## Magic eDeveloper V10



# Magic eDeveloper V10 コーディングサンプル Version 2 (モデル編)



Enabling Business with Superior Technology

本書および添付サンプル(以下、本製品)の著作権は、マジックソフトウェアジャパン株式会社(MSJ)にあります。MSJ の書面による事前の許可なしでは、いかなる条件下でも、本製品のいかなる部分も、電子的、機械的、撮影、録音、その他のいかなる手段によっても、コピー、検索システムへの記憶、電送を行うことはできません。

本製品の内容につきましては、万全を期して作成していますが、万一誤りや不正確な記述があったとしても、MSE(Magic Software Enterprises Ltd.)および MSJ はいかなる責任、債務も負いません。本製品を使用した結果、または使用不可能な結果 生じた間接的、偶発的、副次的な損害(営利損失、業務中断、業務情報の損失などの損害も含む)に関し、事前に損害の可能性 が勧告されていた場合であっても、MSE および MSJ、その管理者、役員、従業員、代理人は、いかなる場合にも一切責任を負い ません。MSE および MSJ は、本製品の商業価値や特定の用途に対する適合性の保証を含め、明示的あるいは黙示的な保証は 一切していません。

本製品に記載の内容は、将来予告なしに変更することがあります。

サードパーティ各社商標の引用は、MSE および MSJ の製品に対する互換性に関しての情報提供のみを目的としてなされるものです。一般に、会社名、製品名は各社の商標または登録商標です。

本製品において、説明のためにサンプルとして引用されている会社名、製品名、住所、人物は、特に断り書きのないかぎり、すべて架空のものであり、実在のものについて言及するものではありません。

初版 2007年10月10日

マジックソフトウェア・ジャパン株式会社

目次

4モデル	
4.1 モデルとは?	6
4.2 モデルの定義	7
4.3 モデルの利用	9
4.4 モデルの継承メカニズム	
4.4.1 継承とは	
4.4.2 モデルと継承の例	
4.4.3 継承の解除	
4.4.4 継承とモデルの変更	
4.4.5 継承の解除と再継承の操作	
4.5 モデルとコンポーネント	
5アプリケーションデータのモデル	
5.1 命名規則	
5.2「項目」クラスのモデルで便利な設定	
5.2.1 書式	20
5.2.2 選択プログラム	
5.2.3DB カラム名	
5.2.4 範囲	21
5.3 データコントロール	
5.3.1 データコントロールとは	
5.3.2 データコントロールの例	23
5.3.3 データコントロールの利用 (開発時)	24
5.3.4 データコントロールの利用 (実行時)	25
6 共通モデル	
6.1 命名規則	
6.2 内部イベントのプッシュボタンモデル	
6.2.1 プッシュボタンモデルの設定	
6.2.2 プッシュボタンモデルのフォームへの配置	
6.3 プッシュボタンモデルを参照する項目モデル	
6.3.1 項目モデルの定義	
6.3.2 データからの継承	
6.3.3 プログラムでの利用	
6.3.4 実行時の動作	
6.3.5 クリックでパーク 特性	35
6.4 ユーザイベントのプッシュボタンモデル	
6.4.1 ユーザ定義イベント	

6.4.2 ユーザ定義イベントを参照するプッシュボタンモデル	37
6.4.3 ユーザ定義イベントのプッシュボタンモデルを参照する項目モデル	37
6.4.4 どんなユーザイベントを用意すべきか?	
6.5GUI 表示形式のモデル	40
6.6GUI 出力形式のモデル	43
6.7 その他のモデル	45
6.8 まとめ	

### <u>4 モデル</u>

V10 の **モデル** は、dbMAGIC V8 およびそれ以前では タイプ と呼ばれていたもので、モデルリポジトリ (旧バー ジョンでは、「タイプ辞書」)にまとめて定義します。モデルリポジトリは、データリポジトリ (ファイル辞書、テーブ ル辞書)、プログラムリポジトリ (プログラム辞書) と並んで、Magic のプロジェクト (アプリケーション)を構成する 主要要素であり、Magic の生産性と保守性の大きな源泉のひとつとなっていました。

本章では、最初に V10 でのモデルのメカニズムについて簡単に説明した後、サンプルアプリケーションでのモ デルリポジトリを例にとりながら、V10 でのモデルの機能を最大限生かすことにより、より生産性と保守性の高 いアプリケーションを作成する方法について学んでいきます。



### 4.1 モデルとは?

モデルとは、Magicのオブジェクトが継承することのできる一連の特性のことです。

旧バージョンでは、モデルは「タイプ」と呼ばれていて、データ項目に対するタイプしか定義することができませんでした。

Magic V9より、タイプリポジトリは モデルリポジトリ と呼ばれるようになり、モデルの対象も、データ項目のみな らず、GUI 部品 (フォーム、コントロール)、ヘルプなどに拡張されました。Magic V10 では、次のようなクラス (種別)のモデルがあります。そして、それぞれに 型 (細分類) が細分されています。



また設定可能な項目も詳細に及ぶようになったので、モデルを活用することにより、更なる生産性の向上を図ることができます。

モデルは、モデルリポジトリに定義します。一度、モデルがモデルリポジトリに定義されれば、アプリケーション 中で繰り返し使うことができます。

モデルを使用する主な理由は次のとおりです。

- 時間の節約 モデルを一度定義したら、何度でも使うことができます。
- メンテナンスの簡易化と修正の集中化 個別のオブジェクトにではなく、一つのモデルを修正するだけですみます。
- 一貫性 異なったテーブルにまたがってカラムのモデルが一致していることを保証します。

更に、同じく V9 から導入されたコンポーネント機能を使って、多くのプロジェクトで共通的に使われるモデルをコンポーネント化することにより、優れたメンテナンス性を得ることができます。

### 4.2 モデルの定義

モデルは、モデルリポジトリに定義します。下図は、サンプルアプリケーションのデータコンポーネント ED\_DATA のモデルリポジトリです。

デルリポジトリ	/特	き性シート	<mark>+</mark>	デルリポジトリ	
😹 ED_DATA – M	asic - 体験版				
7714(E) 編集(E) 項目特性 A=文字 (○分(C)) 全体(A) 日子74 (○分(C)) 全体(A) 日子74 (○分(C)) 全体(A) 日子74 (○分(C)) 全体(A) 日子74 (○分(C)) 全体(A) 日子74 (○) 全体(A) 日子74 日子75	表示(V) 7 <sup>1</sup> DY3 ま示(V) 7 <sup>1</sup> DY3 () () () () () () () () () ()	ch(P) オフジョン(Q) デバッグ   第 日 日 日   第 日 日 日   第 名前 日 日   第 名前 日 日   第 名前 日 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 第 第 日   第 日 日 日   第 担当者CD 日 日   15 担当者 日 16   16 担当者 日 17   18 連 第 年	<ul> <li>ペルフペH)</li> <li>マー・シース</li> <li>クラス</li> <li>コンラ F=項目</li> <li>F=項目</li> </ul>	<ul> <li>記 図 図 町 2</li> <li>型</li> <li>A=文字</li> <li>A=(1)</li> <li>A=(1)<th></th></li></ul>	
開発モード: ED_D	ATA			گ	▶ 広頓 挿入 //

画面の右半分がモデルリポジトリで、このプロジェクトに定義されているモデルが一覧で表示されます。ここには、モデル名とクラス、型、その他の情報が表示されます。

画面の左半分は 特性シート と呼ばれるもので、ここのモデルの特性を詳細に表示するものです。上図では、 「顧客 CD」のモデルの特性を表示しています。

別の例として、新しい空のプロジェクトに、モデルをひとつ定義してみましょう。このモデルは、

- 名前は「拠点コード」
- クラスは「項目」
- 型は「文字型」
- 書式は「3」
- 範囲は「1-199」

です。

モデルリポジトリは次のようになります。

デルの定義				_					
	書式、範 を設定	囲など	の特性			— <mark>名前、</mark>	フラス、型を	設定	
項目特性 № 数値	x	×	🐹 EF.	ルリボジトリ					×
区分(C)   至14(円	/		#   名	前,	クラス	型	Dailly"	公開名	>
	5-3-3	^	1 🖁	処点コード	F=項目	N=数值			
	17 7	=							
□ <b></b> / まて	3								
型	N=数値								
範囲	1-199								
□入力									
選択プログラム	0								
起動モード	B=前置								



モデルの定義は、次の手順で行います。

- 1. モデルリポジトリを開きます。(Shift+F1)
- ポップアップメニューで、「行作成(R)」を選択します。
   (あるいは F4 キーを押します)
   → 新規行が作成されます。
- 3. 名前、クラス、型を設定します。
- ポップアップメニューで「特性(R)」を選びます。(あるいは、Alt+Enterを押します)。
   →特性シートが開きます。
- 5. 特性シート上で、書式と範囲を設定します。

