HITACHI
Inspire the Next

XMAP3 Version 5

画面・帳票サポートシステム

XMAP3 プログラミングガイド

手引・文法書

3020-7-513-60

■ 対象製品

P-262B-5C54 XMAP3 Developer Version 5※ 05-06（適用 OS：Windows 7，Windows 8.1，Windows 10，Windows Server 2008，Windows Server 2008 R2，Windows Server 2012，Windows Server 2012 R2，Windows Server 2016，Windows Server 2019）

P-1M2B-2551 XMAP3 Server Runtime Version 5 05-10（適用 OS：AIX V6.1，AIX V7.1，AIX V7.2）

P-1J2B-2551 XMAP3 Server Runtime Version 5 05-06（適用 OS：HP-UX 11i V2(IPF)，HP-UX 11i V3(IPF)）

注※　この製品は，64bit 版の Windows では 32bit 互換モード（WOW64：Windows On Windows 64）で動作します。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

■ 発行

2021 年 6 月　3020-7-513-60

■ 著作権

変更内容

変更内容（3020-7-513-60）XMAP3 Developer Version 5　05-06

追加・変更内容	変更個所
記述を改善した。	－

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルは，次に示す製品の機能と使い方について説明したものです。

- XMAP3 Developer Version 5
- XMAP3 Server Runtime Version 5（UNIX 版）

■ 対象読者

XMAP3 を利用して，システムを開発する方でマニュアル「XMAP3 Version 5 画面・帳票サポートシステム XMAP3 開発ガイド」を読んでいる方，または UNIX 版 XMAP3 Server Runtime Version 5 を使用する方を対象としています。また，次に示す項目について理解，習得していることを前提とします。

- Windows，UNIX の基本的な操作方法
- XMAP3，Cosminexus，TP1/Web を使ったシステム構成や基本操作
- 一般的な Web システムの概要
- COBOL，C 言語または Java でのプログラミング

■ このマニュアルで使用する記号

このマニュアルで使用する記号を次のように定義しています。

記号	意味
[　]	メニュータイトル，メニュー項目，ボタン，キー，およびアイコンの名称を示します。 例：[ファイル] メニュー 　[OK] ボタン 　[Enter] キー など
[　] + [　]	+の前のキーを押したまま，後ろのキーを押すことを示します。 例：[Shift] + [A] キー 　[Shift] キーを押したまま [A] キーを押します。
[　] − [　]	一つ前のメニューを選択し，続けて後ろの項目を選択することを示します。 例：[ファイル] − [開く] 　[ファイル] メニューから [開く] を選択することを示します。
{ }	この記号で囲まれている複数の項目のうちから一つを選択することを意味します。項目が縦に複数行にわたって記述されている場合は，そのうちの 1 行分を選択します。
|	横に並べられた複数の項目に対して項目間の区切りを示し，「または」を意味します。
〔 〕	この記号で囲まれている項目は省略してよいことを意味します。 **(例)** 〔A〕は「何も指定しない」か，「A を指定する」ことを示します。 〔B | C〕では「何も指定しない」か，「B または C を指定する」ことを示します。
△	半角の空白を入れること，またはスペースキーを指定された数だけ押すことを示します。
斜体	インストールフォルダなど可変の個所を示します。
VV-RR-/S	製品のバージョン・リビジョンを示します。XMAP3 Server Runtime Version 5 05-01 では，05 が VV に，01 が RR に該当します。 例：XMAP3 のバージョン（VV-RR-/S）

記号	意味
	以前のバージョン（VV-RR） など

目次

1 XMAP3 の AP の概要

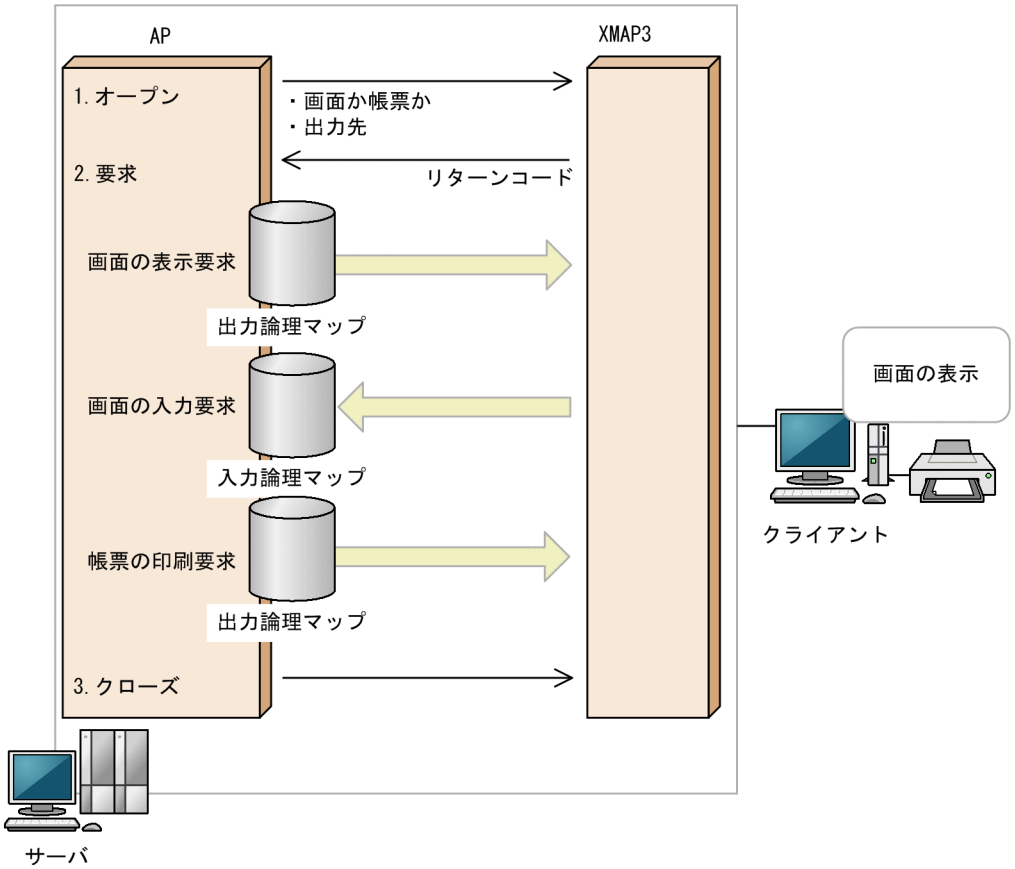
この章では，XMAP3 の AP の処理，および開発の概要について説明します。

1.1　AP の処理の概要

ここでは，XMAP3 のアプリケーションプログラム（AP）を実行したときの処理の概要について説明します。

XMAP3 の AP を実行すると，入出力論理マップをとおして画面の表示，入力，帳票の印刷など，計画した業務処理に従った内容を XMAP3 に要求します。AP と XMAP3 との処理の概要を次に示します。

図 1-1　AP と XMAP3 との処理の概要



1. オープン

   扱うデータは画面用か帳票用か，出力先のディスプレイやプリンタはどこか，などを XMAP3 に渡して，業務を開始します。

2. 要求

   画面の表示，入力，および帳票の印刷要求を行います。AP からデータを渡す場合は，出力論理マップのデータを送信し，AP がデータを受け取る場合は，入力論理マップのデータを受け取ります。

3. クローズ

   業務を終了します。

書式オーバレイ用の AP は，画面・帳票用の AP と処理が異なります。書式オーバレイの処理の概要については，「6.　書式オーバレイでの AP のコーディング方法」を参照してください。

Web 環境用の AP は，業務プログラムとしての AP 以外に通信用の AP が必要です。Web 環境用の AP については，「第 4 編　Web 用の AP 開発」を参照してください。

1.2　AP の開発

XMAP3 の AP は，業務の内容に応じて自由に作成します。XMAP3 では，AP を作成するために次の API を提供しているので，これらを使用します。

- COBOL の命令文
- C 言語用の関数
- 汎用言語用の関数
- Java 用のクラス
- 書式用のコマンド

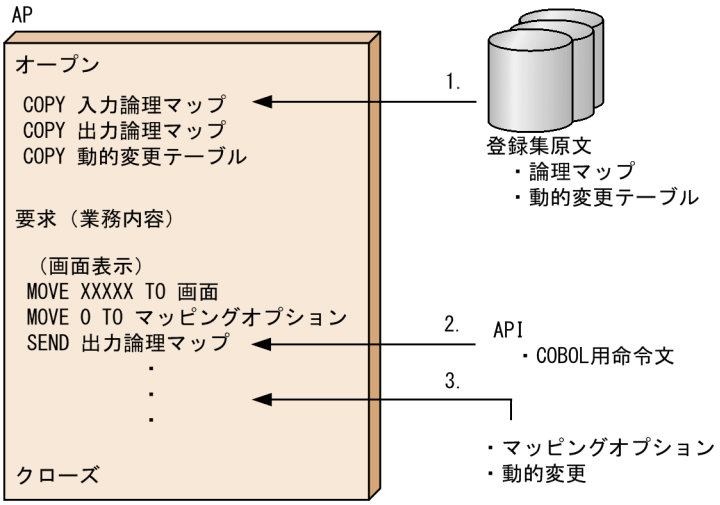
Java 用のクラスについては，XMAP3 Cosminexus 連携用の AP を作成するときにだけ使用できます。XMAP3 Cosminexus 連携用の AP の開発については，「8.　XMAP3 Cosminexus 連携（Java）」を参照してください。

書式用のコマンドについては，書式オーバレイ用の AP を作成するときにだけ使用できます。書式用のコマンドは，Web 実行環境，および Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）では利用できません。書式オーバレイ用の AP の作成については，「6.　書式オーバレイでの AP のコーディング方法」を参照してください。

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は，COBOL の命令文だけ利用できます。

COBOL の命令文の API を使用した AP 開発の概要を次に示します。

図 1-2　COBOL を使用した AP 開発の概要



1. 論理マップや動的変更テーブルを登録集原文として，COPY 文で指定します。
2. 画面表示の要求を XMAP3 の API を使用してコーディングします。
3. 画面を出力するときは画面表示に関係するマッピングオプションを指定します。ほかにも，文字色，書式などを変更する動的変更を指定できます。

使用する API によって，記述する内容は異なります。XMAP3 で作成する AP は，画面を表示する目的で作成する AP（画面用）と，帳票を印刷する目的で作成する AP（帳票用）に分かれます。使用する API は，画面用と帳票用で異なります。画面用の AP 作成に使用できる API については，「第 2 編　画面用の

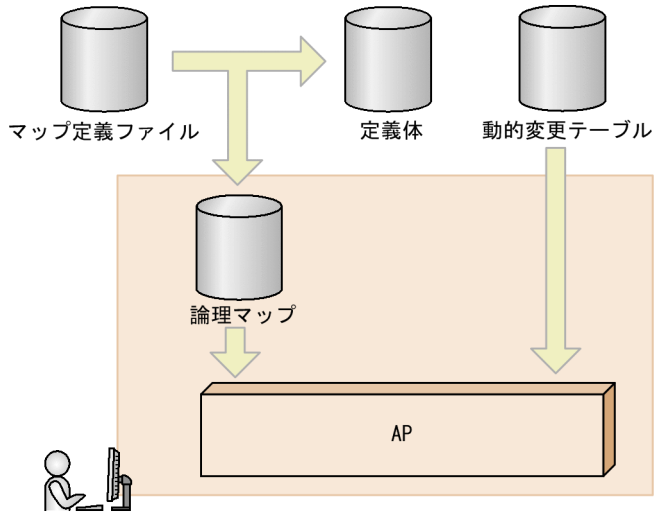
AP 開発」を，帳票用の AP 作成に使用できる API については，「第 3 編　帳票用の AP 開発」を参照してください。

ただし，Web 環境では，画面用または帳票用の AP 以外に，通信用の AP を作成する必要があります。Web 用の AP の作成については，「第 4 編　Web 用の AP 開発」を参照してください。

1.2.1　AP とファイルの関係

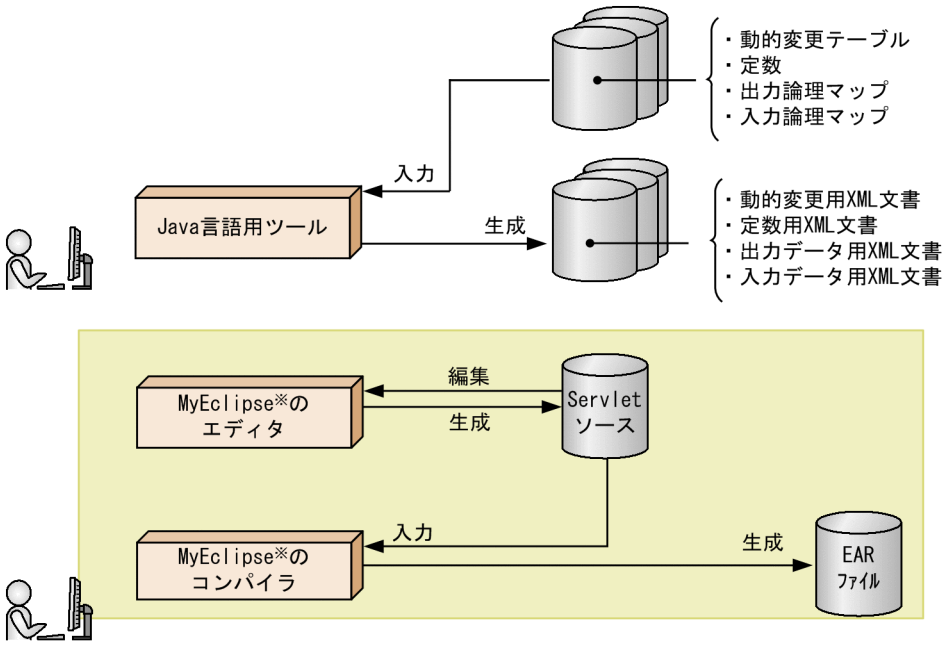
AP には論理マップと動的変更テーブルを組み込む必要があります。XMAP3 の AP を開発するときに関係するファイルの概要を次に示します。

図 1-3　AP を開発するときに関係するファイル



Java で AP を作成する場合，C 言語用のマップを作成したあとに，Java 言語用ツールを使用して，C 言語用マップを XML 文書に変更します。Java で AP を開発するときに関係するファイルを次に示します。

図 1-4　Java で AP を開発するときに関係するファイル



注※　Eclipse，JBuilderも使用できます。

論理マップと動的変更テーブルの作成，および Java 言語用ツールの使用方法については，マニュアル「XMAP3 開発ガイド」を参照してください。

1.2.2　開発環境

XMAP3 では，次に示すプログラミング言語で，AP を作成できます。

- COBOL
- C/C++
- Java

どのプログラミング言語で開発するかは，ドローで指定しておきます。開発言語の指定については，マニュアル「XMAP3 開発ガイド」を参照してください。

ここでは，画面・帳票の開発に必要なソフトウェアを説明します。

(1)　画面・帳票の開発に必要なソフトウェア

スタンドアロンおよび C/S 構成の AP を作成する場合，画面・帳票の開発時に必要なソフトウェアを次の表に示します。

表 1-1　画面，帳票の開発時に必要なソフトウェア

種別	必要なソフトウェア
ユーザプログラム，サーブレットの作成とコンパイル	• C 言語で作成する場合 Visual C++ • COBOL で作成する場合

種別	必要なソフトウェア
	COBOL2002

(2)　XMAP3 Cosminexus 連携（Java）で開発に必要なソフトウェア

画面，帳票の開発時に必要なソフトウェアを次の表に示します。

表 1-2　画面，帳票の開発時に必要なソフトウェア（XMAP3 Cosminexus 連携）（Java）

種別	必要なソフトウェア
ユーザプログラム，サーブレットの作成とコンパイル	uCosminexus Developer Standard， uCosminexus Developer Professional，または uCosminexus Service Architect
	MyEclipse※

注※

MyEclipse のバージョン（VV-RR）は，Cosminexus に依存します。なお，Eclipse，JBuilder も使用できます。

(3)　XMAP3 Cosminexus 連携（COBOL）で開発に必要なソフトウェア

画面，帳票の開発時に必要なソフトウェアを次の表に示します。

表 1-3　画面，帳票の開発時に必要なソフトウェア（XMAP3 Cosminexus 連携）（COBOL）

種別	必要なソフトウェア
ユーザプログラム，COBOL アクセス用 Bean および サーブレットの作成とコンパイル	COBOL2002 Net Server Suite， COBOL2002 Net Client Suite，または COBOL2002 Net Developer※1
	uCosminexus Developer Standard， uCosminexus Developer Professional，または uCosminexus Service Architect
	MyEclipse※2

注※1

COBOL2002 のインストール時に，COBOL2002 の Cosminexus 連携をインストールしてください。

注※2

MyEclipse のバージョン（VV-RR）は，Cosminexus に依存します。なお，Eclipse，JBuilder も使用できます。

(4)　XMAP3 TP1/Web 連携で開発に必要なソフトウェア

画面，帳票の開発時に必要なソフトウェアを次の表に示します。

表 1-4　画面，帳票の開発時に必要なソフトウェア（XMAP3 TP1/Web 連携）

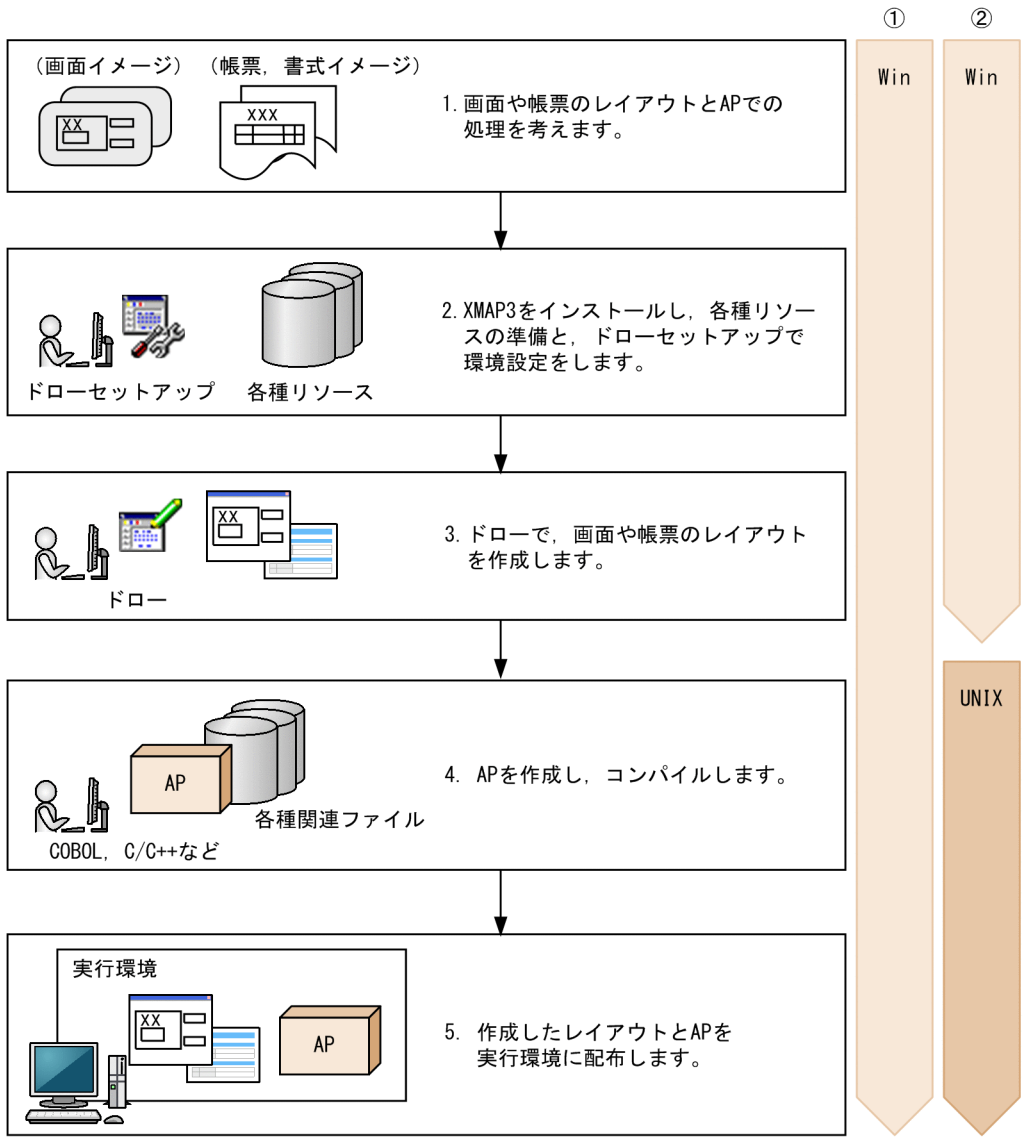
種別	必要なソフトウェア
ユーザプログラムの作成とコンパイル	• C 言語で作成する場合 Visual C++ .NET2003，または Visual C++ 2005 以降

種別	必要なソフトウェア
	• COBOL で作成する場合 COBOL2002

1.3　開発から実行までの流れ

XMAP3 を使用したシステムの，開発から実行までの流れを次の図に示します。

図 1-5　開発から実行までの流れ



（凡例）
①：Windows版XMAP3実行環境（XMAP3 Server Runtime，XMAP3 Client Runtime）の場合の作業環境
②：UNIX版XMAP3実行環境（XMAP3 Server Runtime）の場合の作業環境

Win：Windows版XMAP3開発環境／Windows版XMAP3実行環境で実行する手順を示します。
UNIX：UNIX版XMAP3実行環境で実行する手順を示します。

1. 画面や帳票のレイアウトと AP での処理を考えます。

画面や帳票のレイアウトを設計するときは，次を参照してください。

- マニュアル「XMAP3 開発ガイド」

AP での処理を設計するときは，次を参照してください。

- 「2. 論理マップの概要」

- マニュアル「XMAP3 実行ガイド」

2. XMAP3 をインストールし，各種リソースの準備と，ドローセットアップで環境設定をします。
   インストール，各種リソースの準備，ドローセットアップでの環境設定については，次を参照してください。
   - マニュアル「XMAP3 開発ガイド」
3. ドローで，画面や帳票のレイアウトを作成します。
   ドロー，画面や帳票のレイアウトの作成については，次を参照してください。
   - マニュアル「XMAP3 開発ガイド」
4. AP を作成し，コンパイルします。
   UNIX 版 XMAP3 Server Runtime の場合は，AP 開発をする前にレイアウト作成時に生成したファイルを転送してください。
   画面用の AP 開発およびファイルの転送については，次を参照してください。
   - 「3. 画面での AP のコーディング方法」

   帳票用の AP 開発およびファイルの転送については，次を参照してください。
   - 「5. マップ帳票での AP のコーディング方法」
   - 「6. 書式オーバレイでの AP のコーディング方法」

   Web 用の AP 開発については，次を参照してください。
   - 「8. XMAP3 Cosminexus 連携（Java）」
   - 「9. XMAP3 Cosminexus 連携（COBOL）」
   - 「10. XMAP3 TP1/Web 連携」
5. 作成したレイアウトと AP を実行環境に配布します。
   実行環境の設定については，次を参照してください。
   - マニュアル「XMAP3 実行ガイド」

2 論理マップの概要

この章では，AP とディスプレイ，またはプリンタとの間で受け渡されるデータを格納する領域である論理マップの概要と表示形式について説明します。

2.1　論理マップとは

論理マップとは，AP とディスプレイ，またはプリンタとの間で受け渡されるデータを格納する領域のことで，COBOL の登録集原文または C 言語のヘッダファイルとして記述されています。この節では，論理マップについて説明します。

2.1.1　論理マップの種類と構成

論理マップには，出力論理マップと入力論理マップがあります。

- 出力論理マップ

  AP からディスプレイ，またはプリンタに出力するデータを格納するファイルまたは領域です。ドローでマップを作成したときに，マップ名＋ O という名称でファイルとして出力されます。

- 入力論理マップ

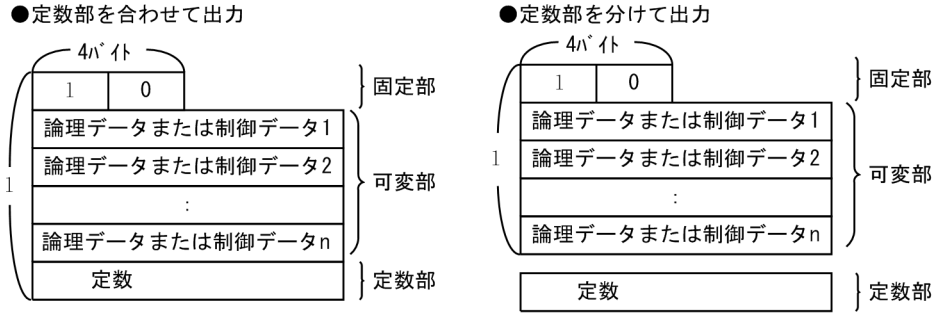
  ディスプレイから表示画面に入力されたデータを格納するファイルまたは領域です。ドローでマップを作成したときに，マップ名＋ I という名称でファイルとして出力されます。

この出力論理マップと入力論理マップは，固定部，可変部，および定数部で構成されます。

定数部は，ドローセットアップの論理マップ属性ダイアログで「定数部への論理マップ長出力」か「定数部の別ファイル出力」を指定したときだけ，生成されます。「定数部の別ファイル出力」を選んだときは，定数部だけが別ファイルに出力されます。

論理マップの構成を次の図に示します。

図 2–1　論理マップの構成



（凡例）l：論理マップ長を示します。

- 論理マップ固定部

  論理マップ固定部は，論理マップの先頭に必ず生成される領域です。この領域には，論理マップ全体の長さである論理マップ長を格納します。

- 論理マップ可変部

  論理マップ可変部は，実際に AP とディスプレイ，またはプリンタとの間で受け渡されるデータを格納する領域です。この領域に格納されるデータには，次に示す論理データと制御データがあります。

  - 論理データ

    AP とディスプレイ，またはプリンタ上の可変項目やメニューなどの各項目（オブジェクト）との間で実際に受け渡されるデータです。

  - 制御データ

ディスプレイでカーソルの位置づけや項目の表示属性を動的変更するとき，またはプリンタで文字色・書体を変更するときなどのように，表示制御を目的としたデータです。

- 定数部

  定数部は，AP を作成するときに制御項目に代入，または参照するカーソル定数やフォーカス定数などの定数をテーブル化したものです。

  なお，表示属性を変更する標準の修飾名はあらかじめ登録集原文として，X3MODTBL が用意されていますので，このテーブルを使って制御項目に代入します。変更した場合，この定数テーブルを基にユーザ責任で追加，変更が必要になります。

2.1.2　論理マップ生成規則で使用する用語

ここでは，論理マップ生成規則で使っている用語と定義方法について説明します。用語と定義方法の対応を次の表に示します。

表 2-1　論理マップ生成規則で使う用語と定義方法の対応

用語	説明	定義方法
再定義名	定義済みのマップを別定義として使用するときの名称。	ドローの画面属性または帳票属性ダイアログの［再定義名...］ボタンで表示される論理マップの再定義名ダイアログで指定する。
修飾名	AP からオブジェクトの出力属性を変更するときに指定する名称。	修飾名とそれに対応する属性は，ドローセットアップの表示属性の動的変更ダイアログで指定する。
埋字	入出力テキスト（フィールド）の場合 画面入力または AP から出力要求されたデータが，ドローで定義した項目の長さより短いときに埋められるデータ。 出力テキスト（フィールド）の場合 AP から出力要求されたデータが，ドローで定義した項目の長さより短いときに埋められるデータ。	初期値は，ドローセットアップの使用目的および詳細目的別データ型（入出力テキスト・フィールド）／（出力テキスト・フィールド）ダイアログで指定する。
桁寄せ	テキストやフィールドに表示されるデータを右または左のどちらかに寄せるための項目。	
出力カーソル項目（2 進）， 出力カーソル項目（10 進）， 論理カーソル項目， 入力カーソル項目（2 進）， 入力カーソル項目（10 進）， 出力フォーカス項目， 入力フォーカス項目	画面の入力などを促すための位置を示す項目。GUI 画面のフィールドボックスではフォーカスおよびカーソルで指定し，それ以外のオブジェクトはフォーカスで指定する。CUI 画面ではカーソルで指定する。	ドローセットアップのカーソルとフォーカスダイアログで指定する。
イベント通知コード	AP に画面からのオペレーション（画面が確定したこと）を通知するためのコード。AP と XMAP3 システム間のインタフェース情報となる。	ドローセットアップのイベント通知コードダイアログで「データ名」，「通知コードの長さ」，および「通知コード」を指定する。
初期クリア文字	各オブジェクトに対して何も入力しなかったとき，論理項目を何でクリアするかの文字。標準では，埋字と同じ文字が仮定される。	ドローセットアップの論理マップ属性ダイアログで指定する。

用語	説明	定義方法
データ消去通知文字	表示データを［End］キー，または［Ctrl］＋［End］キーで消去したり，$(00)_{16}$ のデータを受信したりしたときに，XMAP3 が AP に通知する値。	
エラー通知文字	入力したデータが不当なとき，エラーであることを XMAP3 が AP に通知する値。	
下位項目	一つのオブジェクトの入力または出力領域を階層化して，複数の領域に細分化するための機能。	ドローの各オブジェクトの定義ダイアログで指定する。下位項目はデータ型が「文字（XX）」のときだけ指定できる。

2.1.3　生成される標準のデータ名

論理マップにはドローで設定した情報が格納されています。その情報は，次のような法則で，ドローによって記載されます。

- マップ名＋Ｇ　：　出力論理マップ全体をクリアするときに使用
- マップ名＋Ｋ　：　入力論理マップ全体をクリアするときに使用
- マップ名＋Ｉ　：　入力データ参照用
- マップ名＋Ｏ　：　出力データ代入用
- マップ名＋Ｈ　：　入力データチェック用（スペース，HIGH-VALUE，LOW-VALUE，ほか）
- マップ名＋Ａ　：　表示属性の動的変更制御項目代入用

画面を作成したときに，XMAP3 が標準に生成するデータ名の一覧を次の表に示します。

表 2-2　画面を作成したときに生成される標準のデータ名一覧

対象画面	対象オブジェクト	適用物			データ名	
					入力系データ名	出力系データ名
GUI 画面	－	マップクリア項目			マップ名 K	マップ名 G
		ウィンドウ制御			－	マップ名-CNTRLO
		ウィンドウ位置制御			－	マップ名-WINDOWW
		イベント通知コード			マップ名-INCI	－
		行列（2 進）カーソル	行	1 個目	マップ名-INCURSN	マップ名-OUTCURSL
				2 個目以降	マップ名-INCURSnN	マップ名-OUTCURSnL
			列	1 個目	マップ名-INCURSM	マップ名-OUTCURSC
				2 個目以降	マップ名-INCURSnM	マップ名-OUTCURSnC
		行列（10 進）カーソル	行	1 個目	マップ名-INCURSY	マップ名-OUTCURSY
				2 個目	マップ名-INCURSnY	マップ名-OUTCURSnY

対象画面	対象オブジェクト	適用物			データ名	
					入力系データ名	出力系データ名
				以降		
			列	1 個目	マップ名-INCURSX	マップ名-OUTCURSX
				2 個目以降	マップ名-INCURSnX	マップ名-OUTCURSnX
		論理カーソル		1 個目	マップ名-INCURS-LOCI	マップ名-OUTCURS-LOCO
				2 個目以降	マップ名-INCURSnI	マップ名-OUTCURSnO
		フォーカス制御			マップ名-INFOCUS-I	マップ名-OUTFOCUS-O
		隠しフィールド			マップ名-TRAN-I	−
		確定キーの制御			−	マップ名-INCO
		下位項目※1			マップ名-FIELDnnnn-nnn-I	マップ名-FIELDnnnn-nnn-O
	メニューバー	メニューバー			イベント通知コードに対応	−
		プルダウン				
		カスケード				
		メニュー制御項目			−	マップ名-MENUnnnn-A
	ポップアップメニュー	メニューリスト			−	−
		可変用論理テーブル				マップ名-POPUPnnnn-O
		可変メニューラベル				マップ名-POPUP-LABELnnnn-O
		可変メニューコード				マップ名-POPUP-CODEnnnn-O
		可変メニュー選択ラベル				マップ名-POPUP-TEXTnnnn-O
		可変メニューアクセスキー				マップ名-POPUP-KEYnnnn-O
		ボックス			マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		制御項目			−	マップ名-FIELDnnnn-A
		入力チェック項目			マップ名-FIELDnnnn-H	−
		ポップアップメニューファイル			−	マップ名-POPFILEnnnn-O
	固定テキスト	ボックス			−	−
		制御項目			−	マップ名-FIELDnnnn-A
	出力テキスト	ボックス			−	マップ名-FIELDnnnn-O
		制御項目			−	マップ名-FIELDnnnn-A

対象画面	対象オブジェクト	適用物	データ名	
			入力系データ名	出力系データ名
	入出力テキスト	ボックス	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	プッシュボタン	ボックス	イベント通知コードに対応	−
		ラベルテキスト項目	−	マップ名-BUTTONnnnn-O
		ボタン制御項目		マップ名-BUTTONnnnn-A
	ラジオボタン	ボックス単位通知項目	マップ名-FIELDnnnn-I	−
		可変用論理テーブル	−	マップ名-RADIOnnnn-O
		可変ボタンラベル		マップ名-RADIO-LABELnnnn-O
		可変ボタンコード		マップ名-RADIO-CODEnnnn-O
		固定ボタン制御項目		マップ名-RADIOnnnn-A
		可変ボタン制御項目		マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	チェックボタン	ボタン単位通知項目	マップ名-FIELDnnnn-I	−
		可変用論理テーブル	−	マップ名-CHECKnnnn-O
		可変ボタンラベル		マップ名-CHECK-LABELnnnn-O
		可変ボタンコード		マップ名-CHECK-CODEnnnn-O
		ボタン制御項目		マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	リストボックス	ボックス単位通知項目※2	マップ名-FIELDnnnn-I	−
		可変用論理テーブル	−	マップ名-LISTnnnn-O
		可変リストラベル		マップ名-LIST-LABELnnnn-O
		可変リストコード		マップ名-LIST-CODEnnnn-O
		リスト制御項目		マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	コンボボックス	ボックス	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		可変用論理テーブル	−	マップ名-POPUPnnnn-O

対象画面	対象オブジェクト	適用物	データ名	
			入力系データ名	出力系データ名
		可変メニューラベル		マップ名-POPUP-LABELnnnn-O
		可変メニューコード		マップ名-POPUP-CODEnnnn-O
		制御項目		マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	グラフィック	出力グラフィック	−	マップ名-FIELDnnnn-O
	スピンボックス	ボックス	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
	出力時刻	ボックス	−	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
	出力日付	ボックス	−	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
	入出力時刻	ボックス	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
	入出力日付	ボックス	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		制御項目	−	マップ名-FIELDnnnn-A
FB※3	固定フィールド	固定項目	−	−
	出力フィールド	出力項目		マップ名-FIELDnnnn-O
		出力項目制御		マップ名-FIELDnnnn-A
	入出力フィールド	入出力項目	マップ名-FIELDnnnn-I	マップ名-FIELDnnnn-O
		入出力項目制御	−	マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−
	トグルフィールド	ボックス	マップ名-FIELDnnnn-I	−
		制御項目	−	マップ名-FIELDnnnn-A
		入力チェック項目	マップ名-FIELDnnnn-H	−

（凡例）

−：該当しない。

注 1

この表の「nnnn」は，重複しない 0001 からの数字を自動的に付けます。

注 2

データ名は，COBOL の場合です。C 言語の場合は，−（ハイフン）が__（アンダースコア）になります。

注※1

下位項目は，テキストおよびフィールドで，各オブジェクトの定義ダイアログから設定した場合に生成されます。

注※2

単一選択リストボックスは，ボックス単位の通知項目で，複数選択リストボックスではリスト単位の通知項目となります。この表では，単一選択リストボックスで記述しています。

注※3

FB：フィールドボックス

帳票を作成したときに，XMAP3 が標準に生成するデータ名の一覧を次の表に示します。

表 2-3　帳票を作成したときに生成される標準のデータ名一覧

対象オブジェクト	適用物	出力系データ名
固定フィールド	固定項目	－
出力フィールド	出力項目	マップ名-FIELDnnnn-O
	制御項目	マップ名-FIELDnnnn-A
印刷枚数	－	マップ名-COPIESO
印刷ドキュメント名	出力項目	マップ名-DOCNAMEO
けい線	制御項目	マップ名-FIELDnnnn-A
フレーム	－	マップ名-FRAMEnnnn-O
バーコード	出力バーコード項目	マップ名-BARCODEnnnn-O マップ名-BARCODEnnnn-nnn-O※
OCR	OCR	マップ名-FIELDnnnn-O
グラフィック	出力グラフィック	マップ名-GRAPHnnnn-O

（凡例）

－：該当しない。

注

この表のデータ名は，COBOL の場合です。C 言語の場合は，－（ハイフン）が__（アンダースコア）になります。

注※

連結出力バーコードの場合のデータ名です。

2.1.4　ターゲットでの論理マップの違い

ドローセットアップのターゲットの指定によって，生成する論理マップが異なります。

ドローセットアップの「運用管理者用の設定」で，ターゲットに「Windows 用の画面・帳票開発」を指定した場合，論理マップの展開形式として，リトルエンディアン用，またはビッグエンディアン用のどちらかを設定できます。

ターゲットに「Windows 用の画面・帳票開発」以外を指定した場合，それぞれのターゲットに応じたエンディアンで論理マップを生成します。

2.1.5　使用目的／詳細目的とデータ型

入出力テキスト（フィールド），出力テキスト（フィールド）およびキー入力コンボボックスの使用・詳細目的とデータ型の関係を次に示します。データ型は，COBOL の PICTURE 句で指定するデータの型です。なお，CUI 画面では詳細目的は指定しません。CUI 画面の場合は，「詳細目的」の列を無視してください。

表 2-4　使用目的および詳細目的とデータ型の関係（入出力テキスト・フィールド，キー入力コンボボックス）

使用目的	詳細目的（GUI だけ）	データ型※1	
		COBOL	
		入力データ型	出力データ型
数字※2	数字・数字記号※2	文字（XX）※2	文字（XX）
		99999※2	99999
		99999	ZZZZ9
		その他のピクチャ	その他のピクチャ
	モジュラス	文字（XX）	文字（XX）
		99999	99999
		99999	ZZZZ9
		その他のピクチャ	その他のピクチャ
	数字	文字（XX）	文字（XX）
		99999	99999
金額	数字	文字（XX）	文字（XX）
		99999	99999
	小数点付き	999V9	999.9
		99V99	99.99
		9V999	9.999
		V9999	.9999
	符号付き	S99999	-99999
	符号・小数点付き	S999V9	-999.9
		S99V99	-99.99
		S9V999	-9.999
		SV9999	-.9999
数値	数字	文字（XX）	文字（XX）
		99999	99999
	小数点付き	999V9	999.9

使用目的	詳細目的 (GUI だけ)	データ型※1	
		COBOL	
		入力データ型	出力データ型
		99V99	99.99
		9V999	9.999
		V9999	.9999
	符号付き	S99999	-99999
	符号・小数点付き	S999V9	-999.9
		S99V99	-99.99
		S9V999	-9.999
		SV9999	-.9999
	数字・数字記号	文字（XX）	文字（XX）
カナ※2	カナ・半角※2	文字（XX）	文字（XX）
	カナ		
	カナ・英大・数		
	カナ・英大・数・マイナス		
	カナ・英大・数・長音		
	カナ・英大・数・マイナス・長音		
	カナ・数		
	カナ・数・マイナス		
	カナ・数・長音		
	カナ・数・マイナス・長音		
英数※2	半角※2	文字（XX）	文字（XX）
	英大・カナ・数・記号		
	英大		
	アスタリスク・英大		
	英大・英小		
	英大・数・マイナス		
	英大・数		
	英大・英小・コンマ		
	数・マイナス		
日本語※2	混在（全角・半角）※2	文字（XX）	文字（XX）

使用目的	詳細目的（GUI だけ）	データ型※1	
		COBOL	
		入力データ型	出力データ型
	漢字※2	文字（XX）	文字（XX）
		漢字（NN）	漢字（NN）
		漢字（XX）※3	漢字（XX）※3
パスワード	半角	文字（XX）	文字（XX）
	英大		
	英大・英小		
	英大・数		
MCR	半角	文字（XX）	出力なし
	英大		
	英大・英小		
	英大・数		

注※1

C 言語の場合は，データ型「char」だけ指定できます。

注※2

キー入力コンボボックスで指定できる使用目的，詳細目的およびデータ型です。

注※3

CUI 画面の入出力フィールドの場合にだけ指定できます。

表 2-5　使用目的および詳細目的とデータ型の関係（出力テキスト・フィールド）

使用目的	詳細目的（GUI だけ）	データ型※	
		COBOL	
		入力データ型	出力データ型
数字	数字・数字記号	文字（XX）	文字（XX）
		99999	99999
		99999	ZZZZ9
		その他のピクチャ	その他のピクチャ
	モジュラス	文字（XX）	文字（XX）
		99999	99999
		99999	ZZZZ9
		その他のピクチャ	その他のピクチャ
	数字	文字（XX）	文字（XX）

使用目的	詳細目的 (GUI だけ)	データ型※	
		COBOL	
		入力データ型	出力データ型
		99999	99999
英数	半角	文字（XX）	文字（XX）
	英大・カナ・数・記号		
	英大		
	アスタリスク・英大		
	英大・英小		
	英大・数・マイナス		
	英大・数		
	英大・英小・コンマ		
	数・マイナス		
日本語	混在（全角・半角）	文字（XX）	文字（XX）
	漢字	文字（XX）	文字（XX）
		漢字（NN）	漢字（NN）
		漢字（XX）	漢字（XX）

注※

C 言語の場合は，データ型「char」だけ指定できます。

入出力テキスト・フィールドの使用目的および詳細目的とデータ型の標準値は，ドローセットアップの使用目的別データ型（入出力テキスト・フィールド），使用目的別データ型（出力テキスト・フィールド）で変更することもできます。ただし，表 2-4，表 2-5 以外の組み合わせは指定できません。なお，キー入力コンボボックスのデータ型は変更できません。

2.1.6　ドローセットアップとの関係

ドローセットアップは，画面・帳票定義に関する標準値の変更のためのものです。ドローセットアップでフォルダを指定すると，変更後の標準値（ドローセットアップ情報）を複数用意できます。ドローセットアップで設定した内容が新規に作成するマップ定義ファイルに反映されるため，必ず画面・帳票定義を始める前に済ませておく必要があります。

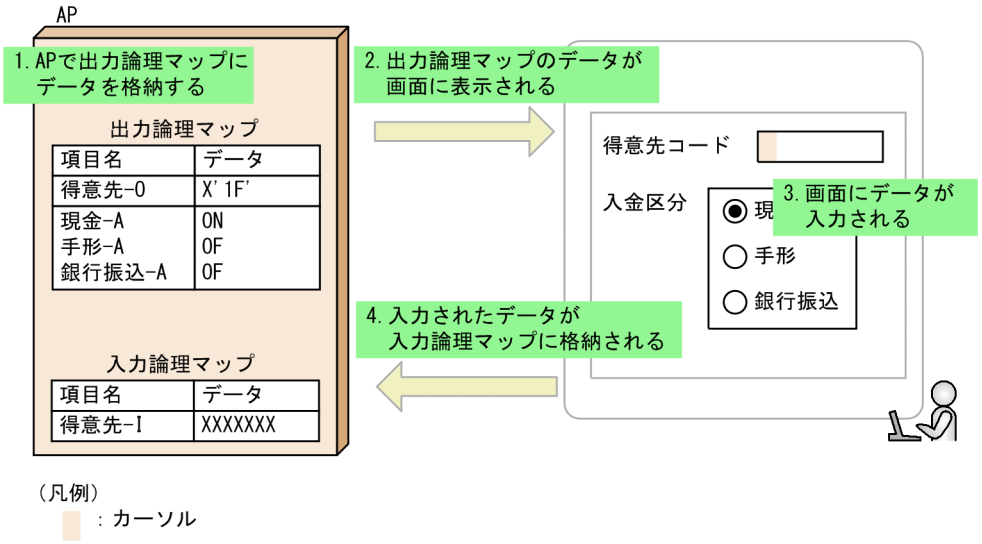
定義したあとに変更する場合は，必ず「セットアップ情報反映」を使用してください。

なお，ドローセットアップについては，マニュアル「XMAP3 開発ガイド」を参照してください。

2.2　論理マップインタフェースの概要

論理マップには，画面に入力されたデータを格納する領域（入力論理マップ）と，画面や帳票に出力するデータを格納する領域（出力論理マップ）があります。画面の入出力と論理マップとの関係を次の図に示します。

図 2-2　画面の入出力と論理マップの関係



1. 画面に表示するデータを AP で出力論理マップに格納します。
2. 出力論理マップに格納されたデータが，画面に表示されます。
3. ユーザによって，画面にデータが入力されます。
4. 入力されたデータが入力論理マップに格納されます。

ここでは，出力論理マップのデータと画面の表示結果の関係，および画面に入力されたデータと入力論理マップに格納されるデータの関係について説明します。

2.2.1　出力論理マップと画面の表示

XMAP3 では，次の条件に基づいて，出力論理マップに格納されたデータを画面に表示します。

XMAP3 では，画面の表示を次の条件で制御しています。

- マッピングオプション

  出力論理マップとレイアウト情報（物理マップ）との関係を決定するオプションです。「マージ」「論理マップだけ」または「物理マップだけ」を AP で指定します。

- 表示形態

  新しく画面を表示するか，直前に表示した画面を表示するか選択する項目です。「全画面書換」「一部上書」または「自動」を，ドローまたはドローセットアップの画面属性ダイアログで指定します。

- 各オブジェクトの属性

  桁寄せ，埋字，使用目的，初期値など，データの画面の表示形式や入力論理マップへの格納形式を決定する項目です。ドローのオブジェクトの属性で指定します。

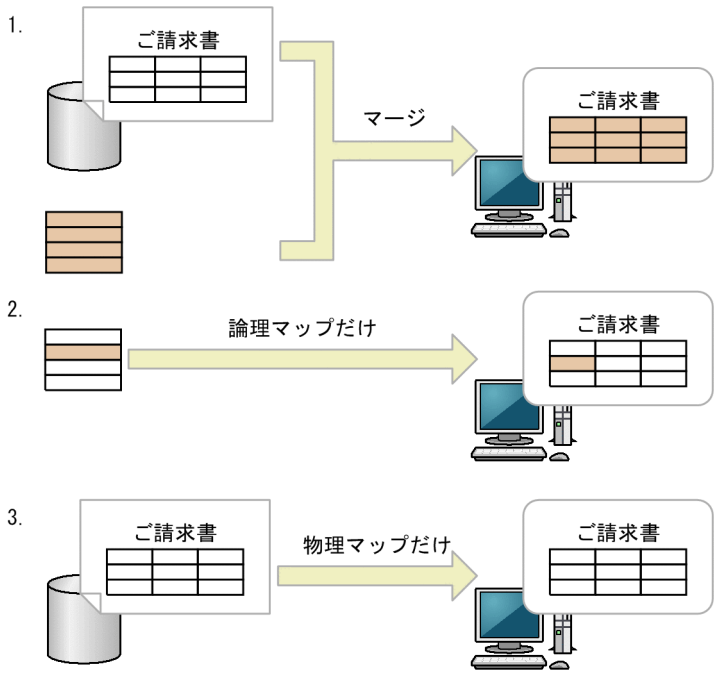
ここでは，マッピングオプションと表示形態について説明します。

(1)　マッピングオプション

マッピングオプションには，「マージ」「論理マップだけ」「物理マップだけ」の三つの指定項目があります。これらは，COBOL で SEND 文および TRANSCEIVE 文を使用したときは MAPPING MODE 句で，COBOL の CALL 文，C 言語，または汎用関数を使用したときはマッピングインタフェースで指定します。

なお，OLTP サーバ構成で使用する場合，「物理マップだけ」は指定できません。画面の表示モードは「マージ」，または「論理マップだけ」を指定してください。

図 2-3　画面の表示モード



1. マージ：

論理マップに代入したデータと物理マップをマージして画面を表示します。同一画面を書き換えるときに使用します。

表示属性の変更要求をしないオブジェクト属性については，画面定義時に指定した表示属性になります。

同じ画面に対し，2 回目以降の表示では固定部分（タイトルやけい線）は再描画しません。

2. 論理マップだけ：

論理マップに代入したデータを使用して画面を表示します。一般的に，2 回目以降の表示で，入力した状態をそのままにして部分的に書き換えるときに使用します※。表示属性の変更指定をしないオブジェクト属性については，直前の表示属性になります。

3. 物理マップだけ：

論理項目，または制御項目に指定したデータを無視して，画面定義時の情報（物理マップ）だけで画面を表示します。論理マップをすべてデータ有無コードでクリアしたときと同じで，メニューなどの初期表示に使用します。

注※

画面属性ダイアログで「入力・選択状態の扱い」の「初期状態」を選んでいる場合，入力データは消去されます。入力データを残したい場合は，「初期状態」以外を選んでください。

(2)　表示形態

表示形態には，「全面書換」，「一部上書」，および「自動」の三つの指定があります。

全面書換

表示中の画面を消去し，次の画面を全画面描画します。

一部上書

直前に表示した画面の，一部のオブジェクトだけを変更して表示します。

なお，直前の画面と異なるマップ名のときは「全面書換」の動作となります。

自動

「一部上書」と「全面書換」を XMAP3 に任せて，特に AP では意識しません。直前の画面と同じマップ名のときは「一部上書」を仮定します。

標準は「自動」です。

(3)　マッピングオプションと表示形態の組み合わせ

指定した表示形態によって，指定できるマッピングオプションが変わります。

マッピングオプションと表示形態の組み合わせを次の表に示します。

表 2-6　マッピングオプションと表示形態の組み合わせ（1 回目：新しいマップ名で表示する場合）

表示形態	マッピングオプション		
	マージ	論理マップだけ	物理マップだけ
全面書換	○	×	○
一部上書	×	×	×
自動	○	×	×

（凡例）

○：指定できる。

×：指定できない。

表 2-7　マッピングオプションと表示形態の組み合わせ（2 回目以降：同じマップ名で表示する場合）

表示形態	マッピングオプション		
	マージ	論理マップだけ	物理マップだけ
全面書換	△	×	×
一部上書	○	○	×
自動	○	○	×

（凡例）

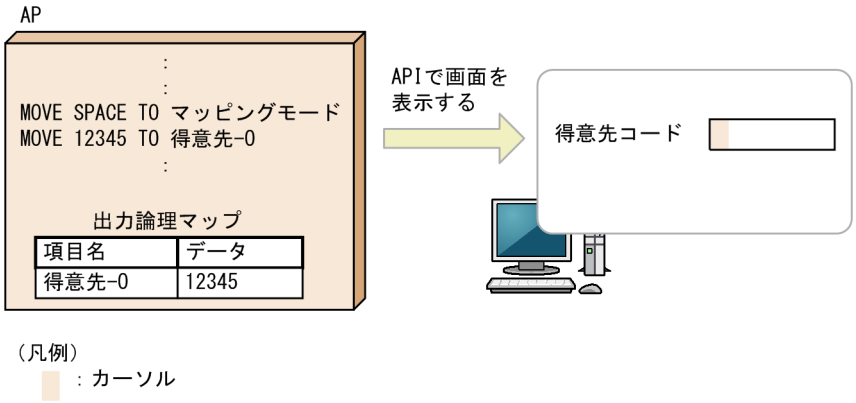
○：指定できる。

△：指定できるが，すべて表示し直すため効率が悪い。

×：指定できない。

2.2.2　出力論理マップと画面への出力（入出力テキストでの例）

ここでは入出力テキストのオブジェクトを使用する場合に，新しい画面を表示するときと，画面に表示されているオブジェクトの一部だけを表示し直すときに分けて，出力論理マップのデータと画面に表示されるデータの関係について説明します。



(1)　新しい画面を表示する場合

新しい画面を表示するには，マッピングオプションの指定によって，次の二つの表示内容に分かれます。

- マッピングオプションにマージを指定する
  新しいレイアウトに論理マップのデータをマージして表示されます。
- マッピングオプションに物理マップだけを指定する
  新しいレイアウトにレイアウトの初期値のデータが表示されます。

ここでは，マッピングオプションの指定項目別に説明します。なお，表示形態には全面書換を指定します。

(a)　マッピングオプションがマージのとき

マッピングオプションをマージに設定すると，物理マップと論理マップの情報をマージして，新しい画面を表示します。ただし，出力論理マップのデータは，オブジェクトの属性によって，画面に表示される結果が変化します。入出力オブジェクトの属性では，桁寄せ，埋字，および初期値の指定によって，画面に表示される結果が異なります。

入出力オブジェクトの桁寄せを「右」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2-8　出力論理マップと画面の表示結果の関係（入出力テキスト）

出力論理マップのデータ	オブジェクトの初期値	画面の表示結果
1 2 3 4 5	すべて	得意先コード 12345
1 2 3 ● ●	すべて	得意先コード ※※123
● 2 3 4 5	指定なし	得意先コード （表示なし）

出力論理マップのデータ	オブジェクトの初期値		画面の表示結果
	LOW(X'00')クリア		得意先コード （表示なし）
	スペースクリア		得意先コード △△△△△
	ゼロクリア		得意先コード 00000
	繰り返し文字の指定 指定文字：@		得意先コード @@@@@
	自由な初期値	ABCDE	得意先コード ABCDE
		ABC	得意先コード ※※ABC
		ABCDEF	得意先コード BCDEF

（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

△：半角スペース

(b)　マッピングオプションが物理マップだけのとき

マッピングオプションを物理マップだけに設定すると，物理マップの情報だけで，新しい画面を表示します。ただし，出力論理マップのデータは，オブジェクトの属性によって，画面に表示される結果が変化します。入出力オブジェクトの属性では，桁寄せ，埋字，および初期値の指定によって，画面に表示される結果が異なります。

入出力オブジェクトの桁寄せを「右」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2-9　出力論理マップと画面の表示結果の関係（入出力テキスト）

出力論理マップのデータ	オブジェクトの初期値	画面の表示結果
1 2 3 4 5 または 1 2 3 ● ● または ● 2 3 4 5	指定なし	得意先コード （表示なし）
	LOW(X'00')クリア	得意先コード （表示なし）
	スペースクリア	得意先コード △△△△△
	ゼロクリア	得意先コード 00000
	繰り返し文字の指定 指定文字：@	得意先コード @@@@@

出力論理マップのデータ	オブジェクトの初期値		画面の表示結果
	自由な初期値	ABCDE	得意先コード ABCDE
		ABC	得意先コード ※※ABC
		ABCDEF	得意先コード BCDEF

（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

△：半角スペース

(2)　直前と同じ画面を表示する場合

直前と同じ画面を表示する場合，マッピングオプションにはマージまたは論理マップだけを，画面の表示形態には一部上書を指定します。出力論理マップのデータは，オブジェクトの属性によって，画面に表示される結果が変化します。入出力オブジェクトの属性では，桁寄せ，埋字，および初期値の指定によって，画面に表示される結果が異なります。

入出力オブジェクトの桁寄せを「右」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2-10　出力論理マップと画面の表示結果の関係（入出力テキスト）

直前の画面	出力論理マップのデータ	オブジェクトの初期値		画面の表示結果
得意先コード 34567	1 2 3 4 5	すべて		得意先コード 12345
	1 2 3 ● ●	すべて		得意先コード ※※123
	● 2 3 4 5	指定なし		得意先コード 34567 （直前に表示された画面の値）
		LOW(X'00')クリア		得意先コード （表示なし）
		スペースクリア		得意先コード △△△△△
		ゼロクリア		得意先コード 00000
		繰り返し文字の指定 指定文字：@		得意先コード @@@@@
		自由な初期値	ABCDE	得意先コード ABCDE
			ABC	得意先コード ※※ABC

直前の画面	出力論理マップのデータ	オブジェクトの初期値		画面の表示結果
			ABCDEF	得意先コード BCDEF

（凡例）

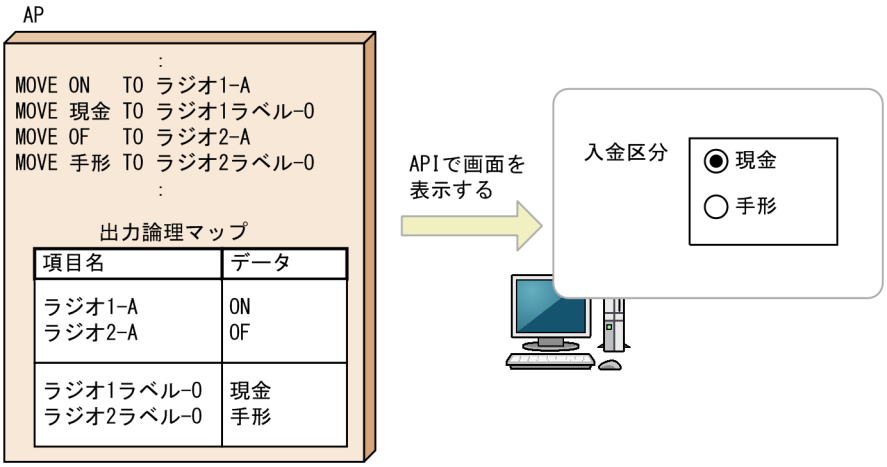
●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

△：半角スペース

2.2.3　出力論理マップと画面への出力（可変ラジオボタンでの例）

ここでは可変ラジオボタンのオブジェクトを使用する場合に，新しい画面を表示するときと，すでに画面に表示されているオブジェクトの一部だけを表示し直すときに分けて，出力論理マップのデータと画面に表示されるデータの関係について説明します。



(1)　新しく画面を表示する場合

新しい画面を表示するには，マッピングオプションの指定によって，次の二つの表示内容に分かれます。

- マッピングオプションにマージを指定する
  ラジオボタンの ON，OFF が出力論理マップのデータに依存して，表示されます。
- マッピングオプションに物理マップだけを指定する
  新しいレイアウトにレイアウトの初期値のデータが表示されます。

ここでは，マッピングオプションの指定項目別に説明します。なお，表示形態には全面書換を指定します。

(a)　マッピングオプションがマージのとき

マッピングオプションをマージに設定すると，物理マップと論理マップの情報をマージして新しい画面が表示されるので，ラジオボタンの ON，OFF が出力論理マップのデータに依存して，表示されます。

可変ラジオボタンのオブジェクトの桁寄せを「左」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2–11　出力論理マップと画面の表示結果の関係（可変ラジオボタン）

出力論理マップのデータ		画面の表示結果 （ラジオボタンの ON と OFF）
ラジオ 1-A	ラジオ 1 ラベル-O	
0 N	現 金 ● ●	入金区分 ◉ 現金※※
● N		入金区分 ○ 現金※※
0 F		入金区分 ○ 現金※※
● F		入金区分 ○ 現金※※

（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

(b)　マッピングオプションが物理マップだけのとき

マッピングオプションを物理マップだけに設定すると，物理マップの情報だけで新しい画面を表示するので，ラジオボタンの ON，OFF は物理マップのデータに依存して，表示されます。

可変ラジオボタンのオブジェクトの桁寄せを「左」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2–12　出力論理マップと画面の表示結果の関係（可変ラジオボタン）

出力論理マップのデータ		画面の表示結果 （ラジオボタンの ON と OFF）
ラジオ 1-A	ラジオ 1 ラベル-O	
0 N	現 金 ● ●	入金区分 ○ 現金※※
● N		入金区分 ○ 現金※※
0 F		入金区分 ○ 現金※※
● F		入金区分 ○ 現金※※

（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

(2)　直前と同じ画面を表示する場合

直前と同じ画面を表示するには，マッピングオプションの指定によって，次の二つの表示内容に分かれます。

- マッピングオプションにマージを指定する
  直前の画面に関係なく，ラジオボタンの ON，OFF が出力論理マップのデータに依存して，表示されます。
- マッピングオプションに論理マップだけを指定する
  直前の画面によって，ラジオボタンの ON，OFF が表示されます。

ここでは，マッピングオプションの指定項目別に説明します。なお，表示形態には一部上書を指定します。

(a)　マッピングオプションがマージのとき

マッピングオプションをマージに設定すると，すでに表示されている画面と論理マップの情報をマージして，同じ画面に異なるデータが表示されるので，ラジオボタンの ON，OFF が出力論理マップのデータに依存して表示されます。

可変ラジオボタンのオブジェクトの桁寄せを「左」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2–13　出力論理マップと画面の表示結果の関係（可変ラジオボタン）

出力論理マップのデータ		画面の表示結果 （ラジオボタンの ON と OFF）
ラジオ 1-A	ラジオ 1 ラベル-O	
0 N	現 金 ● ●	入金区分 ◉ 現金※※
● N		入金区分 ○ 現金※※
0 F		入金区分 ○ 現金※※
● F		入金区分 ○ 現金※※

（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

(b)　マッピングオプションが論理マップだけのとき

マッピングオプションを論理マップだけに設定すると，論理マップの情報だけで，同じ画面に異なるデータが表示されるので，ラジオボタンの ON，OFF が直前の画面と出力論理マップのデータに依存して表示されます。

可変ラジオボタンのオブジェクトの桁寄せを「左」，埋字を「※」に設定したときの，出力論理マップと画面の表示結果の関係について次に示します。

表 2-14　出力論理マップと画面の表示結果の関係（可変ラジオボタン）

直前の画面	出力論理マップのデータ		画面の表示結果（ラジオボタンの ON と OFF）
	ラジオ 1-A	ラジオ 1 ラベル-O	
入金区分 ◉現金※※	0 N	現金●●	入金区分 ◉現金※※
	● N		入金区分 ◉現金※※ （直前の画面表示のまま）
	0 F		入金区分 ○現金※※
	● F		入金区分 ◉現金※※ （直前の画面表示のまま）
入金区分 ○現金※※	0 N	現金●●	入金区分 ◉現金※※
	● N		入金区分 ○現金※※ （直前の画面表示のまま）
	0 F		入金区分 ○現金※※
	● F		入金区分 ○現金※※ （直前の画面表示のまま）

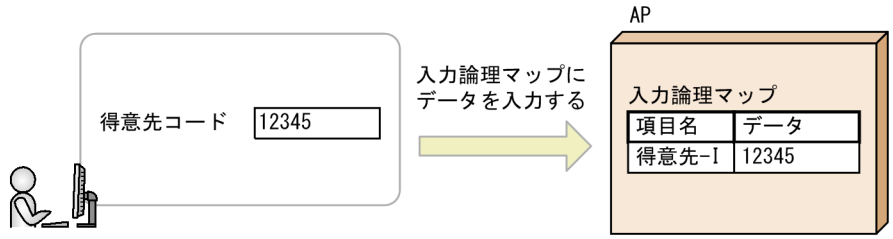
（凡例）

●：データ有無コード（標準値は（1F）$_{16}$）

※：埋字

2.2.4　画面への入力と入力論理マップ（入出力テキストでの例）

ここでは，入出力テキストのオブジェクトを使用してユーザにデータを入力させる場合に，画面に入力されたデータと入力論理マップのデータの関係について説明します。



ユーザにデータを入力させる画面のオブジェクトには，すでに値が入力されているときと入力されていないときがあります。

桁寄せ「右」，使用目的「英数」に設定したときの，画面への入力データと入力論理マップの関係を次に示します。

表 2-15　画面への入力と入力論理マップの関係（入出力テキスト）

オブジェクトの初期値（初期画面）	画面への入力		入力論理マップのデータ
得意先コード [ABCD]	入力あり	得意先コード [12CD] （先頭に入力＝左詰め）	[※][1][2][C][D]
		得意先コード [ABCD] （再度入力した場合）	[※][A][B][C][D]
		得意先コード [　] （文字をクリアした場合）	データ消去文字に指定した値が代入される
	入力なし	得意先コード [ABCD]	初期クリア文字に指定した値が代入される
得意先コード [　]	入力あり	例： 得意先コード [ABCD]	[※][A][B][C][D]
	入力なし	得意先コード [　]	初期クリア文字，またはデータ消去文字に指定した値が代入される

（凡例）

※：埋字

3 画面での AP のコーディング方法

この章では，画面用の AP を作成する方法について説明します。

3.1　XMAP3 の画面用 AP で実行する処理

ここでは，XMAP3 の画面用 AP で実行する処理の概要と，コーディングについて説明します。

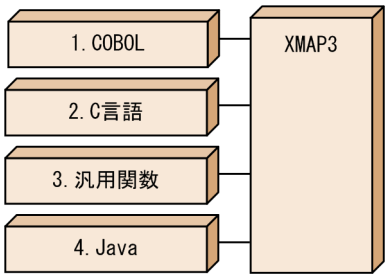
なお，XMAP3 の API はマルチスレッドで動作する AP からの実行に対応していません。マルチスレッドで動作する AP では，XMAP3 の API を使わないようにしてください。

3.1.1　XMAP3 での画面の入出力

(1)　画面用 AP で使用する API の種類

XMAP3 には, AP の開発に使用するプログラミング言語ごとに API を提供しています。API の種類を次の図に示します。

図 3–1　XMAP3 の API の種類



解説

1. COBOL の命令文です。SEND 文，RECEIVE 文，および TRANSCEIVE 文を使用する方法と CALL 文を使用する方法があります。Windows/UNIX で共通の文法です。
2. XMAP3 が提供している C 言語用の関数です。Windows/UNIX で共通の文法です。
3. XMAP3 が提供している Windows 専用の C 言語または C++用の汎用関数です。
4. XMAP3 が提供している Java 用のクラスです。Web 環境で使用できます。使用方法については，「第 4 編　Web 用の AP 開発」を参照してください。

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は，COBOL AP（SEND 文，RECEIVE 文，および TRANSCEIVE 文を使用する方法）だけ利用できます。

(2)　画面用 AP で使用する API の概要

画面用 AP で使用する要求と API の関係を次の表に示します。また，画面用 AP と XMAP3 との処理の概要を次の図に示します。

表 3–1　画面用 AP で使用する要求と API の関係

要求	COBOL の命令文	COBOL の CALL 文	C 言語	汎用関数
オープン※	最初の SEND または TRANSCEIVE 文発行時	'OPEN' 'MDO△' 要求	'OPEN' 'MDO△' 要求	XmapDrvCreateOpen
表示	SEND 文または TRANSCEIVE 文	'SEND' 要求	'SEND' 要求	XmapDrvSend

要求	COBOL の命令文	COBOL の CALL 文	C 言語	汎用関数
入力	RECEIVE 文または TRANSCEIVE 文	'RECV' 要求	'RECV' 要求	XmapDrvReceive
クローズ	DISABLE 文, GOBACK 文, EXIT PROGRAM 文, CANCEL 文 （AP の）STOP RUN	'CLOS' 要求	'CLOS' 要求	XmapDrvClose

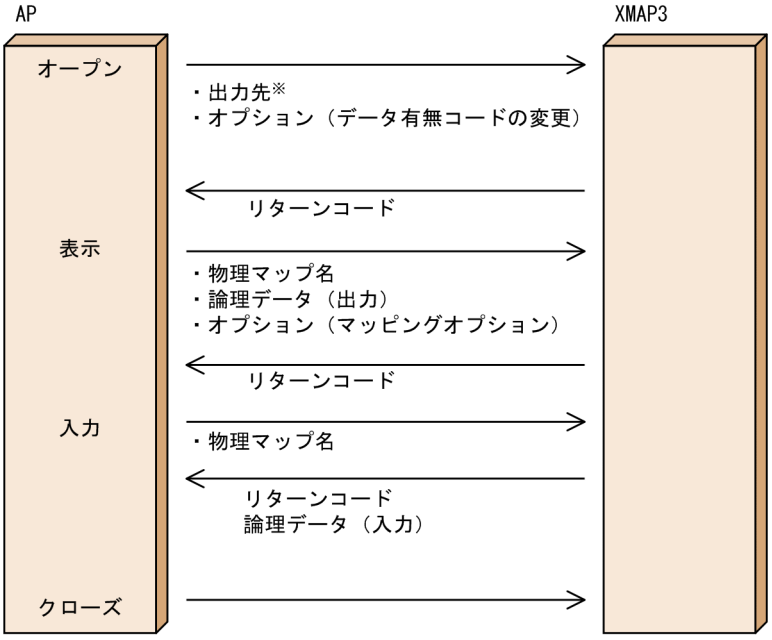
注

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は，COBOL AP（SEND 文，RECEIVE 文，および TRANSCEIVE 文を使用する方法）だけ利用できます。

注※

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

図 3–2　画面用 AP と XMAP3 との処理の概要



注※　必ず指定します。

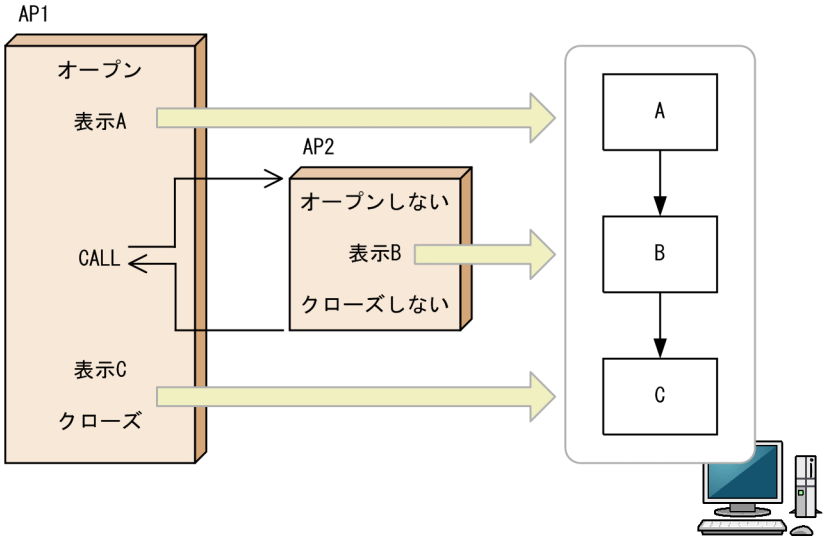
3.1.2　AP 間でオープンを引き継ぐ場合

メッセージを出す共通部分を別 DLL 化したり，プログラムが大きいので分割したりする場合などは，オープンをプログラム間で引き継がせることをお勧めします。COBOL の SEND/RECEIVE/TRANSCEIVE 文を使用した場合，AP 間でオープンを引き継ぐことができます。オープンを引き継ぐためには，COBOL の場合，プログラムを実行させる AP（ここでは AP1）で環境変数「CBLTERMSHAR=YES」を指定しておきます。

COBOL の CALL 文，C 言語，および汎用関数を使用する場合は，インタフェース領域を受け渡しすることで，AP 間のオープンを引き継ぐことができます。

オープンを引き継ぐことによって，AP1 から AP2 を呼び出すとき，AP1 でのクローズ（COBOL のとき DISABLE 文の発行）および AP2 でのオープンは発生しません。そのため画面遷移の時間が短くなります。

図 3-3　AP 間でオープンを引き継ぐ場合



3.1.3　AP 分割時の注意

COBOL では，明示的に CALL 文でオープンを要求する場合を除き，一つのコンパイル単位で最初の SEND 文または TRANSCEIVE 文が発行されると画面がオープンされます。そのため，1 画面が 1 実行ファイル（.exe）のような構成にすると，画面ごとにオープンをすることになります。この場合，オープン・クローズ，SEND 文または TRANSCEIVE 文を発行する実行ファイル，ビジネス処理をする実行ファイルを分けると実行性能が高くなります。ビジネス処理をする実行ファイルは，DLL ファイルでも代用できます。EXE ファイルにするか DLL ファイルにするかは，処理の形態によって次のように使い分けてください。

- .exe：メニュー画面など処理が続く場合（EXE ファイル間でのデータ受け渡しはない）
- .dll：処理が単独で閉じる場合（1 ウィンドウで閉じる）

また，複数のコンパイル単位のものを合わせて一つの実行ファイルにするときは，各コンパイル単位でオープンを発行しないようにするため，COBOL の実行支援の環境変数で，「CBLTERMSHAR=YES」を指定します。「CBLTERMSHAR=YES」は，SEND／RECEIVE／TRANSCEIVE 文で AP を作成したときだけ有効です。CALL 文で AP を作成したときは無効になります。

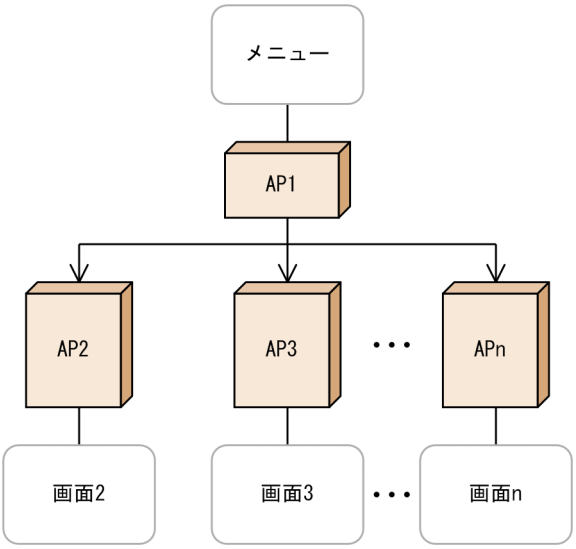
CALL 文で AP を作成する場合，AP 間のオープン引き継ぎをするときは，ユーザが，XMAP3 インタフェースエリアの情報を引き継ぐようなコーディングをする必要があります。このコーディングのひな型として，AP パターンの GENDSP02 および GENDSP03 を利用できます。

(1)　メインの実行ファイルから別の実行ファイルを呼び出す場合

メインメニュー（AP1）から各サブプログラム（AP2～n）を呼び出すケースを次の図に示します。

- AP1，AP2～n の各 AP で，オープン～クローズを行う。
- AP1 から AP2～n を呼び出すときは，表示している画面を一度消してから，新しい画面を表示し直す。

図 3-4　メインメニュー（AP1）から各サブプログラム（AP2～n）を呼び出すケース

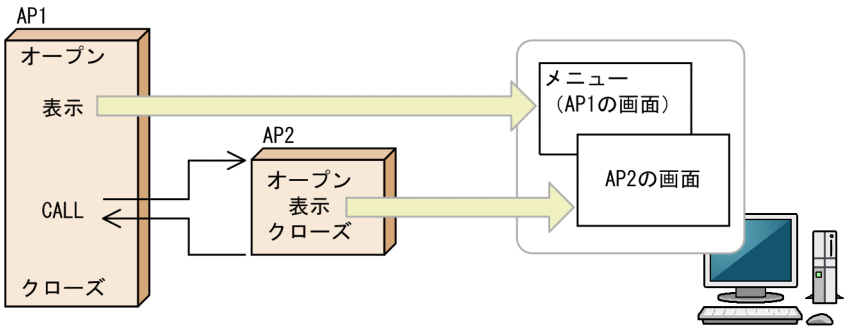


(2)　新しい画面を表示するとき前の画面を消さないで表示しておくケース

AP1 から AP2～n を呼び出すとき，AP1 の画面を消さないで表示しておくケースを次の図に示します。ただし，この方法では動きが遅くなります。

- AP1 で画面をオープン→AP2～n を呼び出す
- AP2～n ではオープン～クローズを行う

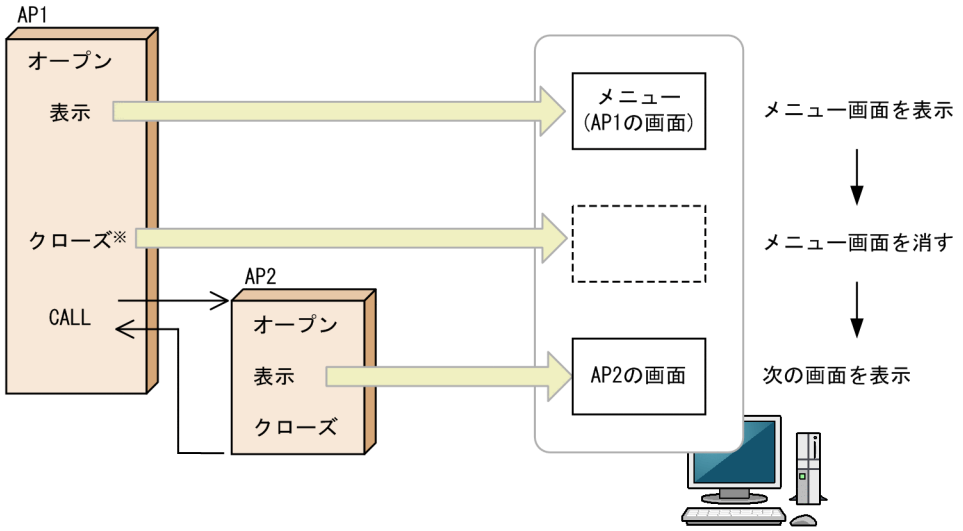
図 3-5　AP1 から AP2～n を呼び出すとき，AP1 の画面を消さないで表示しておくケース



(3)　新しい画面を表示するとき前の画面を消す場合

AP1 から AP2～n を呼び出すとき，AP1 の画面を消して AP2 を表示するケースを次の図に示します。ただし，この方法では動きが遅くなります。

図 3-6　AP1 から AP2～n を呼び出すとき，AP1 の画面を消して AP2 を表示するケース

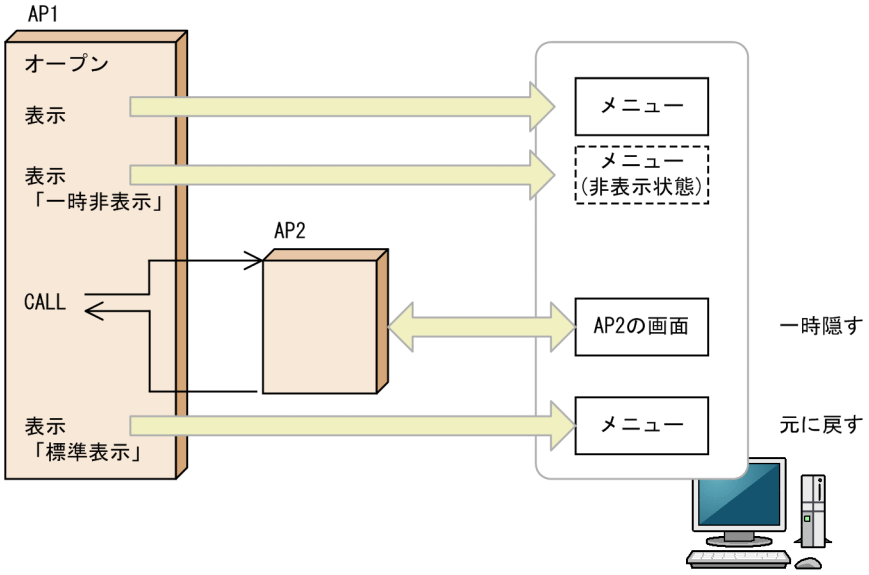


注※　DISABLE文を発行（COBOLだけ）

(4)　画面をクローズしないで一時的に表示中の画面を消す場合（Windows）

画面をクローズすると，次に表示するときには必ずオープンしなければならないので，この間のオープン／クローズの時間が掛かります。このような時間を節約したいときには，GUI 画面の画面属性ダイアログの「Z 位置」で「後ろに表示」か「一時非表示」を指定します。これらの指定をすると，一時的に表示中の画面を見えないようにできます。いちばん後ろに表示された画面を前に表示させたり，一時的に非表示になった画面を元に戻す場合は，ドローセットアップの「表示属性の動的変更」の「位置属性」タブで「XY 位置」を「手前に表示」と指定します。

図 3-7　画面をクローズしないで一時的に表示中の画面を消す場合



(5)　一つの表示サービスに複数画面表示する場合（UNIX）

C/S および OLTP（TP1/NET/XMAP3）構成で，一つの表示サービスに対して複数の画面を表示する場合，同時に表示できる画面数（GUI 画面の場合は，一次／二次ウィンドウの総数）は 15 個までです。16 個以上の画面表示を要求するとエラー（ログ情報に出力される詳細コードが「0x01001303」エラー）となり，16 個目からの画面は表示されませんので注意してください。ログ情報に出力される詳細コードについては，「付録 F　リターンコードと詳細コード」を参照してください。

3.1.4　AP での入力論理マップ上のデータチェック

(1)　イベント通知コードのチェック

画面確定時のイベント通知コード（例えば，ファンクションキー 1 では PF01，ボタンでは A001）を返します。イベント通知コードの値は，ドローセットアップで変更できます。なお，イベント通知コードが AP に渡るタイミングは入力単位の指定（画面確定のタイミング）によって異なります。それぞれの説明については，マニュアル「XMAP3 開発ガイド」を参照してください。

- **画面属性ダイアログの入力単位で「画面」を指定している場合**
  画面上で，一つだけ活性になっているオブジェクトを選択したり，データを入力したりしたあと，画面確定を設定しているメニューバー，プッシュボタンまたは画面確定キーを押すと，AP に制御が渡ります。また，「自動送信」を指定しているオブジェクトからフォーカス・カーソルが離脱したタイミングでも AP に制御が渡ります。
- **画面属性ダイアログの入力単位で「フィールド」を指定している場合**
  画面上で活性になっている，フォーカス・カーソルのあるオブジェクトを選択したり，データを入力したりしたあと，フォーカス・カーソルが次のオブジェクトに移動するタイミングで AP に制御が渡ります。
- **画面属性ダイアログの入力単位で「イベント」を指定している場合**
  画面上で活性になっているすべてのオブジェクトから一つを選択したり，データを入力したりしたあと，フォーカス・カーソルがほかのフィールドに移動するタイミングで AP に制御が渡ります。

(2)　入力データのチェック

XMAP3 では，画面から入力されたデータを使用目的に従ってチェックします。エラーがあれば，入力時にエラーがあると即座に画面に返す場合と，設定に従ったエラー通知文字やデータ有無コードを AP に返し，処理を続行する場合があります。

画面から入力したデータ以外で，AP に返すデータは次のとおりです。なお，データが返されているかどうかは「**データ項目名＋ H**」のデータ名で参照します。

(a)　エラー通知

例えば，カナ文字を扱うテキスト・フィールドに対して全角文字（漢字）を入力した場合に，エラー通知文字（HIGH（X'FF'））を AP に返します。

エラー通知文字は，ドローセットアップの「論理マップ属性」で変更できます。

(b)　データ消去の通知

例えば，**フィールドキー**（[End] キー）でデータ消去した場合に，データ消去通知文字（LOW（X'00'））を AP に返します。

データ消去通知文字は，ドローセットアップの「論理マップ属性」で変更できます。

(c)　データ未入力の通知（初期値を返す場合）

例えば，テキスト・フィールドで**初期値**に ABC の文字を設定し，データの入力をしない場合に初期値の ABC を AP に返します。

(3)　データ未入力の通知（初期クリア文字を返す場合）

初期値の文字を設定せず，データの入力をしない場合に**初期クリア文字**（埋字）を返します。

例えば，使用目的が「英数」の場合は埋字（スペース）を，「数字」の場合は埋字（ゼロ）を AP に返します。

初期クリア文字は，ドローセットアップの「論理マップ属性」で変更できます。

(4)　フォーカス・カーソル位置のチェック

画面確定時に，どのオブジェクトにフォーカス・カーソルが位置づいているかをチェックできます。なお，フォーカス・カーソルについては，マニュアル「XMAP3 開発ガイド」を参照してください。

(a)　フォーカス位置の通知（GUI 画面の場合）

画面確定時のフォーカス位置を，フォーカス定数で返します。

フォーカス定数は，例えばドローで AP が受け取る項目のデータ名を MAP001-FIELD0001-に設定した場合は，MAP001-FIELD0001-T の名称で論理マップ中に生成されます。フォーカス定数と入力フォーカス項目の値が一致するオブジェクトの位置が，画面上に位置づいたフォーカスの位置として AP で認識されます。

(b)　カーソル位置の通知

画面確定時のカーソル位置を，カーソル定数で返します。

カーソル定数は，例えばドローで AP が受け取る項目のデータ名を MAP001-FIELD0001-に設定した場合は，MAP001-FIELD0001-T の名称で論理マップ中に生成されます。カーソル定数と入力カーソル項目の値が一致するオブジェクトの位置が，画面上に位置づいたカーソルの位置として AP で認識されます。

ドローセットアップの「カーソルとフォーカス」で行列（2 進）カーソルや行列（10 進）カーソルに変更できます。ドローセットアップでの設定については，マニュアル「XMAP3 開発ガイド」を参照してください。

3.1.5　AP 作成時の注意事項

(1)　［閉じる］ボタンを使用する場合

XMAP3 の画面で［閉じる］ボタンまたは［閉じる］メニューが操作された場合，XMAP3 では，直接ウィンドウを閉じずに AP に対して［閉じる］ボタンまたは［閉じる］メニューが操作されたことを通知します。このため，AP では，このイベントに対して，次のような処理をしてください。

なお，通知するイベント種別は，［Break］キーに対応するイベント通知コードです。

処理の例

- 通知コードが返ってきたとき，AP で必要な終了処理をする。
- 直ちに終了すると不都合が発生する場合は，サブウィンドウで確認をして，その応答内容によって，終了するかどうかを決定する。

イベントに対応する処理がない場合は，［閉じる］ボタンを使用しても，AP を終了できません。また，「閉じる」処理に対応していない AP を使用する場合は，［閉じる］ボタンおよび［閉じる］メニューを表示しないで運用してください。

なお，ウィンドウ上の［閉じる］ボタンやコントロールメニューの［閉じる］を使用する場合は，表示・印刷セットアップの「デザイン 1」タブで指定します。

(2)　［PF］キーなどに「中断」を割り当てた場合

表示・印刷セットアップの「キー操作 1」タブの「強制確定キーの動作」で，「入力済みデータを送らない」を指定した場合，その設定は［PA1］～［PA3］キー，［BREAK］キー，［SCREEN］キーだけが対象となります。

ドローで［PF］キーなどに「中断」を割り当てた場合は，そのキーを押すと画面上のデータも送られます。入力必須項目や，日付・時刻・金額項目など，通常はデータの妥当性がチェックされる項目についても，チェックがされていない状態のデータが返るので，「中断」を使用する場合には，必ず AP で「中断」イベントが起こっていないことを確認してからデータを処理してください。

(3)　AP に制御が渡る前の画面を表示する場合

表示画面から強制確定キーによって AP に制御が渡るとき，金額，数値，日付，時刻の入出力テキストおよびフィールドについては，表示されていた画面上のデータは保証されません。再び同じ画面を表示する場合は，明示的にそれぞれのオブジェクトのデータを設定するか，画面属性ダイアログの「入力・選択状態の扱い」を「初期状態」にして表示してください。

(4)　ポップアップメニューの動的変更をする場合

ポップアップメニューの通知コードを変更する場合やメニューの数を少なくする場合で，動的変更を実施する前に選択されていたメニューに対する通知コードが動的変更後になくなるようなときは，AP から設定する通知コードをクリアするか，存在するメニューの通知コードを設定して動的変更を実施してください。

3.2　マッピングオプション

マッピングオプションには，「マージ」「論理マップだけ」「物理マップだけ」の三つの指定があります。

マッピングオプションは，AP 作成時に次に示す個所に指定します。

COBOL の SEND／RECEIVE／TRANSCEIVE 文を用いる場合
　通信記述項の MAPPING MODE 句

COBOL の CALL 文を用いる場合
　マッピングインタフェース領域（XMAP-MDO）

C 言語を用いる場合
　マッピングインタフェース領域（XMAP_MDO）

汎用関数を用いる場合
　XmapDrvSetMapOption()の第 2 引数（mode）

マッピングオプションは次の項目で有効になります。

- 可変項目（入出力テキスト／出力テキスト，入出力フィールド／出力フィールド）
- 項目属性（動的変更項目）
- ポップアップ，テキストボックス，コンボボックス，スピンボックス
- プッシュボタン
- ラジオボタン，チェックボタン，リストボックス，トグルフィールド
- 出力グラフィック
- 入出力日付テキスト（フィールド），入出力時刻テキスト（フィールド）

マッピングオプションの説明を次の表に示します。

表 3-2　マッピングオプション

目的		マッピングオプション	MAPPING MODE 句の指定値 (COBOL)	マッピングインタフェース領域 (COBOL)	マッピングインタフェース領域 (C 言語)	XmapDrvSetMapOption()の第 2 引数 (汎用関数)	説明
同じ画面に対して 2 回目以降の表示方法	1 回目または 2 回目でユーザデータを表示したいとき	マージ (論理マップと物理マップをマージ)	空白，または 0	XMAP_MDO_MAPFLD	XMAP-MDO-MAPFLD	0	論理項目，または制御項目に指定したデータを使って画面を表示する。ただし，表示属性の変更指定をしない項目属性については，画面定義時に指定した表示属性となる。同一画面をすべて書き換える（初期化）ときに使用する。
	AP からのデータだけで部分書き換	論理マップだけ	2	XMAP_MDO_LOGFLD	XMAP-MDO-LOGFLD	2	論理項目，または制御項目に指定したデータを使って画面を表示する。ただし，表示属性の変更指定をしない項

目的		マッピングオプション	MAPPING MODE 句の指定値 (COBOL)	マッピングインタフェース領域 (COBOL)	マッピングインタフェース領域 (C 言語)	XmapDrvSetMapOption()の第 2 引数 (汎用関数)	説明
	えをしたいとき						目属性については, 直前の画面の表示属性になる。一般的には，3 回目以降の表示で, 入力した状態をそのままにして部分書き換えをしたいときに使用する。
1 回目（初期）表示用出力するユーザデータなし		物理マップだけ	3	XMAP-MDO-PHFLD	XMAP_MDO_PHFLD	3	論理項目, または制御項目に指定したデータを無視して, 画面定義時の情報だけで画面を表示する。初期表示したいときに使用する（メニュー表示などに使用する)。

3.3　COBOL での画面入出力命令

ここでは，COBOL で画面用の AP を作成する方法について説明します。

3.3.1　コーディング前の準備（Windows）

コーディングをする前に，次の準備をしてください。

1. **AP を格納するためのフォルダを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するフォルダを作成します。開発環境に合わせてフォルダを分類し，作成位置や名称を決めます。

2. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要な登録集原文を標準提供しています。1.で作成したフォルダにコピーして使用することをお勧めします。このとき，コーディングのひな型である AP パターンもコピーしておきます。

   標準提供の登録集原文を次に示します。

   - **動的変更テーブル**（X3MODTBL.CBL）

     動的変更用の修飾名と出力論理マップの初期化に使用する定数が指定されています。

   - **インタフェース領域**（JSVWATBL.CBL）

     CALL 文で画面を送受信するときにパラメタとして使用するインタフェース領域が指定されています。

3.3.2　コーディング前の準備（UNIX）

コーディングをする前に，次の準備をしてください。

1. **XMAP3 Developer で作成したファイルを転送する**

   XMAP3 Developer で UNIX 用に作成した物理マップ，論理マップおよび動的変更テーブルを UNIX マシンへファイル転送します。

   - **物理マップのファイル転送**

     XMAP3 Developer で UNIX 用に作成した物理マップ（拡張子「.pmp」）ファイルは，バイナリ形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

   - **論理マップ・動的変更テーブルのファイル転送**

     XMAP3 Developer で UNIX 用に作成した論理マップ（拡張子「.cbl」）ファイルおよび動的変更テーブル（X3MODTBL.cbl）ファイルは，テキスト形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください。

     XMAP3 Developer で作成する動的変更テーブルは，標準の出力先のままで作成した場合，次に出力されています。別の出力先に作成した場合は，そのファイルをファイル転送してください。

   XMAP3インストールフォルダ¥INCLUDE¥X3MODTBL.cbl

2. **物理マップを移行し形式チェックをする**

   XMAP3 Developer で作成した UNIX 用の物理マップを，UNIX マシンへ転送したあと，cmapcp コマンドを使って，物理マップを実行環境に移行します。

cmapcp コマンドは，次に示すファイルを実行環境に移行し，さらに UNIX 版 XMAP3 で使用できる物理マップの形式かどうかのチェックをします。

- 拡張子が「.pmap」の物理マップファイル
- 拡張子が「.pmp」の物理マップファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

cmapcp コマンドについては，「15.1.1　cmapcp コマンド」を参照してください。

3. **UNIX の EUC 環境で利用する場合，論理マップおよび動的変更テーブルを文字コード変換する**

   XMAP3 Developer で生成される論理マップおよび動的変更ファイルは，文字コードがシフト JIS であるため，UNIX の EUC 環境で利用するには，ファイルを転送したあと，シフト JIS から EUC へコード変換する必要があります。

4. **AP を格納するためのディレクトリを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するディレクトリを作成します。開発環境に合わせてディレクトリを分類し，作成位置や名称を決めます。

5. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要な登録集原文を標準提供しています。4.で作成したフォルダにコピーして使用することをお勧めします。このとき，AP の作成に使う論理マップ，コーディングのひな型である AP パターン，AP 実行時に使用する物理マップもコピーしておきます。

   - **動的変更テーブル**（X3MODTBL.CBL）

     AP の実行中に画面の色などを変更する表示属性や，出力エリア（出力論理マップ）の初期化に使用する定数が指定されています。なお，XMAP3 Developer で UNIX 用の動的変更テーブルを作成した場合は，標準提供のテーブルは使用しないでください。必ず，XMAP3 Developer で作成した動的変更テーブルをファイル転送して使用してください。

   - **インタフェース領域**（JSVWATBL.CBL）

     CALL 文で画面を送受信するときにパラメタとして使用するインタフェース領域が指定されています。SEND 文，RECEIVE 文，または TRANSCEIVE 文を使う場合は必要ありません。

3.3.3　仮想端末の自動割当て

C/S 構成の環境で，サーバ上の AP から複数のクライアントマシンへ画面表示をする場合，仮想端末の自動割当て機能を利用できます。この機能は，C/S 構成の場合だけ利用できます。

