+"3 行目"

while 1
    WAITEVENT
wend

stop
end
```

実行結果

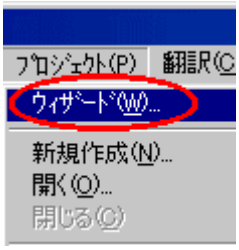
Question

FB63002
コントロールを使ったプログラムの作成例

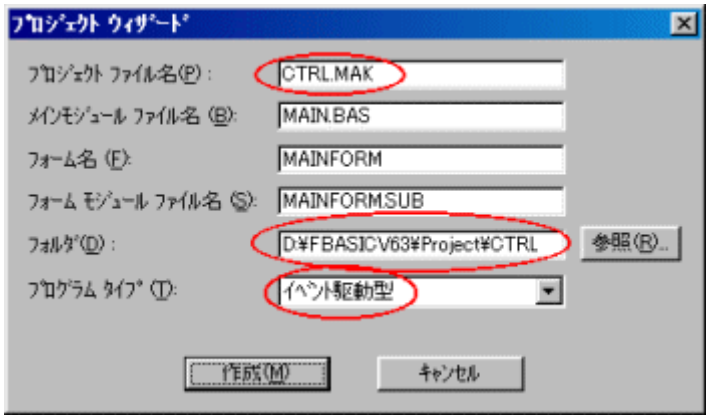
Answer

コントロールを使ったプログラムの作成例を紹介します。

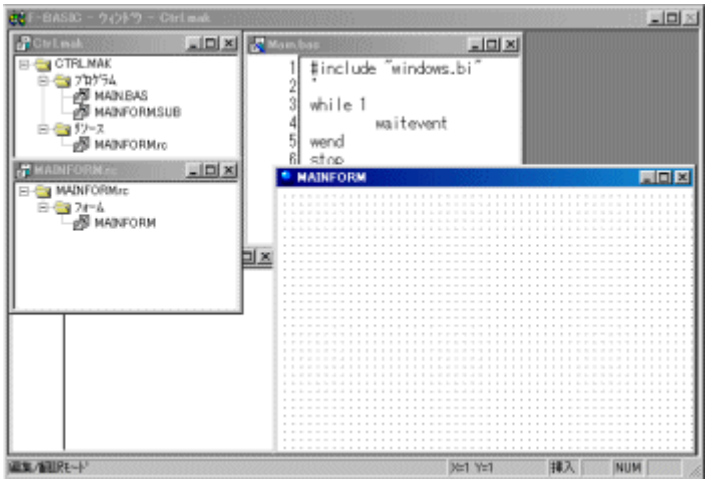
STEP1 プロジェクトを作成し、リソースファイルを登録する。
1-1 [プロジェクト(P)]-[ウィザード(W)...]を選択してプロジェクトを作成します。



1-2 フォルダとプロジェクトファイル名を指定して、プロジェクトタイプが"イベント駆動型"になっていることを確認して、[作成(M)]ボタンを押します。(例: CTRL1.MAK、D:¥FBASICV63¥PROJECT¥CTRL1)。



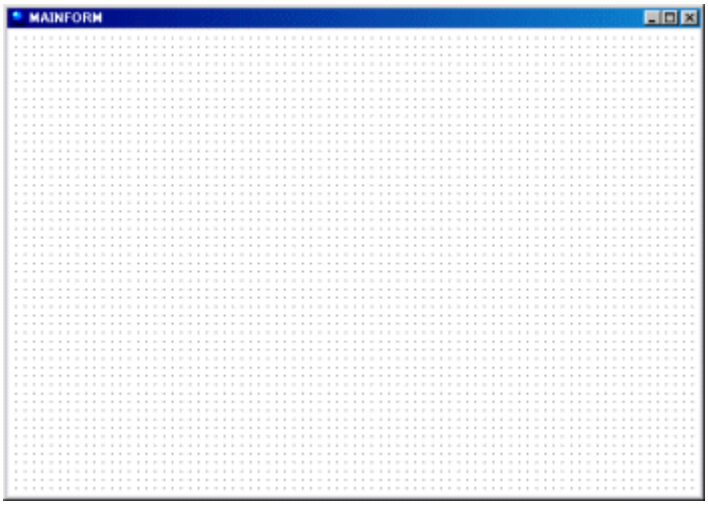
1-3 プロジェクトファイル(CTRL1.MAK)、プログラムファイル(MAIN.BAS)、プログラムファイル(MAINFORM.SUB)、リソースファイル(MAINFORM.RC)が作成されます。



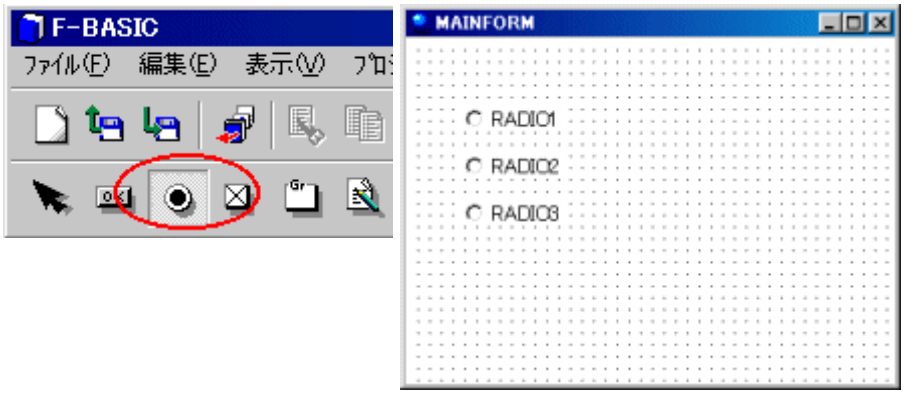
※ ウィザードはプログラムファイルとリソースファイルの登録を自動的に行ないます。ウィザードを使用していない場合には、プログラムファイルとリソースファイルの登録を手作業で行なう必要があります。

STEP2 フォーム上にコントロールを配置する。

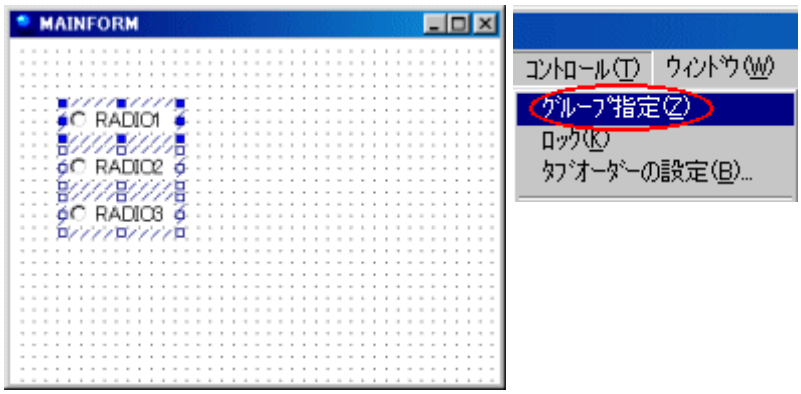
2-1 フォームを前面に表示します。



2-2 ツールバー上のラジオボタンを選択し、フォーム上の任意の場所をクリックして、ラジオボタンを 3 つ配置します。



2-3 フォーム上の 3 つのラジオボタンをグループ化指定します。3 つのラジオボタンを選択状態にして、[コントロール(T)]-[グループ指定(Z)]を選択します。



Question

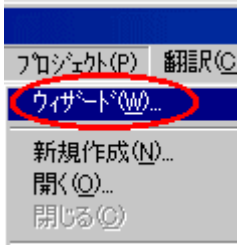
FB63002
コントロールを使ったプログラムの作成例

Answer

次にコントロールを使ったプログラムの作成例を紹介します。

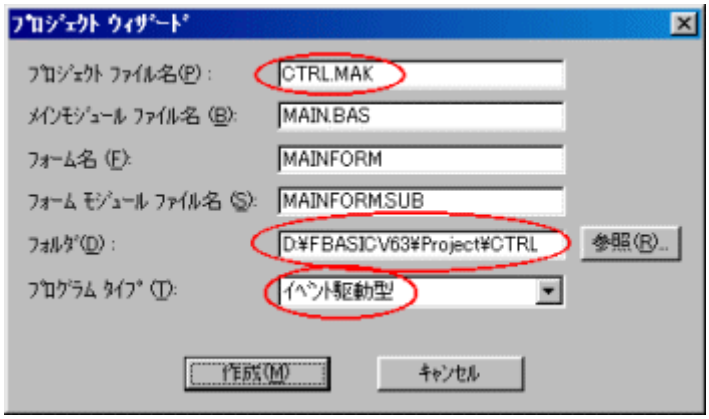
STEP1 プロジェクトを作成し、リソースファイルを登録する。

1-1 [プロジェクト(P)]-[ウィザード(W)...]を選択してプロジェクトを作成します。

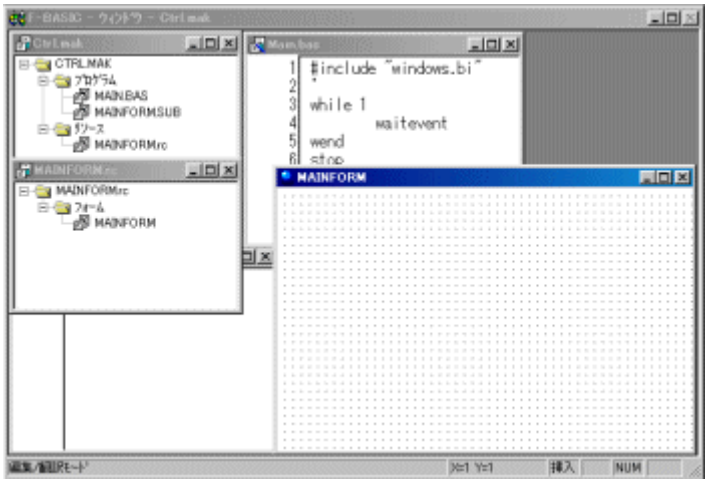


1-2 フォルダとプロジェクトファイル名を指定して、プロジェクトタイプが"イベント駆動型"になっていることを確認して、[作成(M)]ボタンを押します。

(例: CTRL1.MAK、D:¥FBASICV63¥PROJECT¥CTRL1)。



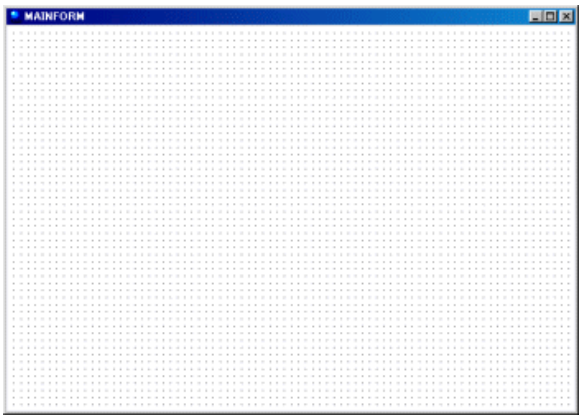
1-3 プロジェクトファイル(CTRL1.MAK)、プログラムファイル(MAIN.BAS)、プログラムファイル(MAINFORM.SUB)、リソースファイル(MAINFORM.RC)が作成されます。



※ ウィザードはプログラムファイルとリソースファイルの登録を自動的に行います。ウィザードを使用していない場合には、プログラムファイルとリソースファイルの登録を手作業で行う必要があります。

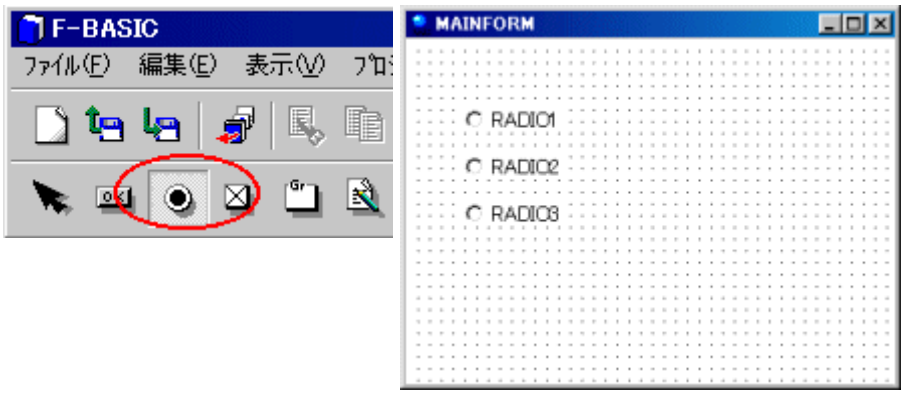
STEP2 フォーム上にコントロールを配置する。

2-1 フォームを前面に表示します。

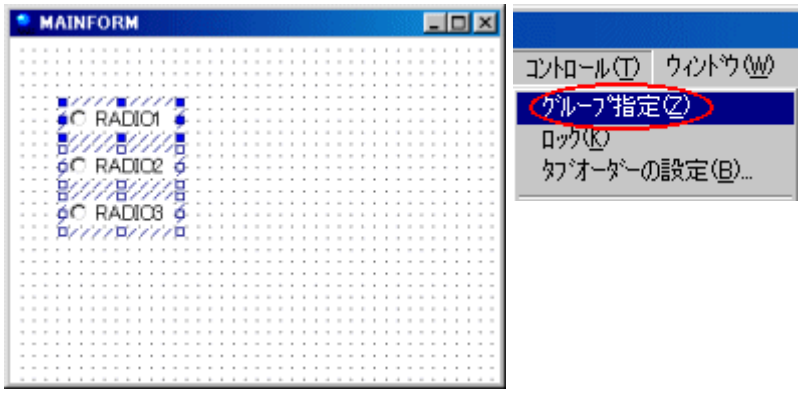


2-2 ツールバー上のラジオボタンを選択し、フォーム上の任意の場所をクリッ