これで、「拠点コード」モデルの定義ができました。



### 4.3 モデルの利用

モデルの利用は、特性シートの「モデル」欄にモデルを設定することにより行います。

例えば、下図は、タスクの中で変数項目「V\_拠点コード」を定義し、「拠点コード」モデルを参照している例です。

特性シートの「モデル」 欄にモデルを設定する     項目の場合には、ここでモデル番号を指定することもできる。       アイルED 編集ED 表示(ソ) 712/20/ED タスが環境(S) オブッパの デバッグ(D) 40.7℃D     ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	モデルの参照(タスクの変数でモデル「拠点コード」を参照)
ModelSample - Magic - 体級版         77/////2       10/2010       92/97/20	特性シートの「モデル」     項目の場合には、ここでモデル番       欄にモデルを設定する     号を指定することもできる。
	ModelSample - Magic     体設版       ファイルE 編集(E)表示(Y)     フロジェンド(E) なび環境(E) オブキン(D) デバッグ(D) ペルブ(E)       ロビ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ ロ
	開発モード: ModelSample スペーム パート

- 上図では、変数「V\_拠点コード」と、その特性シートが表示されています。
- 特性シートの「モデル」欄では、「拠点コード」モデルが設定されています。
- 特性シートの「書式」「範囲」などは、「拠点コード」モデルから特性値を継承しており、それぞれ、「書式」は「3」、「範囲」は「1-199」となっています。
- 特性シートの「汎用」カテゴリにある特性値(項目番号、位置付最小/最大など)は、タスクの中で定義される変数に特有な特性値であり、モデルから継承不可です。従って、これらの特性はモデルリポジトリ中には現れていません。

この図で見るように、項目の特性値には、モデルから継承する特性値と、継承しない特性値とがあり、それぞれ、 通常フォント(黒色)と青色ボールド体で表示されます。モデルからの継承については、次節でもうすこし詳しく 説明します。



項目モデルの場合には、特性シートの「モデル」欄でモデル名を設定することもできますが、タス クエディッタの「V=変数」行の第2パラメータ(モデル番号の欄)で、モデル番号を設定することに よりモデルを参照することも可能です。実際、Magic 開発者はこちらの方法を使うのが一般的で す。

### 4.4 モデルの継承メカニズム

V10 でのモデルのメカニズムは、特性値の 継承 という概念をもとにしています。ここでは、継承について説明します。

#### 4.4.1 継承とは

継承とは、オブジェクト(データ項目、GUIコントロールなど)の特性値をモデルより引き継いで同じ値となる、ということで、基本的にモデルを参照しているオブジェクトは、継承可能なすべての特性の値をモデルから継承します。モデルでの特性値の設定が変わった場合、継承されている特性の値も自動的に変わります。これは、ちょうどガラス越しに遠方の物体を見ているようなものです。

ただし、オブジェクトのレベルで、**継承を解除** することもできます。継承を解除された特性は、モデルとは異なる 値を持つことができます。この場合、モデルで特性値の設定が変わっても、オブジェクトのレベルでその変更は 反映されません。同じガラスのたとえを用いれば、ガラスの上にステッカーが張ってあるようなもので、その向こ う側にある物体が変わったとしても、ガラスのこちら側から見るとステッカーだけが見えて、物体が変わったこと がわからない、というようなものです。

また、**継承不可**の特性もあります。これは、本質的に、継承する意味が少ないもの、あるいは他の項目とのか らみがあって継承値の意味がないものなどです。例としては、項目特性の「記憶形式」や「データベース定義」 などがあります。これらの特性は、格納する DBMS によって最適値が異なるので、データリポジトリのカラムに 設定した時点で Magic が設定を変更しますから、DBMS の情報がわからないモデルのレベルでは設定すること ができない特性です。



このような継承不可の特性は、モデルリポジトリでは表示されません。

### 4.4.2 モデルと継承の例

例として、前述の「拠点コード」モデルの特性シートと、それを参照している変数項目「V\_拠点コード」の特性シートとを並べて見比べてみたものです。



これから、次のようなことがわかります。

- 変数「V\_拠点コード」の「モデル」特性では、「拠点コード」モデルを参照しています。
- 「書式」「範囲」などは、「拠点コード」モデルから特性値を継承しており、通常フォントで表示されていま す。
- 特性シートの「汎用」カテゴリにある特性値(項目番号、位置付最小/最大など)は、タスクの中で定義される変数に特有な特性値であり、モデルから継承不可の特性ですので、青色ボールド体で表示されています。また、これらの特性はモデルリポジトリ中には現れていません。

### 4.4.3 継承の解除

モデルから継承されている特性値のうち、モデルとは異なる値を設定したい場合には、特性シート上で新しい値を直接設定することにより、継承を解除することができます。

例えば、「拠点コード」の例で、「範囲」の値を「1-199,999」にしたいとします。この場合には、特性シートの「範囲」 欄に、直接「1-199,999」とキー入力します。すると、「1-199,999」という値が設定され、継承が解除されているこ とを示すために、青色ボールド体のフォントで表示されます。



### 4.4.4 継承とモデルの変更

ここで、継承されている特性(先の例では「書式」など)と、継承されていない特性(「範囲」など)について、モデ ルリポジトリでモデルの定義を変更したばあいに、変更がどのように反映されるかというと、一般的に

継承されている特性にはモデルの変更が反映されるが、継承を解除されている特性には、モ デルの変更が反映されない

となります。

例えば、次の図は、「拠点コード」モデルの設定を変更し、書式を「3」から「5」、範囲を「1-199」から「1-1999」とした場合に、変更前と変更後の特性の値を表したものです。

- モデルから特性値を継承している 書式の方は、モデルでの変更が反映されて、「5」になります。
- しかし、モデルからの継承を解除されている 範囲の方は、モデルでの変更に関わらず、以前の値「0-199,999」のままになっています。



モデルの継承と継承解除とを自由に行えることが、V10のモデルの便利さでもあるのですが、一方において、 継承解除されている特性はモデルの変更を反映しないので、Magicの生産性と保守性の源の一つであるモデ ルの良さを殺してしまうことにもなります。

従って、一般的に言って、モデルの良さを最大限生かすためには、継承が解除されている特性ができるだけ少 なくなるようにすること、すなわち、

ということを推奨します。逆に言うと、継承解除が最小になるように、モデルを設計することが肝要です。

### 4.4.5 継承の解除と再継承の操作

ここで、継承、継承解除、再継承などを行う操作について、変数にモデルを使う場合を例にとって、簡単に説明します。

継承の解除と再継承を行うには・・・

6. 最初に、変数を定義して、モデル欄にモデル番号を設定した状態では、継承可能な項目はすべて継承されています。

変動項目特性 N=動値	: 拠点コード	X
区分(C) 全体(A)		
□詳細		~
た書	3	
型	N=数値	
範囲	1-199	
データビューの一部	Yes	
□入力		
選択プログラム	0	
起動も卜	B=前置	

変動項目特性 N=動値:拠点コード

3

N=要好值 0-199,999

Yes

0 B=前置

全体(A)

区分(C)

古書

範囲

ァー兜 ⊡**入力** 

~ 炭ビューの一部

選択プログラム

起動やト

□詳細

X

~

7. 継承の解除:特定の値の継承を解除するには、値を直接入力します。右図では、範囲特性の値の継承が解除されました。

8.	再継承: 継承が解除されている特性に、再度継承を復旧するに
	は、その特性にカーソルを合わせて、左側に現れる 🔛 アイコ
	ンをクリックします。

変鼓項目特性 N=鼓値	:拠点コード	×
区分(C) 全体(A)		
□詳細		~
た客	3	-
	N=裝d	
田蔵田	0-199,999	
シータビューの一部	Yes	
日維承特性		
選択プログラム	0	
起動モード	B=前置	

9. すると継承が復旧されて、値がモデルでの設定値に戻るととも に、黒色の通常フォントで表示され、特性が継承されていること を表すようになります。

変鼓項目特性 N=鼓値	:: 誕点コード 🛛 🔀
区分(C) 全体(A)	
□詳細	~
た書	3
刑	NF要如直
範囲	1-199
アー地・コーの一部	Yes
□入力	
選択プログラム	0
おお新日本人が	B= 前罟

以上は、個々の特性について継承を解除・復旧する操作でしたが、すべての継承可能な特性について、一度 に継承解除、継承復旧することもできます。



全特性を一度に継承解除するには・・・

- 1. 特性シートの モデル 欄に4つのアイコンがありますが、そのう 変数項目特性 N=数値: 拠点コード ちの右の二つが、継承解除と再継承に関するものです。
- 2. 特性のすべてを継承解除するには、 2 アイコンをクリックしま す。右図のような確認ダイアログが出るので、はい(Y)を押しま す。
- 3. すると、特性シート上のすべての特性が、継承解除を表す青色 ボールド体のフォントで表示されるようになります。

	<u> </u>	
⊟੮デル	$\sim$	~
ŧデ	拠点コー	
□汎用	$\sim$	
項目番号	1	
項目名	拠点コード	
743		6
1856		
<ol> <li>(?) 継承される全て</li> </ol>	の特性が継承されなくなります。維	続しますか
(はい(公))	いいえ(N) キャンセノ	L
		-
发鼓项目特性 N=	<b>数値:</b> 提点コード	×
区分(C) 全体(	A)	
□詳細		~
古書	3	_
型	N=數値	
範囲	1-199	
データビューの一	Yes	
□入力		
選択プログラム	0	
起動モード	B=前置	
□表示		
- 117*画面	0	
2-17-7"	0	
自動117*	0	
□ スタイル		
ブラウザ形式	E=I7°49h	
ブラウザモ式テー	E=17 12	
GUI表示形式	E=17 12	
GUI表示形式,	E=I7 4oh	
GUI出力形式	E=I7°4ab	~
	/ 1/1	