仮想端末の自動割当て機能は，サーバ AP が各クライアントマシンに対応した仮想端末名を意識することなく一つの仮想端末名だけを意識していれば，各クライアントマシンにある XMAP3 の表示サービスが起動されたタイミングで，AP から送信された情報を画面表示するようにできます。仮想端末の自動割当て機能を利用するときはサーバ側で，AP で指定した自動割当て用の仮想端末名を設定します。

仮想端末の自動割当てについては，マニュアル「XMAP3 実行ガイド」を参照してください。

3.3.4　論理マップの取り込み方法（COBOL）

AP 中に論理マップを取り込む場合，COBOL の WORKING-STORAGE SECTION，または LINKAGE SECTION に COPY 文を指定します。ただし，論理マップ中に定数を展開している場合，LINKAGE SECTION には取り込めません。

AP に論理マップを取り込む例を次に示します。

取り込みの例

```
WORKING-STORAGE SECTION.
  COPY MAP0030.※1    …… 出力論理マップの取り込み
  COPY MAP003I.※1    …… 入力論理マップの取り込み
  COPY X3MODTBL.※2    …… 動的変更テーブルの取り込み
```

注※1

新規作成ダイアログのマップ名に，出力用は「O」，入力用は「I」を付けたものです。ここではマップ名に MAP003 を指定した場合の例になります。

注※2

AP から文字色などの表示属性を変更するときに使用するデータが格納されたテーブルです。

3.3.5　画面表示命令（COBOL）

スタンドアロン環境，および C/S システム環境下では，次に示す方法で AP から画面の入出力ができます。

1. COBOL の SEND 文，RECEIVE 文，および TRANSCEIVE 文による方法
2. COBOL の CALL 文による方法

Windows 版 XMAP3 サーバ/クライアント実行環境(64 ビット)の場合, COBOL の SEND 文, RECEIVE 文, および TRANSCEIVE 文による方法は利用できます。COBOL の CALL 文による方法は利用できません。

(1)　SEND 文，RECEIVE 文，および TRANSCEIVE 文による方法

COBOL の SEND 文，RECEIVE 文，および TRANSCEIVE 文によって画面の入出力を要求します。通信記述項，SEND 文，RECEIVE 文，および TRANSCEIVE 文については，「11.1.1　画面表示命令(COBOL)」を参照してください。

COBOL については，マニュアル「COBOL2002 言語 拡張仕様編」を参照してください。

(2)　CALL 文による方法

ここでは，COBOL の CALL 文によってマッピングライブラリを使用する方法を説明します。

(a)　環境部

CALL 文を使用する場合の環境部（ENVIRONMENT DIVISION）の定義を次に示します。

```
ENVIRONMENT           DIVISION.
  CONFIGURATION       SECTION.
        SPECIAL-NAMES.
        STDCALL IS 一意名1.
  EXTERNAL-PROGRAM    SECTION.
        CALL-CONVENTION.
        'jsvwadrv' IS 一意名1.
```

注

一意名 1 は，任意に設定してください。

環境部に上記の指定をした場合には，コンパイル時に次の指定をする必要があります。

1. コンパイラオプションに次の指定をする。

   -Comp5

   -JPN,Alnum

2. リンカオプションに次の指定をする。

リンカオプション：指定しない
インポートライブラリ／ユーザ指定ライブラリ：
XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

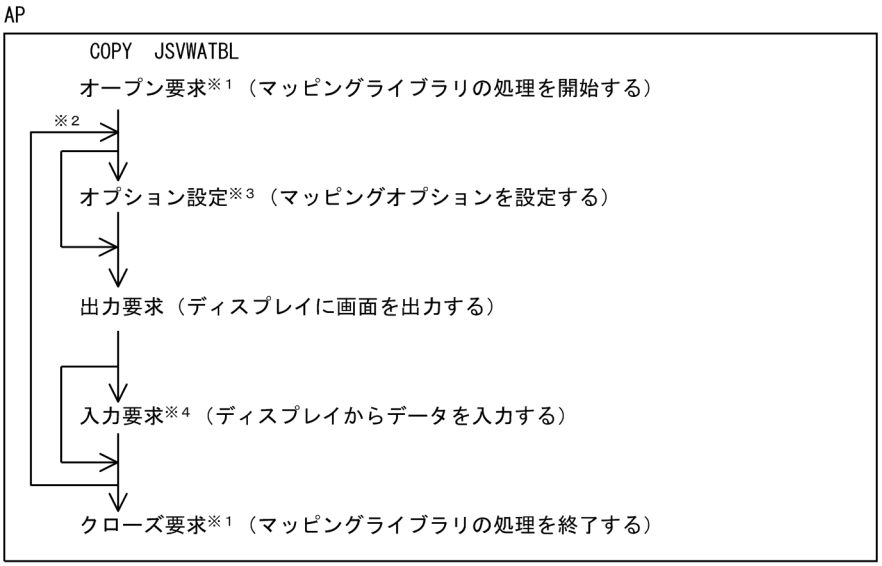
AP に上記のコーディングを追加しない場合は，コンパイル時に次の指定をする必要があります。

1. エディタを使用してファイルを作成する。
ファイル内容　　：jsvwadrv
ファイル名　　：XXXX.cbw（XXXX は任意）
ファイルの格納場所：AP と同じフォルダ
2. コンパイラオプションに次の指定をする。
-Comp5
-StdCall
-JPN,Alnum
3. リンカオプションに次の内容を指定する。
リンカオプション：指定しない
インポートライブラリ／ユーザ指定ライブラリ：
XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

(b)　CALL 文の発行順序

マッピングライブラリの機能と発行順序を次の図に示します。

図 3-8　マッピングライブラリの機能と発行順序



注※1　ディスプレイとプリンタを使用する場合，それぞれに必要です。
注※2　画面数分を繰り返します。
注※3　マッピングオプション変更時に発行します。
注※4　ディスプレイからデータを入力する場合に必要です。

CALL 文の形式については，「11.1.2　CALL 文による方法」を参照してください。

3.3.6　COBOL でのコンパイル（Windows）

(1)　COBOL 開発マネージャを使用したコンパイルと実行のポイント

COBOL 開発マネージャを使用したコンパイルと実行のポイントについて説明します。

(a)　COBOL 開発マネージャの概要

COBOL 開発マネージャは，COBOL2002 から提供されている COBOL プログラムの統合的な開発環境です。COBOL で AP を開発するときに必要な COBOL ソースや登録集原文などの資産をその依存関係に従って管理し，コンパイルなどの作業を自動化します。

COBOL 開発マネージャを使用する場合，次の手順で AP を作成します。

1. プロジェクトの作成
2. 資産の登録，定義
3. ビルド，リビルド

次に，それぞれの手順について説明します。

プロジェクトの作成

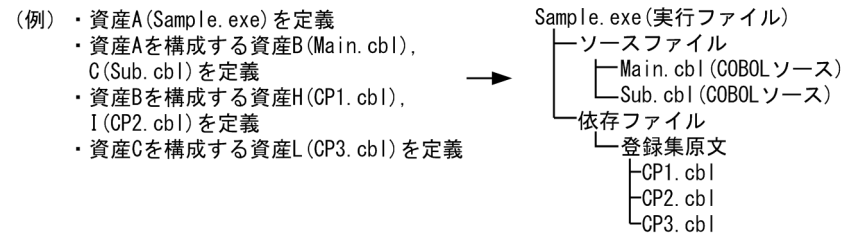
COBOL 開発マネージャでは，一つの EXE または DLL ファイルの開発単位を「プロジェクト」と呼んでいます。COBOL 開発マネージャを使用して AP を作成する場合，このプロジェクトを定義する必要があります。

資産の登録，定義

COBOL 開発マネージャでは，プロジェクトで作成される実行のファイル（.exe，または.dll）やそれを構成する要素を「資産」と呼んでいます。また，COBOL 開発マネージャでは，その依存関係を定義する必要があります。

資産の依存関係を定義すると，COBOL 開発マネージャではどのように表されるかを次の図に示します。

図 3-9　COBOL 開発マネージャでの表示



ビルド，リビルド

COBOL 開発マネージャでは，定義した資産の依存関係に基づいてコンパイルとリンケージをします。

その方法として，各資産の依存関係とタイムスタンプ（ファイル作成／修正日時）の前後関係に基づいてコンパイルとリンケージをする「ビルド」があります。例えば，「実行ファイルの作成／修正日時」より実行ファイルが取り込んでいる「COBOL ソースの作成／修正日時」の方が新しい場合にコンパイルとリンケージが実行されます。

また，タイムスタンプには関係なくコンパイルとリンケージをする「リビルド」もあります。

(b)　コンパイル時のポイント

コンパイラオプションを指定する

COBOL を使用したとき，指定が必要なコンパイルオプションを次に示します。

- -Comp5：COBOL プログラム中の COMP-5 を利用できるようにするオプション
- -JPN,Alnum：論理マップ内で日本語項目を扱えるようにするオプション
- -StdCall※：CALL 文を使用し，環境部に jsvwadrv の指定をしない場合に必要なオプション

注※

コンパイルオプション「-StdCall」は，Windows 版 XMAP3 サーバ/クライアント実行環境（32 ビット）の場合だけ利用できます。Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は利用できません。

AP 間でオープンを引き継ぐ

SEND／RECEIVE／TRANSCEIVE 文を使用して AP を分割してコンパイルする場合，共通画面を表示する個所を別 DLL にします。AP を分割してコンパイルするときは，コンパイル単位で XMAP3 の画面オープン・クローズ要求が毎回発行されないようにします。COBOL の実行支援の環境変数で，「CBLTERMSHAR=YES」を指定します。「CBLTERMSHAR=YES」は，SEND／RECEIVE／TRANSCEIVE 文で AP を作成したときだけ有効です。CALL 文で AP を作成したときは無効になります。

CALL 文で AP を作成するの場合，AP 間のオープン引き継ぎをするときは，XMAP3 インタフェースエリアの情報を引き継ぐようなコーディングをする必要があります。このコーディングのひな型として，AP パターンの GENDSP02 および GENDSP03 を利用できます。

登録集原文，動的変更テーブル（X3MODTBL）を確認する

- AP の COPY 文に，論理マップ（マップ生成時に付けられた名称）や動的変更テーブル（X3MODTBL）が間違って指定されていないか確認してください。
- 論理マップや動的変更テーブルがフォルダ中に用意されているか確認してください。ソースプログラムと登録集原文が同じフォルダで管理されている場合は，論理マップや動的変更テーブルが同一フォルダに格納されていないことが考えられます。また，ソースプログラムと登録集原文が別のフォルダで管理されている場合は，環境変数に指定した登録集原文のフォルダに誤りがあったり，指定したフォルダに論理マップや動的変更テーブルが格納されていなかったりすることが考えられます。動的変更テーブルおよび XMAP3 インタフェース領域テーブルは，「*XMAP3 インストールフォルダ*¥INCLUDE」の下に格納されていますので，必要なフォルダにコピーするか，または登録集原文が格納されているフォルダを環境変数 CBLLIB に指定して使用してください。
- データ名または変数名に対し，不当な文字を AP で指定していないか確認してください。
- CALL 文で AP を作成した場合には，XMAP3 インタフェース領域テーブル（JSVWATBL）が AP の COPY 文に指定されているか確認してください。

(c)　リンケージ時のポイント

XMAP3 使用時のリンケージオプションを指定する

CALL 文で作成した AP で XMAP3 を利用する場合，インポートライブラリ／ユーザ作成ライブラリに，次のように指定します。

XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

XMAP3 のライブラリを設定する

コンパイル環境（CALL 文で作成した AP の場合は，リンケージ環境）のマシンには，XMAP3 の開発環境をインストールしておく必要があります。

(d)　実行時のポイント

AP を実行する前に，物理マップやグラフィックデータ（グラフィックを使用する場合）を実行形式ファイルと同じフォルダに格納してください。

また，マップパスやグラフィックパスを指定する方法もあります。マップパスおよびグラフィックパスは，表示・印刷セットアップで指定します。表示・印刷セットアップについては，マニュアル「XMAP3 実行ガイド」を参照してください。

AP を分割してコンパイルした場合は，環境変数が必要になります。「3.1.2　AP 間でオープンを引き継ぐ場合」を参照してください。

(2)　COBOL 開発マネージャでの XMAP3 の利用方法

COBOL 開発マネージャと XMAP3 との両方をインストールしている場合，COBOL 開発マネージャから XMAP3 の機能を呼び出せます。XMAP3 の資産を登録，定義すれば，ビルド，またはリビルドによってコンパイルとリンケージをして AP を作成できます。

また，AP 作成時にソースプログラムや登録集原文を格納するフォルダを作成しておいてください。

次に，COBOL 開発マネージャから XMAP3 を利用する方法について説明します。

(a)　COBOL 開発マネージャ上での資産の登録，定義

COBOL 開発マネージャ上では，XMAP3 の資産を扱う場合，XMAP3 でのファイル名の規則に従います。また，XMAP3 の「マップ名」を基に作成します。

XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則を次の表に示します。

表 3–3　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則

XMAP3 の資産	COBOL 開発マネージャでの名称	COBOL 開発マネージャで付けるファイル名	内容
マップ定義	マップ定義	マップ名.imp	ユーザが定義した画面の情報を格納している。定義を修正するときにはこのファイル名を指定。
論理マップ	登録集原文	マップ名＋ O.cbl	AP で取り込む出力情報の登録集原文。
		マップ名＋ I.cbl	AP で取り込む入力情報の登録集原文。
物理マップ	－	マップ名＋ XX※.pmp	ユーザが定義した画面のうち，AP ではアクセスしない，固定部分の情報。 COBOL 開発マネージャの管理対象外。

（凡例）

－：該当しない。

注※

XX は，デバイス ID を示します。デバイス ID については，「11.1.1(1)　通信記述項」を参照してください。

また，COBOL 開発マネージャから XMAP3 の資産を登録，定義した場合，どのように表されるかを次に示します。

```
ソースプログラム
 ├ユーザプログラム(COBOLソース)←XMAP3の機能で定義した画面を利用するAP
 └マップ名.imp(マップ定義ファイル)
依存ファイル
 ├マップ名+I.cbl(登録集原文)
 └マップ名+0.cbl(登録集原文)
```

これを基にして，XMAP3 で作成した 2 種類の画面を COBOL 開発マネージャで利用して AP を作成する場合の例を次に示します。

```
（例）XMAP3で作成した画面1(マップ名:MENUM)と画面2(マップ名:MENUS)を
      利用してAPを作成する場合

      APP.exe(実行ファイル)
       ├ソースファイル
       │ ├Sample.cbl(COBOLソース)          ←1.AP
       │ ├MENUM.imp(マップ定義ファイル)    ←2.画面１用　画面定義ファイル
       │ └MENUS.imp(マップ定義ファイル)    ←3.画面２用　画面定義ファイル
       └依存ファイル
         └登録集原文
           ├MENUMI.cbl(登録集原文)         ←4.画面１用　入力論理マップ
           ├MENUM0.cbl(登録集原文)         ←5.画面１用　出力論理マップ
           ├MENUSI.cbl(登録集原文)         ←6.画面２用　入力論理マップ
           └MENUS0.cbl(登録集原文)         ←7.画面２用　出力論理マップ


（凡例）  ──:画面１のマップ名
          ═══:画面２のマップ名
```

注意

- ファイル名は，「表 3-3　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則」に従って定義する必要があります。
- ファイル名は，XMAP3 のマップ名を使用して作成してください。XMAP3 のファイル名を使用しないと，上記に示す資産定義と一致しなくなります。
- XMAP3 で入出力画面の定義をする場合，「1.画面 1 用　画面定義ファイル」，「4.画面 1 用　入力論理マップ」，「5.画面 1 用　出力論理マップ」を，「2.画面 2 用　画面定義ファイル」，「6.画面 2 用　入力論理マップ」，「7.画面 2 用　出力論理マップ」を同じファイル名で定義してください。

このように資産を登録，定義しておけば，定義ファイルに変更があった場合，ビルドによってコンパイル，リンケージされ，自動的に登録集原文（論理マップ）が生成し直されます。

(b)　マップ定義ファイルの作成と修正

マップ定義ファイルを作成，または修正したい場合，次に示す操作をすることで XMAP3 の画面定義ができます。

- COBOL 開発マネージャ上で，マップ定義ファイル（xxx.imp）をダブルクリックします。

(c)　マップ生成

COBOL 開発マネージャのビルド，またはリビルドを使用しないで単独にマップ生成をする場合，次の操作をすることでマップ生成ができます。生成される論理マップや物理マップのファイル名については，「表 3-3　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則」を参照してください。

マップ生成時に使用するリトルエンディアン，およびビッグエンディアンは，オプションで指定してください。

操作方法

特定のマップ定義ファイルからマップ生成をするときにだけ有効とする場合

COBOL 開発マネージャ上で，マップ定義ファイル（xxx.imp）をクリックして，［ファイル］－［ファイルの設定］を選択し，必要なコンパイルオプションを指定します。

すべてのマップ定義ファイルからマップ生成をするときに有効とする場合

COBOL 開発マネージャ上で，［プロジェクト］－［プロジェクトの設定］を選択し，「最適化」タブで必要なコンパイルオプションを指定します。

設定形式

[-BigEndian,Bin]

説明

-BigEndian,Bin または-Bb は，ビッグエンディアンを指定することを示します。指定がない場合，リトルエンディアンが仮定されます。

3.3.7　COBOL でのコンパイル（UNIX）

COBOL2002 で作成した，画面出力用 AP のコンパイル方法を説明します。

使用する UNIX およびターゲットの文字コード（シフト JIS，EUC）に応じて手順が異なります。

(1)　AIX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した画面出力用 AP のソースプログラム（xxx.cbl）を AIX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

AIX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 3-10　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

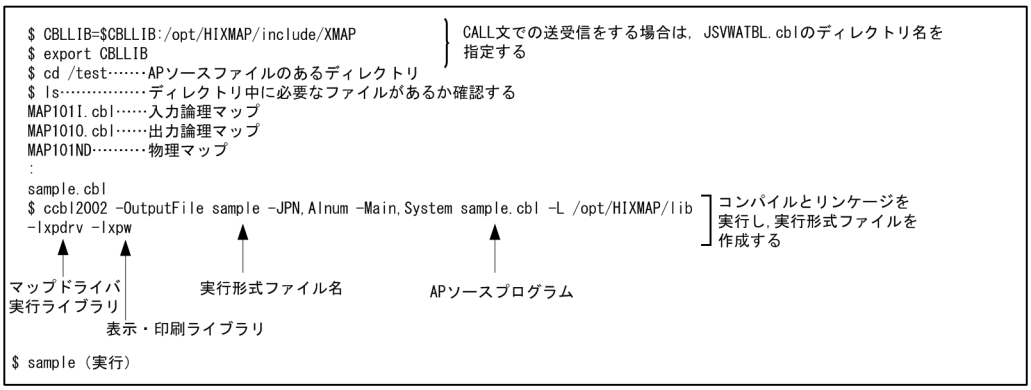
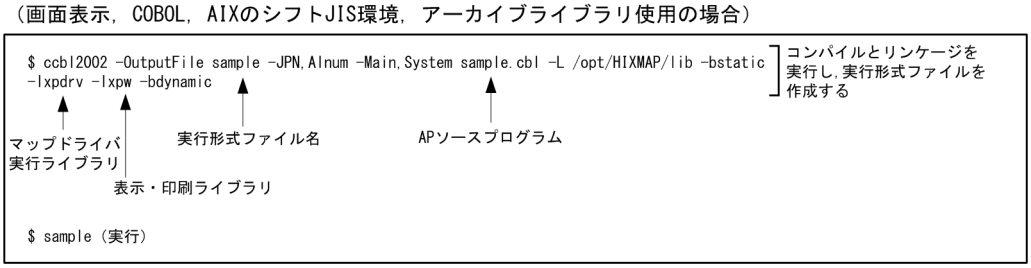


図 3-11　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）



(2)　AIX 環境でのコンパイル（EUC）

COBOL2002 で記述した画面出力用 AP のソースプログラム（xxx.cbl）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

AIX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 3-12　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

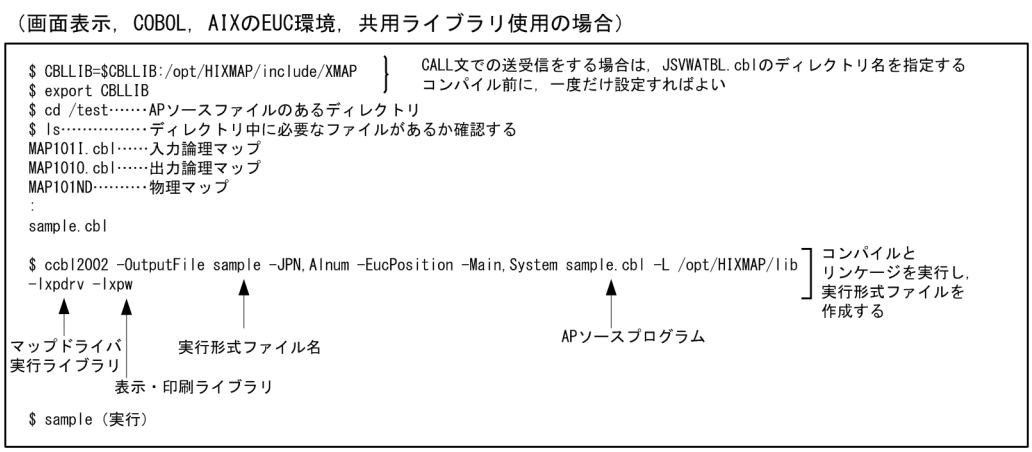
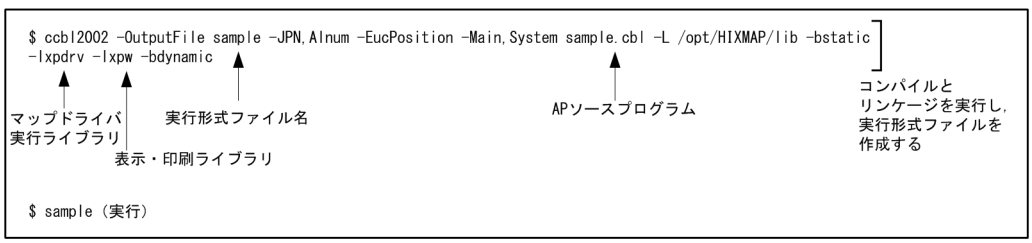


図 3-13　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）



(3)　HP-UX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した画面出力用 AP のソースプログラム（xxx.cbl）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

HP-UX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 3-14　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

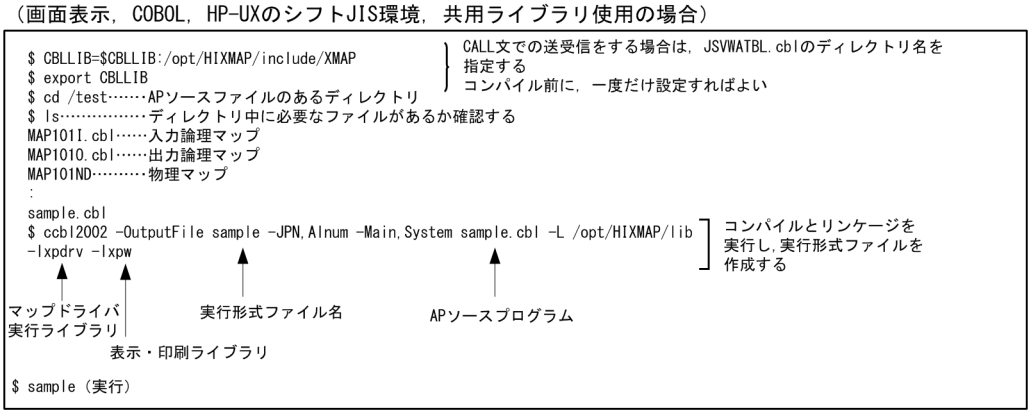
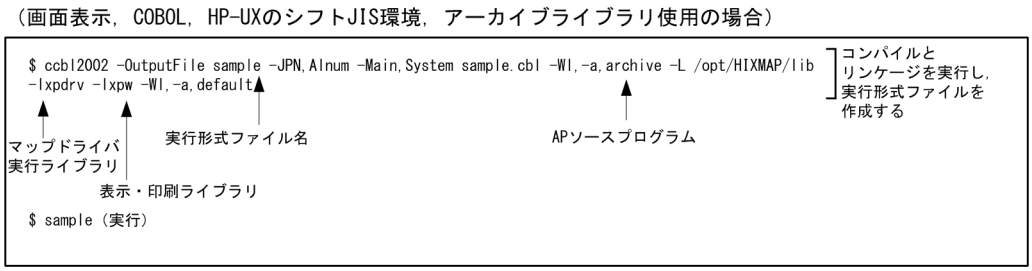


図 3-15　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）



(4)　HP-UX 環境でのコンパイル（EUC）

COBOL2002 で記述した画面出力用 AP のソースプログラム（xxx.cbl）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

HP-UX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 3-16　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

（画面表示，COBOL，HP-UXのEUC環境，共用ライブラリ使用の場合）

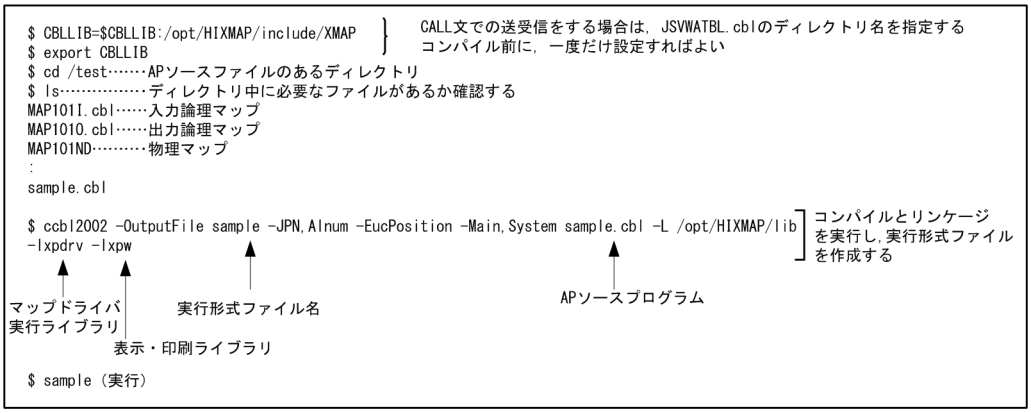
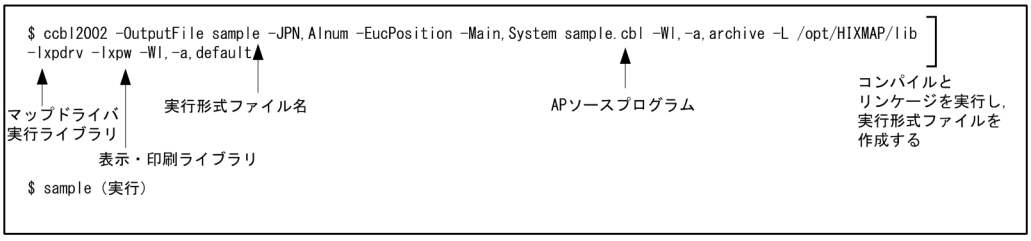


図 3-17　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）

（画面表示，COBOL，HP-UXのEUC環境，アーカイブライブラリ使用の場合）



3.4　C 言語での画面入出力命令

ここでは C 言語の AP のコーディング方法について説明します。

3.4.1　コーディング前の準備（Windows）

コーディングをする前に，次の準備をしてください。C 言語にないクラスなどの概念を持った C++言語で AP を作成する場合は，汎用関数を使用することをお勧めします。詳細については，「3.5　汎用関数での画面入出力命令」を参照してください。

1. **AP を格納するためのフォルダを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するフォルダを作成します。開発環境に合わせてフォルダを分類し，作成位置や名称を決めます。

2. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要なヘッダファイル※を標準提供しています。1.で作成したフォルダにコピーして使用することをお勧めします。このとき，コーディングのひな型である AP パターンもコピーしておきます。

注※

標準提供のヘッダファイルを次に示します。

- **動的変更テーブル**（X3MODTBL.H）

  動的変更用の修飾名と，出力論理マップの初期化に使用する定数が指定されています。

- **インタフェース領域**（JSVWATBL.H）

  jsvwadrv 関数で画面を送信するときにパラメタとして使用するインタフェース領域が指定されています。

3.4.2　コーディング前の準備（UNIX）

コーディングをする前に，次の準備をしてください。

1. **XMAP3 Developer で作成したファイルを転送する**

   XMAP3 Developer で UNIX 用に作成した物理マップ，論理マップおよび動的変更テーブルを UNIX マシンへファイル転送します。

   - **物理マップのファイル転送**

     XMAP3 Developer で UNIX 用に作成した物理マップ（拡張子「.pmp」）ファイルは，バイナリ形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

   - **論理マップ・動的変更テーブルのファイル転送**

     XMAP3 Developer で UNIX 用に作成した論理マップ（拡張子「.h」）ファイルおよび動的変更テーブル（X3MODTBL.h）ファイルは，テキスト形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください。

     XMAP3 Developer で作成する動的変更テーブルは，標準の出力先のままで作成した場合，次に出力されています。別の出力先に作成した場合は，そのファイルをファイル転送してください。

   XMAP3インストールフォルダ¥INCLUDE¥X3MODTBL.h

2. **物理マップを移行し形式チェックをする**

XMAP3 Developer で作成した UNIX 用の物理マップを，UNIX マシンへ転送したあと，cmapcp コマンドを使って，物理マップを実行環境に移行します。

cmapcp コマンドは，次に示すファイルを実行環境に移行し，さらに UNIX 版 XMAP3 で使用できる物理マップの形式かどうかのチェックをします。

- 拡張子が「.pmap」の物理マップファイル
- 拡張子が「.pmp」の物理マップファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

cmapcp コマンドについては，「15.1.1　cmapcp コマンド」を参照してください。

3. **UNIX の EUC 環境で利用する場合，論理マップおよび動的変更テーブルを文字コード変換する**

   XMAP3 Developer で生成される論理マップおよび動的変更ファイルは，文字コードがシフト JIS であるため，UNIX の EUC 環境で利用するには，ファイルを転送したあと，シフト JIS から EUC へコード変換する必要があります。

4. **AP を格納するためのディレクトリを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するディレクトリを作成します。開発環境に合わせてディレクトリを分類し，作成位置や名称を決めます。

5. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要なヘッダファイル※を標準提供しています。4.で作成したフォルダにコピーして使用することをお勧めします。このとき，AP の作成に使う論理マップ，コーディングのひな型である AP パターン，AP 実行時に使用する物理マップもコピーしておきます。

注※

標準提供のヘッダファイルを次に示します。

- **動的変更テーブル（X3MODTBL.H）**

  AP の実行中に画面の色などを変更する表示属性や，出力エリア（出力論理マップ）の初期化に使用する定数が指定されています。なお，XMAP3 Developer で UNIX 用の動的変更テーブルを作成した場合は，標準提供のテーブルは使用しないでください。必ず，XMAP3 Developer で作成した動的変更テーブルをファイル転送して使用してください。

- **インタフェース領域（JSVWATBL.H）**

  jsvwadrv 関数で画面を送信するときにパラメタとして使用するインタフェース領域が指定されています。

3.4.3　仮想端末の自動割当て

C/S 構成の環境で，サーバ上の AP から複数のクライアントマシンへ画面表示をする場合，仮想端末の自動割当て機能を利用できます。この機能は，C/S 構成の場合だけ利用できます。

仮想端末の自動割当て機能は，サーバ AP が各クライアントマシンに対応した仮想端末名を意識することなく一つの仮想端末名だけを意識していれば，各クライアントマシンにある XMAP3 の表示サービスが起動されたタイミングで，AP から送信された情報を画面表示できます。仮想端末の自動割当て機能を利用するときはサーバ側で，AP で指定した自動割当て用の仮想端末名を設定します。

仮想端末の自動割当てについては，マニュアル「XMAP3 実行ガイド」を参照してください。

3.4.4　論理マップの取り込み方法（C 言語）

AP 中に論理マップを取り込むには，#include 指示語を使用します。

AP に論理マップ MAP003O.h，および MAP003I.h を取り込む例を次に示します。

```
#include ”MAP0030.h” … 出力論理マップの取り込み
#include ”MAP003I.h” … 入力論理マップの取り込み
```

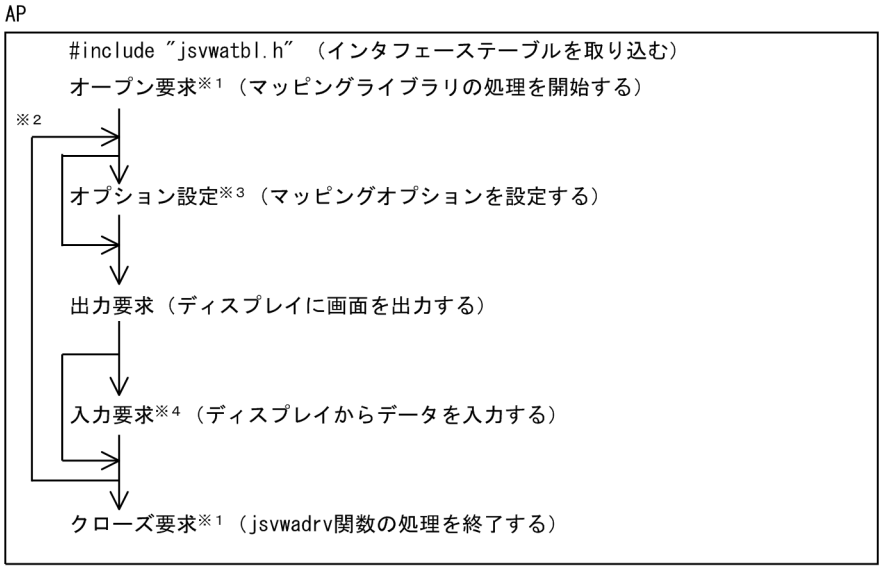
#include 指示語によって AP に実領域が取られます。

3.4.5　画面表示命令（C 言語）

XMAP3 を呼び出すための C 言語の関数として，次に示す jsvwadrv 関数を提供します。

(1)　jsvwadrv 関数の発行順序

jsvwadrv 関数の発行順序を次に示します。



注※1　必ず指定します。
注※2　画面数分を繰り返します。
注※3　マッピングオプション変更時に発行します。
注※4　ディスプレイからデータを入力する場合に必要です。

jsvwadrv 関数の形式については，「12.1.1　画面表示命令（C 言語）」を参照してください。

3.4.6　C 言語でのコンパイル

C 言語で作成した，画面出力用 AP のコンパイル方法を OS ごとに説明します。

(1)　Windows 環境でのコンパイル

C 言語のコンパイラを使用してコンパイルしてください。ライブラリには，次に示すファイルを指定する必要があります。

XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

(2)　AIX 環境でのコンパイル（シフト JIS）

C 言語で記述した画面出力用 AP のソースプログラム（xxx.c）を AIX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -o：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -qmbcs：AP 内でマルチバイト文字（日本語）を使用している場合に指定する
- -I インクルードファイルパス：インクルードファイルの検索パスを指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -bdynamic：共用ライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX のシフト JIS 環境での xlc コマンドによるコンパイル例を次の図に示します。

図 3-18　AIX のシフト JIS 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

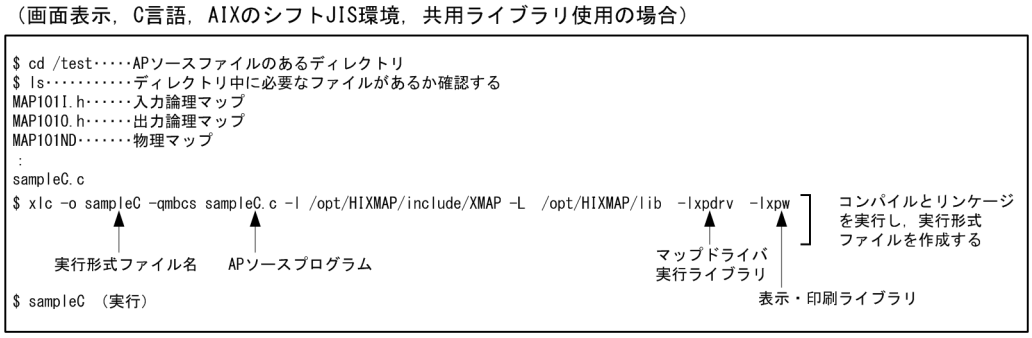
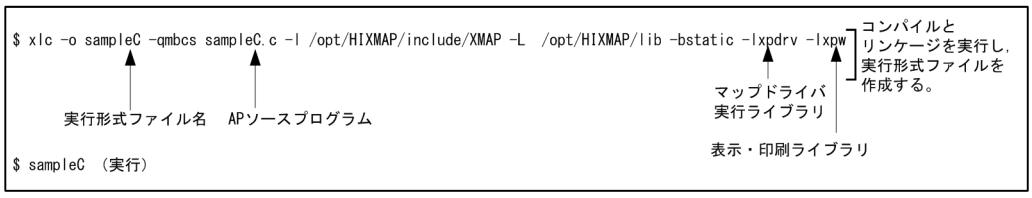


図 3-19　AIX のシフト JIS 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）



(3)　AIX 環境でのコンパイル（EUC）

C 言語で記述した画面出力用 AP のソースプログラム（xxx.c）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

! 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -o：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -qmbcs：AP 内でマルチバイト文字（日本語）を使用している場合に指定する
- -I インクルードファイルパス：インクルードファイルの検索パスを指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -bdynamic：共用ライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX の EUC 環境での xlc コマンドによるコンパイル例を次の図に示します。

図 3-20　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

（画面表示，C言語，AIXのEUC環境，共用ライブラリ使用の場合）

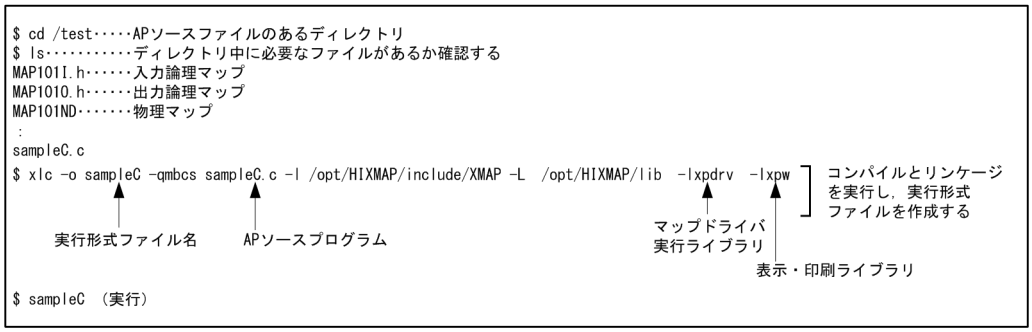
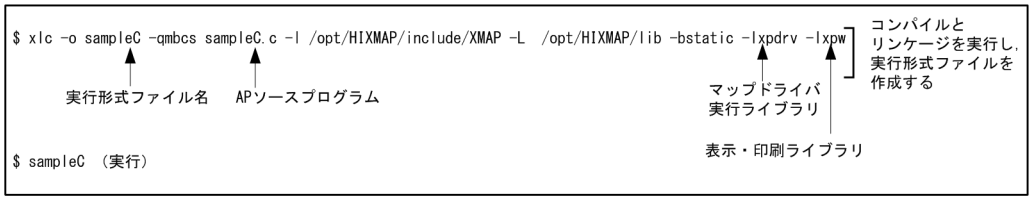


図 3-21　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）

（画面表示，C言語，AIXのEUC環境，アーカイブライブラリ使用の場合）



(4)　HP-UX 環境でのコンパイル（シフト JIS）

C 言語で記述した画面出力用 AP のソースプログラム（xxx.c）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX のシフト JIS 環境での cc コマンドによるコンパイル例を次の図に示します。

図 3-22　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

（画面表示，C言語，HP-UXのシフトJIS環境，共用ライブラリ使用の場合）

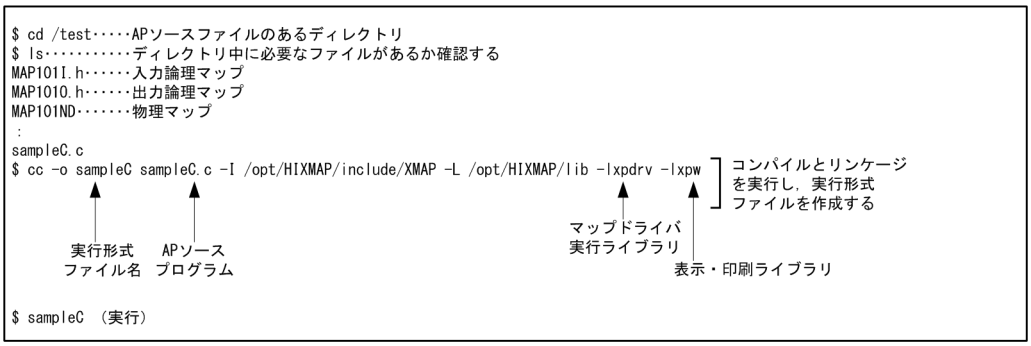
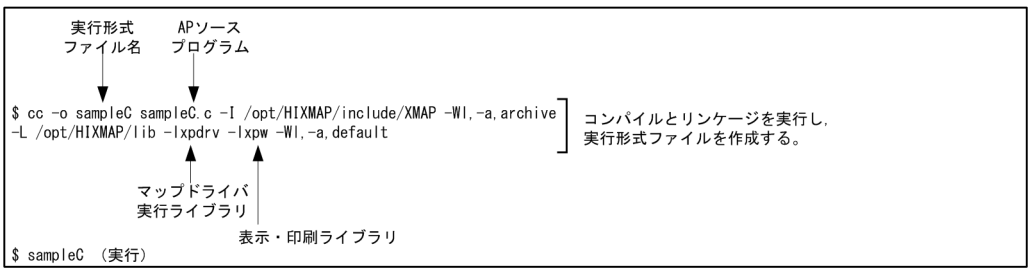


図 3-23　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）

（画面表示，C言語，HP-UXのシフトJIS環境，アーカイブライブラリ使用の場合）



(5)　HP-UX 環境でのコンパイル（EUC）

C 言語で記述した画面出力用 AP のソースプログラム（xxx.c）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

! 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX の EUC 環境での cc コマンドによるコンパイル例を次の図に示します。

図 3-24　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（画面表示）

（画面表示，C言語，HP-UXのEUC環境，共用ライブラリ使用の場合）

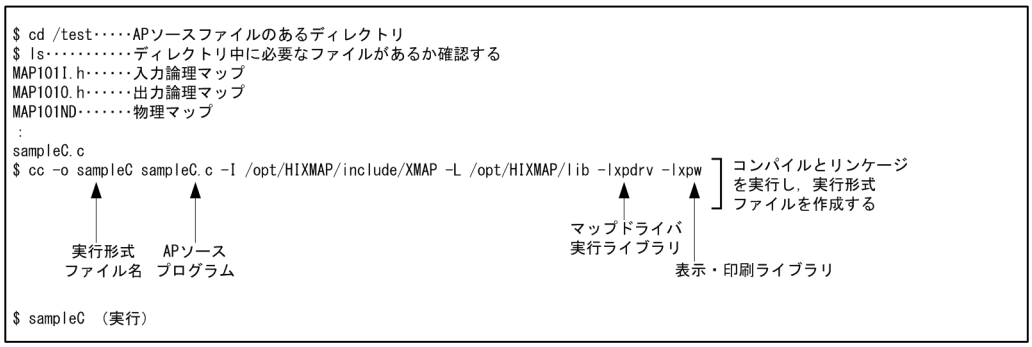
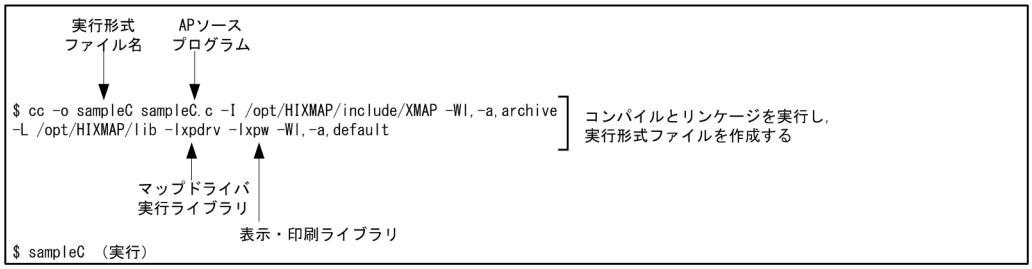


図 3-25　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（画面表示）

（画面表示，C言語，HP-UXのEUC環境，アーカイブライブラリ使用の場合）



3.5　汎用関数での画面入出力命令

画面入出力で使用する Windows 専用の汎用関数について説明します。また，汎用関数を使用して AP で表示する方法について説明します。汎用関数は，C，および C++で使用できます。

なお，OLTP サーバ構成の環境では，汎用関数は使用できません。

3.5.1　汎用関数の機能概要

Windows 専用の汎用入出力関数とそれぞれの機能概要を次の表に，Windows 専用の汎用入出力関数の発行順序を次の図に示します。

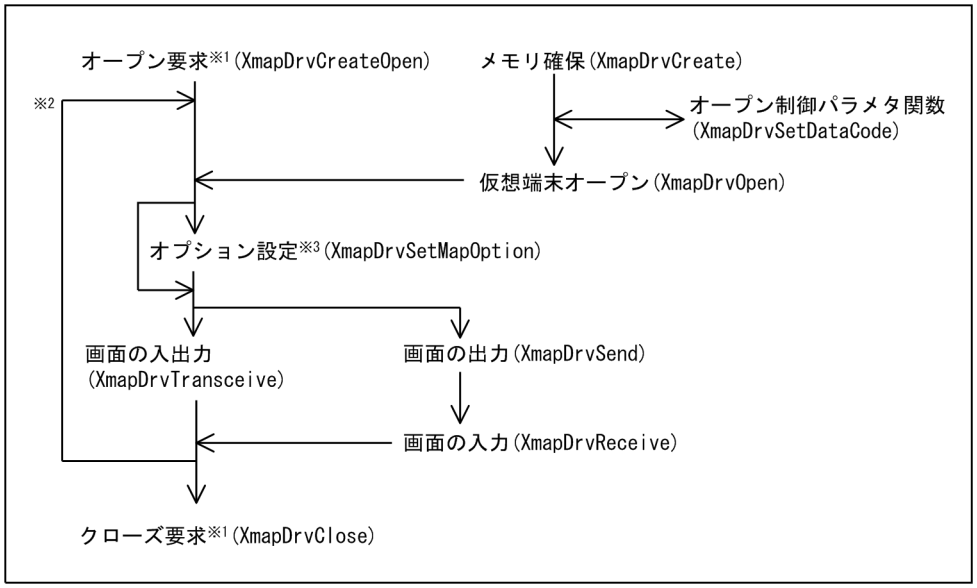
表 3–4　Windows 専用の汎用入出力関数

関数	機能概要	備考
XmapDrvCreateOpen	メモリ確保，仮想端末オープン（ディスプレイ）	OPEN 要求関数
XmapDrvClose	メモリ解放，仮想端末クローズ	CLOSE 要求関数
XmapDrvSend	画面の出力	－
XmapDrvReceive	画面の入力	－
XmapDrvTransceive	画面の入出力	－
XmapDrvGetError	エラーコードの詳細を取得	－
XmapDrvCreate	メモリ確保	－
XmapDrvOpen	仮想端末オープン	OPEN 要求関数
XmapDrvSetMapOption	マッピングオプションを設定	出力制御パラメタ関数
XmapDrvSetDataCode	データ有無コードを設定	オープン制御パラメタ関数

（凡例）

－：該当しない。

図 3-26　Windows 専用の汎用入出力関数の発行順序



注※1 必ず指定します。
注※2 画面数分を繰り返します。
注※3 マッピングオプション変更時に発行します。

汎用関数については,「13.1.1　画面表示命令（汎用関数)」を参照してください。

3.5.2　C 言語での画面の表示方法

C 言語での画面の表示方法について説明します。

(1)　DLL の呼び出し

汎用関数では，XMAP3 が提供する DLL を使用するため，インタフェース宣言ファイルおよびリンケージライブラリを取り込む必要があります。

- インタフェース宣言ファイルを取り込みます。インタフェース宣言ファイルの格納先を次に示します。
  XMAP3 インストールフォルダ¥INCLUDE¥X3MWGD32.H
- リンケージライブラリを取り込みます。リンケージライブラリの格納先を次に示します。
  XMAP3 インストールフォルダ¥LIB¥X3MWGD32.LIB

4 画面のコーディング例

この章では，画面定義と AP のコーディング，および画面の各オブジェクトと AP の関連について説明します。

4.1　画面属性と AP

GUI 画面の画面属性と AP の関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：COBOL
- マップ名の文字数：6 文字
- マップ名：MAP001
- 実行環境：対話型

ドロー，およびドローセットアップについては，マニュアル「XMAP3 開発ガイド」を参照してください。

4.1.1　ウィンドウ属性を変更する

動的変更を利用すると，AP でウィンドウ属性を変更できます。表示中の画面をいったん消去して再び表示したり，表示中の画面の一部の項目だけを変更して表示したりできます。

ウィンドウ属性は次のどちらかの方法で指定します。

- 画面属性ダイアログ
  各マップの画面属性ダイアログの「表示形態」で指定します。標準値は自動です。
- AP
  各マップの画面属性ダイアログの「動的変更（AP からウィンドウ属性を変更する)」を選択して，動的変更を利用して AP で指定します。

画面属性ダイアログの「表示形態」と「動的変更（AP からウィンドウ属性を変更する)」の両方を選択した場合は，AP での指定が優先されます。

ウィンドウ属性を動的変更する場合のコーディング例を次に示します。

図 4-1　ウィンドウ属性を動的変更するコーディング例

```
MOVE  SPACE        TO  マッピングモード.     …マッピングオプションの設定（マージ）
MOVE  XMAP-CNTRL1  TO  MAP001-CNTRLO.        …ウィンドウ属性に一部上書を設定
```

AP でウィンドウ属性を動的変更する場合，制御項目「MAP001-CNTRLO」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

4.1.2　ウィンドウ位置属性を変更する

動的変更を利用すると，AP でウィンドウ位置属性を変更できます。一時的にウィンドウを非表示にしたあとで，手前に再表示するなど，画面のオープン要求やクローズ要求をしないで，表示するウィンドウを切り替えられるので，処理を高速化できます。

ウィンドウ位置属性は，次のどちらかの方法で指定します。

- 画面属性ダイアログ
  各マップの画面属性ダイアログの「XY 位置」で指定します。

- AP
  各マップの画面属性ダイアログの「動的変更（AP からウィンドウ位置属性を変更する)」を選択して，動的変更を利用して AP で指定します。

画面属性ダイアログの「XY 位置」と「動的変更（AP からウィンドウ位置属性を変更する)」の両方を選択した場合は，AP での指定が優先されます。

ウィンドウ位置属性を動的変更する場合のコーディング例を次に示します。

図 4-2　ウィンドウ位置属性を動的変更するコーディング例

```
MOVE  XMAP-WINDOW-VC  TO  MAP001-WINDOWW.   …ウィンドウ位置属性に中央表示を設定
```

AP でウィンドウ位置属性を動的変更する場合，制御項目「MAP001-WINDOWW」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

4.1.3　確定キー属性を変更する

動的変更を利用すると，AP で確定キー属性を変更できます。画面を確定するキーを有効または無効に変更できます。

確定キー属性は次のどちらかの方法で指定します。

- 画面属性ダイアログ
  各マップの画面属性ダイアログの「イベント通知コード」で指定します。画面定義時の仮定値では，すべての PF キー，PA キー，送信キー，割込キー，スクリーン消去キー，およびボタン／メニューが有効です。
- AP
  各マップの画面属性ダイアログの「イベント通知コード」で確定キーを有効にしたあとで，「動的変更(AP から確定キー属性を変更する)」を選択します。動的変更を利用して，AP で確定キーを無効とするように指定します。
  画面属性ダイアログの「イベント通知コード」で確定キーを無効としている場合，AP で確定キーを有効にできません。

各マップの画面属性ダイアログの「イベント通知コード」と「動的変更（AP から確定キー属性を変更する)」の両方を選択した場合は，AP での指定が優先されます。

確定キー属性を動的変更する場合のコーディング例を次に示します。

図 4-3　確定キー属性を動的変更するコーディング例

```
MOVE  XMAP-EVENT-AL  TO  MAP001-INCO.        …確定キー属性にすべて有効を設定
```

AP から確定キー属性を動的変更する場合，制御項目「MAP001-INCO」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

4.2　GUI 画面の各オブジェクトと AP

GUI 画面の各オブジェクトと AP の関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：COBOL
- マップ名数：6 文字
- マップ名：MAP001
- 実行環境：対話型
- 各オブジェクトのデータ名や制御項目データ名：標準値

ドローおよびドローセットアップについては，マニュアル「XMAP3 開発ガイド」を参照してください。

4.2.1　キャラクタコントロール（ラベル）の表示

GUI 画面のキャラクタコントロール（ラベル）とは，次の四つのオブジェクトです。

- 固定テキスト
- 出力テキスト／出力フィールド
- 出力日付テキスト／出力日付フィールド
- 出力時刻テキスト／出力時刻フィールド

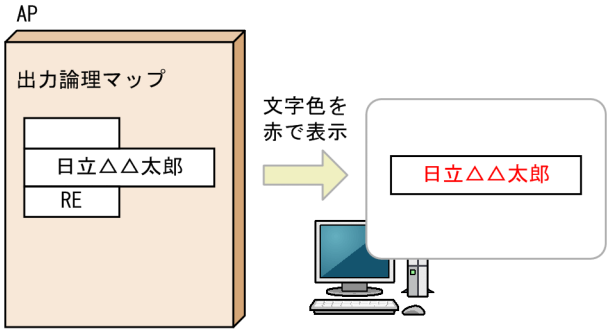
ここでは，AP で出力テキストを表示する場合のコーディング例について説明します。

出力テキストでは，動的変更を利用して，AP で文字，文字色などを指定できます。

ドローで出力テキストボックスのダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，文字色，およびオブジェクトの表示・非表示の切り替えを AP で指定できます。AP で指定しない場合，出力テキストボックスダイアログで指定した属性で表示されます。

AP 実行時に，AP で指定した文字色での出力テキストの表示例を次の図に示します。

図 4-4　指定した文字色での出力テキストの表示例



（凡例）
△：半角の空白

表示したい文字列「日立 太郎」と赤色を表示する修飾名「RE」を出力論理マップに代入すると，AP 実行時には「日立 太郎」が赤色で画面に表示されます。

文字色を変更する場合の出力テキストのコーディング例を次に示します。

図 4-5　文字色を変更する出力テキストのコーディング例

```
MOVE  '日立 太郎'      TO  MAP001-FIELD0001-O.       …出力テキストを代入
MOVE  XMAP-OUT-ATTR1  TO  MAP001-FIELD0001-A.       …文字色を赤色に設定
```

出力テキストにテキストを表示させる場合，出力論理マップの「MAP001-FILED0001-O」にデータを代入します。また，出力テキストの表示属性を変更する場合，制御項目「MAP001-FIELD0001-A」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

! 注意事項

マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合，画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，出力テキストボックスに定義した属性には戻りません。出力テキストボックスに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

4.2.2　キャラクタコントロール（キーエントリ）の表示

GUI 画面のキャラクタコントロール（キーエントリ）とは，次の三つのオブジェクトです。

- 入出力テキスト／入出力フィールド
- 入出力日付テキスト／入出力日付フィールド
- 入出力時刻テキスト／入出力時刻フィールド

ここでは，AP で入出力テキストを表示する場合のコーディング例について説明します。

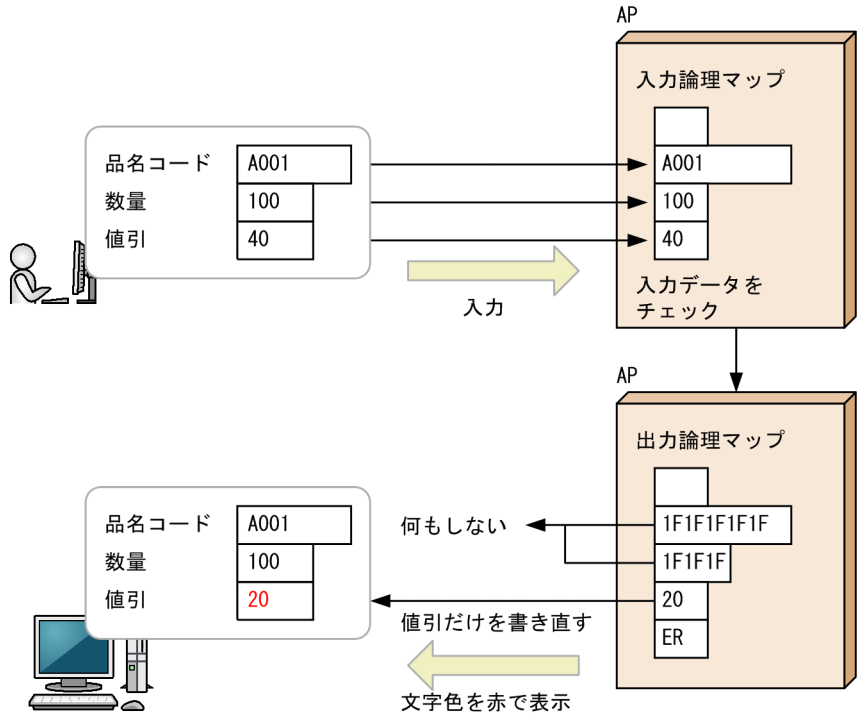
入出力テキストでは，画面に入力されたデータを AP で受け取ったり，AP で文字，文字色などを指定したりできます。出力テキストと同様に特定のデータだけを変更できます。

テキストの表示については，「4.2.1　キャラクタコントロール（ラベル）の表示」を，データ有無コードについては，「16.　論理マップ生成規則とマッピング規則」を参照してください。

ドローで入出力テキストボックスのダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，文字色，背景色，表示方法，遷移条件など，入出力テキストの属性を AP で変更できます。AP で指定しない場合，入出力テキストボックスのダイアログで指定した属性で表示されます。

AP 実行時に，データ有無コードを使用して特定のデータだけを変更する場合の入出力テキストの表示例を次の図に示します。

図 4-6　データ有無コードを使用した入出力テキストの表示例



画面に入力されたデータを AP でチェックして，再び画面に表示します。画面に表示するときには，品名コードと数量はデータを変更しないで表示するので，データ有無コードを使用します。値引のデータだけは，データを変更して赤色で表示します。

データ有無コードを使用した入出力テキストのコーディング例を次に示します。

図 4-7　データ有無コードを使用した入出力テキストのコーディング例

```
MOVE 2              TO  マッピングモード.     …マッピングオプションの設定（論理マップだけ）
MOVE ALL X'1F'      TO  MAP001G.              …データ有無コード(1F)16でクリアする
MOVE 20             TO  MAP001-FIELD0001-0.   …表示データを変更する
MOVE XMAP-IN-ATTR1 TO  MAP001-FIELD0001-A.   …エラー(赤色)表示をする
```

入出力テキストの文字色を変更する場合，制御項目「MAP001-FIELD0001-A」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

! 注意事項

マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合。画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，入出力テキストボックスに定義した属性には戻りません。入出力テキストボックスに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

4.2.3　キャラクタコントロール（選択エントリ）の制御

GUI 画面のキャラクタコントロール（選択エントリ）とは，次の二つのオブジェクトです。

- ポップアップテキスト／ポップアップフィールド

- コンボボックス

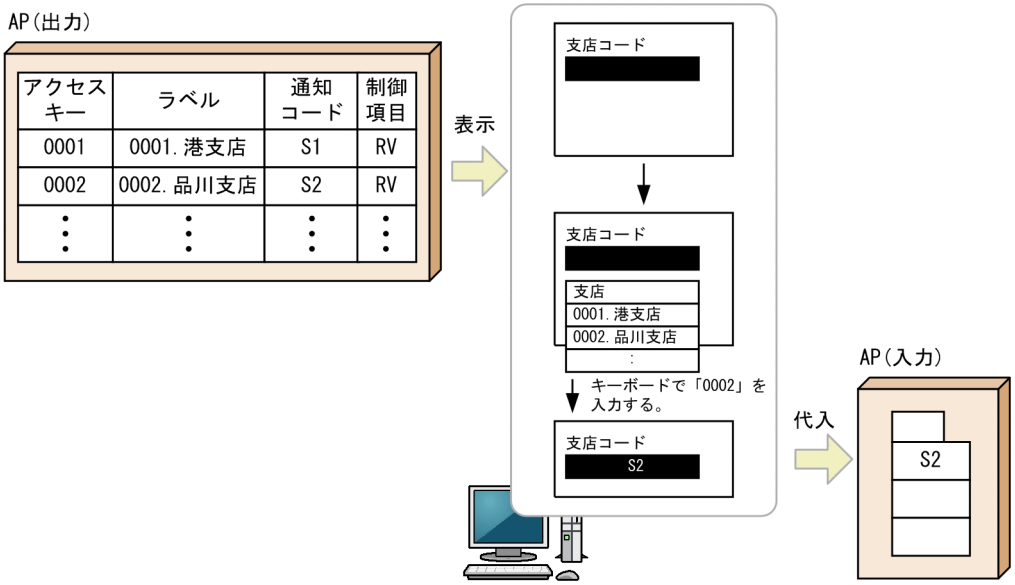
ここでは，AP で可変ポップアップテキストを表示する場合のコーディング例について説明します。

可変ポップアップでは，ポップアップメニューのラベル，通知コード，文字色などを AP で指定できます。

ドローで可変ポップアップメニューのダイアログの「動的変更（AP から表示属性を変更する)」を選択すると，動的変更を利用して文字色，背景色，表示方法，遷移条件など，オブジェクトの属性を AP で変更できます。AP で指定しない場合，可変ポップアップメニューのダイアログで指定した属性で表示されます。

アクセスキーでポップアップメニューを選択する表示例について次の図に示します。

図 4-8　アクセスキーでポップアップメニューを選択する表示例



（凡例）
■：選択されている状態。テキストボックスは反転表示。

注
アクセスキーは，ポップアップメニューが表示されているときだけ有効です。

可変ポップアップテキストの制御項目に修飾名を代入して，可変ポップアップテキストを反転表示させます。可変ポップアップテキストにフォーカスを移すと，ポップアップメニューが表示されます。キーボードでアクセスキー「0002」を入力すると，ポップアップメニューが選択されます。ポップアップメニューで選択されたデータの通知コードが入力論理マップに代入されます。ポップアップテキストには，選択されたメニューの通知コードが表示されます。

ポップアップメニューを選択する可変ポップアップのコーディング例を次に示します。

図 4-9　アクセスキーでポップアップメニューを選択する可変ポップアップのコーディング例

```
MOVE '港支店'          TO  MAP001-POPUP-LABEL0001-O(1).    …ラベルの設定
MOVE '品川支店'        TO  MAP001-POPUP-LABEL0001-O(2).
MOVE 'S1'              TO  MAP001-POPUP-CODE0001-O(1).     …通知コードの設定
MOVE 'S2'              TO  MAP001-POPUP-CODE0001-O(2).
MOVE '0001'            TO  MAP001-POPUP-KEY0001-O(1).      …アクセスキーの設定
MOVE '0002'            TO  MAP001-POPUP-KEY0001-O(2).
MOVE XMAP-IN-ATTR2     TO  MAP001-FIELD0001-A.             …反転表示する
   :
   :
   :
IF MAP001-FIELD0001-I = 'S1'                               …入力した通知コードの参照
```

```
THEN
  :
```

可変ポップアップテキストメニューの表示属性を変更する場合，制御項目「MAP001-FIELD0001-A」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

! 注意事項

マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合，画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，可変ポップアップテキストメニューに定義した属性には戻りません。可変ポップアップテキストメニューに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

4.2.4　候補選択コントロールの制御

GUI 画面の候補選択コントロールとは，次の四つのオブジェクトです。

- ラジオボタン
- チェックボタン
- リスト項目
- トグルフィールド

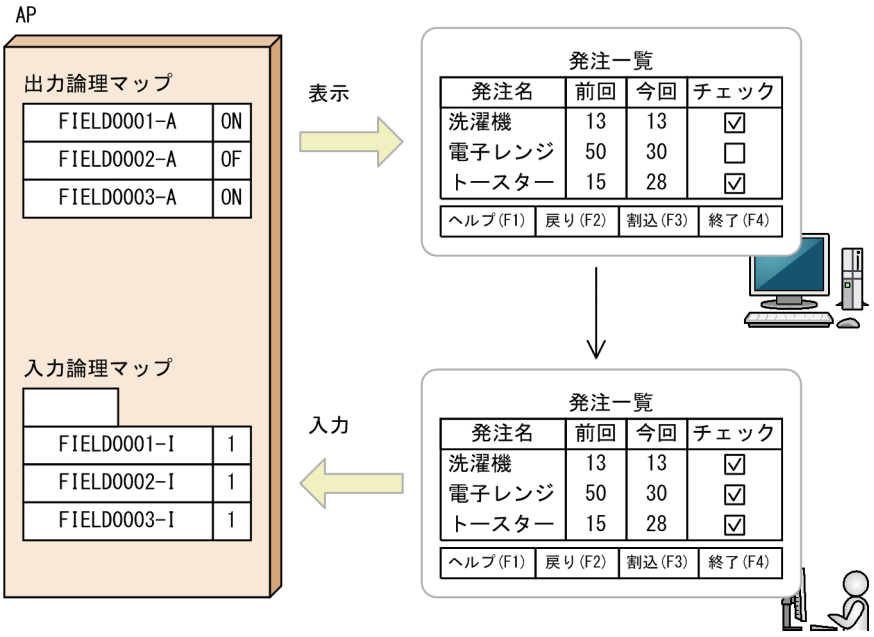
ここでは，トグルフィールドの選択状態を取得する場合のコーディング例について説明します。

トグルフィールドが選択されているかどうかを通知コードで参照できます。

ドローでトグルフィールドのダイアログの「動的変更（AP から表示属性を変更する)」を選択すると，動的変更を利用して，トグルフィールドの選択状態を AP で指定できます。AP で指定しない場合，トグルフィールドのダイアログで指定した属性で表示されます。

トグルフィールドの選択と入出力論理マップの関係を次の図に示します。

図 4-10　トグルフィールドの選択と入出力論理マップの関係



AP でトグルフィールドの選択を指定したとおりに，画面に表示されます。画面でトグルフィールドの選択状態を変更すると，入力論理マップの通知コードにデータが代入されます。通知コードでトグルフィールドの選択状態がわかります。

トグルフィールドの選択状態をチェックするコーディング例を次に示します。

図 4-11　トグルフィールドの選択状態をチェックするコーディング例

```
MOVE  XMAP-BUTTON-SEL2  TO  MAP001-FIELD0001-A.    …選択済みにする
MOVE  XMAP-BUTTON-SEL3  TO  MAP001-FIELD0002-A.    …未選択にする
MOVE  XMAP-BUTTON-SEL2  TO  MAP001-FIELD0003-A.    …選択済みにする
    :
    :
IF MAP001-FIELD0002-I = '1'                        …ユーザの指定をチェックする
THEN
    :
```

トグルフィールドの選択状態を変更する場合，制御項目「MAP001-FIELD0001-A」，「MAP001-FIELD0002-A」などに修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

! 注意事項

マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合，画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，トグルフィールドに定義した属性には戻りません。トグルフィールドに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

4.2.5　コマンドコントロールの制御

GUI 画面のコマンドコントロールとは，次の二つのオブジェクトです。

- プッシュボタン
- メニューバー

ここでは，プッシュボタンボックスにフォーカスがある場合，プッシュボタンの表示を変更するときのコーディング例について説明します。

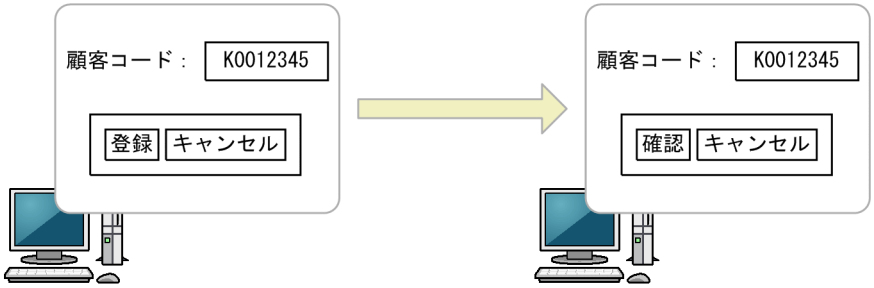
プッシュボタンボックスでは，プッシュボタンが選択されたときに画面が確定されるので，AP で入力論理マップのプッシュボタンに対応したイベント通知コードを参照できます。イベント通知コードを条件にして，再び画面を表示するときに，プッシュボタンのラベルや表示状態を変更できます。

プッシュボタンダイアログの「動的変更（AP から表示属性を変更する)」を選択すると，動的変更を利用して，プッシュボタンの活性，不活性を AP で指定できます。AP での指定がない場合は，プッシュボタンダイアログで指定した属性を表示します。

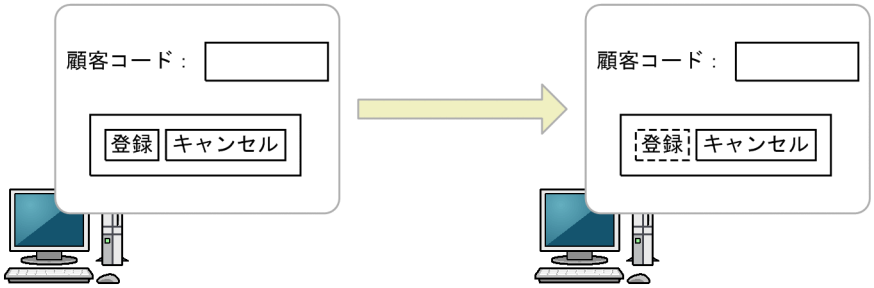
AP 実行時に，条件別にプッシュボタンの状態を変更する例を次の図に示します。

図 4-12　プッシュボタンの状態を変更する例

● データが入力されている場合



● データが入力されていない（すべての桁が半角スペース）場合



入出力テキストにデータが入力されている場合と入力されていない場合で，プッシュボタンの表示を変更します。

入出力テキストにデータが入力されている場合は，次の画面表示で，［登録］ボタンを［確認］ボタンに変更します。入出力テキストにデータが入力されていない場合は，［登録］ボタンの状態を不活性に変更します。

入出力テキストのデータの有無を条件にしたプッシュボタンの表示を変更するコーディング例を次に示します。

図 4-13　プッシュボタンの表示を変更するコーディング例

```
IF MAP001-FILED001-I NOT EQUAL SPACE.                    …入出力テキストの入力有無チェック
  THEN
    MOVE  '  確認  '          TO MAP001-BUTTON0001-O.    …ラベルを変更する
```

```
  ELSE
    MOVE  MAP001-BUTTON-SEL1 TO MAP001-BUTTON0001-A.    …プッシュボタンを不活性にする
  ENDIF
ENDIF
```

プッシュボタンの表示属性を変更する場合，制御項目「MAP001-BUTTON0001-A」に修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，AP に取り込んだ動的変更テーブルの修飾名を制御項目に代入してください。

! 注意事項

マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合，画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，プッシュボタンに定義した属性には戻りません。プッシュボタンに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

AP へ通知するコードとして A001 や A002 を定義した場合，［Ctrl］＋［F1］，［Ctrl］＋［F2］などを押しても同じ通知コードが AP に返ります。プッシュボタン専用の通知コードを AP に返したい場合は，A061～A071 を定義してください。

4.2.6　出力グラフィックの利用

出力グラフィックを使用すると，AP でグラフィックデータを指定して，グラフィック画像を画面に表示できます。

出力グラフィックを利用したグラフの表示例とコーディング例を次に示します。

図 4-14　出力グラフィックを利用したグラフの表示例

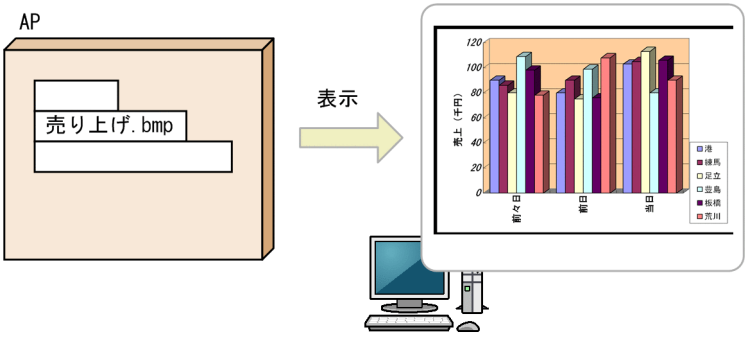


図 4-15　出力グラフィックを利用したグラフ表示のコーディング例

```
MOVE  ALL X'1F'        TO  MAP001-GRAPH0001-O      …データ領域のクリア
MOVE  '売り上げ.bmp'   TO  MAP001-GRAPH0001-O      …直接ファイル名を指定
```

画面または帳票の出力グラフィックで使用するグラフィックデータを格納しているフォルダのパスは，実行環境の表示・印刷セットアップで指定します。

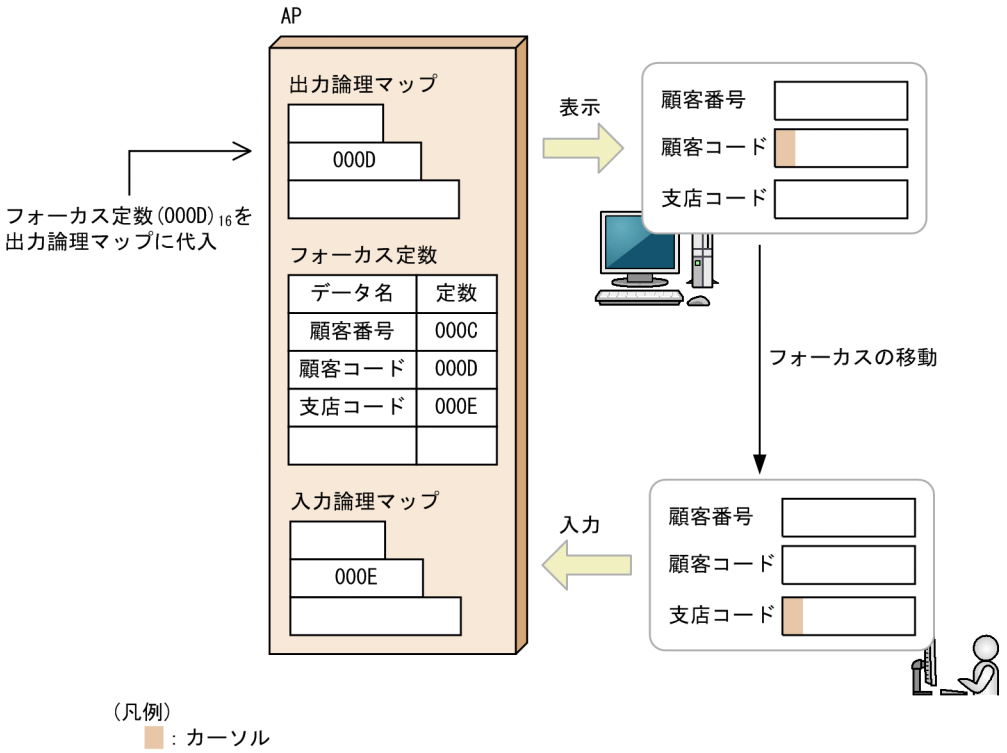
4.2.7　初期フォーカスの動的変更

画面を表示したときの初期フォーカス位置を AP で指定できます。AP でフォーカス位置を指定しないと，フォーカス位置はドローで定義した初期フォーカス位置に指定されます。

ここでは，入出力テキストにフォーカスを位置づけて表示する場合のコーディング例について説明します。

AP 実行時のフォーカスの位置づけを次の図に示します。

図 4-16　フォーカスの位置づけ



顧客コードに初期フォーカス位置を指定するために，出力論理マップにフォーカス定数を代入します。顧客コードにフォーカスがある状態で，画面が表示されます。フォーカスを支店コードに移動すると，入力論理マップにフォーカス定数が格納されます。

フォーカス位置を特定するコーディング例を次に示します。

図 4-17　フォーカス位置を特定するコーディング例

```
MOVE  MAP001-FILED0002-T  TO  MAP001-OUTCURS-LOCO.      …フォーカス定数を設定
    :
    :
IF MAP001- INCURS-LOCI = MAP001-FILED0003-T.            …フォーカス位置を特定
THEN
    :
```

出力論理マップのフォーカス制御項目に，フォーカスを設定するオブジェクトのフォーカス定数を格納します。画面上で，フォーカスのあるオブジェクトのフォーカス定数は，入力論理マップのフォーカス制御項目で受け取ります。

オブジェクトごとのフォーカス定数は，XMAP3 が自動的に生成します。なお，定数は，各オブジェクトの AP が受け取る項目のデータ名に「T」を付けて，定数テーブルとしています。このため，画面のオブジェクトの位置を変更しても，AP を変更する必要はありません。定数テーブルの値は，初期カーソル設定でも使用できるので，フィールドボックスへのフォーカスと，ボックス内の入出力フィールドへのカーソルを同時に設定できます。

5 マップ帳票での AP のコーディング方法

この章では，マップ帳票用の AP を作成する方法について説明します。

5.1　XMAP3 の帳票用 AP で実行する処理

ここでは，XMAP3 の帳票用 AP で実行する処理の概要と，コーディング方法について説明します。

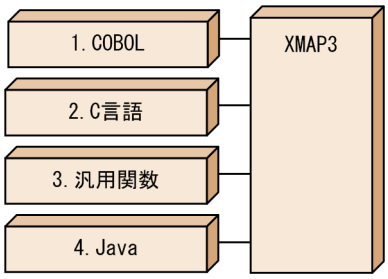
なお，XMAP3 の API はマルチスレッドで動作する AP からの実行に対応していません。マルチスレッドで動作する AP では，XMAP3 の API を使わないようにしてください。

5.1.1　XMAP3 での帳票の出力

(1)　帳票用 AP で使用する API の種類

XMAP3 には，帳票用 AP の開発に使用するプログラミング言語ごとに API を提供しています。API の種類を次の図に示します。

図 5–1　帳票用 AP で使用する API の種類



解説

1. COBOL の命令文です。SEND 文を使用する方法と CALL 文を使用する方法があります。Windows/UNIX で共通の文法です。
2. XMAP3 が提供している C 言語用の関数です。Windows/UNIX で共通の文法です。
3. XMAP3 が提供している Windows 専用の C 言語または C++用の汎用関数です。
4. XMAP3 が提供している Java 用のクラスです。Web 環境で使用できます。詳細については,「第 4 編　Web 用の AP 開発」を参照してください。

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は，COBOL AP（SEND 文を使用する方法）だけ利用できます。

(2)　帳票用 AP で使用する API の概要

帳票用 AP で使用する要求と API の関係を次の表に示します。また，帳票用 AP と XMAP3 との処理の概要を次の図に示します。

表 5–1　帳票用 AP で使用する要求と API の関係

要求	COBOL の命令文	COBOL の CALL 文	C 言語	汎用関数
オープン※	最初の SEND 文発行時	'OPEN'要求	'OPEN'要求	XmapDrvCreateOpen
印刷要求	SEND 文	'SEND'要求	'SEND'要求	XmapDrvSend
クローズ	DISABLE 文, GOBACK 文,	'CLOS'要求	'CLOS'要求	XmapDrvClose

要求	COBOL の命令文	COBOL の CALL 文	C 言語	汎用関数
	EXIT PROGRAM 文, CANCEL 文, AP の STOP RUN			

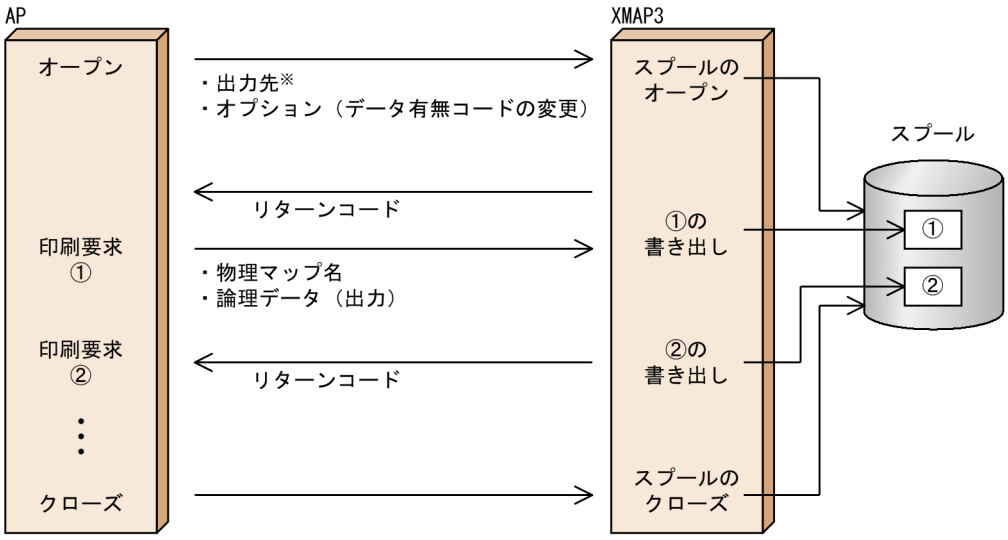
注

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合は，COBOL AP（SEND 文を使用する方法）だけ利用できます。

注※

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

図 5–2　帳票用 AP と XMAP3 との処理の概要



注※　必ず指定します。

! 注意事項

AP から SEND 命令を出しても，プリンタの解放やプログラムの終了処理などによって，印刷処理されないことがあります。

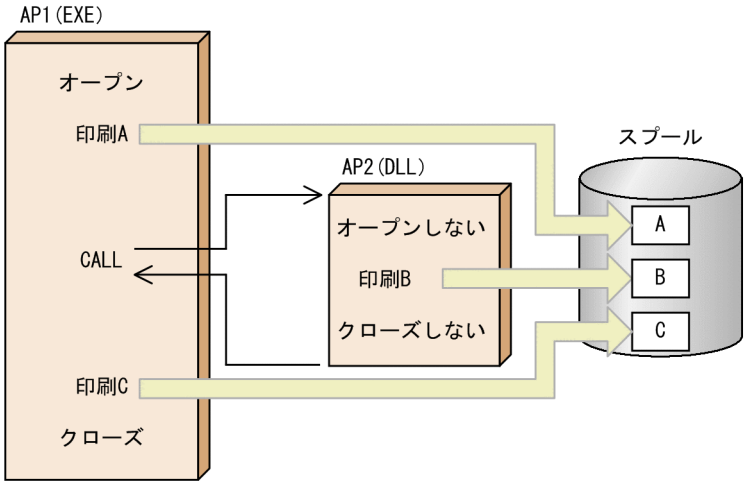
5.1.2　AP 間でオープンを引き継ぐ場合

共通帳票を印刷する部分を別 DLL 化したり，プログラムが大きいので分割したりする場合などは，オープンをプログラム間で引き継がせることをお勧めします。COBOL の SEND 文を使用した場合，AP 間でオープンを引き継ぐことができます。オープンを引き継ぐためには，COBOL の場合，プログラムを実行させる AP（ここでは AP1）で環境変数「CBLTERMSHAR=YES」を指定しておきます。

COBOL の CALL 文，C 言語，および汎用関数を使用する場合は，インタフェース領域を受け渡しすることで，AP 間のオープンを引き継ぐことができます。

オープンを引き継ぐことによって，AP1 から AP2 を呼び出すとき，AP1 でのクローズ（COBOL のとき DISABLE 文の発行）および AP2 でのオープンは発生しません。そのため帳票印刷の時間が短くなります。

図 5-3　AP 間でオープンを引き継ぐ場合

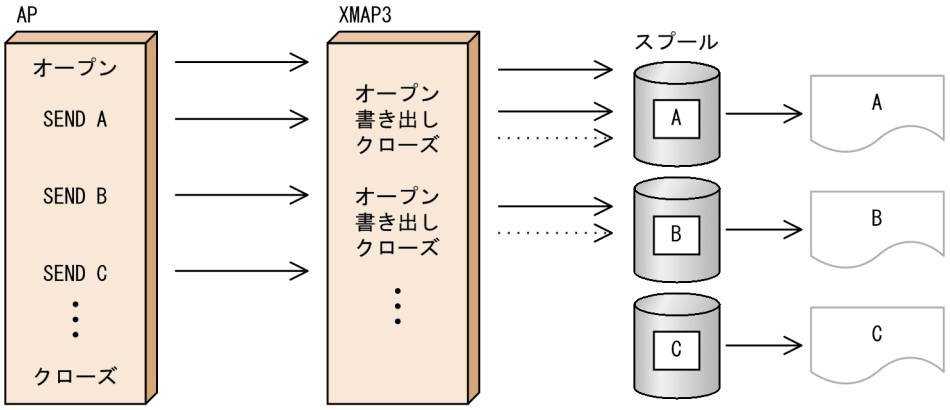


5.1.3　スプールの設定と印刷ページの関係

1 ページを 1 文書でスプール登録する場合と，複数ページを 1 文書でスプール登録する場合について説明します。

(1)　1 ページを 1 文書でスプール登録する

AP と XMAP3 の間で，AP が 1 ページ出力したら，1 ページを一つの文書としてスプール登録および印刷するようにします。



APのSEND命令でXMAP3がオープンとクローズ処理をします。

- Windows の場合

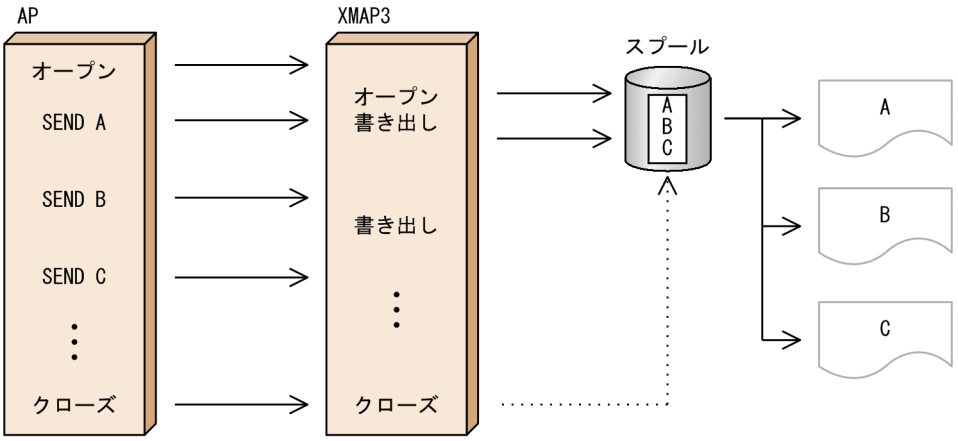
  スプールへの出力単位は，表示・印刷セットアップの「プリンタタブ」の「スプール書き出し単位」の指定に従います。「スプール書き出し単位」は，「1 ページ毎」を選択します。詳細については，マニュアル「XMAP3 実行ガイド」を参照してください。

- UNIX の場合

  1 ページごとに出力するようにするには，XMAP3 サーバを起動する前に環境変数に「XPWCSTMW=OFF」を設定するか，この環境変数を設定しないでください。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

(2)　複数ページを 1 文書でスプール登録する

AP と XMAP3 の間で，AP から複数ページの出力を要求して，AP からのクローズ要求で一つの文書としてスプール登録および印刷するようにします。



APからの最初のSEND命令でXMAP3がオープンし，クローズ命令でクローズ処理をします。

- Windows の場合

  スプールへの出力単位は，表示・印刷セットアップの「プリンタタブ」の「スプール書き出し単位」の指定に従います。「スプール書き出し単位」は，「アプリケーション毎」を選択します。詳細については，マニュアル「XMAP3 実行ガイド」を参照してください。

- UNIX の場合

  複数ページを出力するようにするには，XMAP3 サーバを起動する前に，環境変数に「XPWCSTMW=ON」を設定します。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

! 注意事項

PDF ファイル出力の場合，「アプリケーション毎」を選択してください。1 枚目に指定した印刷ドキュメント名の PDF ファイルにすべての帳票が格納されます。「1 ページ毎」を選択して，印刷ドキュメント名が同じ帳票を連続出力すると，出力のたびに PDF ファイルが上書きされるので，クローズ直前に出力した PDF ファイルだけが登録されます。

5.1.4　AP 分割時の注意

COBOL では，明示的に CALL 文でオープンを要求する場合を除き，一つのコンパイル単位で最初の SEND 文が発行されると帳票がオープンされます。そのため，1 帳票が 1 実行ファイル（.exe）のような構成にすると，1 帳票（1 枚）ごとにオープンすることになります。この場合，オープン・クローズ，SEND 文を発行する実行ファイル，ビジネス処理をする実行ファイルをそれぞれ分けると実行性能が高くなります。ビジネス処理をする実行ファイルは，DLL ファイルでも代用できます。EXE ファイルにするか DLL ファイルにするかは，処理の形態によって次のように使い分けてください。

- .exe：処理が続く場合（EXE ファイル間でのデータ受け渡しはない）
- .dll：処理が単独で閉じる場合（1 ウィンドウで閉じる）

また，複数のコンパイル単位のものを合わせて一つの実行ファイルにするときは，各コンパイル単位でオープンを発行しないようにするため，COBOL の実行支援の環境変数で，「CBLTERMSHAR=YES」を指定

します。「CBLTERMSHAR=YES」は，SEND 文で AP を作成したときだけ有効です。CALL 文で AP を作成したときは無効になります。

CALL 文で AP を作成する場合，AP 間のオープン引き継ぎをするときは，XMAP3 インタフェースエリアの情報を引き継ぐようなコーディングをする必要があります。

5.1.5　AP の命令と XMAP3 の関係

AP 中で使用されている命令は，具体的に XMAP3 をどのような状態にするのか，またプログラミング言語別に，どのような文や関数が使用されているかを示します。

(1)　AP 中の命令と XMAP3

一般に，プログラムの中で「オープン」「出力」「クローズ」と呼ばれている命令に対して，XMAP3 ではどのような状態なのかを次の表に示します。

表 5–2　プログラムの命令文と XMAP3

命令文	XMAP3 の状態
オープン	XMAP3 が使用できる状態になります。 利用するスプールおよびドライバを確保します。
出力	物理マップ名※1，出力先※2，出力データを渡して，印刷要求や実行結果をリターンコードとして返します。
クローズ	XMAP3 が使用できない状態になります。 利用していたスプールやドライバを解放します。

注※1

物理マップ名とは，ドローのマップ名で付けた名前に ID（マップ名が 6 文字のときは 6A，6H，6G など，マップ名が 7 文字のときは P，L，G など）が付いたものです。

注※2

仮想端末名のことです。仮想端末名は，スタンドアロンの場合は表示・印刷セットアップで，C/S 構成の場合は C/S セットアップで指定します。デフォルトは「PRT001」です。

(2)　COBOL による命令と XMAP3

XMAP3 に対する命令を COBOL で表現する場合を次の表に示します。

表 5–3　COBOL での表現

命令文	COBOL での表現
オープン	明示的にはありません。 プログラム中の最初の SEND 文が発行された場合，「オープン」を仮定します。
出力	SEND 文
クローズ※	DISABLE 文

注※

プログラムが終了した場合，DISABLE 文がなくても「クローズ」が仮定されます。

(3)　COBOL の CALL 文や C 言語の関数として使用する場合

COBOL の CALL 文や C 言語などの関数呼び出しで，「オープン」「出力」「クローズ」命令を実行できます。

5.1.6　帳票出力時の XMAP3 と AP の関係

XMAP3 から帳票を印刷する場合の，印刷形態と AP のコーディング内容の関係を示します。なお，ここではプログラミング言語として COBOL による命令が使用されていることを仮定します。

(1)　同じ内容のものを複数枚印刷（コピー印刷）

同じ内容のものを複数枚，印刷する場合です。例えば，3 枚の帳票を印刷するときは，次の表に示す方法でコーディングします。

注

PDF ファイルへ出力する場合，コピー印刷はしません。

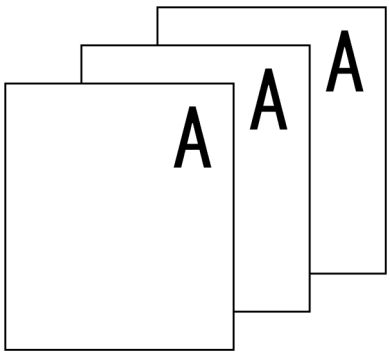
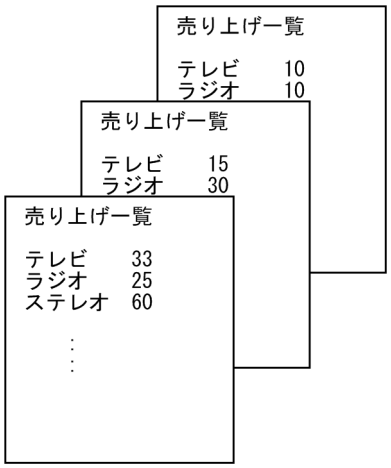


表 5-4　コピー印刷時のコーディング

分類	AP のコーディング	備考
シリアルプリンタ	SEND 文を 3 回発行します。	スプールに同じものを 3 部入れます。
ページプリンタ	何部，印刷するかを指定して SEND 文を 1 回発行します。	プリンタハードウェアの機能を使用してマルチコピーします。スプールの中には 1 部だけ入れます。