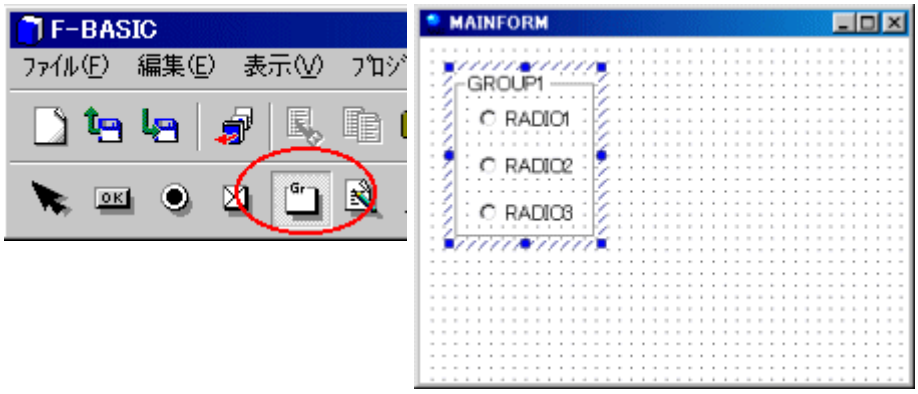
クして、ラジオボタンを 3 つ配置します。



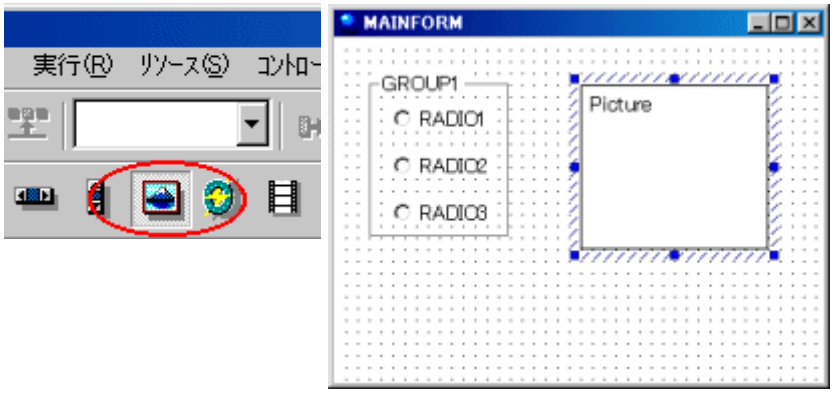
2-3 フォーム上の 3 つのラジオボタンをグループ化指定します。3 つのラジオボタンを選択状態にして、[コントロール(T)]-[グループ指定(Z)]を選択します。



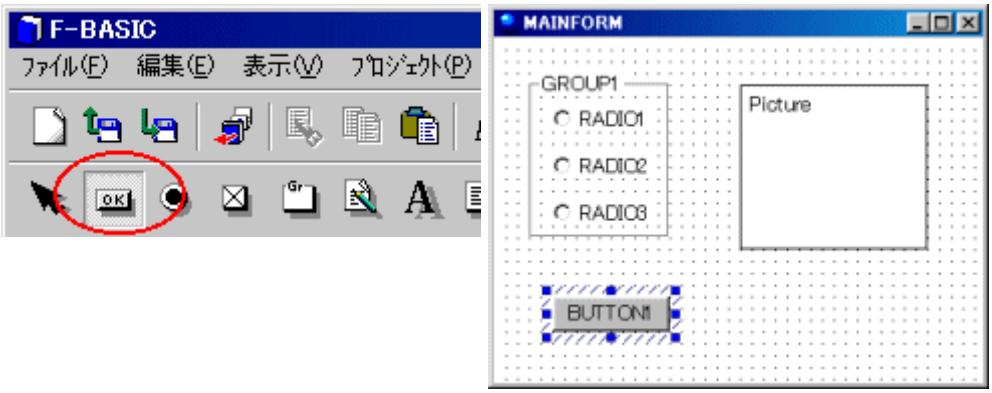
2-2 ツールバー上のグループボックスを選択し、フォーム上のラジオボタンを配置した場所をクリックして、グループボックスを配置し、3 つのラジオボタンが収まる大きさに調節します。



2-2 ツールバー上のピクチャボックスを選択し、フォーム上の任意の場所をクリックして、ピクチャボックスを配置し、任意のサイズに調節します。

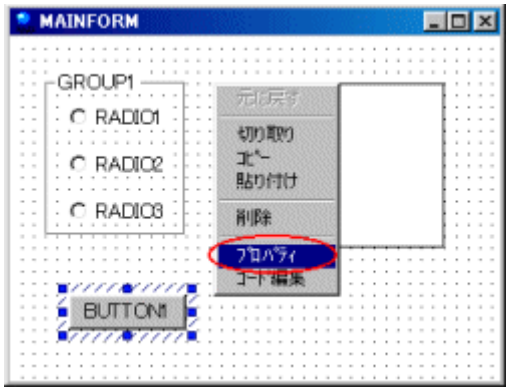


2-3 ツールバー上のコマンドボタンを選択し、フォーム上の任意の場所をクリックして、コマンドボタンを配置します。



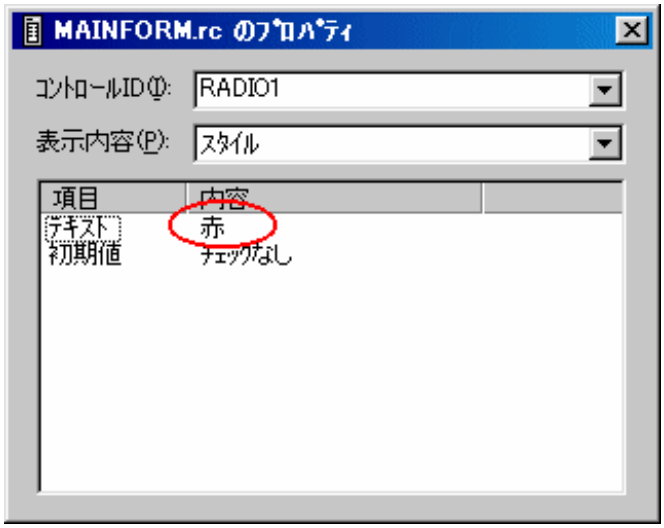
STEP3 コントロールのプロパティを設定する。

3-1 MAINFORM の画面上を右クリックして[プロパティ]を選択して、プロパティウィンドウを表示します。



3-2 [コントロール ID]欄から"RADIO1"を選択します。

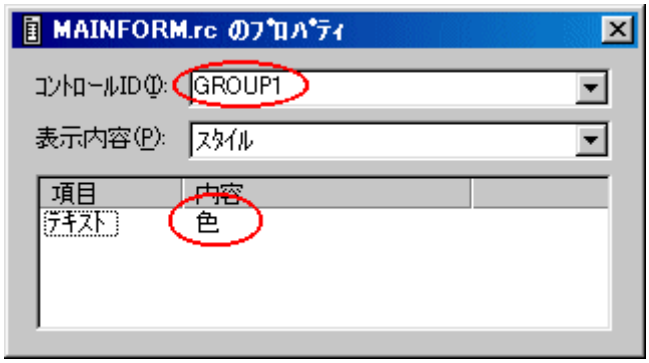
3-3 "テキスト"の項目に書かれている"RADIO1"をクリックします。"RADIO1"の文字列が反転し、キー入力可能状態となりますので、"赤"とキー入力して、Enter キーを押下します。



3-4 同様に、"RADIO2"、"RADIO3"のラジオボタンのテキストも、それぞれ"緑"、"青"に変更します。

3-5 [コントロール ID]欄から"GROUP1"を選択します。

3-6 "テキスト"の項目に書かれている文字列を"GROUP1"から"色"に変更します。



STEP5 MAIN.BAS にイベントチェックルーチンを記述する。

5-1 プログラム中でボタンが押されたかどうかをチェックする個所に WAITEVENT 命令を記述します。

※ WAITEVENT 命令は拡張命令なので、使用する場合にはプログラムの先頭に#include "WINDOWS.BI"を記述します

※ ウィザードを使用した場合には自動的にプログラムの先頭にイベントチェックルーチンが挿入されます。

MAIN.BAS

```
#include "windows.bi"
'
while1
    WAITEVENT
wend
'
stop
end
```

STEP4 各コントロールにオブジェクト変数を割り当てる。

4-1 MAIN.BAS に次のように太字部分を追記します。

MAIN.BAS

```
#include "windows.bi"
'
'オブジェクト変数を宣言
common shared RADIO1 as object
common shared RADIO2 as object
common shared RADIO3 as object
common shared PICT1 as object
'
'オブジェクト変数に各コントロール ID を結び付ける
RADIO1.ATTACH GETDLGITEM("RADIO1")
RADIO2.ATTACH GETDLGITEM("RADIO2")
RADIO3.ATTACH GETDLGITEM("RADIO3")
PICT1.ATTACH GETDLGITEM("PICTURE1")
'
while 1
    WAITEVENT
wend
'
stop
end
```

STEP6 イベントルーチンを記述する。

6-1 ボタンコントロールを右クリックして[プロパティ]を選択して、プロパティウィンドウを表示します。