区公(c) 全(h(A)

×



- 次に、継承可能な特性を一度にすべて再継承させるには、モデ ル行の 「」のアイコンをクリックします。
   右図のような確認ダイアログが出るので、はい(Y)を押します。
- すると、特性シート上の特性のうち、継承可能な特性がすべて、 継承されていることを表す黒色の通常フォントで表示されるよう になります。



再継承すると、値はすべてモデルリポジトリ上の値になります。特性シートで直接設定した値は なくなってしまいますので、注意してください。

### 4.5 モデルとコンポーネント

モデルには大別して、

- アプリケーションに固有なモデル(顧客番号、商品名など)
- アプリケーションに関係なく Magic で作ったアプリケーションであるならばどれでも非常によく使われる 共通モデル(終了プッシュボタン、ファイル名など)

#### とがあります。

dbMAGIC V8 までは、アプリケーションは一つの CTL ファイルに格納され、分割することができなかったので、 アプリケーション固有のモデル(タイプ)も共通モデル(タイプ)も一つの CTL に入れておく必要がありました。

V9からは、アプリケーションはコンポーネントに分割できるようになり、コンポーネントを複数のアプリケーションで共有できるようになりました。

この利点を生かして、サンプルアプリケーションでは、次のようなモデル定義をしています。

- アプリケーション固有のモデルは データコンポーネント ED\_DATA で定義
- 共通モデルは共通コンポーネント ED\_COMMON で定義
- 受注コンポーネント ED\_JUCHU、ホスト コンポーネント ED\_HOST では、モデル定義なし。

コンポーネントとモデル定義



## 5 アプリケーションデータのモデル

モデルのうち、一番多用されるのは、アプリケーションのデータについてのモデルでしょう。このモデルは、サン プルアプリケーションの ED\_DATA コンポーネントに定義されています。(下図)

これには、各種コード(顧客コード、商品コード、出荷先コード、・・・)、名前類(顧客名、商品名、・・・)、住所、電話番号、等々、アプリケーションドメインに現れるデータタイプがすべて含まれます。

💹 ED_DATA - Magic - 体験	ω					- 7 🛛
ファイル(E) 編集(E) 表示(V) ブ	iジュクト(P)	オプション(Q) デバッグ(D) ヘル	,7°(⊞)			
놀 🗟 🕨 📰 🗳 🔳 🔯		▝▌ #   ♣   ₽   ₽	🔁 🌇 😭 🗷	2 🖪 📋 🎰	2 🖻 🖻 🗳	164
項目特性 A=文字 ×	(m)	200#350				
区分(C) 全体(A)		1 343 AS 2 123				
	#	名前	252	型	「別り、「公開名	^
	1	アブリケーションデータ	● F=項目	A=文字		
	2	顧客CD	F=項目	A=文字	顧客CD	
書式 12K0	3	顧客名	F=項目	A=文字	顧客名	
型 A=文字	4	顧客名彩	F=項目	A=文字	顧客名託	
範囲	5	郵便番号	F=項目	A=文字	郵便番号	
日入力	6	都道府県	F=項目	A=文字	都道府県	
選択プログラム 4	7	住所	F=項目	A=文字	住所	
記動モード B=前置	8	電話番号	F=項目	A=文字	電話番号	
日表示	9	決算条件CD	F=項目	N=数值	決算条件CD	
N/ EEE U	10	決算条件名	F=項目	A=文字	決算条件名	
/ 10/9/ 0 白 おわいしつ? 0	11	CMB 決済条件	D=GUI表示形式	C=コンホ゛ホ゛ックス	CMB 決済条件	
	12	決済条件CD(CMB)	F=項目	N=裝/值	決済条件CD(C	
フ <sup>×</sup> うりが形式 F=Tディット	13	相当者CD	F=項目	N=#y/if	相当者印	
ブ <sup>・</sup> うりサ <sup>・</sup> 形式テーフ <sup>・</sup> ルE=Iデ <sup>・</sup> ィット	14	CMB 担当者CD	D=GUI表示形式	C=10#1#1~07	CMB 担当者CC	
GUI表示形式 E=Iディット	15	相当来CD(CMB)	F=1百日	0 当5年 年 7777 N=出版(南	相当来CD(CME	
GUI表示形式テーブE=エディット	16	担当者 2	F=項日	A= 立室	担当者GD(GMC	
GUI出力形式 E=Iディット	17	山荷牛(0)	F=1百日	N=Utrin	山荷生の	
GUI出力形式テーブE=エディット	10	油的(100) 清約牛々	F=項日	14-501世	运动1000 清約生々	
テキスト形式 E=エディット	10	连旧几-石 英日区公	項日	1-大丁	在18元4日 英日区公	
	10	2010년 7		AF-义于 45-女宝	時間にカ	
NULL1E •] No 💌	20	경태마니U 호묘 2	1-4日	A-义子 4-女字	商品し	
Ŧデル	21	宿命治	1-4日	A-又子	<b>阿応治</b> 女日後(国	
コステムノコントロールノ項目の各モデルに対	22	商品単価	1-4日	N- 安久1但	商品単価	
するらくているほど母祖本したり祖本 を解除することができます。	23	商品単位	トニ項目	AF 又子	商品里位	
	24	CMB_单位	D=GUI表示形式	U==1)#`#`-97X	UMB_单位	*
間発モード: FD DATA					1、人 広	<b>括 挿入</b>

これらはほとんどが「項目」クラスのモデルで、この「型」としては、文字、Unicode、数値、論理、日付、時刻、その他があります。その他には、「GUI表示」クラスのコンボボックスも、「データコントロール」として定義されています。データコントロールについては、5.3 データコントロール で説明します。



### 5.1 命名規則

ここでは、モデル名はモデルの意味するものそのままとし(「顧客 CD」「顧客名」など)、特別なプレフィックスなど はつけていません。これは、データリポジトリのカラムテーブルでモデルを参照して設定した場合、カラム名に デフォルトでモデル名が設定されることを考慮したものです。

例外としているのは、データコントロールの設定されているコンボボックス (CMB\_担当者 CD など)です。データ コントロールはデータリポジトリでカラム定義に使われることがまずないことと、データコントロールでない、通常 のモデル(担当者 CD)を区別する必要があるために、データコントロール(この場合 コンボボックス) であること がすぐわかるように、CMB\_ というプリフィックスをつけてあります。

流儀によっては、モデル名の先頭に、モデルの種類(データ型や用途)をプレフィックスとして付けたり、数値の 場合には桁数をつけたりするやりかたもあります。こうしておくと一覧からモデルを選択する場合に区別しやす くなりますが、カラムテーブルで設定した場合に名前以外の情報を取り除く必要があります。

### 5.2.1 書式

書式については、V8までに慣れている開発者にとってはよく慣れたものなので、特記することもないと思いますが、TIPSとして、次のような書式を活用すると便利かもしれません。

- A書式 (全データ型に利用可能): A書式はオートスキップを行う書式であり、書式で決められた桁数のデータが入力された場合に、自動的に次項目に進みます。例えば、書式「5」の項目があった場合、ユーザが5文字を入力したら、TabキーやEnterキー、[→] 矢印キーなどを押さなくとも、自動的に次項目に移動します。
- K書式(文字型およびユニコード型で利用可能): K書式は、IME (仮名漢字変換フロントエンド)を制御 する書式で、エディットコントロールの漢字入力欄に設定するのと同じ効果があります。書き方は、K の直後に数字の0~9の値を設定します。この数字の意味は、以下の通りです。

値	入力モード	文字サイズ	入力方法	入力例
0	OFF	-	-	_
1	全角ローマ字かな	全角	ローマ字	かな
2	全角かな	全角	直接	のちみち(KANA)
3	全角ローマ字カナ	全角	ローマ字	カナ
4	全角カナ	全角	直接	ノチミチ(KANA)
5	半角ローマ字カナ	半角	ローマ字	<b>ከ</b> ታ
6	半角カナ	半角	直接	JFミチ(KANA)
7	全角英数	全角	直接	KANA
8	半角英数	半角	直接	KANA

例えば、10K5 と書くと、10桁で IME は半角ローマ字カナ入カモー ドとなります。また、10K0 と書くと、IME は OFF となります。

サンプルでは、文字型のコード類(顧客番号、商品番号など)の書 式に K0 書式が使われています。右図は顧客コードの書式です。





IME の制限により、上のモードのすべてが有効にならない場合があります。

### 5.2.2 選択プログラム

選択プログラム特性は、その項目でズームした場合に呼び出される 一覧選択プログラムです。各種のコードデータ(顧客コード、商品コー ド)などには、それぞれのマスタから選択するための一覧選択プログ ラムがあると思いますが、そのプログラムを設定しておきましょう。

□入力		
選択プログラム	4	SEL_顧客選択
፼起動モード	B=前置	
□表示		
ヘルプ画面	0	
ツールチッフ。	0	
自動心で	0	

### 5.2.3 DB カラム名

DBカラム名は、SQLデータベースを使った場合に、実際にDBMS 上に作成されるテーブルのカラム名を指定するものです。モデルが データリポジトリで利用されることを考慮して、DBカラム名も設定し ておきましょう。