(2)　同じ帳票レイアウトで中身が異なるものを複数枚印刷

マップ名は同じまま，論理マップ中のデータを変えて，3 回 SEND 文を発行します。

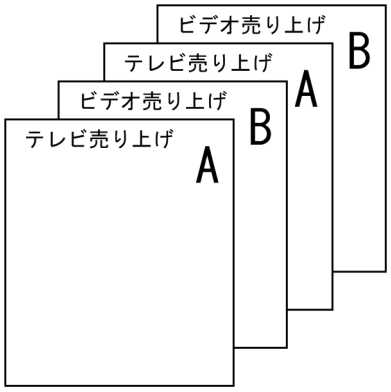


SEND 文の使用例

```
SEND MAP1,データ 1
SEND MAP1,データ 2
SEND MAP1,データ 3
```

(3)　異なる帳票を交互に印刷

それぞれの帳票を印刷するための SEND 文を交互に書きます。



SEND 文の使用例

```
SEND MAP-A,データ 1
SEND MAP-B,データ 2
SEND MAP-A,データ 1
SEND MAP-B,データ 2
```

(4)　異なるプリンタに帳票を印刷

仮想端末名を別プリンタに割り当てて，それぞれに SEND 文を書きます。ただし，スタンドアロンの場合は，途中に DISABLE 文などを入れ，一度クローズする必要があります。

5.2　COBOL での印刷命令

ここでは，COBOL で帳票用の AP を作成する方法について説明します。

5.2.1　コーディング前の準備（Windows）

コーディングをする前に，次の準備をしてください。

1. **AP を格納するためのフォルダを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するフォルダを作成します。開発環境に合わせてフォルダを分類し，作成位置や名称を決めます。

2. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要な登録集原文を標準提供しています。1.で作成したフォルダにコピーして使用することをお勧めします。このとき，コーディングのひな型である AP パターンもコピーしておきます。

   標準提供の登録集原文を次に示します。

   - **動的変更テーブル**（X3MODTBL.CBL）

     出力論理マップの初期化に使用する定数や修飾名が指定されています。

   - **インタフェース領域**（JSVWATBL.CBL）

     CALL 文で帳票を印刷するときにパラメタとして使用するインタフェース領域が指定されています。

5.2.2　コーディング前の準備（UNIX）

コーディングをする前に，次の準備をしてください。

1. **XMAP3 Developer で作成したファイルを転送する**

   XMAP3 Developer で UNIX 用に作成した物理マップ，論理マップおよび動的変更テーブルを UNIX マシンへファイル転送します。

   - **物理マップのファイル転送**

     XMAP3 Developer で UNIX 用に作成した物理マップ（拡張子「.pmp」）ファイルは，バイナリ形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

   - **論理マップ・動的変更テーブルのファイル転送**

     XMAP3 Developer で UNIX 用に作成した論理マップ（拡張子「.cbl」）ファイルおよび動的変更テーブル（X3MODTBL.cbl）ファイルは，テキスト形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください。

     XMAP3 Developer で作成する動的変更テーブルは，標準の出力先のままで作成した場合，次に出力されています。別の出力先に作成した場合は，そのファイルをファイル転送してください。

   XMAP3インストールフォルダ¥INCLUDE¥X3MODTBL.cbl

2. **物理マップを移行し形式チェックをする**

   XMAP3 Developer で作成した UNIX 用の物理マップを UNIX マシンへ転送したあと，cmapcp コマンドを使って，物理マップを実行環境に移行します。

cmapcp コマンドは，次に示すファイルを実行環境に移行し，さらに UNIX 版 XMAP3 で使用できる物理マップの形式かどうかのチェックをします。

- 拡張子が「.pmap」の物理マップファイル
- 拡張子が「.pmp」の物理マップファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

cmapcp コマンドについては，「15.1.1　cmapcp コマンド」を参照してください。

3. **UNIX の EUC 環境で利用する場合，論理マップおよび動的変更テーブルを文字コード変換する**

   XMAP3 Developer で生成される論理マップおよび動的変更ファイルは，文字コードがシフト JIS であるため，UNIX の EUC 環境で利用するには，ファイルを転送したあと，シフト JIS から EUC へコード変換する必要があります。

4. **AP を格納するためのディレクトリを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するディレクトリを作成します。開発環境に合わせてディレクトリを分類し，作成位置や名称を決めます。

5. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要な登録集原文を標準提供しています。4.で作成したフォルダにコピーして使用することをお勧めします。このとき，AP の作成に使う論理マップ，コーディングのひな型である AP パターン，AP 実行時に使用する物理マップもコピーしておきます。

   - **動的変更テーブル**（X3MODTBL.CBL）

     出力エリア（出力論理マップ）の初期化に使用する定数が指定されています。なお，XMAP3 Developer で UNIX 用の動的変更テーブルを作成した場合は，標準提供のテーブルは使用しないでください。必ず，XMAP3 Developer で作成した動的変更テーブルをファイル転送して使用してください。

   - **インタフェース領域**（JSVWATBL.CBL）

     CALL 文で帳票を印刷するときにパラメタとして使用するインタフェース領域が指定されています。SEND 文，RECEIVE 文，または TRANSCEIVE 文を使う場合は必要ありません。

5.2.3　仮想端末の自動割当て

C/S 構成の環境で，サーバ上の AP から複数のクライアントマシンへ帳票印刷をする場合，仮想端末の自動割当て機能を利用できます。この機能は，C/S 構成の場合だけ利用できる機能です。

仮想端末の自動割当て機能は，サーバ AP が各クライアントマシンに対応した仮想端末名を意識することなく一つの仮想端末名だけを意識していれば，各クライアントマシンにある XMAP3 の印刷サービスが起動されたタイミングで，AP から送信された情報を印刷できます。仮想端末の自動割当て機能を利用するときはサーバ側で，AP で指定した自動割当て用の仮想端末名を設定します。

仮想端末の自動割当てについては，マニュアル「XMAP3 実行ガイド」を参照してください。

5.2.4　論理マップの取り込み方法

AP 中に論理マップを取り込む場合，COBOL の WORKING-STORAGE SECTION，または LINKAGE SECTION に COPY 文を指定します。ただし，論理マップ中に定数を展開しているものについては，LINKAGE SECTION に取り込めません。

AP に論理マップ MAP301O を取り込む例を次に示します。

例

```
WORKING-STORAGE SECTION.
 COPY MAP3010.    …… 出力論理マップの取り込み
 COPY X3MODTBL.   …… 動的変更テーブルの取り込み
```

5.2.5　帳票印刷命令

AP からプリンタに対して帳票を印刷するには，次に示す方法があります。

- COBOL の SEND 文による方法
- COBOL の CALL 文による方法

Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合，COBOL の SEND 文による方法は利用できます。COBOL の CALL 文による方法は利用できません。

(1)　SEND 文による印刷

COBOL の SEND 文によって帳票の印刷を要求します。通信記述項，SEND 文，および DISABLE 文については，「11.2.1　SEND 文による印刷」を参照してください。

COBOL については，マニュアル「COBOL2002 言語 拡張仕様編」を参照してください。

(a)　その他

XMAP3 オープンの引き継ぎ

XMAP3 に対するオープン命令は，最初の SEND 文が発行されたときに認識されます。また，AP が終了する場合は，クローズ命令が発行されたと仮定されます。

別々にコンパイルした AP 間では，XMAP3 のオープン状態を引き継げます。AP 間でオープンを引き継ぐ方法については，「5.2.6　COBOL でのコンパイル（Windows)」を参照してください。

(2)　CALL 文による印刷

ここでは，COBOL の CALL 文によってマッピングライブラリを使用する場合の説明をします。

(a)　環境部

CALL 文を使用する場合の環境部（ENVIRONMENT DIVISION）の定義を次に示します。

```
ENVIRONMENT         DIVISION.
         CONFIGURATION         SECTION.
                   SPECIAL-NAMES.
                   STDCALL IS 一意名1.
         EXTERNAL-PROGRAM      SECTION.
                   CALL-CONVENTION.
                   'jsvwadrv' IS 一意名1.
```

注　一意名 1 は，任意に設定してください。

環境部に上記の指定をした場合には，コンパイル時に次の注意が必要です。

1. コンパイラオプションに次の指定をします。

   -Comp5

   -JPN,Alnum

2. リンカオプションに次の内容を指定します。

- リンカオプション：指定しない
- インポートライブラリ／ユーザ指定ライブラリ：
  XMAP3 インストールフォルダ¥Lib¥X3mwdr32.lib

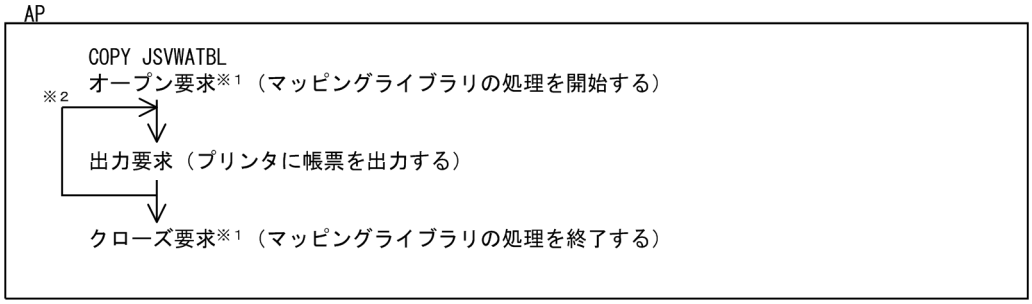
環境部に上記のコーディングを追加しない場合は，コンパイル時に次の指定をする必要があります。

1. エディタを使用してファイルを作成します。
   - ファイル内容：jsvwadrv
   - ファイル名　：XXXX.cbw（XXXX は任意）
   - ファイルの格納場所：AP と同じディレクトリ（またはフォルダ）
2. コンパイラオプションに次の指定をします。
   -Comp5
   -StdCall
   -JPN,Alnum
   stdcall 呼び出し指示ファイル名には，1.で作成したファイル名を指定します。
3. リンカオプションに次の内容を指定します。
   - リンカオプション：指定しない
   - インポートライブラリ／ユーザ指定ライブラリ：
     XMAP3 インストールフォルダ¥Lib¥X3mwdr32.lib

(b)　CALL 文の発行順序

マッピングライブラリの機能と命令の発行順序を次の図に示します。

図 5-4　マッピングライブラリの機能と命令の発行順序



注※1　ディスプレイとプリンタを使う場合，それぞれに必要です。
注※2　帳票の数だけ要求を繰り返します。

CALL 文の形式については，「11.2.2　CALL 文による印刷」を参照してください。

5.2.6　COBOL でのコンパイル（Windows）

(1)　COBOL 開発マネージャを使用したコンパイルと実行のポイント

COBOL 開発マネージャを使用したコンパイルと実行のポイントについて説明します。

(a)　COBOL 開発マネージャの概要

COBOL 開発マネージャは，COBOL2002 から提供されている COBOL プログラムの統合的な開発環境です。COBOL で AP を開発するときに必要な COBOL ソースや登録集原文などの資産を依存関係に従って管理し，コンパイルなどの作業を自動化します。

COBOL 開発マネージャを使用する場合，次の手順で AP を作成します。

1. プロジェクトの作成
2. 資産の登録，定義
3. ビルド，リビルド

次に，それぞれの手順について説明します。

プロジェクトの作成

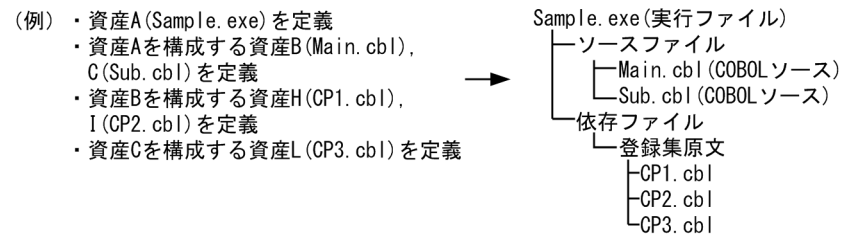
COBOL 開発マネージャでは，一つの EXE または DLL ファイルの開発単位を「プロジェクト」と呼んでいます。COBOL 開発マネージャを使用して AP を作成する場合，このプロジェクトを定義する必要があります。

資産の登録，定義

COBOL 開発マネージャでは，プロジェクトで作成される実行のファイル（.exe，または.dll）やそれを構成する要素を「資産」と呼んでいます。また，COBOL 開発マネージャでは，その依存関係を定義する必要があります。

資産の依存関係を定義すると，COBOL 開発マネージャではどのように表されるかを次の図に示します。

図 5-5　COBOL 開発マネージャでの表示



ビルド，リビルド

COBOL 開発マネージャでは，定義した資産の依存関係に基づいてコンパイルとリンケージをします。その方法として，各資産の依存関係とタイムスタンプ（ファイル作成／修正日時）の前後関係に基づいてコンパイルとリンケージをする「ビルド」があります。例えば，「実行ファイルの作成／修正日時」より実行ファイルが取り込んでいる「COBOL ソースの作成／修正日時」の方が新しい場合にコンパイルとリンケージが実行されます。

また，タイムスタンプには関係なくコンパイルとリンケージをする「リビルド」もあります。

(b)　コンパイル時のポイント

コンパイラオプションを指定する

COBOL を使用したとき，指定が必要なコンパイラオプションを次に示します。

- -Comp5：COBOL プログラム中の COMP-5 を利用できるようにするオプション
- -JPN,Alnum：論理マップ内で日本語項目を扱えるようにするオプション

AP 間でオープンを引き継ぐ

AP を分割してコンパイルするときは，コンパイル単位で XMAP3 のオープン・クローズ要求が毎回発行されないようにします。COBOL の実行支援の環境変数で，「CBLTERMSHAR=YES」を指定して，オープンを各プログラムで引き継ぐようにします。「CBLTERMSHAR=YES」は，SEND 文で AP を作成したときだけ有効です。CALL 文で AP を作成したときは無効になります。

登録集原文を格納するフォルダを確認する

- AP の COPY 文に，論理マップ（マップ生成時に付けられた名称）を間違って指定していないか確認してください。
- 論理マップがフォルダ中に用意されているか確認してください。ソースプログラムと登録集原文を同じフォルダで管理している場合は，論理マップが同一フォルダに格納されていないことが考えられます。また，ソースプログラムと登録集原文を別のフォルダで管理している場合は，環境変数（CBLLIB）に指定した登録集原文のフォルダに誤りがあったり，指定したフォルダに論理マップが格納されていなかったりすることが考えられます。
- データ名，または変数名に不当な文字を指定していることが考えられます。データ名，および変数名に不当な文字を AP で指定していないか確認してください。

(c)　リンケージ時のポイント

XMAP3 使用時のリンケージオプションを指定する

インポートライブラリ／ユーザ作成ライブラリに次のファイルを指定します。

XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

XMAP3 のライブラリを設定する

コンパイル環境（CALL 文で作成した AP の場合は，リンケージ環境）のマシンには，XMAP3 の開発環境をインストールしておく必要があります。

(d)　実行時のポイント

AP を実行する前に，物理マップを実行形式ファイルと同じフォルダに格納してください。

また，マップパスを指定する方法もあります。マップパスは，表示・印刷セットアップで指定します。表示・印刷セットアップについては，マニュアル「XMAP3 実行ガイド」を参照してください。

(2)　COBOL 開発マネージャでの XMAP3 の利用方法

COBOL 開発マネージャと XMAP3 との両方をインストールしている場合，COBOL 開発マネージャから XMAP3 の機能を呼び出せます。XMAP3 の資産を登録，定義すれば，ビルド，またはリビルドによってコンパイルとリンケージをして AP を作成できます。

また，AP 作成時にソースプログラムや登録集原文を格納するフォルダを作成しておいてください。

次に，COBOL 開発マネージャから XMAP3 を利用する方法について説明します。

(a)　COBOL 開発マネージャ上での資産の登録，定義

COBOL 開発マネージャ上では，XMAP3 の資産を扱う場合，XMAP3 でのファイル名の規則に従います。また，XMAP3 の「マップ名」を基に作成します。

XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則を次の表に示します。

表 5-5　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則

XMAP3 の資産	COBOL 開発マネージャでの名称	COBOL 開発マネージャで付けるファイル名	内容
マップ定義	マップ定義	マップ名.imp	ユーザが定義した帳票の情報を格納している。定義を修正するときにはこのファイル名を指定。
論理マップ	登録集原文	マップ名＋ O.cbl	AP で取り込む出力情報の登録集原文。
物理マップ	－	マップ名＋ XX※.pmp	ユーザが定義した帳票のうち，AP ではアクセスしない，固定部分の情報。 COBOL 開発マネージャの管理対象外。

（凡例）

－：なし。

注※

XX は，デバイス ID を示します。デバイス ID については，「11.2.1(1)　通信記述項」を参照してください。

また，COBOL 開発マネージャから XMAP3 の資産を登録，定義した場合，どのように表されるかを次に示します。

```
ソースファイル
 ├ユーザプログラム(COBOLソース)←XMAP3の機能で定義した帳票を利用するAP
 └マップ名.imp(マップ定義ファイル)
依存ファイル
 └マップ名+0.cbl(登録集原文)
```

これを基にして，XMAP3 で作成した帳票を COBOL 開発マネージャで利用して AP を作成する場合の例を次に示します。

（例）XMAP3で作成した帳票(マップ名:REPORT)を利用してAPを作成する場合

```
APP.exe(実行ファイル)
 ├ソースファイル
 │  ├Sample.cbl(COBOLソース)          ←1.AP
 │  └REPORT.imp(マップ定義ファイル)  ←2.帳票定義ファイル
 └依存ファイル
    └登録集原文
       └REPORT0.cbl(登録集原文)        ←3.出力論理マップ
```

注意

- ファイル名は，「表 5-5　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則」に従って定義する必要があります。
- ファイル名は，XMAP3 のマップ名を使用して作成してください。XMAP3 のファイル名を使用しないと，上記に示す資産定義と一致しなくなります。

このように資産を登録，定義しておけば，定義ファイルに変更があった場合，ビルドによってコンパイル，リンケージされ，自動的に登録集原文（論理マップ）が生成し直されます。

(b)　マップ定義ファイルの作成と修正

マップ定義ファイルを作成，または修正したい場合，次に示す操作をすることで XMAP3 の帳票定義ができます。

- COBOL 開発マネージャ上で，マップ定義ファイル（.imp）をダブルクリックします。

(c)　マップ生成

COBOL 開発マネージャのビルド，またはリビルドを使用しないで単独にマップ生成をする場合，次の操作をすることでマップ生成ができます。生成される論理マップや物理マップのファイル名については，「表 5-5　XMAP3 の資産を COBOL 開発マネージャで利用するときのファイル名の規則」を参照してください。

マップ生成時に使用するリトルエンディアン，およびビッグエンディアンは，オプションで指定してください。

操作方法

- 特定のマップ定義ファイルからマップ生成をするときにだけ有効とする場合
  COBOL 開発マネージャ上で，マップ定義ファイル（.imp）をクリックして，［ファイル］－［ファイルの設定］を選択し，必要なコンパイルオプションを指定します。
- すべてのマップ定義ファイルからマップ生成をするときに有効とする場合
  COBOL 開発マネージャ上で，［プロジェクト］－［プロジェクトの設定］を選択し，「最適化」タブで必要なコンパイルオプションを指定します。

設定形式

[-BigEndian,Bin]

説明

-BigEndian,Bin または-Bb は，ビッグエンディアンを指定することを示します。指定がない場合，リトルエンディアンが仮定されます。

5.2.7　COBOL でのコンパイル（UNIX）

COBOL2002 で作成した，帳票出力用 AP のコンパイル方法を説明します。

使用する UNIX およびターゲットの文字コード（シフト JIS，EUC）に応じて手順が異なります。

(1)　AIX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した帳票出力用 AP のソースプログラム（xxx.cbl）を AIX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）

- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

AIX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 5-6　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

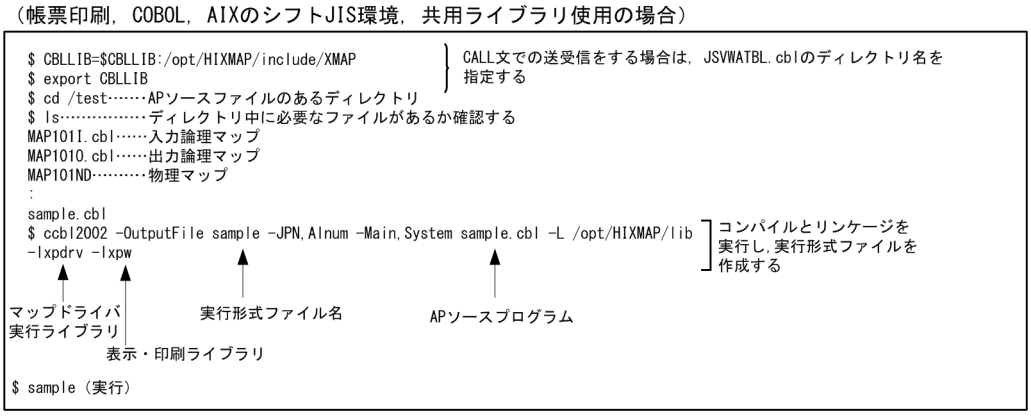
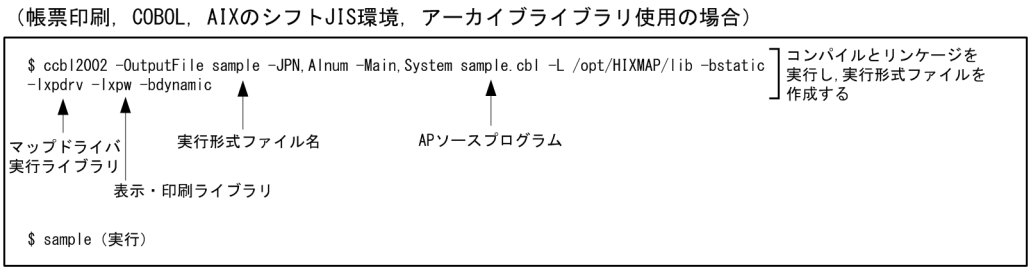


図 5-7　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



(2)　AIX 環境でのコンパイル（EUC）

COBOL2002 で記述した帳票出力用 AP のソースプログラム（xxx.cbl）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

! 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

AIX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 5-8　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

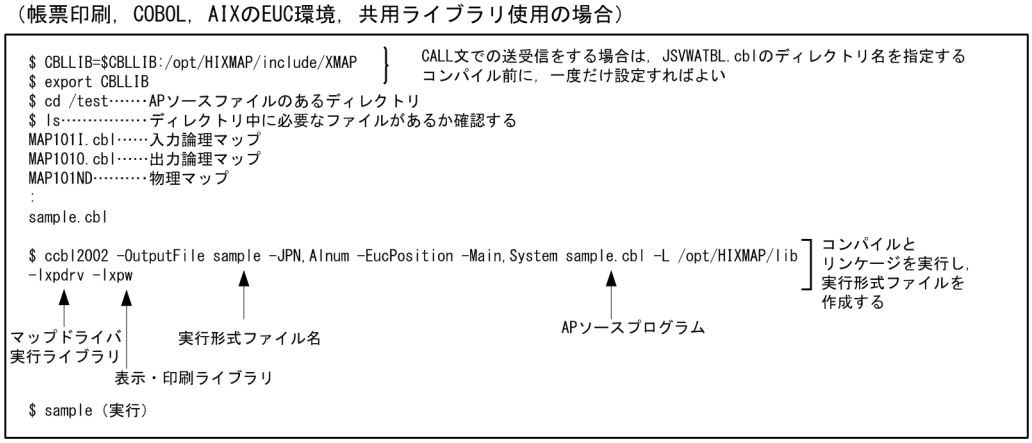
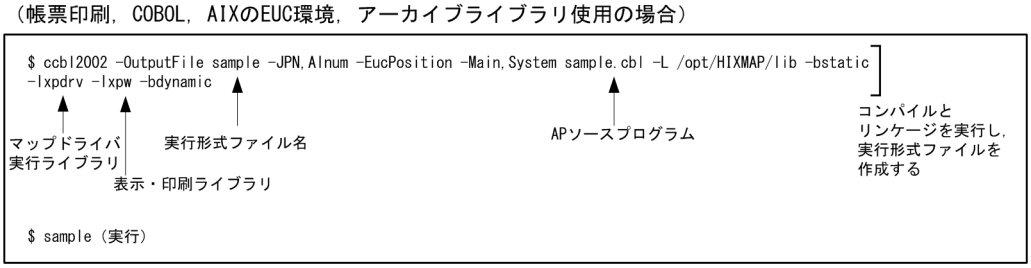


図 5-9　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



(3)　HP-UX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した帳票出力用 AP のソースプログラム（xxx.cbl）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

HP-UX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 5-10　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

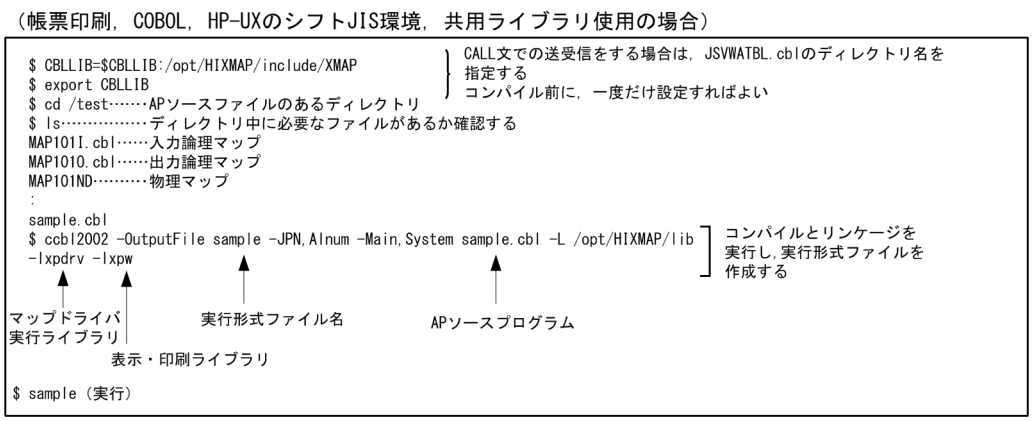
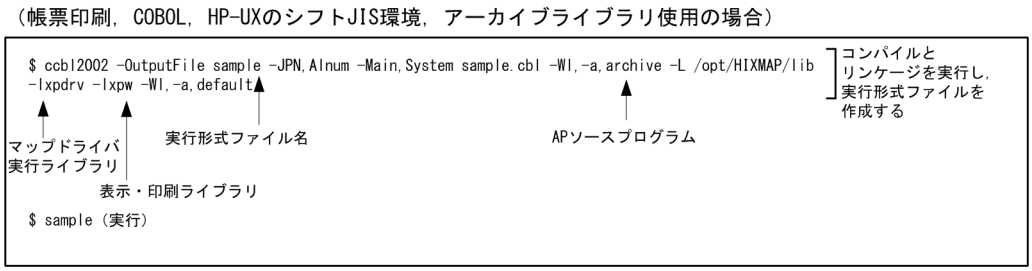


図 5-11　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



(4)　HP-UX 環境でのコンパイル（EUC）

COBOL2002 で記述した帳票出力用 AP のソースプログラム（xxx.cbl）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　環境変数の設定

XMAP3 Server Runtime で提供する登録集原文を COPY 文で AP に取り込む場合は，環境変数 CBLLIB に XMAP3 Server Runtime が提供する登録集原文のパスを設定します。

```
CBLLIB=$CBLLIB:/opt/HIXMAP/include/XMAP
export CBLLIB
```

(b)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(c)　コンパイル例

HP-UX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 5-12　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

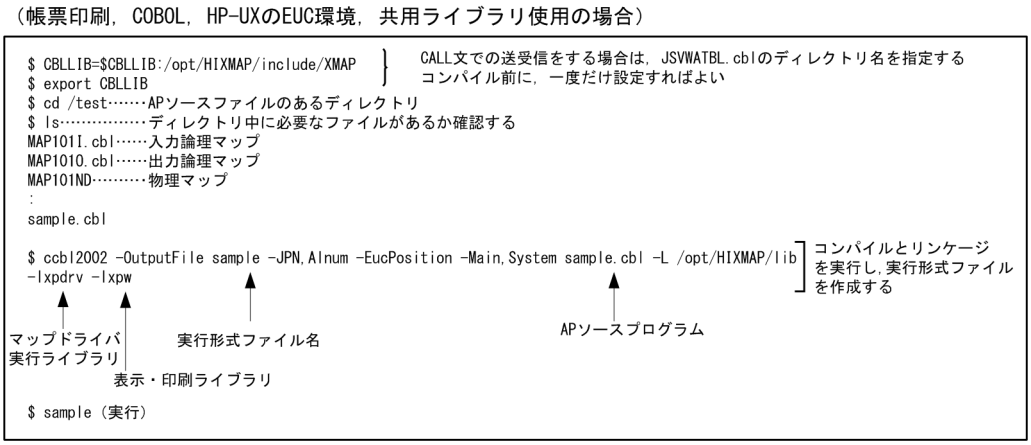
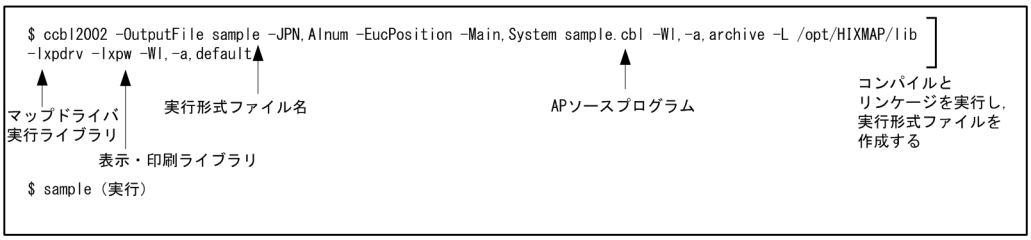


図 5-13　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



5.3　C 言語での印刷命令

この節では，AP 中に論理マップを取り込む方法，および C 言語によるマッピングライブラリ用の jsvwadrv 関数の使用方法を説明します。

5.3.1　コーディング前の準備（Windows）

コーディングをする前に，次の準備をしてください。C 言語にないクラスなどの概念を持った C++で AP を作成する場合には，汎用関数を使用することをお勧めします。詳細については，「5.4　汎用関数での印刷命令」を参照してください。

1. **AP を格納するためのフォルダを作成する**

   ソースプログラムやコンパイル後に生成される作業ファイルを格納するフォルダを作成します。開発環境に合わせてフォルダを分類し，作成位置や名称を決めます。

2. **XMAP3 の標準提供ファイルから，必要なファイルをコピーする**

   XMAP3 では，AP の作成に必要なヘッダファイル※を標準提供しています。1.で作成したフォルダにコピーして使用することをお勧めします。このとき，コーディングのひな型である AP パターンもコピーしておきます。

注※

標準提供のヘッダファイルを次に示します。

- **動的変更テーブル**（X3MODTBL.H）

  出力論理マップの初期化に使用する定数が指定されています。

- **インタフェース領域**（JSVWATBL.H）

  jsvwadrv 関数で帳票を印刷するときにパラメタとして使用するインタフェース領域が指定されています。

5.3.2　コーディング前の準備（UNIX）

コーディングをする前に，次の準備をしてください。

1. **XMAP3 Developer で作成したファイルを転送する**

   XMAP3 Developer で UNIX 用に作成した物理マップ，論理マップおよび動的変更テーブルを UNIX マシンへファイル転送します。

   - **物理マップのファイル転送**

     XMAP3 Developer で UNIX 用に作成した物理マップ（拡張子「.pmp」）ファイルは，バイナリ形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

   - **論理マップ・動的変更テーブルのファイル転送**

     XMAP3 Developer で UNIX 用に作成した論理マップ（拡張子「.h」）ファイルおよび動的変更テーブル（X3MODTBL.h）ファイルは，テキスト形式でファイル転送する必要があります。

     ファイルの拡張子はそのままでファイル転送してください。

     XMAP3 Developer で作成する動的変更テーブルは，標準の出力先のままで作成した場合，次に出力されています。別の出力先に作成した場合は，そのファイルをファイル転送してください。

   XMAP3インストールフォルダ¥INCLUDE¥X3MODTBL.h

2. 物理マップを移行し形式チェックをする

XMAP3 Developer で作成した UNIX 用の物理マップを UNIX マシンへ転送したあと，cmapcp コマンドを使って，物理マップを実行環境に移行します。

cmapcp コマンドは，次に示すファイルを実行環境に移行し，さらに UNIX 版 XMAP3 で使用できる物理マップの形式かどうかのチェックをします。

- 拡張子が「.pmap」の物理マップファイル
- 拡張子が「.pmp」の物理マップファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

cmapcp コマンドについては，「15.1.1　cmapcp コマンド」を参照してください。

3. UNIX の EUC 環境で利用する場合，論理マップおよび動的変更テーブルを文字コード変換する

XMAP3 Developer で生成される論理マップおよび動的変更ファイルは，文字コードがシフト JIS であるため，UNIX の EUC 環境で利用するには，ファイルを転送したあと，シフト JIS から EUC へコード変換する必要があります。

4. AP を格納するためのディレクトリを作成する

ソースプログラムやコンパイル後に生成される作業ファイルを格納するディレクトリを作成します。開発環境に合わせてディレクトリを分類し，作成位置や名称を決めます。

5. XMAP3 の標準提供ファイルから，必要なファイルをコピーする

XMAP3 では，AP の作成に必要なヘッダファイル※を標準提供しています。4.で作成したフォルダにコピーして使用することをお勧めします。このとき，AP の作成に使う論理マップ，コーディングのひな型である AP パターン，AP 実行時に使用する物理マップもコピーしておきます。

注※

標準提供のヘッダファイルを次に示します。

- 動的変更テーブル（X3MODTBL.H）

  出力エリア（出力論理マップ）の初期化に使用する定数が指定されています。なお，XMAP3 Developer で UNIX 用の動的変更テーブルを作成した場合は，標準提供のテーブルは使用しないでください。必ず，XMAP3 Developer で作成した動的変更テーブルをファイル転送して使用してください。

- インタフェース領域（JSVWATBL.H）

  jsvwadrv 関数で帳票を印刷するときにパラメタとして使用するインタフェース領域が指定されています。

5.3.3　仮想端末の自動割当て

C/S 構成の環境で，サーバ上の AP から複数のクライアントマシンへ帳票印刷をする場合，仮想端末の自動割当て機能を利用できます。この機能は，C/S 構成の場合だけ利用できる機能です。

仮想端末の自動割当て機能は，サーバ AP が各クライアントマシンに対応した仮想端末名を意識することなく一つの仮想端末名だけを意識していれば，各クライアントマシンにある XMAP3 の印刷サービスが起動されたタイミングで，AP から送信された情報を印刷できます。仮想端末の自動割当て機能を利用するときはサーバ側で，AP で指定した自動割当て用の仮想端末名を設定します。

仮想端末の自動割当てについては，マニュアル「XMAP3 実行ガイド」を参照してください。

5.3.4　論理マップの取り込み方法

AP 中に論理マップを取り込むには，#include 指示語を使用します。

AP に論理マップ MAP003O.h を取り込む例を次に示します。

```
#include "MAP0030.h" … 出力論理マップの取り込み
```

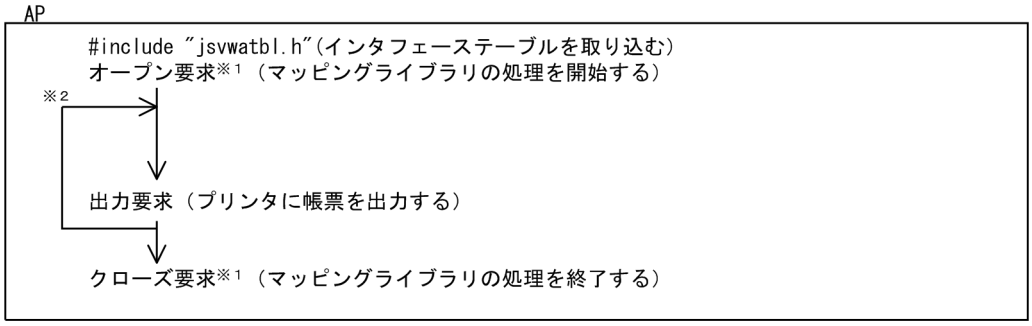
#include 指示語によって AP に実領域が取られます。

5.3.5　帳票印刷命令

(1)　jsvwadrv 関数の発行順序

jsvwadrv 関数の発行順序を次の図に示します。

図 5-14　jsvwadrv 関数の機能と発行順序



注※1　ディスプレイとプリンタを使用する場合，それぞれに必要です。
注※2　帳票の数だけ，要求を繰り返します。

jsvwadrv 関数の形式については，「12.2.1　帳票印刷命令（C 言語)」を参照してください。

5.3.6　C 言語でのコンパイル

C 言語で作成した，帳票出力用 AP のコンパイル方法を OS ごとに説明します。

(1)　Windows 環境でのコンパイル

C 言語のコンパイラを使用してコンパイルしてください。ライブラリには,次に示すファイルを指定する必要があります。

XMAP3 インストールフォルダ¥LIB¥X3MWDR32.LIB

(2)　AIX 環境でのコンパイル（シフト JIS)

C 言語で記述した帳票出力用 AP のソースプログラム（xxx.c）を AIX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -o：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -qmbcs：AP 内でマルチバイト文字（日本語）を使用している場合に指定する

- -I インクルードファイルパス：インクルードファイルの検索パスを指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -bdynamic：共用ライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX のシフト JIS 環境での xlc コマンドによるコンパイル例を次の図に示します。

図 5-15　AIX のシフト JIS 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

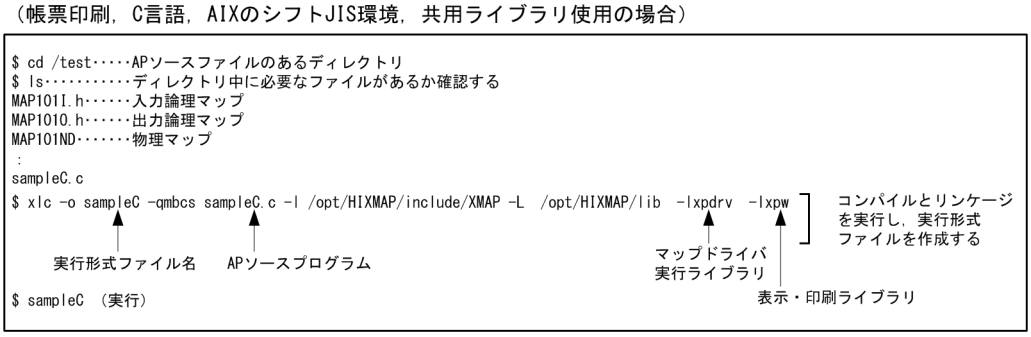
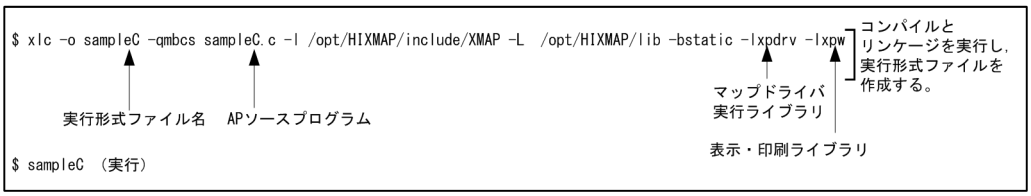


図 5-16　AIX のシフト JIS 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



(3)　AIX 環境でのコンパイル（EUC）

C 言語で記述した帳票出力用 AP のソースプログラム（xxx.c）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

! 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -o：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -qmbcs：AP 内でマルチバイト文字（日本語）を使用している場合に指定する
- -I インクルードファイルパス：インクルードファイルの検索パスを指定する
- -L ライブラリパス：使用ライブラリの検索パスを指定する（必須）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -bdynamic：共用ライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX の EUC 環境での xlc コマンドによるコンパイル例を次の図に示します。

図 5-17　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

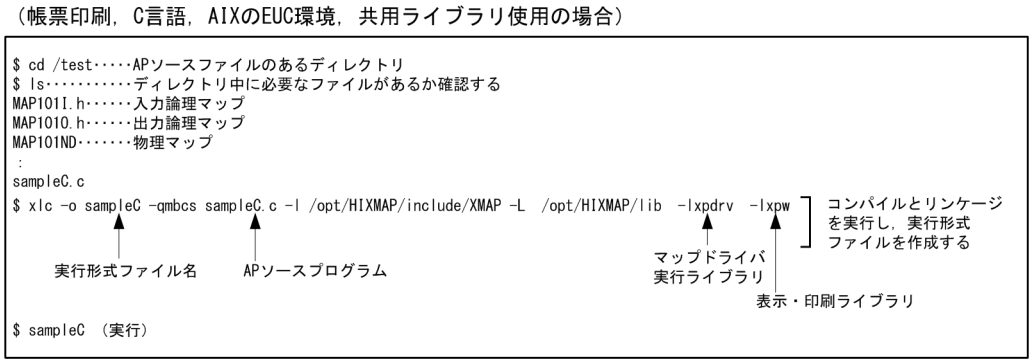
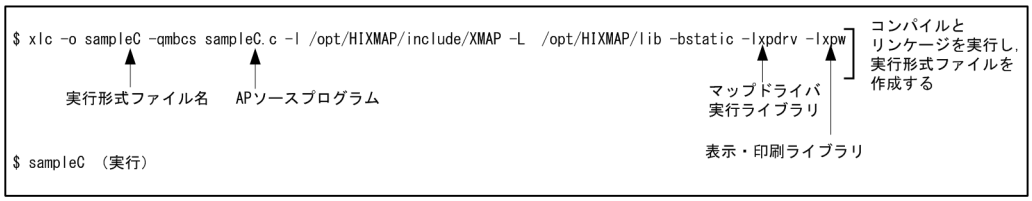


図 5-18　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）



(4)　HP-UX 環境でのコンパイル（シフト JIS）

C 言語で記述した帳票出力用 AP のソースプログラム（xxx.c）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）

- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX のシフト JIS 環境での cc コマンドによるコンパイル例を次の図に示します。

図 5-19　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

（帳票印刷，C言語，HP-UXのシフトJIS環境，共用ライブラリ使用の場合）

```
$ cd /test·····APソースファイルのあるディレクトリ
$ ls···········ディレクトリ中に必要なファイルがあるか確認する
MAP101I.h·····入力論理マップ
MAP101O.h·····出力論理マップ
MAP101ND·····物理マップ
 :
sampleC.c
$ cc -o sampleC sampleC.c -I /opt/HIXMAP/include/XMAP -L /opt/HIXMAP/lib -lxpdrv -lxpw
                                                                                  ] コンパイルとリンケージ
                                                                                    を実行し，実行形式
                                                                                    ファイルを作成する
          実行形式    APソース                                  マップドライバ
          ファイル名  プログラム                                実行ライブラリ
                                                                          表示・印刷ライブラリ

$ sampleC （実行）
```

図 5-20　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）

（帳票印刷，C言語，HP-UXのシフトJIS環境，アーカイブライブラリ使用の場合）

```
          実行形式      APソース
          ファイル名    プログラム

$ cc -o sampleC sampleC.c -I /opt/HIXMAP/include/XMAP -Wl,-a,archive
-L /opt/HIXMAP/lib -lxpdrv -lxpw -Wl,-a,default
                                                                ] コンパイルとリンケージを実行し，
                                                                  実行形式ファイルを作成する
                    マップドライバ
                    実行ライブラリ
                                  表示・印刷ライブラリ
$ sampleC （実行）
```

(5)　HP-UX 環境でのコンパイル（EUC）

C 言語で記述した帳票出力用 AP のソースプログラム（xxx.c）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxpdrv：マップドライバ実行ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX の EUC 環境での cc コマンドによるコンパイル例を次の図に示します。

図 5-21　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（帳票印刷）

（帳票印刷，C言語，HP-UXのEUC環境，共用ライブラリ使用の場合）

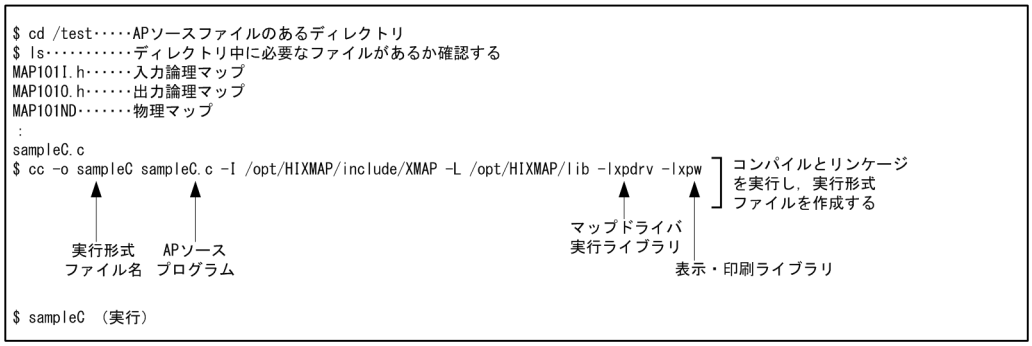
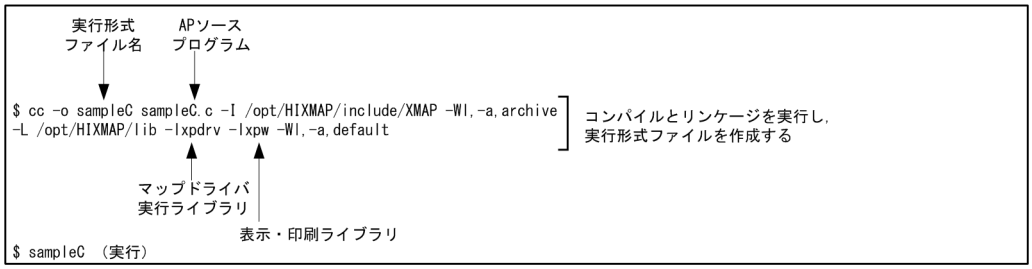


図 5-22　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（帳票印刷）

（帳票印刷，C言語，HP-UXのEUC環境，アーカイブライブラリ使用の場合）



5.4　汎用関数での印刷命令

帳票印刷で使用する Windows 専用の汎用関数について説明します。また，汎用関数を使用して AP で帳票を印刷する方法について説明します。汎用関数は，C，および C++で使用できます。

なお，OLTP サーバ構成の環境では，汎用関数は使用できません。

5.4.1　汎用関数の機能概要

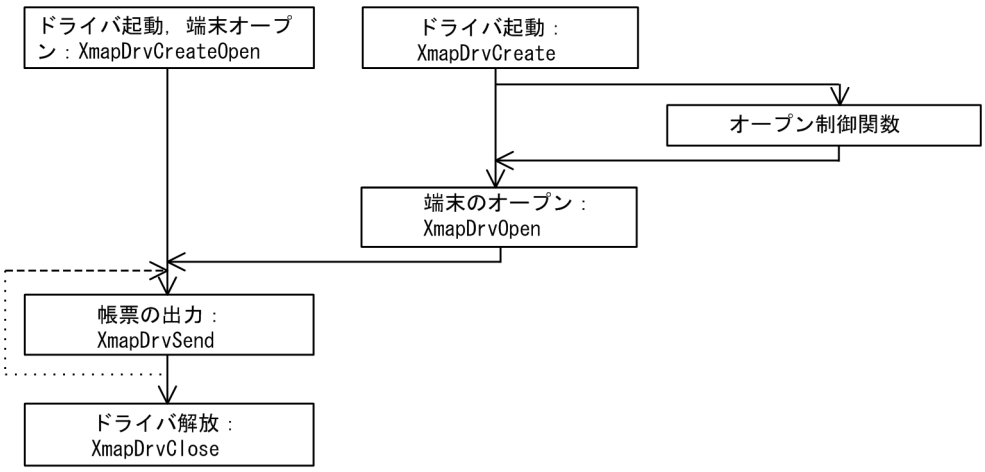
XMAP3 を呼び出すための Windows 専用の汎用関数を次に示します。以降，C 言語ベースの記述で説明します。

表 5-6　Windows 専用の汎用関数

関数	機能概要	備考
XmapDrvCreateOpen	ドライバ起動，仮想端末オープン（プリンタ）	オープン要求関数
XmapDrvClose	仮想端末クローズ，ドライバ解放	クローズ要求関数
XmapDrvSend	帳票の出力	なし
XmapDrvGetError	エラーコードの詳細を取得	
XmapDrvCreate	ドライバ起動	オープン要求関数
XmapDrvOpen	端末オープン	オープン要求関数
XmapDrvSetDataCode	データ有無コードの設定	オープン制御関数

(1)　関数の発行順序

関数の発行順序を次に示します。



注　関数がエラーリターンしたときにXmapDrvGetErrorを発行すると，エラーの詳細を取得できます。
各関数でエラーが発生した場合，自動的に仮想端末をクローズします。

汎用関数については，「13.2.1　帳票印刷命令（汎用関数）」を参照してください。

5.4.2　汎用関数を使用した場合の C 言語での印刷方法

汎用関数を使用した場合の C 言語での印刷方法について説明します。

(1)　DLL の呼び出し

汎用関数では，XMAP3 が提供する DLL を使用します。このため，インタフェース宣言ファイル，およびリンケージライブラリを取り込む必要があります。

- インタフェース宣言ファイルを取り込みます。

  インタフェース宣言ファイルの格納先を次に示します。

  XMAP3 インストールフォルダ¥INCLUDE¥X3MWGD32.H

- リンケージライブラリを取り込みます。

  リンケージライブラリの格納先を次に示します。

  XMAP3 インストールフォルダ¥LIB¥X3MWGD32.LIB

5.5　帳票の FAX 出力

FAXC/SPOOL，または一般の FAX 通信プログラムを使用して，プリンタ出力と同様に FAX に帳票出力ができます。FAXC/SPOOL と連携した FAX 出力をする場合は，帳票出力 AP から XMAP3 の FAX 宛先ファイルへ宛先情報を書き込んで FAX の宛先を動的に変更します。

FAX 出力についてはマニュアル「XMAP3 実行ガイド」を参照してください。なお，FAXC/SPOOL については，FAXC/SPOOL が提供する「使用の手引き.pdf」を参照してください。

6 書式オーバレイでの AP のコーディング方法

この章では，書式オーバレイ用の AP を作成する方法について説明します。

6.1　XMAP3 の書式用 AP で実行する処理

ここでは，XMAP3 の書式用 AP で実行する処理の概要と，コーディング方法について説明します。

なお，XMAP3 の API はマルチスレッドで動作する AP からの実行に対応していません。マルチスレッドで動作する AP では，XMAP3 の API を使わないようにしてください。

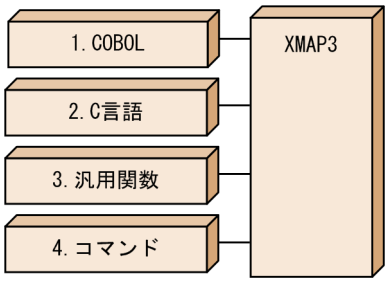
なお，Web 実行環境，および Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合，書式用 AP は利用できません。

6.1.1　XMAP3 での書式の出力

(1)　書式用 AP で使用する API の種類

XMAP3 には，書式用 AP の開発に使用するプログラミング言語ごとに API を提供しています。API の種類を次の図に示します。

図 6–1　書式用 AP で使用する API の種類



1. COBOL の命令文です。レコード名，または一意名の CHARACTER TYPE 句で行単位の表示属性を指定します。WRITE 文で書き出します。Windows/UNIX で共通の文法です。
2. XMAP3 が提供している C 言語用の関数です。jstqlctp 関数を使用して行単位の表示属性を指定します。Windows/UNIX で共通の文法です。
3. XMAP3 が提供している Windows 専用の C 言語または C++用の汎用関数です。XmapFrmSet×××関数を使用して，行単位の表示属性を指定します。
4. XMAP3 が提供している書式オーバレイ印刷用のコマンドです。行データをユーザ責任でテキストファイルに出力して，このコマンドの入力とします。Windows/UNIX で提供コマンドが異なります。

(2)　書式用 AP で使用する API の概要

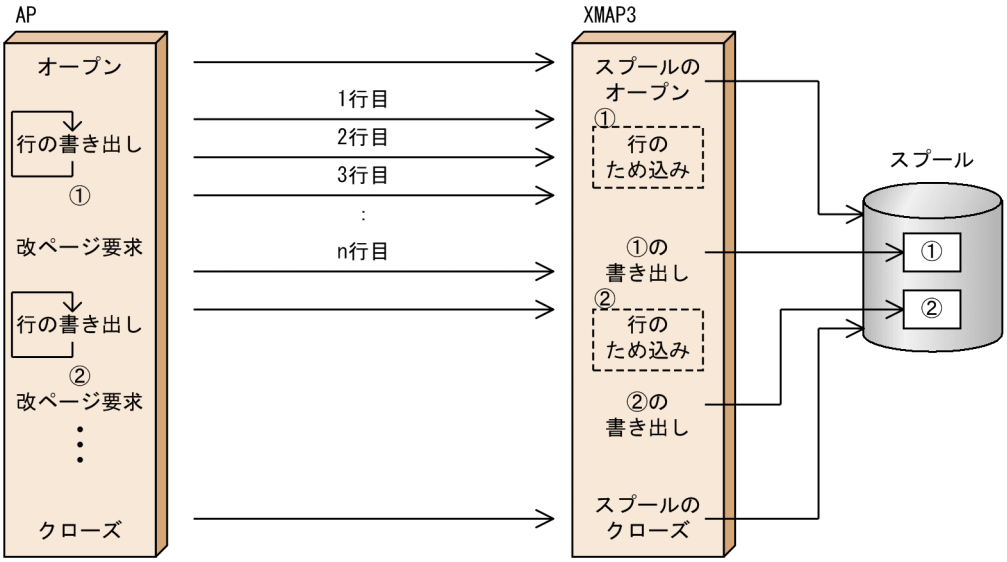
書式用 AP で使用する要求と API の関係を次の表に示します。また，書式用 AP と XMAP3 との処理の概要を次の図に示します。

表 6–1　書式用 AP で使用する要求と API の関係

要求	COBOL の命令文	C 言語	汎用関数
オープン	OPEN 文	jstqlopn 関数	XmapFrmCreateOpen
行の出力	WRITE 文	jstqldat 関数	XmapFrmSetPage
印刷要求	WRITE 文（PAGE 指定）または CLOSE 文	jstqllpt 関数	XmapFrmSetLine

要求	COBOL の命令文	C 言語	汎用関数
クローズ	CLOSE 文	jstqlcls 関数	XmapFrmClose

図 6-2　書式用 AP と XMAP3 との処理の概要



注

行データを出力する場合，行データおよびページデータの上限値によってデータを出力できないことがあります。

ページデータの上限値を超えるデータが AP から指定された場合，XMAP3 が改ページを行い，上限値を超えるデータは次のページに出力されます。

上限値については「付録 B　書式オーバレイの 1 ページデータ量の制限」を参照してください。

(3)　使用時の注意事項

書式オーバレイ使用時の注意事項について説明します。書式オーバレイを使用するときには，次の点に注意してください。

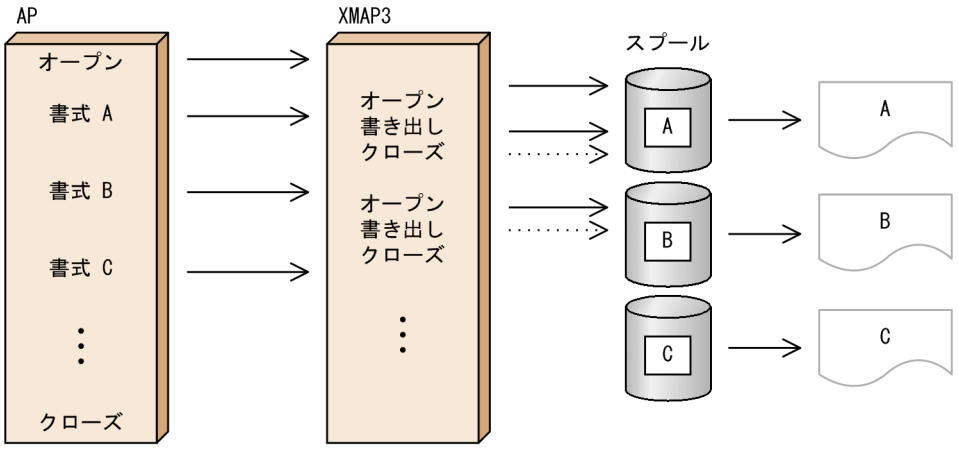
- 書式を印刷する場合は，ページプリンタ用として定義したサービス名を指定してください。
- 書式オーバレイなしの，行データだけの出力はできません。
- 書式オーバレイで出力する行データ中に，$(00)_{16}$～$(1F)_{16}$，$(7F)_{16}$ のデータは使用できません。
- 書式オーバレイで使用する行データの上限値は，16,383 バイトです。
- 書式定義ファイルからの書式ファイルと行制御ファイルの生成は，ドローの保存時または終了時だけできます。マップ生成からは生成されません。
- AP 実行環境で，書式ファイル（拡張子「.fmp」）と行制御ファイル（拡張子「.pci」）は，同じフォルダに格納します。

6.1.2　スプールの設定と印刷ページの関係

1 ページを 1 文書でスプール登録する場合と，複数ページを 1 文書でスプール登録する場合について説明します。

(1)　1 ページを 1 文書でスプール登録する

AP と XMAP3 の間で，AP が 1 ページ出力したら，1 ページを一つの文書としてスプール登録および印刷するようにします。



APからの書式の印刷要求でXMAP3がオープンとクローズ処理をします。

- Windows の場合

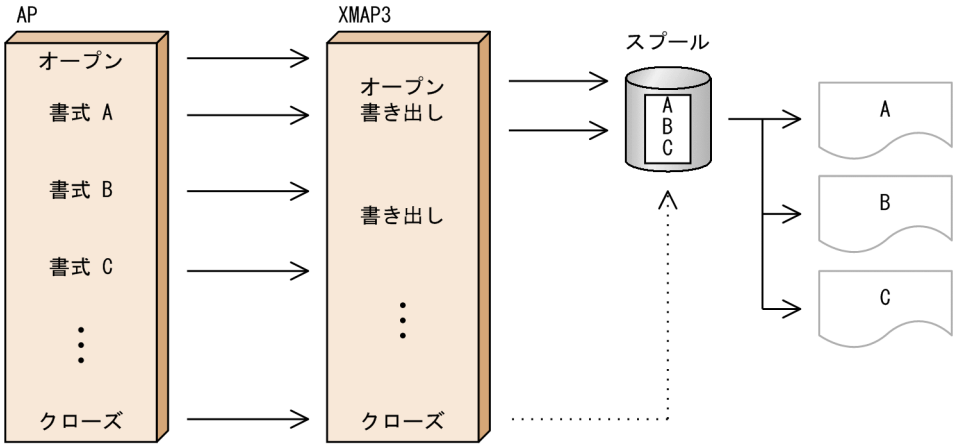
  スプールへの出力単位は，表示・印刷セットアップの「プリンタタブ」の「スプール書き出し単位」の指定に従います。「スプール書き出し単位」は，「1 ページ毎」を選択します。詳細については，マニュアル「XMAP3 実行ガイド」を参照してください。

- UNIX の場合

  1 ページごとに出力するようにするには，XMAP3 サーバを起動する前に環境変数に「XPWCSTMW=OFF」を設定するか，この環境変数を設定しないでください。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

(2)　複数ページを 1 文書でスプール登録する

AP と XMAP3 の間で，AP から複数ページの出力を要求して，AP からのクローズ要求で一つの文書としてスプール登録および印刷するようにします。



APからの最初の書式の印刷要求でXMAP3がオープンし，クローズ命令でクローズ処理をします。

- Windows の場合

スプールへの出力単位は，表示・印刷セットアップの「プリンタタブ」の「スプール書き出し単位」の指定に従います。「スプール書き出し単位」は，「アプリケーション毎」を選択します。詳細については，マニュアル「XMAP3 実行ガイド」を参照してください。

- UNIX の場合

  複数ページを出力するようにするには，XMAP3 サーバを起動する前に，環境変数に「XPWCSTMW=ON」を設定します。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

! 注意事項

PDF ファイル出力の場合，「アプリケーション毎」を選択してください。1 枚目に指定した印刷ドキュメント名の PDF ファイルにすべての帳票が格納されます。「1 ページ毎」を選択して，印刷ドキュメント名が同じ帳票を連続出力すると，出力のたびに PDF ファイルが上書きされるので，クローズ直前に出力した PDF ファイルだけが登録されます。

6.1.3　帳票出力時の XMAP3 と AP の関係

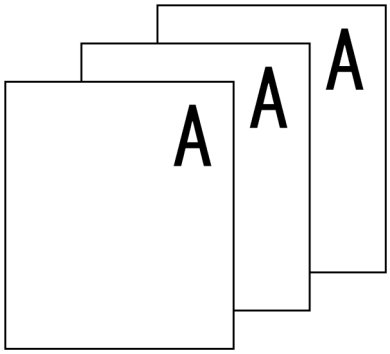
XMAP3 から帳票を印刷する場合の，印刷形態と AP のコーディング内容の関係を示します。なお，ここではプログラミング言語として COBOL が使用されていることを仮定します。

(1)　同じ内容のものを複数枚印刷（コピー印刷）

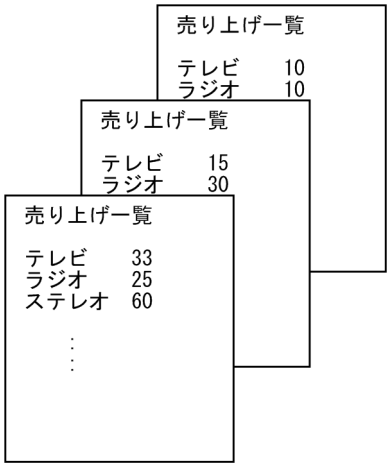
同じ内容のものを複数枚，印刷する場合です。例えば，3 枚の帳票を印刷するときは，3 回同じ印刷を繰り返します。スプールには，同じものが 3 部格納されます。なお，書式名は，1 回だけ指定します。

注

PDF ファイルへ出力する場合，コピー印刷はできません。



(2)　同じ帳票レイアウトで中身が異なるものを複数枚印刷

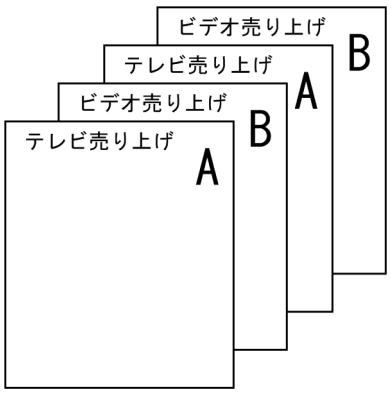


書式オーバレイ名は同じで，データを変えて３回改ページします。

コーディング例

```
MOVE  'OVL01'  TO  書式名      ・・・  書式オーバレイ名指定
WRITE  'テレビ  33'            ・・・  １ページ目
     :
WRITE  'テレビ  15'            ・・・  ２ページ目
     :
WRITE  'テレビ  10'            ・・・  ３ページ目
     :
```

(3)　異なる帳票を交互に印刷



コーディング例

```
MOVE  'OVLAA'  TO  書式名        ・・・  書式名Ａの設定
WRITE                            ・・・  Ａの印刷
     :
MOVE  'OVLBB'  TO  書式名        ・・・  書式名Ｂに切り替え
WRITE                            ・・・  Ｂの印刷
     :
MOVE  'OVLAA'  TO  書式名        ・・・  書式名Ａに戻す
WRITE
     :
```

(4)　異なるプリンタに帳票を印刷

書式オーバレイで一つの AP から異なるプリンタに印刷する場合には，印刷するプリンタごとに印刷サービスをオープンし，印刷要求をします。

ただし，スタンドアロン印刷の場合には，複数の印刷サービスを同時にオープンすることはできません。そのため，プリンタを切り替えるときには，すでにオープン済みの印刷サービスを一度クローズしたあと，出力したいプリンタに対応する印刷サービスをオープンしてから印刷する必要があります。

6.2　COBOL での印刷命令

ここでは，COBOL で書式オーバレイ用の AP を作成する方法について説明します。

XMAP3 Developer で作成した書式イメージ（拡張子「.fmp」）ファイル，行制御データ（拡張子「.pci」）ファイルは，バイナリ形式でファイル転送する必要があります。

ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

XMAP3 Developer で作成した書式イメージファイル，行制御データファイルを，UNIX マシンへ転送したあと，mapfchk コマンドを使って，ファイルの形式チェックを実行します。

mapfchk コマンドは，次のファイルを対象にファイルの形式チェックを実行します。

- 拡張子が「.fmp」の書式イメージファイル
- 拡張子が「.pci」の行制御データファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

mapfchk コマンドについては，「15.2.1　mapfchk コマンド」を参照してください。

6.2.1　プリンタ出力用ファイルの定義

プリンタ出力するファイルは，INPUT-OUTPUT SECTION で順編成を宣言します。

また，XMAP3 で使用する**印刷サービス名**を指定します。スタンドアロン環境で，プリンタマネージャの標準プリンタを使用する場合は，XMAP3 実行環境の表示・印刷セットアップの「プリンタ」タブの印刷モードに，ページプリンタ用として定義したサービス名を設定してください。C/S システム環境の場合は，XMAP3 実行環境の C/S セットアップの印刷サービス名と同じ名称を指定します。印刷サービス名は，COBOL の環境変数「**CBLX_外部装置名=印刷サービス名**」で指定します。AP 実行時，印刷サービス名は，SELECT 句で宣言したプリンタ出力用ファイルの ASSIGN 句で指定した外部装置名に割り当てられます。

例

```
INPUT-OUTPUT SECTION.
  :
FILE-CONTROL.
  SELECT プリンタ  ASSIGN TO 外部装置名.…… 1 .
  :
DATA DIVISION.
FILE SECTION.
FD プリンタ IS GLOBAL
  RECORDING MODE IS F
  LABEL RECORD IS OMITTED
  DATA RECORD IS 行データ.
01 行データ PIC X(71).
```

例の説明

1. プリンタ出力するファイルの宣言

   プリンタ出力するファイルを順編成として宣言します。

   ただし，次に示す指定をした場合，印刷サービス名が物理ファイル名として扱われるため，注意してください。

- ASSIGN 句で定数，またはデータ名を指定した場合

- 外部装置名を COBOL の環境変数「CBLX_外部装置名」で指定した場合

6.2.2　印刷する書式の設定

印刷する書式の名称（書式オーバレイ名）を指定します。書式を設定する場合，I-O-CONTROL に，APPLY　FORMS-OVERLAY 句で出力する書式オーバレイ名を格納するエリアを指定します。このエリアに格納した書式オーバレイを重ねて印刷できます。

例

```
I-O-CONTROL.
APPLY FORMS-OVERLAY TO 書式名 ON プリンタ.              ……1
  :
WORKING-STORAGE SECTION.
01 書式名 PIC X(8) VALUE '書式オーバレイ名'.            ……2
```

例の説明

1. 書式名格納エリア

   書式名のエリアに格納された書式オーバレイを使用して，プリンタに指定したプリンタ出力用ファイルに対して印刷します。

2. 書式オーバレイ名の格納

   書式名格納エリアに書式オーバレイ名を格納します。書式オーバレイ名には，マップ名に ID「6G（マップ名が 6 文字のとき），または F（マップ名が 7 文字のとき）」を付けた名称を指定します。印刷する書式を変更する場合は，書式名格納エリアの内容を書き換えます。書式名はプログラム内で指定するほかに，環境変数「XMAP3_FMP」を使用する方法もあります。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

6.2.3　書式の印刷

(1)　行データの帳票印刷

AP から書式印刷をする場合，印刷用ファイルへ行データを出力するために PROCEDURE DIVISION に処理を記載します。詳細については，「11.3.1　行データの帳票印刷（COBOL）」を参照してください。

(2)　印刷時の優先順位

書式オーバレイ印刷する AP を実行する場合，AP での設定および環境変数で指定できる項目があります。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

6.2.4　COBOL でのコンパイル（Windows）

(1)　COBOL 開発マネージャを使用したコンパイルと実行のポイント

COBOL 開発マネージャを使用したコンパイルと実行のポイントについて説明します。

(a)　COBOL 開発マネージャの概要

COBOL 開発マネージャは，COBOL から提供されている COBOL プログラムの統合的な開発環境です。COBOL で AP を開発するときに必要な COBOL ソースや登録集原文などの資産をその依存関係に従って管理し※，コンパイルなどの作業を自動化します。

注※

ただし，書式オーバレイの場合，論理マップは生成されないため，マップの資産管理はできません。

COBOL 開発マネージャを使用する場合，次の手順で AP を作成します。

1. プロジェクトの作成
2. ビルド，リビルド

次に，それぞれの手順について説明します。

プロジェクトの作成

COBOL 開発マネージャでは，一つの.exe または.dll ファイルの開発単位を「プロジェクト」と呼んでいます。COBOL 開発マネージャを使用して AP を作成する場合，このプロジェクトを定義する必要があります。

ビルド，リビルド

ソースファイル名を指定し，コンパイルとリンケージをします。

その方法として，タイムスタンプ（ファイル作成／修正日時）の前後関係に基づいてコンパイルとリンケージをする「ビルド」があります。例えば，「実行ファイルの作成／修正日時」より実行ファイルが取り込んでいる「COBOL ソースの作成／修正日時」の方が新しい場合にコンパイルとリンケージが実行されます。

また，タイムスタンプには関係なくコンパイルとリンケージをする「リビルド」もあります。

(b)　コンパイル時のポイント

COBOL を使用したとき，指定が必要なコンパイラオプションを次に示します。

- -XMAP,LinePrint：書式オーバレイや印刷制御機能を利用できるようにするオプション

(c)　リンケージ時のポイント

インポートライブラリ／ユーザ指定ライブラリに，次に示すファイルを指定する必要があります。

XMAP3 インストールフォルダ¥LIB¥x3klib32.lib

(d)　実行時のポイント

書式データファイルおよび行制御データファイルの格納

AP を実行する前に，書式データファイル（.fmp）と行制御データファイル（.pci）を実行形式ファイルと同じフォルダに格納してください。

また，XMAP3 実行環境の表示・印刷セットアップでマップパスを指定する方法もあります。マップパスの指定については，マニュアル「XMAP3 実行ガイド」を参照してください。

COBOL での環境変数の指定

XMAP3 で使用するプリンタの識別名（**印刷サービス名**）を，COBOL の環境変数によって指定します。指定形式を次に示します。

形式

```
CBLX_外部装置名　印刷サービス名　　…… 1：外部装置名，2：印刷サービス名
```

形式の説明

1. 外部装置名

   COBOL ソース中の SELECT 句で宣言したプリンタ出力用ファイルの ASSIGN 句で指定した名称です。

2. 印刷サービス名

スタンドアロン環境でプリンタマネージャの標準プリンタを使用する場合は XMAP3 実行環境の表示・印刷セットアップの「プリンタ」タブの印刷モードに，ページプリンタ用として定義したサービス名を指定します。C/S 環境では，C/S セットアップの印刷サービス名と同じ名称を指定します。

6.2.5　COBOL でのコンパイル（UNIX）

COBOL2002 で作成した，書式オーバレイ印刷用 AP のコンパイル方法を説明します。

使用する UNIX およびターゲットの文字コード（シフト JIS，EUC）に応じて手順が異なります。

(1)　AIX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.cbl）を AIX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -XMAP,LinePrint：順編成ファイルを XP プリンタに出力（書式オーバレイおよび行データの印刷制御付き）（必須）
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用するライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 6-3　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

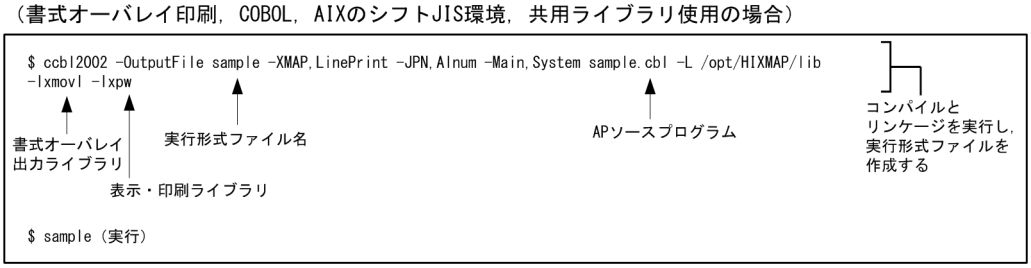
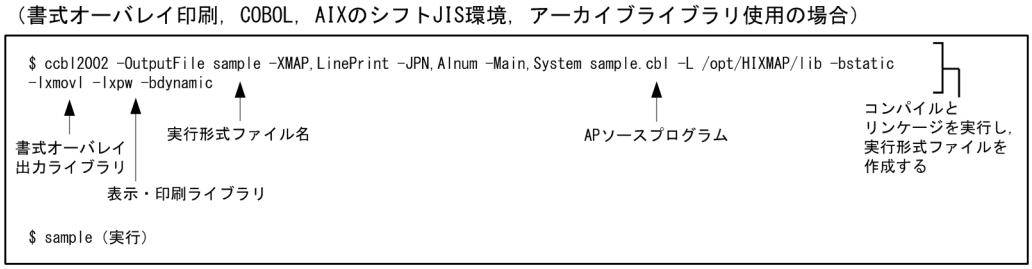


図 6-4　AIX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）



(2)　AIX 環境でのコンパイル（EUC）

COBOL2002 で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.cbl）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

! 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -XMAP,LinePrint：順編成ファイルを XP プリンタに出力（書式オーバレイおよび行データの印刷制御付き）（必須）
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用するライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 6-5　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

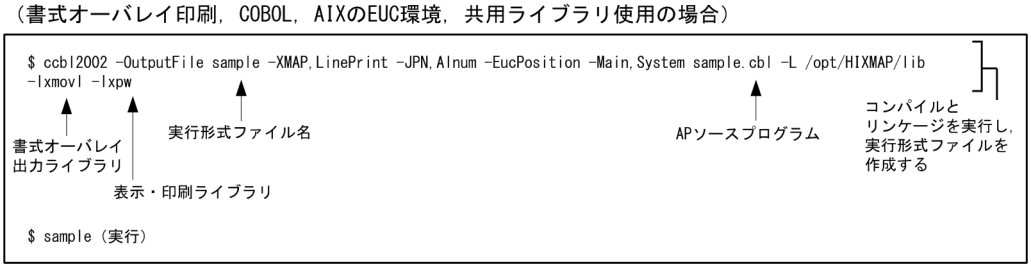
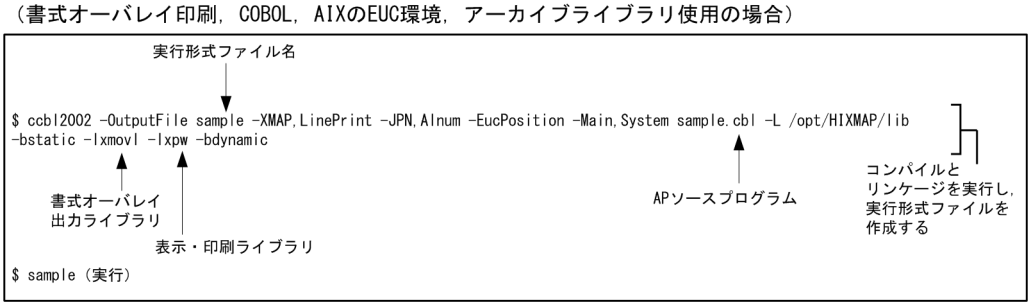


図 6-6　AIX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）



(3)　HP-UX 環境でのコンパイル（シフト JIS）

COBOL2002 で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.cbl）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

(a)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -XMAP,LinePrint：順編成ファイルを XP プリンタに出力（書式オーバレイおよび行データの印刷制御付き）（必須）
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用するライブラリの検索パスを指定する（必須）
- -Wl,-a,default ：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive ：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX のシフト JIS 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 6-7　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

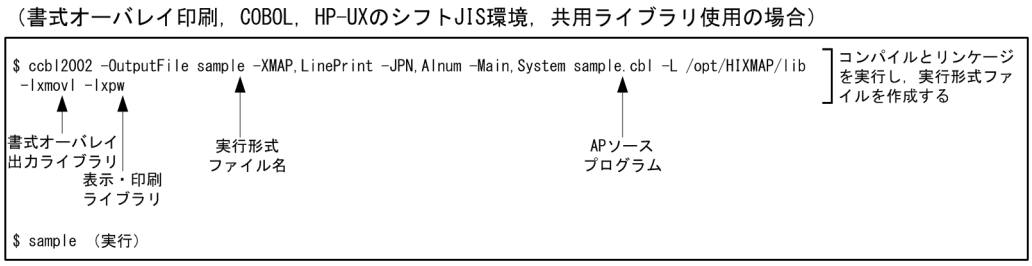
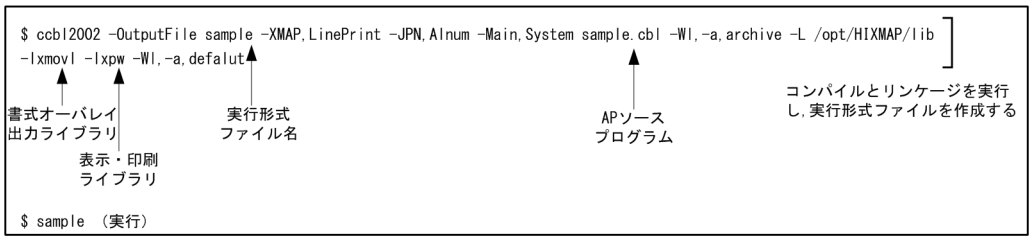


図 6-8　HP-UX のシフト JIS 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）



(4)　HP-UX 環境でのコンパイル（EUC）

COBOL2002 で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.cbl）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，ccbl2002 コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

COBOL AP で使用する論理マップ内のデータ名に半角カタカナを指定した場合，コンパイル時にオプションの指定が必要となることがあります（COBOL コンパイル時に半角カタカナを 2 バイトとして扱うため，72 カラムの制限でエラーとなる場合があります）。

(a)　オプション

使用するコンパイルオプションを次に示します。

- -OutputFile：リンケージ時に実行形式ファイル名を指定する場合に指定する
- -XMAP,LinePrint：順編成ファイルを XP プリンタに出力（書式オーバレイおよび行データの印刷制御付き）（必須）
- -JPN,Alnum：日本語項目などを英数字項目として扱う場合に指定する
- -EucPosition：半角カタカナ使用時に 72 カラム制限を抑止する
- -Main,System モジュール名：メインモジュールのコンパイルの場合に指定する。複数のモジュールをコンパイルする場合は，メインモジュール名の次にサブモジュール名を指定する
- -L ライブラリパス：使用するライブラリの検索パスを指定する（必須）

- -Wl,-a,default ：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive ：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX の EUC 環境での ccbl2002 コマンドによるコンパイル例を次の図に示します。

図 6-9　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

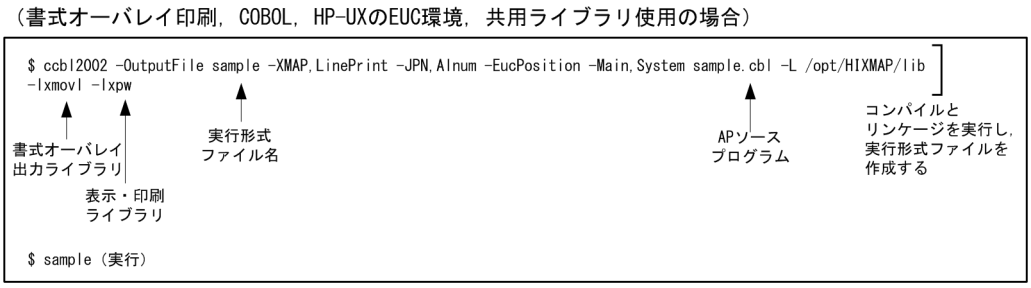


図 6-10　HP-UX の EUC 環境での ccbl2002 コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）

6.3　C 言語での印刷命令

ここでは，C 言語で書式オーバレイの AP を作成する方法について説明します。

XMAP3 Developer で作成した書式イメージ（拡張子「.fmp」）ファイル，行制御データ（拡張子「.pci」）ファイルは，バイナリ形式でファイル転送する必要があります。

ファイルの拡張子はそのままでファイル転送してください（UNIX での拡張子は英小文字に変更する必要があります）。

XMAP3 Developer で作成した書式イメージファイル，行制御データファイルを，UNIX マシンへ転送したあと，mapfchk コマンドを使って，ファイルの形式チェックを実行します。

mapfchk コマンドは，次のファイルを対象にファイルの形式チェックを実行します。

- 拡張子が「.fmp」の書式イメージファイル
- 拡張子が「.pci」の行制御データファイル

なお，形式チェックでは実行環境で参照するすべての情報はチェックしませんので，実行環境でエラーとなる場合があります。

mapfchk コマンドについては，「15.2.1　mapfchk コマンド」を参照してください。

6.3.1　印刷する書式の設定

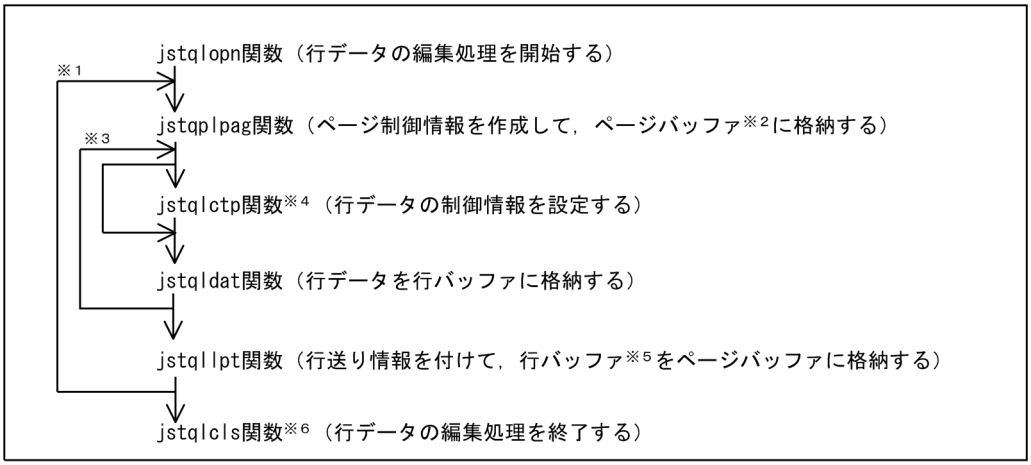
印刷する書式の名称（書式オーバレイ名）を指定します。書式の名称は，jstqlpag 関数の＊ovlname で指定します。jstqlpag 関数については，「12.3.1　行データの帳票印刷（C 言語）」を参照してください。行データは，この関数で指定した書式に重ねて印刷されます。

6.3.2　書式の印刷

(1)　行データの帳票印刷

AP から書式印刷をする場合，行データを出力するために，次の図に示す関数を発行します。関数の機能と発行順序を示します。

図 6-11　関数の機能と発行順序



注※1　行データを1ページ単位で処理するたびに繰り返します。帳票の枚数分繰り返します。
注※2　ページバッファは，1ページ分の行データを格納するためにXMAP3が使用する領域です。
注※3　行データを1行単位で処理するたびに繰り返します。
注※4　行データの制御情報を変更するときだけに発行します。
注※5　行バッファは，1行分の印刷データを格納するためにXMAP3が使用する領域です。
注※6　最後に1回だけ発行します。ただし，各関数がリターンコード=8で異常終了した場合は内部で終了処理をします。この場合，jstqlcls関数は発行しません。

関数については，「12.3.1　行データの帳票印刷（C 言語)」を参照してください。

(2)　印刷時の優先順位

書式オーバレイ印刷する AP を実行する場合，AP での設定および環境変数で指定できる項目があります。環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

6.3.3　C 言語でのコンパイル

C 言語で記述した，書式オーバレイ印刷用 AP のコンパイル方法を OS ごとに説明します。

(1)　Windows 環境でのコンパイル

C 言語のコンパイラを使用してコンパイルしてください。ライブラリには,次に示すファイルを指定する必要があります。

XMAP3 インストールフォルダ¥Lib¥X3KLIB32.LIB

(2)　AIX 環境でのコンパイル（シフト JIS)

C 言語で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.c）を AIX シフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）

- -bstatic：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX のシフト JIS 環境での xlc コマンドによる書式オーバレイ印刷用 AP のコンパイル例を次の図に示します。

図 6-12　AIX シフト JIS 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

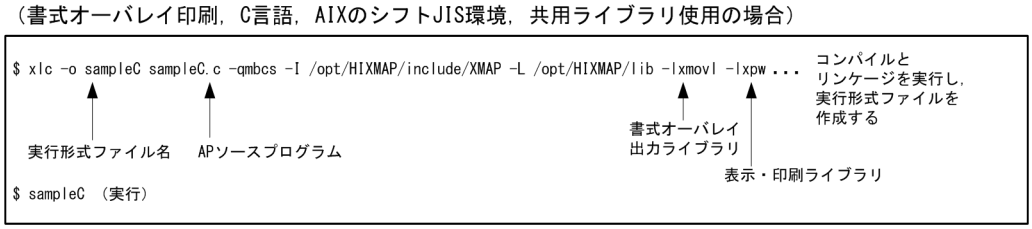
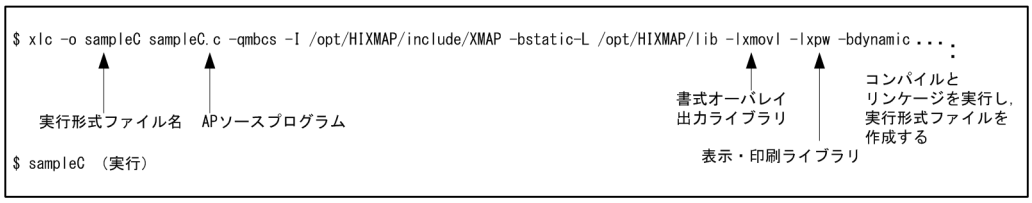


図 6-13　AIX シフト JIS 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）



(3)　AIX 環境でのコンパイル（EUC）

C 言語で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.c）を AIX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，xlc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

AIX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -bdynamic：共用ライブラリを使用する場合に指定する（デフォルト）
- -bstatic：アーカイブライブラリを使用する場合に指定する

- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

AIX の EUC 環境での xlc コマンドによる書式オーバレイ印刷用 AP のコンパイル例を次の図に示します。

図 6-14　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

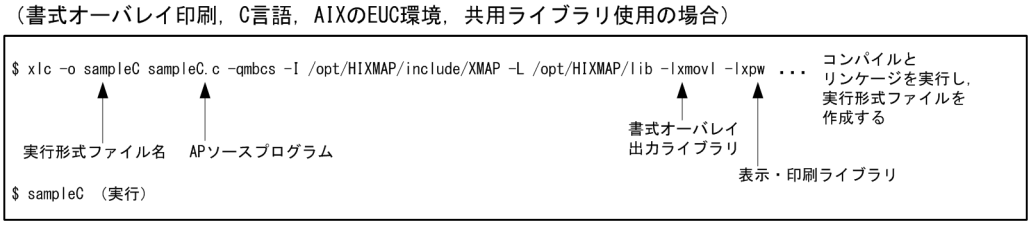
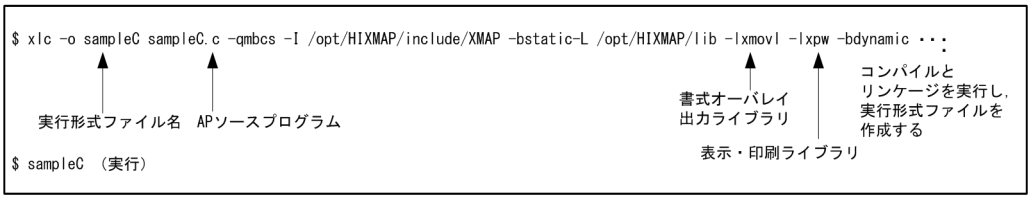


図 6-15　AIX の EUC 環境での xlc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）



(4)　HP-UX 環境でのコンパイル（シフト JIS）

C 言語で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.c）を HP-UX のシフト JIS 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX のシフト JIS 環境での cc コマンドによるコンパイル例を次の図に示します。

図 6-16　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

（書式オーバレイ印刷，C言語，HP-UXのシフトJIS環境，共用ライブラリ使用の場合）

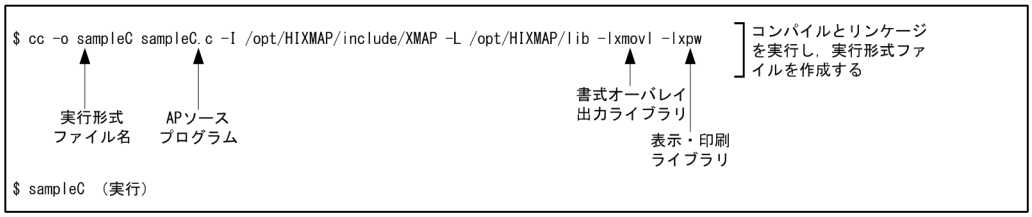
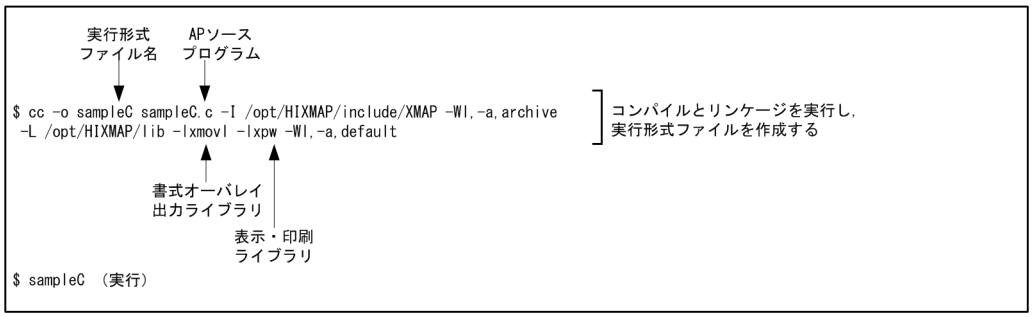


図 6-17　HP-UX のシフト JIS 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）

（書式オーバレイ印刷，C言語，HP-UXのシフトJIS環境，アーカイブライブラリ使用の場合）



(5)　HP-UX 環境でのコンパイル（EUC）

C 言語で記述した書式オーバレイ印刷用 AP のソースプログラム（xxx.c）を HP-UX の EUC 環境でコンパイルする方法について説明します。使用するコマンドは，cc コマンドです。

なお，コンパイル時に使用する論理マップおよび動的変更テーブルは文字コードがシフト JIS のため，EUC へのコード変換が必要です。

！ 注意事項

HP-UX の EUC 環境での半角カタカナは，シフト JIS とは異なり 1 バイト文字ではなく，2 バイト文字として扱われます。このため，半角カタカナを使用した項目の論理項目長は，レイアウト上の長さの 2 倍がデフォルトになります。

(a)　オプション

コンパイル時に使用するオプションを次に示します。

- -I：使用ヘッダファイルの検索パスを指定する
- -L：使用ライブラリの検索パスを指定する（必須）
- -Wl,-a,default：共用ライブラリを使用する場合に指定する（デフォルト）
- -Wl,-a,archive：アーカイブライブラリを使用する場合に指定する
- -lxmovl：書式オーバレイ出力ライブラリを指定する（必須）
- -lxpw：表示・印刷ライブラリを指定する（必須）

(b)　コンパイル例

HP-UX の EUC 環境での cc コマンドによるコンパイル例を次の図に示します。

図 6-18　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，共用ライブラリ使用の場合）（書式オーバレイ印刷）

（書式オーバレイ印刷，C言語，HP-UXのEUC環境，共用ライブラリ使用の場合）

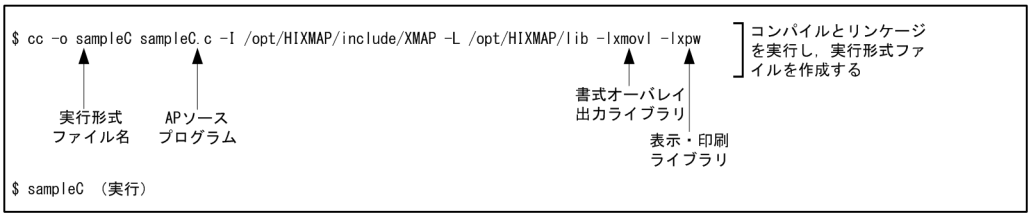
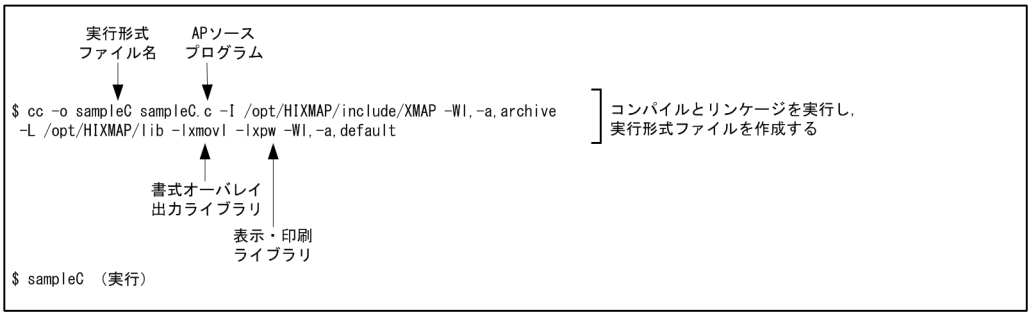


図 6-19　HP-UX の EUC 環境での cc コマンドによる実行例（単一ファイル，アーカイブライブラリ使用の場合）（書式オーバレイ印刷）

（書式オーバレイ印刷，C言語，HP-UXのEUC環境，アーカイブライブラリ使用の場合）



6.4　汎用関数での書式の印刷命令

書式の印刷で使用する Windows 専用の汎用関数について説明します。また，汎用関数を使用して AP で書式を印刷する方法について説明します。汎用関数は，C，および C++で使用できます。