6-2 [表示内容]欄から"イベント"を選択して、"実行"の項目をダブルクリックすると、MAINFORM.SUB 内に、BUTTON1 の実行のイベントルーチンが挿入されます。

6-3 BUTTON1 が押されたときに実行したい命令を sub BUTTON1_ON()と end sub の間に記述します。次の太字部分を追記してください。それぞれの命令・関数については、リファレンスを参考にしてください。

※ MAINFORM.SUB 内でも拡張命令を使用する場合には、プログラムの先頭に#include "windows.bi"を記述します。
※ オブジェクト変数を MAIN.BAS と MAINFORM.SUB の間で共有する場合には、MAINFORM.SUB でも変数の宣言をします。

MAINFORM.SUB

```
#include "windows.bi"
'
common shared RADIO1 as object
common shared RADIO2 as object
common shared RADIO3 as object
common shared PICT1 as object
'
'=====================================
'                                     =
'=====================================
declare sub BUTTON1_ON edecl ()
sub BUTTON1_ON()
    if RADIO1.GETCHECK=1 then
      IRO=RGB(255,0,0)
    else if RADIO2.GETCHECK=1 then
      IRO=RGB(0,255,0)
    else if RADIO3.GETCHECK=1 then
      IRO=RGB(0,0,255)
    else
      IRO=RGB(0,0,0)
    endif
    PICT1.SETBACKCOLOR IRO
    PICT1.cls
end sub
```

STEP7 翻訳する。

7-1 メニューから[翻訳(C)]-[すべて翻訳(A)]を選択してプログラムを翻訳します。

＜プログラムの例＞

MAIN.BAS

```
#include "windows.bi"
'
'オブジェクト変数を宣言
common shared RADIO1 as object
common shared RADIO2 as object
common shared RADIO3 as object
common shared PICT1 as object
'
'オブジェクト変数に各コントロール ID を結び付ける
RADIO1.ATTACH GETDLGITEM("RADIO1")
RADIO2.ATTACH GETDLGITEM("RADIO2")
RADIO3.ATTACH GETDLGITEM("RADIO3")
PICT1.ATTACH GETDLGITEM("PICTURE1")
'
while 1
    WAITEVENT
wend
'
stop
end
```

MAINFORM.SUB

```
#include "windows.bi"
'
common shared RADIO1 as object
common shared RADIO2 as object
common shared RADIO3 as object
common shared PICT1 as object
'
'=====================================
'                                    =
'=====================================
declare sub BUTTON1_ON edecl ()
sub BUTTON1_ON()
    if RADIO1.GETCHECK=1 then
