Coal

言語仕様書編

第９章

第　７．３　版

２０２３年５月２０日

目　　次

9. 付録

9.1. 予約語

以下が予約語である。

(1)コマンド関連

大文字小文字を問わない。

太文字は、太文字でない長さからその長さまでが予約語であることを示す。

AND	ARRAY	AS	BEXP	**BIN**ARY	BREAK	BULK
BYE	CALL	CASE	CATCH	CHAR	CLASS	CONST
CONTINUE	DATE	DBL	**DEC**IMAL	DEF	DEFAULT	DEFINE
DEFVAR	DIM	DO	DOUBLE	DUMP	EACH	ECHO
ELSE	ELSEIF	ELSEL	ELSIF	END	ENDCLASS	ENDDO
ENDFOR	ENDFUNC	ENDFUNCTION	ENDIF	ENDLOOP	ENDPROC	ENDSUB
ENDSW	ENDSWITCH	ENDTRY	ENDUNTIL	ENDWHILE	EP	EXCEPTION
EXEC	EXIT	EXPORT	EXTENDS	FINALLY	FLOAT	FLT
FOR	FUNC	FUNCTION	GLOBAL	IF	IMPORT	IN
INTEGER	**INTER**ACTIVE	IP	LABEL	LEAVE	LET	LNG
LOCAL	LOGPARM	LONG	LOOP	LPRINT	LPRINTF	**MAPPED**ARRAY
MESSAGE	NEXT	NODE_DEFINE	NODE_IMPORT	NODE_SCRIPT	ON	OPTION
OPTIONS	OR	OUTPUT	PRAGMA	PRINT	PRINTF	PRIVATE
PROC	PUBLIC	QUIT	RAISE	READ	REDEF	REDEFINE
RETURN	SAY	SC	SET	SHIFT	SLEEP	SM
SQL	STATIC	STEP	STRING	SUB	SW	SWITCH
THEN	THROW	TO	TRY	TYPE	TYPEDEF	**UINT**EGER
ULNG	ULONG	UNDEF	UNDEFINE	UNKNOWN	UNTIL	VAR
VARIANT	WHILE					

(2)システム変数

先頭にドル記号を付けてもシステム変数となる。

「5.5 システム変数を参照」

9.2. 実行時オプション

オプションには、セッション間オプションとセッション内オプションがある。
セッション間オプションは、Coal起動時に設定される。
セッション内オプションは、スクリプト開始時に、セッション間オプションがコピーされる。

表９．２－１　実行時オプション（１／５）

オプション番号	オプション値の意味(デフォルト値は、0x00)
1	変数値未設定時の処理 0x01 ： 配列要素のとき、NULL値とする。 0x02 ： 配列以外の変数のとき、NULL値とする。 これを設定すると、自動で0x01が設定される。 0x04 ： 連想配列要素を未設定にする。(0x01 優先) 0x10 ： 変数が未定義のとき、エラーとする。
2	演算に関するオプション 0x01 ： 比較時の両辺属性が同じかをチェックする。 0x02 ： 文字列の加減乗除算には文字演算を行う。 0x04 ： 数学関数(SQRT,CBRT,LOG,LOG10)は、２進浮動小数点数で実行する。 0x10 ： 10進浮動小数点数の精度の上限を100としない。 0x20 ： 比較、concatでの配列、リスト等の階層展開の上限を100としない 0x40 ： 整数演算でオーバーフローをチェックしない 0x80 ： 複素数の演算結果を虚数または実数にしない。 0x100 ： 範囲指定またはCOMPLEX()でゼロがあれば虚数または実数にする。 0x200 ： 数値関数実行で、errno<>0のときエラー終了する。 0x400 ： 範囲値または複素数は、先頭の値のみ使用する。
3	テキスト入出力時の改行コードを制御する 0x00 ： LFを出力する 0x01 ： LFを出力しない 0x02 ： CRを出力する 0x04 ： テキスト中の改行コードも制御する 0x08 ： モードの't'を有効にする(windows環境では、Lf-->CrLf) 0x10 ： 入力時に改行コードのLFへの変換をしない。 0x20 ： 行末の改行コードを削除しない。 0x40 ： ＣＳＶ形式で読み込む(２重引用符中の改行コード以降も読み込む)。 0x80 ： ＣＳＶ形式で読み込み、２重引用符中の改行コードを削除する。 0x100 ： １重引用符も２重引用符と同じに扱う 0x200 ：'¥n'と'¥t'以外の非表示文字を<コードの10進数>で出力する。 0x400 ：`(バッククォート)によるコマンド実行の出力の末尾の改行を削除する。
4	式がないときの処理 0x00 ： ワーニング終了(ret=100)、エラーメッセージあり 0x01 ： エラーメッセージなしビット 0x02 ： 正常終了ビット 0x04 ： エラー終了ビット
5	要素なしのリストの扱い 0x00 ： ＮＵＬＬ値とする 0x01 ： NULLリストとする
6	インポートするスクリプトの追加位置を指定する 0x00 ： 末尾に追加する 0x01 ： 先頭に挿入する