なお，OLTP サーバ構成の環境では，汎用関数は使用できません。

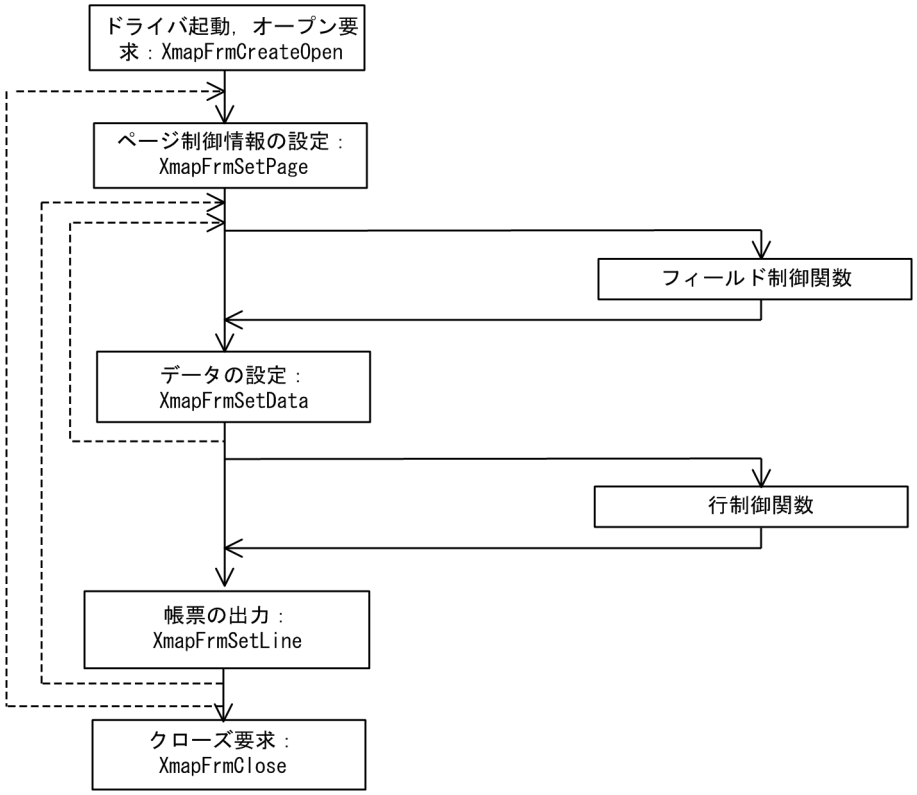
6.4.1　汎用関数の機能概要

XMAP3 を呼び出すための汎用関数を次に示します。以降，C 言語ベースの記述で説明します。

表 6-2　汎用関数一覧

関数	機能概要	備考
XmapFrmCreateOpen	ドライバ起動，サービスオープン	オープン要求関数
XmapFrmClose	ドライバ解放，サービスクローズ	クローズ要求関数
XmapFrmSetPage	ページ制御情報の設定	なし
XmapFrmSetData	行データをバッファリング	
XmapFrmSetLine	バッファデータを出力	
XmapFrmGetError	エラーコードの詳細を取得	
XmapFrmSetPoint	文字サイズの設定	フィールド制御関数
XmapFrmSetInterval	字間値の設定	
XmapFrmSetFont	書体の設定	
XmapFrmSetWidth	平体の設定	
XmapFrmSetNewLine	改行数の設定	行制御関数
XmapFrmSetChannel	チャネルスキップの設定	

(1)　関数の発行順序



注　関数がエラーリターンしたときにXmapFrmGetErrorを発行すると，エラーの詳細を取得できます。
XmapFrmSetPoint/Interval/Font/Widthは，XmapFrmSetDataの前に発行します。
XmapFrmSetNewLine/Channelは，XmapFrmSetLineの前に発行します。
エラーが発生した場合，自動的にクローズ処理が実行されます。

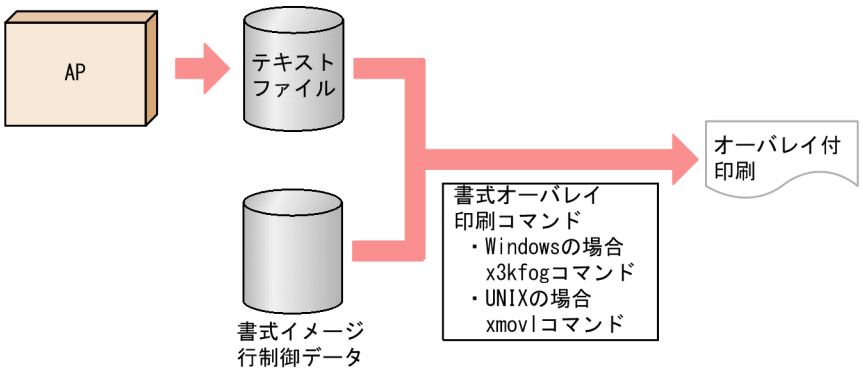
汎用関数については，「13.3.1　書式印刷命令（汎用関数）」を参照してください。

6.5　書式オーバレイの簡易印刷

指定された書式をテキストファイル中の行データと重ねて印刷する書式オーバレイ印刷コマンドの実行方法について説明します。このコマンドを利用することで，テキストファイルがあれば簡単に書式付き帳票を印刷できます。また，書式だけを印刷することもできます。

(1)　書式オーバレイ印刷コマンド

(a)　処理の流れ



(b)　形式の説明

書式オーバレイ印刷コマンドは，次に示す方法のうちどれかの方法で実行します。

- スタートメニューの［ファイル名を指定して実行］
- コマンドプロンプト
- バッチファイル（拡張子：.bat）
- AP からの実行

書式オーバレイ印刷コマンドの形式を次に示します。

Windows の場合

形式

```
x3kfog 〔-s 印刷サービス名〕〔-f 書式名〕 テキストファイル　…書式オーバレイ印刷
x3kfog 〔-s 印刷サービス名〕-T 書式名　…書式だけの印刷
```

UNIX の場合

形式

```
xmovl〔-s 印刷サービス名〕〔-f 書式名〕テキストファイル　…書式オーバレイ印刷
xmovl〔-s 印刷サービス名〕-T 書式名　…書式だけの印刷
```

-s　印刷サービス名

出力先の印刷サービス名を指定します。印刷サービス名を省略したときは，環境変数「XMAP3_PSNAME」で指定した名称が仮定されます。

サービス名ファイルについては，マニュアル「XMAP3 実行ガイド」を参照してください。

-f　書式名

出力する書式名を指定します。書式名は，拡張子（.fmp）を除いた 8 文字以内の文字列で指定します。

書式名を省略したときは，環境変数「XMAP3_FMP」で指定した名称が仮定されます。

テキストファイル

テキストファイルの名称を指定します。テキストファイルは，ASCII コードおよびシフト JIS コードから構成されるファイルで，文字，数字，記号だけのファイルとします。制御コードは，改行，および改ページコードだけで，テキストファイルの先頭から行データとして印刷します。なお，有効となるのは，一行につき 1,024 文字までです。1,025 文字以降は，入力しても無効になります。

-T　書式名

印刷する書式名を指定します。書式名は，拡張子（.fmp）を除いた 8 文字以内の文字列で指定します。書式だけを印刷する場合は，必ず書式名を指定します。

コマンド終了コード

書式オーバレイ印刷コマンドの終了コードを次に示します。

0：正常終了

1：異常終了

(c)　注意事項

- Windows で書式オーバレイをシリアルプリンタに印刷する場合，プリンタの設定が必要になります。表示・印刷セットアップのプリンタの種類で「ページ帳票／書式印刷」のプリンタを選んでください。
- 書式だけを印刷するときの実行環境では，書式イメージファイルと行制御データファイルを同じフォルダに格納してください。
- 書式オーバレイなしの行データだけの印刷はできません。ただし，何も定義していない空の書式を作成すると，行データだけの印刷ができます。
- 印刷時には，AP 環境ファイル（X3MWDRV／XMAPdrv）の mapPath で指定されているフォルダが参照されます。mapPath で指定されているフォルダに指定した書式ファイルがない場合は，実行時フォルダが参照されます。

  Windows の場合，AP 環境ファイルは表示・印刷セットアップで設定してください。
- 指定したテキストファイルの内容は，1 行ごとに行データとして印刷されます。指定できる行データには上限があります。1,024 バイト以下にしてください。指定できる 1 ページデータ量の上限値を算出する概算式については，「付録 B　書式オーバレイの 1 ページデータ量の制限」を参照してください。なお，概算式での「項目データの数」は「1」となります。
- UNIX の場合，テキストファイルに指定する文字コードは，LANG 環境変数に対応する文字コードにしてください。
- UNIX の場合，書式だけの印刷，または AP 実行環境で書式オーバレイをシリアルプリンタに印刷することはできません。
- 書式名を省略し，かつ環境変数「XMAP3_FMP」を設定していない場合，書式属性には次の表に示す値が設定されます。

書式属性		設定値
用紙種別		A4 横
行データのます目	行の間隔	6LPI
	文字間隔	5 ポイント
	文字サイズ	9 ポイント
	書体	標準
マージン	上マージン	8.3mm

書式属性		設定値
	左マージン	8.3mm
レイアウト領域サイズ	ます（行）	46 行
	ます（列）	255 列

6.6　書式オーバレイの FAX 出力

FAXC/SPOOL または一般の FAX 通信プログラムを使用して，プリンタ出力と同様に FAX に書式オーバレイ帳票出力ができます。FAXC/SPOOL と連携した FAX 出力をする場合は，書式オーバレイ帳票出力 AP から XMAP3 の FAX 宛先ファイルへ宛先情報を書き込んで FAX の宛先を動的に変更します。

FAX 出力についてはマニュアル「XMAP3 開発ガイド」を参照してください。なお，FAXC/SPOOL については，FAXC/SPOOL が提供する「使用の手引き.pdf」を参照してください。

7 帳票のコーディング例

この章では，帳票と書式の定義および AP との関連について説明します。

7.1　帳票定義と AP のコーディング

帳票の帳票属性と AP の関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：COBOL
- マップ名の文字数：6 文字
- マップ名：MAP001
- 実行環境：対話型

ドロー，およびドローセットアップについては，マニュアル「XMAP3 開発ガイド」を参照してください。

7.1.1　印刷枚数の指定を変更する

ページプリンタを使用する場合，印刷枚数を AP で指定できます。ここでは，印刷枚数を AP で指定する場合の論理マップおよび AP の関連について説明します。なお，PDF ファイル出力の場合，指定した印刷枚数は無効となります。

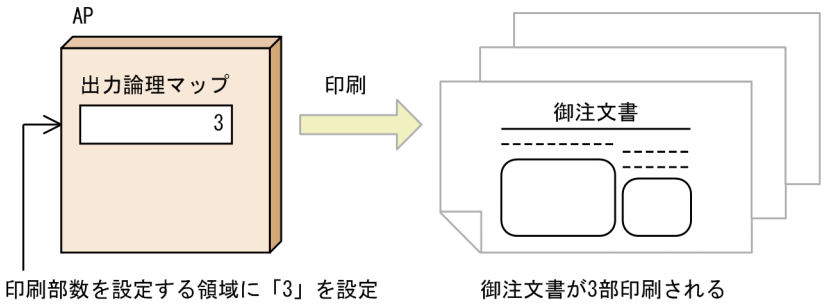
印刷枚数は次のどちらかの方法で指定します。

- 帳票属性ダイアログ
  各マップの帳票属性ダイアログの「印刷部数」で指定します。
- AP
  各マップの帳票属性ダイアログの「印刷部数を AP で変更する」を選択して，動的変更を利用して AP で指定します。

帳票属性ダイアログの「印刷部数」と「印刷部数を AP で変更する」の両方を選択した場合，AP での指定が優先されます。

AP で印刷枚数を指定して帳票をコピー印刷する例を次の図に示します。

図 7-1　AP で印刷枚数を指定して帳票をコピー印刷する例



AP で印刷枚数の数値を代入します。代入した数値分の帳票がコピー印刷されます。

印刷枚数を指定して帳票をコピー印刷するコーディング例を次に示します。

図 7-2　印刷枚数を指定して帳票をコピー印刷するコーディング例

```
MOVE  3  TO  MAP001-COPIESO.          …印刷部数に「3」を代入
```

印刷枚数を AP で変更する場合，制御項目「MAP001-COPIESO」に変更する枚数を代入します。

7.1.2　印刷ドキュメント名を指定する

帳票を印刷するときに，印刷ドキュメント名を AP または環境変数で指定できます。印刷ドキュメント名を AP または環境変数で指定することで，Windows のプリンタスプールに登録される XMAP3 の印刷データの名称や PDF ファイルの名称を動的に変更できます。

印刷ドキュメント名は次のどれかの方法で指定します。

- 帳票属性ダイアログ

  各マップの帳票属性ダイアログの「印刷ドキュメント名」で指定します。指定しない場合，「XMAP3」が仮定されます。

- AP

  各マップの帳票属性ダイアログの「印刷ドキュメント名を AP で変更する」を選択して，動的変更を利用して AP で指定します。

- 環境変数「XMAP3_PRINT_DOCNAME」

  帳票出力前に環境変数「XMAP3_PRINT_DOCNAME」※1 に印刷ドキュメント名を指定します。印刷ドキュメント名は，259 文字※2 以内の文字列で指定します。259 文字を超える文字列が指定された場合，環境変数による印刷ドキュメント名の指定は無効となります。

  注※1

  Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の場合，環境変数はコンフィグファイル（X3MWDR64）に指定します。環境変数「XMAP3_PRINT_DOCNAME」は，コンフィグファイル（X3MWDR64）の X3MWDR_ENV_SEND セクションに記述します。

  注※2

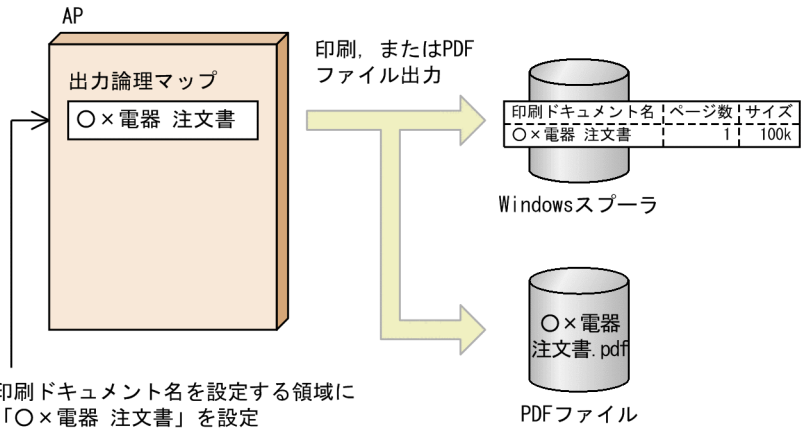
  全角 1 文字は，半角 2 文字分として数えます。

帳票属性ダイアログ，AP，および環境変数「XMAP3_PRINT_DOCNAME」のすべてで印刷ドキュメント名を指定した場合，指定値の優先順位は次のとおりです。

1. 環境変数「XMAP3_PRINT_DOCNAME」の指定
2. AP の指定
3. 帳票属性ダイアログの「印刷ドキュメント名」の指定

印刷ドキュメント名を AP で指定する例を次の図に示します。

図 7-3　印刷ドキュメント名を AP で指定する例



印刷ドキュメント名を AP で指定するコーディング例を次に示します。

図 7-4　印刷ドキュメント名を AP で指定するコーディング例

```
MOVE　‘○×電器 注文書’　TO　MAP001-DOCNAMEO.　　…印刷ドキュメント名を指定
```

印刷ドキュメント名を AP で変更する場合，制御項目「MAP001-DOCNAMEO」に変更する名称を代入します。

7.2　帳票のオブジェクト定義と AP のコーディング

グラフィック帳票の各オブジェクトと AP の関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：COBOL
- マップ名数：6 文字
- マップ名：MAP001
- 実行環境：対話型
- 各オブジェクトのデータ名や制御項目データ名：標準値

ドローおよびドローセットアップについては，マニュアル「XMAP3 開発ガイド」を参照してください。

7.2.1　フィールドの表示を変更する

帳票のフィールドとは，次の四つのオブジェクトです。

- 固定フィールド
- 出力フィールド
- 出力日付フィールド
- 出力時刻フィールド

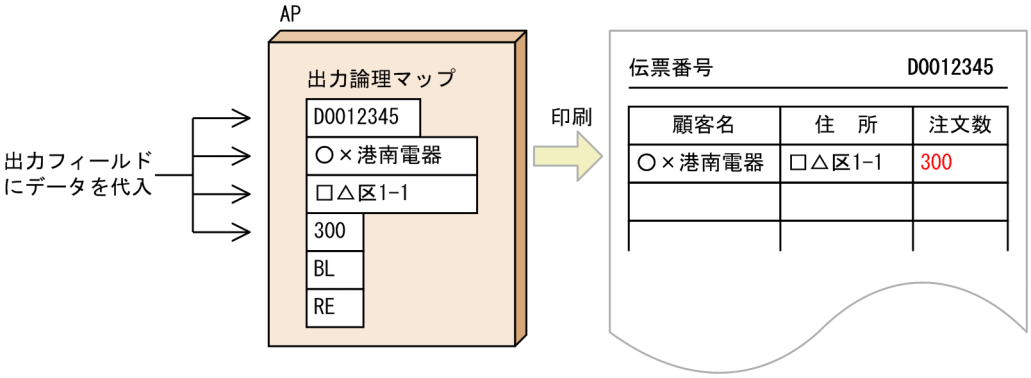
ここでは，AP で出力フィールドを表示する場合のコーディング例について説明します。

出力フィールドでは，動的変更を利用して，AP で文字色，書式などを指定できます。

ドローで出力フィールドのダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，文字色，文字の書体，文字の強調，網掛けなどを AP で指定できます。AP で指定しない場合，出力フィールドボックスダイアログで指定した属性が表示されます。

文字色と文字の書体を変更する場合の出力フィールドの表示例を次の図に示します。

図 7-5　出力フィールドの表示例



出力フィールドのデータとして，「伝票番号」，「顧客名」，「住所」，「注文数」を AP で代入します。「伝票番号」の値である「D0012345」の制御項目に太字を表示する修飾名「BL」を，「注文数」の値である「300」

の制御項目に赤字を表示する修飾名「RE」を代入します。AP を実行すると，「伝票番号」の文字の書体を「太字」で，「注文数」の文字色を赤で出力されます。

文字色と文字の書体を変更するコーディング例を次に示します。

図 7-6　文字色と文字の書体を変更するコーディング例

```
MOVE  'D0012345'       TO  MAP001-FIELD0001-O.      …出力テキストを代入
MOVE  '○×港南電器'    TO  MAP001-FIELD0002-O.      …出力テキストを代入
MOVE  '□△区1-1'       TO  MAP001-FIELD0003-O.      …出力テキストを代入
MOVE  300              TO  MAP001-FIELD0004-O.      …出力テキストを代入
MOVE  XMAP-FIELD-BL    TO  MAP001-FIELD0001-A.      …太字で出力
MOVE  XMAP-FIELD-RE    TO  MAP001-FIELD0004-A.      …文字色を赤色で出力
```

出力フィールドにテキストを出力する場合，出力論理マップのデータの「MAP001-FILED0001-O」，「MAP001-FILED0002-O」などにデータを代入します。また，出力フィールドの表示属性を変更する場合，制御項目の「MAP001-FIELD0001-A」，「MAP001-FIELD0002-A」などに修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，動的変更テーブルの修飾名を制御項目に代入してください。

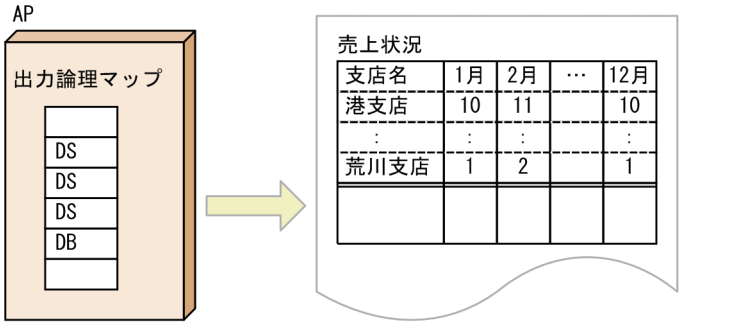
7.2.2　けい線の属性を変更する

グラフィック帳票では，AP でけい線の属性を変更して出力できます。ここでは，AP でけい線の属性を変更するコーディング例について説明します。

ドローでけい線のダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，AP でけい線の種類や太さを指定できます。AP で指定しない場合は，けい線のダイアログで指定した属性で出力されます。

けい線の属性を変更して出力する例を次の図に示します。

図 7-7　けい線の属性を変更して出力する例



・行データのある行だけに，けい線を引く
・行データのない行には，横けい線を引かない

制御項目に破線や二重線を示す修飾名を代入します。印字データがある行間のけい線は「細い破線」で，最終行は「二重線」で印刷されます。

けい線の属性を変更するコーディング例を次に示します。

図 7-8　けい線の属性を変更するコーディング例

```
MOVE  XMAP-LINE-DS  TO  MAP001-FIELD0002-A.      …2本目の線は破線で出力
MOVE  XMAP-LINE-DS  TO  MAP001-FIELD0003-A.      …3本目の線は破線で出力
MOVE  XMAP-LINE-DS  TO  MAP001-FIELD0004-A.      …4本目の線は破線で出力
MOVE  XMAP-LINE-DB  TO  MAP001-FIELD0005-A.      …データの最終行は二重線で出力
```

けい線の表示属性を変更する場合，制御項目の「MAP001-FIELD0002-A」，「MAP001-FIELD0003-A」などに修飾名を代入します。

修飾名はドローセットアップで定義できます。ドローセットアップで定義した修飾名は動的変更テーブルとして保存されます。AP では，動的変更テーブルの修飾名を制御項目に代入してください。

7.2.3　出力バーコードを制御する

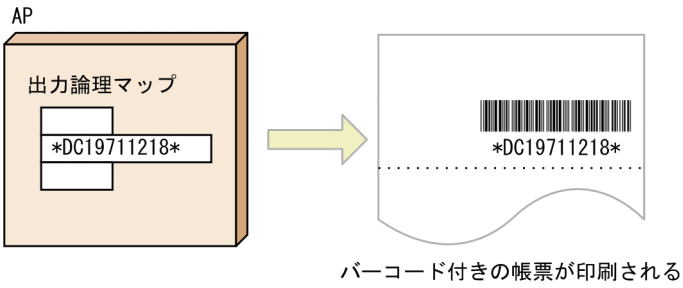
ここでは，グラフィック帳票で出力バーコードを制御する場合のコーディング例について説明します。

出力バーコードのオブジェクトを次の条件で指定します。

- コード種別：CODE39（12 桁）
- データ文字：あり
- AP から設定するデータ：*DC19711218*

AP 実行時のバーコードの印刷例を次の図に示します。

図 7-9　バーコードの出力例



バーコードとバーコードの下に文字が出力されて，印刷されます。

バーコードを出力するコーディング例を次に示します。

図 7-10　バーコードを出力するコーディング例

```
MOVE　‘*DC19711218’　TO　MAP001- BARCODE0001-0.　　…データを代入
```

バーコードにデータを代入する場合，出力論理マップのデータ「MAP001-BARCODE0005-O」にデータを代入します。

7.3　書式オーバレイのコーディング

書式オーバレイでは，1 行（1 レコード）ごとに印刷するデータを出力して，改ページのタイミングで書式を重ねて，1 ページを出力（印刷）します。

AP 作成については，「6.　書式オーバレイでの AP のコーディング方法」，および「第 5 編　リファレンス」を参照してください。

COBOL については，マニュアル「COBOL2002 言語 標準仕様編」を参照してください。

8 XMAP3 Cosminexus 連携（Java）

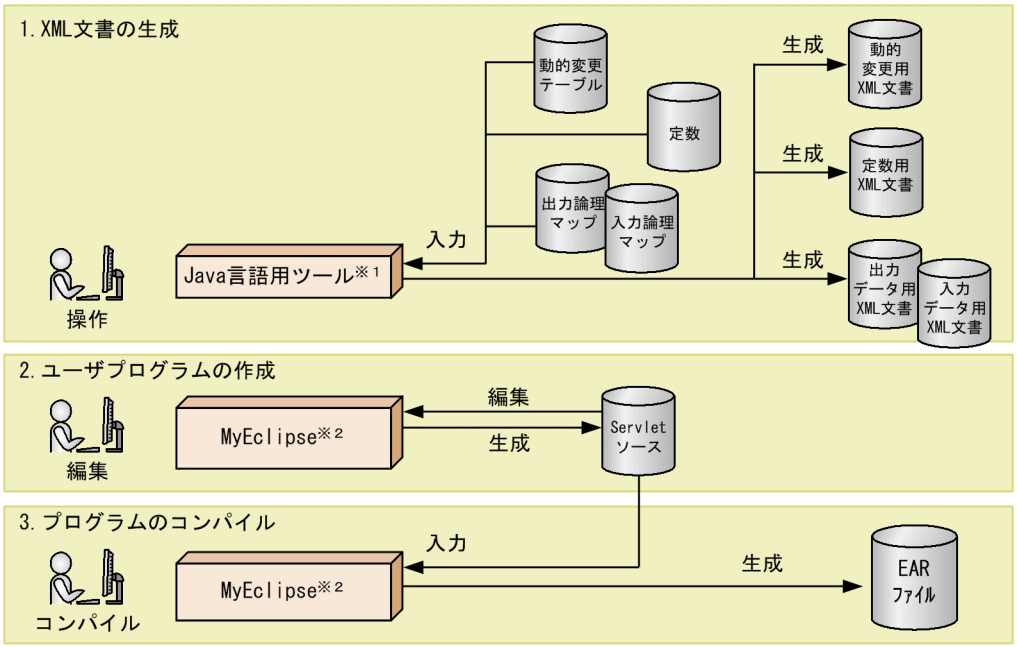
この章では，XMAP3 Cosminexus 連携を利用したシステムで動作する業務サーブレットの作成方法について説明します。

8.1　AP 開発の概要

XMAP3 Cosminexus 連携を利用したシステムの開発手順について説明します。

AP 開発の手順を次の図に示します。

図 8-1　AP 開発の手順



注※1　Java言語用ツールについては，マニュアル「XMAP3 開発ガイド」を参照してください。
注※2　Eclipse，JBuilderも使用できます。

AP 開発の手順について説明します。

1. XML 文書の生成

Java 言語用ツール機能を利用して，AP で使用する XML 文書を作成します。

Java 言語用ツール機能を使用して，次のファイルが生成されます。

- 入力データ用 XML 文書
- 出力データ用 XML 文書
- 定数用 XML 文書
- 動的変更用 XML 文書

Java 言語用ツールを利用して，AP で使用する XML 文書を作成する方法については，マニュアル「XMAP3 開発ガイド」を参照してください。

2. ユーザプログラムの作成

通信処理および業務処理を記述したユーザプログラムを作成します。XMAP3 Cosminexus 連携機能では，これを業務サーブレットと呼びます。

業務サーブレットの作成方法については，「8.4　業務サーブレットの作成」を参照してください。

3. プログラムのコンパイル

業務サーブレットをコンパイルして，EAR ファイルを作成します。

コンパイルの方法については，「8.7　業務サーブレットのコンパイルと EAR ファイルの生成」を参照してください。

8.1.1　開発環境の準備

XMAP3 Cosminexus 連携を利用して Web アプリケーションを開発するための準備について説明します。

開発環境で，資源を格納するフォルダを準備しておくと，開発環境と同じフォルダ構成のまま，実行環境に配置（デプロイ）できます。このため，開発環境は実行環境と同じフォルダ構成で準備してください。

開発環境の準備は，開発する Web アプリケーションのプロジェクト単位に実施する必要があります。

(1)　フォルダの準備

Web アプリケーションのプロジェクト単位に，資源（各ファイル）を格納する専用のフォルダを作成します。推奨するフォルダ構成例を次に示します。

```
+-- WebAppProject A（開発環境フォルダ）
|     +-- MAPS（マップ定義ファイル格納フォルダ）
|     +-- SEQ（遷移ファイル格納フォルダ）
|     +-- SOURCE（Javaソース格納フォルダ）
|     +-- WebApp（実行環境フォルダ）
|          +-- Data（データファイル格納フォルダ）
|          +-- WEB-INF（WEB-INFフォルダ）
|              +-- classes（classファイル格納フォルダ）
|              +-- lib（実行クラスライブラリ格納フォルダ）
|              +-- XMAP3（XMAP3データ参照フォルダ）
+-- WebAppProject B（開発環境フォルダ）
:
```

(2)　各フォルダの説明

各フォルダのフォルダ名および用途について次の表に示します。

表 8-1　各フォルダ名および用途

フォルダの種類	フォルダ名	用途	備考
開発環境フォルダ	任意	Web アプリケーション単位（プロジェクトごと）の開発資源をまとめて管理するためのルートフォルダです。	－
マップ定義ファイル格納フォルダ	任意※1	開発で作成する次のファイルを格納するフォルダです。 • マップ定義ファイル • 入力／出力論理マップファイル • 定数ファイル	－
遷移ファイル格納フォルダ	任意※1	開発で作成する画面遷移ファイルを格納するフォルダです。	－
Java ソース格納フォルダ	任意※1	業務サーブレットを格納するフォルダです。	－
実行環境フォルダ	任意※2	実行資源をまとめて管理するためのフォルダです。EAR ファイル作成時のフォルダとなります。	－
データファイル格納フォルダ	任意※2	実行時に必要な次のファイルを格納するフォルダです。	起動 HTML の「DataPath」パラ

フォルダの種類	フォルダ名	用途	備考
		• 物理マップファイル • ポップアップメニューファイル • グラフィックデータファイル • FAX 宛先ファイル	メタに，このフォルダのパスを指定してください。
WEB-INF フォルダ	WEB-INF（固定）	実行環境サブフォルダです。	－
class ファイル格納フォルダ	classes（固定）	Java ソース格納フォルダ内のソースをコンパイルして作成した class ファイルを格納するフォルダです。また，実行時に必要な次のファイルを格納するフォルダです。 • 環境管理ファイル（xmap3.properties）	－
実行クラスライブラリ格納フォルダ	lib（固定）	実行時に必要な次のファイルを格納するフォルダです。 • XMAP3 実行クラスライブラリ（xmap3server.jar）	－
XMAP3 データ参照フォルダ	XMAP3※3	実行時に必要な次の参照ファイルを格納するフォルダです。 • 通信制御用 XML 文書（X3COMTBL.xml） • 動的変更用 XML 文書 • 入力／出力データ用 XML 文書 • 定数用 XML 文書	「XMAP3」以外のフォルダに参照ファイルを格納する場合は，環境管理ファイルでフォルダのパスを指定してください。

（凡例）

－：該当しない。

注※1

フォルダ名には，特殊文字を除く半角英数字だけを使用してください。

注※2

フォルダ名には，次に示す特殊文字を除く半角英数字だけを使用してください。

「¥」，「/」，「:」，「,」，「;」，「*」，「?」，「"」，「<」，「>」，「¦」

注※3

環境管理ファイルで参照ファイルの格納先を指定する場合は，フォルダ名を任意に変更できます。

(3)　フォルダ構成の注意事項

フォルダを作成するときには，次の内容に注意してください。

- 「WEB-INF フォルダ」，「class ファイル格納フォルダ（classes フォルダ）」，および「実行クラスライブラリ格納フォルダ（lib フォルダ）」は，フォルダ構成やフォルダ名を変更しないでください。また，Web アプリケーションの実行時に，これらのフォルダに必要なファイルが格納されてない場合，Web アプリケーションは動作しません。
- 「XMAP3」フォルダに次のファイルを格納している場合，環境管理ファイルにフォルダパスを指定する必要はありません。
  - 通信制御用 XML 文書
  - 動的変更用 XML 文書
  - 入力／出力データ用 XML 文書

- 定数用 XML 文書

ただし，「XMAP3」フォルダ以外のフォルダにそれぞれのファイルを格納している場合（「XMAP3」フォルダ下にサブフォルダを作成した場合も含む），環境管理ファイルに，それぞれのファイルを格納しているフォルダパスを指定する必要があります。「XMAP3」フォルダ以外のフォルダを利用する場合，フォルダ名には，特殊文字を除いた半角英数字だけを使用してください。

また，上記の四つのファイルすべてを「XMAP3」フォルダ以外に格納していて，「XMAP3」フォルダにファイルが一つもない場合（空フォルダとなる場合）は，「XMAP3」フォルダを作成する必要はありません。

(4)　フォルダ構成のポイント

フォルダを作成するときには，次の内容を参考にしてください。

- 画面や帳票の変更によってプログラムを修正したり更新したりするため，プロジェクト単位に次のフォルダを作成および配置してください。
  - 「マップ定義ファイル格納フォルダ」
  - 「遷移ファイル格納フォルダ」
  - 「Java ソース格納フォルダ」
- 「実行環境フォルダ」下のフォルダ構成は，開発時に生成するファイルをあとでデプロイするため，Web アプリケーションの開発時から，同じフォルダ構成で作業してください。
- 「データファイル格納フォルダ」および「XMAP3 データ参照フォルダ」は，起動 HTML および環境管理ファイルで格納先を指定できるため，「実行環境フォルダ」以外のフォルダにも格納できます。

8.2　XML 文書

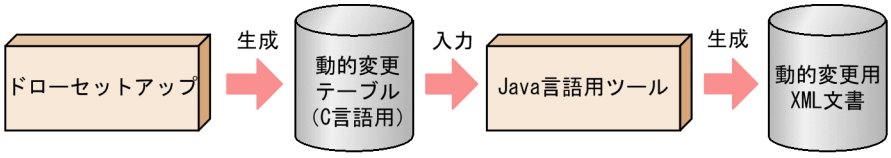
XML 文書は Java 言語用ツールを使用して作成します。Java 言語用ツールを使用すると，Java の環境で実行するときに参照する動的変更用 XML 文書，入力／出力データ用 XML 文書，定数用 XML 文書を生成できます。ここでは，XML 文書の形式について説明します。

8.2.1　動的変更用 XML 文書

ドローセットアップで「C 言語」用に生成した動的変更テーブル（X3MODTBL.h）は，Java 言語用ツールで動的変更用 XML 文書に変換します。

動的変更用 XML 文書を生成する流れの概要を次の図に示します。

図 8-2　動的変更用 XML 文書を生成する流れ



(1)　生成されるファイルの形式

動的変更用 XML 文書は，動的変更テーブルの形式と値を記述する XML ファイル（拡張子は「*.xml」）です。XMAP3 Cosminexus 連携機能の XMAP3 実行クラスライブラリを使用するための参照用ファイルとして，Java 言語用ツールで動的変更テーブルから生成します。生成した動的変更用 XML 文書はカスタマイズできません。

業務サーブレットで画面遷移や業務プログラムを作成する場合，XML ビューアなどを使用して，動的変更用 XML 文書からデータ項目名，形式，および修飾名などを参照できます。ただし，Java 言語用ツール以外で動的変更用 XML 文書を更新した場合は，動作を保証しません。画面や帳票の属性を変更した場合は，Java 言語用ツールを利用して再度，動的変更用 XML 文書を生成してください。

動的変更用 XML 文書の記述例を次に示します。

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<X3MODTBL>
  <CD><INDEX>0</INDEX><LEVEL>1</LEVEL><NAME>XMAP_NODATA</NAME>
      <LENG>1</LENG><TYPE>unsigned char</TYPE><REPEAT>1</REPEAT>
      <VAL>0x1F</VAL></CD>
    :
  <CD><INDEX>0</INDEX><LEVEL>1</LEVEL><NAME>xmap_line_dt</NAME>
      <LENG>2</LENG><TYPE>static char</TYPE><REPEAT>1</REPEAT>
      <VAL>"DT"</VAL></CD>
</X3MODTBL>
```

動的変更用 XML 文書で使用するタグを次の表に示します。これらのタグの記述はカスタマイズできません。

表 8-2　動的変更用 XML 文書で使用するタグ

タグ名	説明
<X3MODTBL>	動的変更用 XML 文書名（X3MODTBL は固定値）です。 ルートオブジェクトとして宣言します。
<CD>	定数のデータ名と値を宣言します。

タグ名	説明
<INDEX>	常に 0 が設定されます。
<LEVEL>	常に 1 が設定されます。
<NAME>	データ名です。
<LENG>	修飾名長です。
<TYPE>	データ型（static char／unsigned char）です。 static char および unsigned char は，文字列または 16 進数を示します。
<REPEAT>	常に 1 が設定されます。
<VAL>	修飾名です。 文字列は，ダブルクォーテーション（"）で囲みます。 文字として表現できないものは，0xHH で表現します。また，複数バイトの場合は，コンマ（,）で連結します。

8.2.2　入力データ用 XML 文書，出力データ用 XML 文書，および定数用 XML 文書

Java 言語用ツールは，入力論理マップ，出力論理マップ，および定数ファイル※1 を基に，**入力データ用 XML 文書**※2，**出力データ用 XML 文書**※2，および**定数用 XML 文書**をそれぞれ生成します。

注※1

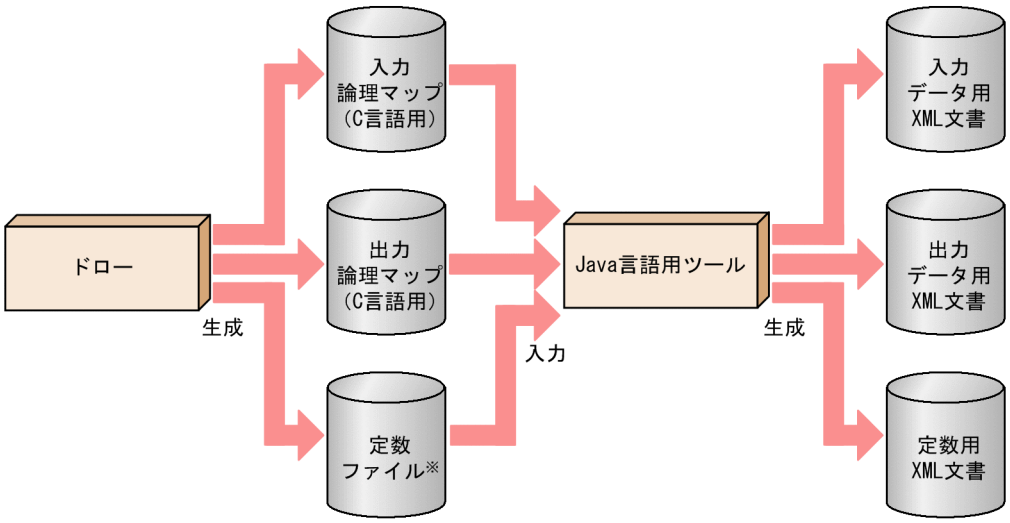
定数ファイルは，ドローセットアップの論理マップ属性で「定数部の別ファイル出力」を選択していない場合，出力論理マップの一部に含まれるため生成されません。この場合，Java 言語用ツールの入力ファイルとして指定する必要はありません。Java 言語用ツールの入力ファイルとして定数ファイルを指定しなくても，出力論理マップに定数情報が含まれるため，自動的に定数用 XML 文書が生成されます。

注※2

ドローの画面属性ダイアログおよび帳票属性ダイアログの「再定義名」を設定して生成した論理マップを基に生成した入力データ用 XML 文書と出力データ用 XML 文書では，「再定義名」の設定は無効となります。

入力データ用 XML 文書，出力データ用 XML 文書，および定数用 XML 文書が生成されるまでの流れの概要を次の図に示します。

図 8-3　入力データ用 XML 文書，出力データ用 XML 文書，および定数用 XML 文書が生成されるまでの流れ



Java 言語用ツールの使用方法については，マニュアル「XMAP3 開発ガイド」を参照してください。

(1)　生成されるファイルの形式

Java 言語用ツールで生成されるファイルは，次のとおりです。

- 入力データ用 XML 文書
- 出力データ用 XML 文書
- 定数用 XML 文書

入力データ用 XML 文書と出力データ用 XML 文書は，画面または帳票で定義したデータ名，長さ，およびデータ型などの形式を記述する XML ファイル（拡張子は「*.xml」）です。定数用 XML 文書は，定数テーブルの形式および値を記述する XML ファイル（拡張子は「*.xml」）です。生成された各 XML ファイルはカスタマイズできません。

業務サーブレットで画面遷移や業務プログラムを作成する場合，XML ビューアなどを使用して，各 XML 文書からデータ項目名，形式，および修飾名などを参照できます。ただし，Java 言語用ツール機能以外で各 XML 文書を更新した場合は，動作を保証しません。画面や帳票の属性を変更した場合，または，画面や帳票の設計を変更した場合は，Java 言語用ツールを利用し再度，各 XML 文書を生成してください。

入力データ用 XML 文書および出力データ用 XML 文書，ならびに定数用 XML 文書の構造について次に説明します。

(2)　入力データ用 XML 文書および出力データ用 XML 文書

入力データ用 XML 文書および出力データ用 XML 文書の記述を，出力論理マップ（MAP001O.h）を変換した場合を例にして次に示します。

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<LOGMAP>
  <FD><INDEX>0</INDEX><LEVEL>1</LEVEL><NAME>MAP001L</NAME>
    <LENG>2</LENG><TYPE>short</TYPE><REPEAT>1</REPEAT></FD>
  <FD><INDEX>2</INDEX><LEVEL>1</LEVEL><NAME>MAP001Z</NAME>
    <LENG>2</LENG><TYPE>unsigned char</TYPE><REPEAT>1</REPEAT></FD>
  :
  <FD><INDEX>950</INDEX><LEVEL>1</LEVEL><NAME>MAP001_FIELD0017_0</NAME>
```

```
    <LENG>6</LENG><TYPE>unsigned char</TYPE><REPEAT>1</REPEAT></FD>
</LOGMAP>
```

入力データ用 XML 文書および出力データ用 XML 文書で使用するタグを次の表に示します。これらの記述はカスタマイズできません。

表 8-3　入力データ用 XML 文書および出力データ用 XML 文書で使用するタグ

タグ名	説明
<LOGMAP>	入力データ用 XML 文書および出力データ用 XML 文書名（LOGMAP は固定値）です。 ルートオブジェクトとして宣言します。
<FD>	画面または帳票で定義している入力データ名および出力データ名を宣言するための共通タグです。
<INDEX>	先頭からの位置です。ただし，フレーム内のデータ名の場合は，フレームの先頭からの位置となります。
<LEVEL>	データ名の階層です。
<NAME>	データ名です。
<LENG>	データ長です。ただし，繰り返し項目の場合および階層を持つ項目の場合は，繰り返し単位の長さとなります。
<TYPE>	データ型（short／unsigned char／struct）です。 short は数値を，unsigned char は文字列または 16 進数を示します。struct はフレーム，メニュー，ボタン，または下位項目を持つテキストなどの階層構造を宣言する場合に使用します。
<REPEAT>	繰り返し回数です。ただし，繰り返しがない項目の場合は，1 が設定されます。

(3)　定数用 XML 文書

定数用 XML 文書の記述例を次に示します。

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<CONST>
  <CD><INDEX>0</INDEX><LEVEL>1</LEVEL><NAME>MAP001T</NAME>
    <LENG>2</LENG><TYPE>short</TYPE><REPEAT>1</REPEAT><VAL>956</VAL></CD>
  <CD><INDEX>0</INDEX><LEVEL>1</LEVEL><TYPE>struct</TYPE>
    <CD><INDEX>0</INDEX><LEVEL>2</LEVEL><NAME>MAP001_transactionfieldT</NAME>
        <LENG>2</LENG><TYPE>unsigned char</TYPE><REPEAT>1</REPEAT></CD>
        :
    <CD><INDEX>74</INDEX><LEVEL>2</LEVEL><NAME>MAP001_PBOX0001_T</NAME>
        <LENG>2</LENG><TYPE>unsigned char</TYPE><REPEAT>1</REPEAT></CD>
    <NAME>MAP001D</NAME><LENG>76</LENG><REPEAT>1</REPEAT>
    <VAL>0x00,0x04 , 0x00,0x13 , 0x00,0x15 , 0x00,0x8D ,
                  :
          0x00,0xF5 , 0x00,0xFD , 0x00,0x0D , 0x80,0x01</VAL>
  </CD>
</CONST>
```

定数用 XML 文書で使用するタグを次の表に示します。これらの記述はカスタマイズできません。

表 8-4　定数用 XML 文書で使用するタグ

タグ名	説明
<CONST>	定数用 XML 文書名（CONST は固定値）です。ルートオブジェクトとして宣言します。
<CD>	画面または帳票で定義している定数のデータ名を宣言するための共通タグです。

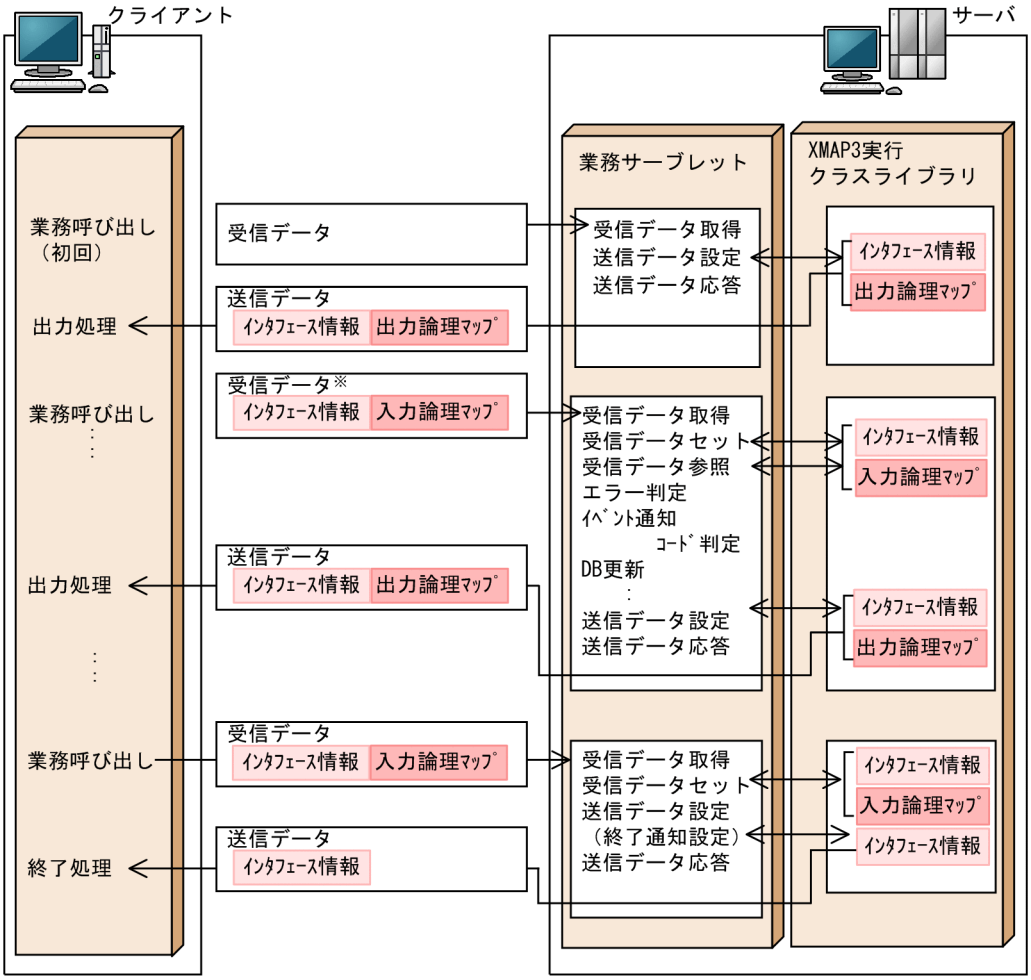
タグ名	説明
<INDEX>	先頭からの位置です。ただし，フレーム内のデータ名の場合は，フレームの先頭からの位置となります。
<LEVEL>	データ名の階層です。
<NAME>	データ名です。
<LENG>	データ長です。ただし，繰り返し項目の場合および階層を持つ項目の場合は，繰り返し単位の長さとなります。
<TYPE>	データ型（short／unsigned char／struct）です。 short は数値を，unsigned char は文字列または 16 進数を示します。struct は，フレーム，メニュー，ボタン，または下位項目を持つテキストなどの階層構造を宣言する場合に使用します。
<REPEAT>	繰り返し回数です。ただし，繰り返しがない項目の場合は，1 が設定されます。
<VAL>	定数値です。

8.3　クライアントとサーバ間の処理の流れ

ユーザが作成するサーバ側のプログラムとして業務サーブレットを作成します。

クライアント側と，サーバ側のプログラムとのやり取りについて，処理の流れを次の図に示します。

図 8-4　クライアントとサーバ間の処理の流れ

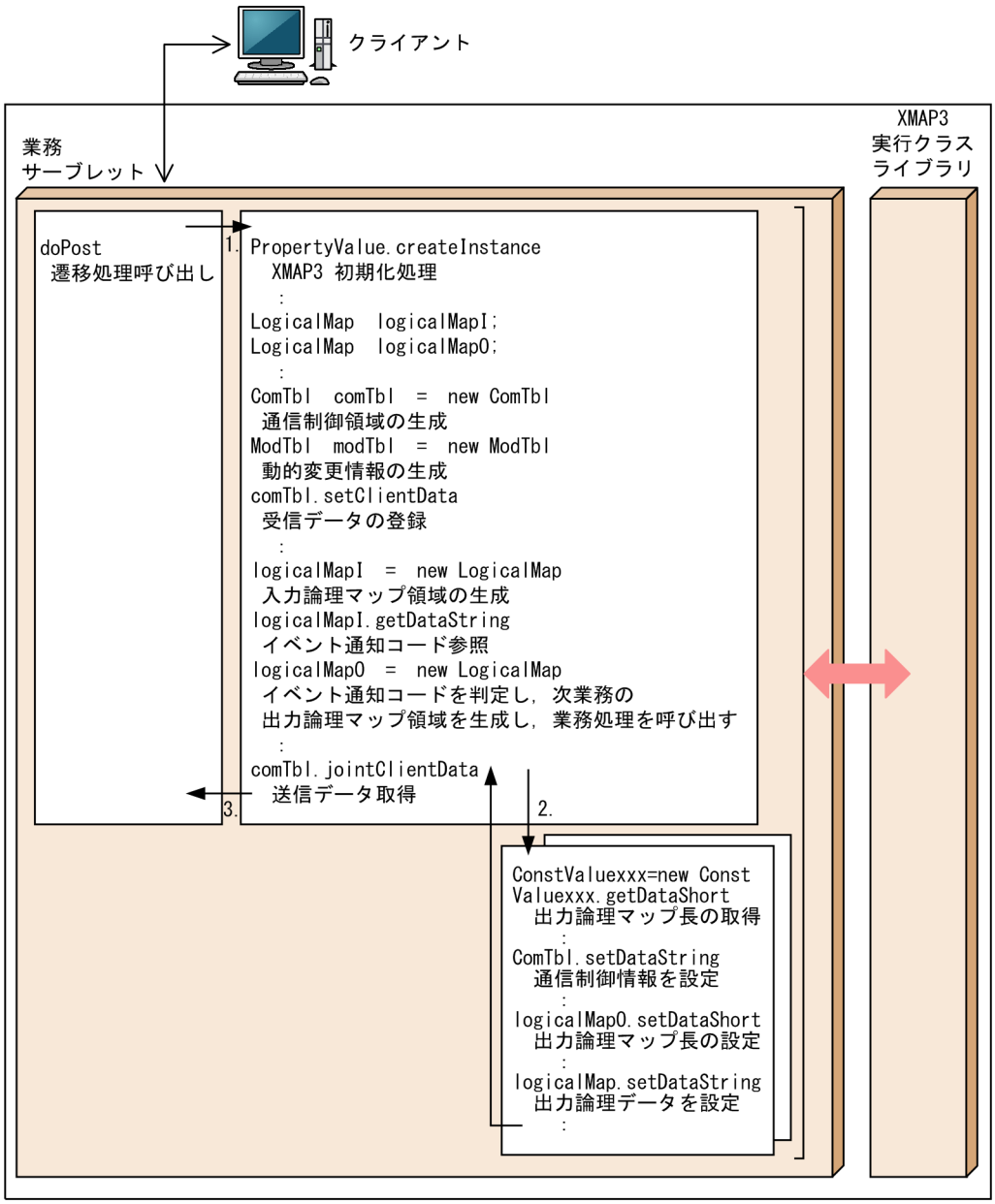


注※　帳票出力およびエラー発生の場合はインタフェース情報だけ設定されます。

XMAP3 Cosminexus 連携機能では，送受信データの長さの情報を定義するデータ長設定領域，インタフェース情報を定義する共通インタフェース領域，および論理マップ領域を合わせた定義を送受信データとして使用します。

クライアント側から業務呼び出しが発生したときの，サーバ側でのプログラム間の処理の流れを次の図に示します。

図 8-5　サーバ側でのプログラム間の処理の流れ



1.業務サーブレット内で遷移処理呼び出し

クライアント側からの呼び出しを受けて業務サーブレットがデータを受信します。

2.出力論理マップ領域の生成と業務処理の呼び出し

業務サーブレットから XMAP3 実行クラスライブラリが提供するクラスおよびメソッドを使用して，クライアント側からの受信データから入力論理データを参照し，次画面または帳票の出力データを設定します。

3.送信データの取得

取得した送信データは業務サーブレット内で組み立てられ，クライアント側へ送信されます。

8.4　業務サーブレットの作成

業務サーブレットは，サーバ側で動作するユーザの業務用 Java プログラムです。通信処理と，業務処理から構成されます。

業務サーブレットには，クライアントとの通信，画面遷移，および業務処理などの処理を記述します。クライアントとの通信および送受信データの制御には，XMAP3 実行クラスライブラリを利用します。開発する Web アプリケーションのシステムに合わせて画面遷移や業務用にクラスを利用するなど，処理方法はユーザの任意です。

業務サーブレットの通信処理部分は，XMAP3/Web for Cosminexus が提供するサンプルソースをカスタマイズして使用します。カスタマイズには，MyEclipse を使用します。

8.4.1　業務サーブレット作成のポイント

業務サーブレットを作成するときには，次のようなポイントがあります。

- 遷移する画面や帳票が多い場合，または遷移条件が複雑な場合には，通信制御用のクラスまたは画面遷移用のクラスを作成してください。
- 画面ごとまたは帳票ごとに詳細な業務処理が必要な場合は，画面ごとまたは帳票ごとにクラスを作成してください。

業務サーブレットを作成するときには，次の内容に注意してください。

入力／出力データ用 XML 文書，定数用 XML 文書の参照方法

各 XML 文書から作成したオブジェクトを作成します。画面や帳票に出力するデータ，または画面から入力したデータは，オブジェクトの setter／getter メソッドを使用して Java プログラムからアクセスします。オブジェクトの setter／getter メソッドは，XMAP3 実行クラスライブラリを利用します。

文字コードについて

Java で扱う文字コード（UCS-2）と各論理マップの文字コード（シフト JIS）は自動的に変換します。外字については，シフト JIS（MS932）の範囲の文字コード（1,880 文字）が利用できます。

例外処理について

各論理マップのアクセスなどで例外が発生した場合に，ユーザ独自の処理を記述できます。

動的変更用 XML 文書の参照方法

動的変更用 XML 文書から作成した動的変更用のオブジェクトを作成します。このオブジェクトは，Java プログラムから setter／getter メソッドを利用してアクセスします。オブジェクトの setter／getter メソッドは，XMAP3 実行クラスライブラリを利用します。オブジェクトは，セッション上のオブジェクトとして利用します。

通信制御用 XML 文書の参照方法

通信制御用 XML 文書から作成した通信制御用のオブジェクトを作成します。このオブジェクトは，Java プログラムから setter／getter メソッドを利用してアクセスします。オブジェクトの setter／getter メソッドは，XMAP3 実行クラスライブラリを利用します。オブジェクトは，セッション上のオブジェクトとして利用します。

クラス名について

業務サーブレットを作成するとき，作成するクラス名として「xmap3」は使用できません。クラス名は大文字と小文字を識別するため「XMAP3」は使用できます。

(1) 通信処理の記述のポイント

業務サーブレットの通信処理部分は，XMAP3/Web for Cosminexus が提供するサンプルソースをカスタマイズして使用します。カスタマイズには，MyEclipse を使用します。XMAP3/Web for Cosminexus では，次に示すフォルダに業務サーブレットの通信処理部分のソース（SampleServlet.java）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥JAVA¥sample¥servlet

業務サーブレットの通信処理部分のサンプルソースを次に示します。

サンプルソース

```
package sample.servlet;

import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import jp.co.Hitachi.soft.XMAP3.server.ComTbl;
import jp.co.Hitachi.soft.XMAP3.server.PropertyValue;
import jp.co.Hitachi.soft.XMAP3.server.XmapEnvironmentException;
import jp.co.Hitachi.soft.XMAP3.server.XmapException;
import jp.co.Hitachi.soft.XMAP3.server.XmapRuntimeException;

import sample.appmanager.AppManager;

public class SampleServlet extends HttpServlet {
    private static final String CONTENT_TYPE = "application/octet-stream";

    public void doPost(
        HttpServletRequest request,
        HttpServletResponse response)
        throws IOException {

        InputStream inputStream = null;
        OutputStream outputStream = null;

        try {
            inputStream = request.getInputStream();
            response.setContentType(CONTENT_TYPE);
            outputStream = response.getOutputStream();

            final int contentLength = request.getContentLength();
            if (contentLength != -1) {

                byte[] clientDataI = new byte[contentLength];
                final int readLength = inputStream.read(clientDataI);

                if (contentLength == 0 || readLength == clientDataI.length) {

                    byte[] clientDataO = null;
                    try {

                        PropertyValue prop =
                            PropertyValue.createInstance(getServletContext());
                        clientDataO = AppManager.invoke(prop, clientDataI);

                    } catch (XmapRuntimeException e) {
                        e.printStackTrace();
                        errorCase(outputStream);
                        return;
                    } catch (XmapException e) {
                        e.printStackTrace();
                        errorCase(outputStream);
                        return;
                    } catch (XmapEnvironmentException e) {
```

```
                        e.printStackTrace();
                        errorCase(outputStream);
                        return;
                    }

                    if (clientData0 != null) {
                        //正常ケース
                        outputStream.write(clientData0, 0, clientData0.length);
                        return;
                    } else {
                        //クライアントに返すデータが作れなかった場合
                        errorCase(outputStream);
                        return;
                    }
                } else {
                    //入力ストリームからのデータの読み込みに失敗した場合
                    errorCase(outputStream);
                    return;
                }
            } else {
                //入力ストリームから読み込めるバイト長が不明な場合
                errorCase(outputStream);
                return;
            }

        } finally {
            try {
                if (inputStream != null) {
                    inputStream.close();
                }
                if (outputStream != null) {
                    outputStream.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

    }

    private final void errorCase(OutputStream outputStream)
        throws IOException {
        byte[] clientData0 = new byte[0];
        try {
            PropertyValue prop =
                PropertyValue.createInstance(getServletContext());
            ComTbl com = new ComTbl(prop, (byte) 0x00);
            com.setDataString("XMAP_ID", "*XWC");
            com.setDataString("XMAP_ENDOPT", "E");
            com.setDataShort("XMAP_URLLNG", (short) 128);
            //エラー時の画面遷移を指定する
            com.setDataString(
                "XMAP_URL",
                "http://localhost:8080/XmapWebApp/error.html");
            clientData0 = com.getClientData();
        } catch (XmapException e) {
            e.printStackTrace();
        } catch (XmapEnvironmentException e) {
            e.printStackTrace();
        }
        outputStream.write(clientData0, 0, clientData0.length);
        return;
    }

}
```

(2)　業務処理の記述のポイント

入力／出力データ用オブジェクトに対して，画面単位または帳票単位にデータの取得処理か設定処理をします。

XMAP3/Web for Cosminexus のサンプルプログラムでは，さまざまな業務処理を業務 JavaAP として提供しています。業務処理部分のサンプルソースとして提供している業務 JavaAP の中から「KAD2GHND_PF02.java」を次に示します。この業務 JavaAP では，ボタンまたは PF02 をクリックしたとき，次画面を表示するための通信制御用のデータと画面表示用のデータを設定しています。

サンプルソース

```
package sample.apps;

import jp.co.Hitachi.soft.XMAP3.server.ComTbl;
import jp.co.Hitachi.soft.XMAP3.server.ConstValue;
import jp.co.Hitachi.soft.XMAP3.server.LogicalMap;
import jp.co.Hitachi.soft.XMAP3.server.PropertyValue;
import jp.co.Hitachi.soft.XMAP3.server.XmapEnvironmentException;
import jp.co.Hitachi.soft.XMAP3.server.XmapException;
import jp.co.Hitachi.soft.XMAP3.server.XmapRuntimeException;

import sample.appmanager.AppInterface;
import sample.appmanager.AppManager;

public class KAD2GHND_PF02 implements AppInterface {

    public LogicalMap doMethod(
        PropertyValue prop,
        ComTbl comTbl,
        LogicalMap logicalMapI,
        LogicalMap logicalMapO,
        ConstValue constValue)
        throws XmapRuntimeException, XmapException, XmapEnvironmentException {

        comTbl.init((byte) 0x00);
        final short lsglng = constValue.getDataShort("KAD4GHT");
        comTbl.setDataString("XMAP_ID", "*XWC");
        comTbl.setDataShort("XMAP_RTN", (short) 0);
        comTbl.setDataShort("XMAP_RSN", (short) 0);
        comTbl.setDataInteger("XMAP_MAPOPT1", 0);
        comTbl.setDataShort("XMAP_URLLNG", (short) 0);
        comTbl.setDataString("XMAP_TNAME", "        ");
        comTbl.setDataString("XMAP_MSG", "BWS ");
        comTbl.setDataString("XMAP_MAPNAME", "KAD4GHND");
        comTbl.setDataString("XMAP_ENDOPT", " ");
        comTbl.setDataShort("XMAP_LSGLNG", lsglng);

        logicalMapO.init(AppManager.XMAP_NODATA);
        logicalMapO.setDataShort("KAD4GHL", lsglng);
        logicalMapO.setDataString("KAD4GH_KNAME_0", "○×Ａ１電器");
        logicalMapO.setDataString("KAD4GH_DCODE_0", "D0012345");
        logicalMapO.setDataString("KAD4GH_KCODE_0", "K0012345");
        logicalMapO.setDataString("KAD4GH_TNAME_0", "日立　太郎");
        logicalMapO.setDataString("KAD4GH_TCODE_0", "T01");
        logicalMapO.setDataString("KAD4GH_SICODE_0", "S1");

        return logicalMapO;
    }

}
```

8.4.2　推奨メソッド

XMAP3/Web for Cosminexus が提供するクラスで，使用を推奨しているメソッドについて説明します。XMAP3/Web for Cosminexus が提供するクラスとメソッドについては，「14.　Java のクラス」を参照してください。

(1)　LogicalMap クラス

業務サーブレットで入力データおよび出力データを編集する場合に使用する setter／getter メソッドは，入力データ用 XML 文書および出力データ用 XML 文書の<TYPE>タグに定義した内容によって異なります。<TYPE>タグの内容と推奨するメソッドを次の表に示します。

表 8-5　<TYPE>タグの内容と推奨するメソッド（LogicalMap クラス）

メソッドの種類	<TYPE>タグ	扱うデータ	推奨するメソッド	備考
setter	short	数値	setDataShort	長さ・枚数に使用します。
	unsigned char	バイナリデータ	setDataByteArray	定数・データ有無コードに使用します。
		文字列	setDataString	項目データに使用します。
getter	short	数値	getDataShort	長さ・枚数に使用します。
	unsigned char	バイナリデータ	getDataByteArray	定数・データ有無コードに使用します。
		文字列	getDataString	項目データに使用します。

(2)　ConstValue クラス

業務サーブレットで定数を取得する場合に使用する getter メソッドは，定数用 XML 文書の<TYPE>タグに定義した内容によって異なります。<TYPE>タグの内容と推奨するメソッドを次の表に示します。推奨するメソッドが複数あるメソッドについては，「繰り返し」の指定の有無によって使い分けてください。

表 8-6　<TYPE>タグの内容と推奨するメソッド（ConstValue クラス）

メソッドの種類	<TYPE>タグ	扱うデータ	推奨するメソッド
getter	short	数値	getDataShort
	unsigned char	バイナリデータ	getDataByteArray

(3)　ModTbl クラス

業務サーブレットで定数を取得する場合に使用する getter メソッドは，動的変更用 XML 文書の<TYPE>タグに定義した内容によって異なります。<TYPE>タグの内容と推奨するメソッドを次の表に示します。

表 8-7　<TYPE>タグの内容と推奨するメソッド（ModTbl クラス）

メソッドの種類	<TYPE>タグ	扱うデータ	推奨するメソッド
getter	static char	文字列	getDataString
	unsigned char	バイナリデータ	getDataByteArray

(4)　ComTbl クラス

業務サーブレットで共通インタフェース領域を編集する場合に使用する setter／getter メソッドは，通信制御用 XML 文書の<TYPE>タグに定義した内容によって異なります。<TYPE>タグの内容と推奨するメソッドを次の表に示します。

表 8-8　<TYPE>タグの内容と推奨するメソッド（ComTbl クラス）

メソッドの種類	<TYPE>タグ	扱うデータ	推奨するメソッド
setter	short	数値	setDataShort
	int	数値	setDataInteger
	String	文字列	setDataString
getter	short	数値	getDataShort
	int	数値	getDataInteger
	String	文字列	getDataString

8.4.3　Java のクラス一覧

XMAP3/Web for Cosminexus が提供する Java のクラス（メソッド）の一覧を次に示します。XMAP3/Web for Cosminexus が提供するクラスとメソッドについては，「14.　Java のクラス」を参照してください。

(1)　PropertyValue クラス

PropertyValue クラスのメソッド一覧を次の表に示します。

表 8-9　PropertyValue クラスのメソッド一覧

戻り値のデータ型	メソッドのシグネチャ	機能	備考
PropertyValue	createInstance (ServletContext context)	環境管理ファイルを読み込み，PropertyValue クラスのインスタンスを生成します。	static メソッド

(2)　LogicalMap クラス

LogicalMap クラスのフィールド一覧，コンストラクター一覧，およびメソッド一覧を次の表に示します。

表 8-10　LogicalMap クラスのフィールド一覧

フィールドのデータ型	フィールド名	機能	備考
int	MAX_LENGTH	入力データの最大長を示します。	static フィールド

表 8-11　LogicalMap クラスのコンストラクター一覧

コンストラクタのシグネチャ	機能
LogicalMap (PropertyValue propValue, String mapName, byte initData, byte fillCharacter)	環境管理用クラスを指定して，LogicalMap オブジェクトを構築します。

表 8-12　LogicalMap クラスのメソッド一覧

戻り値のデータ型	メソッドのシグネチャ	機能
void	setDataShort (String dataName, short data, int rpt)	XML 文書の該当する繰り返し項目に数値データを設定します。

戻り値のデータ型	メソッドのシグネチャ	機能
void	setDataShort (String dataName, short data)	XML 文書の該当するデータ項目に数値データを設定します。
void	setDataByteArray (String dataName, byte[] data, int rpt, Filler filler, byte fillCharacter)	XML 文書の該当する繰り返し項目にバイトデータ配列を設定します。
void	setDataByteArray (String dataName, byte[] data, Filler filler, byte fillCharacter)	XML 文書の該当するデータ項目にバイトデータ配列を設定します。
void	setDataByteArray (String dataName, byte[] data, int rpt)	XML 文書の該当する繰り返し項目にバイトデータ配列を設定します。
void	setDataByteArray (String dataName, byte[] data)	XML 文書の該当するデータ項目にバイトデータ配列を設定します。
void	setDataString (String dataName, String data, int rpt, Filler filler, byte fillCharacter)	XML 文書の該当する繰り返し項目に文字配列を設定します。
void	setDataString (String dataName, String data, Filler filler, byte fillCharacter)	XML 文書の該当するデータ項目に文字配列を設定します。
void	setDataString (String dataName, String data, int rpt)	XML 文書の該当する繰り返し項目に文字配列を設定します。
void	setDataString (String dataName, String data)	XML 文書の該当するデータ項目に文字配列を設定します。
short	getDataShort (String dataName, int rpt)	XML 文書の該当する繰り返し項目の数値データを取得します。
short	getDataShort (String dataName)	論理マップの指定したデータ項目の数値データを取得します。
byte[]	getDataByteArray (String dataName, int rpt)	XML 文書の該当する繰り返し項目のバイトデータ配列を取得します。
byte[]	getDataByteArray (String dataName)	XML 文書の該当するデータ項目のバイトデータ配列を取得します。
String	getDataString (String dataName, int rpt)	XML 文書の該当する繰り返し項目の文字配列を取得します。
String	getDataString (String dataName)	XML 文書の該当するデータ項目の文字配列を取得します。
void	setClientData (byte[] data, int beginIndex, int lastIndex)	クライアントから受信した入力／出力データを設定します。
void	init (byte initData)	入力／出力データ用クラスを初期化します。

(3)　ConstValue クラス

ConstValue クラスのコンストラクター一覧およびメソッド一覧を次の表に示します。

表 8-13　ConstValue クラスのコンストラクター一覧

コンストラクタのシグネチャ	機能
ConstValue (PropertyValue propValue, String fileName)	環境管理クラスを指定して，ConstValue オブジェクトを構築します。

表 8-14　ConstValue クラスのメソッド一覧

戻り値のデータ型	メソッドのシグネチャ	機能
short	getDataShort (String dataName, int rpt)	指定した定数の数値データを取得します。
short	getDataShort (String dataName)	指定した定数の数値データを取得します。
byte[]	getDataByteArray (String dataName, int rpt)	指定した定数のバイトデータ配列を取得します。
byte[]	getDataByteArray (String dataName)	指定した定数のバイトデータ配列を取得します。
String	getDataString (String dataName, int rpt)	指定した定数の文字配列を取得します。
String	getDataString (String dataName)	指定した定数の文字配列を取得します。

(4)　ModTbl クラス

ModTbl クラスのコンストラクター一覧およびメソッド一覧を次の表に示します。

表 8-15　ModTbl クラスのコンストラクター一覧

コンストラクタのシグネチャ	機能
ModTbl (PropertyValue propValue, String fileName)	環境管理クラスを指定して，ModTbl オブジェクトを構築します。

表 8-16　ModTbl クラスのメソッド一覧

戻り値のデータ型	メソッドのシグネチャ	機能
short	getDataShort (String dataName)	動的変更用 XML 文書の該当する定数の数値データを取得します。
byte[]	getDataByteArray (String dataName)	動的変更用 XML 文書の該当する定数のバイトデータ配列を取得します。
String	getDataString (String dataName)	動的変更用 XML 文書の該当する定数の文字配列を取得します。

(5)　ComTbl クラス

ComTbl クラスのフィールド一覧，コンストラクター一覧，およびメソッド一覧を次の表に示します。

表 8-17　ComTbl クラスのフィールド一覧

フィールドのデータ型	フィールド名	機能	備考
int	LENGTH	通信制御データ長を示します。	static フィールド

表 8-18　ComTbl クラスのコンストラクター一覧

コンストラクタのシグネチャ	機能
ComTbl (PropertyValue propValue, byte initData)	環境管理クラスを指定して，ComTbl オブジェクトを構築します。

表 8-19　ComTbl クラスのメソッド一覧

戻り値のデータ型	メソッドのシグネチャ	機能
void	setDataShort (String dataName, short data)	通信制御の該当するデータ項目に数値データを設定します。
void	setDataInteger (String dataName, int data)	通信制御の該当するデータ項目に数値データを設定します。
void	setDataByteArray (String dataName, byte[] data)	通信制御の該当するデータ項目にバイトデータ配列を設定します。
void	setDataString (String dataName, String data)	通信制御の該当するデータ項目に文字配列を設定します。
short	getDataShort (String dataName)	通信制御の該当するデータ項目の数値データを取得します。
int	getDataInteger (String dataName)	通信制御の該当するデータ項目の数値データを取得します。
byte[]	getDataByteArray (String dataName)	通信制御の該当するデータ項目のバイトデータ配列を取得します。
String	getDataString (String dataName)	通信制御の該当するデータ項目の文字配列を取得します。
void	setClientData (byte[] data, int beginIndex, int lastIndex)	クライアントから受信した通信制御用データ文書を設定します。
byte[]	getClientData ( )	クライアントへ送信する通信制御データを取得します。
byte[]	jointClientData (byte[] data)	通信制御の該当するデータと指定したバイトデータ配列を連結します。
void	init (byte initData)	通信制御用クラスを初期化します。

(6)　Filler クラス

Filler クラスのフィールド一覧を次の表に示します。

表 8-20　Filler クラスのフィールド一覧

フィールドのデータ型	フィールド名	機能	備考
Filler	RIGHT	右寄せを指定します。	static フィールド
Filler	LEFT	左寄せを指定します。	static フィールド

8.4.4　通信制御用 XML 文書の形式

通信制御用 XML 文書（X3XCOMTBL.xml）は，XMAP3 で提供する実行クラスライブラリを参照するための XML 形式のファイルです。通信制御用 XML 文書には，送信データおよび受信データの形式を記述します。

通信制御用 XML 文書では，仮想端末名，印刷完了通知，および印刷終了通知などの情報を取得／設定できます。また，Java でプログラミングするときに，XML ビューアなどで XML 文書のデータ名，形式，および長さなどが参照できます。

XMAP3/Web for Cosminexus で提供する通信制御用 XML 文書の形式や内容は変更できません。

通信制御用 XML 文書は，Web アプリケーションを実行するときに必要なファイルです。

通信制御用 XML 文書の記述形式および通信制御用 XML 文書で定義する項目について説明します。

XMAP3/Web for Cosminexus では，次に示すフォルダに通信制御用 XML 文書を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥ETC

(1)　ファイルの記述形式

通信制御用 XML 文書を次に示します。

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<X3XCOMTBL>
  <COM>
    <INDEX>0</INDEX>
    <NAME>XMAP_ID</NAME>
    <LENG>4</LENG>
    <TYPE>String</TYPE>
  </COM>
    :
  <COM>
    <INDEX>20</INDEX>
    <NAME>XMAP_MAPNAME</NAME>
    <LENG>8</LENG>
    <TYPE>String</TYPE>
  </COM>
</X3XCOMTBL>
```

通信制御用 XML 文書で使用するタグを次の表に示します。

表 8-21　通信制御用 XML 文書で使用するタグ

タグ名	説明
<X3XCOMTBL>	通信制御用 XML 文書名（X3XCOMTBL は固定値）です。 ルートオブジェクトとして宣言します。
<COM>	定数のデータ名と値を宣言します。
<INDEX>	COM データの先頭からの位置です。
<NAME>	データ名です。
<LENG>	データ長です。
<TYPE>	データ型（String／short／int／byte）です。String は文字列を宣言する場合に使用します。short および int は数値を宣言する場合に使用します。byte はバイナリデータを宣言する場合に使用します。

(2)　通信制御用 XML 文書で定義する項目

通信制御用 XML 文書で定義する項目を次の表に示します。

表 8-22　通信制御用 XML 文書で定義する項目

データ項目名	データ長 (位置)	データ型	データ名	設定値
アイキャッチャ	4(0)	String	XMAP_ID	'*XWC'
リターン値 1	2(4)	short	XMAP_RTN	0
リターン値 2	2(6)	short	XMAP_RSN	0
仮想端末名	8(8)	String	XMAP_TNAME	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※1
通信種別	4(16)	String	XMAP_MSG	• 画面の場合：'BWS△' • 帳票の場合：'OWS△'
マップ名	8(20)	String	XMAP_MAPNAME	物理マップ名※2
未使用	4(28)	byte	XMAP_RSV1	すべて$(00)_{16}$
印刷完了通知オプション	1(32)	String	XMAP_PRTOPT	• '△'：帳票をスプールに出力後，印刷ドキュメントを完了（クローズ）します。 • '2'：印刷ドキュメントへの出力を継続します（クローズしない）。
終了通知オプション	1(33)	String	XMAP_ENDOPT	• ブラウザ側の業務を終了したい場合：'E' • 画面表示と帳票印刷をする場合：'△'
未使用	6(34)	byte	XMAP_RSV2	すべて$(00)_{16}$
マッピングオプション大分類	4(40)	int	XMAP_MAPOPT1	• マッピングオプションを指定する場合：3 • マッピングオプションを指定しない場合：0
マッピングオプション小分類	4(44)	int	XMAP_MAPOPT2	• マッピングオプション大分類に 3 を指定した場合：次の値を指定 マージ ：13 物理マップだけ：14 論理マップだけ：15 • マッピングオプションを指定しない場合：0
未使用	16(48)	byte	XMAP_RSV3	すべて$(00)_{16}$
URL の長さ	2(64)	short	XMAP_URLLNG	• 次回の呼び出し URL を指定する場合：128 • URL の指定を省略する場合：0
論理マップ領域の長さ	2(66)	short	XMAP_LSGLNG	論理マップ領域の長さ（0～32,000） • 業務終了時：0
未使用	60(68)	byte	XMAP_RSV4	すべて$(00)_{16}$

データ項目名	データ長 (位置)	データ型	データ名	設定値
次回の呼び出し URL	128(128)	String	XMAP_URL	次回の呼び出し先 URL※2 • 業務終了時にブラウザを閉じる場合：すべて空白，またはすべて$(00)_{16}$

注※1

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※2

左詰めで指定し，残りは空白を指定します。

8.5　画面のコーディング例

GUI 画面の画面属性，Java 言語用ツールで生成する XML 文書，および AP（業務サーブレットの業務処理）でのコーディングの関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：Java
- マップ名の文字数：6 文字
- マップ名：MAP001
- 実行環境：対話型

Java で AP を開発するには，C 言語でマップを生成して，マップを Java 言語用ツールで XML 文書に変更する必要があります。ドロー，ドローセットアップ，および Java 言語用ツールについては，マニュアル「XMAP3 開発ガイド」を参照してください。

8.5.1　出力テキストの表示

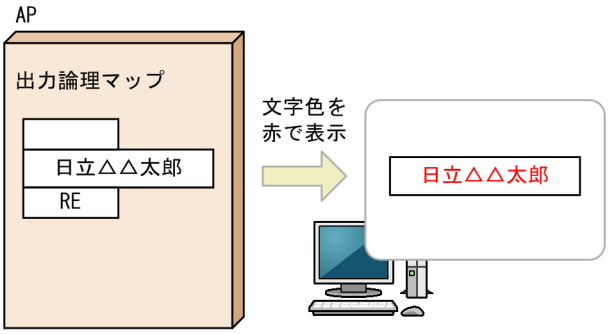
ここでは，AP で出力テキストを表示する場合のコーディング例について説明します。

出力テキストでは，動的変更を利用して，AP で文字と文字色を指定できます。

ドローで出力テキストボックスのダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，文字色，およびオブジェクトの表示・非表示の切り替えを AP で指定できます。AP で指定しない場合，出力テキストボックスダイアログで指定した属性で表示されます。

AP 実行時に，AP で指定した文字色での出力テキストの表示例を次の図に示します。

図 8-6　指定した文字色での出力テキストの表示例（Java）



（凡例）
△：半角の空白

表示したい文字列「日立 太郎」と赤色を表示する修飾名「RE」を出力論理マップに代入すると，AP 実行時には「日立 太郎」が赤色で画面に表示されます。

出力テキストボックスのダイアログで「動的変更（AP から表示属性を変更する）」を選択して作成した論理マップと動的変更テーブルを Java 言語用ツールで変換すると，出力データ用 XML 文書と動的変更用 XML 文書が生成されます。XML 文書に文字配列と表示色を設定する AP のコーディング例を次に示します。

XML 文書に文字配列と表示色を設定する AP のコーディング例

```
ModTbl modTbl;
LogicalMap logicalMap0;
 : //動的変更用クラスのインスタンスをmodTblに代入する
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

//テキストの代入
logicalMap0.setDataString("MAP001_FIELD0001_0", "日立　太郎");
byte[] inAttr1 = modTbl.getDataByteArray("xmap_in_attr1");
logicalMap0.setDataByteArray("MAP001_FIELD0001_A", inAttr1);
```

8.5.2　入出力テキストの表示

ここでは，AP で入出力テキストを表示する場合のコーディング例について説明します。

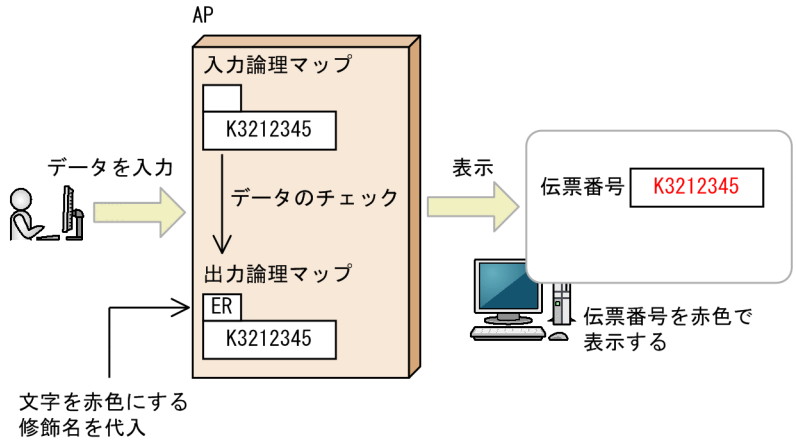
入出力テキストでは，画面に入力されたデータを AP で受け取ったり，AP で文字，文字色などを指定したりできます。出力テキストと同様に特定のデータだけを変更できます。

テキストの表示については，「8.5.1　出力テキストの表示」を，データ有無コードについては，「16.　論理マップ生成規則とマッピング規則」を参照してください。

ドローで入出力テキストボックスのダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，文字色，背景色，表示方法，遷移条件など，入出力テキストの属性を AP で変更できます。AP で指定しない場合，入出力テキストボックスのダイアログで指定した属性で表示されます。

AP 実行時に，データ有無コードを使用して特定のデータだけを変更する場合の入出力テキストの表示例を次の図に示します。

図 8-7　データ有無コードを使用した入出力テキストの表示例（Java）



画面に入力されたデータを AP でチェックして，再び画面に表示します。画面に表示するときには，品名コードと数量はデータを変更しないで表示するので，データ有無コードを使用します。値引のデータだけは，データを変更して赤色で表示します。

定義した論理マップと動的変更テーブルを Java 言語用ツールで変換すると，入力データ用 XML 文書，出力データ用 XML 文書，および動的変更用 XML 文書が生成されます。XML 文書に文字配列と表示色を設定する AP のコーディング例を次に示します。

XML 文書に文字配列と表示色を設定する AP のコーディング例

1. 通常の表示をする

   表示属性は画面定義の標準属性を適用します。

```
ComTbl comTbl;
ModTbl modTbl;
LogicalMap logicalMap0;
 : //通信制御用クラスのインスタンスをcomTblに代入する
 : //動的変更用クラスのインスタンスをmodTblに代入する
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

//マッピングモードを設定する
comTbl.setDataInteger("XMAP_MAPOPT1", 0);
comTbl.setDataInteger("XMAP_MAPOPT2", 0);

//nodataには0番目の値がデータ有無コードであるような長さ1のバイト配列が代入される
byte[] nodata = modTbl.getDataByteArray("XMAP_NODATA");

logicalMap0.setDataByteArray("MAP001_FIELD0001_A", nodata);
logicalMap0.setDataString("MAP001_FIELD0001_0", "K3212345");
```

2. 文字色を赤に変更して表示する

修飾名「ER」は標準の変更属性を使用します。この赤色をほかの色にするためには，あらかじめドローセットアップでの設定が必要です。

```
ComTbl comTbl;
ModTbl modTbl;
LogicalMap logicalMap0;
 : //通信制御用クラスのインスタンスをcomTblに代入する
 : //動的変更用クラスのインスタンスをmodTblに代入する
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

//マッピングモードを設定する
comTbl.setDataInteger("XMAP_MAPOPT1", 3);
comTbl.setDataInteger("XMAP_MAPOPT2", 15);

//nodataには0番目の値がデータ有無コードであるような長さ1のバイト配列が代入される
byte[] nodata = modTbl.getDataByteArray("XMAP_NODATA");

//表示データを変更しない指示
byte[] inAttr1 = modTbl.getDataByteArray("xmap_in_attr1");
logicalMap0.setDataByteArray("MAP001_FIELD0001_A", inAttr1);
```

注

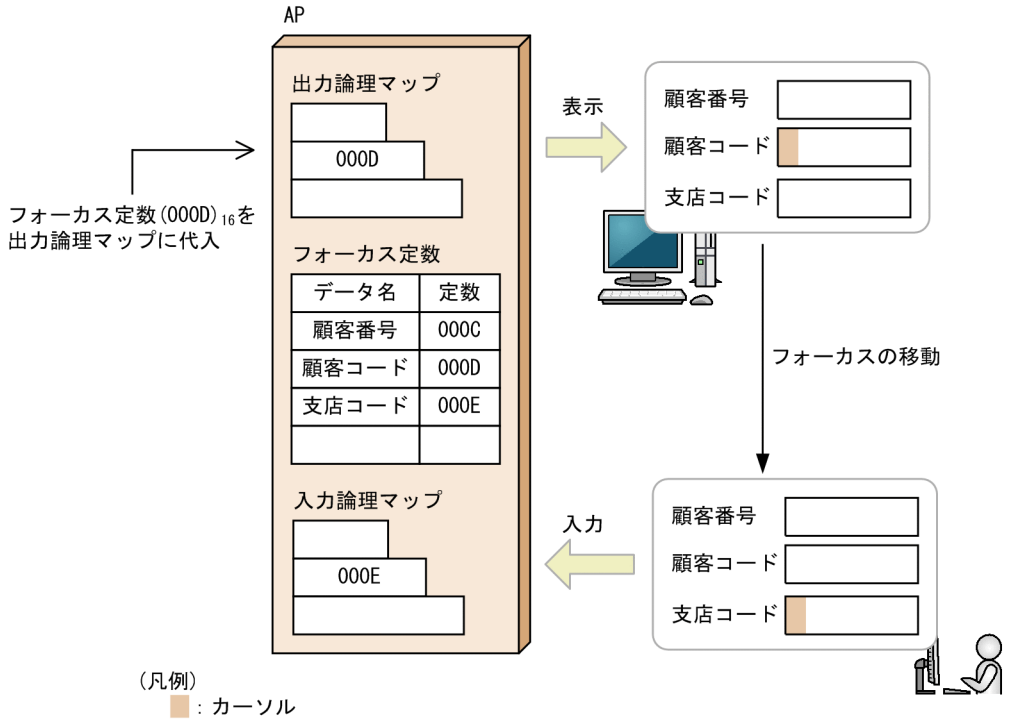
マッピングモードが「論理マップだけ」，表示形態が「一部上書」の場合，画面を表示するときに，データ有無コードや定義されていない修飾名を制御項目に代入しても，入出力テキストボックスに定義した属性には戻りません。入出力テキストボックスに定義した属性に戻したい場合は，ドローセットアップですべての属性を「変更なし」に設定した修飾名を設定し，制御項目名に代入してください。

8.5.3　フォーカスの位置づけ

画面を表示したときの初期フォーカス位置を AP で指定できます。AP でフォーカス位置を指定しないと，フォーカス位置はドローで定義した初期フォーカス位置に指定されます。ここでは，入出力テキストにフォーカスを位置づけて表示する場合のコーディング例について説明します。

AP 実行時のフォーカスの位置づけを次の図に示します。

図 8-8　フォーカスの位置づけ（Java）



顧客コードに初期フォーカス位置を指定するために，出力論理マップにフォーカス定数を代入します。顧客コードにフォーカスがある状態で，画面が表示されます。フォーカスを支店コードに移動すると，入力論理マップにフォーカス定数が格納されます。

定義した論理マップ，動的変更テーブル，および定数を Java 言語用ツールで変換すると，入力データ用 XML 文書，出力データ用 XML 文書，動的変更用 XML 文書，および定数用 XML 文書が生成されます。XML 文書から定数を取得する AP のコーディング例を次に示します。

XML 文書に文字配列と表示色を設定する AP のコーディング例

```
ConstValue constValue;
LogicalMap logicalMap0;
 : //定数用クラスのインスタンスをconstValueに代入する
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

byte[] focus = constValue.getDataByteArray("MAP001_FIELD0001_T");
logicalMap0.setDataByteArray("MAP001_OUTCURS_LOCO", focus);
```

8.6　帳票のコーディング例

帳票の帳票属性，Java 言語用ツールで生成する XML 文書，および AP（業務サーブレットの業務処理）でのコーディングの関連について説明します。ここでは次の条件を前提として説明します。

- ドローセットアップ：標準値
- AP 開発言語：COBOL
- マップ名の文字数：6 文字
- マップ名：MAP001
- 実行環境：対話型

Java で AP を開発するには，C 言語でマップを生成して，マップを Java 言語用ツールで XML 文書に変更する必要があります。ドロー，ドローセットアップ，および Java 言語用ツールについては，マニュアル「XMAP3 開発ガイド」を参照してください。

8.6.1　印刷枚数の指定を変更する

ページプリンタを使用する場合，印刷枚数を AP で指定できます。ここでは，印刷枚数を AP で指定する場合の論理マップおよび AP の関連について説明します。なお，PDF ファイル出力の場合，指定した印刷枚数は無効となります。

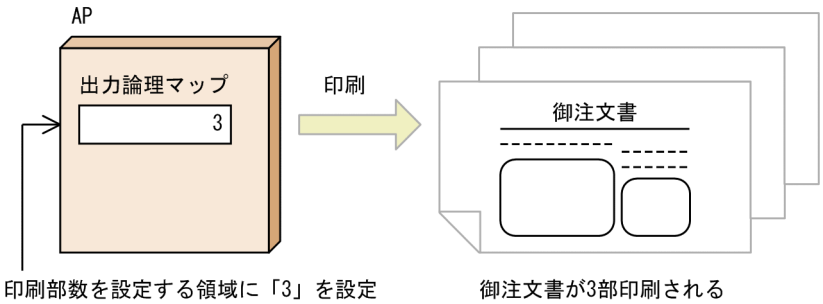
印刷枚数は次のどちらかの方法で指定します。

- 帳票属性ダイアログ
  各マップの帳票属性ダイアログの「印刷部数」で指定します。
- AP
  各マップの帳票属性ダイアログの「印刷部数を AP で変更する」を選択して，動的変更を利用して AP で指定します。

帳票属性ダイアログの「印刷部数」と「印刷部数を AP で変更する」の両方を選択した場合，AP での指定が優先されます。

AP で印刷枚数を指定して帳票をコピー印刷する例を次の図に示します。

図 8–9　AP で印刷枚数を指定して帳票をコピー印刷する例（Java）



定義した論理マップを Java 言語用ツールで変換すると，出力データ用 XML 文書が生成されます。XML 文書に印刷枚数を設定する AP のコーディング例を次に示します。

XML 文書に印刷枚数を設定する AP のコーディング例

```
LogicalMap logicalMap0;
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

logicalMap0.setDataShort("MAP001_COPIES0", 3);      ・・・印刷枚数に「3」を代入
```

8.6.2　出力フィールドの表示を変更する

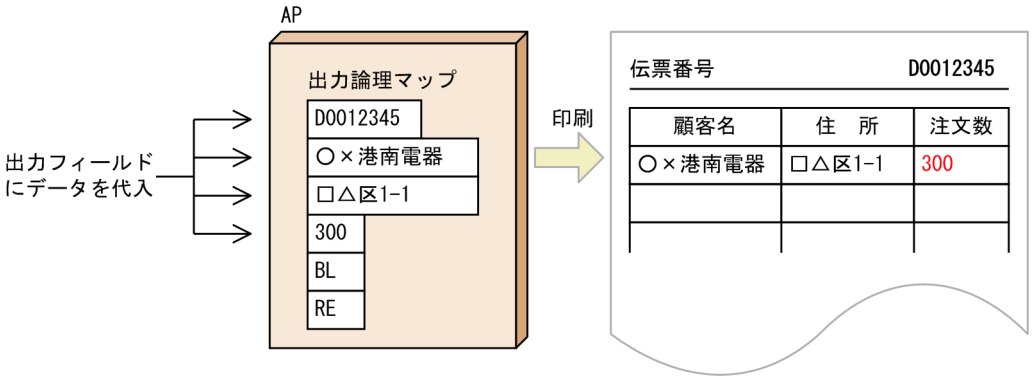
ここでは，AP で出力フィールドを表示する場合のコーディング例について説明します。

出力フィールドでは，動的変更を利用して，AP で文字色や書式などを指定できます。

ドローで出力フィールドのダイアログの「動的変更（AP から表示属性を変更する)」を選択すると，動的変更を利用して，文字色，文字の書体，文字の強調，網掛けなどを AP で指定できます。AP で指定しない場合，出力フィールドボックスダイアログで指定した属性が表示されます。

文字色と文字の書体を変更する場合の出力フィールドの表示例を次の図に示します。

図 8-10　出力フィールドの表示（Java）



出力フィールドのデータとして，「伝票番号」，「顧客名」，「住所」，「注文数」を AP で代入します。「伝票番号」の値である「D0012345」の制御項目に太字を表示する修飾名「BL」を，「注文数」の値である「300」の制御項目に赤字を表示する修飾名「RE」を代入します。AP を実行すると，「伝票番号」の文字の書体を「太字」で，「注文数」の文字色を赤で出力されます。

定義した論理マップと動的変更テーブルを Java 言語用ツールで変換すると，出力データ用 XML 文書と動的変更用 XML 文書が生成されます。XML 文書に文字配列と表示色を設定する AP のコーディング例を次に示します。

XML 文書に文字配列と表示色を設定する AP のコーディング例

```
ModTbl modTbl;
LogicalMap logicalMap0;
 : //動的変更用クラスのインスタンスをmodTblに代入する
 : //出力データ用クラスのインスタンスをlogicalMap0に代入する

logicalMap0.setDataString("MAP001_FIELD0001_0", "D0012345");
logicalMap0.setDataString("MAP001_FIELD0002_0", "○×港南電器");
logicalMap0.setDataString("MAP001_FIELD0003_0", "□△区1-1");
logicalMap0.setDataString("MAP001_FIELD0004_0", "300");
byte[] inBl = modTbl.getDataByteArray("xmap_in_bl");
logicalMap0.setDataByteArray("MAP001_FIELD0001_A", inBl);
byte[] inAttr1 = modTbl.getDataByteArray("xmap_in_attr1");
logicalMap0.setDataByteArray("MAP001_FIELD0004_A", inAttr1);
```