なお、DBカラム名はDBMS上に作成されるテーブルで使われる名 前であるので、利用を想定しているDBMSでの命名規則に則ってい ることが必要です。一般的に言って、あまり長い名前は使わないこ と、特殊文字は使わないこと、半角カナ文字は避けることなどに留 意しましょう。

### 5.2.4 範囲

範囲特性は、この項目に入力可能なデータ範囲を定義するもので す。例えば、取引タイプ CD には、「I:在庫,D:直販」という範囲が設定 されています。

範囲を指定した場合には、できるだけデフォルト値も設定しておきま しょう。デフォルト値を設定していない場合には、文字型の場合は空 文字列、数値型の場合には 0 がデフォルトになりますが、このデフォ ルト値が、範囲指定と矛盾していると、デフォルト値が不正な値、と いうことになり、データモデルの観点から好ましくありません。

⊟SQL	
データベース情報	
DB加名	顧客CD
977°	
2-#*\$77°	

□詳細	
た書	1A
型	A=文字
範囲	I:在庫,D:直販
∃入力	
田教示	
⊞ <b>ス\$イル</b>	
⊡デフォルト/NULL	
NULL値可	No
NULL計算値	
NULL表示文字列	
NULLデドフォルト	No
デフォト値	I
ディータヘィースディンオルト化	

### 5.3.1 データコントロールとは

**データコントロール**というのは、オンライン画面上に配置されるコントロールのうち、多者択一型のコントロール (コンボボックス、リストボックス、及びラジオボタン)であり、その選択肢をデータソース(テーブル)から取ってく るものを言います。

データコントロールは、Magic V9 からサポートされるようになった機能で、V10 でもそのままサポートされています。

V8 およびそれ以前のバージョンでは、他者択一型のコントロールの選択肢は、「範囲」に指定していました。そのため、選択肢はアプリケーション開発時に固定的な値になってしまい、実行時に動的に変更しようとすると、「ラベル」特性に文字型の式を設定し、選択肢をカンマで区切った文字列として連結する、というようなプログラミング技法が必要でした。

V9 以降でも、この手法は引き続きサポートされていますが、それよりもっと簡単なやりかたとして、以下のよう な特性を指定するだけで、実行時に動的に選択肢となるデータを、データソース(テーブル)から取ってくること ができるようになりました。

- ソーステーブル:データリポジトリ中のデータソースの番号を指定します。このデータソースからデータを取ってきます。
- 表示項目:データソースのカラム番号を指定します。このカラムのデータが、選択肢の一覧として画面 上に表示されます。
- リンク項目:データソースのカラム番号を指定します。このカラムのデータが、選択肢の実際のデータ となります。
- インデックス:データソースのインデックスの番号を指定します。画面上での選択肢の並び順を決めます。
- 範囲: ズームすると、項目範囲を指定するダイアログが出てきます。これには、項目(データソースの カラム番号を指定)と、開始、終了(それぞれ式番号を指定します)を指定します。複数の項目を AND 条件で結合することもできます。この特性は、式を参照するものなのでタスクの中でだけ意味がありま すから、モデルリポジトリでは設定できないようになっています。

### 5.3.2 データコントロールの例

例えば、下図は「CMB\_決済条件」というモデル(データコンポーネント ED\_DATA のモデルリポジトリの 11 番に 定義)です。



これに見るように、「クラス」は「D=GUI表示形式」(オンラインフォームとして使うもの)、「型」は「C=コンボボックス」となっています。

その特性として、前述の「ソーステーブル」「表示項目」「リンク項目」「インデックス」などが設定されています。

- 「ソーステーブル」として、「決済条件マスタ」が指定されているので、このテーブルのデータをもとにして 選択肢が表示されます。
- ●「表示項目」として、「決済条件名」(決済条件マスタの第2カラム)が指定されているので、画面上(コンボボックスの内容)としては、決済条件名カラムのデータ内容(「15締め翌月末払い」など)が選択肢として表示されます。
- 「リンク項目」として、「決済条件 CD」(決済条件マスタの第 1 カラム)が指定されているので、ユーザが コンボボックスでデータを選択した場合、実際の内部データとしては決済条件コードの数値型データ (「6」など)が選択されます。
- 「インデックス」はデータソースの第1インデックス「EDM\_決済条件マスタ\_決済条件 CD」となっているので、コンボボックスの内容は決済条件 CD 順に表示されます。

「CMB\_決済条件」のほかに、データコンポーネントの中には、「CMB\_」で始まる名前のデータコントロールのモデルがいくつか定義されています。

- CMB\_担当者 CD
- CMB\_単位
- CMB\_課税区分 CD
- CMB\_申告区分 CD

### 5.3.3 データコントロールの利用(開発時)

次の図は、このモデルを参照しているオンラインタスクのフォームの例です。ここでは、フォームにコンボボック スが配置されており、このコンボボックスではモデルとして、モデルリポジトリの「CMB\_決済条件」を参照してい ます。そうすると、「ソーステーブル」などの特性が継承されてきます。ここでは「範囲」は指定していません(「0」 のままになっています)

決済条件 CD のデータコントロール例 (開発時)			
モデル定義(モデルリポジトリ)			
コントロール特性 ・ コンようについて	1		
区分(C) 全体(A)			
₹デ↓ [デフォレト]			
回計算 選択項目りスト [データ]			
選択表示リスト			
1)114名			
エージー パー パー ソーステープト 4 決済条件マスター	継	承	
表示項目 2 決算条件名			
777項日 1 決算業計60 (2)デックス 1 EDM 決済条件マスタ			
範囲 0			
	コントロール特性:コンボ	ドネックス - 決済条件CD	(CMB) 🛛
フォームエディッタ	区分(C) 全体(A	) 🔻	
圆 OT CMB 決済条件		CMD、违这条件	
	Ĩ	▲ 注管条件CD	0
		[A]	0
	選択表示リスト コントローL名	決済条件CD(CMP	0
		[A] N-裝/值	<i>"</i>
		4 決 2 決	済条件マスタ 質条件名
	りり項目	1 決	算条件CD
	インデーックス 範囲	1 ED	M_)決済条件マスタ
はしていない		1	

### 5.3.4 データコントロールの利用(実行時)

ここで、ソーステーブル「決済条 件マスタ」の内容が、右図のよう になっているとします。

#### 「決済条件マスタ」のデータ内容

照会 -	- 決済条件マスタ				
決算条件	+CD 決算条件名	請求締[	日支払用	引支払日	^
1	随時	0	0	0	
2	10締めの翌06払い	10	1	6	
3	15締めの翌10払い	15	1	10	
4	15締めの翌15払い	15	1	15	
5	15締めの翌20払い	15	1	20	
6	15締めの翌末払い	15	1	99	
7	15締めの翌々15払い	15	2	15	
8	20締めの翌10払い	20	1	10	
9	20締めの翌12払い	20	1	12	
10	20締めの翌15払い	20	1	15	
11	20締めの翌20払い	20	1	20	
12	20締めの翌25払い	20	1	25	~

この状態で、タスクを実行すると、タスクの実行結果 右図のようになり、決済条件 CD のコンボボックスの選択肢 が、決済条件マスタのデータか らとられていることがわかります。



### 6 共通モデル

モデル種類	説明	例
プッシュボタン	よく使う内部イベントやユーザ定義イベントを実行イベントとして設定してあり、よく使われるボタンラベルを書式として設定してあるものです。	PB_終了、 PB_U 実行 E
プッシュボタン項目	上記プッシュボタンモデルに関連付けを行った項目モデルです。	TB_終了、 TB_U 実行 E
GUI 表示形式	よく使われる GUI フォーム、GUI コントロールをモデル化したもので す。	FRM_基本、 TBL_交互色
GUI 出力形式	印刷によく使われる GUI フォーム(出力形式)や、そのコントロールを モデル化したものです。	FRM_GUI 印刷基本 LBL_印刷タイトル (14pt、中央)
その他	一般的な項目モデル	ファイル名、 プログラム名、 インデックス番号

共通コンポーネント ED\_COMMON プロジェクトでは、次のようなモデルが定義されています。

次節以下に、それぞれについて掘り下げて見ていきます。



### 6.1 命名規則

共通コンポーネントで定義しているモデルには、GUI表示形式(オンラインフォーム)、GUI出力形式(GUI印刷) で使うフォームやコントロールのモデルが多くあります。これらは次のような簡単な命名規約に従って名前をつ けています。

種別	命名規約(接頭子)	例
プッシュボタン	PB_	PB_取消
テーブル	TBL_	TBL_交互色
フォーム	FRM_	FRM_選択画面フォーム
スタティックテキスト	LBL_	EDT_中サイス <sup>`</sup> (12pt)
エディット	EDT_	EDT_表示専用
プッシュボタンに関連づけられた項目	TB_	TB_取消
その他の項目	(なし)	ファイル名

### 6.2 内部イベントのプッシュボタンモデル

**内部イベント**というのは、Magic エンジンに対して何らかの動作を行うことを指示する機能を持ったイベントで、 例えば「クローズ(C)」、「キャンセル(C)」「OK」「選択」「ズーム(Z)」などがあります。

内部イベントをプッシュボタンに割り当てることにより、機能を持ったボタンをフォームに配置することができます。 さらに、これをモデル化することによって、フォームへのボタンの配置が非常に簡単になります。