表９．２－１　実行時オプション（２／５）

オプション番号	オプション値の意味(デフォルト値は、0x00)
７	エラー時の処理 0x00 ： 当該手続きをエラー終了する 0x01 ： エラー行出力を行わない 0x02 ： エラーを無視して処理を継続する 0x04 ： 当該セッションをエラー終了する 0x08 ： TRYモードを下位スクリプトに波及させる 0x10 ： ″end of file″エラーメッセージを出力する (TRYモード時は常に出力する) 0x20 ： 入力系関数でエラーのとき、以降の関数実行をスキップし、 正常終了する。 0x40 ： 入力系関数でエラーのとき、処理を中止し式の値をNULL文字で返し、 正常終了する。 0x80 ： シェル実行でエラーのとき、処理を中止する。 0x0100 ： system()を実行後復帰しない環境のときでも実行する。 0x0200 ： 同じ行のメッセージを毎回出力する。 0x0400 ： EXECコマンドで実行開始前にエラーがあってもエラーとしない ($ERRORはセット)。 0xX000 ： 同じ行のメッセージをX回まで出力する。(X=0は、10と見なす)
８	ユーザ定義関数のサーチ順位 0x00 ： 定義済み関数より先にサーチする。 0x01 ： 定義済み関数の後にサーチする。 0x02 ： 手続き／関数の入れ子をサポートする。 クラスを使用するときは、自動で設定される。 0x04 ： 手続き／関数の大文字小文字を区別しない。
９	入力構文の文字コードその他 0x00 ： 引用符外側の英数記号の倍角文字は半角文字に変換する。 0x01 ： 全ての英数記号の倍角文字は半角文字に変換する。 0x02 ： 全て変換しない。 0x04 ： １重引用符(')と２重引用符(″)の機能を交換する。 0x10 ： 文字コード変換にiconv()を使用しない。 0x20 ： 文字列中″¥″があるかどうかチェックしない。 チェックする場合は、″¥″があるときは、iconvを使用しない。
１０	デフォルトの変数種別 0x00 ： ローカル変数とする。 0x01 ： スクリプト変数とする。
１１	１０進浮動(固定)小数点数の０表示 0x00 ： 少数点以下がないときに少数点以下を表示しない。 0x01 ： １未満のときに最初の０を表示しない。 0x02 ： 少数点以下がないときに少数点を表示する。 0x04 ： 少数点以下がないときに少数第１位の０を表示する。 0x20 ： ３桁毎にカンマを付ける。 0x40 ： 指数表示にする。 0x80 ： １０進固定を無効にし、１０進浮動とする。 0x0100 ： 指数表示にするとき、右側を0サプレスする。 0x0200 ： 表示有効桁数の指定がないときに、表示有効桁数を16にしない。
１２	１０進固定小数点数の位取り未満の処理 0x00 ： 四捨五入 0x01 ： 切捨て 0x02 ： 切上げ