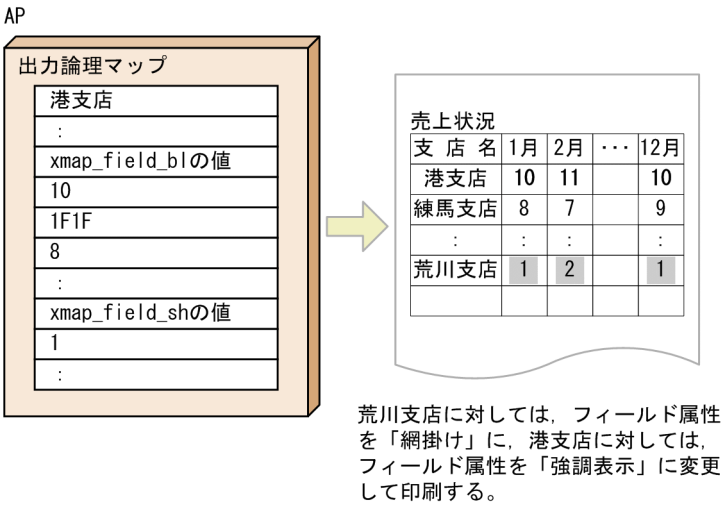
8.6.3　けい線の属性を変更する

グラフィック帳票では，AP でけい線の属性を変更して出力できます。ここでは，AP でけい線の属性を変更するコーディング例について説明します。

ドローでけい線のダイアログの「動的変更（AP から表示属性を変更する）」を選択すると，動的変更を利用して，AP でけい線の種類や太さを指定できます。AP で指定しない場合は，けい線のダイアログで指定した属性で出力されます。

けい線の属性を変更して出力する例を次の図に示します。

図 8-11　けい線の属性を変更して出力する例



制御項目に破線や二重線を示す修飾名を代入します。印字データがある行間のけい線は「細い破線」で，最終行は「二重線」で印刷されます。

定義した論理マップと動的変更テーブルを Java 言語用ツールで変換すると，出力データ用 XML 文書と動的変更用 XML 文書が生成されます。XML 文書にけい線の属性を設定する AP のコーディング例を次に示します。

けい線の属性を設定する AP のコーディング例

```
ModTbl modTbl;
LogicalMap logicalMap0;
: //動的変更用クラスのインスタンスをmodTblに代入する
: //出力データ用クラスのインスタンスをlogicalMap0に代入する

byte[] fieldBl = modTbl.getDataByteArray(”xmap_field_bl”);
byte[] fieldSh = modTbl.getDataByteArray(”xmap_field_sh”);

//フィールド属性の動的変更
for (int i = 0; i < 12; i++) {
 //港支店：標準属性→太字属性 (xmap_field_bl)
 logicalMap0.setDataByteArray(”MDC1PC_DATA_A”, fieldBl, 13 * i);
  :
 //荒川支店：標準属性→網掛け属性 (xmap_field_sh)
 logicalMap0.setDataByteArray(”MDC1PC_DATA_A”, fieldSh, 13 * i + 5);
  :
}
```

8.7　業務サーブレットのコンパイルと EAR ファイルの生成

生成した業務サーブレットのソースプログラムをコンパイルして，EAR ファイルを生成します。プロジェクトを作成，ビルドして EAR ファイルを生成するときは，MyEclipse を使用します。

8.7.1　開発環境に準備しておくフォルダおよびファイル

XMAP3/Web for Cosminexus では，EAR ファイルを作成する前に，実行環境と同じフォルダ構成のフォルダを開発環境に作成することを推奨しています。開発環境と実行環境とで同じフォルダ構成にしておくことで，フォルダ構成を変更することなく実行環境にデプロイできます。

なお，MyEclipse を使用する場合は，次に示すフォルダ構成が自動的に生成されるので，フォルダ構成を作成する必要はありません。

推奨するフォルダ構成および格納ファイルを次に示します。

推奨するフォルダ構成

```
+-- WebApp※-----------起動HTML（x3xwbfrm.htm）および
|    |                起動HTML用スクリプトファイル（x3xwbfrm.js）
|    |                を格納
|    +-- Data※--------物理マップ，ポップアップメニューファイル，
|    |                グラフィックデータ，およびFAX宛先ファイル
|    |                を格納
|    +-- WEB-INF------web.xmlファイルを格納
|        +-- classes--業務サーブレットおよび
|        |            環境管理ファイル（xmap3.properties）を格納
|        +-- lib------XMAP3実行クラスライブラリ（xmap3server.jar）
|        |            を格納
|        +-- XMAP3----通信制御用XML文書，動的変更用XML文書，
|                     入力データ用XML文書，出力データ用XML文書，
|                     および定数用XML文書を格納
```

注※
フォルダ名には，任意の名称（特殊文字を除く半角英数字）を付けてください。

次に，フォルダに格納するファイルについて説明します。

起動 HTML（x3xwbfrm.htm）

Web アプリケーションを実行するための HTML ファイルです。XMAP3/Web for Cosminexus で提供する起動 HTML をカスタマイズして使用できます。

起動 HTML 用スクリプトファイル（x3xwbfrm.js）

起動 HTML の各種情報の定義で使用するスクリプトファイルです。

物理マップ，ポップアップメニューファイル，グラフィックデータ，および FAX 宛先ファイル

- 物理マップは，画面表示・帳票印刷の実行時に使用するマッピング情報を格納したファイルです。
- ポップアップメニューファイルは，ポップアップテキストのメニュー情報を定義するファイルです。ポップアップメニューエディタで作成します。
- グラフィックデータは，画面表示・帳票印刷に必要なグラフィックデータを格納したファイルです。このファイルは，必要に応じて作成してください。
- FAX 宛先ファイルは，送信先の FAX 番号を記述したファイルです。

web.xml ファイル

web.xml ファイルは，Web コンテナのコンテキストを設定する XML ファイルです。業務サーブレットを呼び出す URL の情報を定義します。

業務サーブレットおよび環境管理ファイル（xmap3.properties）

業務サーブレットは，Web サーバ側で動作するユーザの業務用 Java プログラムです。

環境管理ファイルは，XML 文書などの格納先や XMAP3 が提供する Java インタフェースの保守情報（ログ）の出力先などを指定するファイルです。

XMAP3 実行クラスライブラリ（xmap3server.jar）

XMAP3 実行クラスライブラリは，XMAP3 の提供するクラスライブラリ（XMAP3 実行クラスライブラリ）のファイルです。

通信制御用 XML 文書，動的変更用 XML 文書，入力データ用 XML 文書，出力データ用 XML 文書，および定数用 XML 文書

- 通信制御用 XML 文書は，XMAP3 で提供する実行クラスライブラリを参照するための XML 形式のファイルです。
- 動的変更用 XML 文書は，動的変更テーブルを XML 文書に変換した XML 形式のファイルです。
- 入力データ用 XML 文書は，入力論理マップを XML 文書に変換した XML 形式のファイルです。
- 出力データ用 XML 文書は，出力論理マップを XML 文書に変換した XML 形式のファイルです。
- 定数用 XML 文書は，定数ファイルを XML 文書に変換した XML 形式のファイルです。

8.7.2　MyEclipse での EAR ファイルの生成手順

Java アプリケーションを作成するため，MyEclipse での EAR ファイルを生成する手順について説明します。MyEclipse の操作については，Cosminexus アプリケーションサーバのマニュアルを参照してください。

MyEclipse での EAR ファイルの生成手順を次の図に示します。

図 8-12　MyEclipse での EAR ファイルの生成手順

(1)　MyEclipse のセットアップ

日本語版 MyEclipse ポータルサイトから，MyEclipse の前提となる Eclipse アーカイブファイルをダウンロードします。ダウンロードしたアーカイブファイルを解凍し，MyEclipse を利用できるようにします。次に，プラグインを利用できるように環境設定ファイルにプロパティを追加し，Web サーバを起動できるようにリモート管理機能でログインします。

MyEclipse のセットアップ手順については，Cosminexus アプリケーションサーバのマニュアルを参照してください。

(2)　Web プロジェクトおよびパッケージの作成

Web プロジェクトを作成する前に，次に示す web.xml ファイルを編集しておきます。

XMAP3 インストールフォルダ¥Web for Cosminexus¥Sample¥JAVA¥web.xml

1 行目にある「encoding="MS932"」を「encoding="UTF-8"」に変更

その後，MyEclipse のメニューからプロジェクトを選択し，Web プロジェクトを作成します。作成したプロジェクトから，パッケージを作成します。

(3)　EAR ファイルの作成

EAR ファイルを作成するには，取り込む Web プロジェクトを選択します。

MyEclipse のメニューからエンタープライズ Java プロジェクトを作成します。このとき，EAR ファイルに取り込む Web プロジェクトとして，ここまでの手順で作成した Web プロジェクトを選択します。

(4)　MyEclipse で実行するサーバの選択

サーバに Cosminexus を設定します。

(5)　J2EE アプリケーションのデプロイ

MyEclipse から EAR ファイルをデプロイし，J2EE アプリケーションが正常に起動したことを確認します。

(6)　Web サーバ（Cosminexus HTTP Server（Hitachi Web Server）または IIS）の起動

業務を開始する前に，Web サーバ（Cosminexus HTTP Server（Hitachi Web Server）または IIS）を起動します。

なお，Web サーバと J2EE サーバの両方を起動していないと，XMAP3/Web for Cosminexus は実行できません。Web サーバを起動するときは，J2EE サーバも起動してください。

8.7.3　Eclipse および JBuilder での WAR ファイルの生成手順

Eclipse および JBuilder を使用して，サーブレットエンジンモードの場合にデプロイする，WAR ファイルの生成手順について説明します。

J2EE サーバモードで使用する場合は，Eclipse のコマンドで WAR ファイルを EAR ファイルに格納し，EAR ファイルをデプロイする必要があります。Eclipse のコマンドについては，Cosminexus アプリケーションサーバのマニュアルを参照してください。

(1)　Eclipse での WAR ファイル生成手順

Eclipse での WAR ファイル生成手順を次に示します。

1. プロジェクトの作成

   Eclipse のメニューから，［ファイル］－［新規］－［プロジェクト］を開き，「Java プロジェクト」を選択し，任意の名称でプロジェクトを新規作成します。

   新規 Java プロジェクト作成ウィザードで，利用している Cosminexus の JDK のバージョンを指定します（5.0 を選択することを推奨します）。

2. ライブラリの取り込みと出力デフォルトフォルダの設定

   新規 Java プロジェクト作成ウィザードの「Java 設定」画面で，［ライブラリー］タブを選択し，［外部 JAR の追加］ボタンをクリックして次に示す外部 JAR を取り込みます。

   - *Cosminexus インストールフォルダ*¥client¥lib¥j2ee-javax.jar
   - *XMAP3 インストールフォルダ*¥Web for Cosminexus¥Lib¥xmap3server.jar（XMAP3 実行クラスライブラリ）

     「デフォルト出力フォルダー」には，次に示すように WEB-INF の下に classes フォルダを作成して指定します。

   /XmapWebApp/WEB-INF/classes

3. プロジェクトへのソースプログラムの追加

   ［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「新規」－「ソース・フォルダー」を選択し，ソースをインポートするためのフォルダを作成します。

   その後，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「インポート」を選択し，「ファイル・システム」画面で業務サーブレットのソース

プログラム※を追加します。このとき，インポート対象は Java ソースの上位レベルのノード名から選択してください。

ソースフォルダへのインポート後，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「インポート」を選択し，残りのソース以外のフォルダ，ファイルなどをノード直下にインポートします。

注※

ソースプログラムは，必ず 2 階層下の［パッケージ名］フォルダ下に格納してください。

4. Java コンパイラの設定

コンパイルオプションの「エラー/警告」を設定しない場合，警告レベルのエラーメッセージを表示したくないときは，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「プロパティー」を選択します。「Web アプリケーション名」のプロパティダイアログの「Java コンパイラー」－「エラー/警告」から次の設定をします。

- ［プロジェクト特定設定を使用可能にする］のボックスをチェックする
- ［潜在的なプログラミングの問題］－［serialVersionUID なしのシリアライズ可能クラス］を［無視］にする
- ［J2SE 5.0 オプション］－［未検査の総称型操作］を［無視］にする

上記の設定に加え，「Web アプリケーション名」のプロパティダイアログの［Java コンパイラー］－［ビルド］で次の設定をします。

- ［出力フォルダー］－［プロジェクトのクリーン時に出力フォルダーを消しこむ］のボックスのチェックを外す

  この設定をしないと，出力フォルダ下のファイルなどが削除されてしまいます。必要なデータは必ずバックアップしておいてください。

5. WAR ファイル用 DD の作成

Web アプリケーション用の DD（web.xml）を作成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに Web アプリケーション用の DD（web.xml）をサンプルとして提供しています。ユーザ環境に合わせてカスタマイズしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥JAVA

なお，このファイルをインポートする前にあらかじめ編集しておくこともできます。

6. Java プログラムのコンパイル・ビルド

Eclipse のメニューから，［プロジェクト］－［プロジェクトのビルド］で Java プログラムをコンパイル・ビルドします。

なお，デフォルトの設定では「自動的にビルド」となっているため，ソースプログラムをインポートしたときに自動的にコンパイルされます。

7. WAR ファイルの生成

［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「エクスポート」を選択します。「エクスポート」ダイアログで「アーカイブ・ファイル」を選択し，WAR ファイルを生成します。

WAR ファイルを生成する前に，「エクスポート」－「オプション」の次の設定がチェックされていることを確認してください。

- ［ZIP フォーマットで保管］
- ［選択したディレクトリーのみ作成］
- ［ファイルの内容を圧縮］

Eclipse を使用してプロジェクトを作成，ビルドしたときに作成されるフォルダ構成は次のようになります。

```
プロジェクト名
 └Webアプリケーション名
    └WEB-INF
       └classes
          └パッケージ名
```

(2)　JBuilder での WAR ファイル生成手順

JBuilder での WAR ファイル生成手順を次に示します。

1. プロジェクトの作成

JBuilder のメニューから，［ファイル］－［新規プロジェクト］を開き，任意の名称でプロジェクトを新規作成します。

2. Web アプリケーションの作成

JBuilder のメニューから，［ファイル］－［新規］を開き，［Web］タブの［Web アプリケーション］を選択します。Web アプリケーションウィザードダイアログを使用して Web アプリケーションをプロジェクトに追加します。また，作成した Web アプリケーションのプロパティを設定します。

3. プロジェクトへのソースプログラムの追加

プロジェクトペインの Web アプリケーションノードを右クリックし，コンテキストメニューから「ファイル／パッケージ／クラスの追加」を選択します。「プロジェクト名」に追加ダイアログで，編集した業務サーブレットのソースプログラム※を追加します。

注※

ソースプログラムは，［パッケージ名］フォルダ下に配置しておいてください。

4. 必須ライブラリの定義

JBuilder のメニューから，［ツール］－［ライブラリ設定］を選択し，表示されたライブラリ設定ダイアログで［新規］ボタンをクリックします。

表示された新規ライブラリウィザードダイアログの［新規］ボタンで［ライブラリパス］にバージョンに合った servlet.jar ファイルを追加し，［場所］には「ユーザーホーム」を指定して任意の名称でライブラリを定義します。

5. 必須ライブラリの取り込み

JBuilder のメニューから，［プロジェクト］－［プロジェクトプロパティ］を選択します。［パス］タブ内の［必須ライブラリ］タブを選択し，［追加］ボタンで，次に示す「ユーザーホーム」内のライブラリを追加してください。

- XMAP3 実行クラスライブラリ（xmap3server.jar）
- 4.で定義した javax.servlet および javax.servlet.http パッケージを含むライブラリ（servlet.jar）

6. WAR ファイル用の DD の作成

「WebApp 配布ディスクリプタエディタ」を使用して，Web アプリケーション用の DD（web.xml）を作成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに Web アプリケーション用の DD（web.xml）をサンプルとして提供しています。ユーザ環境に合わせてカスタマイズしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥JAVA

7. IDE オプションの設定

JBuilder のメニューから，［ツール］－［IDE オプション］を開き，ファイルタイプ一覧中のファイルタイプに pmp を関連づけます。

次に Web アプリケーションのプロパティ中の［WebApp］タブ－［含まれるファイルタイプ］で pmp に関連づけたファイルタイプをチェックします。

8. WAR ファイルの生成（プロジェクトのビルド）

プロジェクトペインの Web アプリケーションノードを右クリックし，コンテキストメニューから「プロパティ」を選択します。「Web アプリケーション名」のプロパティダイアログの［WebApp］タブで，［ビルド］プルダウンメニューの［プロジェクトのビルド時のみ］を選択します。

また，WAR ファイルの出力先などを指定します。

JBuilder のメニューから，［プロジェクト］－［プロジェクトのメイク］を選択してプロジェクトをビルドします。

JBuilder を使用してプロジェクトを作成，ビルドしたときに作成されるフォルダ構成は次のようになります。

9 XMAP3 Cosminexus 連携（COBOL）

この章では，Cosminexus ベースのシステム，Cosminexus と OpenTP1 サーバで連携するシステム，および Cosminexus と VOS3 DCCM3 で連携するシステムを開発するための COBOL プログラムの生成手順について説明します。

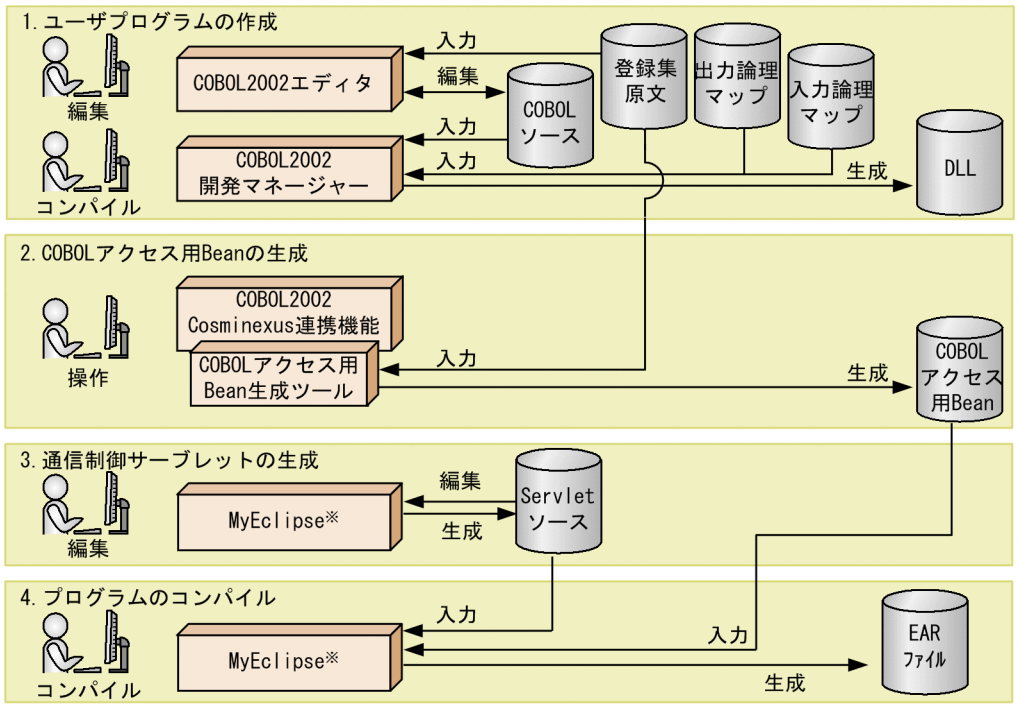
9.1　AP 開発の概要

ユーザが作成する AP は次の三つから構成されています。

- 通信制御サーブレット
- COBOL アクセス用 Bean
- ユーザプログラム

AP を開発する手順を次に示します。

図 9-1　AP 開発の手順



注※　Eclipse，JBuilderも使用できます。

1. ユーザプログラムの作成

XMAP3 Cosminexus 連携機能が提供するデータ送受信用の登録集原文を取り込んだユーザプログラムを生成します（Cosminexus ベースの場合は DLL 形式です。OpenTP1 サーバとの連携の場合は EXE 形式で，入力用と出力用にカスタマイズします）。詳細については，「9.3　ユーザプログラムの作成（Cosminexus ベースの場合）」，「9.4　ユーザプログラムの作成（OpenTP1 サーバ連携の場合）」，「9.5　ユーザプログラムの作成（DCCM3 連携の場合）」を，開発するシステムに合わせて参照してください。

2. COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）の生成

XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文を基に，COBOL2002 Cosminexus 連携機能（または TP1/COBOL adapter for Cosminexus）を使用して COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）を生成します。詳細については，「9.6.1　COBOL アクセス用 Bean の生成（Cosminexus ベースの場合）」，「9.7.1　TP1/COBOL アクセス用 Bean の生成（OpenTP1 サーバ連携の場合）」，「9.8.1　TP1/COBOL アクセス用 Bean の生成（DCCM3 連携の場合）」を開発するシステムに合わせて参照してください。

3. 通信制御サーブレットの生成

XMAP3 Cosminexus 連携機能の Servlet のソースファイルをカスタマイズし，通信制御サーブレットを生成します。詳細については，「9.6.2　通信制御サーブレットの作成（Cosminexus ベースの場合）」，「9.7.2　通信制御サーブレットの作成（OpenTP1 サーバ連携の場合）」，「9.8.2　通信制御サーブレットの作成（DCCM3 連携の場合）」を開発するシステムに合わせて参照してください。

4. プログラムのコンパイル

通信制御サーブレットをコンパイルして，EAR ファイルを作成します。

コンパイルの方法については，「9.10　Bean および通信制御サーブレットのコンパイルと EAR ファイルの生成」を参照してください。

なお，XMAP3 Cosminexus 連携機能で，COBOL を使用して AP を開発する場合は，次に示すシステムがあります。

- Cosminexus ベースのシステム
- Cosminexus と OpenTP1 サーバで連携するシステム
  - TP1/Client/J※1 を使ってバックエンドの OpenTP1（SPP）にサービスを要求

    既存の COBOL データの SPP に，RPC でサービスを要求する形態
  - TP1 Connector※2 を使ってバックエンドの OpenTP1（SPP）にサービスを要求

    既存の COBOL データの SPP に，性能と信頼性を重視した RPC（トランザクション処理）でサービスを要求する形態
- Cosminexus と VOS3 DCCM3 で連携するシステム

注※1

TP1/Client/J は，バックエンドの OpenTP1 で動作する SPP だけでなく，MHP とも通信できます。このマニュアルでは，SPP との通信を想定して説明します。

注※2

TP1 Connector を使用する場合は，別途 TP1/Client/J が必要になります。また，TP1 Connector を使用する場合は，追加の設定が必要になります。

9.1.1　開発環境の準備

Web アプリケーションを開発するための準備について説明します。

(1)　フォルダの準備

開発環境で，資源を格納するフォルダを準備しておくと，開発環境と同じフォルダ構成のまま，実行環境にデプロイできます。このため，開発環境は実行環境と同じフォルダ構成で準備してください。

開発環境の準備は，開発する Web アプリケーションのプロジェクト単位に実施する必要があります。

(2)　フォルダの構成

Web アプリケーションのプロジェクト単位に，資源（各ファイル）を格納する専用のフォルダを作成します。推奨するフォルダ構成例を次に示します。

```
+-- WebAppProject A（開発環境フォルダ）
|    +-- MAPS（マップ定義ファイル格納フォルダ）
|    +-- SEQ（遷移ファイル格納フォルダ）
|    +-- SOURCE（ソース格納フォルダ）
|    +-- WebApp（実行環境フォルダ）
|         +-- Data（データファイル格納フォルダ）
|         +-- WEB-INF（WEB-INFフォルダ）
|             +-- classes（classファイル格納フォルダ）
```

```
+-- WebAppProject B（開発環境フォルダ）
:
```

(3)　各フォルダの説明

各フォルダのフォルダ名および用途について次の表に示します。

表 9-1　各フォルダ名および用途

フォルダの種類	フォルダ名	用途	備考
開発環境フォルダ	任意	Web アプリケーション単位（プロジェクトごと）の開発資源をまとめて管理するためのルートフォルダです。	−
マップ定義ファイル格納フォルダ	任意※1	開発で作成する次のファイルを格納するフォルダです。 • マップ定義ファイル • 入力／出力論理マップファイル	−
遷移ファイル格納フォルダ	任意※1	開発で作成する画面遷移ファイルを格納するフォルダです。	−
ソース格納フォルダ	任意※1	COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean），通信制御サーブレットを格納するフォルダです。	−
実行環境フォルダ	任意※2	実行資源をまとめて管理するためのフォルダです。EAR ファイル作成時のフォルダとなります。	−
データファイル格納フォルダ	任意※2	実行時に必要な次のファイルを格納するフォルダです。 • 物理マップファイル • ポップアップメニューファイル • グラフィックデータ • FAX 宛先ファイル	起動 HTML の「DataPath」パラメタに，このフォルダのパスを指定してください。
WEB-INF フォルダ	WEB-INF（固定）	実行環境サブフォルダです。	−
class ファイル格納フォルダ	classes（固定）	ソース格納フォルダ内のソースをコンパイルして作成した class ファイルを格納するフォルダです。	−

（凡例）

−：該当しない。

注※1

フォルダ名には，特殊文字を除く半角英数字だけを使用してください。

注※2

フォルダ名には，次に示す特殊文字を除く半角英数字だけを使用してください。

「¥」，「/」，「:」，「,」，「;」，「*」，「?」，「"」，「<」，「>」，「¦」

(4)　フォルダ構成の注意事項

フォルダを作成するときには，次の内容に注意してください。

- 「WEB-INF フォルダ」および「class ファイル格納フォルダ（classes フォルダ）」は，フォルダ構成やフォルダ名を変更しないでください。また，Web アプリケーションの実行時に，これらのフォルダに必要なファイルが格納されてない場合，Web アプリケーションは動作しません。

(5)　フォルダ構成のポイント

フォルダを作成するときには，次の内容を参考にしてください。

- 画面や帳票の変更によってプログラムを修正したり更新したりするため，プロジェクト単位に次のフォルダを作成および配置することを推奨します。
  - 「マップ定義ファイル格納フォルダ」
  - 「遷移ファイル格納フォルダ」
  - 「ソース格納フォルダ」
- 「実行環境フォルダ」下のフォルダ構成は，開発時に生成するファイルをあとでデプロイするため，Web アプリケーションの開発時から，同じフォルダ構成で作業することを推奨します。
- 「データファイル格納フォルダ」は，起動 HTML で格納先を指定できるため，「実行環境フォルダ」以外のフォルダにも格納できます。

9.1.2　隠しフィールドへのマップ名の設定

OpenTP1 サーバ連携，および DCCM3 連携を利用する場合，ドローで画面属性を設定するときに，入力論理マップにマップ名（OpenTP1 サーバ連携の場合は先頭 6 バイト，DCCM3 連携の場合は先頭 8 バイト）が格納されるように，マップ定義ファイルに隠しフィールドを設定しておきます。

ドローについては，マニュアル「XMAP3 開発ガイド」を参照してください。

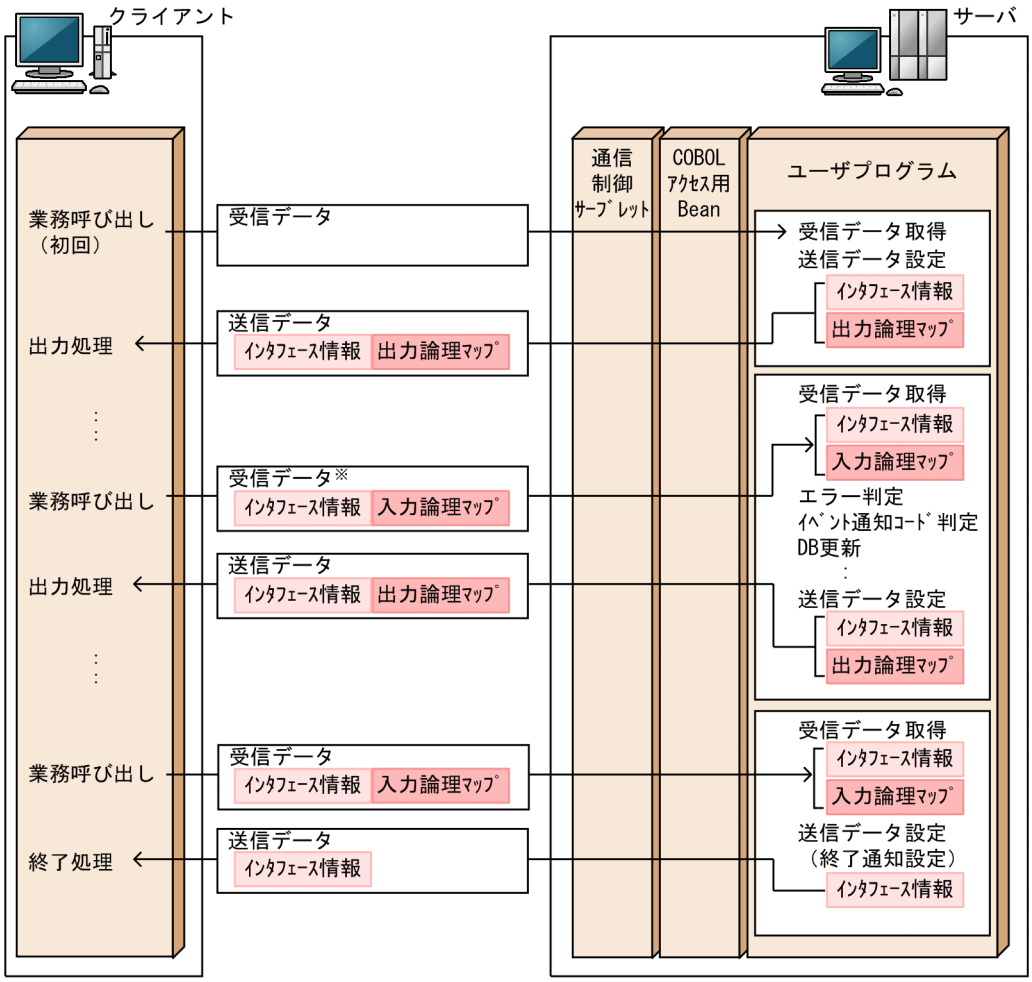
9.2　クライアントとサーバ間の処理の流れ

システム構成別に，クライアント側とサーバ側のプログラムとのやり取りについて，処理の流れを説明します。

9.2.1　Cosminexus ベース

Cosminexus ベースの場合の，クライアントとサーバ間の処理の流れを次の図に示します。

図 9-2　クライアントとサーバ間の処理の流れ（Cosminexus ベース）

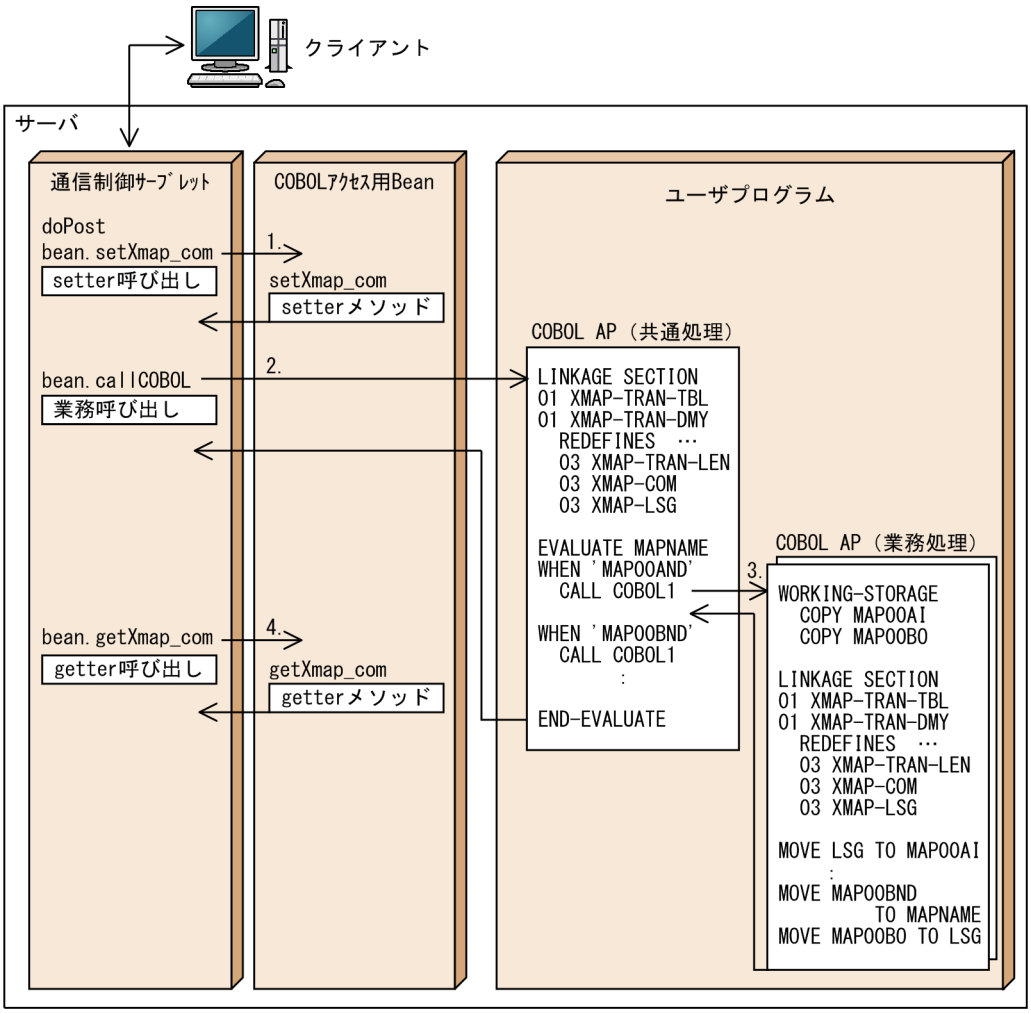


注※　帳票出力およびエラー発生の場合はインタフェース情報だけ設定されます。

XMAP3/Web for Cosminexus では，送受信データの長さの情報を定義するデータ長設定領域，インタフェース情報を定義する共通インタフェース領域，および論理マップ領域を合わせた定義を送受信データとして使用します。

クライアント側から業務呼び出しが発生したときの，サーバ側でのプログラム間の処理の流れを次の図に示します。

図 9-3　サーバ側でのプログラム間の処理の流れ（Cosminexus ベース）



1. 通信制御サーブレットから COBOL アクセス用 Bean の setter 呼び出し

   クライアント側からの呼び出しを受けて，通信制御サーブレットから受信データ設定用の setter メソッドを呼び出します。

2. 通信制御サーブレットから業務呼び出し

   通信制御サーブレットからユーザプログラム中の COBOL AP（共通処理）を呼び出します。

3. ユーザプログラム中の COBOL AP（共通処理）から COBOL AP（業務処理）の呼び出し

   ユーザプログラム中の COBOL AP（共通処理）では，受信データ中のマップ名を判定し，該当する COBOL AP（業務処理）を呼び出して業務処理を行います。

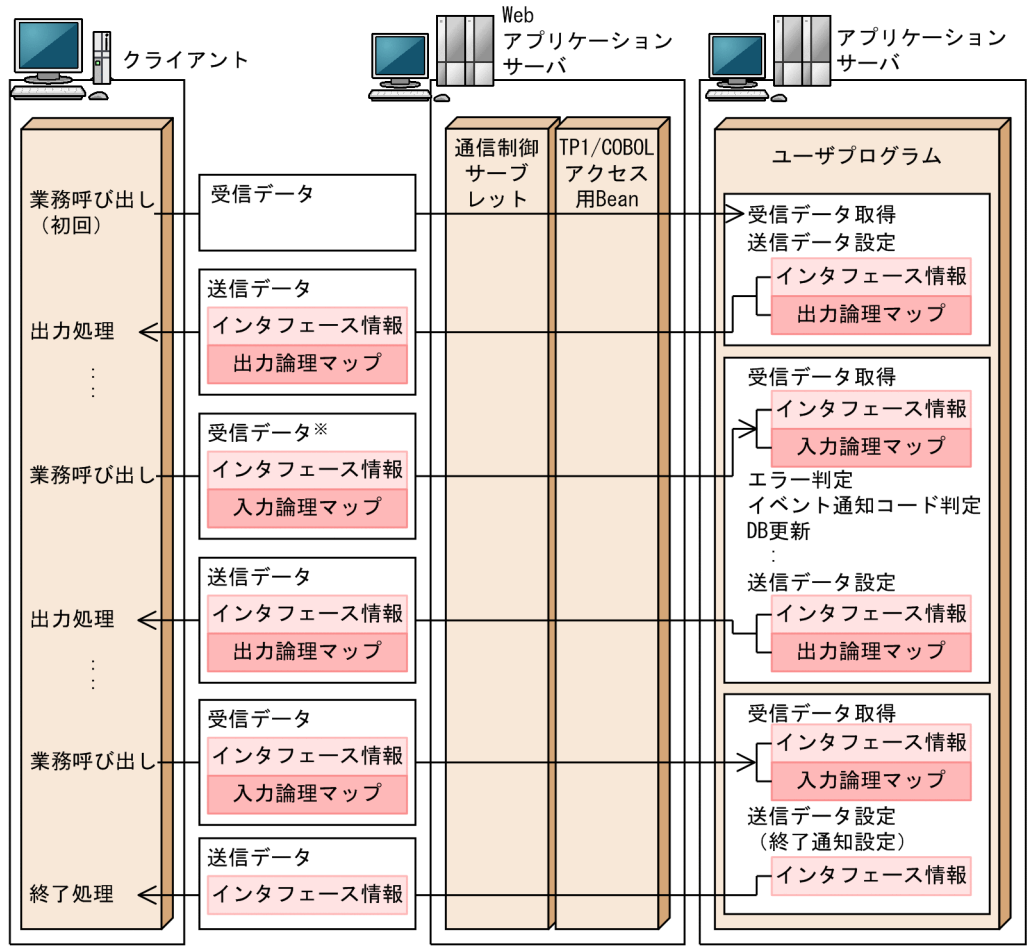
4. 通信制御サーブレットから COBOL アクセス用 Bean の getter 呼び出し

   通信制御サーブレットから，ユーザプログラムの業務処理で設定した送信データを取得するための getter メソッドを呼び出し，クライアント側へ取得した送信データを返送します。

9.2.2　OpenTP1 サーバ連携

OpenTP1 サーバ連携の場合の，クライアントとサーバ間の処理の流れを次の図に示します。

図 9-4　クライアントとサーバ間の処理の流れ（OpenTP1 連携）

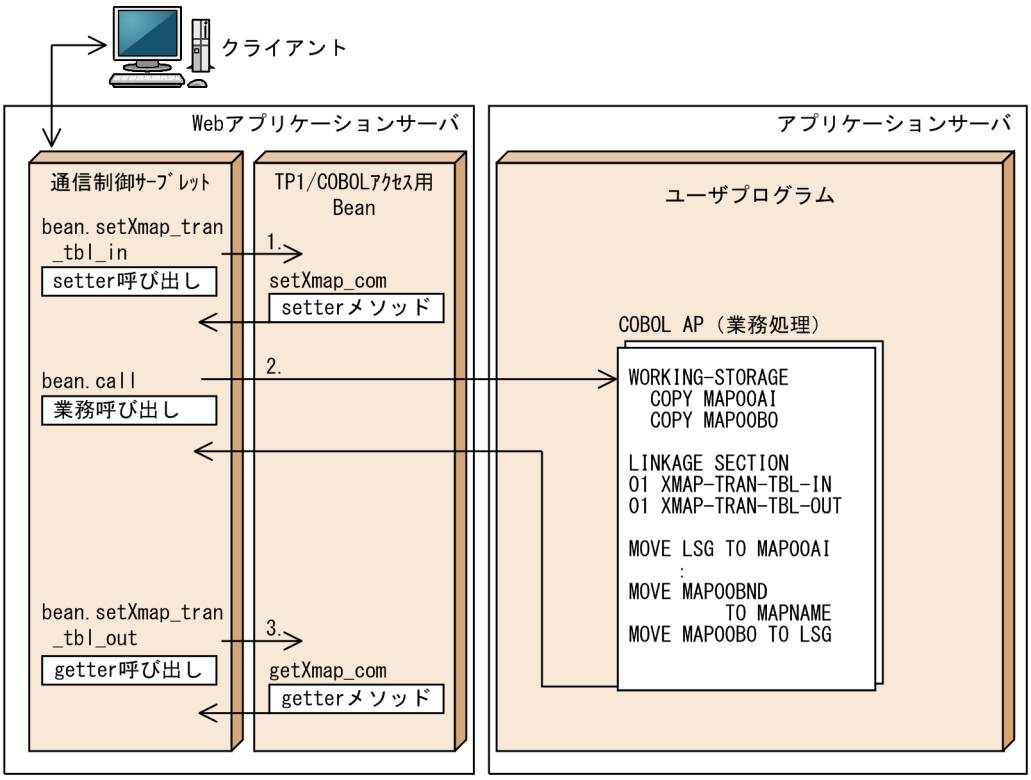


注※　帳票出力およびエラー発生の場合はインタフェース情報だけ設定されます。

XMAP3/Web for Cosminexus では，送受信データの長さの情報を定義するデータ長設定領域，インタフェース情報を定義する共通インタフェース領域，および論理マップ領域を合わせた定義を送受信データとして使用します。

クライアント側から業務呼び出しが発生したときの，サーバ側でのプログラム間の処理の流れを次の図に示します。

図 9-5　サーバ側でのプログラム間の処理の流れ（OpenTP1 連携）



1. 通信制御サーブレットから TP1/COBOL アクセス用 Bean の setter 呼び出し

   クライアント側からの呼び出しを受けて，通信制御サーブレットから受信データ設定用の setter メソッドを呼び出します。

2. 通信制御サーブレットから業務呼び出し

   通信制御サーブレットからユーザプログラム中の COBOL AP（業務処理）を呼び出します。

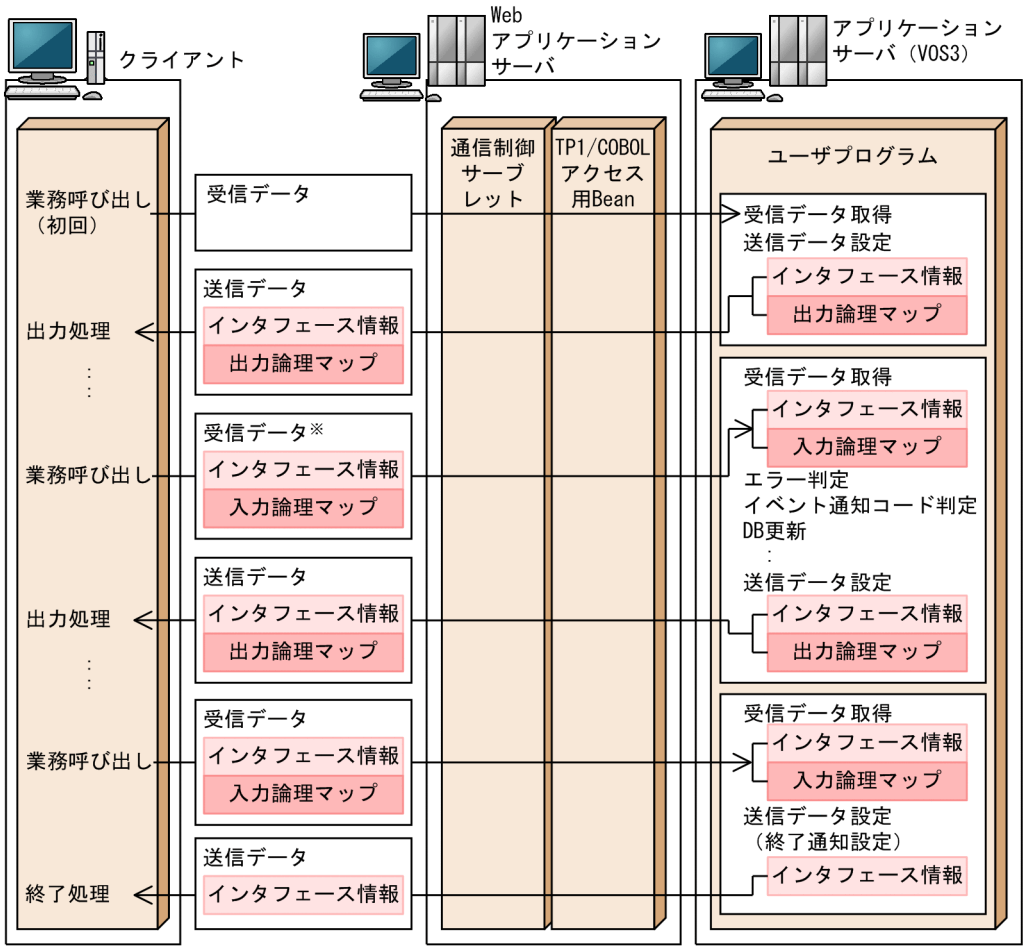
3. 通信制御サーブレットから TP1/COBOL アクセス用 Bean の getter 呼び出し

   通信制御サーブレットから，ユーザプログラムの業務処理で設定した送信データを取得するための getter メソッドを呼び出し，クライアント側へ取得した送信データを返送します。

9.2.3　DCCM3 連携

DCCM3 連携の場合の，クライアントとサーバ間の処理の流れを次の図に示します。

図 9-6　クライアントとサーバ間の処理の流れ（DCCM3 連携）

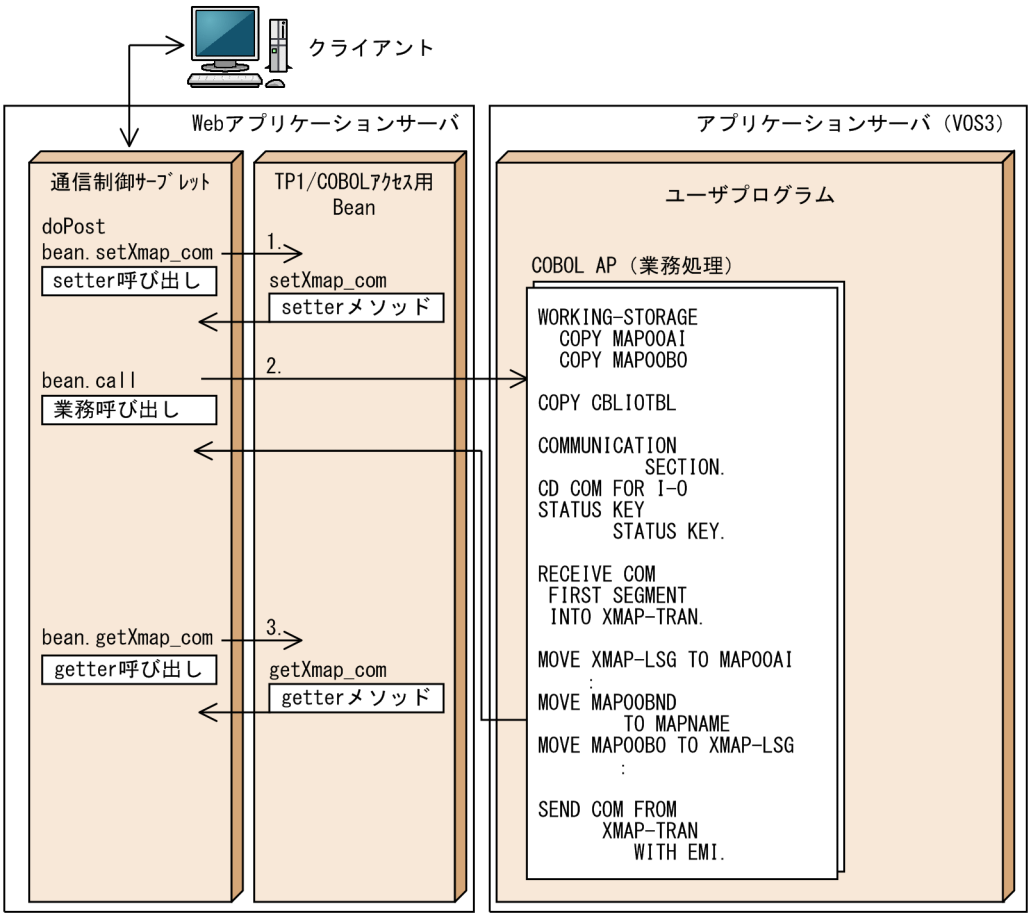


注※　帳票出力およびエラー発生の場合はインタフェース情報だけ設定されます。

XMAP3/Web for Cosminexus では，インタフェース情報を定義する共通インタフェース領域，および論理マップ領域を合わせた定義を送受信データとして使用します。

クライアント側から業務呼び出しが発生したときの，サーバ側でのプログラム間の処理の流れを次の図に示します。

図 9-7　サーバ側でのプログラム間の処理の流れ（DCCM3 連携）



1. 通信制御サーブレットから TP1/COBOL アクセス用 Bean の setter 呼び出し

   クライアント側からの呼び出しを受けて，通信制御サーブレットから受信データ設定用の setter メソッドを呼び出します。

2. 通信制御サーブレットから業務呼び出し

   通信制御サーブレットからユーザプログラム中の COBOL AP（業務処理）を呼び出します。

3. 通信制御サーブレットから TP1/COBOL アクセス用 Bean の getter 呼び出し

   通信制御サーブレットから，ユーザプログラムの業務処理で設定した送信データを取得するための getter メソッドを呼び出し，クライアント側へ取得した送信データを返送します。

9.3　ユーザプログラムの作成（Cosminexus ベースの場合）

ユーザプログラムは，Web ブラウザウィンドウ上に表示された画面とデータをやり取りしたり，帳票を印刷したりする，ユーザが作成する業務処理用の COBOL プログラムです。

XMAP3/Web for Cosminexus では，次に示すフォルダにサンプルプログラムを提供しています。ユーザプログラムを作成する上で参考にしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

ユーザプログラムでは，C/S 構成で使用していた XMAP3 システムと同様に論理マップデータを作成し，送信データに設定します。送信データには，論理マップデータのほかに仮想端末名や印刷完了通知，終了通知などのインタフェース情報や送信データの長さを設定します。

送信データに設定する論理マップは，クライアント側へ自動的にダウンロードされる XMAP3 ActiveX コントロールで，物理マップとともに画面の入出力や帳票の出力に使用されます。

受信データと送信データは，共通の領域（データ長設定領域，共通インタフェース領域および論理マップ領域）を使用します。領域の定義は，登録集原文として XMAP3/Web for Cosminexus が提供します。詳細については，「9.3.3(1)　共通処理」および「9.3.3(2)　業務処理」を参照してください。

9.3.1　受信データの形式（Cosminexus ベースの場合）

受信データは，データ長設定領域，共通インタフェース領域および論理マップ領域（入力論理マップの領域）から成ります。使用できる文字コードは，シフト JIS です。

ユーザプログラムで受け取る受信データに設定されている値を次の表に示します。初回の業務呼び出し時はクライアント側からの受信データがないため，通信制御サーブレットで初期化された値が設定されています。

表 9-2　受信データの内容

データ項目名	長さ（位置）	データ形式	データ名	内容
データ長設定領域				
通信データ長	4	S9(9) USAGE COMP	XMAP-TRAN-LEN	受信データ全体の長さ
共通インタフェース領域（XMAP-COM）				
アイキャッチャ	4(4)	X(4)	XMAP-COM-ID	前回送信時の設定値
リターン値 1	2(8)	9(4) COMP-5	XMAP-COM-RTN	• 0：正常 • 0 以外：クライアント側で発生したエラーコード
リターン値 2	2(10)	9(4) COMP-5	XMAP-COM-RSN	• 0：正常 • 0 以外：クライアント側で発生したエラーコード

データ項目名	長さ(位置)	データ形式	データ名	内容
仮想端末名	8(12)	X(8)	XMAP-COM-TNAME	前回送信時の設定値
通信種別	4(20)	X(4)	XMAP-COM-MSG	前回送信時の設定値
マップ名	8(24)	X(8)	XMAP-COM-MAPNAME	前回送信時の設定値
未使用	4(32)	X(4)	XMAP-COM-RSV1	前回送信時の設定値
印刷完了通知オプション	1(36)	X	XMAP-COM-PRTOPT	前回送信時の設定値
終了通知オプション	1(37)	X	XMAP-COM-ENDOPT	前回送信時の設定値
未使用	6(38)	X(6)	XMAP-COM-RSV2	前回送信時の設定値
マッピングオプション大分類	4(44)	9(8) COMP-5	XMAP-COM-MAPOPT1	前回送信時の設定値
マッピングオプション小分類	4(48)	9(8) COMP-5	XMAP-COM-MAPOPT2	前回送信時の設定値
未使用	16(52)	X(16)	XMAP-COM-RSV3	前回送信時の設定値
URL データ長	2(68)	9(4) COMP-5	XMAP-COM-URLLNG	前回送信時の設定値
論理マップ長	2(70)	9(4) COMP-5	XMAP-COM-LSGLNG	論理マップ領域に設定されている入力論理マップの長さ（0～32,000）※1
未使用	60(72)	X(60)	XMAP-COM-RSV4	前回送信時の設定値
URL データ	128(132)	X(128)	XMAP-COM-URL	前回送信時の設定値
論理マップ領域				
論理マップ領域	32,000(260)	X(32,000)	XMAP-LSG	入力論理マップデータ※2

注※1

初回の業務呼び出し，帳票出力およびエラー発生時には 0 が設定されています。

注※2

初回の業務呼び出し，帳票出力およびエラー発生時には設定されません。

9.3.2　送信データの形式（Cosminexus ベースの場合）

送信データは，データ長設定領域，共通インタフェース領域および論理マップ領域（出力論理マップの領域）から成ります。使用できる文字コードは，シフト JIS です。

ユーザプログラムで設定する，送信データの値を次の表に示します。

表 9-3　送信データの設定値

データ項目名	長さ (位置)	データ形式	データ名	設定値
データ長設定領域				
通信データ長	4	S9(9) USAGE COMP	XMAP-TRAN-LEN	送信データ全体の長さ
共通インタフェース領域（XMAP-COM）				
アイキャッチャ	4(4)	X(4)	XMAP-COM-ID	'*XWC'
リターン値 1	2(8)	9(4) COMP-5	XMAP-COM-RTN	0（正常）
リターン値 2	2(10)	9(4) COMP-5	XMAP-COM-RSN	0（正常）
仮想端末名	8(12)	X(8)	XMAP-COM-TNAME	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※1
通信種別	4(20)	X(4)	XMAP-COM-MSG	• 画面の場合：'BWS△' • 帳票の場合：'OWS△'
マップ名	8(24)	X(8)	XMAP-COM-MAPNAME	物理マップ名※2
未使用	4(32)	X(4)	XMAP-COM-RSV1	すべて$(00)_{16}$
印刷完了通知オプション	1(36)	X	XMAP-COM-PRTOPT	• 画面の場合：'△' • 帳票の場合：'△'または'2'※3
終了通知オプション※4	1(37)	X	XMAP-COM-ENDOPT	• 画面入出力／帳票出力をする場合：'△' • ブラウザ側の XMAP3 業務を終了する場合：'E'
未使用	6(38)	X(6)	XMAP-COM-RSV2	すべて$(00)_{16}$
マッピングオプション大分類	4(44)	9(8) COMP-5	XMAP-COM-MAPOPT1	• マッピングオプションを指定する場合：3 • マッピングオプションを指定しない場合：0※5
マッピングオプション小分類	4(48)	9(8) COMP-5	XMAP-COM-MAPOPT2	• マッピングオプションを指定する場合：次の値を指定 マージ：13 物理マップだけ：14 論理マップだけ：15 • マッピングオプションを指定しない場合：0※5
未使用	16(52)	X(16)	XMAP-COM-RSV3	すべて$(00)_{16}$
URL データ長※4	2(68)	9(4) COMP-5	XMAP-COM-URLLNG	• URL データ（次回の呼び出し先 URL）を指定する場合：128

データ項目名	長さ(位置)	データ形式	データ名	設定値
				• URL データ（次回の呼び出し先 URL）を指定しない場合：0
論理マップ長	2(70)	9(4) COMP-5	XMAP-COM-LSGLNG	• 画面入出力／帳票出力をする場合：論理マップ領域に設定した出力論理マップの長さ • ブラウザ側の XMAP3 業務を終了する場合：0
未使用	60(72)	X(60)	XMAP-COM-RSV4	すべて$(00)_{16}$
URL データ※4	128(132)	X(128)	XMAP-COM-URL	次回の呼び出し先 URL※6
論理マップ領域				
論理マップ領域	32,000(260)	X(32,000)	XMAP-LSG	出力論理マップデータ※7

注※1

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※2

物理マップ名を左詰めで指定し，残りは空白を指定します。

注※3

帳票の場合の指定値と内容を次に示します。

指定値	内容
'△'	プリンタスプールに帳票を出力後，その印刷ドキュメントを完了（クローズ）する。
'2'	プリンタスプールに登録されている，印刷ドキュメント中の 1 ページとして帳票を出力する。 印刷ドキュメントは完了（クローズ）しないで継続する。

- 通常は'△'を指定します。'2'を指定する場合でも，最後のページの出力時には，必ず'△'を指定してください。
- プリンタ構成ファイル（X3PPINF）でスプール書き出し単位を「1 ページ毎」にした場合は，この指定は無効となり常に印刷ドキュメント中の 1 ページとして，プリンタスプールに登録されます。

注※4

「終了通知オプション」,「URL データ長」および「URL データ」の組み合わせによる動作は次のとおりです。

終了通知オプション	URL データ長	URL データ	動作
'△'	0	任意	前回と同じ URL を呼び出す。
	128	指定	指定された URL を呼び出す。
'E'	0	任意	業務終了後にブラウザを閉じる。
	128	指定	業務終了後に指定された URL を呼び出す。
		すべて空白，または先頭がヌル	業務終了後にブラウザを閉じる。

注※5

マッピングオプション大分類，およびマッピングオプション小分類が「0」の場合，前回の画面表示時に設定したマッピングオプションの内容が引き継がれます。ただし，一度もマッピングオプションを設定していない場合（例えば，初回の画面表示時）は，マッピングオプションに「マージ」が設定されたとして動作します。

注※6

次回の呼び出し先 URL を左詰めで指定し，残りは空白を指定します。

起動 HTML の通信制御サーブレット URL に指定した，ブラウザ側から呼び出す Web サーバ上の URL を変更する場合に指定します。

注※7

ブラウザ側の XMAP3 業務を終了する場合は，設定不要です。

9.3.3　ソースプログラムの記述（Cosminexus ベースの場合）

ユーザプログラムは業務を振り分ける共通処理と，各業務を実行する業務処理で構成されます。共通処理と業務処理について説明します。

それぞれのソースプログラムを作成する際は，XMAP3/Web for Cosminexus で提供しているサンプルプログラムを参照してください。

(1)　共通処理

共通処理では，受信データ中のマップ名を判定し，該当する業務処理を呼び出します。

共通処理をするユーザプログラム中の COBOL AP（共通処理）の例を次に示します。

COBOL AP（共通処理）には，データ送受信用の登録集原文を取り込む必要があります。

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.       プログラム名.

    :

 LINKAGE  SECTION.
  COPY    CBLIOTBL.                                    ]◀― データ送受信用の登録集原文の取り込み

 PROCEDURE  DIVISION    USING  BY REFERENCE XMAP-TRAN-TBL.
    :

* マップ名のチェックと該当処理の呼び出し
   EVALUATE  XMAP-COM-MAPNAME                         ┐
     WHEN  'KAD1GCND'                                 │
       CALL '業務処理プログラム名'                    │
                 USING  XMAP-TRAN-TBL                 │
            :                                         ├◀― 業務を振り分ける
     WHEN  OTHER                                      │
             :                                        │
                                                      │
   END-EVALUATE                                       ┘

     :

   MOVE    '*XWC'  TO      XMAP-COM-ID.               ┐
   MOVE    0       TO      XMAP-COM-RTN.             ├◀― 共通インタフェース領域にデータ設定
   MOVE    0       TO      XMAP-COM-RSN.             │
   MOVE    0       TO      XMAP-COM-URLLNG.          ┘

   COMPUTE XMAP-TRAN-LEN = XMAP-COM-LSGLNG + 256.  ]◀― 通信データ長に送信データ長を設定

 END  PROGRAM     プログラム名.
```

(a)　データ送受信用の登録集原文の取り込み

XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文（CBLIOTBL.cbl）を，COBOL AP（共通処理）の LINKAGE SECTION に COPY 文を使用して取り込んでください。

XMAP3/Web for Cosminexus では，次に示すフォルダにデータ送受信用の登録集原文を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

(2)　業務処理

業務処理では，受信データ中の入力論理マップデータを基に業務を行い，送信データを設定します。

業務処理をするユーザプログラム中の COBOL AP（業務処理）の例を次に示します。

COBOL AP（業務処理）には，論理マップの登録集原文および共通処理から引き渡されるデータ送受信用の登録集原文を取り込む必要があります。

```
* インタフェース領域と論理マップの取り込み
   COPY     MAP001I.                                    ┐◀― 論理マップの取り込み
   COPY     MAP0010.                                    ┘
   COPY     X3MODTBL.                                   ┐◀― 動的変更テーブルの取り込み

  LINKAGE SECTION.
   COPY CBLIOTBL.                                       ┐◀― データ送受信用の登録集原文の取り込み

*****************************************************************
*  業務処理                                                     *
*****************************************************************


* 受信した論理マップデータを入力論理マップデータに複写
   MOVE     XMAP-LSG          TO       MAP001I.        ┐◀― 受信データの論理マップ領域の値を
                                                       ┘  入力論理マップに代入
     :

* 入力論理マップのデータを基に業務処理を実行し，出力論理マップに
* 出力データを設定する
   MOVE     ALL  X'1F'        TO  MAP002G              ┐◀― 出力論理マップを初期化
   MOVE     MAP002T           TO  XMAP-COM-LSGLNG
     :
   MOVE     '         '       TO  XMAP-COM-TNAME       ┐
   MOVE     'BWS '            TO  XMAP-COM-MSG         │
   MOVE     'MAP002ND'        TO  XMAP-COM-MAPNAME     │◀― 共通インタフェース領域にデータ設定
   MOVE     ' '               TO  XMAP-COM-ENDOPT      │
   MOVE     0                 TO  XMAP-COM-MAPOPT1     │
   MOVE     0                 TO  XMAP-COM-MAPOPT2     ┘

   MOVE     MAP0020           TO  XMAP-LSG             ┐◀― 論理マップ領域に出力論理マップを設定

 業務処理-END.
```

(a)　論理マップの登録集原文の取り込み

XMAP3 での画面・帳票の開発時に生成した論理マップの登録集原文を，COBOL AP（業務処理）の WORKING-STORAGE SECTION に COPY 文を使用して取り込んでください。各業務処理内容によって入力論理マップや出力論理マップの設定方法が異なるので注意してください。

(b)　論理マップデータの取り込みと設定

受信データの論理マップ領域に設定された入力論理マップデータを入力論理マップに代入してください。また，出力論理マップにデータを設定したあと，送信データの論理マップ領域に設定した出力論理マップを代入してください。

9.3.4　ユーザプログラムのコンパイル（Cosminexus ベースの場合）

ユーザプログラムのコンパイルには，COBOL2002 を使用してください。

(1)　コンパイル時のポイント

(a)　プロジェクトの作成

ユーザプログラムを作成するには，COBOL2002 の開発マネージャでプロジェクトを作成するときに次の内容を選択します。

- 最終生成物の種類：ダイナミックリンクライブラリ
- プロジェクトの種類：-Dll 指定（DLL を作成する）
- DLL 呼び出し規約選択：DLL 属性を stdcall 属性にする
- 出力ファイル名：生成する DLL 名を指定する

(b)　コンパイラオプションの指定

コンパイルする時には，次に示すコンパイラオプションを指定する必要があります。-JPN,Alnum オプションは必要に応じて指定してください。

- -MainNotCBL
  副プログラムとしてコンパイルするオプション
- -Dll,Stdcall
  DLL 形式のオブジェクトを出力する（DLL 属性は stdcall 属性にする）オプション
- -DllInit
  呼び出し時に DLL を初期状態にするオプション
- -MultiThread
  マルチスレッド機能を使用するオプション
- -Comp5
  COMP-5 を指定できるようにするオプション
- -JPN,Alnum
  論理マップ内で日本語項目を扱えるようにするオプション

! 注意事項

COBOL2002 のコンパイラオプションに-BigEndian,Bin を指定してユーザプログラムを作成する場合，XMAP3/Web for Cosminexus のインタフェースでは，論理マップデータ以外のデータがリトルエンディアン形式となるように注意してください。論理マップのデータは，マップ作成時のマップ展開形式に合わせてください。

(c)　リンケージ時のポイントリンカオプションの指定

リンケージする時には，次に示すリンカオプションを指定する必要があります。

- -Dll,Stdcall
  stdcall 呼び出し規約の DLL を作成するためのオプション

9.4　ユーザプログラムの作成（OpenTP1 サーバ連携の場合）

ユーザプログラムは，Web ブラウザウィンドウ上に表示された画面とデータをやり取りしたり，帳票を印刷したりする，ユーザが作成する業務処理用の COBOL プログラムです。

XMAP3/Web for Cosminexus では，次に示すフォルダにサンプルプログラムを提供しています。ユーザプログラムを作成する上で参考にしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

ユーザプログラムでは，C/S 構成で使用していた XMAP3 システムと同様に論理マップデータを作成し，送信データに設定します。送信データには，論理マップデータのほかに仮想端末名や印刷完了通知，終了通知などのインタフェース情報や送信データの長さを設定します。

送信データに設定する論理マップは，クライアント側へ自動的にダウンロードされる XMAP3 ActiveX コントロールで，物理マップとともに画面の入出力や帳票の出力に使用されます。

受信データと送信データは，共通の領域（データ長設定領域，共通インタフェース領域および論理マップ領域）を使用します。領域の定義は，登録集原文として XMAP3/Web for Cosminexus が提供します。詳細については，「9.4.3　ソースプログラムの記述（OpenTP1 サーバ連携の場合）」を参照してください。

9.4.1　受信データの形式（OpenTP1 サーバ連携の場合）

受信データは，データ長設定領域および論理マップ領域（入力論理マップの領域）から構成されます。使用できる文字コードは，シフト JIS です。

ユーザプログラムで受け取る受信データに設定されている値を次の表に示します。初回の業務呼び出し時はクライアント側からの受信データがないため，通信制御サーブレットで初期化された値が設定されています。

表 9-4　受信データの内容

データ項目名	長さ（位置）	データ形式	データ名	内容
共通インタフェース領域（XMAP-COM）				
アイキャッチャ	4	X(4)	XMAP-COM-ID	前回送信時の設定値
リターン値 1	2(4)	9(4) COMP-5※1	XMAP-COM-RTN	• 0：正常 • 0 以外：クライアント側で発生したエラーコード
リターン値 2	2(6)	9(4) COMP-5※1	XMAP-COM-RSN	• 0：正常 • 0 以外：クライアント側で発生したエラーコード
仮想端末名	8(8)	X(8)	XMAP-COM-TNAME	前回送信時の設定値
通信種別	4(16)	X(4)	XMAP-COM-MSG	前回送信時の設定値
マップ名	8(20)	X(8)	XMAP-COM-MAPNAME	前回送信時の設定値

データ項目名	長さ（位置）	データ形式	データ名	内容
未使用	4(28)	X(4)	XMAP-COM-RSV1	前回送信時の設定値
印刷完了通知オプション	1(32)	X	XMAP-COM-PRTOPT	前回送信時の設定値
終了通知オプション	1(33)	X	XMAP-COM-ENDOPT	前回送信時の設定値
未使用	6(34)	X(6)	XMAP-COM-RSV2	前回送信時の設定値
マッピングオプション大分類	4(40)	9(8) COMP-5※1	XMAP-COM-MAPOPT1	前回送信時の設定値
マッピングオプション小分類	4(44)	9(8) COMP-5※1	XMAP-COM-MAPOPT2	前回送信時の設定値
未使用	16(48)	X(16)	XMAP-COM-RSV3	前回送信時の設定値
URL データ長	2(64)	9(4) COMP-5※1	XMAP-COM-URLLNG	前回送信時の設定値
論理マップ長	2(66)	9(4) COMP-5※1	XMAP-COM-LSGLNG	論理マップ領域に設定されている入力論理マップの長さ（0～32,000）※2
未使用	60(68)	X(60)	XMAP-COM-RSV4	前回送信時の設定値
URL データ	128(128)	X(128)	XMAP-COM-URL	前回送信時の設定値
論理マップ領域				
論理マップ領域	32,000(256)	X(32,000)	XMAP-LSG	入力論理マップデータ※3

注※1

アプリケーションサーバが UNIX の場合，「COMP-5」を「COMP」にしてください。

注※2

初回の業務呼び出し，帳票出力およびエラー発生時には 0 が設定されています。

注※3

初回の業務呼び出し，帳票出力およびエラー発生時には設定されません。

9.4.2　送信データの形式（OpenTP1 サーバ連携の場合）

送信データは，データ長設定領域および論理マップ領域（出力論理マップの領域）から構成されます。使用できる文字コードは，シフト JIS です。

ユーザプログラムで設定する，送信データの値を次の表に示します。

表 9-5　送信データの設定値

データ項目名	長さ（位置）	データ形式	データ名	設定値
共通インタフェース領域（XMAP-COM）				

データ項目名	長さ (位置)	データ形式	データ名	設定値
アイキャッチャ	4	X(4)	XMAP-COM-ID	'*XWC'
リターン値 1	2(4)	9(4) COMP-5※1	XMAP-COM-RTN	0（正常）
リターン値 2	2(6)	9(4) COMP-5※1	XMAP-COM-RSN	0（正常）
仮想端末名	8(8)	X(8)	XMAP-COM-TNAME	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※2
通信種別	4(16)	X(4)	XMAP-COM-MSG	• 画面の場合：'BWS△' • 帳票の場合：'OWS△'
マップ名	8(20)	X(8)	XMAP-COM-MAPNAME	物理マップ名※3
未使用	4(28)	X(4)	XMAP-COM-RSV1	すべて$(00)_{16}$
印刷完了通知オプション	1(32)	X	XMAP-COM-PRTOPT	• 画面の場合：'△' • 帳票の場合：'△'または'2'※4
終了通知オプション※5	1(33)	X	XMAP-COM-ENDOPT	• 画面入出力／帳票出力をする場合：'△' • ブラウザ側の XMAP3 業務を終了する場合：'E'
未使用	6(34)	X(6)	XMAP-COM-RSV2	すべて$(00)_{16}$
マッピングオプション大分類	4(40)	9(8) COMP-5※1	XMAP-COM-MAPOPT1	• マッピングオプションを指定する場合：3 • マッピングオプションを指定しない場合：0※6
マッピングオプション小分類	4(44)	9(8) COMP-5※1	XMAP-COM-MAPOPT2	• マッピングオプションを指定する場合：次の値を指定 マージ：13 物理マップだけ：14 論理マップだけ：15 • マッピングオプションを指定しない場合：0※6
未使用	16(48)	X(16)	XMAP-COM-RSV3	すべて$(00)_{16}$
URL データ長※5	2(64)	9(4) COMP-5※1	XMAP-COM-URLLNG	• URL データ（次回の呼び出し先 URL）を指定する場合：128 • URL データ（次回の呼び出し先 URL）を指定しない場合：0
論理マップ長	2(66)	9(4) COMP-5※1	XMAP-COM-LSGLNG	• 画面入出力／帳票出力をする場合：論理マップ領域に設定した出力論理マップの長さ • ブラウザ側の XMAP3 業務を終了する場合：0

データ項目名	長さ（位置）	データ形式	データ名	設定値
未使用	60(68)	X(60)	XMAP-COM-RSV4	すべて$(00)_{16}$
URL データ※5	128(128)	X(128)	XMAP-COM-URL	次回の呼び出し先 URL※7
論理マップ領域				
論理マップ領域	32,000(256)	X(32,000)	XMAP-LSG	出力論理マップデータ※8

注※1

アプリケーションサーバが UNIX の場合，「COMP-5」を「COMP」にしてください。

注※2

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※3

物理マップ名を左詰めで指定し，残りは空白を指定します。

注※4

帳票の場合の指定値と内容を次に示します。

指定値	内容
'△'	プリンタスプールに帳票を出力後，その印刷ドキュメントを完了（クローズ）する。
'2'	プリンタスプールに登録されている，印刷ドキュメント中の 1 ページとして帳票を出力する。 印刷ドキュメントは完了（クローズ）しないで継続する。

- 通常は'△'を指定します。'2'を指定する場合でも，最後のページの出力時には，必ず'△'を指定してください。
- プリンタ構成ファイル（X3PPINF）でスプール書き出し単位を「1 ページ毎」にした場合は，この指定は無効となり常に印刷ドキュメント中の 1 ページとして，プリンタスプールに登録されます。

注※5

「終了通知オプション」，「URL データ長」および「URL データ」の組み合わせによる動作は次のとおりです。

終了通知オプション	URL データ長	URL データ	動作
'△'	0	任意	前回と同じ URL を呼び出す。
	128	指定	指定された URL を呼び出す。
'E'	0	任意	業務終了後にブラウザを閉じる。
	128	指定	業務終了後に指定された URL を呼び出す。
		すべて空白，または先頭がヌル	業務終了後にブラウザを閉じる。

注※6

マッピングオプション大分類，およびマッピングオプション小分類が「0」の場合，前回の画面表示時に設定したマッピングオプションの内容が引き継がれます。ただし，一度もマッピングオプションを設定していない場合（例えば，初回の画面表示時）は，マッピングオプションに「マージ」が設定されたとして動作します。

注※7

次回の呼び出し先 URL を左詰めで指定し，残りは空白を指定します。

起動 HTML の通信制御サーブレット URL に指定した，ブラウザ側から呼び出す Web サーバ上の URL を変更する場合に指定します。

注※8

ブラウザ側の XMAP3 業務を終了する場合は，設定不要です。

9.4.3　ソースプログラムの記述（OpenTP1 サーバ連携の場合）

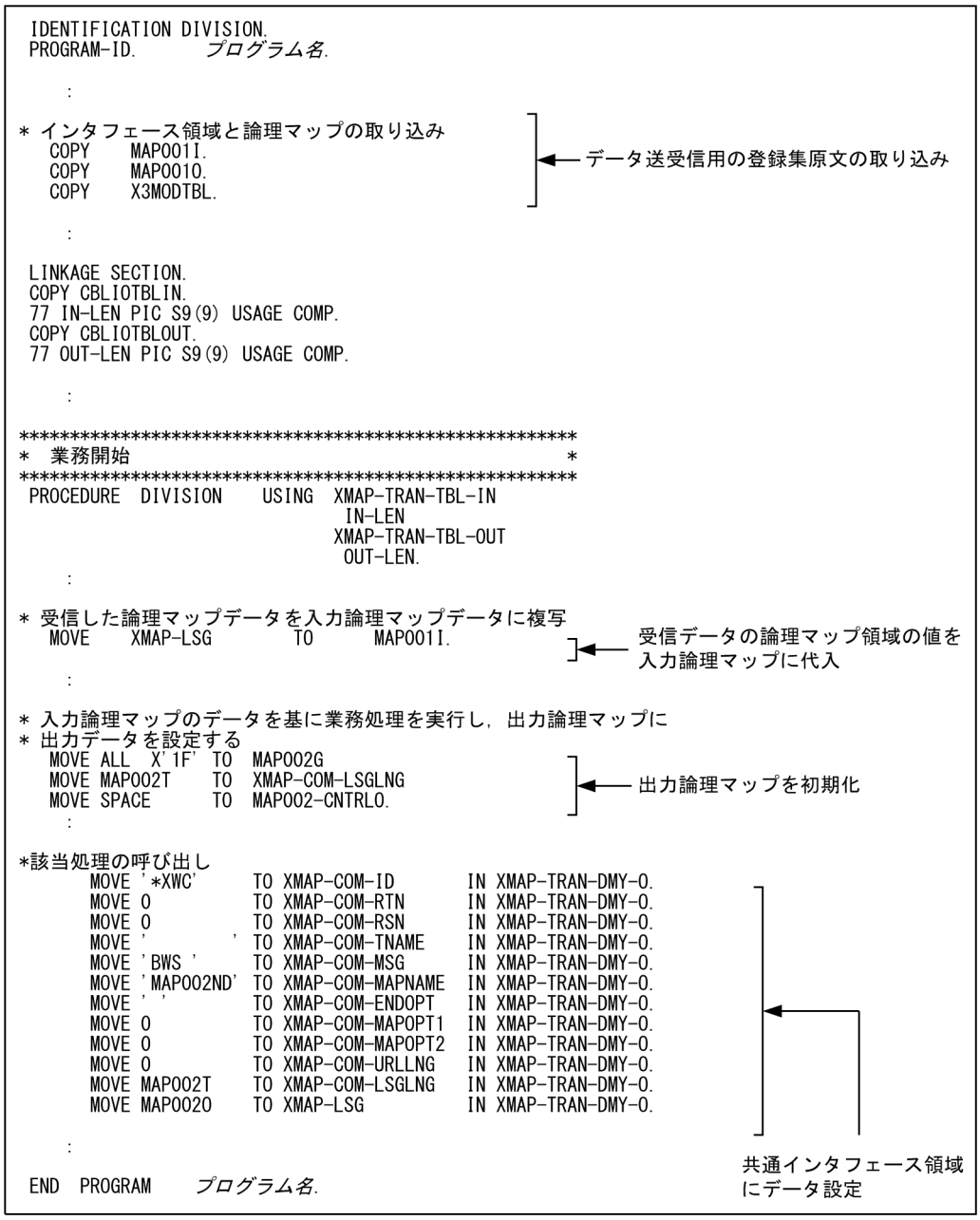
ユーザプログラムの業務処理について説明します。

ソースプログラムは，XMAP3/Web for Cosminexus が Cosminexus ベース用に提供しているサンプルを参考に作成してください。

OpenTP1 の COBOL SPP に関する定義（ユーザサービス定義，RPC インタフェース定義）は，TP1/COBOL adapter for Cosminexus のサンプルプログラムを参考にして作成してください。

COBOL AP では，データ送受信用の登録集原文を取り込み，受信データ中の入力論理マップデータを基に業務を実行し，送信データを設定します。

ユーザプログラム中の COBOL AP（業務処理）の例を次に示します。



(1) データ送受信用の登録集原文の取り込み

XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文（CBLIOTBL.cbl）をカスタマイズして，COBOL AP（業務処理）の LINKAGE SECTION に COPY 文を使用して取り込んでください。

XMAP3/Web for Cosminexus では，次に示すフォルダにデータ送受信用の登録集原文を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

COBOL adapter for Cosminexus ではインタフェース領域の長さは可変長だったので，長さを入れておく XMAP-TRAN-LEN が必要ですが，TP1/COBOL adapter for Cosminexus のインタフェース領域は固定長なので，XMAP-TRAN-LEN は必要なくなります。登録集原文の次の個所を修正してください。

- 「XMAP-TRAN-LEN」は削除

- 登録集原文を入力用と出力用で分け，「XMAP-TRAN-DMI」を「XMAP-TRAN-DMI-I（入力用）」と「XMAP-TRAN-DMI-O（出力用）」に修正
- アプリケーションサーバが UNIX の場合は，「COMP-5」を「COMP」に修正

自動生成ウィザード画面でも，データ属性は可変長ではなくバイト配列データ（byte[]）のままで生成してください。

TP1/COBOL adapter for Cosminexus の送信用インタフェース領域

```
01 XMAP-TRAN-TBL-IN     PIC X(32256).
01 XMAP-TRAN-DMY-I      REDEFINES       XMAP-TRAN-TBL.
                                              ← XMAP-TRAN-LENは削除
  03 XMAP-COM.
    04 XMAP-COM-ID           PIC X(4).
    04 XMAP-COM-RTN          PIC 9(4) COMP-5.
    04 XMAP-COM-RSN          PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME        PIC X(8).
    04 XMAP-COM-MSG          PIC X(4).
    04 XMAP-COM-MAPNAME      PIC X(8).
    04 XMAP-COM-RSV1         PIC X(4).
    04 XMAP-COM-PRTOPT       PIC X.
    04 XMAP-COM-ENDOPT       PIC X.
    04 XMAP-COM-RSV2         PIC X(6).
    04 XMAP-COM-MAPOPT1      PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2      PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3         PIC X(16).
    04 XMAP-COM-URLLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4         PIC X(60).
    04 XMAP-COM-URL          PIC X(128).
  03 XMAP-LSG                PIC X(32000).
```

TP1/COBOL adapter for Cosminexus の受信用インタフェース領域

```
01 XMAP-TRAN-TBL-OUT     PIC X(32256).
01 XMAP-TRAN-DMY-O      REDEFINES       XMAP-TRAN-TBL.
                                              ← XMAP-TRAN-LENは削除
  03 XMAP-COM.
    04 XMAP-COM-ID           PIC X(4).
    04 XMAP-COM-RTN          PIC 9(4) COMP-5.
    04 XMAP-COM-RSN          PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME        PIC X(8).
    04 XMAP-COM-MSG          PIC X(4).
    04 XMAP-COM-MAPNAME      PIC X(8).
    04 XMAP-COM-RSV1         PIC X(4).
    04 XMAP-COM-PRTOPT       PIC X.
    04 XMAP-COM-ENDOPT       PIC X.
    04 XMAP-COM-RSV2         PIC X(6).
    04 XMAP-COM-MAPOPT1      PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2      PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3         PIC X(16).
    04 XMAP-COM-URLLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4         PIC X(60).
    04 XMAP-COM-URL          PIC X(128).
  03 XMAP-LSG                PIC X(32000).
```

(2)　論理マップの登録集原文の取り込み

XMAP3 での画面・帳票の開発時に生成した論理マップの登録集原文を，COBOL AP（業務処理）の WORKING-STORAGE SECTION に COPY 文を使用して取り込んでください。各業務処理内容によって入力論理マップや出力論理マップの設定方法が異なるので注意してください。

(3)　論理マップデータの取り込みと設定

受信データの論理マップ領域に設定された入力論理マップデータを入力論理マップに代入してください。また，出力論理マップにデータを設定したあと，送信データの論理マップ領域に設定した出力論理マップを代入してください。

9.4.4　ユーザプログラムのコンパイル（OpenTP1 サーバ連携の場合）

ユーザプログラムのコンパイルには，COBOL2002 を使用してください。

以降，makefile の作成例とコンパイル手順について説明します。

(1)　Windows 版 OpenTP1 の場合

makefile のコーディング例

任意の名称を指定する個所は，斜体とアンダーラインで示します。ただし，「sv」の部分は共通の名称にする必要があります。さらに，実行形式ファイル名（.exe）に日本語は使えません。英文字で記述してください。改行する場合は，¥マークを付けてください。

```
!include <ntwin32.mak>
IDIRS     = -I$(DCDIR)¥include
CFLAGS= $(cflags) $(cvarsdll) $(IDIRS)

LLIBS     = libbetran.lib msvcrt.lib $(conlibsdll)
CC        = cl
LK        = ccbl2002 -Lib,CUI
CCBL      = ccbl2002

ZAIKO.exe : sv_sstb.obj SPP.obj 商品在庫.obj
  $(LK) -OUT:$@ SPP.obj 商品在庫.obj sv_sstb.obj $(LLIBS)

sv_sstb.c : sv.def
  $(DCDIR)¥bin¥stbmake sv.def

SPP.obj : spp.CBL
  $(CCBL) -Compile,NoLink -Lib,CUI -TDInf -Comp5 -Main,System  spp.CBL

商品在庫.obj : 商品在庫.cbl
    $(CCBL) -Compile,NoLink -Lib,CUI -TDInf -Comp5 商品在庫.cbl

clean:
-del      ZAIKO.exe SPP.obj 商品在庫.obj¥
sv_sstb.obj sv_sstb.c 商品在庫.cbc 商品在庫.cbo 商品在庫.cbp spp.cbo spp.cbp
```

コンパイル手順

makefile，RPC インタフェース定義ファイル，COBOL SPP 引数定義ファイル，ファイル，COBOL ソース，インタフェース領域，入出力論理マップを一つのフォルダに配置します。その後，コマンドプロンプトで次のコマンドを入力します。

- 実行形式ファイル（.exe）を生成するコマンド：
  <各ファイル配置ディレクトリ>nmake
- 実行形式ファイル（.exe）を削除するコマンド：
  <各ファイル配置ディレクトリ>nmake clean

生成された実行形式ファイル（.exe）を，%DCDIR%¥aplib に移動します。

COBOL SPP 引数定義ファイルは，TP1/COBOL adapter のサンプルプログラム，ファイル名 spp.cbl をそのまま流用できます。

(2)　UNIX 版 OpenTP1 の場合

makefile のコーディング例

任意の名称を指定する個所は，斜体とアンダーラインで示します。ただし，「sv」の部分は共通の名称にする必要があります。さらに，実行形式ファイル名（.exe）に日本語は使えません。英文字で記述してください。改行する場合は，¥マークを付けてください。

```
SIDIR     = $(DCDIR)
SVSRC     = spp.cbl 商品在庫.cbl
STBSRC    = sv_sstb.c
SVOBJ     = spp.o 商品在庫.o
MODULE    = ZAIKO

# set include directory temporarily
IDIRS     = -I$(SIDIR)/include
CFLAGS    = -c $(IDIRS)
CBLFG1    = -Compile,NoLink -Main,System
CBLFG2    = -Compile,NoLink
OFLAGS    =
LDIRS     = -brtl -L$(SIDIR)/lib -L/opt/HILNGcbl2k/lib
LIBS      = -lbetran -lcbl2k -lcbl2kml -ltactk -lm

# Compile program
CC = cc
CCBL      = ccbl2002
STBMK     = $(SIDIR)/bin/stbmake

all : $(MODULE)

ZAIKO : $(SVOBJ)
  $(CC) -o $@ $(SVOBJ) $(LDIRS) $(LIBS)

# set dependence of .o & .cbl
spp.o:spp.cbl
  $(CCBL) $(CBLFG1) spp.cbl
商品在庫.o:商品在庫.cbl
  $(CCBL) $(CBLFG2)  商品在庫.cbl

# set dependence of .o & .c
sv_sstb.o:sv_sstb.c

# set dependence of stb.c & .def
sv_sstb.c:sv.def
  $(STBMK) sv.def
clean :
  -rm $(SVOBJ) $(CLTOBJ)
```

コンパイル手順

makefile，RPC インタフェース定義ファイル，COBOL SPP 引数定義ファイル，ファイル，COBOL ソース，インタフェース領域，入出力論理マップを一つのフォルダに配置します。その後，シェルから次のコマンドを入力します。

- 実行形式ファイルを生成するコマンド：
  <各ファイル配置ディレクトリ>nmake
- 実行形式ファイルを削除するコマンド：
  <各ファイル配置ディレクトリ>nmake clean

生成された実行形式ファイル（.exe）を，$DCDIR/aplib に移動します。

COBOL SPP 引数定義ファイルは，TP1/COBOL adapter のサンプルプログラム，ファイル名 spp.cbl をそのまま流用できます。

9.5　ユーザプログラムの作成（DCCM3 連携の場合）

ユーザプログラム（業務 MPP）は，Web ブラウザウィンドウ上に表示された画面とデータをやり取りしたり，帳票を印刷したりする，ユーザが作成する業務処理用の COBOL プログラムです。Cosminexus と VOS3 DCCM3 で連携するシステムの場合，ユーザプログラムは，メインフレーム環境の COBOL85 で作成します。

XMAP3/Web for Cosminexus では，次に示すフォルダにサンプルプログラムを提供しています。ユーザプログラムを作成する上で参考にしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL_KEIS

ユーザプログラムでは，C/S 構成で使用していた XMAP3 システムと同様に論理マップデータを作成し，送信データに設定します。送信データには，論理マップデータのほかに仮想端末名や印刷完了通知，終了通知などのインタフェース情報や送信データの長さを設定します。

送信データに設定する論理マップは，クライアント側へ自動的にダウンロードされる XMAP3 ActiveX コントロールで，物理マップとともに画面の入出力や帳票の出力に使用されます。

受信データと送信データは，共通の領域（データ長設定領域，共通インタフェース領域および論理マップ領域）を使用します。領域の定義は，登録集原文として XMAP3/Web for Cosminexus が提供します。詳細については，「9.5.4　ソースプログラムの記述（DCCM3 連携の場合）」を参照してください。

9.5.1　受信データの形式（DCCM3 連携の場合）

受信データは，データ長設定領域，共通インタフェース領域および論理マップ領域（入力論理マップの領域）から成ります。使用できる文字コードは，KEIS78，KEIS83（ビッグエンディアン）です。

ユーザプログラムで受け取る受信データに設定されている値を次の表に示します。初回の業務呼び出し時はクライアント側からの受信データがないため，通信制御サーブレットで初期化された値が設定されています。

表 9-6　受信データの内容

データ項目名	長さ (位置)	データ形式	データ名	内容
データ長設定領域				
LL	2(0)	S9(4) BINARY	XMAP-TRAN-LL	受信データ長の総和 DCCM3 で設定するため，業務 MPP で設定する必要はない（$(30268)_{10}$ 固定）。
RR	2(2)	S9(4) BINARY	XMAP-TRAN-RR	XDM 情報 DCCM3 で設定するため，業務 MPP で設定する必要はない（$(0000)_{16}$ 固定）。
トランザクションコード	8(4)	X(8)	XMAP-TRAN-CODE	初回は通信制御サーブレットで指定した値が，2 回目以降は次に示す内容が返る。 画面入出力時： 入力論理マップの隠しフィールドの先頭 8 バイト

データ項目名	長さ (位置)	データ形式	データ名	内容
				帳票印刷時： セッションに格納したトランザクションコードの先頭８バイト トランザクションコードに指定できる文字を次に示す。 • 半角英字（大文字） • 半角数字 隠しフィールドに指定したトランザクションコードが８バイト未満の場合は，不足している文字数部分は半角スペースになる。
共通インタフェース領域（XMAP-COM）				
アイキャッチャ	4(12)	X(4)	XMAP-COM-ID	「*XWC」で固定
リターン値１	2(16)	9(4) COMP	XMAP-COM-RTN	• 0 正常 • 0 以外：クライアント側で発生したエラーコード
リターン値２	2(18)	9(4) COMP	XMAP-COM-RSN	• 0：正常 • 0 以外：クライアント側で発生したエラーコード
仮想端末名	8(20)	X(8)	XMAP-COM-TNAME	前回送信時の設定値※1
通信種別	4(28)	X(4)	XMAP-COM-MSG	前回送信時の設定値※1
マップ名	8(32)	X(8)	XMAP-COM- MAPNAME	前回送信時の設定値※1
共通インタフェース領域の文字コード	4(40)	S9(8) BINARY	XMAP-COM- COMCODE	前回送信時の設定値※1
印刷完了通知オプション	1(44)	X	XMAP-COM-PRTOPT	前回送信時の設定値※1
終了通知オプション	1(45)	X	XMAP-COM-ENDOPT	前回送信時の設定値※1
未使用	6(46)	X(6)	XMAP-COM-RSV2	すべて LOW-VALUE
マッピングオプション大分類	4(52)	9(8) COMP	XMAP-COM-MAPOPT1	前回送信時の設定値※1
マッピングオプション小分類	4(53)	9(8) COMP	XMAP-COM-MAPOPT2	前回送信時の設定値※1
未使用	16(60)	X(16)	XMAP-COM-RSV3	すべて LOW-VALUE
URL データ長	2(76)	9(4) COMP	XMAP-COM-URLLNG	前回送信時の設定値※1

データ項目名	長さ（位置）	データ形式	データ名	内容
論理マップ長	2(78)	9(4) COMP	XMAP-COM-LSGLNG	論理マップ領域に設定されている入力論理マップの長さ（0～30,000）※2
未使用	60(80)	X(60)	XMAP-COM-RSV4	すべて LOW-VALUE
URL データ	128(140)	X(128)	XMAP-COM-URL	前回送信時の設定値※1
論理マップ領域				
論理マップ領域	30,000(0)	X(30,000)	XMAP-LSG	入力論理マップデータ※3

注※1
　初回の業務呼び出し，帳票出力およびエラー発生時には LOW-VALUE が設定されています。

注※2
　初回の業務呼び出し，帳票出力およびエラー発生時には 0 が設定されています。

注※3
　初回の業務呼び出し，帳票出力およびエラー発生時には設定されません。

9.5.2　送信データの形式（DCCM3 連携の場合）

送信データは，データ長設定領域，共通インタフェース領域および論理マップ領域（出力論理マップの領域）から成ります。使用できる文字コードは，KEIS78，KEIS83（ビッグエンディアン）です。

ユーザプログラムで設定する，送信データの値を次の表に示します。

表 9-7　送信データの設定値

データ項目名	長さ（位置）	データ形式	データ名	設定値
データ長設定領域				
LL	2(0)	S9(4) BINARY	XMAP-TRAN-LL	送信データ長の総和（$(30268)_{10}$ 固定）。
RR	2(2)	S9(4) BINARY	XMAP-TRAN-RR	XDM 情報（$(0000)_{16}$ 固定）。
トランザクションコード	8(4)	X(8)	XMAP-TRAN-CODE	トランザクションコード（業務 MPP 名）を指定する。トランザクションコードに指定できる文字を次に示す。 • 半角英字（大文字） • 半角数字
共通インタフェース領域（XMAP-COM）				
アイキャッチャ	4(12)	X(4)	XMAP-COM-ID	「*XWC」で固定
リターン値 1	2(16)	9(4) COMP	XMAP-COM-RTN	0（正常）
リターン値 2	2(18)	9(4) COMP	XMAP-COM-RSN	0（正常）
仮想端末名	8(20)	X(8)	XMAP-COM-TNAME	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※1

データ項目名	長さ (位置)	データ形式	データ名	設定値
通信種別	4(28)	X(4)	XMAP-COM-MSG	• 画面の場合：'BWS△' • 帳票の場合：'OWS△'
マップ名	8(36)	X(8)	XMAP-COM-MAPNAME	物理マップ名※2
共通インタフェース領域の文字コード※3	4(40)	S9(8) BINARY	XMAP-COM-COMCODE	SJIS：0000 KEIS78：0001 KEIS83：0002 省略または不正な値が指定されたときは，0000 とする。
印刷完了通知オプション	1(44)	X	XMAP-COM-PRTOPT	• 画面の場合：'△' • 帳票の場合：'△'または'2'※4
終了通知オプション※5	1(45)	X	XMAP-COM-ENDOPT	• 画面入出力／帳票出力をする場合：'△' • ブラウザ側の XMAP3 業務を終了する場合：'E'
未使用	6(46)	X(6)	XMAP-COM-RSV2	すべて LOW-VALUE
マッピングオプション大分類	4(52)	9(8) COMP	XMAP-COM-MAPOPT1	• マッピングオプションを指定する場合：XMAP_MDO_SFLD • マッピングオプションを指定しない場合：0※6
マッピングオプション小分類	4(56)	9(8) COMP	XMAP-COM-MAPOPT2	• マッピングオプションを指定する場合：次の値を指定 マージ：XMAP_MDO_MAPFLD 物理マップだけ：XMAP_MDO_PHFLD 論理マップだけ：XMAP_MDO_LOGFLD • マッピングオプションを指定しない場合：0※6
未使用	16(60)	X(16)	XMAP-COM-RSV3	すべて LOW-VALUE
URL データ長※5	2(76)	9(4) COMP	XMAP-COM-URLLNG	「128」で固定
論理マップ長	2(78)	9(4) COMP	XMAP-COM-LSGLNG	• 画面入出力／帳票出力をする場合：論理マップ領域に設定した出力論理マップの長さ（0～30,000） • ブラウザ側の XMAP3 業務を終了する場合：0
未使用	60(80)	X(60)	XMAP-COM-RSV4	すべて LOW-VALUE
URL データ※5	128(140)	X(128)	XMAP-COM-URL	次回の呼び出し先 URL※7
論理マップ領域				
論理マップ領域	30,000(0)	X(30,000)	XMAP-LSG	出力論理マップデータ※8

注※1

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※2

物理マップ名を左詰めで指定し，残りは空白を指定します。

注※3

共通インタフェース領域の文字コードに 0001 または 0002 を指定した場合，共通インタフェース領域の英数字項目には KEIS コードの文字が指定され，論理マップは KEIS コードの論理マップと見なされます。

注※4

帳票の場合の指定値と内容を次に示します。

指定値	内容
'△'	プリンタスプールに帳票を出力後，その印刷ドキュメントを完了（クローズ）する。
'2'	プリンタスプールに登録されている，印刷ドキュメント中の 1 ページとして帳票を出力する。 印刷ドキュメントは完了（クローズ）しないで継続する。

- 通常は'△'を指定します。'2'を指定する場合でも，最後のページの出力時には，必ず'△'を指定してください。
- プリンタ構成ファイル（X3PPINF）でスプール書き出し単位を「1 ページ毎」にした場合は，この指定は無効となり常に印刷ドキュメント中の 1 ページとして，プリンタスプールに登録されます。

注※5

「終了通知オプション」,「URL データ長」および「URL データ」の組み合わせによる動作は次のとおりです。

終了通知オプション	URL データ長	URL データ	動作
'△'	0	任意	前回と同じ URL を呼び出す。
	128	指定	指定された URL を呼び出す。
'E'	0	任意	業務終了後にブラウザを閉じる。
	128	指定	業務終了後に指定された URL を呼び出す。
		すべて空白，または先頭がヌル	業務終了後にブラウザを閉じる。

注※6

マッピングオプション大分類，およびマッピングオプション小分類が「0」の場合，前回の画面表示時に設定したマッピングオプションの内容が引き継がれます。ただし，一度もマッピングオプションを設定していない場合（例えば，初回の画面表示時）は，マッピングオプションに「マージ」が設定されたとして動作します。

注※7

次回の呼び出し先 URL を左詰めで指定し，残りは空白を指定します。

起動 HTML の通信制御サーブレット URL に指定した，ブラウザ側から呼び出す Web サーバ上の URL を変更する場合に指定します。

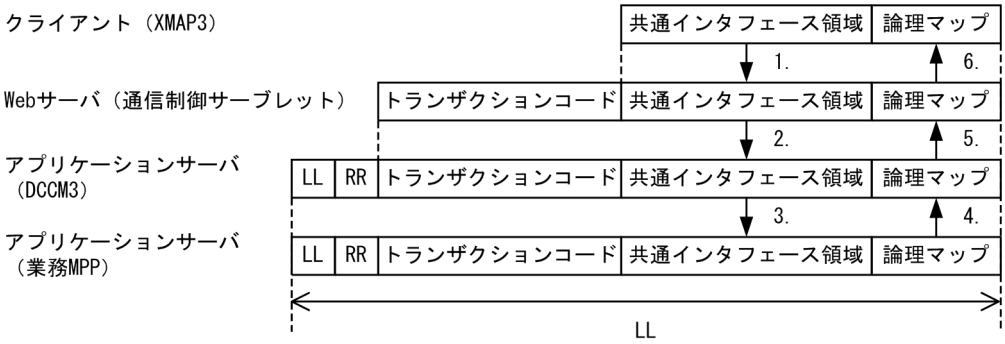
注※8

ブラウザ側の XMAP3 業務を終了する場合は，設定不要です。

9.5.3　各コントロール間で扱うデータ形式（DCCM3 連携の場合）

Cosminexus と VOS3 DCCM3 で連携するシステムで扱う送受信データの関係を次の図に示します。

図 9-8　Cosminexus と VOS3 DCCM3 で連携するシステムで扱う送受信データの関係



この図の各項番を説明します。

1. クライアント上の XMAP3 実行制御コントロールは，入力データ（共通インタフェース領域+入力論理マップ）を通信制御サーブレットに送信します。
2. Web サーバ上の通信制御サーブレットは，クライアントから受信した入力データの先頭にトランザクションコードを連結し，連結した入力データを TP1/COBOL アクセス用 Bean の setter メソッドでアプリケーションサーバ上の DCCM3 に送信します。連結するトランザクションコードを次に示します。

   画面の場合：入力論理マップの隠しフィールド

   帳票の場合：通信制御サーブレットのセッションデータ
3. アプリケーションサーバ上の DCCM3 は，通信制御サーブレットから受信した入力データの先頭に，LL，RR を連結し，連結した入力データをアプリケーションサーバ上の業務 MPP に送信します。DCCM3 が LL に設定する値は DCCM3 上で連結した入力データ長（LL+RR+トランザクションコード+共通インタフェース領域+入力論理マップの総和（$(30268)_{10}$ 固定））です。RR には XDM 情報が設定されます。
4. アプリケーションサーバ上の業務 MPP は，DCCM3 から受信した入力データを参照し，次回出力する出力データ（LL+RR+トランザクションコード+共通インタフェース領域+出力論理マップ）を生成します。生成した出力データをアプリケーションサーバ上の DCCM3 経由で通信制御サーブレットに送信します。
5. アプリケーションサーバ上の DCCM3 は，業務 MPP から受信した出力データの LL，RR を取り除いたデータ（トランザクションコード+共通インタフェース領域+出力論理マップ）を Web サーバ上の通信制御サーブレットに送信します。
6. Web サーバ上の通信制御サーブレットは，アプリケーションサーバ上の DCCM3 から受信した出力データのトランザクションコードを取り除いたデータ（共通インタフェース領域+出力論理マップ）をクライアントに送信します。

9.5.4　ソースプログラムの記述（DCCM3 連携の場合）

ユーザプログラムの処理について説明します。

それぞれのソースプログラムを作成する際は，XMAP3/Web for Cosminexus で提供しているサンプルプログラムを参照してください。

業務処理では，受信データ中のマップ名を判定し，該当する業務処理を呼び出し，受信データ中の入力論理マップデータを基に業務を実行し，送信データを設定します。

業務処理をするユーザプログラム中の COBOL AP（業務処理）の例を次に示します。

COBOL AP（業務処理）には，論理マップの登録集原文および業務処理から引き渡されるデータ送受信用の登録集原文を取り込む必要があります。

```
IDENTIFICATION DIVISION.
PROGRAM-ID.       プログラム名.