例えば、「終了(X)」という名前で、クリックすると「クローズ(C)」イベントを発行する(タスクが終了する)というプッ シュボタンは大半のオンラインフォームで使われる非常に頻度の高いプッシュボタンですが、こういうプッシュボ タンはモデル化することにより開発時の手間が省けるようになります。

下図は、このプッシュボタンモデルを使った簡単な例です。

🖬 終了末タン 📃 🗖 🗙	
ファイル名:	<ul> <li>● クリックすると、クローズイベントが 発生し、タスクが終了する。</li> <li>● パークしない</li> </ul>

### 6.2.1 プッシュボタンモデルの設定

このプッシュボタンモデルは、共通コンポーネント ED\_COMMON のモデルリポジトリに「PB\_終了」という名前で登録されています。

モデルリボジトリ					×	コントロール特	性:フ <b>゚</b> ッシ	<u>ホタン - PB_終</u>
名前	クラス	型	フォルダ 公開	月名		区分(C)	全体( <u>A</u> )	
1 ● 内部イベントゥ	ボ F=項目	A=文字				ロモデル		
2 PB_取消	D-GUI表示形式	P-7°-9944°99		取消		€7° L		[デフォレト]
3 PB_終了	D=GUI表示形式	P=7°-934#°92	PB_	終了		□詳編		
4 <u>PB_取消(hp-7*/A*)</u>	中 D=GUI表示形式	P=7°-99=4°99	PB_	取消(加-2)(1)		22/10-6	名	PB_終了
5 PB_取消終了	D=GUI表示形式	P=プッシュボタン	PB_	取消終了		た書		終了(&X)
6 PB OK	D=GUI表示形式	P=7°-09±#°\$>	PB	OK	<u> </u>	型		[疗*~为]
						#*\$,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	b	P=プッシュボタン
						了加的	たチアア仙宅	1
						実行へ	DF	/በ-ズ(C)
						<u>実行元</u>		8-3)77 474
						3254754	la~	
						ト・ラック・許	१ ज	No
						ト <sup>*</sup> ロッフ <sup>®</sup> 許	<sup>२</sup> न	No
						日入力		
						修正許可	ſ	Yes
						選択プロ	プラム	[データ]
						起動モード	2	[元*~久]
						りりっりで	1-2	No

このプッシュボタンモデルは、次のような設定になっています。

- オンラインフォームで使うものなので、「クラス」は「D=GUI 表示形式」
- プッシュボタンコントロールとしたいので、「型」は「P=プッシュボタン」

<sup>「</sup>クローズ(C)」内部イベントのプッシュボタンモデル 実行時のイメージ

- ボタンラベルを「終了(X)」としたいので、「書式」は「終了(&X)」
- ユーザがボタンを押したときにタスクを終了させたいので、ボタンを押したときに発行すべきイベントとして、「実行イベント」に内部イベント「クローズ(C)」を設定

### 6.2.2 プッシュボタンモデルのフォームへの配置



- コントロールパレット上で、プッシュボタンのところで、マウスで右クリックします。→ モデルリポジトリに 登録されているプッシュボタンモデルの一覧が表示されますので、この中から「PB\_終了」を選択します。
- ② ドラッグ&ドロップで、フォーム上にこのプッシュボタンを配置します。
- ③ プッシュボタンコントロールのコントロール特性を見てみると、「モデル」でモデルリポジトリの PB\_終了 モデルを参照しており、特性がすべてここから継承されていることがわかります。これにより、書式(ボ タンラベル)、実行イベントなどが自動的に設定されます。



### 6.3 プッシュボタンモデルを参照する項目モデル

旧バージョンでもそうであったように、前節の例の終了ボタンのように、プッシュボタンのみをフォームに配置した場合には、マウスでプッシュボタンを押すことはできますが、キーボード操作でカーソルがプッシュボタンにパークしません。

キーボード操作でカーソルがボタン上にパークするようにしたい場合には、プッシュボタンに変数項目と組み合わせなければなりません。

この場合、タスクの「データビュー」画面で変数項目を定義し、フォーム上でプッシュボタンコントロールを定義して、関連付けを行っても良いのですが、変数項目の定義とプッシュボタンコントロールの定義をワンセットにして モデルで定義しておくことができれば、もっと便利です。特に、「終了(X)」ボタンなどは、ほとんどすべてのオンラ イン画面につけるものなので、モデル化できれば大変有効です。

V8 でも、タイプリポジトリの「スタイル」タブで、オンラインフォームでの表示形式を指定することができましたが (下図)、V9 および V10 ではもっときめ細かに設定できるようになっています。

ネイプリポジトリの 	「スタイル」タブ						
📕 タイプ特性: PB	タイプ特性: PB_終了(X)						
(A)関連	( <u>D</u> )テ <sup>°</sup> フォルト/Null	( <u>S</u> )スタイル					
□表示設定	•						
コント	コールタイブ: 阔 - アッシュ	#*\$ <u>)</u>					
<b>ア</b> クシ	ョン: <mark>β:クロ</mark> ・	- <del>x</del>					
スタイル	レ: 8=凸立(	<b></b>					
境界:	N=なし						
将复要执行于?	編集: No	_					
色;		1					
フォン	⊦: <b></b>	9 基本ブッシュボタンフ:	ォント				
		C OK	取消				

本節では、プッシュボタンを関連付けた項目モデルについて説明します。

### 6.3.1 項目モデルの定義

V10のモデルリポジトリで「クラス」が「F=項目」であるモデルを定義した場合、その項目をフォーム上に配置した 場合、どのような形式のコントロールで表示されるかは、項目特性の「スタイル」カテゴリの中にある各特性に設 定します。

#### プッシュボタンモデル PB\_終了 を参照する項目モデル TB\_終了

① モデルリポジトリ

3 モデルリポジトリ									
#	名前	752	型	7加約1公	^				
8	PB_ズーム	D=GUI表示形式	P=プッシュボタン	PB_;	_				
8	TB_取消	F=項目	A=文字	TB_:					
(10	TB_終了	F=項目	A=文字 )	TB_i					
	TB_取肖(カローズイベント)	<del>F=項目</del>	A=文字	TB_:					
12	TB_取肖終了	F=項目	A=文字	TB_:					
13	TB_OK	F=項目	A=文字	TB_I					
14	TB_選択	F=項目	A=文字	TB_;					
15	TB_ズーム	F=項目	A=文字	TB_:	~				
L									
	÷								

② 項目モデル TB\_終了 の特性



④ モデル一覧

X

モデルー	·覧	X
表示:	全て	
Ħ	名前	~
2	PB_取消	
3	PB_終了	
4	PB_取消(クローズイベント)	
5	PB_取消終了	
6	PB_0K	
7	PB_選択	
8	PB_ズーム	
17	PB_U実行E	
18	PB_U実行C	
19	PB_U実行R	
20	PB_U照会	
21	PB_U修正	
22	PB_U登録	_
23	PB U肖·IF余	*
一説明-		
·		_
	選択 ++ンセル	

例として、終了ボタンに関連づけられた項目モデル TB\_終了の定義を見てみます。

- ① ED\_COMMON プロジェクトのモデルリポジトリを開いてください。10 番目に **TB\_終了** という名前のモデ ルがあります。「クラス」は「F=項目」で、「型」は「A=文字」です。
- ② 特性シートを開いて、特性を見てください。次のような特性が設定されています。

③ GUI表示形式 の特性

- ◆ データサイズは最大 16 バイトとしたいので、「書式」は「16」 にしています。
- ◆ オンラインフォームに配置した場合、デフォルトではプッシュボタンとして配置したいので、「GUI表示形式」および「GUI表示形式テーブル」特性では、いずれも「P=プッシュボタン」に設定しています。
- ◆ ボタンの表示ラベルのデフォルトは「終了(X)」としたいので、「デフォルト値」に「終了(&X)」と設定し



項目特性の「スタイル」カテゴリにある設定項目は、 それぞれの形式のフォーム上に配置した場合に、 デフォルトでどの形式のコントロールとして配置され るかを定義するものです。

- **ブラウザ形式**: ブラウザクライアントフォームに配置する場合のコントロール形式。
- GUI 表示形式:オンラインフォームに配置する場合のコントロール形式。

項目特性 A=文字	X
区分(C) 全体(A)	
⊡ <b>スタイル</b>	~
ブラウザ形式 E=エディット	
ブラウザ形式テーブル E=エディット	
GUI表示形式 P=ブゥシュギタ	2
GUI表示形式テーブル P=ブゥシュギタ	2
GUI出力形式 EFIディット	
GUI出力形式テーブル E=エディット	_
テキスト形式 E=エディット	

- GUI 出力形式: GUI 印刷のフォームに配置する 場合のコントロール形式。
- テキスト形式: テキスト形式のフォームに配置する場合のコントロール形式。

「~テーブル」という名前が付いているものは、テーブルコントロール中に配置される場合(ラインモードの場合)であり、付いていないものは、テーブル外に配置される場合(スクリーンモードの場合)の設定です。

このように、V8 ではオンラインフォームの場合のコントロールタイプだけを1種類設定できましたが、V9/V10 では各フォームタイプ毎、表示形式(スクリーン/ラインモード)ごとに設定できるようになりました。

- ③「GUI表示形式」の欄からズームしてください。プッシュボタン についての特性が表示されます。この設定はそのまま、オン ラインフォームに配置された場合のプッシュボタンの特性のデ フォルト値となります。
- ④ この特性シートで、「モデル」特性がありますが、ここでモデル リポジトリ中に定義されているプッシュボタンモデルを参照す ることもできます。