表９．２－１　実行時オプション（３／５）

オプション番号	オプション値の意味(デフォルト値は、0x00)
１３	ＰＲＩＮＴ文での配列、構造体出力形式 0x00 ： 詳細情報を出力しない 0x01 ： 詳細情報を出力する 0x02 ： (F)PUTLINE()で、'/'から始まる文字列をオプションと見なす。 '¥¥'でエスケープ。 0x04 ： データをカンマで区切る。 0x08 ： 文字データを２重引用符で囲む。 0x10 ： データの前に ″変数名=″ を出力する。 0x20 ： ３桁毎にカンマを付ける。 0x40 ： [F]PUTLINE()またはEEDIT,[F]PRINTFの書式が″%s″/″%S″で 一般変数でない１つの式はエラーにする。 0x80 ： [F]PUTLINE()またはEEDIT,[F]PRINTFの書式が″%s″で 一般変数でない１つの式の出力結果を″<>″で囲まない。 0x100 ： FPUTLINE()を使って出力する。 0x200 ： リストを'[]'で囲まない。 0x400 ： 文字コードを出力する。
１４	メッセージの言語種別 ０ ： 日本語 １ ： 英語 ２ ： その他
１５	配列の開始インデックスとデータの配置 0x00 ： 開始インデックスは、０ データの配置は、C言語形式 0x01 ： 開始インデックスは、１ 0x02 ： データの配置は、FORTRAN形式
１６	１０進浮動(固定)小数点数のオーバーフロー、アンダーフロー時の処理 数学関数のアンダーフロー時の処理 0x00 ： オーバーフロー時、エラー終了。メッセージを出力する。 アンダーフロー時、続行。 0x01 ： オーバーフロー時、続行。 0x02 ： オーバーフロー時、メッセージを出力する。 0x04 ： アンダーフロー時、エラー終了。(数学関数を含む) 0x08 ： アンダーフロー時、メッセージを出力する。(数学関数を含む) 0x10 ： (scale処理)１０進固定小数点数のオーバーフロー時、数値は そのままとする。
１７	１０進浮動小数点数になる場合の数字列の変換 0x00 ： １０進浮動小数点数に変換 0x01 ： ２進浮動小数点数に変換 0x02 ： =0:Lなしはint、Lありはlongとする。=1:longとする。 0x10 ： 数字列の後ろに数字以外があってもエラーとしない。 0x20 ： カンマを無視する。 0x40 ： 複数符号を許す。 0x80 ： 整数値が正/負でオーバーフローするときは、正/負の最大値を 設定する。

表９．２－１　実行時オプション（４／５）

オプション番号	オプション値の意味(デフォルト値は、0x00)
１８	サーチ開始位置指定時の返却値 0x00 ： INSTR,INLIKE,REPLIKEで、指定文字列の先頭を基準とする。 0x01 ： INSTRにおいて、サーチ開始位置を基準とする。 0x02 ： INLIKE,INREGEXにおいて、サーチ開始位置を基準とする。 0x04 ： REPLIKE,REPREGEXにおいて、サーチ開始位置以降を返却する。
１９	デフォルトのDATE_FORMAT 0x00 ： SQL_DATE_FORMAT 0x01 ： UNIX_DATE_FORMAT
２０	１０進浮動(固定)小数点数を指数表示する場合の桁数 (1)１０進浮動(固定)小数点数の場合 m*256 + n n ： 仮数桁数（1～54） m ： 指数桁数（1～5）デフォルト値=2 これを指定すると以下の固定長になる。 -1.23456...E+12.. 1.23456...E-12.. (2)２進浮動小数点数の場合 n n ： 有効桁数（1～17） フォーマットは、"%.ng"
２１	スクリプト/入力/出力の日本語コード 0xX4X3X2X1 or D4.D3.D2.D1 or 10進 形式で設定 X1～X4 ： 16進 2桁 D1～D4 ： 10進 0～255 4バイト整数の各1バイト 省略は0と見なす (D4～D1は0xXXの16進形式でも良い) X1,D1:出力データ（上4ビット:ファイル、下4ビット:標準出力/エラー） X2,D2:入力データ（上4ビット:ファイル、下4ビット:標準入力） X3,D3:スクリプト X4,D4:予約 0 ： デフォルト(スクリプト=S-JIS/入力=UTF-8/出力=UTF-8) 1 ： EUC 2 ： S-JIS 3 ： JIS 4 ： EBCDIC 5 ： UTF-8 6 ： UCS-4
２２	配列の操作 0x00 ： 0x01 ： データ設定時の個数オーバーでワーニングを出力する。
２３	外部変数定義と代入 0x00 ： ２回目以降の定義時は、エラーにしない。定義情報は、無視する。 0x01 ： ２回目以降の再定義時は、エラーにする。