      IRO=RGB(255,0,0)
```

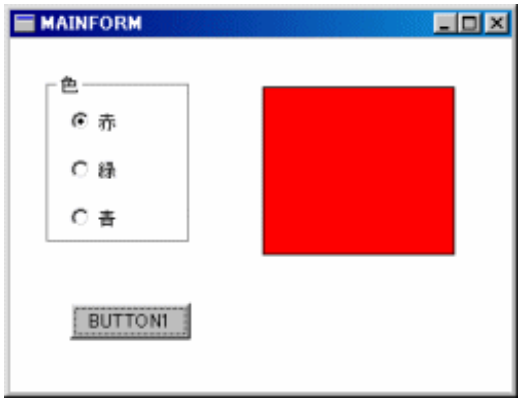
```
    else if RADIO2.GETCHECK=1 then
      IRO=RGB(0,255,0)
    else if RADIO3.GETCHECK=1 then
      IRO=RGB(0,0,255)
    else
      IRO=RGB(0,0,0)
    endif
    PICT1.SETBACKCOLOR IRO
    PICT1.cls
end sub
```

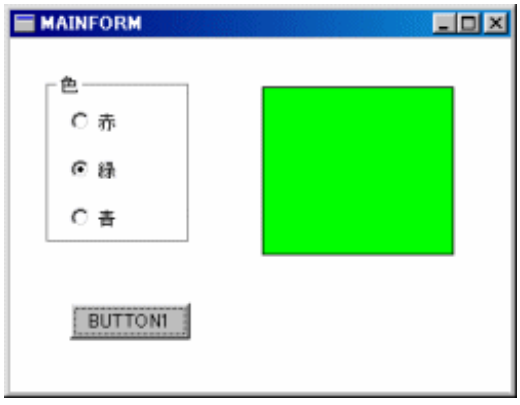
MAINFORM.RC

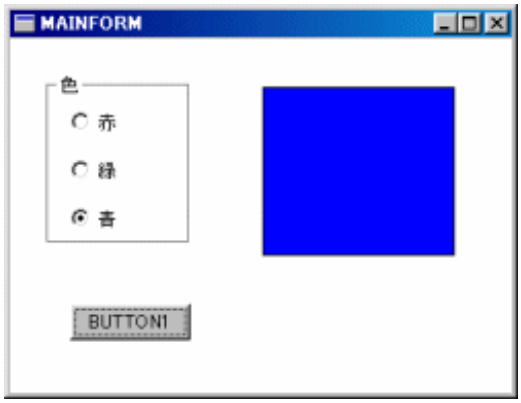
(RADIO1 のプロパティ)
テキスト 赤
(RADIO2 のプロパティ)
テキスト 緑
(RADIO3 のプロパティ)
テキスト 青
(GROUP1 のプロパティ)
テキスト 色
(BUTTON1 のプロパティ)
実行 BUTTON1_ON

プログラム内容
BUTTON1 を押すと選択されているラジオボタンの色でピクチャボックスの中が塗り潰される。

実行画面







＜イベント＞

Question

FB64001
リターンキーで入力のフォーカスが移動するようにできますか？

Answer

できません。フォーカスの移動はタブキーで行うようにしてください。

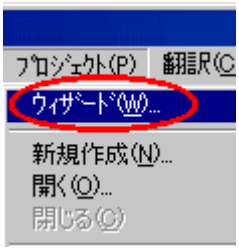
Question

FB64002
イベントを使ったプログラムの作成例

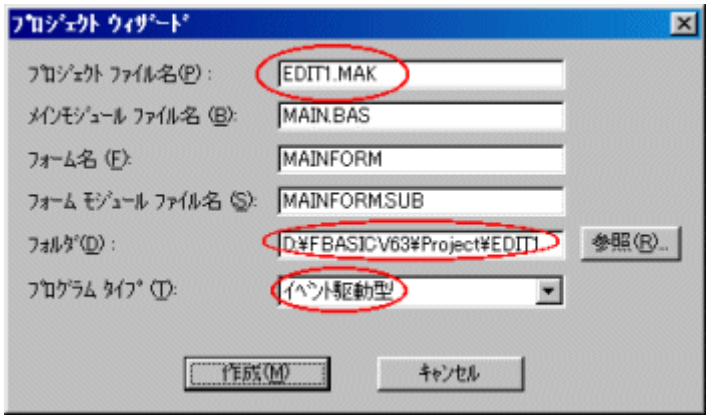
Answer

イベントを使ったプログラムの作成例を紹介します。

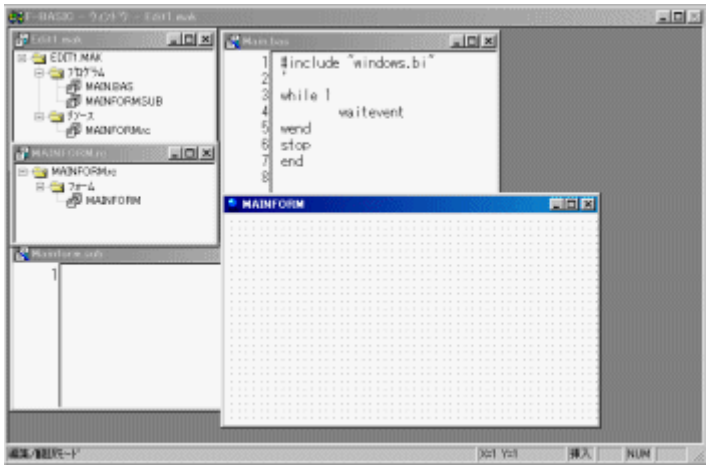
STEP1 プロジェクトを作成し、リソースファイルを登録する。
1-1 [プロジェクト(P)]-[ウィザード(W)...]を選択してプロジェクトを作成します。



1-2 フォルダとプロジェクトファイル名を指定して、プロジェクトタイプが"イベント駆動型"になっていることを確認して、[作成(M)]ボタンを押します。(例: EVENT1.MAK、D:¥FBASICV63¥PROJECT¥EVENT1)。



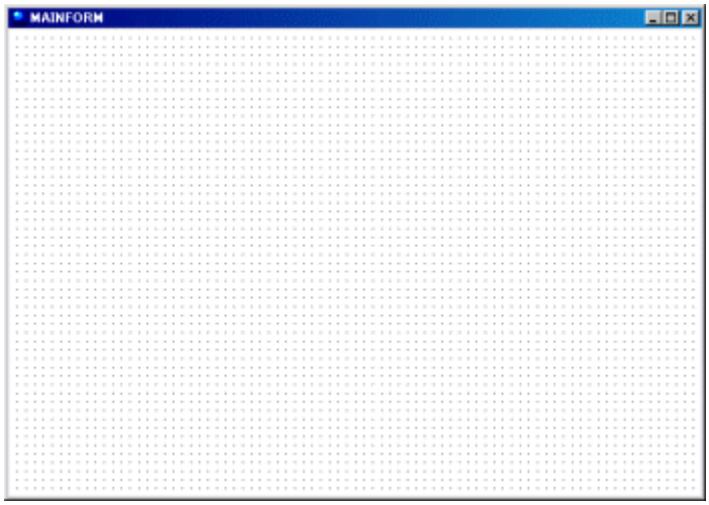
1-3 プロジェクトファイル(EVENT1.MAK)、プログラムファイル(MAIN.BAS)、プログラムファイル(MAINFORM.SUB)、リソースファイル(MAINFORM.RC)が作成されます。



※ ウィザードはプログラムファイルとリソースファイルの登録を自動的に行ないます。ウィザードを使用していない場合には、プログラムファイルとリソースファイルの登録を手作業で行なう必要があります。

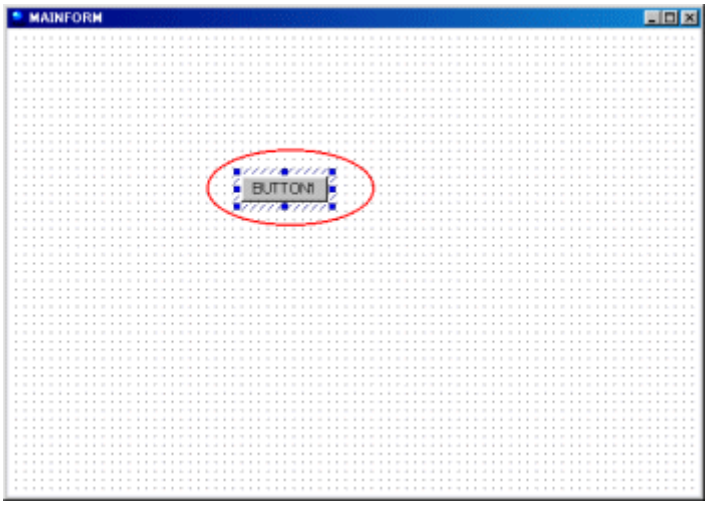
STEP2 フォーム上にコントロールを配置する。

2-1 フォームを前面に表示します。



2-2 ツールバー上のコマンドボタンを選択します。

2-3 フォーム上の任意の場所をクリックして、コマンドボタンを配置します。



STEP3 イベントルーチン名を設定する。

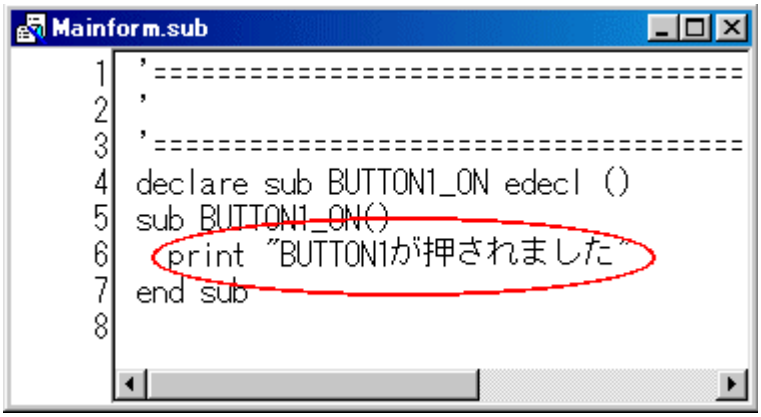
3-1 ボタンを右クリックして[プロパティ]を選択して、プロパティウィンドウを表示します。

3-2 [表示内容]欄から"イベント"を選択して、"実行"の項目に書かれている内容がイベントルーチンの名前になります。イベントルーチン名を変更する場合にはここで編集します。

STEP4 イベントルーチンを記述する。

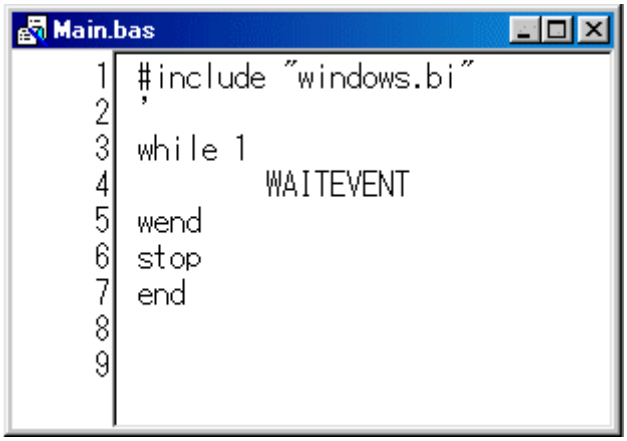