    :

 DATA DIVISION.
 WORKING-STORAGE SECTION.
 COPY    KAD1GCI.
 COPY    KAD2GCO.
 COPY    CBLIOTBL.
   COMMUNICATION SECTION.
   CD COM FOR I-O
 STATUS KEY STATUS-KEY.

 PROCEDURE DIVISION.
   RECEIVE COM FIRST SEGMENT INTO XMAP-TRAN.                    ←― RECEIVE要求の発行
   MOVE XMAP-LSG
    IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA TO KAD1GCI.              ←― 入力論理マップの取り込み

    MOVE 'KAD2GC'
      TO XMAP-TRAN-CODE   IN XMAP-TRAN.                         ←― トランザクションコードの設定

    MOVE 30268      TO  XMAP-TRAN-LL IN XMAP-TRAN.
    MOVE ALL X'00'  TO  XMAP-TRAN-RR IN XMAP-TRAN.              ←― LL RR の設定
    MOVE '*XWC'
      TO XMAP-COM-ID       IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0
      TO XMAP-COM-RTN      IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0
      TO XMAP-COM-RSN      IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE '        '
      TO XMAP-COM-TNAME    IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 'BWS '
      TO XMAP-COM-MSG      IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 'KAD2GCND'
      TO XMAP-COM-MAPNAME IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0001
      TO XMAP-COM-COMCODE IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.   ←― 共通インタフェース
    MOVE ' '                                                        領域の設定
      TO XMAP-COM-ENDOPT   IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0
      TO XMAP-COM-MAPOPT1 IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0
      TO XMAP-COM-MAPOPT2 IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE 0
      TO XMAP-COM-URLLNG   IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.
    MOVE KAD2GCT
      TO XMAP-COM-LSGLNG   IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.

    MOVE ALL  X'1F' TO KAD2GCG.
    MOVE SPACE      TO KAD2GC-CNTRLO.                           ←― 出力論理マップの生成
    MOVE KAD2GCO
      TO XMAP-LSG IN XMAP-TRAN-DMY IN XMAP-TRAN-AREA.

    SEND COM FROM XMAP-TRAN WITH EMI.                           ←― SEND 要求の発行

    GOBACK.
```

(1) データ送受信用の登録集原文の取り込み

XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文（CBLIOTBL.cbl）を，COBOL AP（業務処理）の LINKAGE SECTION に COPY 文を使用して取り込んでください。

XMAP3/Web for Cosminexus では，次に示すフォルダにデータ送受信用の登録集原文を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

(2)　論理マップの登録集原文の取り込み

XMAP3 での画面・帳票の開発時に生成した論理マップの登録集原文を，COBOL AP（業務処理）の WORKING-STORAGE SECTION に COPY 文を使用して取り込んでください。各業務処理内容によって入力論理マップや出力論理マップの設定方法が異なるので注意してください。

(3)　論理マップデータの取り込みと設定

受信データの論理マップ領域に設定された入力論理マップデータを入力論理マップに代入してください。また，出力論理マップにデータを設定したあと，送信データの論理マップ領域に設定した出力論理マップを代入してください。

9.5.5　ユーザプログラムのコンパイル（DCCM3 連携の場合）

ユーザプログラムのコンパイルとリンケージには，メインフレームの COBOL85 を使用してください。

9.6　Bean とサーブレット（Cosminexus ベースの場合）

Bean とサーブレットのソースプログラムの生成や記述方法およびコンパイルについて説明します。

9.6.1　COBOL アクセス用 Bean の生成（Cosminexus ベースの場合）

COBOL アクセス用 Bean は，ユーザプログラムからの送受信データを通信制御サーブレットを経由してそのままクライアント側に受け渡すための，setter および getter を記述した Java プログラムです。

COBOL アクセス用 Bean は，XMAP3/Web for Cosminexus がサンプルプログラムとして提供するデータ送受信用の登録集原文から，COBOL2002 Cosminexus 連携機能の COBOL アクセス用 Bean 生成ツールを使用して生成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに登録集原文（CBLIOTBL.cbl）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

また，提供する登録集原文の内容は次のとおりです。

```
01 XMAP-TRAN-TBL        PIC X(32260).
01 XMAP-TRAN-DMY        REDEFINES       XMAP-TRAN-TBL.
  03 XMAP-TRAN-LEN          PIC S9(9)       USAGE COMP.
  03 XMAP-COM.
    04 XMAP-COM-ID          PIC X(4).
    04 XMAP-COM-RTN         PIC 9(4) COMP-5.
    04 XMAP-COM-RSN         PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME       PIC X(8).
    04 XMAP-COM-MSG         PIC X(4).
    04 XMAP-COM-MAPNAME     PIC X(8).
    04 XMAP-COM-RSV1        PIC X(4).
    04 XMAP-COM-PRTOPT      PIC X.
    04 XMAP-COM-ENDOPT      PIC X.
    04 XMAP-COM-RSV2        PIC X(6).
    04 XMAP-COM-MAPOPT1     PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2     PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3        PIC X(16).
    04 XMAP-COM-URLLNG      PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG      PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4        PIC X(60).
    04 XMAP-COM-URL         PIC X(128).
  03 XMAP-LSG               PIC X(32000).
```

XMAP3/Web for Cosminexus では，COBOL アクセス用 Bean 生成ウィザードを利用して COBOL アクセス用 Bean を生成することをお勧めします。

生成した COBOL アクセス用 Bean の例を次に示します。

```
package パッケージ名;

import jp.co.hitachi_sk.j2cb.*;

public class クラス名 extends CBLAccess {
                :
  //インデックスを定義
  private static final int xmap_tran_tblIndex = 0;
  private static boolean noload = true;
  private static int[] mySize = null;
  private static GroupAccess[] myGroupAccess = null;
  private static String[] myTypeInfo = null;
  private static int myLength = 0;
/**
 *コンストラクタ
 */
                :
/**
 *初期化処理
 */
  private void init( ) throws J2CBException {
    //初期ロードだけ実行する
                :
  }
/**
 *初期化処理2
 */
  private synchronized void init2( ) throws J2CBException {
                :

  }
/**
 *xmap_tran_tbl設定メソッド
 *@param data byte[]オブジェクト
 *@param len データ長
 *@return void
 */
  public void setXmap_tran_tbl( Object data,int len) throws J2CBException {      ← setter メソッド
    setData(xmap_tran_tblIndex, "xmap_tran_tbl", makeVarData(data, len));
  }
/**
 *xmap_tran_tbl取得メソッド
 *@return byte[]オブジェクト
 */
  public Object getXmap_tran_tbl( ) throws J2CBException {                       ← getter メソッド
    return getData(xmap_tran_tblIndex, "xmap_tran_tbl");
  }
}
```

COBOL アクセス用 Bean 生成の注意事項

- XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文は，カスタマイズしないで，そのまま使用してください。登録集原文は，データ長設定領域（4 バイト）+共通インタフェース領域（256 バイト）＋論理マップ最大長（32,000 バイト）の固定形式です。
- COBOL アクセス用 Bean は，登録集原文の先頭行の「XMAP-TRAN-TBL PIC X(32260)」の基本項目で生成します。データ属性は「可変長データ（byte[])」を指定してください。

COBOL アクセス用 Bean 生成時に設定するパッケージ名やクラス名は，ユーザプログラムの設計に従って指定してください。

9.6.2　通信制御サーブレットの作成（Cosminexus ベースの場合）

通信制御サーブレットは，COBOL アクセス用 Bean を利用して，ユーザプログラムからの送受信データをそのままクライアント側に受け渡すための Java プログラムです。

通信制御サーブレットは，XMAP3 Cosminexus 連携機能のサンプルソースをカスタマイズして使用します。カスタマイズには，MyEclipse を使用します。XMAP3/Web for Cosminexus では，次に示すフォルダに通信制御サーブレットのソース（CBLSRV.java）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

通信制御サーブレットの例を次に示します。

```
package パッケージ名;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.beans.Beans;
import jp.co.hitachi_sk.j2cb.*;
import java.math.BigDecimal;

public class クラス名 extends HttpServlet {
  ServletContext c;

  //グローバル変数の初期化
  public void init(ServletConfig config) throws ServletException {
    super.init(config);
    c = config.getServletContext();
  }

  //HTTP Get リクエストの処理
  public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException {                                    ← doGet メソッド
    res.setContentType("Application/Octet-Stream");
    ServletOutputStream out = res.getOutputStream();

    /* XMAP3_COMにデータを受信 */
    byte XMAP3_COM[] = new byte[32256];
    ServletInputStream in = req.getInputStream();                           ← データの受信
    int value = 0;
    for (int i = 0; value != -1; i++) {
      value = in.readLine(XMAP3_COM, i, 1);
    }

    webuap bean = null;
    try {
      /* Beanのインスタンス生成 */
      bean = (webuap) Beans.instantiate(this.getClass().getClassLoader(), "smpcbl.webuap");
    } catch (ClassNotFoundException excp) {                                 ← インスタンスの生成
      excp.printStackTrace();
      return;
    }

    try {
      /* 受信データをCOBOLの領域に設定 */
      bean.setXmap_tran_tbl(XMAP3_COM, 32256);                              ← setter 呼び出し

      bean.callCOBOL();                                                     ← 業務 呼び出し

      /* COBOLの領域からデータを取得 */
      byte[] rdata = (byte[])bean.getXmap_tran_tbl();                       ← getter 呼び出し
      /* 取得データを送信 */
      out.write((byte[])rdata, 0, rdata.length);                            ← データの送信
    } catch (J2CBException e) {
      e.printStackTrace();
      return;
    }
  }

  //HTTP Post リクエストの処理
  public void doPost(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {                                             ← doPost メソッド
    doGet(req, res);
  }
}
```

通信制御サーブレット生成の注意事項

- setter 呼び出し，getter 呼び出しのメソッド名や引数は，生成した COBOL アクセス用 Bean に合わせて記述してください。
- 業務呼び出し（callCOBOL メソッド）は setter 呼び出しのあとに記述してください。

- XMAP3 Cosminexus 連携機能の通信制御サーブレットのソース（CBLSRV.java）を編集するとき，XMAP3 Cosminexus 連携機能とのインタフェース部（ソースプログラム中のデータの受信，データの送信の個所）のプログラミングについては，カスタマイズしないでそのまま使用してください。
- 通信制御サーブレットと COBOL アクセス用 Bean 間のデータ送受信では電文内容は参照しないため，エンコードはしていません。
- ContentType プロパティには，「Application/Octet-Stream」を指定してください。

9.7　Bean とサーブレット（OpenTP1 サーバ連携の場合）

Bean とサーブレットのソースプログラムの生成や記述方法およびコンパイルについて説明します。

9.7.1　TP1/COBOL アクセス用 Bean の生成（OpenTP1 サーバ連携の場合）

TP1/COBOL アクセス用 Bean は，ユーザプログラムからの送受信データを通信制御サーブレットを経由してそのままクライアント側に受け渡すための，setter および getter を記述した Java プログラムです。

(1)　TP1/COBOL アクセス用 Bean の生成手順

TP1/COBOL アクセス用 Bean は，XMAP3/Web for Cosminexus がサンプルプログラムとして提供するデータ送受信用の登録集原文から，TP1/COBOL adapter for Cosminexus の TP1/COBOL アクセス用 Bean 生成ツールを使用して生成します。生成した TP1/COBOL アクセス用 Bean をカスタマイズしてください。

XMAP3/Web for Cosminexus では，次に示すフォルダに登録集原文（CBLIOTBL.cbl）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

提供する登録集原文の内容は次のとおりです。

```
01 XMAP-TRAN-TBL        PIC X(32260).
01 XMAP-TRAN-DMY        REDEFINES       XMAP-TRAN-TBL.
  03 XMAP-TRAN-LEN          PIC S9(9)       USAGE COMP.
  03 XMAP-COM.
    04 XMAP-COM-ID          PIC X(4).
    04 XMAP-COM-RTN         PIC 9(4) COMP-5.
    04 XMAP-COM-RSN         PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME       PIC X(8).
    04 XMAP-COM-MSG         PIC X(4).
    04 XMAP-COM-MAPNAME     PIC X(8).
    04 XMAP-COM-RSV1        PIC X(4).
    04 XMAP-COM-PRTOPT      PIC X.
    04 XMAP-COM-ENDOPT      PIC X.
    04 XMAP-COM-RSV2        PIC X(6).
    04 XMAP-COM-MAPOPT1     PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2     PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3        PIC X(16).
    04 XMAP-COM-URLLNG      PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG      PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4        PIC X(60).
    04 XMAP-COM-URL         PIC X(128).
  03 XMAP-LSG               PIC X(32000).
```

XMAP3/Web for Cosminexus では，TP1/COBOL アクセス用 Bean 生成ウィザードを利用して TP1/COBOL アクセス用 Bean を生成することをお勧めします。

生成した TP1/COBOL アクセス用 Bean の例を次に示します。

```
package パッケージ名;

import jp.co.hitachi_sk.j2cb.*;

public class クラス名 extends CBLAccess {
                :
  //インデックスを定義
  private static final int xmap_tran_tblIndex = 0;
  private static boolean noload = true;
  private static int[] mySize = null;
  private static GroupAccess[] myGroupAccess = null;
  private static String[] myTypeInfo = null;
  private static int myLength = 0;
/**
 *コンストラクタ
 */
                :
/**
 *初期化処理
 */
  private void init( ) throws J2CBException {
    //初期ロードだけ実行する
                :
  }
/**
 *初期化処理2
 */
  private synchronized void init2( ) throws J2CBException {
                :

  }
/**
 *xmap_tran_tbl設定メソッド
 *@param data byte[]オブジェクト
 *@param len データ長
 *@return void
 */
  public void setXmap_tran_tbl( Object data,int len) throws J2CBException {      ← setter メソッド
    setData(xmap_tran_tblIndex, "xmap_tran_tbl", makeVarData(data, len));
  }
/**
 *xmap_tran_tbl取得メソッド
 *@return byte[]オブジェクト
 */
  public Object getXmap_tran_tbl( ) throws J2CBException {                       ← getter メソッド
    return getData(xmap_tran_tblIndex, "xmap_tran_tbl");
  }
}
```

TP1/COBOL アクセス用 Bean 生成の注意事項

- XMAP3 Cosminexus 連携機能のデータ送受信用の登録集原文は，カスタマイズしてください。登録集原文は，共通インタフェース領域（256 バイト）＋論理マップ最大長（32,000 バイト）の固定形式です。
- アプリケーションサーバが UNIX の場合は，「COMP-5」を「COMP」に修正してください。
- TP1/COBOL アクセス用 Bean は，登録集原文の先頭行の「XMAP-TRAN-TBL-IN（入力用）／XMAP-TRAN-TBL-OUT（出力用）」に「PIC X(32256)」の基本項目で生成します。データ属性は「バイト配列データ（byte[]）」を指定してください。

TP1/COBOL アクセス用 Bean 生成時に設定するパッケージ名やクラス名は，ユーザプログラムの設計に従って指定してください。

！ 注意事項

TP1 Connector を使用する場合は，TP1/COBOL アクセス用 Bean ウィザード画面で［TP1 Connector 経由のアクセスを使用する。］を選んでください。

(2) TP1/COBOL アクセス用 Bean のカスタマイズ

COBOL adapter for Cosminexus ではインタフェース領域の長さは可変長だったので，長さを入れておく XMAP-TRAN-LEN が必要ですが，TP1/COBOL adapter for Cosminexus のインタフェース領域は固定長なので，XMAP-TRAN-LEN は必要なくなります。登録集原文の次の個所を修正してください。

- 「XMAP-TRAN-LEN」は削除
- 登録集原文を入力用と出力用で分け，「XMAP-TRAN-DMI」を「XMAP-TRAN-DMI-I（入力用）」と「XMAP-TRAN-DMI-O（出力用）」に修正

自動生成ウィザード画面でも，データ属性は可変長ではなくバイト配列データ（byte[]）のままで生成してください。

TP1/COBOL adapter for Cosminexus の送信用インタフェース領域

```
01 XMAP-TRAN-TBL-IN      PIC X(32256).
01 XMAP-TRAN-DMY-I       REDEFINES       XMAP-TRAN-TBL.
                                              ← XMAP-TRAN-LENは削除
  03 XMAP-COM.
    04 XMAP-COM-ID           PIC X(4).
    04 XMAP-COM-RTN          PIC 9(4) COMP-5.
    04 XMAP-COM-RSN          PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME        PIC X(8).
    04 XMAP-COM-MSG          PIC X(4).
    04 XMAP-COM-MAPNAME      PIC X(8).
    04 XMAP-COM-RSV1         PIC X(4).
    04 XMAP-COM-PRTOPT       PIC X.
    04 XMAP-COM-ENDOPT       PIC X.
    04 XMAP-COM-RSV2         PIC X(6).
    04 XMAP-COM-MAPOPT1      PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2      PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3         PIC X(16).
    04 XMAP-COM-URLLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4         PIC X(60).
    04 XMAP-COM-URL          PIC X(128).
  03 XMAP-LSG                PIC X(32000).
```

TP1/COBOL adapter for Cosminexus の受信用インタフェース領域

```
01 XMAP-TRAN-TBL-OUT     PIC X(32256).
01 XMAP-TRAN-DMY-O      REDEFINES       XMAP-TRAN-TBL.
                                              ← XMAP-TRAN-LENは削除
  03 XMAP-COM.
    04 XMAP-COM-ID           PIC X(4).
    04 XMAP-COM-RTN          PIC 9(4) COMP-5.
    04 XMAP-COM-RSN          PIC 9(4) COMP-5.
    04 XMAP-COM-TNAME        PIC X(8).
    04 XMAP-COM-MSG          PIC X(4).
    04 XMAP-COM-MAPNAME      PIC X(8).
    04 XMAP-COM-RSV1         PIC X(4).
    04 XMAP-COM-PRTOPT       PIC X.
    04 XMAP-COM-ENDOPT       PIC X.
    04 XMAP-COM-RSV2         PIC X(6).
    04 XMAP-COM-MAPOPT1      PIC 9(8) COMP-5.
    04 XMAP-COM-MAPOPT2      PIC 9(8) COMP-5.
    04 XMAP-COM-RSV3         PIC X(16).
    04 XMAP-COM-URLLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-LSGLNG       PIC 9(4) COMP-5.
    04 XMAP-COM-RSV4         PIC X(60).
    04 XMAP-COM-URL          PIC X(128).
  03 XMAP-LSG                PIC X(32000).
```

9.7.2 通信制御サーブレットの作成（OpenTP1 サーバ連携の場合）

通信制御サーブレットは，TP1/COBOL アクセス用 Bean を利用して，ユーザプログラムからの送受信データをそのままクライアント側に受け渡すための Java プログラムです。

(1)　通信制御サーブレットの作成手順

通信制御サーブレットは，XMAP3 Cosminexus 連携機能のサンプルソースをカスタマイズして使用します。カスタマイズには，MyEclipse を使用します。XMAP3/Web for Cosminexus では，次に示すフォルダに通信制御サーブレットのソース（CBLSRV.java）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

通信制御サーブレットの例を次に示します。

```
package パッケージ名;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import java.beans.Beans;
import jp.co.hitachi_sk.j2cb.*;
import java.math.BigDecimal;

public class クラス名 extends HttpServlet {
  ServletContext c;

  //グローバル変数の初期化
  public void init(ServletConfig config) throws ServletException {
    super.init(config);
    c = config.getServletContext();
  }

  //HTTP Get リクエストの処理
  public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws ServletException, IOException {
    res.setContentType("Application/Octet-Stream");
    ServletOutputStream out = res.getOutputStream();

    /* XMAP3_COMにデータを受信 */
    byte XMAP3_COM[] = new byte[32256];
    ServletInputStream in = req.getInputStream();
    int value = 0;
    for (int i = 0; value != -1; i++) {
      value = in.readLine(XMAP3_COM, i, 1);
    }

    webuap bean = null;
    try {
      /* Beanのインスタンス生成 */
      bean = (webuap) Beans.instantiate(this.getClass().getClassLoader(), "smpcbl.webuap");
    } catch (ClassNotFoundException excp) {
      excp.printStackTrace();
      return;
    }

    try {
      /* 受信データをCOBOLの領域に設定 */
      bean.setXmap_tran_tbl(XMAP3_COM, 32256);

      bean.callCOBOL();

      /* COBOLの領域からデータを取得 */
      byte[] rdata = (byte[])bean.getXmap_tran_tbl();
      /* 取得データを送信 */
      out.write((byte[])rdata, 0, rdata.length);
    } catch (J2CBException e) {
      e.printStackTrace();
      return;
    }
  }

  //HTTP Post リクエストの処理
  public void doPost(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
    doGet(req, res);
  }
}
```

doGet メソッド

データの受信

インスタンスの生成

setter 呼び出し

業務呼び出し

getter 呼び出し

データの送信

doPost メソッド

(2)　通信制御サーブレットのカスタマイズ

生成した通信制御サーブレットでマップ名を判定して，業務処理に振り分ける OpenTP1 用サービス名を決定して SPP を呼び出せるようにします。コーディング例を次に示します。アプリケーションサーバが UNIX の場合は，エンディアン変換が必要です。エンディアン変換部分は，図中に網掛けで示します。

```
//HTTP Get リクエストの処理
      public void doGet(HttpServletRequest req,  HttpServletResponse res)
      throws ServletException, IOException {
        String server  =  “” ;                  //接続サーバ
        String[] host  = new String[1];         //OpenTP1ホスト名
        String group   = "";                   //サービスグループ名
        String service = "";                    //サービス名
        int    wkport  = 0;                     //ポート番号
        int    port    = 10020;
        int    len     = 6;                    //隠しフィールドの値の長さ

        try {
          server  = "rap";
          host[0] = "xmap03";
          wkport  = 10020;
        } catch (Exception e) {
         e.printStackTrace();
        }

        port = wkport;

        res.setContentType("Application/Octet-Stream");
        ServletOutputStream out = res.getOutputStream();

      /* XMAP3_COMにデータを受信 */
        byte XMAP3_COM[] = new byte[32256];
        ServletInputStream in = req.getInputStream();
        int value = 0;
        for (int i = 0; value != -1; i++) {
          value = in.readLine(XMAP3_COM, i, 1);
        }

      // エンディアン変換（リトル->ビッグ）
      byte work;
      // リターン値1
      work = XMAP3_COM[4];
      XMAP3_COM[4] = XMAP3_COM[5];
      XMAP3_COM[5] = work;

      // リターン値2
      work = XMAP3_COM[6];
      XMAP3_COM[6] = XMAP3_COM[7];
      XMAP3_COM[7] = work;

      // マッピングオプション大分類
      work = XMAP3_COM[40];
      XMAP3_COM[40] = XMAP3_COM[43];
      XMAP3_COM[43] = work;
      work = XMAP3_COM[41];
      XMAP3_COM[41] = XMAP3_COM[42];
      XMAP3_COM[42] = work;
```

```
      // マッピングオプション小分類
      work = XMAP3_COM[44];
      XMAP3_COM[44] = XMAP3_COM[47];
      XMAP3_COM[47] = work;
      work = XMAP3_COM[45];
      XMAP3_COM[45] = XMAP3_COM[46];
      XMAP3_COM[46] = work;

      // URLデータ長
      work = XMAP3_COM[64];
      XMAP3_COM[64] = XMAP3_COM[65];
      XMAP3_COM[65] = work;

      // 論理マップ長
      work = XMAP3_COM[66];
      XMAP3_COM[66] = XMAP3_COM[67];
      XMAP3_COM[67] = work;

          try {
          /*隠しフィールドの値を取得*/
          String lsg = new String(XMAP3_COM, 260, len);
          /*初回の場合*/
          if (false == id.equals("*XWC")) {
            lsg =  “SMPL1START";
          }
} catch (Exception e) {
            e.printStackTrace();
        }

        try {
          group   = lsg;
          service = lsg;
        } catch (Exception e) {
          e.printStackTrace();
        }

    // Beanのインスタンス生成
      CBLBEAN bean  = new CBLBEAN();
      TP1Client tp1 = new TP1Client();

      try {
        int[] idArr = new int[1];

        /* 受信データをCOBOLの領域に設定 */
        bean.setXmap_tran_tbl_inI(XMAP3_COM);

        try {
          if (server.equals("rap") ) {
            // rapサーバ  接続確立
            tp1.openConnection(host[0],port);
          } else {

            tp1.rpcOpen();
          }
        } catch(TP1ClientException ex) {
          System.out.println("open failed : " + ex.toString());
          ex.printStackTrace();
          return;
        }
```

```
    try {
           // SPPの呼び出し
           bean.call(tp1,group,service,TP1Client.DCNOFLAGS);

           if (server.equals("rap") ) {
             // rapサーバ　接続開放
             tp1.closeConnection();
           } else {
             // scdサーバ　接続開放
             tp1.rpcClose();
              }
         } catch(Exception e2) {
           e2.printStackTrace();
           System.out.println("error" + e2.toString());
           return;
         }

         /* COBOLの領域からデータを取得 */
         byte[] rdata = (byte[])bean.getXmap_tran_tbl_out0();

         // エンディアン変換（ビッグ->リトル）
         // リターン値1
         work = rdata[4];
         rdata[4] = rdata[5];
         rdata[5] = work;

         // リターン値2
         work = rdata[6];
         rdata[6] = rdata[7];
         rdata[7] = work;

         // マッピングオプション大分類
         work = rdata[40];
         rdata[40] = rdata[43];
         rdata[43] = work;
         work = rdata[41];
         rdata[41] = rdata[42];
         rdata[42] = work;

         // マッピングオプション小分類
         work = rdata[44];
         rdata[44] = rdata[47];
         rdata[47] = work;
         work = rdata[45];
         rdata[45] = rdata[46];
         rdata[46] = work;

         // URLデータ長
         work = rdata[64];
         rdata[64] = rdata[65];
         rdata[65] = work;

         // 論理マップ長
         work = rdata[66];
         rdata[66] = rdata[67];
         rdata[67] = work;

         /* 取得データを送信 */
         out.write((byte[])rdata, 0, 32256);
       } catch (Exception e) {
         System.out.println("test failed  ");
         e.printStackTrace();
         return;
       }
     }
```

(3) 通信制御サーブレットのカスタマイズ（TP1 Connector）

TP1 Connector で通信する場合は，Bean の呼び出し前に Connection インスタンス生成・InteractionSpec プロパティ設定処理を追加し，TP1 Connector 経由で RPC を実行するようにカスタマイズしてください。このカスタマイズは，処理を一つのトランザクションとしない場合でも必要です。コーディング例を次に示します。

```
try {
       // SPPの呼び出し
       bean.call(tp1,group,service,TP1Client.DCNOFLAGS);

       if (server.equals("rap") ) {
         // rapサーバ　接続開放
         tp1.closeConnection();
       } else {
         // scdサーバ　接続開放
         tp1.rpcClose();
          }
     } catch(Exception e2) {
       e2.printStackTrace();
       System.out.println("error" + e2.toString());
       return;
     }

     /* COBOLの領域からデータを取得 */
     byte[] rdata = (byte[])bean.getXmap_tran_tbl_out0();

     /* 取得データを送信 */
     out.write((byte[])rdata, 0, 32256);
   } catch (Exception e) {
     System.out.println("test failed  ");
     e.printStackTrace();
     return;
   }
 }
```

(4)　通信制御サーブレットのカスタマイズ（TP1 Connector で 2 フェーズコミットを実行する場合）

TP1 Connector で通信する場合に，処理を一つのトランザクションとする（2 フェーズコミットを実行する）ときは，UserTransaction インスタンスを生成し，トランザクションの開始と終了を追加します。コーディング例を次に示します。

```
                    :
// Beanのインスタンス生成
CBLBEAN bean  = new CBLBEAN();

javax.transaction.UserTransaction ut = null;
javax.resource.cci.Connection cx = null;
javax.resource.cci.Interaction ix = null;
try {
    //**************************************************
    //* Managed環境
    //**************************************************
    Context ctx = new InitialContext();

    // JNDI名前空間からUserTransactionインスタンスを取得
    ut = (UserTransaction)ctx.lookup("java:comp/UserTransaction");

    // JNDI名前空間からConnectionFactoryインスタンスを取得
    javax.resource.cci.ConnectionFactory cxf =
                    (javax.resource.cci.ConnectionFactory)
                                ctx.lookup("java:comp/env/OpenTP1RA");
    // トランザクション開始
    ut.begin();

    //**************************************************
    //* Managed環境, NonManaged環境共通処理
    //**************************************************
    // コネクション生成
    cx = cxf.getConnection();

    // インタラクション生成
    ix = cx.createInteraction();

    // インタラクションスペックにプロパティを設定
    InteractionSpecImpl ixSpec =  new InteractionSpecImpl();
    ixSpec.setServiceGroupName(group);        // サービスグループ名
    ixSpec.setServiceName(service);           // サービス名
    ixSpec.setFlags(ixSpec.DCNOFLAGS);        // RPC種別
    ixSpec.setWatchTime(180);                 // 応答待ち時間

    // 入力データをBeanのSetterを使用してセットする
    bean.setXmap_tran_tbl_inI(inData);

   // SPPを呼び出す
    bean.call(cxf, ix, ixSpec);

    // 出力データをBeanのGetterを使用して取得する
    outData = (byte[])bean.getXmap_tran_tbl_outO();

    // トランザクションをコミット
    ut.commit();

} catch (Exception e) {
    // 例外処理
    try {
         if (ut != null) {
             // トランザクションをロールバック
              ut.rollback();
         }
    } catch (Exception ex) {
        System.out.println("error :" + ex.toString());
    }
    throw e;
} finally {
    try {
        if(ix != null) {
            ix.close();   // インタラクションのクローズ ※1
        }
        if(cx != null) {
            cx.close();  // コネクションのクローズ ※2
        }
    } catch(ResourceException re) {}
}
```

注※1　インタラクションは，使用後に必ずclose()メソッドでクローズしてください。
注※2　コネクションは，使用後に必ずclose()メソッドでクローズしてください。

9.8　Bean とサーブレット（DCCM3 連携の場合）

Bean とサーブレットのソースプログラムの生成や記述方法およびコンパイルについて説明します。

9.8.1　TP1/COBOL アクセス用 Bean の生成（DCCM3 連携の場合）

TP1/COBOL アクセス用 Bean は，ユーザプログラムからの送受信データを通信制御サーブレットを経由してそのままクライアント側に受け渡すための，setter および getter を記述した Java プログラムです。

TP1/COBOL アクセス用 Bean は，XMAP3/Web for Cosminexus がサンプルプログラムとして提供するデータ送受信用の登録集原文から，TP1/COBOL adapter for Cosminexus の **TP1/COBOL アクセス用 Bean 生成ツール**を使用して生成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに登録集原文（CBLIOTBL_IN_BEAN.cbl, CBLIOTBL_OUT_BEAN.cbl）とひな形サンプル（CBLBEAN_KEIS.java）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL_KEIS

また，提供する登録集原文の内容は次のとおりです。

入力データ用登録集原文（CBLIOTBL_IN_BEAN.cbl）

```
01 XMAP-TRAN-AREA-IN                PIC X(30264).
01 XMAP-TRAN-IN                     REDEFINES XMAP-TRAN-AREA-IN.
  02 XMAP-TRAN-CODE                 PIC X(8).
  02 XMAP-TRAN-TBL-IN               PIC X(30256).
  02 XMAP-TRAN-DMY-I                REDEFINES       XMAP-TRAN-TBL-IN.
    03 XMAP-COM.
      04 XMAP-COM-ID                PIC X(4).
      04 XMAP-COM-RTN               PIC 9(4) COMP.
      04 XMAP-COM-RSN               PIC 9(4) COMP.
      04 XMAP-COM-TNAME             PIC X(8).
      04 XMAP-COM-MSG               PIC X(4).
      04 XMAP-COM-MAPNAME           PIC X(8).
      04 XMAP-COM-COMCODE           PIC S9(8) BINARY.
      04 XMAP-COM-PRTOPT            PIC X.
      04 XMAP-COM-ENDOPT            PIC X.
      04 XMAP-COM-RSV2              PIC X(6).
      04 XMAP-COM-MAPOPT1           PIC 9(8) COMP.
      04 XMAP-COM-MAPOPT2           PIC 9(8) COMP.
      04 XMAP-COM-RSV3              PIC X(16).
      04 XMAP-COM-URLLNG            PIC 9(4) COMP.
      04 XMAP-COM-LSGLNG            PIC 9(4) COMP.
      04 XMAP-COM-RSV4              PIC X(60).
      04 XMAP-COM-URL               PIC X(128).
    03 XMAP-LSG                     PIC X(30000).
```

出力データ用登録集原文（CBLIOTBL_OUT_BEAN.cbl）

```
01 XMAP-TRAN-AREA-OUT               PIC X(30264).
01 XMAP-TRAN-OUT                    REDEFINES XMAP-TRAN-AREA-OUT.
  02 XMAP-TRAN-CODE                 PIC X(8).
  02 XMAP-TRAN-TBL-OUT              PIC X(30256).
  02 XMAP-TRAN-DMY-O                REDEFINES       XMAP-TRAN-TBL-OUT.
    03 XMAP-COM.
      04 XMAP-COM-ID                PIC X(4).
      04 XMAP-COM-RTN               PIC 9(4) COMP.
      04 XMAP-COM-RSN               PIC 9(4) COMP.
      04 XMAP-COM-TNAME             PIC X(8).
      04 XMAP-COM-MSG               PIC X(4).
      04 XMAP-COM-MAPNAME           PIC X(8).
      04 XMAP-COM-COMCODE           PIC S9(8) BINARY.
      04 XMAP-COM-PRTOPT            PIC X.
      04 XMAP-COM-ENDOPT            PIC X.
      04 XMAP-COM-RSV2              PIC X(6).
      04 XMAP-COM-MAPOPT1           PIC 9(8) COMP.
      04 XMAP-COM-MAPOPT2           PIC 9(8) COMP.
```

```
        04 XMAP-COM-RSV3                PIC X(16).
        04 XMAP-COM-URLLNG              PIC 9(4) COMP.
        04 XMAP-COM-LSGLNG              PIC 9(4) COMP.
        04 XMAP-COM-RSV4                PIC X(60).
        04 XMAP-COM-URL                 PIC X(128).
      03 XMAP-LSG                       PIC X(30000).
```

XMAP3/Web for Cosminexus では，TP1/COBOL アクセス用 Bean 生成ウィザードを利用して TP1/COBOL アクセス用 Bean を生成することをお勧めします。

生成した TP1/COBOL アクセス用 Bean の例（通信制御サーブレットが参照する部分だけ）を次に示します。

```
public class CBLBEAN_KEIS extends TP1RPC {
                    :
  public void call( int cltid, String group, String service, int flags) throws J2CBException {
    init();
    super.call(cltid, group, service, flags);
  }
                    :
  public void setXmap_tran_area_inI( Object xmap_tran_area_inI) throws J2CBException {
    setInData(inIndex, "xmap_tran_area_in", xmap_tran_area_inI);
  }
                    :
  public Object getXmap_tran_area_outO( ) throws J2CBException {
    return getOutData(outIndex, "xmap_tran_area_out");
  }
}
```

call メソッド

setter メソッド

getter メソッド

TP1/COBOL アクセス用 Bean 生成の注意事項

- TP1/COBOL アクセス用 Bean を生成する際は，サンプルをカスタマイズしないで，そのまま使用してください。
- 通信制御サーブレットは TP1/Client/P 経由で業務 MPP を呼び出すため，TP1/COBOL アクセス用 Bean 生成ツールでは「TP1/Client/P または W 経由のアクセスを使用する。」を選択してください。
- TP1/COBOL アクセス用 Bean は，登録集原文の先頭行「XMAP-TRAN-AREA-IN PIC X(30264)（入力の場合）」「XMAP-TRAN-AREA-OUT PIC X(30264)（出力の場合）」の基本項目で生成し，データ属性は「バイト配列データ（byte[]）」を指定してください。

TP1/COBOL アクセス用 Bean 生成時に設定するパッケージ名やクラス名は，ユーザプログラムの設計に従って指定してください。

9.8.2 通信制御サーブレットの作成（DCCM3 連携の場合）

通信制御サーブレットは，TP1/COBOL アクセス用 Bean を利用して，ユーザプログラムからの送受信データをそのままクライアント側に受け渡すための Java プログラムです。

通信制御サーブレットは，XMAP3 Cosminexus 連携機能のサンプルソースをカスタマイズして使用します。カスタマイズには，MyEclipse を使用します。XMAP3/Web for Cosminexus では，次に示すフォルダに通信制御サーブレットのソース（CBLSRV_KEIS.java）を提供しています。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL_KEIS

通信制御サーブレットの例を次に示します。

```
package パッケージ名;

import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import jp.co.hitachi_sk.j2cb.*;
public class CBLSRV_KEIS extends HttpServlet{
  String defPath = "C:¥¥BETRAN.INI";          //定義ファイルのパス
  String targetHost = "xxx.xxx.xxx.xxx";      //TP1/Server
  String logname = "xmap";                    //ログイン名
  String password = "xmap";                   //パスワード
  int timeOut = 600;                          //メッセージ受信の最大待ち時間(秒指定)
  String group = "xmap";                      //サービスグループ名
  static final String FIRST_SERVICE = "xmap";   //初回起動時のトランザクションコード

  public void doGet(HttpServletRequest req, HttpServletResponse res)
  throws IOException, ServletException{
    ServletInputStream in = req.getInputStream();       //入力ストリーム
    ServletOutputStream out = res.getOutputStream();    //出力ストリーム
    res.setContentType("Application/Octet-Stream");     //ContentTypeの設定

    byte[] sessionData = new byte[8];     //トランザクションコード(セッションデータ)
    byte[] rData;                         //MPPからの取得データ
    byte[] outData;                       //クライアントへの送信データ
    byte[] sendData = new byte[30264];    //MPPへの送信データ
    String service;
    byte[] XMAP3_COM = new byte[32256];   //クライアントとの送受信データ
    int[] cltid = new int[1];             //クライアントID
    String[] setHost = new String[8];     //認証要求したホスト名
    HttpSession session;                  //セッションオブジェクト
    CBLBEAN_KEIS tp1Bean;                 //TP1/COBOLアクセス用Bean
    TP1Access tp1AccessObj;               //TP1/COBOL基本Beanのインスタンス
    try {
      int value = 0;
      int length = 0;
      for (int i = 0; value != -1; i++) {
          value = in.readLine(XMAP3_COM, i, 1);
          length++;
      }
      length--;

      tp1AccessObj = new TP1Access();

      tp1Bean = new CBLBEAN_KEIS();

      session = req.getSession(false);
      sessionData = getSession(req, XMAP3_COM, session);

      if(length > 0){
        sendData = makeSendData(XMAP3_COM, sessionData);
        service = convTranCode(sessionData);
      } else {
        service = FIRST_SERVICE;
      }
```

ライブラリのインポート

OpenTP1の環境設定

クライアントからのデータ受信

OpenTP1アクセス用のインスタンス生成

TP1/COBOLアクセス用Beanのインスタンス生成

トランザクションコードの切り出し

CALLメソッドで使用するトランザクションコードおよび送信データの生成

```
            tp1AccessObj.cltin(cltid, defPath, targetHost, logname,
            password, setHost, TP1Const.DCCLT_NO_AUTHENT);
            tp1Obj.open(cltid[0], TP1Const.DCNOFLAGS);
            tp1AccessObj.connect(cltid[0], TP1Const.DCNOFLAGS);

            tp1Bean.setXmap_tran_area_inI(sendData);

            tp1Bean.call(cltid[0], group, service, TP1Const.DCNOFLAGS);

            rData = (byte[])tp1Bean.getXmap_tran_area_outO();

            setSession(rData, req, session);

            outData = makeRecvData(rData);

            tp1AccessObj.disconnect(cltid[0], TP1Const.DCNOFLAGS);
            tp1AccessObj.close(cltid[0], TP1Const.DCNOFLAGS);
            tp1AccessObj.cltout(cltid[0], TP1Const.DCNOFLAGS);

            out.write(outData, 0, 32256);

            } catch (Exception e) {
              e.printStackTrace();
              return;
            }
}
```

業務MPPの開始手続き

setter呼び出し

業務MPPへのcall要求

getter呼び出し

トランザクションコードへのセッション格納

クライアントへ送信するデータの生成

業務MPPの終了手続き

クライアントへのデータ送信

例外処理

通信制御サーブレット生成の注意事項

- setter 呼び出し，getter 呼び出しのメソッド名や引数は，生成した TP1/COBOL アクセス用 Bean の記載に合わせて記述してください。
- 業務 MPP への call 要求は，setter 呼び出しのあとに記述してください。
- 通信制御サーブレットをカスタマイズする場合，次に示す処理以外はカスタマイズしないでください。

  ・OpenTP1 の環境設定

  ・setter，getter 呼び出しのメソッド名や引数
- 業務 MPP の開始または終了時に発行する cltin()，cltout()メソッドは，通信制御サーブレット内で実行してください。
- ContentType プロパティには，「Application/Octet-Stream」を指定してください。

9.9　次画面処理へのデータ引き継ぎ

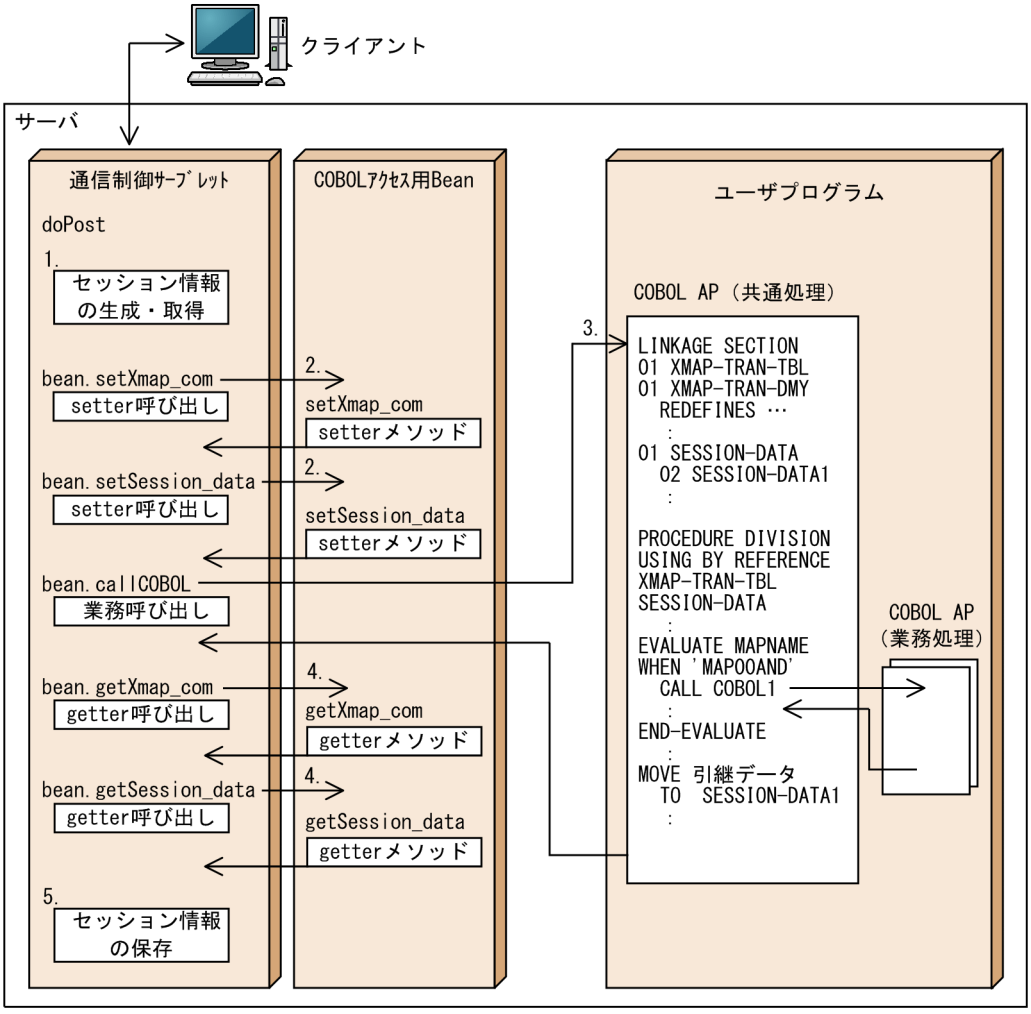
ユーザプログラムで，対話的な処理で必要なデータ（前回処理した業務データや呼び出し回数のカウンタなど）を次画面での処理へ引き継ぎたい場合，通信制御サーブレットと送受信し，Java 言語を利用してセッション情報として保存・取得する必要があります。

次画面での処理へデータを引き継ぐ場合に必要な，サーバ側でのプログラム間の処理の流れとプログラム作成のポイントについて説明します。

9.9.1　サーバ側でのプログラム間の処理の流れ

サーバ側でのプログラム間（COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean），ユーザプログラム，および通信制御サーブレット）の処理の流れを次の図に示します。

図 9-9　サーバ側でのプログラム間の処理の流れ（次画面処理へのデータ引き継ぎ）



1. 通信制御サーブレットでのセッション情報の生成・取得

通信制御サーブレットでは，初回の業務起動時にセッション情報を生成します。次画面以降の処理では，セッション情報に保存されている引き継ぎデータを取得します。

2. 通信制御サーブレットから COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）の setter 呼び出し

   通信制御サーブレットから受信データ設定用の setter メソッド，および引き継ぎデータ設定用の setter メソッドを呼び出します。

3. 通信制御サーブレットから業務呼び出し

   通信制御サーブレットからユーザプログラム中の COBOL AP（共通処理）を呼び出します。ユーザプログラムは受信データ領域と引き継ぎデータを引数によって受け渡します。

4. 通信制御サーブレットから COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）の getter 呼び出し

   通信制御サーブレットから送信データ（ユーザプログラムの業務処理で設定した送信データ）取得用の getter メソッド，および引き継ぎデータ取得用の getter メソッドを呼び出します。

5. 通信制御サーブレットでのセッション情報の保存

   通信制御サーブレットでは，引き継ぎデータをセッション情報に保存します。

9.9.2　プログラム作成のポイント

COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean），ユーザプログラム，および通信制御サーブレットを作成するときのポイントについて説明します。

(1)　COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）の作成

次の手順で作成してください。

1. 通信制御サーブレットとユーザプログラム間で受け渡しするためのデータ引き継ぎ用の登録集原文を作成する。
2. データ引き継ぎ用の登録集原文で，XMAP3 が提供する登録集原文（CBLIOTBL.cbl）を COPY 句で取り込む。
3. COBOL2002 Cosminexus 連携機能（または TP1/COBOL adapter for Cosminexus）の COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）生成ツールで生成する。

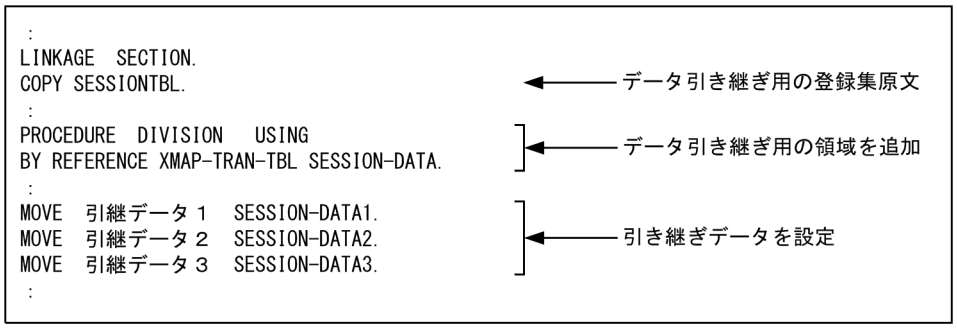
データ引き継ぎ用の登録集原文の例を次に示します。

```
COPY CBLIOTBL.                          ←――― XMAP3が提供する登録集原文
01 SESSION-DATA.
  02 SESSION-DATA1 PIC X(8).
  02 SESSION-DATA2 PIC X(8).            ←――― データ引き継ぎ用の領域
  02 SESSION-DATA3 PIC X(8).
```

(2)　ユーザプログラムの作成

ユーザプログラムは，次の手順で作成してください。

1. データ引き継ぎ用の登録集原文を LINKAGE SECTION で取り込む。
2. PROCEDURE DIVISION で受け取る引数にデータ引き継ぎ用の領域を追加する。
3. データ引き継ぎ領域に引き継ぎたいデータを設定する。



(3)　通信制御サーブレットの作成

通信制御サーブレットは，次の手順で作成してください。

1. Java 言語の HttpSession インタフェースを利用し，セッション情報として引き継ぎデータを保存・取得する。
2. COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）の setter，および getter を使用し，ユーザプログラムと引き継ぎデータを送受信する。

9.10　Bean および通信制御サーブレットのコンパイルと EAR ファイルの生成

生成した COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）および編集した通信制御サーブレットのソースプログラムをコンパイルして，EAR ファイルを生成します。プロジェクトを作成，ビルドして EAR ファイルを生成するときは，MyEclipse を使用します。

9.10.1　開発環境に準備しておくフォルダおよびファイル

XMAP3/Web for Cosminexus では，EAR ファイルを作成する前に，実行環境と同じフォルダ構成のフォルダを開発環境に作成することを推奨しています。開発環境と実行環境とで同じフォルダ構成にしておくことで，フォルダ構成を変更することなく実行環境にデプロイできます。

なお，MyEclipse を使用する場合は，次に示すフォルダ構成が自動的に生成されるので，フォルダ構成を作成する必要はありません。

推奨するフォルダ構成および格納ファイルを次に示します。

推奨するフォルダ構成

```
+-- WebApp※-----------起動HTML（x3xwbfrm.htm）および
|    |                 起動HTML用スクリプトファイル（x3xwbfrm.js）
|    |                 を格納
|    +-- Data※--------物理マップ，ポップアップメニューファイル，
|    |                 グラフィックデータ，およびFAX宛先ファイル
|    |                 を格納
|    +-- WEB-INF------web.xmlファイルを格納
|        +-- classes--COBOLアクセス用Bean，
|                     および通信制御サーブレットを格納
```

注※
フォルダ名には，任意の名称を付けてください。

次に，フォルダに格納するファイルについて説明します。

起動 HTML（x3xwbfrm.htm）

Web アプリケーションを実行するための HTML ファイルです。XMAP3/Web for Cosminexus で提供する起動 HTML をカスタマイズして使用できます。

起動 HTML 用スクリプトファイル（x3xwbfrm.js）

起動 HTML の各種情報の定義で使用するスクリプトファイルです。

物理マップ，ポップアップメニューファイル，グラフィックデータ，および FAX 宛先ファイル

- 物理マップは，画面表示・帳票印刷の実行時に使用するマッピング情報を格納したファイルです。
- ポップアップメニューファイルは，ポップアップテキストのメニュー情報を定義するファイルです。ポップアップメニューエディタで作成します。
- グラフィックデータは，画面表示・帳票印刷に必要なグラフィックデータを格納したファイルです。このファイルは，必要に応じて作成してください。
- FAX 宛先ファイルは，送信先の FAX 番号を記述したファイルです。

web.xml ファイル

web.xml ファイルは，Web コンテナのコンテキストを設定する XML ファイルです。通信制御サーブレットを呼び出す URL の情報を定義します。

COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）および通信制御サーブレット

通信制御サーブレットから呼び出される Bean です。

通信制御サーブレットは，COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）を利用して，ユーザプログラムからの送受信データをそのままクライアント側に受け渡すための Java プログラムです。

9.10.2　MyEclipse での EAR ファイルの生成手順

Java アプリケーションを作成するため，MyEclipse での EAR ファイルを生成する手順について説明します。MyEclipse の操作については，Cosminexus アプリケーションサーバのマニュアルを参照してください。

MyEclipse での EAR ファイルの生成手順を次の図に示します。

図 9-10　MyEclipse での EAR ファイルの生成手順

(1)　MyEclipse のセットアップ

日本語版 MyEclipse ポータルサイトから，MyEclipse の前提となる Eclipse アーカイブファイルをダウンロードします。ダウンロードしたアーカイブファイルを解凍し，MyEclipse を利用できるようにします。次に，プラグインを利用できるように環境設定ファイルにプロパティを追加し，Web サーバを起動できるようにリモート管理機能でログインします。

MyEclipse のセットアップ手順については，Cosminexus アプリケーションサーバのマニュアルを参照してください。

(2)　Web プロジェクトおよびパッケージの作成

Web プロジェクトを作成する前に，次に示す web.xml ファイルを編集しておきます。

XMAP3 インストールフォルダ¥Web for Cosminexus¥Sample¥COBOL¥web.xml

1 行目にある「encoding="MS932"」を「encoding="UTF-8"」に変更

その後，MyEclipse のメニューからプロジェクトを選択し，Web プロジェクトを作成します。作成したプロジェクトから，パッケージを作成します。

(3)　COBOL アクセス Bean，Servlet ソースの追加

AP を COBOL で作成する場合は，COBOL アクセス Bean，Servlet ソースを追加します。

エクスプローラで C:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus¥Sample¥COBOL を表示し，CBLBEAN.java と CBLSRV.java を選択し，プロジェクトのパッケージに上書きしてソースを登録します。また，プロジェクトの WEB-INF の下にある web.xml に，（2）で修正した web.xml を上書きします。

(4)　EAR ファイルの作成

EAR ファイルを作成するには，取り込む Web プロジェクトを選択します。

MyEclipse のメニューからエンタープライズ Java プロジェクトを作成します。このとき，EAR ファイルに取り込む Web プロジェクトとして，ここまでの手順で作成した Web プロジェクトを選択します。

(5)　MyEclipse で実行するサーバの選択

サーバに Cosminexus を設定します。

(6)　J2EE アプリケーションのデプロイ

MyEclipse から EAR ファイルをデプロイし，J2EE アプリケーションが正常に起動したことを確認します。

(7)　Web サーバ（Cosminexus HTTP Server（Hitachi Web Server）または IIS）の起動

業務を開始する前に，Web サーバ（Cosminexus HTTP Server（Hitachi Web Server）または IIS）を起動します。

なお，Web サーバと J2EE サーバの両方が起動していないと，XMAP3/Web for Cosminexus は実行できません。Web サーバを起動するときは，J2EE サーバも起動してください。

9.10.3　Eclipse および JBuilder での WAR ファイルの生成手順

Eclipse および JBuilder を使用して，サーブレットエンジンモードの場合にデプロイする，WAR ファイルの生成手順について説明します。

J2EE サーバモードで使用する場合は，Eclipse のコマンドで WAR ファイルを EAR ファイルに格納し，EAR ファイルをデプロイする必要があります。Eclipse のコマンドについては，Cosminexus アプリケーションサーバのマニュアルを参照してください。

(1)　Eclipse での WAR ファイル生成手順

Eclipse での WAR ファイル生成手順を次に示します。

1. プロジェクトの作成

Eclipse のメニューから，［ファイル］－［新規］－［プロジェクト］を開き，「Java プロジェクト」を選択し，任意の名称でプロジェクトを新規作成します。

新規 Java プロジェクト作成ウィザードで，利用している Cosminexus の JDK のバージョンを指定します（5.0 を選択することを推奨します）。

2. ライブラリの取り込みと出力デフォルトフォルダの設定

新規 Java プロジェクト作成ウィザードの「Java 設定」画面で，［ライブラリー］タブを選択し，［外部 JAR の追加］ボタンをクリックして次に示す外部 JAR を取り込みます。

Cosminexus ベースの場合

・*Cosminexus インストールフォルダ*¥CC¥client¥lib¥j2ee-javax.jar

・*COBOL2002 インストールフォルダ*¥lib¥j2cdrun.jar

Cosminexus と OpenTP1 サーバで連携の場合

・J2TP ライブラリ

・TP1/Client/J の TP1Client.jar

・TP1 Connector の tp1connector.jar（TP1 Connector 使用時）

・Cosminexus の hitj2ee.jar（TP1 Connector 使用時）

・*Cosminexus インストールフォルダ*¥CC¥client¥lib¥j2ee-javax.jar

Cosminexus と VOS3 DCCM3 で連携の場合

・J2TP ライブラリ

・*Cosminexus インストールフォルダ*¥CC¥client¥lib¥j2ee-javax.jar

「デフォルト出力フォルダー」には，次に示すように WEB-INF の下に classes フォルダを作成して指定します。

```
/XmapWebApp/WEB-INF/classes
```

3. プロジェクトへのソースプログラムの追加

［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「新規」－「ソース・フォルダー」を選択し，ソースをインポートするためのフォルダを作成します。

その後，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「インポート」を選択し，「ファイル・システム」画面で業務サーブレットのソースプログラム※を追加します。このとき，インポート対象は Java ソースの上位レベルのノード名から選択してください。

ソースフォルダへのインポート後，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「インポート」を選択し，残りのソース以外のフォルダ，ファイルなどをノード直下にインポートします。

注※

ソースプログラムは，必ず 2 階層下の［パッケージ名］フォルダ下に格納してください。

4. Java コンパイラの設定

コンパイルオプションの「エラー/警告」を設定しない場合，警告レベルのエラーメッセージを表示したくないときは，［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「プロパティー」を選択します。「Web アプリケーション名」のプロパティダイアログの「Java コンパイラー」－「エラー/警告」から次の設定をします。

- ［プロジェクト特定設定を使用可能にする］のボックスをチェックする

- ［潜在的なプログラミングの問題］－［serialVersionUID なしのシリアライズ可能クラス］を［無視］にする
- ［J2SE 5.0 オプション］－［未検査の総称型操作］を［無視］にする

5. WAR ファイル用 DD の作成

Web アプリケーション用の DD（web.xml）を作成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに Web アプリケーション用の DD（web.xml）をサンプルとして提供しています。ユーザ環境に合わせてカスタマイズしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

なお，このファイルをインポートする前にあらかじめ編集しておくこともできます。

6. Java プログラムのコンパイル・ビルド

Eclipse のメニューから，［プロジェクト］－［プロジェクトのビルド］で Java プログラムをコンパイル・ビルドします。

なお，デフォルトの設定では「自動的にビルド」となっているため，ソースプログラムをインポートしたときに自動的にコンパイルされます。

7. WAR ファイルの生成

［パッケージ・エクスプローラー］の Web アプリケーションノードを右クリックし，コンテキストメニューから「エクスポート」を選択します。「エクスポート」ダイアログで「アーカイブ・ファイル」を選択し，WAR ファイルを生成します。

WAR ファイルを生成する前に，「エクスポート」－「オプション」の次の設定がチェックされていることを確認してください。

- ［ZIP フォーマットで保管］
- ［選択したディレクトリーのみ作成］
- ［ファイルの内容を圧縮］

Eclipse を使用してプロジェクトを作成，ビルドしたときに作成されるフォルダ構成は次のようになります。

```
プロジェクト名
  └Webアプリケーション名
      └WEB-INF
          └classes
              └パッケージ名
```

(2)　JBuilder での WAR ファイル生成手順

JBuilder での WAR ファイル生成手順を次に示します。

1. プロジェクトの作成

JBuilder のメニューから，［ファイル］－［新規プロジェクト］を開き，任意の名称でプロジェクトを新規作成します。

2. Web アプリケーションの作成

JBuilder のメニューから，［ファイル］－［新規］を開き，［Web］タブの［Web アプリケーション］を選択します。Web アプリケーションウィザードダイアログを使用して Web アプリケーションをプロジェクトに追加します。また，作成した Web アプリケーションのプロパティを設定します。

3. プロジェクトへのソースプログラムの追加

プロジェクトペインの Web アプリケーションノードを右クリックし，コンテキストメニューから「ファイル／パッケージ／クラスの追加」を選択します。「プロジェクト名」に追加ダイアログで，生成した COBOL アクセス用 Bean（または TP1/COBOL アクセス用 Bean）および編集した通信制御サーブレットのソースプログラム※を追加します。

注※

ソースプログラムは，［パッケージ名］フォルダ下に配置しておいてください。

4. 必須ライブラリの定義

JBuilder のメニューから，［ツール］－［ライブラリ設定］を選択し，表示されたライブラリ設定ダイアログで［新規］ボタンをクリックします。

表示された新規ライブラリウィザードダイアログの［新規］ボタンで［ライブラリパス］にバージョンに合った servlet.jar ファイルを追加し，［場所］には「ユーザーホーム」を指定して任意の名称でライブラリを定義します。

5. 必須ライブラリの取り込み

JBuilder のメニューから，［プロジェクト］－［プロジェクトプロパティ］を選択します。［パス］タブ内の［必須ライブラリ］タブを選択し，［追加］ボタンで，次に示す「ユーザーホーム」内のライブラリを追加してください。

Cosminexus ベースの場合

・J2cb2k ライブラリ

・4.で定義した javax.servlet および javax.servlet.http パッケージを含むライブラリ（servlet.jar）

Cosminexus と OpenTP1 サーバで連携の場合

・J2TP ライブラリ

・TP1/Client/J の TP1Client.jar

・TP1 Connector の tp1connector.jar（TP1 Connector 使用時）

・Cosminexus の hitj2ee.jar（TP1 Connector 使用時）

・4.で定義した javax.servlet および javax.servlet.http パッケージを含むライブラリ（servlet.jar）

Cosminexus と VOS3 DCCM3 で連携の場合

・J2TP ライブラリ

・4.で定義した javax.servlet および javax.servlet.http パッケージを含むライブラリ（servlet.jar）

6. WAR ファイル用の DD の作成

「WebApp 配布ディスクリプタエディタ」を使用して，Web アプリケーション用の DD（web.xml）を作成します。

XMAP3/Web for Cosminexus では，次に示すフォルダに Web アプリケーション用の DD（web.xml）をサンプルとして提供しています。ユーザ環境に合わせてカスタマイズしてください。

XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥COBOL

7. IDE オプションの設定

JBuilder のメニューから，［ツール］－［IDE オプション］を開き，ファイルタイプ一覧中のファイルタイプに pmp を関連づけます。

次に Web アプリケーションのプロパティ中の［WebApp］タブ－［含まれるファイルタイプ］で pmp に関連づけたファイルタイプをチェックします。

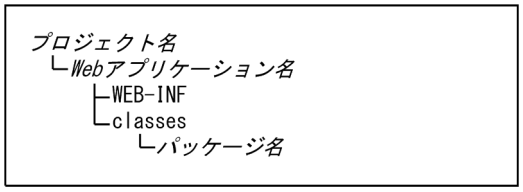
8. WAR ファイルの生成（プロジェクトのビルド）

プロジェクトペインの Web アプリケーションノードを右クリックし，コンテキストメニューから「プロパティ」を選択します。「Web アプリケーション名」のプロパティダイアログの［WebApp］タブで，［ビルド］プルダウンメニューの［プロジェクトのビルド時のみ］を選択します。

また，WAR ファイルの出力先などを指定します。

JBuilder のメニューから，［プロジェクト］－［プロジェクトのメイク］を選択してプロジェクトをビルドします。

JBuilder を使用してプロジェクトを作成，ビルドしたときに作成されるフォルダ構成は次のようになります。



10 XMAP3 TP1/Web 連携

この章では，XMAP3 TP1/Web 連携用の AP の作成方法や，XMAP3 TP1/Web 連携用の AP から呼び出す XMAP3 Web 実行環境ライブラリについて説明します。

10.1　XMAP3 TP1/Web 連携用の AP の概要

XMAP3 TP1/Web 連携用の AP の概要について説明します。

なお，XMAP3/Web for Cosminexus では，次に示すフォルダに XMAP3 TP1/Web 連携機能用のサンプルプログラムを提供しています。AP を作成する上で参考にしてください。

XMAP3インストールフォルダ¥Web for TP1¥SAMPLE

10.1.1　XMAP3 TP1/Web 連携用の AP

XMAP3 TP1/Web 連携機能用の AP とは，Web ブラウザウィンドウ上に表示された画面とデータをやり取りしたり，帳票を印刷したりするユーザが作成するプログラムです。

XMAP3 TP1/Web 連携機能は，AP のインタフェースとして次の二つを使用できます。

- Web ブラウザへ出力するための画面・帳票データを生成するインタフェース
- 画面・帳票の出力結果および画面に入力されたデータを解析するインタフェース

これらのインタフェースを使用して，TP1/Web のサービスセットに対応した AP を作成します。XMAP3 TP1/Web 連携機能では，TP1/Web のサービスセットを呼び出しません。したがって，TP1/Web のサービスセットは，必要に応じてユーザが明示的に呼び出してください。AP は，TP1/Web が提供する「DC_USR サービスセット」を利用して，XMAP3 Web 実行環境ライブラリの API を制御し，画面とのデータ送受信を実行します。

XMAP3 Web 実行環境ライブラリの API を呼び出す AP は，TP1/Web のサービスプロセスのスタティックセションスケジュール，ダイナミックセションスケジュールのどちらでも利用できます。

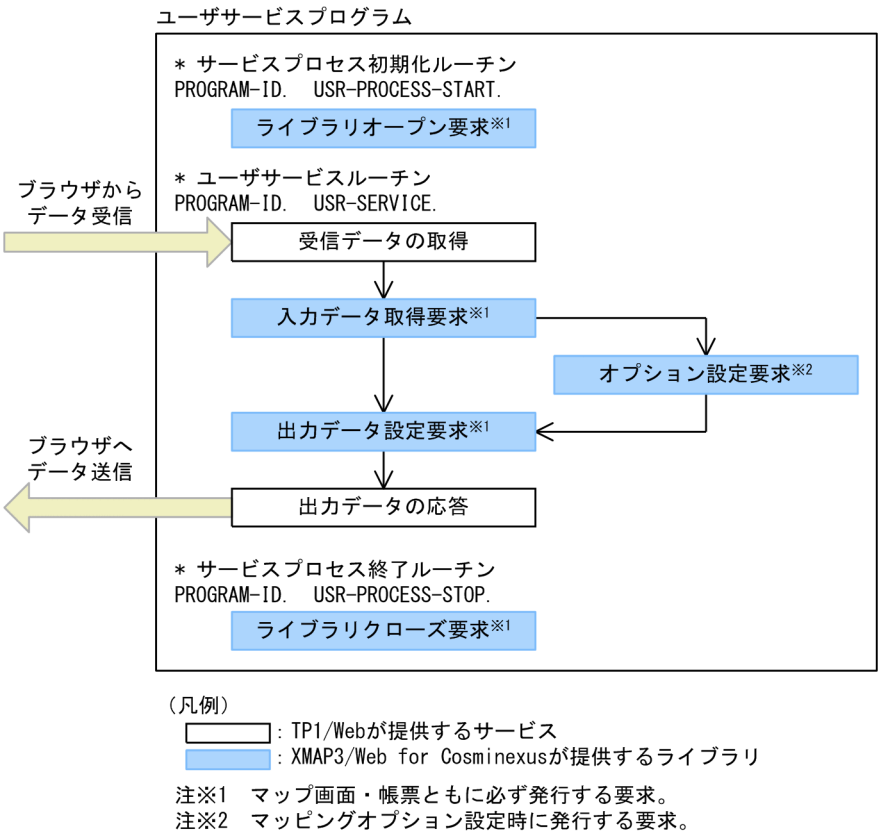
AP の開発言語には COBOL，または C 言語を適用できます。

10.1.2　XMAP3 TP1/Web 連携用の AP の構成

AP から XMAP3 Web 実行環境ライブラリのインタフェースを呼び出す場合，次の図に示す順序で呼び出します。

XMAP3 TP1/Web 連携機能を利用した AP からブラウザへデータ送信する場合，XMAP3 Web 実行環境ライブラリで編集された送信データを返信してください。XMAP3 TP1/Web 連携機能を使用した業務では，XMAP3 Web 実行環境ライブラリで編集されたデータ以外をブラウザ側に送信する TP1/Web 機能は利用できません。XMAP3 TP1/Web 連携機能で編集されたデータ以外をブラウザ側に送信した場合，ブラウザ業務は終了します。

図 10-1　XMAP3 Web 実行環境ライブラリインタフェースの発行順序

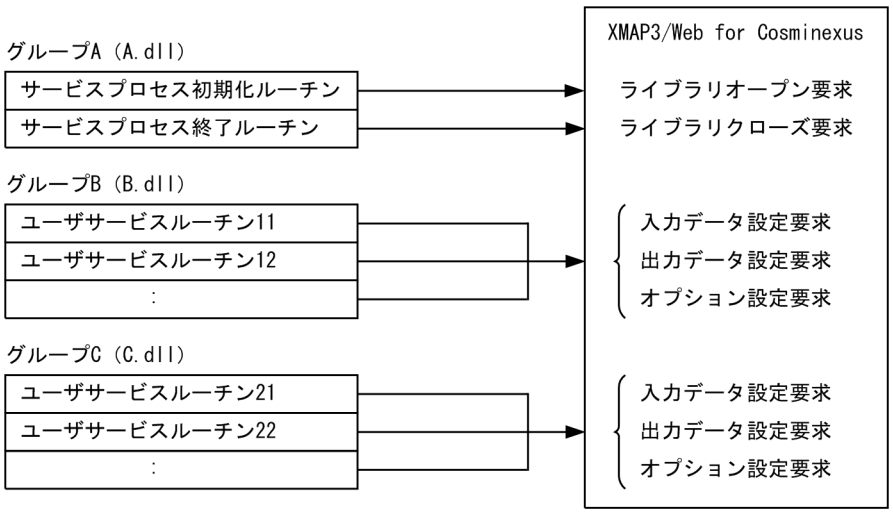


AP の作成に使用する言語ごとに，推奨するプログラム構成を説明します。

- COBOL を使用する場合

  推奨するプログラム構成の例を次の図に示します。XMAP3 TP1/Web 連携機能のライブラリオープン要求，およびライブラリクローズ要求をするグループは，一つだけ作成してください。グループごとにライブラリオープン要求，およびライブラリクローズ要求を呼び出す必要はありません。

  図 10-2　推奨するプログラム構成の例（COBOL を使用する場合）



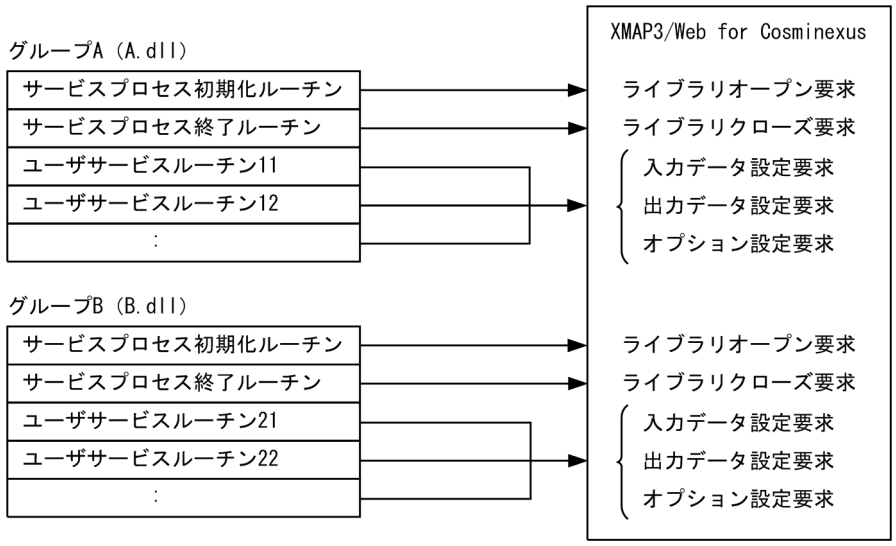
グループごとにライブラリオープン要求およびライブラリクローズ要求を呼び出し，同時に複数のライブラリオープン要求を呼び出す運用にする場合は，共通インタフェース領域をグループごとに定義する必要があります。

x3weblib.cbl に定義済みの共通インタフェース領域 XMAP-WCOM は，「EXTERNAL DYNAMIC」指定によって同一プロセスで動作するすべての COBOL のアプリケーションで共通に利用できる領域定義となります。このため，同時に複数のライブラリオープン要求を呼び出す運用にする場合は，x3weblib.cbl に XMAP-WCOM2 を定義し，グループ 1 のライブラリオープン要求には XMAP-WCOM を指定し，グループ 2 のライブラリオープン要求には XMAP-WCOM2 を指定する必要があります。

- C 言語を使用する場合

  推奨するプログラム構成の例を次の図に示します。XMAP3 TP1/Web 連携機能のライブラリオープン要求，およびライブラリクローズ要求は，グループごとに発行してください。

図 10-3　推奨するプログラム構成の例（C 言語を使用する場合）



10.2　XMAP3 TP1/Web 連携用の AP の作成（COBOL）

10.2.1　ソースプログラムの記述

AP を COBOL で作成する場合，XMAP3 Web 実行環境ライブラリの呼び出しには CALL 文を使用します。

(1)　環境部の定義

CALL 文を使用する場合，環境部（ENVIRONMENT DIVISION）の定義を次に示します。

```
ENVIRONMENT  DIVISION.
CONFIGURATION  SECTION.
    SPECIAL-NAMES.
    STDCALL  IS  呼び名1.
EXTERNAL-PROGRAM  SECTION.
    CALL-CONVENTION.
    'jsvwwlib'  IS  呼び名1.
```

注　呼び名 1 は，ユーザが任意に設定するデータ項目名です。

(2)　jsvwwlib ライブラリの呼び出し

CALL 文による XMAP3 Web 実行環境ライブラリの呼び出し形式を次に示します。

```
CALL  'jsvwwlib'  USING  XMAP-WCOM
                         XMAP-WREQ
                         データ名3
                         データ名4
                         データ名5.
```

(3)　jsvwwlib ライブラリで使用するインタフェース領域

(a)　XMAP-WCOM

XMAP-WCOM には，共通インタフェース領域を指定します。指定形式の詳細を次の表に示します。

表 10-1　XMAP-WCOM に指定する共通インタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
アイキャッチャ	4	X(4)	XMAP-WCOM-ID	'*WEB'
リターン値 1	2(4)	9(4) COMP-5	XMAP-WCOM-RTN	リターン値は 2 進形式で返されます。
リターン値 2	2(6)	9(4) COMP-5	XMAP-WCOM-RSN	
未使用	3(8)	X(3)	XMAP-WCOM-RSV1	$(00)_{16}$
id 区分	1(11)	X	XMAP-WCOM-ITYPE	'I'
仮想端末名	8(12)	X(8)	XMAP-WCOM-TNAME	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※1
未使用	4(20)	X(4)	XMAP-WCOM-RSV2	$(00)_{16}$

データ項目名	長さ	データ形式	データ名	指定内容
通信種別	4(24)	X(4)	XMAP-WCOM-MSG	画面の場合：'BWS△' 帳票の場合：'OWS△'
未使用	44(28)	X(44)	XMAP-WCOM-RSV3	$(00)_{16}$
マップ名	8(72)	X(8)	XMAP-WCOM-MAPNAME	デバイス ID※3 付きの物理マップ名を左詰めで指定し，残りは空白を指定する
未使用	8(80)	X(8)	XMAP-WCOM-RSV4	$(00)_{16}$
論理マップ長	4(88)	S9(8) COMP-5	XMAP-WCOM-LSGLNG	入出力論理マップ長または 0※2
未使用	260(92)	X(260)	XMAP-WCOM-RSV5	$(00)_{16}$

注※1

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※2

画面の場合：

・出力データの設定を要求する場合は，出力論理マップ長を指定します。

・入力データの取得を要求する場合は，入力論理マップ長を指定します。

帳票の場合：

・出力データの設定を要求する場合は，出力論理マップ長を指定します。

・入力データの取得を要求する場合は，0 を指定します。

注※3

画面のデバイス ID については「11.1.1(1)　通信記述項」を，帳票のデバイス ID については「11.2.1(1)　通信記述項」を参照してください。

(b)　XMAP-WREQ

XMAP-WREQ には，要求インタフェース領域を指定します。指定形式の詳細を次の表に示します。

表 10–2　XMAP-WREQ に指定する要求インタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
要求種別	4	X(4)	XMAP-WREQ-TYPE	要求別に指定する※1
オプション 1	1(4)	X	XMAP-WREQ-OPT1	'F'
オプション 2	1(5)	X	XMAP-WREQ-OPT2	'1'
未使用	2(6)	X(2)	XMAP-WREQ-RSV1	空白
オプション 3	1(8)	X	XMAP-WREQ-OPT3	• 画面の場合：空白 • 帳票の場合：'1'または'2'※2
オプション 4	1(9)	X	XMAP-WREQ-OPT4	'△'：画面入出力／帳票出力を応答する 'E'：ブラウザ側の XMAP3 業務の終了を応答する

データ項目名	長さ	データ形式	データ名	指定内容
未使用	6(10)	X(6)	XMAP-WREQ-RSV2	空白とする

注※1

要求動作に応じて次の値を指定します。

要求動作	指定する値
XMAP3 Web 実行環境ライブラリのオープン要求	'OPEN'
XMAP3 Web 実行環境ライブラリのクローズ要求	'CLOS'
出力データの設定要求	'SEND'
入力データの取得要求	'RECV'
オプションの設定要求	'MDO△'（△は半角空白を表す）

注※2

帳票の場合の指定値と内容を次に示します。

指定値	内容
'1'	プリンタスプールに帳票を出力後，その印刷ドキュメントを完了（クローズ）する。通常は'1'を指定する。
'2'	プリンタスプールに登録されている，印刷ドキュメント中の 1 ページとして帳票を出力する。 印刷ドキュメントは完了（クローズ）しないで継続する。

帳票印刷セットアップでスプール書き出し単位を「1 ページ毎」にした場合は，この指定は無効となり常に'1'（ドキュメントを完了（クローズ）する）となります。

(c)　データ名 3

XMAP3 Web 実行環境ライブラリのオープン要求時にオープンインタフェース領域を指定します。データ名 3 に指定する詳細を次の表に示します。

表 10-3　データ名 3 に指定するオープンインタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
サーバ環境定義ファイル名	8	X(8)	XMAP-WOPN-SRVNAME	サーバ環境定義ファイル名を左詰めで指定し，残りは空白を指定する※
未使用	8(8)	X(8)	XMAP-WOPN-RSV	空白を指定

注※

実行時に，指定したサーバ環境定義ファイルを「*XMAP3 インストールフォルダ*¥Web for TP1¥ETC」から検索します。

データ名 3 に 0 を指定した場合や，サーバ定義ファイル名に 8 けたの空白を指定した場合は，サーバ環境定義ファイルとして「X3WEBSRV」を参照します。

・出力データ設定要求時

出力論理マップを指定します。Web ブラウザ側の XMAP3 処理を終了する場合には 0 を指定します。

・入力データ取得要求時

入力論理マップを指定します。ブラウザからの 1 回目の呼び出しや帳票印刷時など，入力論理マップが存在しない場合には 0 を指定します。

・オプション設定要求時
必ず 1 を指定します。

(d)　データ名 4

出力データ設定要求時，データ名 4 には送信データ出力領域を指定します。入力データ設定要求時，データ名 4 には受信データ領域を指定します。また，オプション設定要求時，データ名 4 にはマッピングインタフェース領域（XMAP-WMDO）を指定します。データ名 4 に指定する内容を要求別に示します。

表 10-4　データ名 4 に指定する送信データ出力領域

データ項目名	長さ	データ形式	指定内容
出力領域長	4	S9(9) COMP	出力領域の長さ（n）を指定
出力領域	n(4)	X(n)	$(00)_{16}$

送信データを格納するために必要な出力領域の大きさは，次に示す概算式で求められます。

出力領域サイズ(バイト)＝(500＋URL長※1×3＋出力論理マップ長※2)×1.5

注※1
次に示す URL の長さを指します。

- サーバ環境定義ファイルに指定した URL（環境設定ファイルパス（EtcPath）やデータファイルパス（DataPath）など）の最大長
- SEND 要求や CLOSE 要求で指定する，次に呼び出す URL の最大長

注※2
AP で使用する出力論理マップの最大長を指します。

それぞれ，URL の最大長は 2,048 バイト，出力論理マップの最大長は 32,000 バイトとなっているため，上記の概算式に従えば出力領域に必要な領域サイズの最大値は次のようになります。

(500＋2,048×3＋32,000)×1.5＝57,966バイト

表 10-5　データ名 4 に指定する受信データ領域

データ項目名	長さ	データ形式	指定内容
受信データ長	4	S9(9) COMP	受信データの長さ（n）を指定
受信データ	n(4)	X(n)	TP1/Web の API で取得した受信データを指定

受信データを格納するために必要な領域の大きさは，次に示す概算式で求められます。

受信データサイズ(バイト)＝120＋(500＋入力論理マップ長※)×1.5

注※
AP で使用する入力論理マップの最大長を指します。ただし，帳票の場合は 0 となります。

入力論理マップの最大長は 32,000 バイトとなっているため，上記の概算式に従えば入力領域に必要な領域サイズの最大値は次のようになります。

120＋(500＋32,000)×1.5＝48,870バイト

表 10-6　データ名 4 に指定するマッピングインタフェース領域（XMAP-WMDO）

データ項目名	長さ	データ形式	データ名	指定内容
マッピングオプション大分類	4	9(8) COMP-5	XMAP-WMDO-OPT1	XMAP-WMDO-SFLD の内容を代入
マッピングオプション小分類	4(4)	9(8) COMP-5	XMAP-WMDO-OPT2	• XMAP-WMDO-MAPFLD マージ • XMAP-WMDO-PHFLD 物理マップだけ • XMAP-WMDO-LOGFLD 論理マップだけ

(e)　データ名 5

データ名 5 には，出力データ設定要求時，次に呼び出すサービスの URL を指定※します。SSL 通信を使用する場合は，「https://」で始まる URL を指定してください。

注※

Web ブラウザ側の XMAP3 処理を終了後に Web ブラウザを閉じる場合，要求インタフェース領域のオプション 4 に'E'を指定し，データ名 5 に 0 を指定します。

データ名 5 に指定する内容を次の表に示します。

表 10-7　データ名 5 に指定する次処理サービス領域

データ項目名	長さ	データ形式	指定内容
URL データ長	4	9(8) COMP-5	URL データの長さ（n）を指定
URL データ	n(4)	X(n)	次回の呼び出し先 URL を指定※

注※

起動 HTML の業務開始 URL に指定した，Web サーバ上の URL を指定してください。要求インタフェース領域のオプション 4 に'E'を指定した場合だけ，任意の Web サーバ上の URL を指定できます。

(4)　jsvwwlib ライブラリ使ったデータの受け渡し

(a)　XMAP3 Web 実行環境ライブラリのオープン要求

AP のプロセス開始時に，XMAP3 Web 実行環境ライブラリのオープン要求を発行します。

例 1

AP で共通のサーバ環境定義ファイルを使用する場合

```
INITIALIZE XMAP-WCOM                    共通エリアの初期化
    REPLACING NUMERIC DATA BY ZERO
    ALPHANUMERIC DATA BY LOW-VALUE.
MOVE ALL SPACE TO XMAP-WREQ.            要求エリアの初期化
MOVE 'OPEN' TO XMAP-WREQ-TYPE.          XMAP3/Web for Cosminexusライブラリの
                                                  オープン要求の代入
MOVE '*WEB' TO XMAP-WCOM-ID.            IDの代入
MOVE 'I'    TO XMAP-WCOM-ITYPE.         ID区分の代入
CALL 'jsvwwlib' USING XMAP-WCOM         共通インタフェース領域
                      XMAP-WREQ         要求インタフェース領域
                      BY VALUE 0
                      BY VALUE 0
                      BY VALUE 0.
```

例 2

AP 別に，異なるサーバ環境定義ファイルを使用する場合

```
INITIALIZE XMAP-WCOM                    共通エリアの初期化
    REPLACING NUMERIC DATA BY ZERO
    ALPHANUMERIC DATA BY LOW-VALUE.
MOVE ALL SPACE TO XMAP-WREQ.            要求エリアの初期化
MOVE ALL SPACE TO XMAP-WOPN.            オープンインタフェース領域の初期化

MOVE 'OPEN' TO XMAP-WREQ-TYPE.          XMAP3/Webライブラリのオープン要求の代入
MOVE '*WEB'     TO XMAP-WCOM-ID.            IDの代入
MOVE 'I'        TO XMAP-WCOM-ITYPE.         ID区分の代入
MOVE ファイル名 TO XMAP-WOPN-SRVNAME.       サーバ環境定義ファイル名※
CALL 'jsvwwlib' USING XMAP-WCOM         共通インタフェース領域
                      XMAP-WREQ         要求インタフェース領域
                      XMAP-WOPN         オープンインタフェース領域
                      BY VALUE 0
                      BY VALUE 0.
```

注※　ユーザサービスプログラムごとに異なるサーバ環境定義を使用する場合は，「*XMAP3 インストールフォルダ*¥ETC」下に，一意の名称でサーバ環境定義ファイルを用意してください。

(b)　入力データの取得要求

TP1/Web の API を使って Web ブラウザから取得した受信データを解析し，入力論理マップやリターンコードを取得します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例 1

画面出力後，Web ブラウザからの応答

```
MOVE 'RECV'  TO XMAP-WREQ-TYPE.         入力データ取得要求の代入
MOVE MAP001S TO XMAP-WCOM-LSGLNG.       入力論理マップ長の代入
CALL 'jsvwwlib' USING XMAP-WCOM         共通インタフェース領域
                      XMAP-WREQ         要求インタフェース領域
                      MAP001I           入力論理マップ
                      WKAREA-I          ブラウザからの受信データ
                      BY VALUE 0.
```

注　入力データ取得要求が正常終了した場合，XMAP-WCOM-LSGLNG には取得した入力論理マップデータの長さが設定されます。

例 2

帳票出力後，Web ブラウザからの応答

```
MOVE 'RECV' TO XMAP-WREQ-TYPE.              入力データ取得要求の代入
MOVE 0      TO XMAP-WCOM-LSGLNG.            '0'の代入（必須）
CALL 'jsvwwlib' USING XMAP-WCOM             共通インタフェース領域
                      XMAP-WREQ             要求インタフェース領域
                      BY VALUE 0
                      BY REFERENCE WKAREA-I ブラウザからの受信データ
                      BY VALUE 0.
```

(c)　マッピングオプションの設定要求

Web ブラウザへ画面を表示する場合，マッピングオプションを指定できます。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例