表９．２－１　実行時オプション（５／５）

オプション番号	オプション値の意味(デフォルト値は、0x00)
２４	ＩＦ文等での式の処理 0x00 ： ＡＮＤ、ＯＲで、全ての式の評価を行う。 0x01 ： ＡＮＤ、ＯＲで、真偽確定時に以降の式の評価を行わない。 0x10 ： DEFINEで、複数変数を定義するとき、変数毎の初期値設定を可能としない。
２５	リダイレクト 0x0010 ： HEREDOCの入力が終端文字列に達しとき、または、HERERDOC中のルーチンが終了してもファイルをクローズしない。 0x0020 ： HEREDOCの入力行の前方のタブを削除する。 0x0040 ： HEREDOCの入力行の前方の半角スペースを削除する。 0x0080 ： シェル実行時に、標準エラー出力を標準出力にする。 0x01XX ： シェルのコマンド実行の最初の入力待ち時間(0xXX)を秒単位で指定する。XXが0のときは、タイムアウトしない。 0x0200 ： SHELL()でstdoutがメモリにREDIRECTされていないときでもpopen()を実行する。 0x0400 ： debugログのリダイレクト設定を無効にしない。 0x0800 ： HEREDOCをファイルから入力する。
２６	判定関数をオプティマイズしない 0x00 ： オプティマイズする。 0x01 ： IIF() 0x02 ： NVAL() 0x04 ： NULLIF() 0x08 ： NSVAL() 0x10 ： NDEF() 0x20 ： AND() 0x40 ： OR()
２７	ループ回数オーバー処理 0x00 ： $MAX_LOOP_WHILEが変更されていないときのみループ回数オーバーのメッセージを出力する。 0x01 ： ループ回数オーバー時には、必ずメッセージを出力する。 0x10 ： ループ回数オーバーをチェックしない。（上限をINT_MAXとする）
２８	
２９	
３０	オプション指定の無視 0x80000000 がオフのとき有効 0x7fffffff ： スクリプト内でのオプション指定について、ビット位置に対応するオプション番号の指定ををを無視する。
３０	ループコマンドの有効化（coal_mini専用） 0x80000000 がオンのとき有効 0x00 ： loop のみが有効 0x01 ： while,until　　を有効にする 0x02 ： for (;;)　　を有効にする 0x04 ： for to step　　を有効にする 0x08 ： for each　　を有効にする 0x10 ： do while,until を有効にする

9.3. データ属性の値

データＩＤおよびデータ型の値を以下に示す。

表11.4-1　データＩＤ

項番	データＩＤ	値（文字）	値（整数）	データ長(バイト)
1	一般データ	’ ’	３２	テータ型による
2	配列名	一般配列　:’R’ MAPPED配列:’A’	’R’：８２ ’A’：６５	３２
3	リスト	’L’	７６	３２
4	データ並び	’N’	７８	３２
5	構造体名	’T’	８４	３２
6	構造体定義名	’P’	８０	３２
7	関数名	’F’	７０	３２
8	クラス名	’C’	６７	３２
9	インスタンス名	’I’	７３	３２
１０	メソッド名	’M’	７７	３２
１１	手続き名	’O’	７９	３２
１２	名前指定	’E’	６９	３２

表11.4-2　一般データのデータ型とデータ長

項番	データ型	値	データ長(バイト)
1	文字	1	実データ長
2	整数	2	4
3	２進倍精度浮動小数点数	3	8
4	１０進小数点数	4	４０
5	ｂｕｌｋ	5	実データ長
6	日付	6	４０
7	バリアント(注)	7	0

(注)データ未設定時のみ存在し、データ設定後は、そのデータ型になる。

日付型は、14バイトから成る。各バイトのデータ内容を以下に示す。

表11.4-3　日付型のデータ内容

Index	意味	値
0	年の上２桁	0 - 99
1	年の下２桁	0 - 99
2	月	0 - 11
3	日	1 - 31
4	時	0 - 23
5	分	0 - 59
6	秒	0 - 59
7から９	マイクロ秒(intの下３バイト)	0 - 999999
１０	ＡＭ／ＰＭ	0 / 1
１１	曜日	0 - 6 (0:日曜)
１２，１３	年通算日数(short)	0 - 365 (0:１月１日)

9.4. 演算子の優先順位と結合順序

表１１．５－１　演算子の優先順位と結合順序

順位	演算子	結合順序
１	．（ ）　［ ］　｛ ｝　後置演算＋＋　－－	左から右
２	＊＊	左から右
３	単項演算子　！　～　－　＋　＋＋　－－　＊	右から左
４	キャスト	左から右
５	＊　／　％	左から右
６	＋　－	左から右
７	＜＜　＞＞	左から右
８	文字列演算子　CONCAT　SUBSTR　REP　CONDAS　TO　IS	左から右
９	＜　＞　＜＝　＞＝	左から右
１０	＝＝　！＝	左から右
１１	＆	左から右
１２	＾	左から右
１３	｜	左から右
１４	＆＆	左から右
１５	｜｜	左から右
１６	？：	右から左
１７	..	左から右
１８	<== ==>	左から右
１９	＝ += -= *= /= %= <<= >>= &= ^= |= &+= ~=	右から左
２９	,	左から右