モデルを参照すると、そのモデルの特性がすべて継承されます。



ここではモデル 特性にプッシュボタンのモデル PB\_終了 が参照されています。このため、PB\_終了 に定義されている特性が継承されて、実行イベント が 内部イベントの クローズ(C) になっています。

継承されている特性は、黒色の通常フォントで表示されますが、一箇所だけ、緑色のフォントとなっている特性 「書式」があります。これは「データからの継承」を意味するものですが、これについては次に説明します。

### 6.3.2 データからの継承

GUI 表示形式の特性は、ほとんどがモデル PB\_終了 から継承しますが、一つだけ継承を解除している特性があります。それは 書式 特性で、継承ではなく [データ] として緑色で表示されています。

これは「データからの継承」を意味するもので、「書式」特性については 「モデル」に設定されている「PB\_終了」から継承するのではなく、項目 特性から継承することを意味します。

ここの例では、項目モデル「TB\_終了」の書式は 16 なので、フォームに 配置した場合のプッシュボタンの書式は 16 となります。



継承を [データ] にしておくと、実行時には次のようになります。

- プッシュボタンの書式は、固定的な「終了(X)」ではなく、項目の書式「16」となります。
- 開発時、フォームエディッタに配置する場合、書式 16 を元にしてプッシュボタンの幅を計算します。
- 実行時、プッシュボタンに表示される文字列は、変数項目に格納されている文字列データとなります。

これにより、実行時ダイナミックにプッシュボタンの表示文字を変更することができるようになります。

ただし、固定的に「終了(X)」でもよい場合にでも変数にデータを設定する手間を省くため、項目モデルの デフォ ルト値 特性に「終了(X)」を設定しておきます。



- 1. プロパティシートの 書式 特性にカーソルをおきます。
- 2. 左にある 🖽 ボタンをクリックします。
- 3. メニューが出てくるので、データを選択します。

コントロール特性:フ*ァシュホミ	シ-PB_終了	
区分(©) 全体( <u>A</u> )		
⋳モデル		~
स्त <sup>8</sup>	PB_終了	
□詳細		
动和北名	PB_終了	
は書式		
維承	[元》为]	
データ	P=プッシュボタン	
解除 77 仙名		
25111-21	/ת∽תל (כ)	
実行元	C=コンテナ タスク	
ͻϽ;ϯ;ϒͿϫϫϫ		

### 6.3.3 プログラムでの利用

このように定義されている項目モデルをプログラムで利用するときは、次のようになります。

- モデルリポジトリには、プッシュボタンモデル PB\_終了と、これを GUI 表示形式で参照している項目モデル TB\_終了 があります。
- ② プログラムのデータビューで、変数項目を作成し、モデル TB\_終了 を参照します。この変数項目の特性はすべてモデルリポジトリから継承します。
- ③ この変数項目をフォームに配置します。このとき、項目の GUI 表示形式 特性が P=プッシュボタン になっているので、フォーム上ではプッシュボタンとして配置されます。そして、このプッシュボタンの特性については、書式はデータから継承して「16」になり、実行イベントはモデルリポジトリの PB\_終了 から継承して「クローズ(C)」になります。



### 6.3.4 実行時の動作

このようにしてフォームにプッシュボタンを配置すると、実行時には次のような動作になります。

- 1. カーソルがプッシュボタンにパークします。
- マウスでプッシュボタンをクリックすると、クローズイベントが発生し、タスクが終了します。

カーソルがパークするところが、ボタンのみを配置した場合と動作が異なるところです。

### 6.3.5 クリックでパーク 特性

V10 のプッシュボタンには、**クリックでパーク**という特性があります。この 特性のデフォルト値は Yes ですが、ED\_COMMON コンポーネントに登録さ れているプッシュボタンモデルではすべて No に設定してあります。

この特性は、コントロールレベルのイベントのサイクル(コントロール前処 理、コントロール検証、コントロール後処理)の実行を制御するもので、こ れを No にしておくことにより、プッシュボタンをクリックした時のフォーカス の移動を抑えて、コントロール検証の処理をスキップすることができるよう になります。

詳しいことはここでは省略しますが、リファレンスヘルプの表示フォーム > GUI コントロール > GUI 表示コントロール特性 > プッシュボタンコントロー ル特性 を参照してください。

■ 終了ポタン項目	
ファイル名:	
	終了( <u>X</u> )



### 6.4 ユーザイベントのプッシュボタンモデル

タスクを終了するための「クローズ」イベント、入力内容を取り消すための「キャンセル」イベント、選択プログラ ムで特定のレコードを選択するための「選択」イベントなどは、内部イベントとして予め Magic エンジンに実装さ れているので、プッシュボタンの「実行イベント」特性にはそれぞれの内部イベントを設定しました。

ー方、「印刷」「出力」「検索」などといったボタンについては、Magic エンジンでそれに対応する内部イベントはないので、ユーザイベントを発行させて、対応するイベントハンドラをプログラムに定義する、という形になります。 従って、プッシュボタンモデルの実行イベントには、適当なユーザ定義イベントを設定することになります。

### 6.4.1 ユーザ定義イベント

印刷、出力、検索などをトリガーするユーザ定義イベントは、多くのアプリケーションでよく使うものなので、共通 コンポーネント ED\_COMMON で定義しておきます。 グローバルなイベントにする必要があるため、メインプログ ラムで定義し、公開名をつけて、コンポーネントの外部に公開しておくようにします。

ED\_COMMON では、下図のようなユーザ定義イベントが定義されています。

ተላጋ	þ	1 - 🔥 ታለ	ンプログラム	<b>x</b>				
#		名前	トリカ・タイフ。	わが	በእ° ラメータ	強制終了	公開名	公開
	1	GU_ズーム	I=内部	ን,-ዮ(S)	0	E=編集	GU_ズーム	<ul> <li>Image: A set of the set of the</li></ul>
	2	GU_照会	N⊨なし		0	E=編集	GU_照会	<b>~</b>
	3	GU_修正	N≕なし		0	E=編集	GU_修正	<b>~</b>
	4	GU_登録	N≕なし		0	E=編集	GU_登録	<b>~</b>
	5	GU_削除	N≕なし		0	E=編集	GU_肖邩涂	<b>~</b>
	6	GU_出力	N≕なし		0	E=編集	GU_出力	<b>~</b>
	7	GU_ED刷	N≕なし		0	E=編集	GU_ED刷	<b>~</b>
	8	GU_取消	N≕なし		0	E=編集	GU_取消	<b>~</b>
	9	GU_終了	N≕なし		0	E=編集	GU_終了	<b>~</b>
1	10	GU_選択	N≕なし		0	E=編集	GU_選択	<b>~</b>
1	11	GU_検索	N≕なし		0	E=編集	GU_検索	<b>~</b>
1	12	GU_実行N	N=なし		0	N=なし	GU_実行N	<b>~</b>
1	13	GU_実行E	N=なし		0	E=編集	GU_実行E	<b>~</b>
1	14	GU_実行C	N=なし		0	C=⊐)Ւם-W	GU_実行C	<b>~</b>
1	15	GU_実行R	N=なし		0	R=レコード更新前	GU_実行R	<b>~</b>

### 6.4.2 ユーザ定義イベントを参照するプッシュボタンモデル

① モデルリポジトリ

モデルリポジトリでは、ユーザ定義イベントを使うプッシュボタンのモデルが、モデル 17番以降に定義されています。

例として、ユーザ定義イベント GU 印刷 を発行するプッシュボタンイベント PB\_U 印刷 の特性を見てみると、次の図のようになっています。

🎒 モデルリポジトリ 名前 クラス 型 「フォルジ|公開名 # ● ユーザイベント F=項目 16 A=文字 17 PB\_U実行E D=GUI表示形式 P=7°ッシュボタン PB\_U実行E 18 PB\_U実行C D=GUI表示形式 P=プッシュボタン PB\_U実行C P=7°-,91#\*92 19 PB U実行R D=GUI表示形式 PB II実行R P=7°-934\*\*92 20 PB 1 昭会 PB I昭会 DEGIII表示形式 21 PB\_U修正 P=プッシュボタン PB\_U修正 D=GUI表示形式 P=7°->シュ市\*タン 22 PB\_U登録 D=GUI表示形式 PB\_U登録 P=プッシュボタン 23 PB\_U削除 D=GUI表示形式 PB\_U削除 DEGUI表示形式 PB UUU P= 7° asia t° b PB UH 25 PB\_UED刷 D=GUI表示形式 P=プッシュボタン PB\_UED刷 26 PB U取消 DEGUI表示形式 P=7"-094#197 PB U取消 D=GUI表示形式 P=プッシュボタン 27 PB U終了 PB U終了 P=フ°ぃシっ击゙なン 28 PB 16曜択 D=GUI表示形式 PB II避祝 P=プッシュボタン 29 PB U検索 D=GUI表示形式 PB U検索 PB\_U印刷モデルの特性 コントロール特性:プッシュボタン - PB\_U印刷 🛛 区分(<u>C</u>) 全体(<u>A</u>) 🗆 tī 🕯 [÷`]#I □詳細 PB\_U印刷 印刷(&0) リカー名 **方告** ③実行イベントの詳細 ボタンスタイル P=7°-ッシュホ\*タン GU\_EP 実行へい 美行元 280 0-1077 イベントのタイプと実行するイベントを指定してください。 -1);5\$2,6%\*\*\* うック゛許可 No /\\*`>F9/7\*: U=2-3 ト゛ロッフ。許可 No へつた GU ED扇I □入力 修正許可 Yes [データ] OK ++>tell 選択プログラム クリックでれ・-ク No □表示 フォント 9 <u>↓||,¬°)</u>浦(西 ि ≓ि्य