```
MOVE 'MDO '          TO XMAP-WREQ-TYPE.         オプション設定要求の代入
MOVE XMAP-WMDO-SFLD   TO XMAP-WMDO-OPT1.        マッピングオプション大分類の代入
MOVE XMAP-WMDO-LOGFLD TO XMAP-WMDO-OPT2.        マッピングオプション小分類の代入
CALL 'jsvwwlib' USING XMAP-WCOM                 共通インタフェース領域
                      XMAP-WREQ                 要求インタフェース領域
                      BY VALUE 1                '1'を指定（必須）
                      BY REFERENCE XMAP-WMDO    マッピングインタフェース領域
                      BY VALUE 0.
```

(d)　出力データの設定要求

Web ブラウザ側での業務に対して，画面表示，または帳票印刷情報を設定した応答メッセージを生成します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例 1

画面出力データを設定した応答メッセージを生成する場合

```
MOVE 'SEND'      TO XMAP-WREQ-TYPE.          出力データ設定要求の代入
MOVE 'F'         TO XMAP-WREQ-OPT1.          'F'の代入（必須）
MOVE '1'         TO XMAP-WREQ-OPT2.          '1'代入（必須）
MOVE ' '         TO XMAP-WREQ-OPT4.          1けたの空白を代入（必須）
MOVE MAP001L     TO XMAP-WCOM-LSGLNG.        出力論理マップ長の代入
MOVE 'BWS '      TO XMAP-WCOM-MSG.           通信種別の代入
MOVE '        '  TO XMAP-WCOM-TNAME.         仮想端末名（8けたの空白）の代入
MOVE 'MAP001ND'  TO XMAP-WCOM-MAPNAME.       物理マップ名の代入
MOVE データ      TO MAP0010.                 出力したいデータを出力論理マップへ代入

CALL 'jsvwwlib' USING XMAP-WCOM              共通インタフェース領域
                      XMAP-WREQ              要求インタフェース領域
                      MAP0010                出力論理マップ
                      WKAREA-O               送信データ作成領域
                      NEXTURL.               次回の呼び出し先URL領域
```

例 2

帳票出力データを設定した応答メッセージを生成する場合

```
MOVE 'SEND'      TO XMAP-WREQ-TYPE.          出力データ設定要求の代入
MOVE 'F'         TO XMAP-WREQ-OPT1.          'F'の代入（必須）
MOVE '1'         TO XMAP-WREQ-OPT2.          '1'の代入（必須）
MOVE '1'         TO XMAP-WREQ-OPT3.          スプールへの出力方法の設定※
MOVE ' '         TO XMAP-WREQ-OPT4.          1けたの空白の代入
MOVE MAP001L     TO XMAP-WCOM-LSGLNG.        出力論理マップ長の代入
MOVE 'OWS '      TO XMAP-WCOM-MSG.           通信種別の代入
MOVE 'PRT001  '  TO XMAP-WCOM-TNAME.         仮想端末名の代入
MOVE 'MAP0016G'  TO XMAP-WCOM-MAPNAME.       物理マップ名の代入
MOVE データ      TO MAP0010.                 出力したいデータの出力論理マップへの代入

CALL 'jsvwwlib' USING XMAP-WCOM              共通インタフェース領域
                      XMAP-WREQ              要求インタフェース領域
                      MAP0010                出力論理マップ
                      WKAREA-O               送信データ作成領域
                      NEXTURL.               次回の呼び出し先URL領域
```

注※　プリンタスプールに，n ページ／ドキュメントとして出力したい場合には'2'を指定します。ブラウザ側の XMAP3 環境設定で，プリンタスプールへの書き出し単位が「アプリケーション毎」を指定した場合に有効です。

例 3

Web ブラウザ側の業務終了用の応答メッセージを生成する場合

```
MOVE 'SEND' TO XMAP-WREQ-TYPE.               出力データ設定要求の代入
MOVE 'E'    TO XMAP-WREQ-OPT4.               'E'の代入（必須）
CALL 'jsvwwlib' USING XMAP-WCOM              共通インタフェース領域
                      XMAP-WREQ              要求インタフェース領域
                      BY VALUE 0
                      WKAREA-O               送信データ作成領域
                      NEXTURL.               次回の呼び出し先URL領域※
```

注※　URL 領域を指定した場合，ブラウザ業務終了後に指定した URL を呼び出します。0 を指定した場合には，ブラウザ業務終了後にブラウザを閉じます。

(e)　XMAP3 Web 実行環境ライブラリのクローズ要求

AP のプロセス終了時に，XMAP3 Web 実行環境ライブラリのクローズ要求を発行します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例

```
MOVE 'CLOS'     TO XMAP-WREQ-TYPE.        XMAP3/Web for Cosminexus
                                                ライブラリのクローズ要求の代入
CALL 'jsvwwlib' USING XMAP-WCOM           共通インタフェース領域
                      XMAP-WREQ           要求インタフェース領域
                      BY VALUE 0
                      BY VALUE 0
                      BY VALUE 0.
```

10.2.2　コンパイルと実行

コンパイルと実行には，COBOL2002 を使用してください。COBOL2002 Net Developer では，実行はできません。

(1)　コンパイル時のポイント

(a)　インタフェース領域の取り込み

XMAP3 TP1/Web 連携機能のインタフェース領域ファイル（X3WEBLIB.cbl）を AP に取り込むには，WORKING-STORAGE SECTION に COPY 文を使用して取り込んでください。

論理マップは，WORKING-STORAGE SECTION，または LINKAGE SECTION に COPY 文を指定して取り込んでください。ただし，論理マップ中に定数を展開している場合，LINKAGE SECTION には取り込めません。

なお，XMAP3/Web for Cosminexus では，インタフェース領域ファイルを提供しています。インタフェース領域ファイルが格納されているフォルダを次に示します。

XMAP3インストールフォルダ¥Web for TP1¥INCLUDE

(b)　TP1/Web を使用して Web ブラウザとデータの送受信をする

AP を作成する上で，必要な TP1/Web のインタフェースを次の表に示します。

表 10-8　TP1/Web のインタフェースとその使用方法

処理項目	使用するインタフェースと使用方法
受信データの取得	CBLDCWEB('GETQDATA')を使用して，クライアントからのデータを受信する。ここで受信したデータを jsvwwlib ライブラリの RECV 要求で指定する。
出力データの応答	CBLDCWEB('SETHDINF')を使用して，Content-type に「XMAP-CONTENT-TYPE」を設定し，jsvwwlib ライブラリの SEND 要求で生成した出力データを，CBLDCWEB('PUTANY ')を使用して送信する。
業務終了	CBLDCWEB('SSDISCON')を使用して，サービスセットを終了する。業務終了の出力データを送信したあと，サービスルーチンを終了する前に発行する。

(c)　プロジェクトの作成

TP1/Web のサービスプログラムを作成するには，COBOL2002 の開発マネージャでプロジェクトを作成するときに次の内容を選択します。

- 最終生成物の種類：ダイナミックリンクライブラリ
- プロジェクトの種類：-Dll 指定（DLL を作成する）
- DLL 呼び出し規約選択：DLL の属性を Cdecl にする

- 出力ファイル名：生成する DLL 名を指定する

(d)　コンパイラオプションの指定

次に示すコンパイラオプションを指定する必要があります。-JPN,Alnum オプションは必要に応じて指定してください。

- -Comp5
  COBOL プログラム内で COMP-5 を利用できるようにするオプション
- -JPN,Alnum
  論理マップ内で日本語項目を扱えるようにするオプション

(2)　リンケージ時のポイント

(a)　インポートライブラリの取り込み

XMAP3 TP1/Web 連携機能が提供するインポートライブラリ（X3WEBLIB.lib）と TP1/Web が提供するインポートライブラリ（LIBTP1WEB.lib）を AP に取り込み，DLL を作成します。XMAP3 TP1/Web 連携機能のライブラリが格納されているフォルダを次に示します。

XMAP3インストールフォルダ¥Web for TP1¥LIB

(b)　リンカオプションの指定

次に示すリンカオプションを指定する必要があります。

- -Dll,Cdecl
  TP1/Web で使用できる cdecl 規約の DLL を作成するためのオプション

(3)　実行時のポイント

データ有無コードは，サーバ環境定義ファイルに指定します。サーバ上で動作する AP で使用できるデータ有無コードは，AP ごとに一つだけとなります。仮想端末ごとに異なるデータ有無コードは使用できないため，注意が必要です。

AP で参照する環境変数は，Windows のコントロールパネルで設定するシステム環境変数に登録してください。

10.3　XMAP3 TP1/Web 連携用の AP の作成（C 言語）

10.3.1　ソースプログラムの記述

C 言語で AP を作成する場合，C 言語からは関数の呼び出しによって XMAP3 Web 実行環境ライブラリを呼び出します。

(1)　jsvwwlib ライブラリの呼び出し

関数呼び出しによる，XMAP3 Web 実行環境ライブラリの呼び出し形式を次に示します。

```
long APIENTRY jsvwwlib(XMAP_WCOMアドレス,
                       XMAP_WREQアドレス,
                       パラメタ3,
                       パラメタ4,
                       パラメタ5);
```

(2)　jsvwwlib ライブラリで使用するインタフェース領域

(a)　XMAP-WCOM

XMAP-WCOM には，共通インタフェース領域を指定します。指定形式の詳細を次の表に示します。

表 10-9　XMAP-WCOM に指定する共通インタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
アイキャッチャ	4	char[4]	xmap_wcom_id	'*WEB'
リターン値 1	2(4)	unsigned short	xmap_wcom_rtn	リターン値は 2 進形式で返されます。
リターン値 2	2(6)	unsigned short	xmap_wcom_rsn	
未使用	3(8)	char[3]	xmap_wcom_rsv1	$(00)_{16}$
id 区分	1(11)	char	xmap_wcom_itype	'I'
仮想端末名	8(12)	char[8]	xmap_wcom_tname	• 画面の場合：すべて空白 • 帳票の場合：仮想端末名※1
未使用	4(20)	char[4]	xmap_wcom_rsv2	$(00)_{16}$
通信種別	4(24)	char[4]	xmap_wcom_msg	画面の場合，'BWS△ 帳票の場合，'OWS△'
未使用	44(28)	char[44]	xmap_wcom_rsv3	$(00)_{16}$
マップ名	8(72)	char[8]	xmap_wcom_mapname	デバイス ID※3 付きの物理マップ名を左詰めで指定し，残りは空白を指定する
未使用	8(80)	char[8]	xmap_wcom_rsv4	$(00)_{16}$
論理マップ長	4(88)	long	xmap_wcom_lsglng	入出力論理マップ長または 0※2
未使用	260(92)	char[260]	xmap_wcom_rsv5	$(00)_{16}$

注※1

帳票の場合は，出力先となる仮想端末名を左詰めで指定し，残りは空白を指定します。

注※2

画面の場合：

・出力データの設定を要求する場合は，出力論理マップ長を指定します。

・入力データの取得を要求する場合は，入力論理マップ長を指定します。

帳票の場合：

・出力データの設定を要求する場合は，出力論理マップ長を指定します。

・入力データの取得を要求する場合は，0 を指定します。

注※3

画面のデバイス ID については，「11.1.1(1)　通信記述項」を，帳票のデバイス ID については「11.2.1(1)　通信記述項」を参照してください。

(b)　XMAP-WREQ

XMAP-WREQ には，要求インタフェース領域を指定します。指定形式の詳細を次の表に示します。

表 10–10　XMAP-WREQ に指定する要求インタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
要求種別	4	char[4]	xmap_wreq_type	要求別に指定する※1
オプション 1	1(4)	char	xmap_wreq_opt1	'F'
オプション 2	1(5)	char	xmap_wreq_opt2	'1'
未使用	2(6)	char[2]	xmap_wreq_rsv1	空白
オプション 3	1(8)	char	xmap_wreq_opt3	• 画面の場合：空白 • 帳票の場合：'1'または'2'※2
オプション 4	1(9)	char	xmap_wreq_opt4	'△'：画面入出力／帳票出力を応答する 'E'：ブラウザ側の XMAP3 業務の終了を応答する
未使用	6(10)	char[6]	xmap_wreq_rsv2	空白

注※1

要求動作に応じて次の値を指定します。

要求動作	指定する値
XMAP3 Web 実行環境ライブラリのオープン要求	'OPEN'
XMAP3 Web 実行環境ライブラリのクローズ要求	'CLOS'
出力データの設定要求	'SEND'
入力データの取得要求	'RECV'
オプションの設定要求	'MDO△'（△は半角空白を表す）

注※2

帳票の場合の指定値と内容を次に示します。

指定値	内容
'1'	プリンタスプールに帳票を出力後，その印刷ドキュメントを完了（クローズ）する。通常は'1'を指定する。
'2'	プリンタスプールに登録されている，印刷ドキュメント中の 1 ページとして帳票を出力する。 印刷ドキュメントは完了（クローズ）しないで継続する。

帳票印刷セットアップでスプール書き出し単位を「1 ページ毎」にした場合は，この指定は無効となり常に'1'（ドキュメントを完了（クローズ）する）となります。

(c)　パラメタ 3

XMAP3 Web 実行環境ライブラリのオープン要求時にオープンインタフェース領域を指定します。パラメタ 3 に指定する詳細を次の表に示します。

表 10-11　パラメタ 3 に指定するオープンインタフェース領域

データ項目名	長さ	データ形式	データ名	指定内容
サーバ環境定義ファイル名	8	char[8]	xmap_wopn_srvname	サーバ環境定義ファイル名を左詰めで指定し，残りは空白を指定※
未使用	8(8)	char[8]	xmap_wopn_rsv	空白を指定

注※

実行時に，指定したサーバ環境定義ファイルを「*XMAP3 インストールフォルダ*¥Web for TP1¥ETC」から検索します。

パラメタ 3 に 0 を指定した場合や，サーバ定義ファイル名に 8 けたの空白を指定した場合は，サーバ環境定義ファイルとして「X3WEBSRV」を参照します。

・出力データ設定要求時
　出力論理マップを指定します。Web ブラウザを終了する場合には 0 を指定します。
・入力データ取得要求時
　入力論理マップを指定します。ブラウザからの 1 回目の呼び出しや帳票印刷時など，
　入力論理マップが存在しない場合には 0 を指定します。
・オプション設定要求時
　必ず 1 を指定します。

(d)　パラメタ 4

出力データ設定要求時，パラメタ 4 には送信データ出力領域を指定します。入力データ設定要求時，パラメタ 4 には受信データ領域を指定します。また，オプション設定要求時，パラメタ 4 にはマッピングインタフェース領域（xmap_wmdo）を指定します。パラメタ 4 に指定する内容を要求別に示します。

表 10-12　パラメタ 4 に指定する送信データ出力領域

データ項目名	長さ	データ形式	指定内容
出力領域長	4	unsigned long	出力領域の長さ（n）を指定
出力領域	n(4)	char[n]	$(00)_{16}$

送信データを格納するために必要な出力領域の大きさは，次に示す概算式で求められます。

出力領域サイズ(バイト)＝(500＋URL長※1×3＋出力論理マップ長※2)×1.5

注※1

次に示す URL の長さを指します。

- サーバ環境定義ファイルに指定した URL（環境設定ファイルパス（EtcPath）やデータファイルパス（DataPath）など）の最大長
- SEND 要求や CLOSE 要求で指定する，次に呼び出す URL の最大長

注※2

AP で使用する出力論理マップの最大長を指します。

それぞれ，URL の最大長は 2,048 バイト，出力論理マップの最大長は 32,000 バイトとなっているため，上記の概算式に従えば出力領域に必要な領域サイズの最大値は次のようになります。

(500＋2,048×3＋32,000)×1.5＝57,966バイト

表 10-13　パラメタ 4 に指定する受信データ領域

データ項目名	長さ	データ形式	指定内容
受信データ長	4	unsigned long	受信データの長さ（n）を指定
受信データ	n(4)	char[n]	TP1/Web の API で取得した受信データを指定

受信データを格納するために必要な領域の大きさは，次に示す概算式で求められます。

受信データサイズ(バイト)＝120＋(500＋入力論理マップ長※)×1.5

注※

AP で使用する入力論理マップの最大長を指します。ただし，帳票の場合は 0 となります。

入力論理マップの最大長は 32,000 バイトとなっているため，上記の概算式に従えば入力領域に必要な領域サイズの最大値は次のようになります。

120＋(500＋32,000)×1.5＝48,870バイト

表 10-14　パラメタ 4 に指定するマッピングインタフェース領域（xmap_wmdo）

データ項目名	長さ	データ形式	データ名	指定内容
マッピングオプション大分類	4	unsigned long	xmap_wmdo_opt1	XMAP_WMDO_SFLD の内容を代入
マッピングオプション小分類	4(4)	unsigned long	xmap_wmdo_opt2	• XMAP_WMDO_MAPFLD マージ • XMAP_WMDO_PHFLD 物理マップだけ • XMAP_WMDO_LOGFLD 論理マップだけ

(e)　パラメタ 5

パラメタ 5 には，出力データ設定要求時，次に呼び出すサービスの URL を指定※します。SSL 通信を使用する場合は，「https://」で始まる URL を指定してください。

注※

Web ブラウザ側の XMAP3 処理を終了後に Web ブラウザを閉じる場合，要求インタフェース領域のオプション 4 に'E'を指定し，パラメタ 5 に 0 を指定します。

パラメタ 5 に指定する内容を次の表に示します。

表 10-15　パラメタ 5 に指定する次処理サービス領域

データ項目名	長さ	データ形式	指定内容
URL データ長	4	unsigned long	URL データの長さ（n）を指定
URL データ	n(4)	char[n]	次回の呼び出し先 URL を指定※

注※

起動 HTML の業務開始 URL に指定した，Web サーバ上の URL を指定してください。要求インタフェース領域のオプション 4 に'E'を指定した場合だけ，任意の Web サーバ上の URL を指定できます。

(f)　リターンコード

jsvwwlib 関数のリターンコードを次の表に示します。

表 10-16　jsvwwlib 関数のリターンコード

リターンコード	意味
0	正常終了
8	異常終了
12	パラメタ不正

(3)　jsvwwlib ライブラリを使ったデータの受け渡し

(a)　XMAP3 Web 実行環境ライブラリのオープン要求

AP のプロセス開始時に，XMAP3 Web 実行環境ライブラリのオープン要求を発行します。

例 1

AP で共通のサーバ環境定義ファイルを使用する場合

```
memset(&com, 0, sizeof(XMAP_WCOM));      共通エリアの初期化
memset(&req, ' ', sizeof(XMAP_WREQ));    要求エリアの初期化
memcpy(req.xmap_wreq_type, "OPEN",       XMAP3/Web for Cosminexusライブラリ
       sizeof(req.xmap_wreq_type));                    のオープン要求の代入
memcpy(com.xmap_wcom_id, "*WEB",         IDの代入
       sizeof(com.xmap_wcom_id));
com.xmap_wcom_itype = 'I';               ID区分の代入
jsvwwlib(&com,                           共通インタフェース領域
         &req,                           要求インタフェース領域
         0L, 0L, 0L);
```

例 2

AP 別に，異なるサーバ環境定義ファイルを使用する場合

```
memset(&com, 0, sizeof(XMAP_WCOM));          共通エリアの初期化
memset(&req, ' ', sizeof(XMAP_WREQ));        要求エリアの初期化
memset(&opn, ' ', sizeof(XMAP_WOPN));        オープンインタフェース領域
                                                        の初期化
memcpy(req.xmap_wreq_type, "OPEN",           XMAP3/Web for Cosminexus
       sizeof(req.xmap_wreq_type));          ライブラリのオープン要求の代入
memcpy(com.xmap_wcom_id, "*WEB",             IDの代入
       sizeof(com.xmap_wcom_id));
com.xmap_wcom_itype = 'I';                   ID区分の代入
memcpy(opn.xmap_wopn_srvname, ファイル名,    サーバ環境定義ファイル名※
       sizeof(opn.xmap_wopn_srvname));
jsvwwlib(&com,                               共通インタフェース領域
         &req,                               要求インタフェース領域
```

```
        &opn,                                   オープンインタフェース領域
        0L, 0L);
```

注※　ユーザサービスプログラムごとに異なるサーバ環境定義を使用する場合は，「*XMAP3 インストールフォルダ*¥ETC」下に，一意の名称でサーバ環境定義ファイルを用意してください。

(b)　入力データの取得要求

TP1/Web の API を使って Web ブラウザから取得した受信データを解析し，入力論理マップやリターンコードを取得します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例 1

画面出力後，Web ブラウザからの応答

```
memcpy(req.xmap_wreq_type, "RECV"        入力データ取得要求の代入
       sizeof(req.xmap_wreq_type));
com.xmap_wcom_lsglng =                   入力論理マップ長の代入
       sizeof(MAP001I.MAP001S);
jsvwwlib(&com,                           共通インタフェース領域
         &req,                           要求インタフェース領域
         &MAP001I,                       入力論理マップ
         &wkarea_I,                      ブラウザからの受信データ
         0L);
```

注　入力データ取得要求が正常終了した場合，xmap_wcom_lsglng には取得した入力論理マップデータの長さが設定されます。

例 2

帳票出力後，Web ブラウザからの応答

```
memcpy(req.xmap_wreq_type, "RECV"        入力データ取得要求の代入
       sizeof(req.xmap_wreq_type));
com.xmap_wcom_lsglng = 0;                '0'の代入（必須）
jsvwwlib(&com,                           共通インタフェース領域
         &req,                           要求インタフェース領域
         0L,
         &wkarea_I,                      ブラウザからの受信データ
         0L);
```

(c)　マッピングオプションの設定要求

Web ブラウザへ画面を表示する場合，マッピングオプションを指定できます。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例

```
memcpy(req.xmap_wreq_type, "MDO ",       オプション設定要求の代入
       sizeof(req.xmap_wreq_type));
mdo.xmap_wmdo_opt1 = XMAP_WMDO_SFLD;     マッピングオプション大分類の代入
mdo.xmap_wmdo_opt2 = XMAP_WMDO_LOGFLD;   マッピングオプション小分類の代入
jsvwwlib(&com,                           共通インタフェース領域
         &req,                           要求インタフェース領域
         1L,                             '1'を指定（必須）
         &mdo,                           マッピングインタフェース領域
         0L);
```

(d)　出力データの設定要求

Web ブラウザ側での業務に対して，画面表示，または帳票印刷情報を設定した応答メッセージを生成します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例 1

画面出力データを設定した応答メッセージを生成する場合

```
memcpy(req.xmap_wreq_type, "SEND",                出力データ設定要求の代入
       sizeof(req.xmap_wreq_type));
```

```
req.xmap_wreq_opt1 = 'F';                          'F'の代入（必須）
req.xmap_wreq_opt2 = '1';                          '1'代入（必須）
req.xmap_wreq_opt4 = ' ';                          1けたの空白を代入（必須）
com.xmap_wcom_lsglng = sizeof(MAP0010.MAP001L);    出力論理マップ長の代入
memcpy(com.xmap_wcom_msg, "BWS ",                  通信種別の代入
       sizeof(com.xmap_wcom_msg));
memcpy(com.xmap_wcom_tname, "        ",            仮想端末名（8けたの空白）
       sizeof(com.xmap_wcom_tname));                          の代入
memcpy(com.xmap_wcom_mapname, "MAP001ND",          物理マップ名の代入
       sizeof(com.xmap_wcom_mapname));
jsvwwlib(&com,                                     共通インタフェース領域
         &req,                                     要求インタフェース領域
         &MAP0010,                                 出力論理マップ
         &wkarea_o,                                送信データ作成領域
         &nexturl);                                次回の呼び出し先URL領域
```

例 2

帳票出力データを設定した応答メッセージを生成する場合

```
memcpy(req.xmap_wreq_type, "SEND",                 出力データ設定要求の代入
       sizeof(req.xmap_wreq_type));
req.xmap_wreq_opt1 = 'F';                          'F'の代入（必須）
req.xmap_wreq_opt2 = '1';                          '1'の代入（必須）
req.xmap_wreq_opt3 = '1';                          スプールへの出力方法の設定※
req.xmap_wreq_opt4 = ' ';                          1けたの空白の代入
com.xmap_wcom_lsglng = sizeof(MAP0010.MAP001L);    出力論理マップ長の代入
memcpy(com.xmap_wcom_msg, "OWS ",                  通信種別の代入
       sizeof(com.xmap_wcom_msg));
memcpy(com.xmap_wcom_tname, "PRT001  ",            仮想端末名の代入
       sizeof(com.xmap_wcom_tname));
memcpy(com.xmap_wcom_mapname, "MAP0016G",          物理マップ名の代入
       sizeof(com.xmap_wcom_mapname));
jsvwwlib(&com,                                     共通インタフェース領域
         &req,                                     要求インタフェース領域
         &MAP0010,                                 出力論理マップ
         &wkarea_o,                                送信データ作成領域
         &nexturl);                                次回の呼び出し先URL領域
```

注※　プリンタスプールに，n ページ／ドキュメントとして出力したい場合には'2'を指定します。ブラウザ側の XMAP3 環境設定で，プリンタスプールへの書き出し単位が「アプリケーション毎」を指定した場合に有効です。

例 3

Web ブラウザ側の業務終了用の応答メッセージを生成する場合

```
memcpy(req.xmap_wreq_type, "SEND",         出力データ設定要求の代入
       sizeof(req.xmap_wreq_type));
req.xmap_wreq_opt4 = 'E';                  'E'の代入（必須）
jsvwwlib(&com,                             共通インタフェース領域
         &req,                             要求インタフェース領域
         0L,
         &wkarea_o,                        送信データ作成領域
         &nexturl);                        次回の呼び出し先URL領域※
```

注※　URL 領域を指定した場合，ブラウザ業務終了後に指定した URL を呼び出します。0 を指定した場合には，ブラウザ業務終了後にブラウザを閉じます。

(e)　XMAP3 Web 実行環境ライブラリのクローズ要求

AP のプロセス終了時に，XMAP3 Web 実行環境ライブラリのクローズ要求を発行します。XMAP3 Web 実行環境ライブラリのオープン要求で使用したインタフェース領域の指定が必要です。

例

```
memcpy(req.xmap_wreq_type, "CLOS",         XMAP3/Web for Cosminexus
       sizeof(req.xmap_wreq_type));              ライブラリのクローズ要求の代入
jsvwwlib(&com,                             共通インタフェース領域
         &req,                             要求インタフェース領域
         0L, 0L, 0L);
```

10.3.2　コンパイルと実行

(1)　コンパイル時のポイント

(a)　インタフェース領域の取り込み

XMAP3 TP1/Web 連携機能のインタフェース領域ファイル（X3WEBLIB.h）を AP に取り込むには，#include 文を使用して取り込んでください。

なお，XMAP3/Web for Cosminexus では，インタフェース領域ファイルを提供しています。インタフェース領域ファイルが格納されているフォルダを次に示します。

XMAP3 インストールフォルダ¥Web for TP1¥INCLUDE

(b)　TP1/Web を使用して Web ブラウザとデータの送受信をする

AP を作成する上で，必要な TP1/Web インタフェースを次の表に示します。この表に示す関数を使用するには，TP1/Web が提供するヘッダファイル（dcweb.h）を取り込む必要があります。

表 10-17　TP1/Web のインタフェースとその使用方法

処理項目	使用するインタフェースと使用方法
受信データの取得	dc_web_get_query_data 関数を使用して，クライアントからのデータを受信します。ここで受信したデータを jsvwwlib ライブラリの RECV 要求で指定します。
出力データの応答	dc_web_set_header_inf 関数を使用して，Content-type に「XMAP_CONTENT_TYPE」を設定し，jsvwwlib ライブラリの SEND 要求で生成した出力データを，dc_web_put_any_data 関数を使用して送信します。
業務終了	dc_web_service_set_disconnect 関数を使用して，サービスセットを終了します。業務終了の出力データを送信したあと，サービスルーチンを終了する前に発行します。

(2)　リンケージ時のポイント

(a)　インポートライブラリの取り込み

XMAP3 が提供するインポートライブラリ（X3WEBLIB.lib）と TP1/Web が提供するインポートライブラリ（LIBTP1WEB.lib）を，AP に取り込み，DLL を作成します。XMAP3 Web 実行環境ライブラリが格納されているフォルダを次に示します。

XMAP3 インストールフォルダ¥Web for TP1¥LIB

(b)　モジュール定義ファイルの作成

生成する DLL の関数を外部から呼び出せるようにするため，モジュール定義ファイル（.def）を作成し，取り込む必要があります。モジュール定義ファイルの作成例を次に示します。

- モジュール定義ファイル（module.def）

```
EXPORTS
    function1
    function2
    function3
```

(3)　実行時のポイント

データ有無コードは，サーバ環境定義ファイルに指定します。サーバ上で動作する AP で使用できるデータ有無コードは，AP ごとに一つだけとなります。仮想端末ごとに異なるデータ有無コードは使用できないため，注意が必要です。

AP で参照する環境変数は，サーバ側の OS で設定するシステム環境変数に登録してください。

11 COBOL の入出力命令

この章では COBOL を用いて，画面の入出力および帳票と書式を印刷する方法について説明します。

11.1　画面の入出力命令（COBOL）

ここでは，COBOL を用いて，画面を入出力する方法について説明します。

画面用 AP で使用する COBOL 用の API の一覧を次に示します。

表 11-1　画面用 AP で使用する COBOL 用の API 一覧

項番	COBOL の命令文	機能概要	備考
1	SEND 文	表示要求	最初の SEND 文はオープン要求となる。
2	RECEIVE 文	入力要求	なし。
3	TRANSCEIVE 文	表示および入力要求	最初の TRANSCEIVE 文はオープン要求となる。
4	DISABLE 文	クローズ要求	次の命令文でもクローズ要求となる。 GOBACK 文, EXIT PROGRAM 文, CANCEL 文 （AP の）STOP RUN
5	CALL 文※	CALL 命令文で，オープン，オプション設定，出力，入力，およびクローズ要求	要求は引数で指定する。

注

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

注※

COBOL の CALL 文は，Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）で動作する AP では利用できません。

11.1.1　画面表示命令（COBOL）

画面の入出力をするには，通信記述項を記載したあとで，SEND 文，RECEIVE 文，TRANSCEIVE 文，および DISABLE 文を使用します。ここでは指定例を用いて画面の入出力を説明します。

(1)　通信記述項

画面の入出力に必要な情報を COMMUNICATION SECTION 中に指定します。

例

```
COMMUNICATION SECTION.
 *
      ┌──────────────────────────────────────────────────────────── 1
     ┌┘
  CD DSP FOR I-O WS ······························································ 2
   MAP NAME IS 画面マップ名 ····························································· 3

   SYMBOLIC TERMINAL    IS  画面端末名. ····························· 4
   MAPPING MODE         IS  マッピングモード  ····· 5
     [STATUS KEY        IS  画面-RC] ····· 6
     [DATA ABSENCE CODE IS  データ有無コード]  ····· 7
```

注

1，2，3 および 5 は必ず指定してください。ほかの項目は，必要に応じて指定してください。

例の説明

1. **通信記述名（DSP）**（必須）

   SEND 文，RECEIVE 文，および TRANSCEIVE 文で使用する名称です。

2. **通信種別（I-O WS）**（必須）

   ディスプレイ用であることを示します。

3. **物理マップ名格納エリア**（必須）

   SEND 文，RECEIVE 文，および TRANSCEIVE 文で出力，または入力したい場合に物理マップ名を格納するエリアです。格納するマップ名には，次に示すデバイス ID を付ける必要があります。

   ●GUI 画面の場合

   マップ名 6 文字：ND

   マップ名 7 文字：O

   ●CUI 画面の場合

   マップ名 6 文字：NC

   マップ名 7 文字：S

4. **仮想端末名格納エリア**

   最初の SEND 文および TRANSCEIVE 文を実行する前に，**仮想端末名**を格納しておくエリアです。正しい名称を指定しないと，実行時に表示先が決定できなくなりエラーになります。スタンドアロンの場合は，標準の「DSP001」を指定します。C/S システムの場合は，C/S セットアップで指定した仮想端末名を指定してください。

   AP 中で仮想端末名を指定していない場合，仮想端末名の先頭 1 バイト目にスペースを指定している場合，または「SYMBOLIC TERMINAL IS」を省略している場合，環境変数「CBLTERMID」で指定された仮想端末名が仮定されます。環境変数「CBLTERMID」も指定されていない場合は，サーバ上に画面を表示します。

   また，環境変数「CBLTERM_xxx」を指定することで，出力先の仮想端末名を変更することもできます。

   CBLTERMID，CBLTERM_xxx については，マニュアル「COBOL2002 ユーザーズガイド」を参照してください。

5. **マッピングオプション**（必須）

   論理マップ，または物理マップのマッピング方式を指定します。画面を表示するときのオプションとして，ディスプレイ用に次に示すものが指定できます。

目的	MAPPING MODE 句の指定値
2 回目または 1 回目でユーザデータを表示したいとき	△，または 0〈マージ〉
同じ画面に対して一部上書をしたいとき	2〈論理マップだけ〉
ユーザデータなしで 1 回目（初期）表示したいとき	3〈物理マップだけ〉

   詳細については，「3.2　マッピングオプション」を参照してください。

6. **リターンコード格納エリア**

   SEND 文，RECEIVE 文，および TRANSCEIVE 文のリターンコードを格納するエリアです。
   SEND 文，RECEIVE 文，および TRANSCEIVE 文が正常に処理されたかを判定する場合，このエリアを参照します。STATUS KEY 句のリターンコードの意味を次に示します。

コード	意味	状態／要因／対処方法
'00000'	正常終了	データの入出力が正常に終了した。

コード	意味	状態／要因／対処方法
'00008'	論理エラー	COBOL の AP に誤りがある。 AP に誤りがないかを調べて修正する。
'10004' '10008'	マッピングエラー	画面データのマッピング中にエラーが発生した。STATUS KEY 句の指定がない場合，実行を中止する。 STATUS KEY 句の指定がある場合，処理を続行する。画面入出力時のリターンコードの内容を調査して対処する必要がある※。
'20008'	入出力エラー	画面データの入出力中にエラーが発生した。STATUS KEY 句の指定がない場合，実行を中止する。 STATUS KEY 句の指定がある場合，処理を続行する。画面入出力時のリターンコードの内容を調査して対処する必要がある※。

注※

画面入力時のリターンコードの内容については，マニュアル「XMAP3 実行ガイド」を参照してください。

なお，これらのリターンコードは COBOL のコンソール画面に表示されます。

7. データ有無コード格納エリア

データ有無コードを変更したい場合には，最初の SEND 文を実行する前にデータ有無コードを格納しておく必要があります。「DATA ABSENCE CODE IS」を省略すると，データ有無コードとして標準の$(1F)_{16}$が仮定されます。なお，データ有無コードはすべての AP にかかわるコードなので，標準値を利用することをお勧めします。

(2)　SEND 文

ディスプレイに画面を表示します。画面表示時に変更したいデータは，出力論理マップへ格納したあとに実行します。

画面の送信から受信までの間に AP でほかの処理をする場合などに使用します。この場合，表示画面への入力操作を抑止するために，ドローの定義で，画面属性の「キーボードのロック状態を解除する」のチェックを外した画面を作成してください。

この画面を最初に SEND 文で表示し，AP 側の処理が終了して表示画面への入力操作ができるようになったとき，AP からウィンドウ属性の「キーボードのロック状態を解除する」の修飾名を指定して，SEND／RECEIVE 文（または TRANSCEIVE 文）を発行してください。

なお，二次ウィンドウ，または二次ウィンドウがある一次ウィンドウの場合の送受信には，SEND 文を使用しないで TRANSCEIVE 文を使用してください。

例

```
PROCEDURE DIVISION.
   :
 MOVE 'MAP003ND' TO 画面マップ名  … 1.マップ格納エリアにマップ名
                                    'MAP003ND'を格納
 SEND DSP                        … 2.通信記述名を指定(ディスプレイ用)
    FROM MAP0030                 … 3.出力用論理マップ名
    WITH EMI                     … 4.必ず指定すること
```

(3)　RECEIVE 文

ディスプレイ端末からの画面入力を行います。画面入力の結果は，入力論理マップに格納されます。SEND 文と対で指定してください。

例

```
PROCEDURE DIVISION.
   :
 SEND DSP
  :
```

```
RECEIVE DSP ····· 1.SEND文と同じ通信記述名を指定
    FIRST SEGMENT ····· 2.必ず指定すること
    INTO MAP003I  ····· 3.入力用論理マップ名
```

(4)　TRANSCEIVE 文

ディスプレイ端末での画面の入出力をまとめて行います。SEND 文と RECEIVE 文の機能を持ちます。通常の画面の送受信には，TRANSCEIVE 文を使用してください。

例

```
PROCEDURE DIVISION.
  :
MOVE 'MAP003ND' TO 画面マップ名 …  1.マップ名格納エリアに物理マップ名
                                    'MAP003ND'を格納
 TRANSCEIVE DSP       …  2.通信記述名を指定(ディスプレイ用)
       FROM MAP0030   …  3.出力用論理マップ名を指定
       INTO MAP003I   …  4.入力用論理マップ名を指定
```

(5)　DISABLE 文

AP 中の最初の SEND 文または，TRANSCEIVE 文が発行されたとき，ディスプレイをオープンし，プログラム終了までディスプレイを解放（クローズ）しません。ディスプレイを明示的に解放（クローズ）したい場合には，DISABLE 文を使用します。

「例」のようにコーディングすると，次のオープン命令が発行された場合，ディスプレイ上に表示されているウィンドウは消え，新たにウィンドウが表示されます。

例

```
PROCEDURE DIVISION.
  :
SEND DSP
  :
DISABLE DSP          … 1.通信記述名を指定(ディスプレイ用)
```

11.1.2　CALL 文による方法

COBOL の CALL 文で AP を作成するときは，CALL 命令を発行します。CALL 命令でのオープン要求，オプション設定要求，出力要求，入力要求，およびクローズ要求の方法を説明します。

CALL 文による方法は，Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）で動作する AP では利用できません。

(1)　CALL 命令

CALL 命令の形式を次に示します。

```
CALL 'jsvwadrv' USING XMAP-COM XMAP-REQ データ名3 データ名4.
```

1.XMAP-COM

共通インタフェース領域を指定します。この領域は，画面を出力したい端末単位に作成し，オープンしたときの領域をクローズ要求まで引き継いで使用します。また，次に示す各要求時にマッピング，および入出力に必要な情報を代入します。

- オープン要求時　：仮想端末名，通信種別を指定する。
- 出力要求時　　：物理マップ名を指定する。
- 入力要求時　　：入力論理マップ長を指定する。

共通インタフェース領域の形式の詳細を次の表に示します。

表 11–2　共通インタフェース領域の形式（XMAP-COM，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
アイキャッチャ	4	X(4)	'＊XP△'とする（XMAP-COM-ID）。
リターンコード	2(4)	9(4)COMP-5	2 進形式で格納（XMAP-COM-RTN）。
リターンコード詳細 （画面入出力時のリターンコード）	2(6)	9(4)COMP-5	2 進形式で格納（XMAP-COM-RSN）。
未使用	3(8)	X(3)	$(00)_{16}$ とする（XMAP-COM-RSV1）。
id 区分	1(11)	X	必ず'I'とする（XMAP-COM-ITYPE）。
仮想端末名	8(12)	X(8)	スタンドアロンの場合，標準の「DSP001」を指定する。C/S システムの場合，出力先の仮想端末名を指定する（XMAP-COM-TNAME）。
未使用	4(20)	X(4)	$(00)_{16}$ とする（XMAP-COM-RSV2）。
通信種別	4(24)	X(4)	ディスプレイ要求時は'BWS△'とする（XMAP-COM-MSG）。
未使用	44(28)	X(44)	$(00)_{16}$ とする（XMAP-COM-RSV3）。
マップ名	8(72)	X(8)	物理マップ名を左詰めでデバイス ID※付きで指定する。残りは空白（XMAP-COM-MAPNAME）。
未使用	8(80)	X(8)	$(00)_{16}$ とする（XMAP-COM-RSV4）。
入力論理マップ長	4(88)	S9(8)COMP-5	入力論理マップ長を指定する（XMAP-COM-INLNG）。
未使用	68(92)	X(68)	$(00)_{16}$ とする（XMAP-COM-RSV5）。

注※

画面のデバイス ID については「11.1.1(1)　通信記述項」を参照してください。

2. XMAP-REQ

要求インタフェース領域を指定します。形式を次の表に示します。この領域は，必ず指定してください。この領域を指定した内容によって，要求種別が決まります。

表 11–3　要求インタフェース領域の形式（XMAP-REQ，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
要求種別	4	X(4)	'OPEN'：オープン要求 'CLOS'：クローズ要求 'SEND'：出力要求 'RECV'：入力要求 'MDO△'：オプション設定要求 （XMAP-REQ-TYPE）
RECEIVE オプション	1(4)	X	'F'：入力要求時 '△'：上記以外 （XMAP-REQ-OPT1）
SEND	1(5)	X	'1'：出力要求時

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
オプション			'△'：上記以外 （XMAP-REQ-OPT2）
未使用	2(6)	X(2)	'△△'とする（XMAP-REQ-RSV）

3. データ名 3

- 出力要求時　　：出力論理マップを指定する。
- 入力要求時　　：入力論理マップを指定する。
- クローズ要求時　：0L を指定する。
- オプション設定要求時：1L を指定する。
- オープン要求時　：オープンインタフェース領域のアドレスを必ず指定する。

オープンインタフェース領域の形式の詳細を次の表に示します。

表 11–4　オープンインタフェース領域の形式（XMAP-OPN，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
データ有無 コード使用選択	1	X	'1'　：データ有無コードを指定する その他 ：データ有無コードを指定しない（標準値「1F」を仮定） （XMAP-OPN-DCODE-SET）
データ有無 コード	1(1)	X	データ有無コードを指定する場合，2 桁の 16 進数（00～FF）で指定する （XMAP-OPN-DCODE）
未使用	2(2)	X(2)	'△△'とする（XMAP-OPN-RSV）

なお，データ有無コード使用選択項目でデータ有無コードを無効にする指定をした場合は，標準値「1F」が仮定されます。

4. データ名 4

オプション設定要求時は，マッピングインタフェース領域を指定します。形式の詳細を次の表に示します。なお，マッピングオプションを指定しない場合は，「0」を仮定します。

表 11–5　マッピングインタフェース領域の形式（XMAP-MDO，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
マッピング オプション 大分類	4	9(8)COMP-5	XMAP-MDO-SFLD の内容を代入する （XMAP-MDO-OPT1）
マッピング オプション 小分類 (MAPPING MODE 句相当)	4(4)	9(8)COMP-5	XMAP-MDO-MAPFLD※ XMAP-MDO-PHFLD※ XMAP-MDO-LOGFLD※ （XMAP-MDO-OPT2）

注※

指定内容とマッピングオプションは次のように対応しています。なお，マッピングオプションについては，「3.2 マッピングオプション」を参照してください。

・XMAP-MDO-MAPFLD：マージ

・XMAP-MDO-PHFLD：物理マップだけ
・XMAP-MDO-LOGFLD：論理マップだけ

(2)　インタフェース領域の取り込み方法

XMAP3 が提供しているインタフェース領域（JSVWATBL.CBL）を AP に取り込む場合，COBOL の WORKING-STORAGE SECTION，または LINKAGE SECTION に COPY 文を指定します。また，コンパイルする前に JSVWATBL.CBL のフォルダ名を CBLLIB に設定しておく必要があります。

インタフェース領域を次の図に示します。

図 11-1　インタフェース領域

```
*
* COMMON INTERFACE AREA
 01 XMAP-COM.
  03 XMAP-COM-ID          PIC  X(4).
  03 XMAP-COM-RTN         PIC  9(4)  COMP-5.
  03 XMAP-COM-RSN         PIC  9(4)  COMP-5.
  03 XMAP-COM-RSV1        PIC  X(3).
  03 XMAP-COM-ITYPE       PIC  X.
  03 XMAP-COM-TNAME       PIC  X(8).
  03 XMAP-COM-RSV2        PIC  X(4).
  03 XMAP-COM-MSG         PIC  X(4).
  03 XMAP-COM-RSV3        PIC  X(44).
  03 XMAP-COM-MAPNAME     PIC  X(8).
  03 XMAP-COM-RSV4        PIC  X(8).
  03 XMAP-COM-INLNG       PIC  S9(8)  COMP-5.
  03 XMAP-COM-RSV5        PIC  X(68).
*
* REQUEST INTERFACE AREA
 01 XMAP-REQ.
  03 XMAP-REQ-TYPE        PIC  X(4).
  03 XMAP-REQ-OPT1        PIC  X.
  03 XMAP-REQ-OPT2        PIC  X.
  03 XMAP-REQ-RSV         PIC  X(2).
*
* OPEN INTERFACE AREA
 01 XMAP-OPN.
  03 XMAP-OPN-DCODE-SET     PIC  X.
  03 XMAP-OPN-DCODE       PIC  X.
  03 XMAP-OPN-RSV         PIC  X(2).
*
* MAPPING OPTION INTERFACE AREA
 01 XMAP-MDO.
  03 XMAP-MDO-OPT1        PIC  9(8)  COMP-5.
  03 XMAP-MDO-OPT2        PIC  9(8)  COMP-5.
*
* MAPPING OPTION VALUE
 77  XMAP-MDO-SFLD         PIC  9(8)  COMP  VALUE  3.
 77  XMAP-MDO-MAPFLD       PIC  9(8)  COMP  VALUE 13.
 77  XMAP-MDO-PHFLD        PIC  9(8)  COMP  VALUE 14.
 77  XMAP-MDO-LOGFLD       PIC  9(8)  COMP  VALUE 15.
```

(3)　オープン要求

CALL 命令を使用する場合,画面を表示したい端末に付けた仮想端末名単位にオープン要求をします。複数の仮想端末（ディスプレイ）をオープン要求する場合，インタフェース領域は，仮想端末ごとに準備してください。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

例

```
   PROCEDURE DIVISION.
      .
      .
    MOVE 0    TO XMAP-COM.          … 1.共通エリアをクリアする
    MOVE SPACE  TO XMAP-REQ.        … 2.要求エリアをクリアする
   *
```

```
MOVE 'OPEN' TO XMAP-REQ-TYPE.    … 3.オープン要求を代入する
MOVE '*XP△' TO XMAP-COM-ID.      … 4.IDを代入する
MOVE 'I'    TO XMAP-COM-ITYPE.   … 5.ID区分を代入する
MOVE 'BWS△' TO XMAP-COM-MSG.     … 6.ディスプレイの通信種別を代入する
MOVE 'DSP001△△' TO XMAP-COM-TNAME.   … 7.仮想端末名を代入する

MOVE '1'    TO XMAP-OPN-DCODE-SET.
MOVE X'1F'  TO XMAP-OPN-DCODE.       … 8.データ有無コードを代入する
MOVE SPACE  TO XMAP-OPN-RSV.
CALL 'jsvwadrv' USING XMAP-COM           … 9.共通インタフェース領域
           XMAP-REQ                  … 10.要求インタフェース領域
           XMAP-OPN                  … 11.オープンインタフェース領域
           BY VALUE 0.
```

(4)　オプション設定要求

仮想端末へ画面表示をする場合,マッピングオプションを指定できます。オープン要求で使用した共通インタフェースを引き継いで使用します。

例

```
MOVE 'MDO△' TO XMAP-REQ-TYPE.      … 1.オプション設定要求を代入する
MOVE '△'    TO XMAP-REQ-OPT1.      … 2.必ず空白を代入する
MOVE '△'    TO XMAP-REQ-OPT2.      … 3.必ず空白を代入する
MOVE XMAP-MDO-SFLD  TO XMAP-MDO-OPT1. … 4.マッピングオプション
                                         大分類を代入する
MOVE XMAP-MDO-LOGFLD TO XMAP-MDO-OPT2. … 5. マッピングオプション
                                         小分類を代入する
CALL 'jsvwadrv' USING XMAP-COM     … 6.共通インタフェース領域
           XMAP-REQ                … 7.要求インタフェース領域
        BY VALUE 1                 … 8.必ず1を指定する
        BY REFERENCE XMAP-MDO.     … 9. マッピングオプションインタフェース領域
```

(5)　出力要求

オープン要求した仮想端末へ画面を表示します。オープン要求で使用した共通インタフェース領域を引き継いで使用します。

例

```
MOVE 'SEND'  TO XMAP-REQ-TYPE. … 1.出力要求を代入する
MOVE '△'     TO XMAP-REQ-OPT1. … 2.出力要求時に必ず空白を代入する
MOVE '1'     TO XMAP-REQ-OPT2.  … 3.出力要求時に必ず'1'を代入する
MOVE 'MAP003NC'TO XMAP-COM-MAPNAME … 4.物理マップ名を代入する
MOVE データ  TO MAP0030.           … 5.画面表示したいデータを
                                        出力論理マップへ代入する
CALL 'jsvwadrv' USING XMAP-COM     … 6.共通インタフェース領域
           XMAP-REQ                … 7.要求インタフェース領域
           MAP0030                 … 8.出力論理マップ
           BY VALUE 0.
```

(6)　入力要求

入力要求によって,画面から入力した情報を AP へ代入します。画面表示要求で使用したインタフェース領域を引き継いで使用します。

例

```
MOVE 'RECV' TO XMAP-REQ-TYPE.    … 1.入力要求を代入する
MOVE 'F'    TO XMAP-REQ-OPT1.    … 2.入力要求時に必ず'F'を代入する
MOVE '△'    TO XMAP-REQ-OPT2.    … 3.入力要求時に必ず空白を代入する
MOVE map003S TO XMAP-COM-INLNG. … 4.入力論理マップ長を代入する
*
CALL 'jsvwadrv' USING XMAP-COM  … 5.共通インタフェース
           XMAP-REQ             … 6.要求インタフェース
           MAP003I              … 7.入力論理マップ
           BY VALUE 0.
IF MAP003-INCI                  … 8.入力論理マップのデータを判定する
```

(7)　クローズ要求

仮想端末をクローズします。オープン要求，出力要求，および入力要求で使用したインタフェースを引き継いで使用します。

例

```
MOVE ’CLOS’ TO XMAP-REQ-TYPE.    … 1.クローズ要求を代入する
MOVE ’△’   TO XMAP-REQ-OPT1.    … 2.必ず空白を代入する
MOVE ’△’   TO XMAP-REQ-OPT2.    … 3.必ず空白を代入する
CALL ’jsvwadrv’ USING XMAP-COM   … 4.共通インタフェース領域
            XMAP-REQ             … 5.要求インタフェース領域
            BY VALUE 0 0.
```

11.2　帳票の印刷命令（COBOL）

ここでは，COBOL を用いて帳票を印刷する方法について説明します。

AP からプリンタに対して帳票を印刷するには，次に示す方法があります。

- COBOL の SEND 文による方法
- COBOL の CALL 文による方法

帳票用 AP で使用する COBOL 用の API の一覧を次に示します。

表 11-6　帳票用 AP で使用する COBOL 用の API 一覧

項番	COBOL の命令文	機能概要	備考
1	SEND 文	印刷要求	最初の SEND 文はオープン要求となる。
2	DISABLE 文	クローズ要求	次の命令文でもクローズ要求となる。 GOBACK 文, EXIT PROGRAM 文, CANCEL 文 （AP の）STOP RUN
3	CALL 文※	CALL 命令文で，オープン，出力，およびクローズ要求	要求は引数で指定する。

注

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

注※

COBOL の CALL 文は，Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）で動作する AP では利用できません。

11.2.1　SEND 文による印刷

ここでは，COBOL の SEND 文で帳票を印刷する方法について，指定例を使用して説明します。

(1)　通信記述項

帳票の印刷に必要な情報を COMMUNICATION SECTION 中に指定します。

例

```
COMMUNICATION SECTION.
*    ┌──────────────────────────────────────────────────────── 1.
     ┌┐
CD PRT FOR OUTPUT WS            ──────────────────────────────── 2.
  MAP NAME          IS 帳票マップ名    ──────────────────────── 3.
  DATA ABSENCE CODE IS データ有無コード    ────────────────────── 4.
  STATUS KEY        IS 帳票-RC         ──────────────────────── 5.
  SYMBOLIC TERMINAL IS 帳票端末名.     ──────────────────────── 6.
```

例の説明

1.通信記述名

SEND 文で使用する名称を定義します。ここでは例として「PRT」を指定します。

PRT：プリンタ用

2. 通信種別

通信種別を定義します。必ず「FOR OUTPUT WS」と指定してください。

3. 物理マップ名格納エリア

SEND 文で出力する物理マップ名を格納する領域を定義します。格納するマップ名には，ドローで指定したマップ名に，次に示す ID を付けて格納します。この ID をデバイス ID といいます。

定義対象	マップ名が 6 文字のとき	マップ名が 7 文字のとき
けい線帳票（カット紙／連続紙）	6A	P
プレプリント帳票（カット紙／連続紙）	6H	L
網掛け帳票	6B	R
グラフィック帳票	6G	G
書式オーバレイ	6G	F

4. データ有無コード格納エリア

データ有無コードを設定するときにデータ有無コードを格納する領域を定義します。データ有無コードは，最初の SEND 文の前に必ず格納しておきます。なお，省略すると$(1F)_{16}$が仮定されます。

5. リターンコード格納エリア

SEND 文のリターンコードを格納する領域を定義します。AP 中で SEND 文が正常に処理されたかどうかを判定するときに，この領域を参照します。STATUS KEY 句のリターンコードを次の表に示します。

表 11–7　STATUS KEY 句のリターンコード

コード	意味	状態／要因／対処方法
'00000'	正常終了	データの出力が正常に終了しました。
'00008'	論理エラー	COBOL の AP に誤りがあります。AP の誤りを調べて修正します。
'10008'	マッピングエラー	帳票データのマッピング中にエラーが発生しました。 STATUS KEY 句の指定がない場合，実行を中止します。 STATUS KEY 句の指定がある場合，処理を続行します。エラーコードの内容を調査して対処する必要があります※。
'20008'	出力エラー	帳票データの出力中にエラーが発生しました。 STATUS KEY 句の指定がない場合，実行を中止します。 STATUS KEY 句の指定がある場合，処理を続行します。エラーコードの内容を調査して対処する必要があります※。

注※

エラーコードの内容については，「付録 F　リターンコードと詳細コード」を参照してください。

6. 仮想端末名格納エリア

出力するプリンタを指定したい場合に，プリンタに対応づけてある仮想端末名を格納する領域を定義します。

仮想端末名は，最初の SEND 文の前に指定する必要があります。XMAP3 実行環境の表示・印刷セットアップ，または C/S セットアップで指定します。

仮想端末名について，次に示します。

1 台目の標準プリンタ：PRT001

2 台目以降のプリンタ：PRT002～（環境設定が必要）

なお，AP 中で仮想端末名を指定していない場合，仮想端末名の先頭 1 バイト目にスペースを指定している場合，または「SYMBOLIC TERMINAL IS」を省略している場合，環境変数「CBLPRNTID」で指定された仮想端末名が仮定されます。環境変数「CBLPRNTID」も指定されていない場合は，仮想端末として画面を仮定して動作するため，デバイス不一致（$(1400)_{16}$）エラーとなります。

また，環境変数「CBLPRNT_xxx」を指定することで，出力先の仮想端末名を変更することもできます。

CBLPRNTID，CBLPRNT_xxx については，マニュアル「COBOL2002 ユーザーズガイド」を参照してください。

(2) SEND 文

プリンタへの帳票印刷をします。

例

```
PROCEDURE DIVISION.
     :
MOVE 'MAP3016B' TO 帳票マップ名.… 1.マップ名格納エリアに物理マップ名を格納
     :
SEND PRT          …… 2.通信記述名を指定（プリンタ用）
   FROM MAP3010       …… 3.出力用論理マップ名
   WITH EMI        …… 4.必ず指定する
```

(3) DISABLE 文

AP で帳票印刷をする場合，プログラム終了までプリンタを解放しません。このため，印刷していないときには，DISABLE 文を使用してプリンタを解放し，ほかのプログラムで印刷できるようにします。

例

```
SEND PRT
     :
DISABLE PRT … 1.通信記述名を指定
```

11.2.2　CALL 文による印刷

ここでは，COBOL の CALL 文での，オープン要求，出力要求，およびクローズ要求について説明します。

CALL 文による印刷は，Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）で動作する AP では利用できません。

(1) CALL 命令

CALL 命令の形式を次に示します。

```
CALL 'jsvwadrv' USING XMAP-COM XMAP-REQ データ名３　データ名４
```

- XMAP-COM

  共通インタフェース領域を指定します。この領域は，帳票を出力したい端末単位に作成し，オープンしたときの領域をクローズ要求まで引き継いで使用します。また，次に示す各要求時にマッピング，および出力に必要な情報をセットします。

  - オープン要求時：仮想端末名，通信種別を指定します。
  - 出力要求時：物理マップ名を指定します。

  共通インタフェース領域の形式の詳細を次の表に示します。

表 11-8　共通インタフェース領域の形式（XMAP-COM，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
アイキャッチャ	4	X(4)	'＊XP△'とします（XMAP-COM-ID）
リターンコード	2(4)	9(4)COMP-5	2 進形式で格納します（XMAP-COM-RTN）
リターンコード詳細	2(6)	9(4)COMP-5	2 進形式で格納します（XMAP-COM-RSN）
未使用	3(8)	X(3)	$(00)_{16}$ とします（XMAP-COM-RSV1）
id 区分	1(11)	X	必ず'I'とします（XMAP-COM-ITYPE）
仮想端末名	8(12)	X(8)	仮想端末定義ファイルで指定した名称を左詰めで指定し，残りは空白とします （XMAP-COM-TNAME）※1
未使用	4(20)	X(4)	$(00)_{16}$ とします（XMAP-COM-RSV2）
通信種別	4(24)	X(4)	プリンタ要求時は'OWS△'とします （XMAP-COM-MSG）
未使用	44(28)	X(44)	$(00)_{16}$ とします（XMAP-COM-RSV3）
マップ名	8(72)	X(8)	物理マップ名を左詰めでデバイス ID※2 付きで指定し，残りは空白とします （XMAP-COM-MAPNAME）
未使用	8(80)	X(8)	$(00)_{16}$ とします（XMAP-COM-RSV4）
未使用	4(88)	S9(8)COMP-5	0 とします（XMAP-COM-INLNG）
未使用	68(92)	X(68)	$(00)_{16}$ とします（XMAP-COM-RSV5）

注※1

仮想端末名を省略できるのは，画面定義時だけです。

注※2

帳票のデバイス ID については，「11.2.1(1)　通信記述項」を参照してください。

- XMAP-REQ

要求インタフェース領域を指定します。要求インタフェース領域の形式を次の表に示します。この領域は，必ず指定します。この領域に指定した内容によって，要求種別が決まります。

表 11-9　要求インタフェース領域の形式（XMAP-REQ，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
要求種別	4	X(4)	'OPEN'：オープン要求 'CLOS'：クローズ要求 'SEND'：出力要求 （XMAP-REQ-TYPE）
未使用	1(4)	X	'△'とします（XMAP-REQ-OPT1）
SEND オプション	1(5)	X	'1'：出力要求 '△'：上記以外（XMAP-REQ-OPT2）
未使用	2(6)	X(2)	'△△'とします（XMAP-REQ-RSV）

- データ名 3

  次に示す各要求に対応する情報を指定します。

  - 出力要求時：出力論理マップを指定します。
  - クローズ要求時：0 を指定します。
  - オープン要求時：オープンインタフェース領域を指定します。オープンインタフェース領域の形式を次の表に示します。この領域は，必ず指定します。

表 11-10　オープンインタフェース領域の形式（XMAP-OPN，COBOL）

データ項目名	長さ（位置）	データ形式	指定内容（データ名）
データ有無コード使用選択	1	X	'1'：データ有無コードを指定します その他：データ有無コードを指定しません（標準値「1F」を仮定） （XMAP-OPN-DCODE-SET）
データ有無コード	1(1)	X	データ有無コードを指定する場合，2 桁の 16 進数（00～FF）で指定します （XMAP-OPN-DCODE）
未使用	2(6)	X(2)	'△△'とします（XMAP-OPN-RSV）

なお，データ有無コード使用選択項目でデータ有無コードを無効にする指定をした場合は，標準値「1F」が仮定されます。

- データ名 4

  次に示す各要求に対応する情報を指定します。

  - 出力要求時：0 を指定します。
  - クローズ要求時：0 を指定します。
  - オープン要求時：0 を指定します。

(2)　インタフェース領域の取り込み方法

XMAP3 が提供しているインタフェース領域（Jsvwatbl.cbl）を AP に取り込むには，COPY 文を使用します。COPY 文は，WORKING-STORAGE SECTION，または LINKAGE SECTION に指定します。また，コンパイルする前に Jsvwatbl.cbl のフォルダ名を CBLLIB に設定しておきます。インタフェース領域を次の図に示します。

図 11-2　インタフェース領域

```
* COMMON INTERFACE AREA
 01 XMAP-COM.
     03 XMAP-COM-ID        PIC X(4).
     03 XMAP-COM-RTN       PIC 9(4) COMP-5.
     03 XMAP-COM-RSN       PIC 9(4) COMP-5.
     03 XMAP-COM-RSV1      PIC X(3).
     03 XMAP-COM-ITYPE     PIC X.
     03 XMAP-COM-TNAME     PIC X(8).
     03 XMAP-COM-RSV2      PIC X(4).
     03 XMAP-COM-MSG       PIC X(4).
     03 XMAP-COM-RSV3      PIC X(44).
     03 XMAP-COM-MAPNAME   PIC X(8).
     03 XMAP-COM-RSV4      PIC X(8).
     03 XMAP-COM-INLNG     PIC S9(8) COMP-5.
     03 XMAP-COM-RSV5      PIC X(68).
*
* REQUEST INTERFACE AREA
 01 XMAP-REQ
     03 XMAP-REQ-TYPE      PIC X(4).
```

```
     03 XMAP-REQ-OPT1      PIC X.
     03 XMAP-REQ-OPT2      PIC X.
     03 XMAP-REQ-RSV       PIC X(2).
*
* OPEN INTERFACE AREA
 01 XMAP-OPN.
     03 XMAP-OPN-DCODE-SET PIC X.
     03 XMAP-OPN-DCODE     PIC X.
     03 XMAP-OPN-RSV       PIC X(2).
```

(3) オープン要求

CALL 命令を使用する場合，帳票を印字したいプリンタに付けた仮想端末名単位にオープン要求をします。複数の仮想端末（プリンタ）をオープン要求する場合，インタフェース領域は，各端末に用意します。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

例

```
     PROCEDURE DIVISION.
        :
     INITIALIZE    XMAP-COM
        REPLACING    NUMERIC   DATA  BY  ZERO    ┐
           ALPHANUMERIC   DATA   BY   LOW-VALUE. ┘ …… 1.共通エリアをクリア

     MOVE ALL SPACE  TO XMAP-REQ.            …… 2.要求エリアをクリア
     *
     MOVE 'OPEN' TO XMAP-REQ-TYPE.           …… 3.オープン要求をセット
     MOVE '*XP△' TO XMAP-COM-ID.             …… 4.IDをセット
     MOVE 'I'    TO XMAP-COM-ITYPE.          …… 5.ID区分をセット
     MOVE 'OWS△' TO XMAP-COM-MSG.            …… 6.プリンタの通信種別をセット
     MOVE 'PRT001△△' TO XMAP-COM-TNAME.      …… 7.仮想端末名を代入
     MOVE '1'    TO XMAP-OPN-DCODE-SET.      …… 8.データ有無コードをセット
     MOVE X'1F'  TO XMAP-OPN-DCODE.
     MOVE SPACE  TO XMAP-OPN-RSV.
     CALL 'jsvwadrv' USING XMAP-COM          …… 9.共通インタフェース領域
                           XMAP-REQ          …… 10.要求インタフェース領域
                           XMAP-OPN          …… 11.オープンインタフェース領域
                           BY VALUE 0.
```

(4) 出力要求

オープン要求した仮想端末へ帳票印刷をします。オープン要求で使用した共通インタフェース領域を引き継いで使用します。

例

```
     MOVE 'SEND' TO XMAP-REQ-TYPE.     …… 1.出力要求をセット
     MOVE '△'    TO XMAP-REQ-OPT1.     …… 2.必ず空白をセット
     MOVE '1'    TO XMAP-REQ-OPT2.     …… 3.出力要求の場合' 1 'をセット
     MOVE 'MAP3016B' TO XMAP-COM-MAPNAME. … 4.物理マップ名をセット
     MOVE データ TO MAP3010.     …… 5.帳票出力したいデータを論理マップへセット
     CALL 'jsvwadrv' USING XMAP-COM    …… 6.共通インタフェース領域
                           XMAP-REQ    …… 7.要求インタフェース領域
                           MAP3010     …… 8.出力論理マップ
                           BY VALUE 0.
```

(5) クローズ要求

仮想端末をクローズします。オープン要求,および出力要求で使用したインタフェース領域を引き継いで使用します。

例

```
     MOVE 'CLOS' TO XMAP-REQ-TYPE.       …… 1.クローズ要求をセット
     MOVE '△'    TO XMAP-REQ-OPT1.       …… 2.必ず空白をセット
     MOVE '△'    TO XMAP-REQ-OPT2.       …… 3.必ず空白をセット
     CALL 'jsvwadrv' USING XMAP-COM      …… 4.共通インタフェース領域
                           XMAP-REQ      …… 5.要求インタフェース領域
                           BY VALUE 0 0.
```

11.3　書式の印刷命令（COBOL）

ここでは，COBOL を用いて書式オーバレイで印刷する方法について説明します。

書式オーバレイ印刷は，Web 実行環境，および Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）では利用できません。

書式用 AP で使用する COBOL 用の API の一覧を次に示します。

表 11-11　書式用 AP で使用する COBOL 用の API 一覧

項番	COBOL の命令文	機能概要	備考
1	OPEN 文	オープン要求	なし
2	WRITE 文	印刷要求	改行を指定して，印刷要求を実行する
3	CLOSE 文	クローズ要求	なし

注

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

11.3.1　行データの帳票印刷（COBOL）

AP から書式印刷をする場合，印刷用ファイルへ行データを出力するために PROCEDURE DIVISION で指定する文を次に示します。また，AP で行データをコーディングするときに，印刷時のずれを防ぐために設定する行データの属性と指定値について説明します。

(1)　書式印刷するときの PROCEDURE DIVISION の指定

表 11-12　PROCEDURE DIVISION で指定する文

命令文	内容
オープン (OPEN 文)	プリンタ出力用ファイルの処理を開始する準備をします OPEN 文のモードに出力専用（OUTPUT）を指定して実行します
出力 (WRITE 文)	レコードをプリンタ出力用ファイルに書き出します 明示的または間接的※に ADVANCING 指定をします
クローズ (CLOSE 文)	プリンタ出力用ファイルの処理を終了します

注※

同じファイルに対するほかの WRITE 文に ADVANCING 指定されていることを意味します。

例

```
PROCEDURE DIVISION.
OPEN OUTPUT プリンタ.                                    …… 1.
  :
WRITE 行データ [FROM データ名]
 {AFTER|BEFORE}ADVANCING{n LINE(S)|呼び名|PAGE}.        …… 2.
  :
CLOSE プリンタ.                                          …… 3.
```

例の説明

1. **プリンタのオープン**

   FILE-CONTROL で指定した出力先のプリンタをオープンします。

2. **プリンタへのレコードの書き出し**

   データ名で指定した領域の内容が行データに指定した領域に格納されたあと，行データを出力先のプリンタに書き出します。WRITE 文で書き出されたレコードの内容は，使用できなくなります。

   AFTER：ADVANCING 以降の記述内容を実行したあとで，WRITE 文が実行されます。

   BEFORE：WRITE 文を実行したあとで，ADVANCING 以降の記述内容が実行されます。

   ADVANCING：行送り（改行）または改ページを指定します。

   n LINE（S）：n に指定したページ分だけ改行します。n が 1 の場合は「1　LINE」になり，n が 2 以上の場合は「n　LINES」になります。

   呼び名：CSP（改行しない），または C01（改ページする）を指定します。

   PAGE：改ページします。

3. **プリンタのクローズ**

   オープンした出力先のプリンタをクローズします。

(2)　行データの属性の設定

書式印刷時にずれを生じさせないように，書式設計前に AP で行データの属性を決定しておきます。AP で設定できる行データの属性と指定値を次の表に示します。

表 11-13　行データの属性と指定値

属性	指定値	内容
文字サイズ※	5／7／9／12	文字サイズを 5，7，9，または 12 ポイントにして文字を配置します
文字の間隔※	0	文字の間隔を空けないで文字を配置します
	1～7	文字の間隔を 1～7 ポイントにして文字を配置します
書体※	0	文字の書体を元に戻します
	1	文字を明朝体にして配置します
	2	文字をゴシック体にして配置します
	9	文字を OCR 体にして配置します
拡大	指定あり	横方向に 2 倍に拡大した文字を配置します
	指定なし	横方向の拡大を解除した文字を配置します

注※

属性の指定がない場合で，上位の項目に属性の指定があるときは，その属性の指定値に戻して文字を配置します。上位の項目に属性の指定がないときは，ドローで定義した属性で文字を配置します。上位の項目とは，レベル番号が上位にあるデータ記述項を示します。

AP で属性を設定する場合，レコード名または一意名に CHARACTER　TYPE 句を指定します。CHARACTER　TYPE 句の形式を次に示します。

形式

```
CHARACTER TYPE IS POINT-l                    …… 1.
                 INTERVAL-m                  …… 2.
```

```
FORMAT-n                                ····· 3.
WIDE                                    ····· 4.
```

形式の説明

1. POINT-l

   文字サイズを指定します。l に指定できる文字サイズは，5／7／9／12 です。

2. INTERVAL-m

   文字の間隔（字間値）を指定します。m に指定できる文字の間隔は，0～7 です。

3. FORMAT-n

   書体を指定します。n に指定できる書体は，0／1／2／9 です。

4. WIDE

   文字を横方向に 2 倍に拡大します。

CHARACTER　TYPE 句については，マニュアル「COBOL2002 言語 拡張仕様編」を参照してください。

WRITE 文の形式によって，レコード名の CHARACTER　TYPE 句を有効にするか，一意名の CHARACTER　TYPE 句を有効にするかが異なります。

(a) 「WRITE　レコード名」の場合

「WRITE　レコード名」の場合は，レコード名に CHARACTER　TYPE 句があるときだけ，行データの印刷制御をします。WRITE 文と，CHARACTER　TYPE 句との関係を次の図に示します。

図 11-3　WRITE 文と，CHARACTER　TYPE 句との関係（「WRITE　レコード名」の場合）

```
FD A-FILE.
01 A-REC.                                          ····· 1.
 02 A-REC-1 PIC N(10) CHARACTER TYPE IS POINT-7 WIDE.
 02 A-REC-2 PIC N(10) CHARACTER TYPE IS POINT-9 FORMAT-1.
  :
PROCEDURE DIVISION.
  :
 WRITE A-REC AFTER ADVANCING 1 LINE.               ····· 2.
  :
```

1. レコード名に CHARACTER TYPE 句を宣言

   レコード名に CHARACTER TYPE 句を宣言して，行データの印刷制御を行います。

2. プリンタへのレコード書き出し

   1 行改行したあとに，行データをプリンタに出力します。行データの印刷制御は，レコード名で宣言した CHARACTER TYPE 句での設定を使用します。

(b) 「WRITE　レコード名　FROM　一意名」の場合

「WRITE　レコード名　FROM　一意名」の場合は，一意名に CHARACTER　TYPE 句があるときだけ，行データの印刷制御をします。このとき，レコード名に CHARACTER　TYPE 句があっても無視されます。WRITE 文と，CHARACTER　TYPE 句との関係を次の図に示します。

図 11-4　WRITE 文と，CHARACTER　TYPE 句との関係（「WRITE　レコード名　FROM　一意名」の場合）

```
FD A-FILE.
01 A-REC PIC N(80).                                ····· 1.
  :
WORKING-STORAGE SECTION.
01 DATA1.                                          ····· 2.
 02 A-REC-1 PIC N(10) CHARACTER TYPE IS POINT-7 WIDE.
 02 A-REC-2 PIC N(10) CHARACTER TYPE IS POINT-9 FORMAT-1.
```

```
  :
PROCEDURE DIVISION.
  :
 WRITE A-REC FROM DATA1 AFTER ADVANCING 1 LINE.     …… 3.
  :
```

1. レコード名の宣言

   レコード名を宣言します。

2. 一意名に CHARACTER TYPE 句を宣言

   一意名に CHARACTER TYPE 句を宣言して，行データの印刷制御を行います。

3. プリンタへのレコードの書き出し

   1 行改行したあとに，行データをプリンタに出力します。また，行データの印刷制御は，一意名で宣言した CHARACTER TYPE 句での設定を使用します。

(3)　推奨コーディング

行データを出力する場合，行データおよびページデータの上限値によってデータを出力できない場合があります。必要に応じて，推奨コーディングを参考にしてください。

(a)　ページ単位で，文字サイズ，間隔，書体，拡張などがすべて同じ場合

ドローの書式作成時に，書式属性ダイアログ，および行データ属性ダイアログで文字サイズなどを設定します。AP で CHARACTER TYPE 句を指定する必要はありません。

(b)　行単位で，文字サイズ，間隔，書体，拡張などがすべて同じ場合

行データ設定用のテーブルと送信用のテーブルを用意します。WRITE 文の直前で，データを送信用のテーブルにコピーしてから，送信用のテーブルを使用して，WRITE 文を実行します。

行データ設定用のテーブルは，項目ごとに定義します。送信用のテーブルは，行データ設定用のテーブルの長さを 1 項目として定義します。CHARACTER TYPE 句は，送信用のテーブルに指定します。

図 11-5　推奨コーディング

（例）文字サイズが，7ポイントと9ポイントの行を出力する場合

```
  :
DATA DIVISION.
FILE SECTION.
FD プリンタ IS GLOBAL
   RECORDING MODE IS F
   LABEL RECORD IS OMITTED
   DATA RECORD IS 行データ.

01  行データ PIC X(12).                        … 1.
  :
WORKING-STORAGE SECTION.                  ┐
01  設定用テーブル.                        │
  02  DATA01 PIC X(2).                    ├ … 2.
  02  DATA02 PIC X(8).                    │
  02  DATA03 PIC X(2).                    ┘

01  ７ポテーブル CHARACTER TYPE POINT-7.  ┐ … 3.
  02  FILLER PIC X(12).                   ┘

01  ９ポテーブル CHARACTER TYPE POINT-9.  ┐ … 4.
  02  FILLER PIC X(12).                   ┘


  :
PROCEDURE DIVISION.
  :
MOVE  '01'        TO  DATA01.
MOVE  '12345678'  TO  DATA02.
```

```
MOVE  '99'        TO  DATA03.
  :
MOVE  設定用テーブル  TO  ７ポテーブル.      … 5.
WRITE 行データ FROM７ポテーブル
              BEFORE ADVANCING 1 LINE.     … 6.

MOVE  設定用テーブル  TO  ９ポテーブル.      … 7.
WRITE 行データ FROM ９ポテーブル
              BEFORE ADVANCING 1 LINE.     … 8.
  :
```

1.CHARACTER TYPE 句は指定しない

2.行データ設定用のテーブルを項目ごとに定義する

3.設定用テーブルの全体長を１項目として文字サイズ７ポイントを定義する

4.設定用テーブルの全体長を１項目として文字サイズ９ポイントを定義する

5.７ポイント用テーブルに内容をコピーする

6.印字後，改行する

7.９ポイント用テーブルに内容をコピーする

8.印字後，改行する

(c)　ページ単位または行単位で，文字サイズ，間隔，書体，拡張などがそろっていない場合

1 行中の行データを，行データの上限値を超えないようにデータを分けます。改行は，1 行中の最終データを出力するときにします。1 行中の最終データを出力するまでは，改行しないようにしてください。行データの上限値は，概算式を参考にします。概算式については，「付録 B　書式オーバレイの 1 ページデータ量の制限」を参照してください。

文字サイズなど，同じ項目が連続する場合には，行データ設定用のテーブルと送信用のテーブルを用意します。WRITE 文の直前で，送信用のテーブルにデータをコピーしてから，送信用のテーブルを使用して WRITE 文を実行します。送信用のテーブルでは，文字サイズなど，同じ項目が連続する領域を一つの項目として定義してください。CHARACTER TYPE 句は，送信用のテーブルに指定します。

図 11-6　推奨コーディング

（例）1行データを2回に分け，同じCHARACTER TYPE句を指定する項目が連続する場合

```
  :
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
CSP IS 改行抑止.                          … 1.
  :
DATA DIVISION.
FILE SECTION.
FD プリンタ IS GLOBAL
   RECORDING MODE IS F
   LABEL RECORD IS OMITTED
   DATA RECORD IS 行データ.
01  行データ PIC X(150).                 … 2.
  :
WORKING-STORAGE SECTION.
01  設定用テーブル.
  02  設定１.
    03  DATA11 PIC X(2).                ┐
    03  DATA12 PIC X(8).                ├… 3.
    03  DATA13 PIC X(4).                ┘
      :
  02  設定２.
    03  DATA21 PIC X(4).                ┐
    03  DATA22 PIC X(4).                ├… 4.
    03  DATA23 PIC X(2).                ┘
      :
```

```
01  送信用テーブル１．                                          ┐
  02  送信１．                                                  ├… 5.
    03  FILLER PIC X(100)  CHARACTER TYPE FORMAT-1.             ┘

01  送信用テーブル２．                                          ┐
  02  前データ．                                                │
    03  FILLER PIC X(100)  VALUE ALL SPACE.                     ├… 6.
  02  送信２．                                                  │
    03  FILLER PIC X(20)   CHARACTER TYPE FORMAT-2.             │
    03  FILLER PIC X(30)   CHARACTER TYPE FORMAT-1.             ┘
   :
PROCEDURE DIVISION.
   :
MOVE  '01'        TO  DATA11.
MOVE  '12345678'  TO  DATA12.
MOVE  '9999'      TO  DATA13.
   :
MOVE  設定１  TO  送信１．                        … 7.
WRITE 行データ FROM 送信用テーブル１
              BEFORE ADVANCING 改行抑止.     … 8.
MOVE  設定２  TO  送信２．                        … 9.
WRITE 行データ FROM 送信用テーブル２
              BEFORE ADVANCING 1 LINE.       … 10.
   :
```

1. CSP(改行しない指定)を登録する
2. CHARACTER TYPE 句は指定しない
3. 行データの上限値を超えないように定義する(ここでは 100 バイトとする)
4. 行データの上限値を超えないように定義する(ここでは 50 バイトとする)
5. 設定 1 の送信用のテーブルを定義する
   CHARACTER TYPE 句が同じ項目を一つの項目にまとめる
6. 設定 2 の送信用のテーブルを定義する
   1 回目に送信したデータサイズと同じ数のスペースデータを設定する
   CHARACTER TYPE 句が同じ項目を一つの項目にまとめる
7. 送信用のテーブルに内容をコピーする
8. 印字後，改行しない
9. 送信用のテーブルに内容をコピーする
10. 印字後，改行する

12 C 言語の入出力命令

この章では，C 言語を用いて，画面の入出力および帳票と書式を印刷する方法について説明します。

12.1　画面の入出力命令（C 言語）

ここでは C 言語を用いて画面を入出力する方法について説明します。

画面用 AP で使用する C 言語用の関数を次に示します。

表 12–1　画面用 AP の関数の一覧

項番	関数	機能概要	備考
1	jsvwadrv	オープン，クローズ，出力，入力，およびオプション設定要求	要求は引数で指定する

12.1.1　画面表示命令（C 言語）

ここでは，C 言語で開発するときに使用する関数の形式と，オープン要求，クローズ要求，出力要求，入力要求，およびオプション設定要求について説明します。

(1)　jsvwadrv 関数

jsvwadrv 関数の形式を次に示します。

```
long APIENTRY jsvwadrv（XMAP_COMアドレス，XMAP_REQアドレス，
                        パラメタ3，パラメタ4）
```

1. XMAP_COM アドレス

共通インタフェース領域のアドレスを指定します。この領域は，画面を出力したい端末単位に作成し，オープンしたときの領域をクローズ要求まで引き継いで使用します。また，次に示す各要求時にマッピング，および入出力に必要な情報を代入します。

- オープン時　：　仮想端末名，通信種別を指定する。
- 出力要求時　：　物理マップ名を指定する。
- 入力要求時　：　入力論理マップ長を指定する

共通インタフェース領域の形式の詳細を次の表に示します。

表 12–2　共通インタフェース領域の形式（XMAP_COM，C 言語）

項目	長さ（位置）	データ形式	格納内容	データ名
アイキャッチャ	4	char[4]	'＊XP△'とする	xmap_com_id
リターンコード※	2(4)	unsigned short	2 進形式で XMAP3 が格納	xmap_com_rtn
画面入出力時のリターンコード※	2(6)	unsigned short	2 進形式で XMAP3 が格納	xmap_com_rsn
未使用	3(8)	char[3]	$(00)_{16}$ とする	xmap_com_rsv1
id 区分	1(11)	char	必ず'I'とする	xmap_com_itype
仮想端末名	8(12)	char[8]	出力先の仮想端末名を指定する（標準は'DSP001△△'を指定する）	xmap_com_tname

項目	長さ (位置)	データ形式	格納内容	データ名
未使用	4(20)	char[4]	$(00)_{16}$ とする	xmap_com_rsv2
通信種別	4(24)	char[4]	'BWS△'とする	xmap_com_msg
未使用	44(28)	char[44]	$(00)_{16}$ とする	xmap_com_rsv3
マップ名	8(72)	char[8]	物理マップ名を左詰めでデバイス ID 付きで指定する 残りは空白※	xmap_com _mapname
未使用	8(80)	char[8]	$(00)_{16}$ とする	xmap_com_rsv4
入力論理 マップ長	4(88)	long	入力論理マップ長を指定する	xmap_com_inlng
未使用	68(92)	char[68]	$(00)_{16}$ とする	xmap_com_rsv5

注※

論理マップ名，リターンコード，および画面のデバイス ID については，「11.1.1(1)　通信記述項」を参照してください。

2. XMAP_REQ アドレス

要求インタフェース領域のアドレスを指定します。形式の詳細を次の表に示します。この領域は，必ず指定してください。この領域で指定した内容によって要求種別が決まります。

表 12-3　要求インタフェース領域の形式（XMAP_REQ，C 言語）

項目	長さ (位置)	データ形式	指定内容	データ名
要求種別	4	char[4]	'OPEN'：オープン要求 'CLOS'：クローズ要求 'RECV'：入力要求 'SEND'：出力要求 'MDO△'：オプション設定要求	xmap_req_type
RECEIVE オプション	1(4)	char	'F'：入力要求時 '△'：上記以外	xmap_req_opt1
SEND オプション	1(5)	char	'1'：出力要求時 '△'：上記以外	xmap_req_opt2
未使用	2(6)	char[2]	'△△'とする	xmap_req_rsv

3. パラメタ 3

- 出力要求時：出力論理マップのアドレスを指定する。
- 入力要求時：入力論理マップのアドレスを指定する。
- クローズ要求時：0L を指定する。
- オプション設定要求時：1L を指定する。
- オープン要求時：オープンインタフェース領域のアドレスを必ず指定する。

オープンインタフェース領域の形式の詳細を次の表に示します。

表 12-4　オープンインタフェース領域の形式（XMAP_OPN，C 言語）

項目	長さ (位置)	データ形式	指定内容	データ名
データ有無コード使用選択	1	char	'1'：データ有無コードを指定する その他：データ有無コードを指定しない（標準値「1F」を仮定）	xmap_opn_dcode_set
データ有無コード	1(1)	unsigned char	データ有無コードを指定する場合，2 桁の 16 進数（00～FF）で指定する	xmap_opn_dcode
未使用	2(2)	char[2]	'△△'とする	xmap_opn_rsv

なお，データ有無コード使用選択項目でデータ有無コードを無効にする指定をした場合は，標準値「1F」が仮定されます。

4. パラメタ 4

オプション設定要求時は，マッピングインタフェース領域のアドレスを指定します。形式の詳細を次の表に示します。なお，マッピングオプションを指定しない場合には，0 を指定します。

表 12-5　マッピングインタフェース領域の形式（XMAP_MDO，C 言語）

項目	長さ (位置)	データ形式	指定内容	データ名
マッピングオプション大分類	4	unsigned long	XMAP_MDO_SFLD の内容をセットする	xmap_mdo_opt1
マッピングオプション小分類	4(4)	unsigned long	XMAP_MDO_MAPFLD※ XMAP_MDO_PHFLD※ XMAP_MDO_LOGFLD※	xmap_mdo_opt2

注※

指定内容とマッピングオプションは次のように対応しています。

XMAP_MDO_MAPFLD：マージ

XMAP_MDO_PHFLD：物理マップだけ

XMAP_MDO_LOGFLD：論理マップだけ

なお，マッピングオプションの指定値については「3.2　マッピングオプション」を参照してください。

(a)　リターン情報

共通インタフェース領域の xmap_com_rtn にリターン値が設定されます。リターン値を次に示します。

0：正常終了

4，8：異常終了（詳細については，共通インタフェース領域の xmap_com_rsn に設定される）

12：パラメタ不正

(b)　インタフェーステーブルの取り込み方法

XMAP3 が提供しているインタフェーステーブル（jsvwatbl.h）を AP に取り込む場合，#include 指示語を使用します。

```
#include ”jsvwatbl.h”　…　インタフェーステーブルの取り込み
```

ただし，jsvwatbl.h 中のインタフェーステーブルは，構造体の形式などを定義しており，#include 指示語によって AP に実領域は確保されません。このため，AP 中でインタフェース領域を用意する必要があります。例を次に示します。

例

```
#include ”jsvwatbl.h”

XMAP_COM com;     /* 共通インタフェース領域       */
XMAP_REQ req;     /* 要求インタフェース領域       */
XMAP_OPN opn;     /* オープンインタフェース領域   */
XMAP_MDO mdo;     /* マッピングインタフェース領域 */
```

インタフェーステーブルを次の図に示します。

図 12-1　インタフェーステーブル

```
/***     COMMON INTERFACE AREA     ********************************/
typedef struct {
  char          xmap_com_id[4];
  unsigned short    xmap_com_rtn;
  unsigned short    xmap_com_rsn;
  char          xmap_com_rsv1[3];
  char          xmap_com_itype;
  char          xmap_com_tname[8];
  char          xmap_com_rsv2[4];
  char          xmap_com_msg[4];
  char          xmap_com_rsv3[44];
  char          xmap_com_mapname[8];
  char          xmap_com_rsv4[8];
  long          xmap_com_inlng;
  char          xmap_com_rsv5[68];
}     XMAP_COM;

/***     REQUEST INTERFACE AREA     *****************************/
typedef struct {
  char          xmap_req_type[4];
  char          xmap_req_opt1;
  char          xmap_req_opt2;
  char          xmap_req_rsv[2];
}     XMAP_REQ;

/***     OPEN INTERFACE AREA     *********************************/
typedef struct {
  char          xmap_opn_dcode_set;
  unsigned char     xmap_opn_dcode;
  char          xmap_opn_rsv[2];
}     XMAP_OPN;


/***     MAPPING OPTION INTERFACE AREA     **********************/
typedef struct {
  unsigned long     xmap_mdo_opt1;
  unsigned long     xmap_mdo_opt2;
}     XMAP_MDO;

/***     MAPPING OPTION VALUE     *********************************/
#define         XMAP_MDO_SFLD              3
#define         XMAP_MDO_MAPFLD           13
#define         XMAP_MDO_PHFLD            14
#define         XMAP_MDO_LOGFLD           15
```

(2)　オープン要求

jsvwadrv 関数を使用する場合，画面を表示したい端末に付けた仮想端末名単位にオープン要求をします。複数の仮想端末をオープン要求する場合，インタフェース領域は仮想端末ごとに用意する必要があります。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

例

```
memset(&req,' △',sizeof(XMAP_REQ));                ··· 1.要求エリアのクリア
memcpy(req.xmap_req_type,"OPEN",sizeof(req.xmap_req_type));
                                                    ··· 2.オープン要求の代入
memset(&com,0,sizeof(XMAP_COM));                   ··· 3.共通エリアのクリア
memcpy(com.xmap_com_id,"*XP△",sizeof(com.xmap_com_id));
                                                    ··· 4.IDの代入
com.xmap_com_itype='I';                             ··· 5.ID区分の代入
memcpy(com.xmap_com_tname,"DSP001△△",sizeof(com.xmap_com_tname));
                                                    ··· 6.仮想端末名の代入
memcpy(com.xmap_com_msg,"BWS△",sizeof(com.xmap_com_msg));
                                                  ··· 7.ディスプレイの通信種別の代入

opn.xmap_opn_dcode_set='1';
opn.xmap_opn_dcode=0x1f;                          ··· 8.データ有無コードの代入
memset(opn.xmap_opn_rsv,' △',sizeof(opn.xmap_opn_rsv));
                                                  ··· 9.予備エリアの空白クリア
jsvwadrv(&com,&req,&opn,0L);
          ↑    ↑    ↑ オープンインタフェース領域アドレス
          ↑    └──── 要求インタフェース領域アドレス
          └─────────── 共通インタフェース領域アドレス
```

(3)　クローズ要求

仮想端末をクローズします。オープン要求，出力要求，および入力要求で使用したインタフェースを引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"CLOS",sizeof(req.xmap_req_type));
                                                    ··· 1.クローズ要求の代入
req.xmap_req_opt1=' △';                           ··· 2.必ず空白を代入する
req.xmap_req_opt2=' △';                           ··· 3.必ず空白を代入する

jsvwadrv(&com,&req,0L,0L);
          ↑    ↑ 要求インタフェース領域アドレス
          ↑    └──── 共通インタフェース領域アドレス
```

(4)　出力要求

オープン要求した仮想端末へ画面を表示します。オープン要求で使用した共通インタフェース領域を引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"SEND",sizeof(req.xmap_req_type));…1.出力要求の代入
req.xmap_req_opt1=' △';                           …2.必ず空白を代入する
req.xmap_req_opt2='1';                            …3.出力要求時1を代入する
memcpy(com.xmap_com_mapname,"MAP003NC",sizeof(com.xmap_com_mapname));
                                                  …4.物理マップ名を代入する
jsvwadrv(&com,&req,&MAP0030,0L);
          ↑    ↑    ↑ 出力論理マップのアドレス
          ↑    └──── 要求インタフェース領域アドレス
          └─────────── 共通インタフェース領域アドレス
```

注

jsvwadrv 関数を呼ぶ前に，表示したいデータを出力論理マップにセットしてください。

(5)　入力要求

入力要求によって,画面から入力した情報を AP へ入力します。画面表示要求で使用したインタフェース領域を引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"RECV",sizeof(req.xmap_req_type));…1.入力要求を代入
req.xmap_req_opt1='F';                …2.入力要求時必ず"F"を代入
req.xmap_req_opt2='△';                …3.必ず空白を代入
com.xmap_com_inlng=sizeof(MAP003I);   …4.入力論理マップ長の代入
jsvwadrv(&com,&req,&MAP003I,0L);
                                  入力論理マップのアドレス
                                  要求インタフェース領域アドレス
                                  共通インタフェース領域アドレス
```

(6)　オプション設定要求

仮想端末へ画面を表示する場合,マッピングオプションを指定できます。オープン要求で使用した共通インタフェース領域を引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"MDO△",sizeof(req.xmap_req_type));
                                          …1.オプション設定要求の代入
req.xmap_req_opt1='△';                    …2.必ず空白を代入
req.xmap_req_opt2='△';                    …3.必ず空白を代入
mdo.xmap_mdo_opt1=XMAP_MDO_SFLD;              …4.マッピングオプション大分類
                                                を代入
mdo.xmap_mdo_opt2=XMAP_MDO_LOGFLD;            …5.マッピングオプション小分類
                                                に論理データだけを代入
jsvwadrv(&com,&req,1L,&mdo);
                                  マッピングインタフェース領域アドレス
                                  必ず” 1 ” を指定する
                                  要求インタフェース領域アドレス
                                  共通インタフェース領域アドレス
```

12.2　帳票の印刷命令（C 言語）

ここでは C 言語を用いて帳票の印刷をする方法について説明します。

帳票用 AP で使用する C 言語用の関数を次に示します。

表 12-6　帳票用 AP の関数の一覧

項番	関数	機能概要	備考
1	jsvwadrv	オープン，クローズ，出力要求	要求は引数で指定する

12.2.1　帳票印刷命令（C 言語）

ここでは，C 言語で開発するときに使用する関数の形式と，オープン要求，クローズ要求，および出力要求について説明します。

(1)　jsvwadrv 関数

jsvwadrv 関数の形式を次に示します。