（注）ピリオドはドット式で使われるとき。

9.5. 変数、手続きまたは関数のサーチ順

（1）変数のサーチ順

スコープ指定がないときは、以下の順にサーチする。

ローカル変数 → スクリプト変数 → 外部変数

手続きまたは関数の入れ子をサポートするときは、ローカル変数は、以下でサーチする。

実行済みの手続きまたは関数を実行の逆順に構造上の上位に向かってサーチする。、

（2）手続きまたは関数のサーチ順

（A）手続きまたは関数の入れ子をサポートしないとき

スクリプトの先頭から、第一レベルにある手続きまたは関数をサーチする。

（B）手続きまたは関数の入れ子をサポートするとき

手続きまたは関数を呼び出している手続きまたは関数の中 →上位の手続きまたは関数の中 → その上位の手続きまたは関数の中 ・・・

（C）初に通常のスクリプト内をサーチし、見つからなかったときは、インポートされたスクリプト内をサーチする。

9.6. デバッグ形式での出力

（1）一般形式

（A）内容

ログヘッダ	出力元が設定したデータ	情報構造体の内容

データＩＤまたはデータ型による個別情報出力

（B）ログヘッダ

標準出力形式

[coal]

ファイル出力形式

yyyy/mm/dd hh:mm:ss coal/<ソースファイル名>(<ソース行数>):

(注)<>は、これで囲まれた情報が出力されることを示す。

（C）出力元が設定したデータ

出力対象となった最上位の変数のとき

式=

（2）個別情報

（A）式の並びまたはデータ並び式

並びの最後の式の値を一般形式で出力し、次に、並びの情報を″ PARMINFO2:″を「出力元が設定したデータ」として一般形式で出力する。

（B）配列

情報構造体の内容

```
pInfo=%08x id=%c attr=%d scale=0x%02x dlen=%d len(pScCT)=%08x hlen(gid)=%d
 aux=0x%02x 0x%02x auxlen=%04x %08x data=%08x
```

配列の情報

```
(VarIndex=%08x Attr=%d %d %d %d size=%d index=%d[%d,%d,%d] xhp=%08x
  (_xhash Id=%c%c keylen=%d max=%d pre=%d ha=%08x next=%08x datlen=%d dreg=%08x))
 varnam=[配列名]
```

(注)"(_xhash ・・・ dreg=%08x)"は、連想配列のとき出力する。

（C）その他

「情報構造体の内容」は共通で、その後に個別情報が出力される。

情報構造体の内容

```
pInfo=%08x id=0x%02x[%c] attr=%d scale=0x%02x code=%d dlen=%d len(gid)=%d hlen=%d pos
=%d(%08x) aux=0x%02x 0x%02x auxlen=%04x %08x data=%08x
```

（a）構造体または構造体定義

ヘッダ

```
size=%d data=0x%08x ntype=%d vname=0x%08x pType=0x%08x varnam=[<オブジェクト名>]
```

メンバー情報（メンバー数分出力する）

```
"vnlen=<メンバー名長> <メンバー名>: "を「出力元が設定したデータ」として、一般形式で出
力
```

（b）リスト

```
 rb_bfsize=%d rb_max=%d rb_num=%d rb_used=%d rb_pos=%d rb_raddr=0x%08x rb_waddr=0x%08
x rb_wpriv=0x%08x rb_cur=0x%08x
```

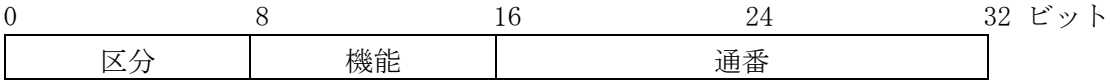
（c）その他

データ型	単純	範囲
データへのポインタがNULL	NULL	なし
CHAR	[%s]	[%s]..[%s]
BIN	%d	%d..%d
FLOAT	%e	%e..%e
DEC	10進表示	10進表示..10進表示
BULK または PARMINFO2の２要素目	先頭32バイトの16進表示	なし
その他	**INVALID**	なし