4-1 [表示内容]欄から"イベント"を選択して、"実行"の項目をダブルクリックすると、MAINFORM.SUB 内に、BUTTON1 の実行のイベントルーチンが挿入されます。

4-2 BUTTON1 が押されたときに実行したい命令を sub BUTTON1_ON() と end sub の間に記述します。例では、「BUTTON1 が押されました」と実行画面に表示させる処理を書きこんでいます。



STEP5 イベントチェックルーチンを記述する。

5-1 プログラム中でボタンが押されたかどうかをチェックする個所に WAITEVENT 命令を記述します。



※ WAITEVENT 命令は拡張命令なので、使用する場合にはプログラムの先頭に #include "WINDOWS.BI" を記述します
※ ウィザードを使用した場合には自動的にプログラムの先頭にイベントチェックルーチンが挿入されます。

STEP6 翻訳する。
6-1 メニューから[翻訳(C)]-[すべて翻訳(A)]を選択してプログラムを翻訳します。

＜プログラムの例＞

MAIN.BAS

```
#include "WINDOWS.BI"      '拡張命令を有効にする
do
  WAITEVENT
loop
end
```

MAINFORM.SUB

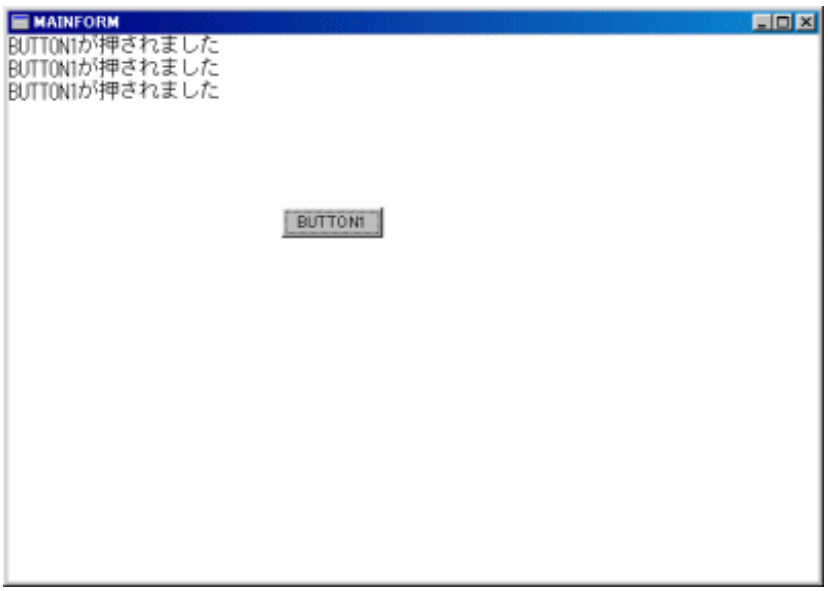
```
declare sub BUTTON1_ON edecl()
sub  BUTTON1_ON()
  print "BUTTON1 が押されました"
end sub
```

MAINFORM.RC

```
(BUTTON1 のプロパティ)
実行 BUTTON1_ON
```

プログラム内容
BUTTON1 を押すと、"BUTTON1 が押されました"と表示します。

実行画面



＜メニュー＞

Question

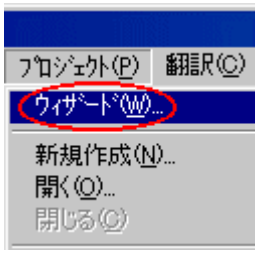
FB65001
メニューを使ったプログラムの作成例

Answer

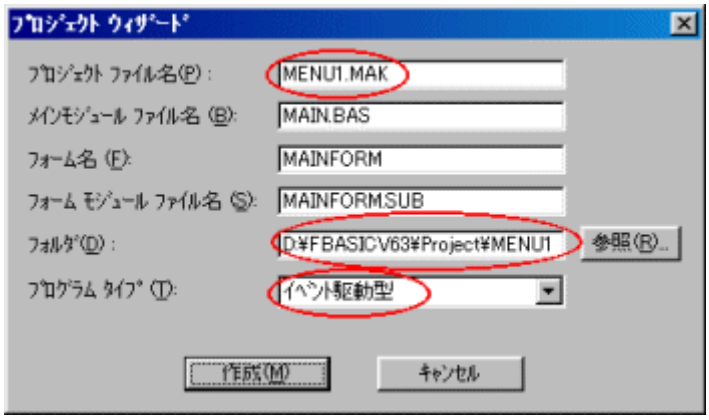
ここでは、例として次のようなメニューを表示するプログラムを作成する手順を紹介します。

STEP1 プロジェクトを作成し、リソースファイルを登録する。

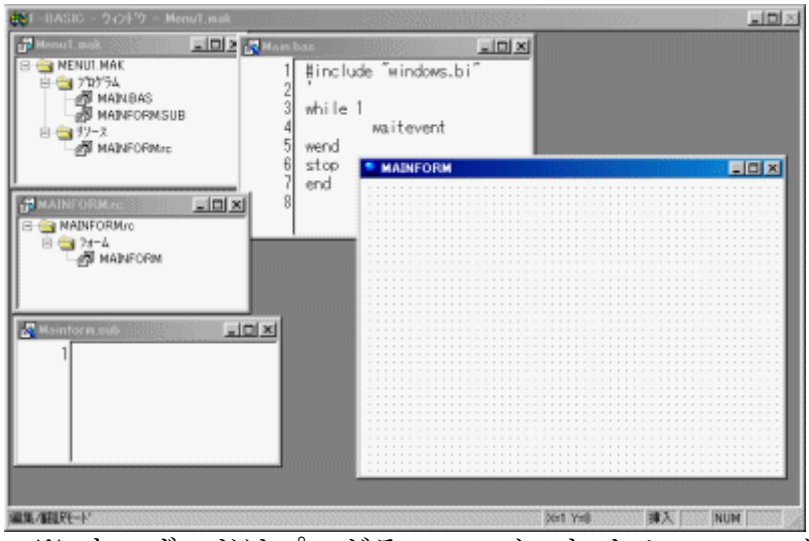
1-1 [プロジェクト(P)]-[ウィザード(W)...]を選択してプロジェクトを作成します。



1-2 フォルダとプロジェクトファイル名を指定して、プロジェクトタイプが"イベント駆動型"になっていることを確認して、[作成(M)]ボタンを押します。(例: MENU1.MAK、D:¥FBASICV63¥PROJECT¥MENU1)。



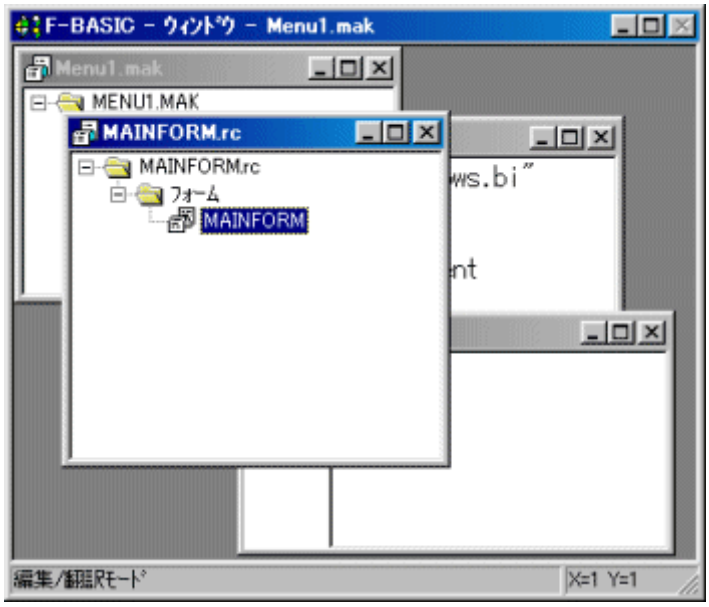
1-3 プロジェクトファイル(MENU1.MAK)、プログラムファイル(MAIN.BAS)、プログラムファイル(MAINFORM.SUB)、リソースファイル(MAINFORM.RC)が作成されます。



※ ウィザードはプログラムファイルとリソースファイルの登録を自動的に行ないます。ウィザードを使用していない場合には、プログラムファイルとリソースファイルの登録を手作業で行なう必要があります。

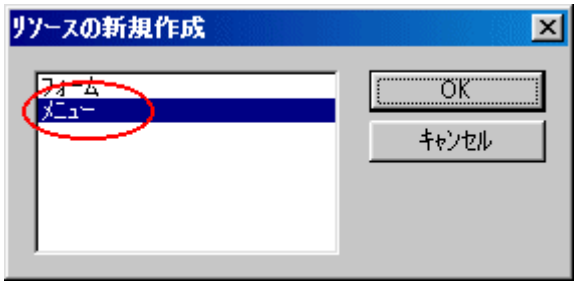
STEP2 メニューを作成する。

2-1 リソースファイルを前面に表示します。

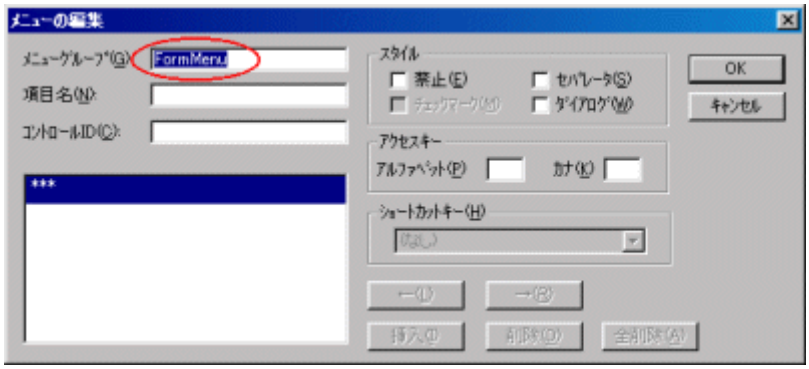


2-2 メニューから[リソース]-[新規作成]を選択します。

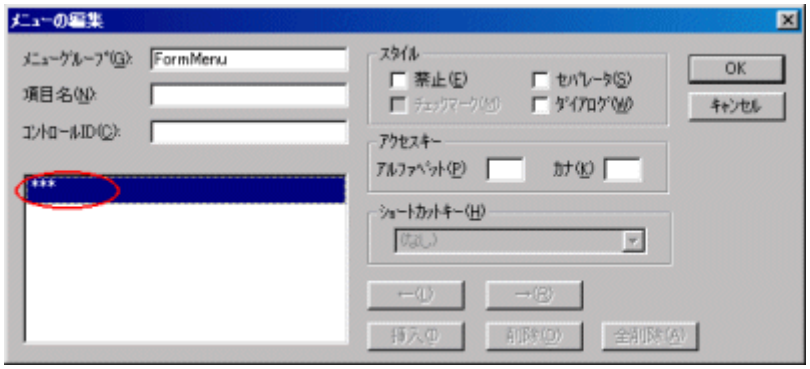
2-3 "メニュー"を選択して、[OK]ボタンを押します。



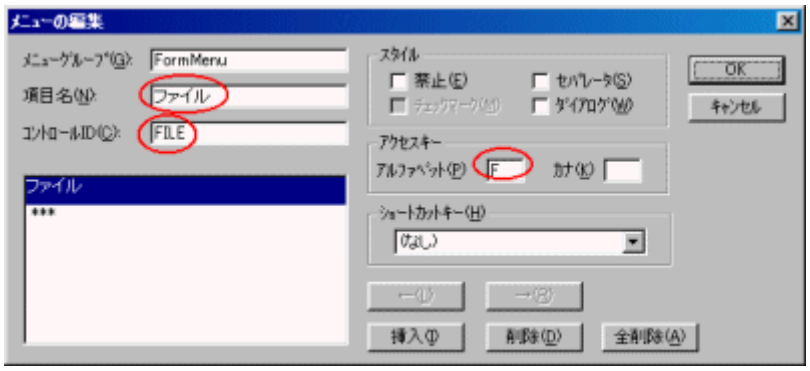
2-4 メニューグループ名を指定します。(例: FORMMENU)



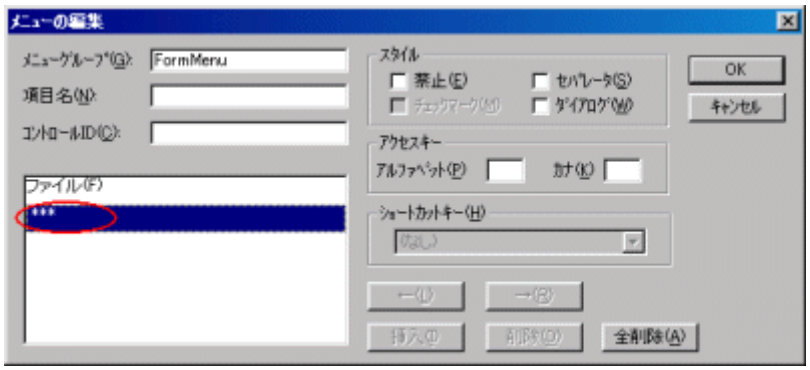