- ① モデルリポジトリの PB\_U 印刷 行から、プロパティシートを開きます。
- ② プロパティシートでは、コントロール名、書式、実行イベントなどの特性が設定されています。
- ③ 実行イベントからズームすると、ユーザ定義イベント GU\_印刷 が設定されています。

これを見てわかるように、6.2 内部イベントのプッシュボタンモデル で説明した、プッシュボタンモデルとほとんど同じで、ただ、実行イベントとしてユーザ定義イベントが設定されている、ということだけが異なります。

### 6.4.3 ユーザ定義イベントのプッシュボタンモデルを参照する項目モデル

前述のように定義されている一連のプッシュボタンモデルを参照して、項目モデルがモデル30番以降に定義されています。これらの項目モデルについては設定内容は内部イベントの場合と同様なので、説明は省略します。

### 6.4.4 どんなユーザイベントを用意すべきか?

サンプルアプリケーションの ED\_COMMON コンポーネントでは、ユーザ定義イベントが 15 個定義されています。 そして、それに合わせて、プッシュボタンモデルも定義されています。ユーザ定義イベントは、文字通りユーザ が自由に定義することのできるイベントなので、定義できる数に制限はありません。それでは、どんなユーザ定 義イベントを何個くらい用意しておけば良いでしょうか?

この答えは、もちろんアプリケーションに依存します。しかし、モデル利用の目的である、共有による開発・保守 効率の向上を考慮すれば、基本的な指針は出すことができます。

具体的な例を使って考えてみると、「印刷」「編集」などといった、多くの画面で出てくるような機能のボタンに対しては、モデル化する利点が高いといえます。同じ設定を何回も繰り返して行う必要がなくなるからです。

ー方で、「ログイン」というようなボタンは、アプリケーション中で「ログイン」画面にしか出てこないかもしれません。このように一箇所だけでしか現れないプッシュボタンに対しても、ユーザ定義イベント(例えば、GU\_ログインとか)を定義し、それに対応するプッシュボタンモデルを定義し、それを参照する項目モデルを定義し、タスクの データビューでその項目モデルを参照する、というのは、いかにも迂遠で、返って開発効率が悪くなります。

このような出現頻度の非常に低いプッシュボタンに対しては、それ専用のユーザ定義イベント/プッシュボタンモ デル/項目モデルを作らずに、汎用のユーザ定義イベント/プッシュボタンモデル/項目モデルを定義しておいて、 ボタンラベルだけをプログラムで変える、というようにするのが簡便です。

このような目的の汎用定義が、ユーザ定義イベント GU\_実行 E、プッシュボタンモデル PB\_U 実行 E、項目モデル TB\_U 実行 E です。次ページの図は、受注入カプログラムで「確定」 ボタンを定義している例です。

- ① モデルリポジトリでは、ユーザ定義イベント GU\_実行 E を使った プッシュボタンモデル PB\_U 実行 E、 項目モデル TB\_U 実行 E が登録されています。
- ② プログラムでは、データビューで変数項目 TB\_U 確定 を定義します。この変数は、モデル TB\_U 実行 を参照します。
- ③ 変数特性はほとんどがモデルより継承しますが、ボタンラベルを変更するために、**デフォルト値**を「確 定(&C)」にしています。
- ④ この変数をフォームに配置します。変数の GUI 表示形式 特性が プッシュボタン になっているので、 フォーム上ではプッシュボタンとして配置されます。
   このプッシュボタンの特性は、モデル PB\_U 実行 から継承しますが、書式だけはデータから継承して、「16」となります。
- ⑤ 実行時にこのプッシュボタンをクリックすると、実行イベントとして設定されている GU\_実行 E イベント が発生するので、プログラムではこのイベントに対するハンドラを定義しておきます。 この例では、内部フラグをいくつか設定して、タスクを終了するために 終了 イベント(内部イベント)を指 定して イベント実行 コマンドを実行しています。



⑤ タスクのロジックで、GU\_実行E に対するハンドラを定義

データビ	ュー ロジック	フォーム	4						
42	🗉 E={\*`ን} 🤇	ED_CON	ON.GU	実行E	101			スコーフ* T=5	はり
43	項目更新	V=項日	G	₩_変更確定	Ξ?	値:	25	'TRUE'LOG	
44	項目更新	V=項目	F	₩S_後処理フ	50°	値:	21	'再開'	
45	べい実行	終了						<u> </u>	No



この例での「確定」ボタンのように、汎用実行イベント (GU\_実行 E イベント)を使うボタンを、一つのフォーム上に複数配置することも可能です。この場合には、どのボタンでクリックされても GU\_実行 E イベントが発生するので、どのボタンでクリックされたかを区別するために、イベントハンドラのヘッダ行でコントロール名パラメータを設定します。

### 6.5 GUI 表示形式のモデル

V9から、モデルリポジトリにフォームや GUI コントロールなどの GUI オブジェクトもモデルとして登録できるよう になりましたが、まだ実際にはそれほど活用されていないようです。ここでは、オンラインフォームでの GUI オブ ジェクトモデルである「GUI 表示形式」のモデルについて説明します。



それでは、GUIオブジェクトのモデル化の可能性はいかなるものがあるでしょうか?サンプルアプリで見ていくことにしましょう。

下図は、受注入力画面のイメージ(修正モード時)で、商品 CD 欄からズームして商品一覧を表示しているところです。

ED_HOST						- 7
ファイル(E) 編集(E)	) サンブルアプリケーション	オプジョン( <u>O</u> ) ウィ	♡トウ₩			
🚔 🚔   🖾 🛅	🔢 🔁 🚑 🖗 🕅	li 🖅 🖹 🖥				
受注番号 8	000000012 <mark>%7°</mark> I:	在庫 🔽		取引日 07/04/26	納入予定日 07/	/04/27
顧客CDOO	130 アールエ. アールエオンキ	ス <u>産業</u> ゛ョウ		出荷先CD 0 アールエス	産業	
住所 4	76-XXXX 愛知県	商品選択				
3	東海市南柴田町八の割X3	商品CD	△ 商品名		<u>^</u>	
		01035	白砂塘			
CEIII TEIII CEIII CEIIII CEIII CEIIII CEIII CEI	48-285-6754	01036	塩			X52-6X3-X2X7
顧客発注番号		01037	ホットシリアル		=	
担当者CD	6 安部 考之	01038	ヌードル			
		01039	マヨネーズ			
# 商品CD	商品名	01040	タイ米			頁 金額 約期日 🔤 🗠
1 01039	マヨネーズ	01041	オリーブオイル			3 7,833 07/04/27
2 01039	マヨネーズ	01042	チリ			6 9,791 07/04/27
		01043	ホットマスタード			
		01044	スポーツドリンク			
		01045	ライスミルク			
		01046	ホワイトワイン			
		01047	パウダーミルク			
		01048	アップルサイダー酢			
		01049	ドリンクミックス			
		01050	チキンスープ		×	
			選択( <u>S</u> )	取消( <u>C</u> )		×
修正						金額 17,624
		受注検索(3)	ED刷(P) 削除(D) 修	正(M) 照会(Q) 登	<b>録(<u>C</u>)</b> 確定(	<u>C)</u> 取消( <u>C</u> ) 終了( <u>X</u> )
実行: ED_HOST						照会

この画面と、その他の画面とを総合して検討してみると、次のような GUI オブジェクトが共通した属性を持つこと に気が付きます。

種別	内容	主要設定属性
フォーム	受注入力や受注取引画面参照プログラムでは、Magic の ウィンドウー杯にフォームが広がって、灰色の背景色で表 示されます。また、フォームエディッタでのグリッド間隔は X が 0.25、Y が 0.125 です。	ウィンドウタイプ: MDI 調整 タイトルバー: No 色: 4 グリッド間隔(X): 0.25 グリッド間隔(Y): 0.125
	商品選択、顧客選択、その他の選択画面は、画面中央に 表示され、タイトルバーにはシステムメニューや最大化/ 最小化/閉じるアイコンがありません。色やグリッド間隔も 上記と同様です。	システムメニュー: No 開始時の位置: MDIの中央 色: 4 グリッド間隔(X): 0.25 グリッド間隔(Y): 0.125
テーブル	テーブルコントロールは交互色表示で表示されており、 フォーカスのある行は青地に白色で表示されます。	テーブル色の指定: テーブルに依存 交互表示色: 101 ハイライト行の色: 3
テキスト	表示データのテキストラベルは、青地に白色の文字で表 示されています。	色:3
エディット	表示専用の項目は、灰色の背景色をしています。また、 パークしません。	修正許可: No 色: 4 パーキング可: No

ボタン	終了、取消、選択、修正、印刷、検索などのボタンは、い	書式: (それぞれのボタン表示)
	ろいろなプログラムに使われています。	実行イベント: (それぞれの機能に応
		じたイベント)