9.7. 例外の種類

9.7.1. 例外番号の構成

（1）値の構成



（2）名称の構成

区分名称__[機能名称__[個別名称__]]ＥＸＣＥＰＴＩＯＮ

9.7.2. 区分

No.	区分		内容
	名　　称	値	
1	ＩＳ	1	検査
2	ＴＯ	2	変換
3	ＣＭＰＲ	3	比較
4	ＳＴＲＩＮＧ	4	文字列操作
5	ＭＡＴＨ	5	数値演算
6	ＦＩＬＥ	6	ファイル
7	ＬＯＧ	7	ログ
8	ＳＱＬ	8	ＳＱＬ
9	ＣＯＭＭ	9	通信
10	ＵＳＥＲ	0x7c	ユーザ
11	ＳＹＳＴＥＭ	0x7d	システム
12	ＥＴＣ	0x7e	その他
13	ＡＬＬ	0x80	全ての例外を表す。 例外の値は、これとのＯＲを取った値となる。

9.7.3. 機能

（5）ＭＡＴＨ

No.	機能		個別		内容
	名　　称	値	名　　称	値	
	ＣＯＭＰ	1	［ ＥＲＲＯＲ ］	0	演算エラー全般
			ＤＥＶＩＤＥ	1	０割
	ＥＴＣ	254	［ ＥＲＲＯＲ ］	0	その他エラー全般

（6）ＦＩＬＥ

No.	機能		個別		内容
	名　　称	値	名　　称	値	
	ＯＰＥＮ	1	［ ＥＲＲＯＲ ］	0	オープン・エラー全般
			ＮＯＴＦＯＵＮＤ	2	file not found
	ＣＬＯＳＥ	2	［ ＥＲＲＯＲ ］	0	クローズ・エラー全般
	ＲＥＡＤ	3	［ ＥＲＲＯＲ ］	0	入力エラー全般
			ＥＮＤ	1	end of file
	ＷＲＩＴＥ	4	［ ＥＲＲＯＲ ］	0	出力エラー全般
	ＥＴＣ	254	［ ＥＲＲＯＲ ］	0	その他エラー全般

9.7.4. 例外番号

No.	区分	例外番号名	値
1		ALL_EXCEPTION	0x80000000
2		IS_EXCEPTION	ALL_EXCEPTION | 0x01000000
3		TO_EXCEPTION	ALL_EXCEPTION | 0x02000000
4		CMPR_EXCEPTION	ALL_EXCEPTION | 0x03000000
5		MATH_EXCEPTION	ALL_EXCEPTION | 0x04000000
6		MATH_COMP_EXCEPTION	MATH_EXCEPTION | 0x010000
7		MATH_COMP_ERROR_EXCEPTION	MATH_COMP_EXCEPTION
8		MATH_COMP_DEVIDE_EXCEPTION	MATH_COMP_EXCEPTION | 0x0001
9		MATH_ETC_EXCEPTION	MATH_EXCEPTION | 0xfe0000
10		MATH_ETC_ERROR_EXCEPTION	MATH_ETC_EXCEPTION
11		STRING_EXCEPTION	ALL_EXCEPTION | 0x05000000
12		FILE_EXCEPTION	ALL_EXCEPTION | 0x06000000
13		FILE_OPEN_EXCEPTION	FILE_EXCEPTION | 0x010000
14		FILE_OPEN_ERROR_EXCEPTION	FILE_OPEN_EXCEPTION
15		FILE_OPEN_NOTFOUND_EXCEPTION	FILE_OPEN_EXCEPTION | 0x0002
16		FILE_CLOSE_EXCEPTION	FILE_EXCEPTION | 0x020000
17		FILE_CLOSE_ERROR_EXCEPTION	FILE_CLOSE_EXCEPTION
18		FILE_READ_EXCEPTION	FILE_EXCEPTION | 0x030000
19		FILE_READ_ERROR_EXCEPTION	FILE_READ_EXCEPTION
20		FILE_READ_END_EXCEPTION	FILE_READ_EXCEPTION | 0x0001
21		FILE_WRITE_EXCEPTION	FILE_EXCEPTION | 0x040000
22		FILE_WRITE_ERROR_EXCEPTION	FILE_WRITE_EXCEPTION
23		FILE_ETC_EXCEPTION	FILE_EXCEPTION | 0xfe0000
24		FILE_ETC_ERROR_EXCEPTION	FILE_ETC_EXCEPTION