2-5 [ファイル(F)]の挿入位置をリストボックスから選択します。ここではメニューの末尾に挿入するので"***"を選択します。



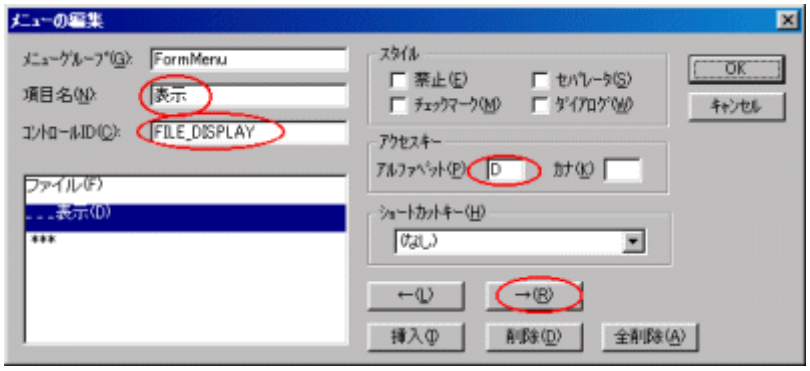
2-6 [項目名]欄に"ファイル"、[コントロール ID]欄に"FILE"、[アクセスキー]欄に"F"を入力します。



2-7 [表示(D)]の挿入位置をリストボックスから選択します。ここでは[ファイル(F)]の直後に挿入するので"***"を選択します。

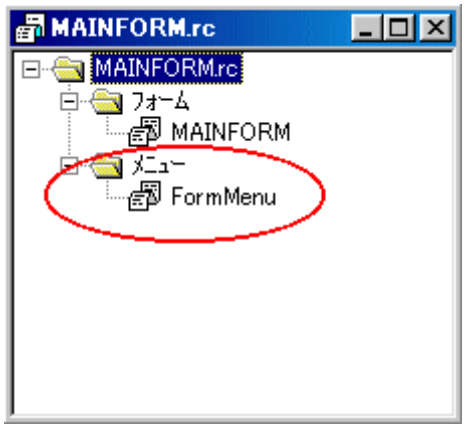


2-8 [項目名]欄に"表示"、[コントロール ID]欄に"FILE_DISPLAY"、[アクセスキー]欄に"D"を入力して、[→(R)]ボタンを押します。



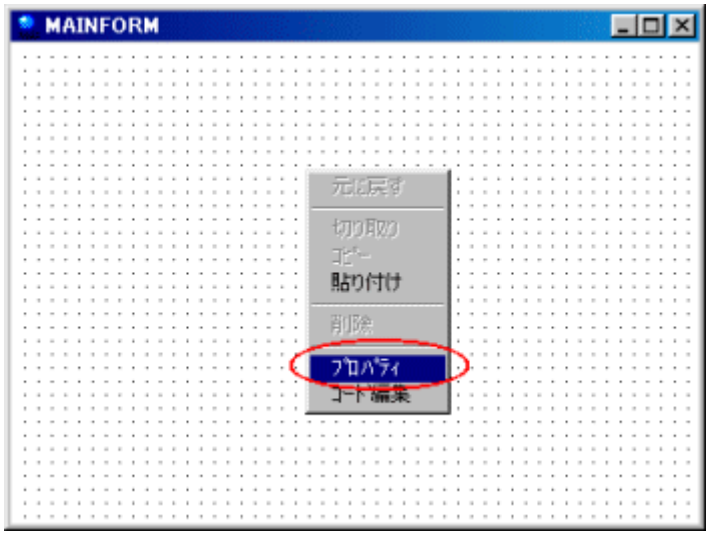
2-9 [OK]ボタンを押します。

2-10 作成したメニューがリソースファイルに追加されます。

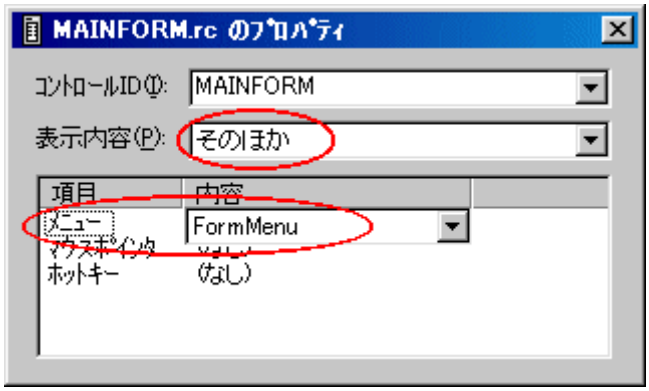


STEP3 フォーム上にメニューを配置する。

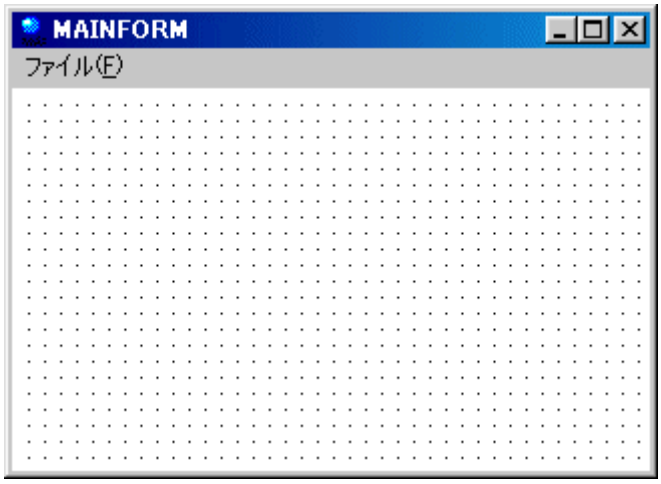
3-1 フォームを右クリックして[プロパティ]を選択して、プロパティウィンドウを表示します。



3-2 [コントロール ID]欄から"MAINFORM"、[表示内容]欄から"そのほか"を選択して、"メニュー"の項目から"FORMMENU"を選択してください。

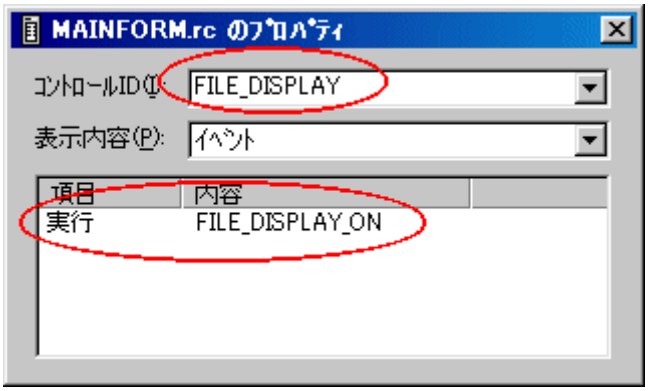


3-3 フォームにメニューが表示されます。



STEP4 イベントルーチン名を設定する。

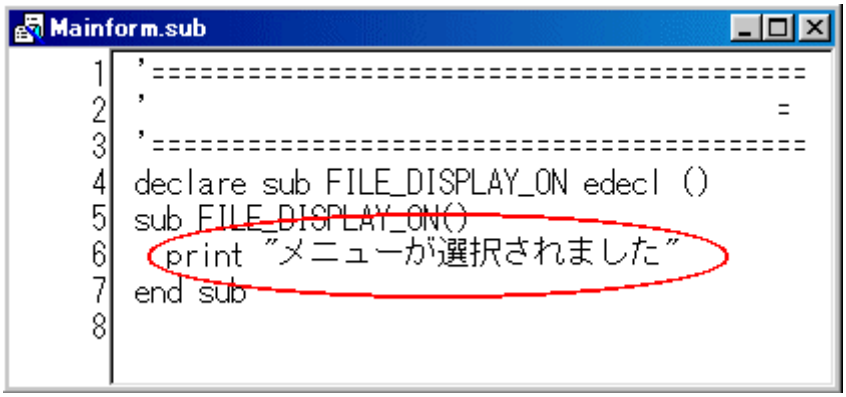
4-1 プロパティウィンドウの[コントロール ID]欄から"FILE_DISPLAY"、[表示内容]欄から"イベント"を選択して、"実行"の項目にイベントルーチン名を指定します。(例:"FILE_DISPLAY_ON")



STEP5 イベントルーチンを記述する。

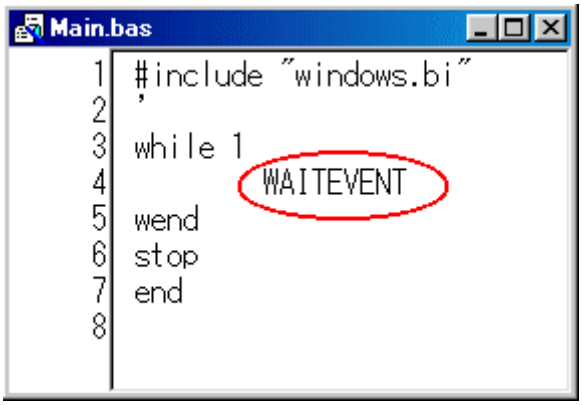
5-1 プロパティウィンドウの[コントロール ID]欄から"FILE_DISPLAY"、[表示内容]欄から"イベント"を選択して、"実行"の項目をダブルクリックすると、MAINFORM.SUB 内に、[ファイル(F)]-[表示(D)]が選択されたときに実行されるイベントルーチンが挿入されます。

5-2 [ファイル(F)]-[表示(D)]が選択されたときに実行したい命令を sub FILE_DISPLAY_ON() と end sub の間に記述します。例では、「メニューが選択されました」と表示する処理を記述しています。



STEP6 イベントチェックルーチンを記述する。

6-1 プログラム中でメニューが選択されたかどうかのチェックを行なう個所に WAITEVENT 命令を記述します。

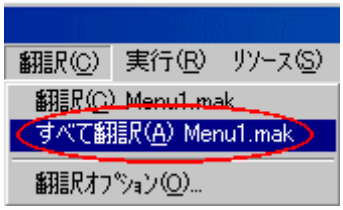


※ WAITEVENT 命令は拡張命令なので、使用する場合にはプログラムの先頭に#include "WINDOWS.BI" を記述します。

※ ウィザードを使用した場合には自動的にプログラムの先頭にイベントチェックルーチンが挿入されます。

STEP7 翻訳する。

7-1 メニューから[翻訳(C)]-[すべて翻訳(A)]を選択してプログラムを翻訳します。



＜プログラムの例＞

MAIN.BAS

```
#include "WINDOWS.BI"      '拡張命令を有効にする。
do
  WAITEVENT                'イベントのチェック。
loop

end
```

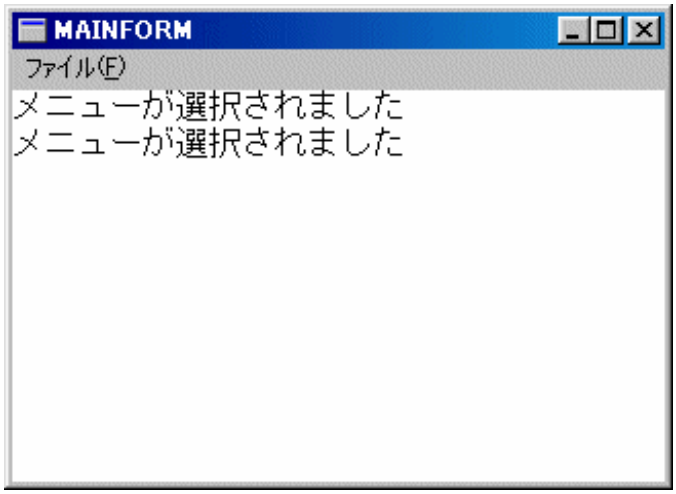
MAINFORM.SUB