このような GUI オブジェクトをモデル化することにより、一貫した GUI インターフェースを作成するのが容易になり、個々に設定するときの設定し忘れなどがなくなります。また、変更する場合もモデルのみの変更で対応する ことができます。

サンプルでは、これらの GUI オブジェクトのモデルは、共通コンポーネント ED\_COMMON のモデルリポジトリ 43 番以降に登録しました。

23	ŧ	テルリポジトリ				×
#		名前	752	型	フォルジ 公開名	>
	43	● GUIコントロール漢頁	F=項目	A=文字		
	44	TBL_交互色	D=GUI表示形式	T=テーブル	TBL_交互色	
	45	TBL_交互色(位置100)	D=GUI表示形式	T=テーブル	TBL_交互色(位置100)	
	46	FRM_基本	D=GUI表示形式	F=フォーム	FRM_基本	
	47	FRM_選択画面フォーム	D=GUI表示形式	F=フォーム	FRM_選択画面フォーム	
	48	FRM_スクリーン入力フォーム	D=GUI表示形式	F=フォーム	FBM_スクリーン入力フォーム	
	49	FRM_MDI7W	D=GUI表示形式	F=フォーム	FRM_MDI70	_
	50	LBL_ <del>j</del> ^*l	D=GUI表示形式	S=スタティック	LBL_ <del>j</del> ∿°⊯	=
	51	EDT_表示専用	D=GUI表示形式	E=Iディット	EDT_表示専用	
	52	EDT_表示専用センタリング	D=GUI表示形式	E=Iディット	EDT_表示専用センタリング	
	53	EDT_∃≷ガナ	D=GUI表示形式	E=Iディット	EDT_ヨミカ <sup>ゃ</sup> ナ	
	54	EDT_D&D可能	D=GUI表示形式	E=Iディット	EDT_D&D可能	~

これらのモデルでは、「クラス」欄はすべて「D=GUI表示形式」となっており、「型」欄はフォームまたはコントロールの種別を表しています。

例えば、テーブルコントロールのモデル「TBL\_交互色」の特性は右 図のようになっています。これは交互色テーブルを実現するため のもので、次のように設定されています。

- 交互色テーブルを実現するため、「テーブル色の指定」は「T=テーブルに依存」としています。
- 交互色の色は、薄い空色を背景とした黒色文字としたいので、
   「交互表示色」として 101 番 を設定しています。
- ハイライト行は、青色を背景とした白色文字としたいので、「ハイ ライト行の色」は3番としています。

交互色テーブルはアプリケーション中で多用していますが、モデル に定義しておけば、フォームエディッタ上でテーブルコントロールの 「モデル」特性でこのモデルを参照することにより、簡単に交互色 テーブルとすることができます。

コントロール特性:テーン	プル - TBL	X
区分(C) 全体(A	)	
□表示		~
色	1	_
テープ・色の指定	T=テーブルに依存	
交互表示色	101	
ツールチッフ。	0	
አንብዞ	₩=₩indows	-
境界のスタイル	旧太線	
スクロールバット	Yes	
区切り	No	
ハイライト行のスタイル	C=背景とコントロール	
ハイライト行の色	3	
下辺の間隔	N≕なし	
肺の区切線	No	
最終区切線	Yes	
ウィント・ウヘテーブル	Yes	
サイズ変更可	Yes	
抗の並び替え可	Yes	
フルチフーキング	Yes	$\mathbf{\mathbf{x}}$
<b>モデル</b> フォーム/コントロール/項目 性を再継承したり継承	の各モデルに対するすべての なを解除することができます。	)特

ここでは、各モデルの詳細についての説明は省略します。GUI インターフェースはアプリケーション (エンドユー ザや開発者の標準や嗜好など)により千差万別であり、サンプルに登録したものはごくごく基本的なものだけな ので、読者の皆さんが自分の財産として開発されるとよいと思います。

### 6.6 GUI 出力形式のモデル

GUI 出力形式は、GUI 印刷用のフォームで使うコントロールに関してモデル化を行うものです。



オンラインフォーム上の GUI オブジェクトを定義する GUI 表示形式のモデルに比べ、印刷用の GUI 出力形式 のモデルはバリエーションが少ないと思いますが、それでもいくつかの候補が挙げられるでしょう。

下図は、サンプルの受注報告書印刷フォームをフォームエディッタ上で開いたものです。

□ 入出力フォーム: クラス 1					
	— 受注報告	₩198° —			
- X000X00000000000000000000000000000000	受注報	3告書			ページ: ##### 日付: YYYY/MM/DD 時刻: HH:MM:SS
		//s <b>EE</b> //s			
(人工部号)00(###################################	出荷先 摘要 決算条件	## XXXXXXXX XXXXXXXX ### XXXX	00000000000000000000000000000000000000	XXXXXXXX 取引 XXXXXXXXX XXXXXXXXXXXXX	タイプ XXXXXXXX
ND 商品コート <sup>*</sup> 商品名	販売数量	課税区分	経理処理 単価	消費税金額	金額 納期日
	_ 受注報告	書合計 ——			
		; 書 フッタ	승카 -1	, ###, ###, ### -	#, ###, ###, ###
			総合計 -1	, ###, ###, ### -	#,###,###,###

ここでのモデル化候補としては、次のようなものがありましょう。

種別	内容	主要設定属性
フォーム	グリッド間隔がデフォルトより細かい。	グリッド間隔(X): 0.100 グリッド間隔(Y): 0.100
テキスト	レポートのタイトル「受注報告書」が大きいフォントで印刷 される。	フォント: 5
	「受注番号」が中くらいのフォントで印刷される。	フォント: 7
エディット	受注番号が中くらいのフォントで印刷される。	フォント: 7

これらをモデル化したものが、共通コンポーネント ED\_COMMON のモデルリポジトリ 55 番以降に登録されています。

🛞 モデルリポジトリ						
#		名前	7 <del>5</del> 7	型	フォルジ 公開名	>
	55	● GUIEP刷用	F=項目	A=文字		
	56	LBL_印刷外NW(14pt,中	O=GUI出力形式	S=スタティック	LBL_印刷炉仆W(14pt,中	
	57	LBL_中外NW(12pt)	O=GUI出力形式	S=スタティック	LBL_中外仆(12pt)	
	58	EDT_中サイズ(12pt)	O=GUI出力形式	EFIディット	EDT_中サイズ(12pt)	-
	59	FRM_GUI印刷基本	0=GUI出力形式	F=フォーム	FRM_GUIEP刷基本	~

GUI 出力形式も、GUI 表示形式と同様、アプリケーションによってもっとバラエティの多いものとなろうと思います。帳票設計にも、モデルを活用するようにしましょう。

### <u>6.7 その他のモデル</u>

その他に共通コンポーネント ED\_COMMON に登録されているモデルとしては、60番以降に次のようなものがあります。

種別	内容	主要設定属性
項目	ファイル名	書式: 255 選択プログラム: (ファイルダイアログを表示)
	プログラム管理用のデータのモデル。タスクモード、プロ グラム番号、プログラム名など。	書式

🗱 モデルリポジトリ					
#	名前	クラス	型	フォルダ 公開名	~
6	0 🔴 その他	F=項目	A=文字		
6	1 ファイル名	F=項目	A=文字	ファイル名	
6	2   ● 内部制御用 ●	F=項目	A=文字		
6	3 タスクモード	F=項目	A=文字	タスクモード	
6	4 インデックス番号	F=項目	N=裝y值	インデックス番号	
6	5 ブログラム番号	F=項目	N=裝y值	ブログラム番号	
6	6 ブログラム名	F=項目	A=文字	ブログラム名	
6	7 プログラムID	F=項目	N=裝y值	プログラムID	
6	8 メニューID	F=項目	N=娄y/直	メニューID	
6	9 メニュー名	F=項目	A=文字	メニュー名	
					<u>×</u>

これらはアプリケーションに依存するデータではなく、共通的に使えるデータですので、モデル化しておきました。

### 6.8 まとめ

本章では、サンプルアプリケーションに定義されているモデルについて、特に共通コンポーネントのモデルに焦点を当てて説明してきました。

プッシュボタンのモデルは、イベント指向プログラミングにおいて非常に役にたつモデルです。本章ではイベント 指向プログラミングについて詳しく説明しませんでしたが、今後公開予定の「イベント編」でもっと掘り下げた内 容を説明する予定です。

GUI 表示形式および GUI 出力形式のモデルは、V8 までにはなかったものですので、Magic のバージョンが V9、V9Plus、V10 に移行してもなかなか使われていないようです。しかしこのタイプのモデルもうまく活用すれば、 項目モデル(V8 までのタイプ辞書)と同様、生産性と保守性を大きく向上させる可能性を持っています。ぜひ、 GUI 表示/出力形式のモデル化も進めていってください。

### Magic eDeveloper V10



Magic eDeveloper V10

コーディングサンプル Version 2 (モデル編)

Copyright  $\ensuremath{\mathbb{O}}$  2007, Magic Software Japan K.K., All rights reserved.

第1版 2007年10月10日

発行 〒151-0053 東京都渋谷区代々木三丁目二十五番地三号
 あいおい損保新宿ビル 14 階
 マジック ソフトウェア・ジャパン(株)
 http://www.magicsoftware.co.jp/