25		LOG_EXCEPTION	ALL_EXCEPTION | 0x07000000
26		SQL_EXCEPTION	ALL_EXCEPTION | 0x08000000
27		COMM_EXCEPTION	ALL_EXCEPTION | 0x09000000
28		USER_EXCEPTION	ALL_EXCEPTION | 0x7c000000
29		SYSTEM_EXCEPTION	ALL_EXCEPTION | 0x7d000000
30		ETC_EXCEPTION	ALL_EXCEPTION | 0x7e000000

9.8. ＳＱＬコマンドの処理番号

表１１．１－１に示す。

表１１．１－１　ＳＱＬコマンドの処理番号

項番	処理番号	処　理　内　容
1	0	何もしない
2	1	START TRANSACTION（更新開始）　を発行する
3	2	COMMIT WORK　（実更新）　を発行する
4	3	ROLLBACK WORK　（更新の取消）　を発行する
5	4	何もしない
6	5	START TRANSACTION（更新開始）　を強制発行する
7	6	COMMIT WORK　（実更新）　を強制発行する
8	7	ROLLBACK WORK　（更新の取消）　を強制発行する

9.9. スクリプトの例

9.9.1. Hellow World

```
proc main;
        print 'Hellow World.';
        return 0;
end proc;
```

9.9.2. クロージャ

test_class3.cl

```
//
Class Counter;
        int i=0;
        f = _count;
        func Counter;
                return f;       // クラス変数を使って関数を返す。
        end func;
        func Counter(n);
                i = n;
                return f;
        end func;
        func Counter(x,y);
                i = x + y;
                return My._count;       // Myは、本クラスを示す。
        end func;
        func _count();
                return ++i;
        end func;
end class;

proc main;
        c = new(Counter);
        print c();
        print c();
        d = new(Counter,10);
        print d();
        print d();
        print c();
        x = new(Counter,100,1);
        print x();
        print x();
        return ERROR;
end proc;
```

実行結果

```
145:/home/coal/test>coal test_class3
c()=1
c()=2
d()=11
d()=12
c()=3
x()=102
x()=103
```

9.9.3. 構造体を使った擬似クラス

test_class3.cl

```
//
define type struct Greeter
         variant salute
        ,variant name           // variantは、データ型指定なしと同じ。
;
define var g as Greeter;        // define var g,h as Greeter;
define var h as Greeter;        // gとhは、まとめて定義可能。

func _new (g,name);             // gには、構造体への参照が渡される。
        g.name = name;
//      g.salute = _salute;             // 直接関数名を指定しても同じ。
        g.salute = (FUNC)'{func _salute;// {}で囲った文字列を関数化して指定可能。
        print My.name&'' World!!'';     // 文字定数中の引用符は２つ続けて指定する。
        return 0;
end func;}';
        return 0;
end func;
/*
func _salute;
        print My.name&' World!!';       // Myは、構造体で修飾したときのみその構造体を示す。
        return 0;
end func;
*/
proc main;
        _new(g,'Hellow');
        g.salute();
        _new(h,'Good night');
        h.salute();
        return ERROR;
end proc;
```

実行結果

```
146:/home/coal/test>coal test_struct_class
My.name&' World!!'="Hellow World!!"
My.name&' World!!'="Good night World!!"
```

(注)printは、対象となった式=結果の形式で出力する。(ただし、定数は値のみを出力する。)

9.10. 辞書登録

9.10.1. ヘボン式ローマ字変換用

以下の英単語が登録済み。

変換元：

"コード","アクション","フラグ","データ","レベル"
,"ファイル","パートナ","ベンダー","テスト","オーダ"
,"サービス","セット","プロジェクト","オープン","インターネット"
,"パック","ステータス","コンサルティング","ロック","バージョン"
,"シリーズ","カウント","ラック","メール","アドレス"
,"レンタル","ディスパッチャー","ライセンス","ロット","グループ"
,"プレ","システム","メッセージ","アサイン","ソリューション"
,"アプリケーション","パスワード"

変換先：

"cd","action","flg","data","level"
,"file","partner","vendor","test","order"
,"service","set","project","open","internet"
,"pack","status","consulting","lock","version"
,"series","count","rack","mail","address"
,"rental","dispatcher","lisence","rot","group"
,"pre","system","message","asign","solution"
,"application","password"