```
declare sub FILE_DISPLAY_ON edecl()
sub  FILE_DISPLAY_ON()
  print "メニューが選択されました"
end sub
```

MAINFORM.RC

```
(MAINFORM のプロパティ)
メニュー  FORMMENU
(FILE_DISPLAY のプロパティ)
実行  FILE_DISPLAY_ON
(FORMMENU の内容)
ファイル  コントロール ID="FILE"
---表示  コントロール ID="FILE_DISPLAY"
```

プログラム内容
[ファイル(F)]-[表示(D)]を選択すると、"メニューが選択されました"と表示します。
実行結果



＜独立型プログラムの作り方＞

Question

FB67001
デバッガで独立型プログラムを実行できないのですか？

Answer

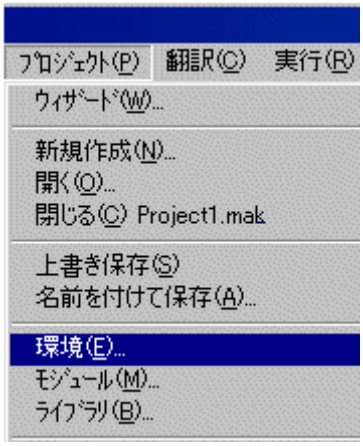
できません。デバッグモードが実行可能なのはランタイム型（デバッグ用）のみです。

Question

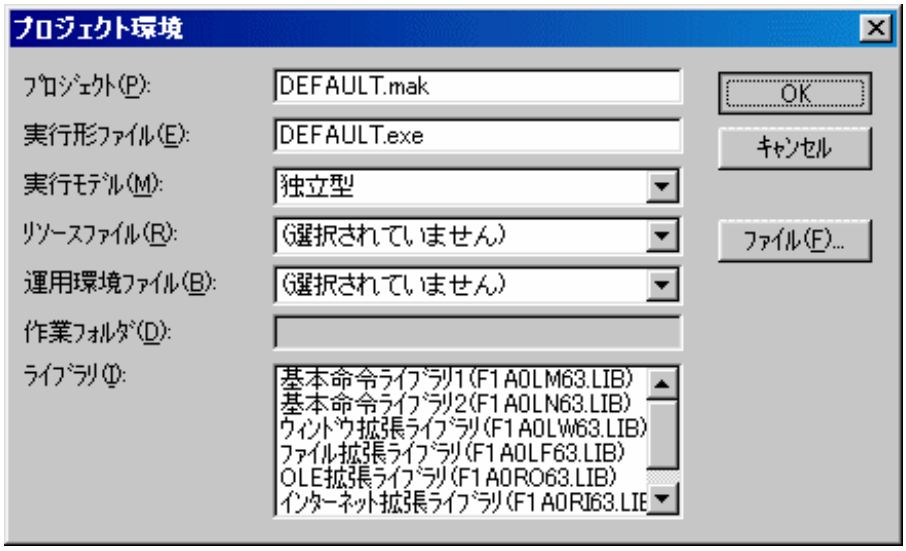
FB67002
独立型プログラムの作成方法は？

Answer

独立実行型で翻訳し実行ファイルを作成しますと、必要なランタイムライブラリを含めた実行ファイルを作成するため、ランタイムライブラリの配布が不要となります。従って実行ファイルのファイルサイズはランタイム実行型よりは大きくなります。
次に独立実行型プログラムを作成手順を示します。
（1） F-BASIC 上でプログラムソースを開きます。
（2） [プロジェクト]メニューの[環境]コマンドを選択します。



（3） [実行モデル]で「独立型」を選択し、[OK]ボタンを押下します。



（4） 翻訳します。

※OLE 機能やインターネット機能を利用したプログラムは独立実行型では実行ファイルは作成できません。ランタイム型で作成するようにしてください。

＜プログラミング（上級）＞

＜データベース連携＞

Question

FB70001
Microsoft Access で作成したデータベースとの連携は可能ですか？

Answer

ODBC を使うことにより、データの内容を参照したり、データの内容を編集したりすることが可能です。
F-BASIC の製品に添付されているサンプルプログラムの ODBC フォルダ内(C:¥FBASICV63¥Sample_B¥ODBC)にある ODBC.txt に ODBC の設定方法が掲載されています。

※フォルダ名の例は、F-BASIC V6.3 が C ドライブにインストールされている場合です。

Question

FB70002
F-ISAM との連携は可能ですか？F-ISAM との連携方法に関する情報はどこにあるのですか？

Answer

F-ISAM との連携は可能です。F-BASIC の製品に添付されているサンプルを参考にしてください。また、F-ISAM との連携方法に関するリファレンス情報は、サンプルと同じフォルダ内(C:¥Fbasicv63¥Sample_B¥ISAM)にある「ISAM.HLP」に掲載されています。
※フォルダ名は、F-BASIC V6.3 が C ドライブにインストールされている場合です。

Question

FB70003
MS-ISAM との互換性はありますか？

Answer

ありません。

Question

FB70004
Btrieve との連携は可能ですか？

Answer

可能です。F-BASIC の製品に添付されているサンプルプログラム(C:¥Fbasicv63¥Sample_B¥Btrieve)を参考にしてください。
※フォルダ名は、F-BASIC V6.3 が C ドライブにインストールされている場合です。

＜機器制御（GP-IB など）＞

Question

FB72001
GP-IB ボードを制御することが可能ですか？また、その制御方法は？

Answer

ボードメーカの提供するダイナミックリンクライブラリ(*.DLL)を呼び出すことによって制御することができます。
下記のトピックを参考にしてください。

One Point Lesson 第六回 「GP-IB ボードの制御をしよう」	GP-IB ボードを制御するためのプログラムの基本的な流れや構造について説明しています。 動作確認済みのボードメーカへのリンクがあり、それぞれのボードを制御するためのサンプルプログラムがダウンロードできます。
RS-232C/GP-IB ボード活用ガイド	RS-232C や GP-IB インターフェースの制御について解説しています。 ガイドを PDF 形式の文書としてダウンロードできます。

Question

FB72002
動作確認済みの GP-IB ボードはありますか？

Answer

「ダウンロード」ページの「**GP-IB ボード制御サンプルプログラム：**」に動作確認済みのボードメーカの一覧とそれぞれのメーカへのリンクがあります。またサンプルプログラムのダウンロードもできます。

Question

FB72003
GP-IB 以外のボードを操作する方法は？

Answer

GP-IB 以外のボードでも同様に、プログラムからダイナミックリンクライブラリ（*.DLL）呼び出しによって制御します。ダイナミックリンクライブラリはボードに添付されている C 言語インタフェースのものを利用してください。ダイナミックリンクライブラリの呼び出しについては、F-BASIC の製品に添付されているマニュアルを参照してください。

＜その他＞

＜アイコン編集＞

Question

FB90001
PC をハイカラー以上のモードにして F-BASIC の「アイコン編集」を使用してアイコンファイルを作成した場合、作成時と保存して再度表示させたときの色が異なっていることがあるのですが？

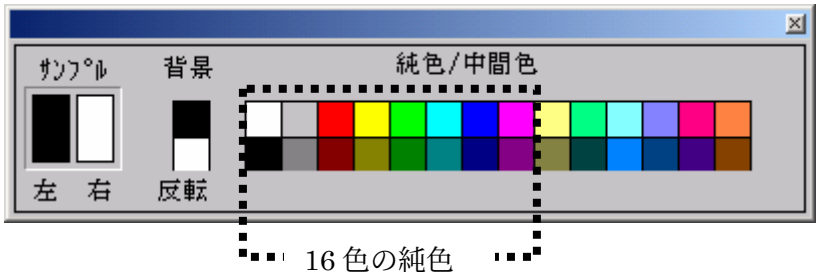
Answer

PC をハイカラー以上のモードにして F-BASIC の「アイコン編集」を使用した場合、カラーパレット画面には 28 色分の色が表示され、16 色の純色以外の色を使用して描画できます。しかし、16 色アイコンの作成を目的としているアイコンエディタのため、保存時に 16 色の減色処理が発生し、お使いいただく時の画面モードによっては、再度表示させたときに編集時と異なった色で表示される場合がございます。

アイコン編集のカラーパレット画面には常に 28 色分の色が表示されるようになっておりますが、PC のカラーモードによって以下のように使用できる色に制限があります。

PC のカラーモード	使用可能色
ハイカラー以上	16 の純色
16 色または 256 色	28 色（16 の純色以外は擬似色）

【カラーパレット画面】



◆使用上の注意事項：
PC をハイカラー以上にしてアイコン編集をご使用になる場合は、カラーパレットの 16 色の純色のみを使用し描画してください。256 色または 16 色モードで使用した場合は、カラーパレットの 16 色の純色以外を擬似色としてお使いいただけます。

Question

FB90002
他の画像データの一部を「コピー＆ペースト」によって F-BASIC の「アイコン編集」に貼り付けとき、元の画像データの色と異なって表示されることがあるのですが？

Answer

デジカメで撮影した写真など、多色の画像データを「コピー＆ペースト」によって F-BASIC のアイコン編集に貼り付けますと、F-BASIC の「アイコン編集」は 16 色へ減色を行って表示しますので、元の画像データとは色が異なって表示されることがあります。

F-BASIC の「アイコン編集」は 16 色アイコンの編集を前提としており、16 色を超える画像を「コピー＆ペースト」によって取り込む場合は結果を保証しておりません。

他の多色の画像データから「コピー＆ペースト」によって取り込む場合は、16 色への減色機能を持つ画像編集ソフトであらかじめ 16 色に減色してからご使用いただくよう、お願いいたします。