```
long APIENTRY jsvwadrv（XMAP_COMアドレス，XMAP_REQアドレス，パラメタ３，パラメタ４）
```

- XMAP_COM アドレス

  共通インタフェース領域のアドレスを指定します。この領域は，帳票を出力したい端末単位に作成し，オープンしたときの領域をクローズ要求まで引き継いで使用します。また，次に示す各要求時にマッピング，および出力に必要な情報をセットします。

  - オープン要求時：仮想端末名，通信種別を指定します。
  - 出力要求時：物理マップ名を指定します。

  共通インタフェース領域の形式の詳細を次の表に示します。

表 12-7　共通インタフェース領域の形式（XMAP_COM，C 言語）

データ項目名	長さ (位置)	データ形式	指定内容（データ名）
アイキャッチャ	4	char[4]	'＊XP△'とします（xmap_com_id）
リターンコード	2(4)	unsigned short	2 進形式で格納します（xmap_com_rtn）※1
リターンコード詳細	2(6)	unsigned short	2 進形式で格納します（xmap_com_rsn）※1
未使用	3(8)	char[3]	$(00)_{16}$ とします（xmap_com_rsv1）
id 区分	1(11)	char	必ず'I'とします（xmap_com_itype）
仮想端末名	8(12)	char[8]	仮想端末定義ファイルで指定した仮想端末名を左詰めで指定し，残りは空白とします（xmap_com_tname）
未使用	4(20)	char[4]	$(00)_{16}$ とします（xmap_com_rsv2）
通信種別	4(24)	char[4]	プリンタ要求時は'OWS△'とします（xmap_com_msg）

データ項目名	長さ (位置)	データ形式	指定内容（データ名）
未使用	44(28)	char[44]	$(00)_{16}$ とします（xmap_com_rsv3）
マップ名	8(72)	char[8]	物理マップ名を左詰めでデバイス ID※2 付きで指定し，残りは空白とします （xmap_com_mapname）
未使用	8(80)	char[8]	$(00)_{16}$ とします（xmap_com_rsv4）
未使用	4(88)	long	$(00)_{16}$ とします（xmap_com_inlng）
未使用	68(92)	char[68]	$(00)_{16}$ とします（xmap_com_rsv5）

注※1

jsvwadrv 関数のリターン値が 12 の場合，リターンコードおよびリターンコード詳細には何も設定できません。

注※2

帳票のデバイス ID については，「11.2.1(1)　通信記述項」を参照してください。

- XMAP_REQ アドレス

  要求インタフェース領域のアドレスを指定します。形式を次の表に示します。この領域は，必ず指定してください。この領域で指定した内容によって要求種別が決まります。

表 12-8　要求インタフェース領域の形式（XMAP_REQ，C 言語）

データ項目名	長さ (位置)	データ形式	指定内容（データ名）
要求種別	4	char[4]	'OPEN'：オープン要求 'CLOS'：クローズ要求 'SEND'：出力要求 （xmap_req_type）
未使用	1(4)	char	'△'とします（xmap_req_opt1）
SEND オプション	1(5)	char	'1'：出力要求時 '△'：上記以外 （xmap_req_opt2）
未使用	2(6)	char[2]	'△△'とします（xmap_req_rsv）

- パラメタ 3
  - 出力要求時：出力論理マップのアドレスを指定します。
  - オープン要求時：オープンインタフェース領域のアドレスを指定します。
  - クローズ要求時：0 を指定します。

  オープンインタフェース領域の形式を次の表に示します。

表 12-9　オープンインタフェース領域の形式（XMAP_OPN，C 言語）

データ項目名	長さ (位置)	データ形式	指定内容（データ名）
データ有無コード使用選択	1	char	'1'：データ有無コードを指定します

データ項目名	長さ (位置)	データ形式	指定内容（データ名）
			その他：データ有無コードを指定しません（標準値「1F」を仮定） （xmap_opn_dcode_set）
データ有無コード	1(1)	unsigned char	データ有無コードを指定する場合，2 桁の 16 進数（00～FF）で指定します （xmap_opn_dcode）
未使用	2(2)	char[2]	'△△'とします（xmap_opn_rsv）

なお，データ有無コード使用選択項目でデータ有無コードを無効にする指定をした場合は，標準値「1F」が仮定されます。

- パラメタ 4

  各要求時ともに 0 を指定します。

(a)　リターン情報

共通インタフェース領域の xmap_com_rtn にリターン値が設定されます。リターン値を次に示します。

- 0：正常終了
- 4，8：異常終了（詳細については，共通インタフェース領域の xmap_com_rsn に設定される）
- 12：パラメタ不正

(b)　インタフェーステーブルの取り込み方法

XMAP3 で用意しているインタフェーステーブル（jsvwatbl.h）を AP に取り込む場合，#include 指示語を使用します。

```
#include ”jsvwatbl.h” … インタフェーステーブルの取り込み
```

ただし，jsvwatbl.h 中のインタフェーステーブルは，構造体の形式などを定義しており，#include 指示語によって AP に実領域は取られません。このため，AP 中でインタフェース領域を取る必要があります。インタフェースを取る例を次に示します。

例

```
#include ”jsvwatbl.h”

XMAP_COM com;   /* 共通インタフェース領域   */
XMAP_REQ req;   /* 要求インタフェース領域   */
XMAP_OPN opn;   /* オープンインタフェース領域 */
```

インタフェーステーブルを次の図に示します。

図 12-2　インタフェーステーブル

```
/***     COMMON INTERFACE AREA     ********************************/
typedef struct {
 char         xmap_com_id[4];
 unsigned short    xmap_com_rtn;
 unsigned short    xmap_com_rsn;
 char         xmap_com_rsv1[3];
 char         xmap_com_itype;
 char         xmap_com_tname[8];
 char         xmap_com_rsv2[4];
 char         xmap_com_msg[4];
 char         xmap_com_rsv3[44];
 char         xmap_com_mapname[8];
```

```
 char         xmap_com_rsv4[8];
 long         xmap_com_inlng;
 char         xmap_com_rsv5[68];
}     XMAP_COM;

/***     REQUEST INTERFACE AREA     ****************************/
typedef struct {
 char         xmap_req_type[4];
 char         xmap_req_opt1;
 char         xmap_req_opt2;
 char         xmap_req_rsv[2];
}     XMAP_REQ;

/***     OPEN INTERFACE AREA     ********************************/
typedef struct {
 char         xmap_opn_dcode_set;
 unsigned char     xmap_opn_dcode;
 char         xmap_opn_rsv[2];
}     XMAP_OPN;
```

(2)　オープン要求

jsvwadrv 関数を使用する場合，帳票を出力したい端末に付けた仮想端末名単位にオープン要求をします。複数の仮想端末をオープン要求する場合，インタフェース領域は各仮想端末に用意する必要があります。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

例

```
memset(&req,' ',sizeof(XMAP_REQ));                              … 1.要求領域のクリア
memcpy(req.xmap_req_type,"OPEN",sizeof(req.xmap_req_type));      … 2.オープン要求の代入
memset(&com,0,sizeof(XMAP_COM));                                 … 3.共通領域のクリア
memcpy(com.xmap_com_id,"*XP ",sizeof(com.xmap_com_id));          … 4.ＩＤの代入
com.xmap_com_itype='I';                                          … 5.ＩＤ区分の代入
memcpy(com.xmap_com_tname,"PRT001    ",sizeof(com.xmap_com_tname));
                                                                 … 6.仮想端末名の代入
memcpy(com.xmap_com_msg,"OWS ",sizeof(com.xmap_com_msg));        … 7.プリンタの通信種別の代入
opn.xmap_opn_dcode_set='1';
opn.xmap_opn_dcode=0x1f;                                         … 8.データ有無コードの代入
memset(opn.xmap_opn_rsv,' ',sizeof(opn.xmap_opn_rsv));           … 9.予備領域の空白クリア
jsvwadrv(&com,&req,&opn,0L);
          ↑   ↑   ↑
          |   |   └── オープンインタフェース領域アドレス
          |   └────── 要求インタフェース領域アドレス
          └────────── 共通インタフェース領域アドレス
```

(3)　クローズ要求

仮想端末をクローズします。オープン要求，および出力要求で使用したインタフェースを引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"CLOS",sizeof(req.xmap_req_type));  … 1.クローズ要求のセット
req.xmap_req_opt1=' ';                                       … 2.必ず空白をセットする
req.xmap_req_opt2=' ';                                       … 3.必ず空白をセットする
jsvwadrv(&com,&req,0L,0L);
          ↑   ↑
          |   └── 要求インタフェース領域アドレス
          └────── 共通インタフェース領域アドレス
```

(4)　出力要求

オープン要求した仮想端末へ帳票を出力します。オープン要求で使用した共通インタフェース領域を引き継いで使用します。

例

```
memcpy(req.xmap_req_type,"SEND",sizeof(req.xmap_req_type)); … 1.出力要求の代入
req.xmap_req_opt1=' ';                                      … 2.必ず空白を代入する
req.xmap_req_opt2='1';                                      … 3.出力要求時に１を代入する
memcpy(com.xmap_com_mapname,"MAP0036B",sizeof(com.xmap_com_mapname));
                                                            … 4.物理マップ名を代入する
jsvwadrv(&com,&req,&MAP0030,0L);
                                                  出力論理マップのアドレス
                                                  要求インタフェース領域アドレス
                                                  共通インタフェース領域アドレス
```

注　jsvwadrv関数を呼ぶ前に，印刷したいデータを出力論理マップに代入してください。

12.3　書式の印刷命令（C 言語）

ここでは，C 言語を用いて書式オーバレイで印刷する方法について説明します。

書式用の AP で使用する C 言語用の関数一覧を次に示します。

表 12-10　書式用 AP の C 言語用の関数一覧

項番	関数名	機能概要	備考
1	jstqlopn	行データの編集処理の開始	オープン要求関数
2	jstqlpag	ページ制御情報の作成，ページバッファへの格納	なし
3	jstqlctp	行データの制御情報の設定	
4	jstqldat	行データの行バッファへの格納	
5	jstqllpt	行バッファのページバッファへの格納	
6	jstqlcls	行データの編集処理の終了	クローズ要求関数

12.3.1　行データの帳票印刷（C 言語）

(1)　jstqlopn 関数

環境変数を取り込んで，行データの編集を開始します。なお，環境変数については，マニュアル「XMAP3 実行ガイド」を参照してください。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqlopn(com)
JSTQL_COM       *com;                    …… 1.
```

(a)　形式の説明

1. JSTQL_COM アドレス

AP で確保したユーザインタフェース領域の先頭アドレスを指定します。ユーザインタフェース領域の形式を次の表に示します。

表 12-11　ユーザインタフェース領域の形式

属性および名称	位置（バイト）	長さ（バイト）	内容	備考
unsigned char lcomid	0	4	テーブル ID：'XMP3'	jstqlopn 関数を呼び出す前に設定します
unsigned char *lcomsvnm	4	4	印刷サービス名のアドレス	
unsigned char lcomdevnm	8	6	予備	jstqlopn 関数を呼び出す前に必ず，NULL（0x00）でクリアします※1
unsigned char lcomrsv1	14	2	予備	
unsigned char lcompgcnm	16	8	予備	

属性および名称	位置（バイト）	長さ（バイト）	内容	備考
unsigned char lcomrsv2	24	4	予備	
unsigned char *lcompgedr	28	4	予備	
long　lcomrc	32	4	リターンコード	関数呼び出し後に参照します※3
long　lcomerr	36	4	エラー詳細コード※2	
unsigned char lcomsync	40	1	予備	jstqlopn 関数を呼び出す前に必ず NULL（0x00）でクリアします※1
unsigned char lcomrsv3	41	1	予備	
unsigned char lcomrsv4	42	1	予備	
unsigned char lcomrsv5	43	1	予備	
unsigned char lcomrsv6	44	12	予備	
unsigned char *lcomxppdr	56	4	予備	
unsigned char lcomperr	60	4	ページ制御エラー情報	ページ制御のエラー発生時に参照します※3
unsigned char *lcomxperr	64	4	予備	jstqlopn 関数を呼び出す前に必ず NULL（0x00）でクリアします※1
unsigned char lcomwrk	68	188	XMAP3 の作業領域	jstqlopn 関数を呼び出す前に必ず NULL（0x00）でクリアします※1

注※1

NULL（0x00）でクリアしたあとは，使用しないでください。

注※2

エラー詳細コードについては，「付録 F　リターンコードと詳細コード」を参照してください。

注※3

各関数を呼び出す前に，必ず NULL（0x00）でクリアしてください。

ユーザインタフェース領域には，次に示す項目に情報を設定し，そのほかの項目は NULL（0x00）でクリアします。

- lcomid（テーブル ID）

  テーブル ID として文字列'XMP3'を設定します。

- lcomsvnm（印刷サービス名アドレス）

  印刷サービス名は，14 バイト以内の文字列で，NULL を終端とします。印刷サービス名は，スタンドアロン環境では，必ず，表示・印刷セットアップの［プリンタ］タブの印刷モードに，ページプリンタ用として定義したサービス名を指定します。C/S システム環境の場合は，C/S セットアップの印刷サービス名と同じ名称を指定します。

  印刷サービス名アドレスの指定が NULL ポインタのとき，文字列が NULL またはスペースだけのときは，環境変数「XMAP3_PSNAME」で指定した名称が仮定されます。環境変数の指定がない場合は，印刷サービス名として「#PRT」が仮定されます。

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了
- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(2)　jstqlpag 関数

ページ制御情報を作成して，ページバッファに格納します。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqlpag(com,pgcnm,rsv,ovlname)
JSTQL_COM          *com;                    …… 1.
unsigned char      *pgcnm;                  …… 2.
unsigned char      *rsv;                    …… 3.
unsigned char      *ovlname;                …… 4.
```

(a)　形式の説明

1. JSTQL_COM アドレス
   jstqlopn 関数で指定したユーザインタフェース領域の先頭アドレスを指定します。
2. 予備
   必ず，NULL を指定します。
3. 予備
   必ず，NULL を指定します。
4. 書式名格納領域のアドレス
   書式名（マップ名に ID を付けた名称（ID はマップ名が 6 文字のときは 6G，7 文字のときは F））が格納されている領域のアドレスを必ず指定します。書式名は，8 バイト以内の文字列で，NULL を終端とします。書式名を 9 バイト以上で指定した場合は，先頭から 8 バイト分を書式名と見なします。
   書式名の指定が NULL ポインタのとき，文字列が NULL または空白だけのときは，環境変数「XMAP3_FMP」で指定した名称が仮定されます。

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了
- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(c)　注意事項

jstqldat 関数での行データを jstqllpt 関数で出力していない場合，この関数によって出力されます。また，この関数を発行する jstqlcls 関数を発行するまでは，前に発行した関数の情報（書式マップ名など）が引き継がれます。

(3)　jstqlctp 関数

書式印刷時にずれを生じさせないように，行データの制御情報を作成して，行バッファに格納します。ここで指定した制御情報は，jstqllpt 関数の発行，または改ページまで有効です。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqlctp(com,argpt,cnt)
JSTQL_COM        *com;                    …… 1.
JSTQL_CTL        *argpt;                  …… 2.
long              cnt;                    …… 3.
```

(a)　形式の説明

1. JSTQL_COM アドレス

   jstqlopn 関数で指定したユーザインタフェース領域の先頭アドレスを指定します。

2. JSTQL_CTL アドレス

   行データの制御情報を設定する制御情報テーブルの先頭アドレスを指定します。制御情報テーブルの形式と，制御情報テーブルに指定できる定数と値を次の図に示します。

図 12-3　行データの制御情報テーブル（JSTQL_CTL）の形式

```
typedef struct{
     long ctl_type;
     long ctl_val;
}JSTQL_CTL;
#define JSTQ_TYPE_PNT  1
#define JSTQ_TYPE_INT  2
#define JSTQ_TYPE_FMT  3
#define JSTQ_TYPE_WID  4
```

表 12-12　行データの制御情報テーブル（JSTQL_CTL）に指定できる定数と値

制御情報の種類	定数（ctl_type）	指定値（ctl_val）	意味
文字サイズ	JSTQ_TYPE_PNT	50	文字サイズに 5 ポイントを指定します
		70	文字サイズに 7 ポイントを指定します
		90	文字サイズに 9 ポイントを指定します
		120	文字サイズに 12 ポイントを指定します
文字の間隔	JSTQ_TYPE_INT	0	文字の間隔を指定しません
		1～7	文字の間隔に 1～7 ポイントを指定します
書体	JSTQ_TYPE_FMT	0	書体を元に（標準値に）戻します
		1	書体に明朝体を指定します
		2	書体にゴシック体を指定します
		9	書体に OCR 体を指定します※
拡大（平体）	JSTQ_TYPE_WID	0	拡大を指定しません
		1	拡大を指定します

注※

OCR 体は，文字サイズに 9 ポイントを指定したときだけ有効です。

3. 項目数

制御情報テーブル中に設定した制御情報の項目数（1 以上の値）を指定します。

例

```
JSTQL_CTL arg[4];
long     cnt;

cnt = 0;
argpt[cnt].ctl_type = JSTQ_TYPE_PNT;        ……文字サイズの設定
argpt[cnt].ctl_val = 90;                    ……９ポイントを指定
cnt++;
argpt[cnt].ctl_type = JSTQ_TYPE_FMT;        ……書体の設定
argpt[cnt].ctl_val = 1;                     ……明朝体を指定
cnt++;
jstqlctp(com,argpt,cnt);                    ……文字サイズと書体の制御情報編集
```

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了
- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(c)　注意事項

行の先頭での制御情報の初期値は，文字サイズ，文字の間隔，および書体はドローで定義した値，平体は指定なしです。ただし，環境定数（XMAP3_FORMAT）で書体を指定している場合，環境変数で指定した書体が有効となります。ここで指定した制御情報は，jstqllpt 関数の発行，または改ページまで有効です。制御情報の指定がない場合には，この関数を発行しないでください。

(4)　jstqldat 関数

文字データを編集して，行バッファに格納します。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqldat(com,datp,datl)
JSTQL_COM          *com;                    …… 1.
unsigned char      *datp;                   …… 2.
long                datl;                   …… 3.
```

(a)　形式の説明

1. JSTQL_COM アドレス

   jstqlopn 関数で指定したユーザインタフェース領域の先頭アドレスを指定します。

2. 文字データのアドレス

   出力する文字データ（行データ）のアドレスを指定します。

3. 文字データの長さ

   出力する文字データ（行データ）の長さを，バイト数で指定します。

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了

- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(c)　注意事項

この関数では，印刷する文字データ（行データ）だけを指定し，行送りなどは jstqllpt 関数で指定します。

(5)　jstqllpt 関数

行バッファに格納されている行データに行送り情報を付けて，1 行分のデータをページバッファに出力します。ただし，改ページを指定したときや，改ページが発生したときは，ページバッファに格納されている 1 ページ分のデータをプリンタに出力します。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqllpt(com,argpt,tim)
JSTQL_COM        *com;                    …… 1.
JSTQL_CTL        *argpt;                  …… 2.
long              tim;                    …… 3.
```

(a)　形式の説明

1. JSTQL_COM アドレス

   jstqlopn 関数で指定したユーザインタフェース領域の先頭アドレスを指定します。

2. JSTQL_CTL アドレス

   行送り情報を設定する制御情報テーブルの先頭アドレスを指定します。制御情報テーブルの形式と，制御情報テーブルに指定できる定数と値を次に示します。

   図 12-4　行送りの制御情報テーブル（JSTQL_CTL）の形式

```
typedef struct{
     long ctl_type;
     long ctl_val;
}JSTQL_CTL;
#define JSTQ_TYPE_NLN  2
#define JSTQ_TYPE_CNL  3
```

   表 12-13　行送りの制御情報テーブル（JSTQL_CTL）に指定できる定数と値

制御情報の種類	定数（ctl_type）	指定値（ctl_val）	意味
改行	JSTQ_TYPE_NLN	1～99	1～99 行分改行します
チャネルスキップ	JSTQ_TYPE_CNL	0	行送りをしません
		1	紙送りチャネル番号 C01（改ページ）を指定します

3. 行出力と行送りのタイミング

   行データを行送り後に出力するか，または行送り前に出力するかを指定します。

   行送り後に行データを出力する場合は「0」を，行送り前に出力する場合は「1」を指定します。0 または 1 以外の値を指定した場合は「0」を仮定します。

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了

- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(c)　注意事項

この関数を発行しない場合でも，ページバッファに格納された行データの数が 1 ページに印刷できる行数を超えるとき，またはページバッファに格納できるデータ長の制限を超えるときは，自動的に改ページされます。

(6)　jstqlcls 関数

行データの編集を終了します。

形式

```
#include ”jstqlcom.h”
long APIENTRY jstqlcls(com)
JSTQL_COM       *com;                    …… 1.
```

(a)　形式の説明

1. JSTQL_COM アドレス
   jstqlopn 関数で指定したユーザインタフェース領域の先頭アドレスを指定します。

(b)　リターン情報

関数のリターン値を次に示します。

- 0：正常終了
- 8：異常終了（ユーザインタフェース領域内のエラー詳細コードを参照してください）
- 12：パラメタ不正（ユーザインタフェース領域のアドレスまたは ID が不正です）

(c)　注意事項

この関数を発行したとき，バッファ中に残っている行データは，すべて出力されます。なお，行バッファの文字データは，ページバッファに格納され，ページバッファから出力されます。

13 汎用関数の入出力命令

この章では，汎用関数の入出力命令について説明します。

13.1　画面の入出力命令（汎用関数）

ここでは，汎用関数を用いて画面を入出力する方法について説明します。

画面用 AP で使用する汎用関数を次に示します。

表 13-1　画面用 AP の汎用関数の一覧

項番	関数	機能概要	備考
1	XmapDrvCreateOpen	メモリ確保，仮想端末オープン（ディスプレイ）	OPEN 要求関数
2	XmapDrvClose	メモリ解放，仮想端末クローズ	CLOSE 要求関数
3	XmapDrvSend	画面の出力	なし
4	XmapDrvReceive	画面の入力	
5	XmapDrvTransceive	画面の入出力	
6	XmapDrvGetError	エラーコードの詳細を取得	
7	XmapDrvCreate	メモリ確保	
8	XmapDrvOpen	仮想端末オープン	OPEN 要求関数
9	XmapDrvSetMapOption	マッピングオプションを設定	出力制御パラメタ関数
10	XmapDrvSetDataCode	データ有無コードを設定	オープン制御パラメタ関数

13.1.1　画面表示命令（汎用関数）

関数の形式，処理内容，およびリターンコードについて説明します。

(1)　XmapDrvCreateOpen 関数

インタフェース用メモリを確保したり，仮想端末をオープンしたりします。

形式

```
long APIENTRY XmapDrvCreateOpen(tname)
   unsigned char        *tname;             /* 仮想端末名 */
```

仮想端末名（tname）：仮想端末定義ファイルで指定した名称を指定します。

処理内容

XMAP_COM，XMAP_REQ，XMAP_OPN および XMAP_MDO テーブルを確保し，仮想端末をオープンする。

リターンコード

0：異常終了

1 以上：正常終了（仮想端末識別子：確保した領域の先頭アドレス）

注意事項

仮想端末名は 8 バイト以内で指定し，最後に NULL で終わる文字列を指定してください。

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

データ有無コードには標準値「1F」が仮定されます。

(2)　XmapDrvClose 関数

仮想端末をクローズします。

形式

```
long APIENTRY XmapDrvClose(termid)
  long                  termid;           /* 仮想端末識別子 */
```

仮想端末識別子（termid）：オープン要求関数の戻り値を指定します。

処理内容

1. 仮想端末をクローズする。
2. XMAP_COM，XMAP_REQ，XMAP_OPN，XMAP_MDO テーブル領域を解放する。

リターンコード

0：正常終了

−1：異常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

(3)　XmapDrvSend 関数

画面を表示します。

形式

```
long APIENTRY XmapDrvSend(termid, pmpname, olog);
  long                  termid;           /* 仮想端末識別子   */
  unsigned char        *pmpname;          /* 物理マップ名     */
  unsigned char        *olog;             /* 出力論理マップ   */
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定する。
2. 物理マップ名（pmpname）：出力する物理マップ名を指定する。
3. 出力論理マップ（olog）：出力論理セグメントを指定する。

処理内容

画面を表示します。

リターンコード

0：正常終了

−1：異常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

エラーが発生した場合，自動的にクローズ処理が実行されます。

(4)　XmapDrvReceive 関数

画面に入力されたデータを取得します。

形式

```
long APIENTRY XmapDrvReceive(termid, ilog);
  long                  termid;           /* 仮想端末識別子 */
  unsigned char        *ilog;             /* 入力論理マップ */
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定する。
2. 入力論理マップ（ilog）：入力論理マップを指定する。

3. 入力論理マップの全体長を入力論理マップの先頭（2 バイト）に指定する。

処理内容

画面に入力されたデータを取得します。

リターンコード

0：正常終了

−1：異常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

エラーが発生した場合，自動的にクローズ処理が実行されます。

入力論理マップ長として実際より短い値を指定した場合，エラーを返します。

(5)　XmapDrvTransceive 関数

画面を表示したり，画面に入力されたデータを取得したりします。

形式

```
long APIENTRY XmapDrvTransceive(termid, pmpname, olog, ilog);
  long                 termid;          /* 仮想端末識別子 */
  unsigned char        *pmpname;        /* 物理マップ名   */
  unsigned char        *olog;           /* 出力論理マップ */
  unsigned char        *ilog;           /* 入力論理マップ */
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定する。
2. 物理マップ名（pmpname）：出力する物理マップ名を指定する。
3. 出力論理マップ（olog）：出力論理マップを指定する。
4. 入力論理マップ（ilog）：入力論理マップを指定する。
5. 入力論理マップの全体長を入力論理マップの先頭（2 バイト）に指定する。

処理内容

画面を表示したり，画面に入力されたデータを取得したりします。

リターンコード

0：正常終了

−1：異常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

エラーが発生した場合，自動的にクローズ処理が実行されます。

入力論理マップ長として実際より短い値を指定した場合，エラーを返します。

(6)　XmapDrvGetError 関数

エラーの詳細を取得します。

形式

```
long APIENTRY XmapDrvGetError()
```

処理内容

直前に発行された関数のエラー詳細を取得します。

リターンコード

0 以上：正常終了（詳細エラーコード）

(7)　XmapDrvCreate 関数

メモリを確保します。

形式

```
long APIENTRY XmapDrvCreate()
```

処理内容

XMAP_COM，XMAP_REQ，XMAP_OPN，XMAP_MDO テーブル領域を確保します。

リターンコード

0：異常終了

1 以上：正常終了（仮想端末識別子：確保した領域の先頭アドレス）

(8)　XmapDrvOpen 関数

仮想端末をオープンします。

形式

```
long APIENTRY XmapDrvOpen(termid, tname)
  long                 termid;          /* 仮想端末識別子 */
  unsigned char        *tname;          /* 仮想端末名     */
```

1. 仮想端末識別子（termid）：XmapDrvCreate の戻り値を指定する。
2. 仮想端末名（tname）：仮想端末定義ファイルで指定した名称を指定する。

処理内容

仮想端末をオープンします。

リターンコード

0：正常終了

−1：異常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

仮想端末名には，最後にヌルで終わる文字列を指定してください。9 バイト以上の場合，9 バイト以降は切り捨てられます。

一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

(9)　XmapDrvSetMapOption 関数

マッピングオプションを指定する。

形式

```
long APIENTRY XmapDrvSetMapOption(termid, mode)
  long                 termid;          /* 仮想端末識別子       */
  long                 mode;            /* マッピングオプション */
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定する。
2. マッピングオプション（mode）：マッピングオプションを指定する。

   指定できるマッピングオプションは，次のとおりです。

内容	指定値
マージ	0

内容	指定値
論理マップだけ	2
物理マップだけ	3

注

mode の内容については，「3.2　マッピングオプション」を参照してください。

処理内容

マッピングオプションを設定します。

リターンコード

0：正常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

mode に不当な値を指定した場合，「マージ」を仮定します。

(10)　XmapDrvSetDataCode 関数

データ有無コードを設定する。

形式

```
long APIENTRY XmapDrvSetDataCode(termid, code)
  long                termid;           /* 仮想端末識別子     */
  unsigned char       code;             /* データ有無コード   */
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定する。
2. データ有無コード（code）：データ有無コードを指定する。

処理内容

データ有無コード（$(00)_{16}$～$(1F)_{16}$ または$(FF)_{16}$）を設定します。データ有無コードはすべての AP にかかわるコードのため，できるだけ標準値（$(1F)_{16}$）を利用することをお勧めします。

リターンコード

0：正常終了

注意事項

クローズ済みの仮想端末識別子を指定した場合，アクセス違反を起こす場合があります。

この関数を発行しない場合，データ有無コードには標準値「1F」が仮定されます。

13.2　帳票の印刷命令（汎用関数）

ここでは，汎用関数を用いて帳票を印刷する方法について説明します。

帳票用 AP で使用する汎用関数を次に示します。

表 13-2　帳票用 AP の汎用関数の一覧

項番	関数	機能概要	備考
1	XmapDrvCreateOpen	ドライバ起動，仮想端末オープン（プリンタ）	オープン要求関数
2	XmapDrvClose	仮想端末クローズ，ドライバ解放	クローズ要求関数
3	XmapDrvSend	帳票の出力	なし
4	XmapDrvGetError	エラーコードの詳細を取得	
5	XmapDrvCreate	ドライバ起動	オープン要求関数
6	XmapDrvOpen	端末オープン	オープン要求関数
7	XmapDrvSetDataCode	データ有無コードの設定	オープン制御関数

13.2.1　帳票印刷命令（汎用関数）

関数の形式とリターン情報について説明します。

(1)　XmapDrvCreateOpen 関数

ドライバを起動し，仮想端末をオープンします。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

データ有無コードには標準値「1F」が仮定されます。

形式

```
long APIENTRY XmapDrvCreateOpen(tname)
  unsigned char          *tname;                    /*仮想端末名*/
```

仮想端末名（tname）：仮想端末名を指定します。仮想端末名（8 バイト以下，9 バイト以降は切り捨て）には，最後に NULL で終わる文字列を指定してください。

リターンコード

1 以上：正常終了（仮想端末識別子）

0：異常終了

(2)　XmapDrvClose 関数

仮想端末をクローズし，ドライバを解放します。

形式

```
long APIENTRY XmapDrvClose(termid)
  long               termid;                        /*仮想端末識別子*/
```

仮想端末識別子（termid）：オープン要求関数の戻り値を指定します。

リターンコード

0：正常終了

－1：異常終了

(3)　XmapDrvSend 関数

帳票を出力します。

形式

```
long APIENTRY XmapDrvSend(termid, pmpname, olog);
  long              termid;                    /* 仮想端末識別子 */
  unsigned char     *pmpname;                  /* 物理マップ名   */
  unsigned char     *olog;                     /* 出力論理マップ */
```

1.仮想端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.物理マップ名（pmpname）：出力する物理マップ名を指定します。

3.出力論理マップ（olog）：出力論理セグメントを指定します。

リターンコード

0：正常終了

－1：異常終了

(4)　XmapDrvGetError 関数

直前に発行した関数のエラー詳細を取得します。

形式

```
long APIENTRY XmapDrvGetError( )
```

リターンコード

0 以上：正常終了（詳細エラーコード）

(5)　XmapDrvCreate 関数

ドライバを起動します。

形式

```
long APIENTRY XmapDrvCreate( )
```

リターンコード

1 以上：正常終了（仮想端末識別子）

0：異常終了

(6)　XmapDrvOpen 関数

仮想端末をオープンします。一つの AP（プロセス）内で同時にオープンできる端末数の上限は 15 個です。

形式

```
long APIENTRY XmapDrvOpen(termid, tname)
  long              termid;                    /* 仮想端末識別子 */
  unsigned char     *tname;                    /* 仮想端末名     */
```

1.仮想端末識別子（termid）：XmapDrvCreate の戻り値を指定します。

2.仮想端末名（tname）：仮想端末定義ファイルで指定した名称を指定します。

リターンコード

0：正常終了

－1：異常終了

(7)　XmapDrvSetDataCode 関数

データの有無コードを設定します。この関数を発行しない場合，データ有無コードには標準値「1F」が仮定されます。

形式

```
long APIENTRY XmapDrvSetDataCode(termid, code)
  long               termid;                    /*仮想端末識別子*/
  unsigned char      code;                      /*データ有無コード*/
```

1. 仮想端末識別子（termid）：オープン要求関数の戻り値を指定します。
2. データ有無コード（code）：データ有無コード（$(00)_{16}$～$(1F)_{16}$，または$(FF)_{16}$）を設定します。
   データ有無コードは，すべての AP にかかわるコードのため，できるだけ標準値（$(1F)_{16}$）を利用することをお勧めします。

リターンコード

0：正常終了

13.3　書式の印刷命令（汎用関数）

ここでは，汎用関数を用いて書式オーバレイで印刷する方法について説明します。

書式用 AP で使用する汎用関数を次に示します。

表 13-3　書式用 AP の汎用関数の一覧

項番	関数	機能概要	備考
1	XmapFrmCreateOpen	ドライバ起動，サービスオープン	オープン要求関数
2	XmapFrmClose	ドライバ解放，サービスクローズ	クローズ要求関数
3	XmapFrmSetPage	ページ制御情報の設定	なし
4	XmapFrmSetData	行データをバッファリング	
5	XmapFrmSetLine	バッファデータを出力	
6	XmapFrmGetError	エラーコードの詳細を取得	
7	XmapFrmSetPoint	文字サイズの設定	フィールド制御関数
8	XmapFrmSetInterval	字間値の設定	フィールド制御関数
9	XmapFrmSetFont	書体の設定	フィールド制御関数
10	XmapFrmSetWidth	平体の設定	フィールド制御関数
11	XmapFrmSetNewLine	改行数の設定	行制御関数
12	XmapFrmSetChannel	チャネルスキップの設定	行制御関数

13.3.1　書式印刷命令（汎用関数）

関数の形式とリターン情報について説明します。

(1)　XmapFrmCreateOpen 関数

ドライバを起動し，サービスをオープンします。

形式

```
long APIENTRY XmapFrmCreateOpen(tname)
  unsigned char          *tname;                    /* サービス名 */
```

サービス名（tname）：印刷サービス名を指定します。

印刷サービス名は，14 バイト以内の文字列で NULL を終端とします。スタンドアロンの環境では，表示・印刷セットアップの「プリンタ」タブの印刷モードに，ページプリンタ用として定義したサービス名を指定します。C/S システム環境の場合は，C/S セットアップの印刷サービス名と同じ名称を指定します。

また，印刷サービス名アドレスの指定が NULL ポインタ，文字列が NULL，またはスペースだけの場合，環境変数「XMAP3_PSNAME」で指定した名称が仮定されます。環境変数の指定がない場合は，印刷サービス名として「#PRT」が仮定されます。

リターンコード

1 以上：正常終了（端末識別子）

0：異常終了

(2)　XmapFrmClose 関数

サービスをクローズし，ドライバを解放します。

形式

```
long APIENTRY XmapFrmClose(termid)
  long               termid;                  /*端末識別子*/
```

端末識別子（termid）：オープン要求関数の戻り値を指定します。

リターンコード

0：正常終了

−1：異常終了

注意事項

この関数を発行したとき，バッファ内に残っている行データは，すべて出力されます。なお，行バッファの文字データは，ページバッファに格納され，ページバッファから出力されます。

(3)　XmapFrmSetPage 関数

ページ制御情報を作成して，ページバッファに格納します。

形式

```
long APIENTRY XmapFrmSetPage(termid,pagec,fmpname);
  long               termid;                  /*端末識別子*/
  unsigned char    *pagec;                    /*予備*/
  unsigned char    *fmpname;                  /*書式マップ名*/
```

1. 端末識別子（termid）：オープン要求関数の戻り値を指定します。
2. 予備（pagec）：必ず NULL を指定します。
3. 書式マップ名（fmpname）：書式マップ名を指定します。

   書式名（マップ名に ID を付けた名称（ID はマップ名が 6 文字のときは 6G，7 文字のときは F））が格納されている領域のアドレスを必ず指定します。書式名は，8 バイト以内の文字列で，NULL を終端とします。書式名を 9 バイト以上で指定した場合は，先頭から 8 バイト分を書式名と見なします。

   ただし，書式名の指定が NULL ポインタ，文字列が NULL または空白だけのとき，環境変数「XMAP3_FMP」で指定した名称が仮定されます。

リターンコード

0：正常終了

−1：異常終了

注意事項

XmapFrmSetData 関数での行データを XmapFrmSetLine 関数で帳票出力していない場合，この関数によって帳票が出力されます。また，この関数を発行するかまたは XmapFrmClose 関数を発行するまでは，前に発行した関数の情報（書式マップ名など）が引き継がれます。

(4)　XmapFrmSetData 関数

文字データを編集して行バッファに格納します。

形式

```
long APIENTRY XmapFrmSetData(termid,datp,datl);
  long               termid;                  /*端末識別子*/
```

```
unsigned char   *datp;                          /*文字データアドレス*/
long             datl;                          /*文字データ長*/
```

1. 端末識別子（termid）：オープン要求関数の戻り値を指定します。
2. 文字データのアドレス（datp）：出力する文字データ（行データ）のアドレスを指定します。
3. 文字データの長さ（datl）：出力する文字データ（行データ）の長さをバイト数で指定します。

リターンコード

0：正常終了

−1：異常終了

注意事項

この関数では，印刷する文字データ（行データ）だけを指定し，行送りなどは XmapFrmSetLine 関数で指定します。

(5)　XmapFrmSetLine 関数

行バッファに格納されている行データに行送り情報を付けて，1 行分のデータをページバッファに出力します。ただし，改ページを指定したときや，改ページが発生したときは，ページバッファに格納されている 1 ページ分のデータをプリンタに出力します。

形式

```
long APIENTRY XmapFrmSetLine(termid,timing);
  long             termid;                    /*端末識別子*/
  long             timing;                    /*行送りタイミング*/
```

1. 端末識別子（termid）：オープン要求関数の戻り値を指定します。
2. 行送りタイミング（timing）：行送りのタイミングを指定します。

   0：行送り後に，行データを出力します。

   1：行送り前に，行データを出力します。

リターンコード

0：正常終了

−1：異常終了

注意事項

この関数を発行していないとき，XmapFrmSetData 関数での行データの数が 1 ページに印刷できる行数を超えるている，またはページバッファに格納できるデータ長の制限を超えている場合は，自動的に改ページされます。

また，この関数の前に，XmapFrmSetChannel または XmapFrmSetNewLine を発行しなかった場合，チャネル「0：(行送りをしない)」を仮定します。

(6)　XmapFrmGetError 関数

直前に発行した関数のエラー詳細を取得します。

形式

```
long APIENTRY XmapFrmGetError( );
```

リターンコード

0 以上：正常終了（詳細エラーコード）

(7)　XmapFrmSetPoint 関数

文字サイズを指定します。

形式

```
long APIENTRY XmapFrmSetPoint(termid,val);
  long              termid;                    /*端末識別子*/
  long              val:                       /*文字サイズ*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.文字サイズ（val）：文字サイズを指定します。

内容	指定値
5 ポイント	50
7 ポイント	70
9 ポイント	90
12 ポイント	120

リターンコード

0：正常終了

注意事項

文字サイズだけを変更します。以降の XmapFrmSetData で有効になります。指定の有効範囲は，XmapFrmSetLine の発行または改ページされるまでです。

この関数を発行しない場合は，ドローの書式属性ダイアログで設定した文字サイズが有効となります。

(8)　XmapFrmSetInterval 関数

文字の間隔を指定します。

形式

```
long APIENTRY XmapFrmSetInterval(termid,val);
  long              termid;                      /*端末識別子*/
  long              val:                         /*文字間隔*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.文字の間隔（val）：文字の間隔を指定します。

内容	指定値
文字間隔なし	0
1l～7l	1～7

リターンコード

0：正常終了

注意事項

文字間隔だけを変更します。以降の XmapFrmSetData で有効になります。指定の有効範囲は，XmapFrmSetLine の発行または改ページされるまでです。

この関数を発行しない場合は，ドローの書式属性ダイアログで設定した文字間隔が有効となります。

(9)　XmapFrmSetFont 関数

文字の書体を指定します。

形式

```
long APIENTRY XmapFrmSetFont(termid,val);
  long              termid;                    /*端末識別子*/
  long              val:                       /*書体*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.書体（val）：書体を指定します。

内容	指定値
書体復帰	0
明朝	1
ゴシック	2
OCR※	9

注※

OCR 体は，文字サイズに 9 ポイントを指定したときだけに有効です。

リターンコード

0：正常終了

注意事項

書体だけを変更します。以降の XmapFrmSetData で有効になります。

指定の有効範囲は，XmapFrmSetLine の発行または改ページされるまでです。

(10)　XmapFrmSetWidth 関数

文字の拡大を指定します。

形式

```
long APIENTRY XmapFrmSetWidth(termid,val);
  long              termid;                    /*端末識別子*/
  long              val:                       /*横倍角有無*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.横倍角有無（val）：横倍角の有無を指定します。

内容	指定値
横倍角する	1
横倍角しない	0

リターンコード

0：正常終了

注意事項

横倍指定だけを変更します。以降の XmapFrmSetData で有効になります。

指定の有効範囲は，XmapFrmSetLine の発行または改ページされるまでです。

(11)　XmapFrmSetNewLine 関数

改行数を指定します。

形式

```
long APIENTRY XmapFrmSetNewLine(termid,val);
  long              termid;                         /*端末識別子*/
  long              val:                            /*改行数*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.改行数（val）：改行数を指定します。

内容	指定値
改行数	0～99

リターンコード

0：正常終了

注意事項

改行数だけを変更します。以降の XmapFrmSetLine で有効になります。

指定の有効範囲は，XmapFrmSetLine までです。

また，XmapFrmSetNewLine と XmapFrmSetChannel は一緒に指定できません。

(12)　XmapFrmSetChannel 関数

チャネルスキップを指定します。

形式

```
long APIENTRY XmapFrmSetChannel(termid,val);
  long              termid;                         /*端末識別子*/
  long              val:                            /*チャネル番号*/
```

1.端末識別子（termid）：オープン要求関数の戻り値を指定します。

2.チャネル番号（val）：チャネル番号を指定します。

内容	指定値
行送りしない	0
紙送りチャネル番号（改ページ）C01 を指定します	1

リターンコード

0：正常終了

注意事項

チャネルスキップだけを変更します。以降の XmapFrmSetLine で有効になります。

指定の有効範囲は，XmapFrmSetLine までです。

また，XmapFrmSetNewLine と XmapFrmSetChannel は一緒に指定できません。

14 Java のクラス

この章では，XMAP3 Cosminexus 連携を利用したシステムで動作する AP で使用する Java のクラスについて説明します。

14.1　提供するクラス

XMAP3/Web for Cosminexus では，論理マップ，動的変更テーブルなどを Java で扱えるようにするために XML 文書を生成します。このため，XMAP3/Web for Cosminexus では，各 XML 文書のオブジェクトに対応するクラスを提供しています。

XMAP3/Web for Cosminexus が提供するクラスを使用する場合は，次に示すパッケージを使用してください。

[jp.co.Hitachi.soft.XMAP3.server]

XMAP3/Web for Cosminexus が提供するクラスの一覧を次の表に示します。

表 14–1　XMAP3/Web for Cosminexus が提供するクラス一覧

クラス	クラス名	概要
PropertyValue	環境管理用クラス	環境管理ファイルのオブジェクトを構築します。
LogicalMap※	入力出力データ用クラス	入力／出力データ用 XML 文書のオブジェクトを構築します。
ConstValue※	定数用クラス	定数用 XML 文書のオブジェクトを構築します。
ModTbl	動的変更用クラス	動的変更用 XML 文書のオブジェクトを構築します。
ComTbl	通信制御用クラス	通信制御用 XML 文書のオブジェクトを構築します。
Filler	桁寄せクラス	桁寄せを指定する際に指定する定数オブジェクトです。

注※

LogicalMap クラスまたは ConstValue クラスで繰り返し項目にアクセスする場合，繰り返し項目のインデックスを指定するときは 0 から指定してください。

フレーム内およびフレーム外の繰り返しフィールドにアクセスする場合について次に説明します。

フレーム内の繰り返しフィールドにアクセスする場合

フレームを意識しないでフィールド単位にアクセスしてください。

フレーム外の繰り返しフィールドにアクセスする場合

フィールド単位にアクセスしてください。

String および byte[ ]を指定する場合には，ヌルオブジェクトを指定しないでください。ヌルオブジェクトを指定した場合，XmapRuntimeException をスローします。

各クラスで指定できるデータ（数値）の範囲を次に示します。範囲外の数値を指定した場合は，XmapException をスローします。

- short を指定する場合：0～32,767
- int を指定する場合：0～2,147,483,647

setter でデータを設定する場合，設定元のデータのデータ長が，設定先のデータ長より長いとき，設定先のデータ長だけデータを設定します。

XMAP3/Web for Cosminexus が提供する例外クラスの一覧を次の表に示します。

表 14–2　XMAP3/Web for Cosminexus が提供する例外クラス一覧

例外クラス	概要
XmapException	XMAP3 が提供するメソッドのパラメタが正しくないとき，またはデータ定義 XML の定義が正しくないときに発生する例外クラスです。この例外が発生した場合は，AP およびデータ定義 XML を見直す必要があります。
XmapEnvironmentException	ユーザの XMAP3 の実行環境が正しくない場合に発生する例外クラスです。この例外が発生した場合，Throwable クラスの getCause()メソッドで例外の詳細を調べて，例外に対処する必要があります。
XmapRuntimeException	業務を実行中に XMAP3 が提供するメソッドの延長で発生する例外です。この例外が発生した場合，Throwable クラスの getCause()メソッドで例外の詳細を調べて，例外に対処する必要があります。

XMAP3/Web for Cosminexus が提供する各クラスのメソッドでは，ログを出力します。

この章では，XMAP3/Web for Cosminexus が提供するクラスが提供するメソッドについて説明します。

14.2　PropertyValue クラス

PropertyValue クラスが提供するメソッドについて説明します。

14.2.1　createInstance

形式

```
public static PropertyValue createInstance(ServletContext context)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

PropertyValue クラスのインスタンスを生成します。

パラメタ

context：サーブレットのサーブレットコンテキスト

戻り値

PropertyValue クラスのインスタンス

例外

XmapException：次の場合に発生します。

- Context に null を指定した場合
- 環境管理ファイルに定義項目「LogFileDir」の指定がない場合

XmapEnvironmentException：環境管理ファイルがない場合

14.3　LogicalMap クラス

LogicalMap クラスが提供するフィールド，コンストラクタ，およびメソッドについて説明します。

14.3.1　MAX_LENGTH フィールド

形式

```
public static int MAX_LENGTH
```

説明

入力データの最大長を示します。

14.3.2　LogicalMap コンストラクタ

形式

```
public LogicalMap(PropertyValue propValue,
                  String mapName,
                  byte initData,
                  byte fillCharacter)
    throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

LogicalMap オブジェクトを構築し，入力／出力データ XML を解析します。

パラメタ

propValue：LogicalMap オブジェクトを管理する環境管理クラス

mapName：入力／出力データ用 XML の文書名

initData：初期化文字

fillCharacter：埋字

戻り値

なし

例外

XmapException：次の場合に発生します。

- 入力／出力データ用 XML の定義が正しくない場合
- propValue または mapName にヌルオブジェクトを指定した場合

XmapEnvironmentException：次の場合に発生します。

- 入力／出力データ用 XML の配置が正しくない場合
- 入力／出力データ用 XML の構文が正しくない場合

XmapRuntimeException：その他実行したときの例外

14.3.3　setDataShort

(1)　setDataShort（形式 1）

形式

```
public void setDataShort(String dataName,
                         short data,
                         int rpt)
                         throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目に数値データを設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する数値データ

rpt：繰り返し項目のインデックス

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合
- data に 0～32,767 以外の値を指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　setDataShort（形式 2）

形式

```
public void setDataShort(String dataName,
                         short data)
                         throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目に数値データを設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する数値データ

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合
- data に 0～32,767 以外の値を指定した場合

XmapRuntimeException：その他実行したときの例外

14.3.4　setDataByteArray

(1)　setDataByteArray（形式 1）

形式

```
public void setDataByteArray(String dataName,
                             byte[] data,
                             int rpt,
                             Filler filler,
                             byte fillCharacter)
                             throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目にバイトデータ配列を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定するバイトデータ配列

rpt：繰り返し項目のインデックス

filler：桁寄せオプション

fillCharacter：埋字

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName または filler にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　setDataByteArray（形式 2）

形式

```
public void setDataByteArray(String dataName,
                             byte[] data,
                             Filler filler,
                             byte fillCharacter)
                             throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目にバイトデータ配列を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定するバイトデータ配列

filler：桁寄せオプション

fillCharacter：埋字

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName または filler にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(3)　setDataByteArray（形式 3）

形式

```
public void setDataByteArray(String dataName,
                             byte[] data,
                             int rpt)
                             throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目にバイトデータ配列を設定します。

桁寄せは左寄せとし，埋字はコンストラクタで指定した埋字を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定するバイトデータ配列

rpt：繰り返し項目のインデックス

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(4)　setDataByteArray（形式 4）

形式

```
public void setDataByteArray(String dataName,
                             byte[] data)
                             throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目にバイトデータ配列を設定します。

桁寄せは左寄せとし，埋字はコンストラクタで指定した埋字を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定するバイトデータ配列

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.3.5　setDataString

(1)　setDataString（形式 1）

形式

```
public void setDataString(String dataName,
                          String data,
                          int rpt,
                          Filler filler,
                          byte fillCharacter)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定した繰り返し項目に文字配列を MS932 文字列として設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する文字配列

rpt：繰り返し項目のインデックス

filler：桁寄せオプション

fillCharacter：埋字

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName または filler にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

(2)　setDataString（形式 2）

形式

```
public void setDataString(String dataName,
                          String data,
                          Filler filler,
                          byte fillCharacter)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目に文字配列を MS932 文字列として設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する文字配列

filler：桁寄せオプション

fillCharacter：埋字

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定したデータ項目がない場合
- dataName または filler にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

(3)　setDataString（形式 3）

形式

```
public void setDataString(String dataName,
                          String data,
                          int rpt)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定した繰り返し項目に文字配列を設定します。

桁寄せは左寄せとし，埋字はコンストラクタで指定した埋字を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する文字配列

rpt：繰り返し項目のインデックス

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

(4)　setDataString（形式 4）

形式

```
public void setDataString(String dataName,
                          String data)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目に文字配列を設定します。

桁寄せは左寄せとし，埋字はコンストラクタで指定した埋字を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する文字配列

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.3.6　getDataShort

(1)　getDataShort（形式 1）

形式

```
public short getDataShort(String dataName,
                          int rpt)
                           throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目の数値データを取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定した繰り返し項目から取得した数値データ

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataShort（形式 2）

形式

```
public short getDataShort(String dataName)
                          throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目の数値データを取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した数値データ

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.3.7　getDataByteArray

(1)　getDataByteArray（形式 1）

形式

```
public byte[] getDataByteArray(String dataName,
                               int rpt)
                               throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目のバイナリデータ配列を取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定した繰り返し項目から取得したバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataByteArray（形式 2）

形式

```
public byte[] getDataByteArray(String dataName)
                                  throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目のバイナリデータ配列を取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得したバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.3.8　getDataString

(1)　getDataString（形式 1）

形式

```
public String getDataString(String dataName,
                            int rpt)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定した繰り返し項目の Unicode 文字配列を取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定した繰り返し項目から取得した文字配列

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataString（形式 2）

形式

```
public String getDataString(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目の Unicode 文字配列を取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した文字配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.3.9　setClientData

形式

```
public void setClientData(byte[] data,
                          int beginIndex,
                          int lastIndex)
                   throws XmapException, XmapRuntimeException
```

説明

クライアントから受信したデータを beginIndex から（lastIndex - 1）分設定します。

パラメタ

data：設定するバイトデータ配列

beginIndex：開始インデックス（この値を含む）

lastIndex：終了インデックス（この値を含まない）

戻り値

なし

例外

XmapException：次の場合に発生します。

- beginIndex が負の値である場合
- lastIndex が入出力データの長さより大きい場合
- beginIndex が lastIndex より大きい場合
- data にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.3.10　init

形式

```
public void init(byte initData)
             throws XmapRuntimeException
```

説明

入力／出力データ用クラスを初期化します。

パラメタ

initData：初期化文字

戻り値

なし

例外

XmapRuntimeException：その他実行したときの例外

14.4　ConstValue クラス

ConstValue クラスが提供するコンストラクタおよびメソッドについて説明します。

14.4.1　ConstValue コンストラクタ

形式

```
public ConstValue(PropertyValue propValue,
                  String fileName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

PropertyValue オブジェクトを ConstValue オブジェクト内に展開し，ConstValue オブジェクトを構築します。

パラメタ

propValue：ConstValue オブジェクトを管理する環境管理オブジェクト

fileName：定数用 XML ファイル名

戻り値

なし

例外

XmapException：次の場合に発生します。

- 定数用 XML の定義が正しくない場合
- propValue または fileName にヌルオブジェクトを指定した場合

XmapEnvironmentException：次の場合に発生します。

- 定数用 XML の配置が正しくない場合
- 定数用 XML の構文が正しくない場合

XmapRuntimeException：その他実行したときの例外

14.4.2　getDataShort

(1)　getDataShort（形式 1）

形式

```
public short getDataShort(String dataName,
                          int rpt)
                           throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目の定数を数値として取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定した繰り返し項目から取得した定数の数値データ

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataShort（形式 2）

形式

```
public short getDataShort(String dataName)
                                throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目の定数を数値として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数の数値データ

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.4.3　getDataByteArray

(1)　getDataByteArray（形式 1）

形式

```
public byte[] getDataByteArray(String dataName,
                                    int rpt)
                                    throws XmapException, XmapRuntimeException
```

説明

指定した繰り返し項目の定数をバイナリデータ配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定したデータ項目から取得した定数のバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataByteArray（形式 2）

形式

```
public byte[] getDataByteArray(String dataName)
                                    throws XmapException, XmapRuntimeException
```

説明

指定したデータ項目の定数をバイナリデータ配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数のバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.4.4　getDataString

(1)　getDataString（形式 1）

形式

```
public String getDataString(String dataName,
                                int rpt)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定した繰り返し項目の定数を Unicode 文字配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

rpt：繰り返し項目のインデックス

戻り値

指定したデータ項目から取得した定数の文字配列

例外

XmapException：次の場合に発生します。

- dataName と rpt で指定した繰り返し項目がない
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

(2)　getDataString（形式 2）

形式

```
public String getDataString(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目の定数を Unicode 文字配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数の文字配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.5　ModTbl クラス

ModTbl クラスが提供するコンストラクタおよびメソッドについて説明します。

14.5.1　ModTbl コンストラクタ

形式

```
public ModTbl(PropertyValue propValue,
                    String fileName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

PropertyValue オブジェクトを ModTbl オブジェクト内に展開し，ModTbl オブジェクトを構築します。

パラメタ

propValue：ModTbl オブジェクトを管理する環境管理オブジェクト

fileName：動的変更用 XML の文書名

戻り値

なし

例外

XmapException：次の場合に発生します。

- 動的変更用 XML の定義が正しくない場合
- propValue または fileName にヌルオブジェクトを指定した場合

XmapEnvironmentException：次の場合に発生します。

- 動的変更用 XML の配置が正しくない場合
- 動的変更用 XML の構文が正しくない場合

XmapRuntimeException：その他実行したときの例外

14.5.2　getDataShort

形式

```
public int getDataShort(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目の定数を数値として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数の数値データ

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.5.3　getDataByteArray

形式

```
public byte[] getDataByteArray(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目の定数をバイナリデータ配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数のバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.5.4　getDataString

形式

```
public String getDataString(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

指定したデータ項目の定数を Unicode 文字配列として取得します。

パラメタ

dataName：取得する繰り返し項目名

戻り値

指定したデータ項目から取得した定数の Unicode 文字配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.6　ComTbl クラス

ComTbl クラスが提供するフィールド，コンストラクタ，およびメソッドについて説明します。

14.6.1　LENGTH フィールド

形式

```
public static int LENGTH
```

説明

通信制御データ長を示します。

14.6.2　ComTbl コンストラクタ

形式

```
public ComTbl(PropertyValue propValue,
                    byte initData)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

PropertyValue オブジェクトを ComTbl オブジェクト内に展開し，ComTbl オブジェクトを構築します。

パラメタ

propValue：ComTbl オブジェクトを管理する環境管理オブジェクト

戻り値

なし

例外

XmapException：次の場合に発生します。

- 通信制御用 XML の定義が正しくない場合
- propValue にヌルオブジェクトを指定した場合

XmapEnvironmentException：次の場合に発生します。

- 通信制御用 XML の配置が正しくない場合
- 通信制御用 XML の構文が正しくない場合

XmapRuntimeException：その他実行したときの例外

14.6.3　setDataShort

形式

```
public void setDataShort(String dataName,
                         short data)
                         throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目に数値データを設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する数値データ

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合
- data に 0～32,767 以外の値を指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.4　setDataInteger

形式

```
public void setDataInteger(String dataName,
                           int data)
                            throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目に数値データを設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する数値データ

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合
- data に 0～2,147,483,647 以外の値を指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.5　setDataByteArray

形式

```
public void setDataByteArray(String dataName,
                             byte[] data)
                             throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目にバイトデータ配列を設定します。

パラメタ

dataName：設定するデータ項目名

data：設定するバイトデータ配列

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.6　setDataString

形式

```
public void setDataString(String dataName,
                          String data)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目に文字配列を MS932 文字列として設定します。

パラメタ

dataName：設定するデータ項目名

data：設定する文字配列

戻り値

なし

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.6.7　getDataShort

形式

```
public short getDataShort(String dataName)
                          throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目の数値データを取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した定数の数値データ

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.8　getDataInteger

形式

```
public int getDataInteger(String dataName)
                          throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目の数値データを取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した定数の数値データ

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.9　getDataByteArray

形式

```
public byte[] getDataByteArray(String dataName)
                                  throws XmapException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目のバイトデータ配列を取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した定数のバイナリデータ配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.10　getDataString

形式

```
public String getDataString(String dataName)
     throws XmapException, XmapEnvironmentException, XmapRuntimeException
```

説明

通信制御に該当するデータ項目の文字配列を Unicode 文字列として取得します。

パラメタ

dataName：取得するデータ項目名

戻り値

指定したデータ項目から取得した定数の文字配列

例外

XmapException：次の場合に発生します。

- dataName で指定したデータ項目がない場合
- dataName にヌルオブジェクトを指定した場合

XmapEnvironmentException：アプリケーション実行環境が MS932 をサポートしていない場合

XmapRuntimeException：その他実行したときの例外

14.6.11　setClientData

形式

```
public void setClientData(byte[] data,
                          int beginIndex,
                          int lastIndex)
                   throws XmapException, XmapRuntimeException
```

説明

通信制御クラスにクライアントから受信したデータを beginIndex から（lastIndex - 1）分設定します。

パラメタ

data：設定するバイトデータ配列

beginIndex：開始インデックス（この値を含む）

lastIndex：終了インデックス（この値を含まない）

戻り値

なし

例外

XmapException：次の場合に発生します。

- beginIndex が負の値である場合
- lastIndex が入出力データの長さより大きい場合
- beginIndex が lastIndex より大きい場合
- data にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.12　getClientData

形式

```
public void getClientData( )
                       throws XmapRuntimeException
```

説明

通信制御クラスからクライアントへ送信するデータを取得します。

パラメタ

なし

戻り値

なし

例外

XmapRuntimeException：その他実行したときの例外

14.6.13　jointClientData

形式

```
public byte[] jointClientData(LogicalMap lMap)
     throws XmapException, XmapRuntimeException
```

説明

通信制御のデータと論理データ

パラメタ

lMap：連結する論理データを含む論理マップクラスのインスタンス

戻り値

なし

例外

XmapException：lMap にヌルオブジェクトを指定した場合

XmapRuntimeException：その他実行したときの例外

14.6.14　init

形式

```
public void init(byte initData)
                throws XmapRuntimeException
```

説明

通信制御クラスを初期化します。

パラメタ

initData：初期化文字

戻り値

なし

例外

XmapRuntimeException：その他実行したときの例外

14.7　Filler クラス

桁寄せを指定するための Filler クラスが提供するフィールドについて説明します。

14.7.1　RIGHT フィールド

形式

```
public static Filler RIGHT
```

説明

右寄せを指定します。

14.7.2　LEFT フィールド

形式

```
public static Filler LEFT
```

説明

左寄せを指定します。

14.8　例外クラス

XMAP3/Web for Cosminexus が提供する例外クラスについて説明します。

14.8.1　XmapException

XMAP3 実行クラスライブラリの不正な利用を通知する例外クラスです。Exception クラスを継承しています。この例外の詳細情報は，Throwable クラスの getCause メソッドで取得できます。

14.8.2　XmapEnvironmentException

XMAP3 実行クラスライブラリの処理内で，実行時に発生した例外を通知する例外クラスです。Exception クラスを継承しています。この例外の詳細情報は，Throwable クラスの getCause メソッドで取得できます。

14.8.3　XmapRuntimeException

XMAP3 実行クラスライブラリの処理内で，実行時に発生した例外を通知する例外クラスです。RuntimeException クラスを継承しています。この例外の詳細情報は，Throwable クラスの getCause メソッドで取得できます。

15 コマンドリファレンス

この章では XMAP3 で提供するコマンドについて説明します。

15.1　物理マップの移行と形式チェック（UNIX）

物理マップの移行と形式チェックには，cmapcp コマンドを使用します。

15.1.1　cmapcp コマンド

cmapcp コマンドを実行する前に，環境変数 PATH に「/opt/HIXMAP/bin」を設定してください。

形式

```
cmapcp　〔オプション〕
```

機能

対象の拡張子を削除したファイル名でファイルを複写します。対象となるファイルを次に示します。

- 拡張子が「.pmap」の物理マップファイル
- 拡張子が「.pmp」の物理マップファイル

なお，形式チェックでは XMAP3 実行環境で参照するすべての情報はチェックしませんので，XMAP3 実行環境でエラーとなる場合があります。

オプション

-i dir

対象の拡張子が付けられたファイルをサーチするディレクトリ名を指定します。入力ディレクトリ下のすべてのディレクトリをサーチの対象とします。省略時はカレントディレクトリを入力ディレクトリとします。

-o dir

ファイル名から拡張子を削除したファイルを出力するディレクトリ名を指定します。省略時はカレントディレクトリにファイルを出力します。

実行結果

cmapcp コマンドを実行した結果は，次の内容で標準出力に出力されます。

なお，形式チェックの結果が正常なファイルについては，ファイル名は出力されません。

```
Physicalmap is broken.(file:/tmp/MAP001ND.pmp)　… 1.
        :
10 FILE(S) COPIED                                ……… 2.
```

1. 物理ファイルのエラーメッセージ
2. 物理ファイルを移行した結果が正常であったファイル数

エラーメッセージ

cmapcp コマンド実行時にエラーが発生した場合，次のエラーメッセージが標準出力に出力されます。

表 15-1　cmapcp コマンド実行時のエラーメッセージ

項番	出力メッセージ	エラー内容
1	Physicalmap endian error.（file：ファイル名）	物理マップのエンディアンが不正です。 (S)処理を続行します。 (P)XMAP3 実行環境に合ったエンディアンとなるように，物理マップを再作成します。
2	Physicalmap is broken.（file：ファイル名）	物理マップの形式が不正です。 ファイル転送時にテキストファイルとして転送している場合が考えられます。

項番	出力メッセージ	エラー内容
		(S)処理を続行します。 (P)ファイル転送に誤りがないか確認し，正しい物理マップを指定します。
3	Physicalmap version error.(file:ファイル名)	物理マップ作成時のバージョン（VV-RR）が合っていません。物理マップ作成時に使用した XMAP3 のバージョン（VV-RR）が XMAP3 Server Runtime よりも新しいため，使用できません。 (S)処理を続行します。 (P)物理マップ作成時に使用した XMAP3 のバージョン（VV-RR）に合う XMAP3 Server Runtime を使用し，再実行します。
4	cmapcp error.	コマンド実行できないエラーが発生しました。 (S)処理を終了します。 (P)UNIX のリソース不足やファイル参照できない環境になっていないか確認し，問題を解決して再実行します。

（凡例）

(S)：システムの対処

(P)：プログラマの対処

15.2　書式イメージ・行制御データの形式チェック(UNIX)

書式イメージファイル，行制御データファイルの形式チェックには，mapfchk コマンドを使用します。

15.2.1　mapfchk コマンド

mapfchk コマンドを実行する前に，環境変数 PATH に「/opt/HIXMAP/bin」を設定してください。

形式

```
mapfchk　〔オプション〕
```

機能

対象ファイルの形式をチェックします。対象となるファイルを次に示します。

- 拡張子が「.fmp」の書式イメージファイル
- 拡張子が「.pci」の行制御データファイル

なお，形式チェックでは XMAP3 実行環境で参照するすべての情報はチェックしませんので，XMAP3 実行環境でエラーとなる場合があります

オプション

-i dir

対象の拡張子が付けられているファイルをサーチするディレクトリ名を指定します。入力ディレクトリ下のすべてのディレクトリをサーチの対象とします。

省略時はカレントディレクトリを入力ディレクトリとします。

-h

ヘルプ情報を出力します。

実行結果

mapfchk コマンドを実行した結果は，次の内容で標準出力に出力されます。

なお，形式チェックの結果が正常なファイルについては，ファイル名は出力されません。

```
Formmap is broken.(file:/tmp/JYU1FH6G.fmp)     ……1.
        :
FORMMAP 10 FILE(S) CHECKED                     ……2.
PAGEC MODULE 10 FILE(S) CHECKED                ……3.
```

1. 書式イメージファイルのエラーメッセージ
2. 書式イメージファイルをチェックした結果が正常であったファイル数
3. 行制御イメージファイルをチェックした結果が正常であったファイル数

エラーメッセージ

mapfchk コマンド実行時にエラーが発生した場合，次のエラーメッセージが標準出力に出力されます。

表 15-2　mapfchk コマンド実行時のエラーメッセージ

項番	出力メッセージ	エラー内容
1	Formmap endian error.（file：ファイル名）	物理マップのエンディアンが不正です。 (S)処理を続行します。

項番	出力メッセージ	エラー内容
		(P)XMAP3 実行環境に合ったエンディアンとなるように，物理マップを再作成します。
2	Formmap is broken.（file：ファイル名)	書式イメージファイルの形式が不正です。 ファイル転送時にテキストファイルとして転送している場合が考えられます。 (S)処理を続行します。 (P)ファイル転送に誤りがないか確認し，正しい書式イメージファイルを指定します。
3	Formmap version error.（file：ファイル名）	書式イメージファイル作成時のバージョン（VV-RR）が合っていません。書式イメージファイル作成時に使用した XMAP3 のバージョン（VV-RR）が XMAP3 Server Runtime よりも新しいため，使用できません。 (S)処理を続行します。 (P)書式イメージファイル作成時に使用した XMAP3 のバージョン（VV-RR）に合う XMAP3 Server Runtime を使用し，再実行します。
4	Pagec module is broken.（file：ファイル名）	行制御データファイルの形式が不正です。 ファイル転送時にテキストファイルとして転送している場合が考えられます。 (S)処理を続行します。 (P)ファイル転送に誤りがないか確認し，正しい行制御データファイルを指定します。
5	Pagec module version error.（file：ファイル名）	行制御データファイル作成時のバージョン（VV-RR）が合っていません。行制御データファイル作成時に使用した XMAP3 のバージョン（VV-RR）が XMAP3 Server Runtime よりも新しいため，使用できません。 (S)処理を続行します。 (P)行制御データファイル作成時に使用した XMAP3 のバージョン（VV-RR）に合う XMAP3 Server Runtime を使用し，再実行します。
6	mapfchk error.	コマンド実行できないエラーが発生しました。 (S)処理を終了します。 (P)UNIX のリソース不足やファイル参照できない環境になっていないか確認し，問題を解決して再実行します。
7	指定オプション option error.	コマンドに指定されたオプションに誤りがあります。 (S)処理を終了します。 (P)指定したオプションを見直し，再実行します。
8	指定ディレクトリ is not exist.	コマンドに指定されたディレクトリが存在しません。 (S)処理を終了します。 (P)正しいディレクトリを指定し，再実行します。

（凡例）

(S)：システムの対処

(P)：プログラマの対処

16 論理マップ生成規則とマッピング規則

GUI 画面，CUI 画面，または帳票をレイアウトして保存すると，論理マップが生成されます。この章では，生成された論理マップとマッピング規則の関係について説明します。

16.1　固定部

ここでは,固定部の入出力に関する論理マップ生成規則とマッピング規則について説明します。固定部の論理マップ生成規則とマッピング規則は，再定義名の有無によって内容が変わります。

固定部のマップに関する定義

固定部のマップに関係する定義は，再定義名の指定によって変わってきます。

再定義名は，画面属性ダイアログ，または帳票属性ダイアログの［再定義名...］ボタンで表示される論理マップの再定義名ダイアログで指定します。

16.1.1　出力論理マップ

固定部の出力論理マップ生成規則とマッピング規則を次に示します。

(1)　出力論理マップ生成規則

(a)　再定義名を指定した場合

COBOL

論理マップ可変部の集団項目化を指定した場合

・Windows リトルエンディアン用

```
01 マップ名O REDEFINES 出力再定義名.
  02 マップ名L PIC S9(4) COMP-5.
  02 マップ名Z PIC S9(4) COMP.
  02 マップ名G.
```

・Windows ビッグエンディアン用，UNIX 用

```
01 マップ名O REDEFINES 出力再定義名.
  02 マップ名L PIC S9(4) COMP.
  02 マップ名Z PIC S9(4) COMP.
  02 マップ名G.
```

論理マップ可変部の集団項目化を指定していない場合

・Windows リトルエンディアン用

```
01 マップ名O REDEFINES 出力再定義名.
  02 マップ名L PIC S9(4) COMP-5.
  02 マップ名Z PIC S9(4) COMP.
```

・Windows ビッグエンディアン用，UNIX 用

```
01 マップ名O REDEFINES 出力再定義名.
  02 マップ名L PIC S9(4) COMP.
  02 マップ名Z PIC S9(4) COMP.
```

C 言語

C 言語の場合，集団項目化の指定は無視されます。

・Windows リトルエンディアン用，UNIX 用

```
struct  マップ名O {
      short           マップ名L;
      unsigned char   マップ名Z[2];
          :
} 出力再定義名;
```

注

short の大きさは，2 バイトとして展開されます。

・Windows ビッグエンディアン用

```
struct  マップ名0 {
        unsigned char   マップ名L[2];
        unsigned char   マップ名Z[2];
              :
} 出力再定義名 ;
```

(b)　再定義名を指定しない場合

COBOL

論理マップ可変部の集団項目化を指定した場合

・Windows リトルエンディアン用

```
01 マップ名0.
  02 マップ名L PIC S9(4) COMP-5 〔VALUE＋長さ〕.
  02 マップ名Z PIC S9(4) COMP   〔VALUE＋0〕.
  02 マップ名G.
```

・Windows ビッグエンディアン用，UNIX 用

論理マップ長が 10,000 バイト未満のとき

```
01 マップ名0.
  02 マップ名L PIC S9(4) COMP   〔VALUE＋長さ〕.
  02 マップ名Z PIC S9(4) COMP   〔VALUE＋0〕.
  02 マップ名G.
```

論理マップ長が 10,000 バイト以上のとき

```
01 マップ名0.
  02 マップ名Q PIC X(2) VALUE X'長さ'.
  02 マップ名L REDEFINES マップ名Q PIC S9(4) COMP.
  02 マップ名Z PIC S9(4) COMP VALUE +0.
  02 マップ名G.
```

論理マップ可変部の集団項目化を指定していない場合

・Windows リトルエンディアン用

```
01 マップ名0.
  02 マップ名L PIC S9(4) COMP-5 〔VALUE＋長さ〕.
  02 マップ名Z PIC S9(4) COMP   〔VALUE＋0〕.
```

・Windows ビッグエンディアン用，UNIX 用

論理マップ長が 10,000 バイト未満のとき

```
01 マップ名0.
  02 マップ名L PIC S9(4) COMP   〔VALUE＋長さ〕.
  02 マップ名Z PIC S9(4) COMP   〔VALUE＋0〕.
```

論理マップ長が 10,000 バイト以上のとき

```
01 マップ名0.
  02 マップ名Q PIC X(2) VALUE X'長さ'.
  02 マップ名L REDEFINES マップ名Q PIC S9(4) COMP.
  02 マップ名Z PIC S9(4) COMP VALUE +0.
```

C 言語

C 言語の場合，集団項目化の指定は無視されます。

・Windows リトルエンディアン用，UNIX 用

```
struct {
      short        マップ名L;
      unsigned char マップ名Z[2];
            :
}  マップ名0 〔={長さ,{0x00,0x00}}〕 ;
```

注

short の大きさは，2 バイトとして展開されます。

・Windows ビッグエンディアン用

```
struct {
        unsigned char マップ名L[2];
        unsigned char マップ名Z[2];
              :
}  マップ名0  〔={{0xnn,0xnn},{0x00,0x00}}〕 ;
```

(2)　マッピング規則

(a)　再定義名を指定した場合

論理マップ長部（マップ名 L で生成される項目）に代入した内容とその結果

論理マップ長部に代入した内容	結果
展開した論理マップより小さい値	代入された論理マップ長が示す範囲のデータを使って画面・帳票出力のデータとして使用される。範囲外の部分にはデータ有無コードが代入されているものとして扱う。
展開した論理マップ長と等しい値	論理マップ内のデータを，画面・帳票出力のデータとして使用する。
展開した論理マップより大きい値	マップ生成機能が展開した論理マップの範囲までのデータを，画面・帳票出力のデータとして使用する。それ以外の部分は無視される。

(b)　再定義名を指定しない場合

論理マップ長部（マップ名 L で生成される項目）に代入した内容とその結果は，再定義名を指定した場合と同様になります。

16.1.2　入力論理マップ

固定部の入力論理マップ生成規則とマッピング規則を次に示します。

(1)　入力論理マップ生成規則

(a)　再定義名を指定した場合

COBOL

論理マップ可変部の集団項目化を指定した場合

・Windows リトルエンディアン用

```
01 マップ名I REDEFINES入力再定義名.
  02 マップ名S PIC S9(4) COMP-5.
  02 マップ名0 PIC S9(4) COMP.
  02 マップ名K.
```

・Windows ビッグエンディアン用，UNIX 用

```
01 マップ名I REDEFINES入力再定義名.
 02 マップ名S PIC S9(4) COMP.
 02 マップ名0 PIC S9(4) COMP.
 02 マップ名K.
```

論理マップ可変部の集団項目化を指定していない場合

・Windows リトルエンディアン用

```
01 マップ名I REDEFINES入力再定義名.
  02 マップ名S PIC S9(4) COMP-5.
  02 マップ名0 PIC S9(4) COMP.
```

・Windows ビッグエンディアン用，UNIX 用

```
01 マップ名I REDEFINES入力再定義名.
  02 マップ名S PIC S9(4) COMP.
  02 マップ名0 PIC S9(4) COMP.
```

C 言語

C 言語の場合，集団項目化の指定は無視されます。

・Windows リトルエンディアン用，UNIX 用

```
struct マップ名I  {
      short         マップ名S;
      unsigned char マップ名0[2];
          :
}  入力再定義名 ;
```

注

short の大きさは，2 バイトとして展開されます。

・Windows ビッグエンディアン用

```
struct マップ名I  {
      unsigned char マップ名S[2];
      unsigned char マップ名0[2];
          :
}  入力再定義名 ;
```

(b)　再定義名を指定しない場合

COBOL

論理マップ可変部の集団項目化を指定した場合

・Windows リトルエンディアン用

```
01 マップ名I.
 02 マップ名S PIC S9(4) COMP-5 [VALUE +長さ].
 02 マップ名0 PIC S9(4) COMP [VALUE +0].
 02 マップ名K.
```

・Windows ビッグエンディアン用，UNIX 用

論理マップ長が 10,000 バイト未満のとき

```
01 マップ名I.
 02 マップ名S PIC S9(4) COMP [VALUE +長さ].
 02 マップ名0 PIC S9(4) COMP [VALUE +0].
 02 マップ名K.
```

論理マップ長が 10,000 バイト以上のとき

```
01 マップ名I.
 02 マップ名F PIC X(2) VALUE X'長さ'.
 02 マップ名S REDEFINES マップ名Q PIC S9(4) COMP.
 02 マップ名O PIC S9(4) COMP VALUE +0.
 02 マップ名K.
```

論理マップ可変部の集団項目化を指定していない場合

・Windows リトルエンディアン用

```
01 マップ名I.
  02 マップ名S PIC S9(4) COMP-5〔VALUE＋長さ〕.
  02 マップ名0 PIC S9(4) COMP  〔VALUE＋0〕.
```

・Windows ビッグエンディアン用，UNIX 用

論理マップ長が 10,000 バイト未満のとき

```
01 マップ名I.
  02 マップ名S PIC S9(4) COMP  〔VALUE＋長さ〕.
  02 マップ名0 PIC S9(4) COMP  〔VALUE＋0〕.
```

論理マップ長が 10,000 バイト以上のとき

```
01 マップ名I.
  02 マップ名F PIC X(2) VALUE X'長さ'.
  02 マップ名S REDEFINES マップ名Q PIC S9(4) COMP.
  02 マップ名O PIC S9(4) COMP VALUE +0.
```

C 言語

C 言語の場合，集団項目化の指定は無視されます。

・Windows リトルエンディアン用，UNIX 用

```
struct {
      short        マップ名S;
      unsigned char マップ名0[2];
            :
}  マップ名I  〔={長さ,{0x00,0x00}}〕 ;
```

注

short の大きさは，2 バイトとして展開されます。

・Windows ビッグエンディアン用

```
struct {
      unsigned char マップ名S[2];
      unsigned char マップ名0[2];
            :
}  マップ名I  〔={{0xnn,0xnn},{0x00,0x00}}〕 ;
```

(2)　マッピング規則

(a)　再定義名を指定した場合

論理マップ長部（マップ名 S で生成される項目）に代入した内容とその結果

論理マップ長部に代入した内容	結果
展開した論理マップより小さい値	入力マッピング時，論理マップ長部はチェックされない（マッピング制御機能が入力論理マップ長の値を代入する）ため，ほかの領域を壊すことがある。必ずマップ長分の領域を用意する必要がある。
展開した論理マップ長と等しい値	指定された論理マップの範囲に入力データが代入される。
展開した論理マップより大きい値	マップ生成機能が展開した入力論理マップの範囲に入力データを代入し，それ以外の部分は無視する（データは代入されない）。論理マップ長部には，マッピング制御機能が入力論理マップ長の値を代入する。

(b)　再定義名を指定しない場合

論理マップ長部（マップ名 S で生成される項目）に代入した内容とその結果は，再定義名を指定した場合と同様になります。

16.2　可変部（画面共通）

ここでは，可変部の入出力に関する論理マップ生成規則とマッピング規則について説明します。この節では，COBOL の数字項目の記述を次のようにしています。

PIC 9（長さ）と表記している項目

実際に生成される論理マップは，PIC 99999 のように 9 が長さ分生成されます。

論理項目に指定する文字データとして，ヌル$(00)_{16}$およびデータ有無コード$(1F)_{16}$を除く制御コード（JIS：$(00)_{16}$～$(1F)_{16}$，$(7F)_{16}$）は指定できません。

16.2.1　ウィンドウ表示

(1)　ウィンドウ表示の制御情報

制御情報の標準値

- ウィンドウ制御項目データ名：マップ名-CNTRLO

  データ名は，ドローセットアップの「ドローの設定」から表示する，表示属性の動的変更ダイアログの「データ名」タブで変更できます。
- 長さ：2

  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定...］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「ウィンドウ属性」を選択し，「修飾名長」で長さを設定します。

(2)　論理マップ生成規則

ウィンドウ表示制御情報の論理マップ生成例を次に示します。

COBOL

```
{02|03} マップ名-CNTRLO  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_CNTRLO [長さ];
```

(3)　マッピング規則

ウィンドウ表示制御情報のマッピング規則について説明します。

AP がウィンドウ表示制御情報のデータ名の領域に代入した内容とその結果

AP がウィンドウ表示制御情報のデータ名の領域に代入した内容	結果
修飾名と同じ	ウィンドウ表示制御情報の修飾名に対応するウィンドウ制御情報を使って画面が表示される。
データの先頭またはすべてがデータ有無コード，または修飾名以外（上記以外）。	標準のウィンドウ表示制御情報を使って画面が表示される。画面属性ダイアログの表示形態が「一部上書」，または「自動」の場合は，直前の表示画面と同じ画面のとき「一部上書」で，異なる画面のとき「全面書換」で表示する。

注

修飾名と修飾名に対応したウィンドウ表示制御情報は，ドローセットアップの「表示属性の動的変更」の「ウィンドウ属性」タブで変更できます。

16.2.2　出力カーソル制御

(1)　出力カーソル項目の定義

行列（2 進）カーソルの標準値

- 出力カーソル項目データ名：（行）マップ名-OUTCURSL,（列）マップ名-OUTCURSC
- 長さ：4

行列（10 進）カーソルの標準値

- 出力カーソル項目データ名：（行）マップ名-OUTCURSY,（列）マップ名-OUTCURSX
- 長さ：8

論理カーソルの標準値

- 論理カーソル項目データ名：　マップ名-OUTCURS-LOCO
- 長さ：2

データ名は，ドローセットアップの「カーソルとフォーカス」で変更できます。

(2)　論理マップ生成規則

出力カーソル制御の論理マップ生成例を次に示します。GUI 画面のフォーカス・カーソル制御については，「16.3.8　フォーカス・カーソル位置」を参照してください。

(a)　行列（2 進）カーソルを指定した場合

COBOL

・Windows リトルエンディアン用

```
{02|03} マップ名-OUTCURSL  PIC S9(4)  COMP-5.
{02|03} マップ名-OUTCURSC  PIC S9(4)  COMP-5.
```

・Windows ビッグエンディアン用，UNIX 用

```
{02|03} マップ名-OUTCURSL  PIC S9(4)  COMP.
{02|03} マップ名-OUTCURSC  PIC S9(4)  COMP.
```

C 言語

```
unsigned char マップ名_OUTCURSL[2];
unsigned char マップ名_OUTCURSC[2];
```

(b)　行列（10 進）カーソルを指定した場合

COBOL

```
{02|03} マップ名-OUTCURSY  PIC 9(4).
{02|03} マップ名-OUTCURSX  PIC 9(4).
```

C 言語では，行列（2 進）カーソルに変換されます。

(c)　論理カーソルを指定した場合

論理カーソル定数については「16.5.1(2)(b)　カーソル定数の出力の指定」を参照してください。

COBOL

```
{02|03} マップ名-OUTCURS-LOCO  PIC X(2).
```

C 言語

```
unsigned char マップ名_OUTCURS_LOCO[2];
```

(3)　マッピング規則

出力カーソル制御のマッピング規則について説明します。GUI 画面のフォーカス・カーソル制御については，「16.3.8　フォーカス・カーソル位置」を参照してください。

(a)　行列（2 進）カーソルを指定した場合

AP がカーソル項目データ名の領域に代入した内容とその結果

AP がカーソル項目データ名の領域に代入した内容	結果
不正なカーソル位置※， $(00)_{16}$ クリア，またはデータ有無コードクリア	《ドローの定義画面で初期カーソル位置の指定がある》 初期カーソルを指定したフィールドの先頭にカーソルが位置づく。 《初期カーソル位置の指定がない》 先頭の行列位置にある入力できるフィールドにカーソルが位置づく。カーソルはフィールドの先頭に位置づく。
正しいカーソル位置 （1 行／1 列～画面サイズ）	出力カーソル項目データ名の領域に代入された値（行／列位置）にカーソルが位置づく。 代入した位置にワンタッチクリアを指定したフィールドがある場合，カーソルはフィールドの先頭に位置づく。

注※

例えば，行／列が画面の大きさを超えている場合が該当します。

(b)　行列（10 進）カーソルを指定した場合

AP がカーソル項目データ名の領域に代入した内容とその結果

AP がカーソル項目データ名の領域に代入した内容	結果
不正なカーソル位置※， $(00)_{16}$ クリア，またはデータ有無コードクリア	《ドローの定義画面で初期カーソル位置の指定がある》 初期カーソルを指定したフィールドの先頭にカーソルが位置づく。 《初期カーソル位置の指定がない》 先頭の行列位置にある入力できるフィールドにカーソルが位置づく。カーソルはフィールドの先頭に位置づく。
正しいカーソル位置 （1 行／1 列～画面サイズ）	出力カーソル項目データ名の領域に代入された値（行／列位置）にカーソルが位置づく。 代入した位置にワンタッチクリアを指定したフィールドがある場合，カーソルはフィールドの先頭に位置づく。

注※

例えば，行／列が画面の大きさを超えている場合が該当します。

(c)　論理カーソルを指定した場合

AP がカーソル項目データ名の領域に代入した内容とその結果

AP がカーソル項目データ名の領域に代入した内容	結果
$(00)_{16}$ クリア，または	《ドローの定義画面で初期カーソル位置の指定がある》 初期カーソルを指定したフィールドの先頭にカーソルが位置づく。

APがカーソル項目データ名の領域に代入した内容	結果
マップ生成機能が生成するカーソル定数以外の値	《初期カーソル位置の指定がない》 先頭の行列位置にある入力できるフィールドにカーソルが位置づく。カーソルはフィールドの先頭に位置づく。
マップ生成機能が生成するカーソル定数	カーソル定数に対応したフィールドの先頭にカーソルが位置づく。

16.2.3　入力カーソル制御

(1)　入力カーソル項目の定義

行列（2進）カーソルの標準値

- 入力カーソル項目データ名：(行）マップ名-INCURSN,（列）マップ名-INCURSM
- 長さ：4

データ名は，ドローセットアップの「カーソルとフォーカス」で変更できます。

行列（10進）カーソルの標準値

- 入力カーソル項目データ名：(行）マップ名-INCURSY,（列）マップ名-INCURSX
- 長さ：8

データ名は，ドローセットアップの「カーソルとフォーカス」で変更できます。

論理カーソルの標準値

- 論理カーソル項目データ名：マップ名-INCURS-LOCI
- 長さ：2

データ名は，ドローセットアップの「カーソルとフォーカス」で変更できます。

(2)　論理マップ生成規則とマッピング規則

入力カーソル制御の論理マップ生成例とマッピング規則について説明します。GUI画面のフォーカス・カーソル制御については，「16.3.8　フォーカス・カーソル位置」を参照してください。

(a)　行列（2進）カーソルを指定した場合

COBOL

・Windowsリトルエンディアン用

```
{02|03} マップ名-INCURSN  PIC S9(4)  COMP-5.
{02|03} マップ名-INCURSM  PIC S9(4)  COMP-5.
```

・Windowsビッグエンディアン用，UNIX用

```
{02|03} マップ名-INCURSN  PIC S9(4)  COMP.
{02|03} マップ名-INCURSM  PIC S9(4)  COMP.
```

C言語

```
unsigned char マップ名_INCURSN[2];
unsigned char マップ名_INCURSM[2];
```

(b)　行列（10進）カーソルを指定した場合

COBOL

```
{02|03} マップ名-INCURSY  PIC 9(4).
{02|03} マップ名-INCURSX  PIC 9(4).
```

C 言語では，行列（2 進）カーソルに変換されます。

(c)　論理カーソルを指定した場合

論理カーソル定数については，「16.5.1(2)(b)　カーソル定数の出力の指定」を参照してください。

COBOL

```
03 マップ名-INCURS-LOCI  PIC X(2).
```

C 言語

```
unsigned char マップ名_INCURS_LOCI[2];
```

(3)　マッピング規則

入力カーソル制御のマッピング規則について説明します。

(a)　行列（2 進）カーソルを指定した場合

AP のカーソル項目データ名の領域に代入される内容

画面送信時の入力カーソル位置（行／列位置）が行，列それぞれのデータ名に代入されます。

(b)　行列（10 進）カーソルを指定した場合

AP のカーソル項目データ名の領域に代入される内容

画面送信時の入力カーソル位置（行／列位置）が行，列それぞれのデータ名に代入されます。

(c)　論理カーソルを指定した場合

AP のカーソル項目データ名の領域に代入される内容

入力操作	結果
画面送信時のカーソル位置が入力・入出力フィールド内に収まっていない。	$(00)_{16}$ を代入する。
画面送信時のカーソル位置が入力・入出力フィールド内に収まっている。	該当するフィールドに対応する論理カーソル定数を代入する。

注

入力単位が「フィールド単位」または「イベント単位」の場合，キー操作をしたオブジェクトのカーソル定数を代入します。

16.2.4　イベント通知コード

(1)　イベント通知コードの定義

イベント通知項目は，通常マップ生成を実行すると自動的に生成されます。

イベント通知項目の標準値

- イベント通知項目：マップ名-INCI
- 長さ（標準値）：4

  標準値は，ドローセットアップのイベント通知コードの「通知コードのデータ名」および「通知コードの長さ」で変更できます。

(2)　論理マップ生成規則

イベント通知項目の論理マップ生成規則について説明します。

COBOL

```
{02|03} マップ名-INCI  PIC X(4)
```

C 言語

```
unsigned char マップ名_INCI[4];
```

(3)　マッピング規則

イベント通知項目のマッピング規則について説明します。

入力操作と結果（画面単位／フィールド単位の場合）

入力操作	結果
指定されたイベント通知コードと対応するキーが入力された。	イベント通知項目の領域にイベント通知コードを代入する。余白がある場合，余白は空白（$(20)_{16}$）で埋める。
指定されたイベント通知コードと対応しないキーが入力された。または MCR 入力によって自動送信された。	初期クリア文字に従う。初期クリア文字が埋字の場合，イベント通知項目は空白（$(20)_{16}$）で埋める。

注 1

イベント通知コードは，ドローセットアップのイベント通知コードダイアログで変更できます。フィールドの全桁入力による自動送信の場合，送信キーが入力されたものとして動作します。フィールド単位，または自動送信属性が定義された項目に対して MCR 入力するとき，項目長よりも長いデータを入力した場合は，MCR 入力による自動送信ではなくフィールド離脱による自動送信の扱いとなります。

注 2

ターゲットの文字コードが EUC の場合，半角カナは 2 バイトの領域を必要とします。そのため，イベント通知コードの長さが半角カナの上位バイト（1 バイト目）までしか設定できない場合には，イベント通知コードの末尾に不当な文字コードが設定されます。

入力操作と結果（イベント単位の場合）

入力操作	結果
マウスまたはキー操作でのフィールド離脱で，イベントが発生した。	イベント通知項目に，イベント定数を代入する。繰り返しを指定している項目の場合，繰り返しの先頭項目と同じイベント定数を代入する。イベント定数については「16.5.2　イベント定数の論理マップ生成規則」を参照のこと。
指定されたイベント通知コードと対応するキーが入力された。	イベント通知項目の領域にイベント通知コードを代入する。余白がある場合，余白は空白（$(20)_{16}$）で埋める。
指定されたイベント通知コードと対応しないキーが入力された。または MCR 入力によって自動送信された。	初期クリア文字に従う。初期クリア文字が埋字の場合，イベント通知項目は空白（$(20)_{16}$）で埋める。

注

MCR 入力するとき，項目長よりも長いデータを入力した場合は，MCR 入力による自動送信ではなくフィールド離脱による自動送信の扱いとなります。

16.2.5　可変項目

(1)　可変項目の定義

CUI 画面，GUI 画面，および GUI 画面内のフィールドボックスでは，各項目の名称が次の表のように異なります。

表 16–1　可変項目の名称の違い

CUI 画面	GUI 画面	フィールドボックス
出力フィールド	出力テキストボックス	出力フィールド
入出力フィールド	入出力テキストボックス	入出力フィールド

各項目の名称は，それぞれの画面での名称に置き換えてください。

出力項目

出力テキストボックス，または出力フィールドに対して生成されます。

- 出力項目　　：マップ名-FIELDnnnn-O
- 繰り返し回数　：縦，または横の反復回数（出力フィールドだけ）
- 長さ　　：論理項目の長さ。指定がないときはフィールドのデータ長
- データ型　　：テキストボックス，フィールドのデータ型で設定
- 使用目的　　：テキストボックス，フィールドの使用目的で設定

出力項目の使用目的を次の表に示します。

表 16–2　出力項目の使用目的と使用できる項目

使用目的	使用できる項目
数字	文字，数字編集項目
英数	文字
日本語	文字，漢字

入力項目

入出力テキストボックス，または入出力フィールドに対して生成されます。次に示す入力項目以外の定義内容は「出力項目」と同じになります。

- 入力項目　：　マップ名-FIELDnnnn-I

入力項目の使用目的を次の表に示します。

表 16–3　入力項目の使用目的と使用できる項目

使用目的	使用できる項目	使用目的	使用できる項目
数字	文字，数字編集項目	英数	文字
金額	文字編集項目	日本語	文字，漢字
数値	文字編集項目	MCR	文字
カナ	文字	パスワード	文字

入出力項目

入出力テキストボックス，または入出力フィールドに対して生成されます。論理マップは，入力論理マップと出力論理マップに分けて出力されます。定義内容は出力項目と同じになります。

(2)　可変項目の出力論理マップ生成規則とマッピング規則

可変項目の出力論理マップ生成規則と論理マップ生成規則について説明します。

(a)　出力論理マップ生成規則

初期値を設定しない場合

COBOL

COBOL の可変項目の生成規則は，使用目的によってデータ型が変わります。

・数字項目の場合

```
{02|03} マップ名-FIELDnnnn-0  PIC 9(長さ)〔OCCURS 回数〕.
```

・文字項目の場合

```
{02|03} マップ名-FIELDnnnn-0  PIC X(長さ)〔OCCURS 回数〕.
```

・漢字項目の場合

```
{02|03} マップ名-FIELDnnnn-0  PIC N(長さ÷2)〔OCCURS 回数〕.
```

・数字編集項目の場合

```
{02|03} マップ名-FIELDnnnn-0  PIC 編集文字〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_0  〔[回数]〕[長さ];
```

注

- C 言語使用時は，数字編集項目は指定できません。
- 日本語では，「桁寄せ」は左寄せになります。

初期値を指定した場合

初期値を指定した場合の論理マップ生成例は初期値を指定しないときと同じです。

(b)　マッピング規則

初期値を設定しない場合

AP が出力項目データ名の領域に代入した内容と結果

AP が出力項目データ名の領域に代入した内容	マッピングオプション	結果
すべてデータの場合	マージ，論理マップ	代入されたデータを表示する。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例として X'1F414234'または X'1F1F1F'のようなデータが該当する。このとき，表示形態が「全面書換」か「一部上書」によって次のように異なる。 全面書換の場合：何も表示しない。 一部上書の場合：直前に表示したデータのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例として X'4142431F1F'のようなデータが該当する。この場合，「桁寄せ」の指定に従って桁寄せをし，指定した埋字を埋めて表示する。埋字と桁寄せ

AP が出力項目データ名の領域に代入した内容	マッピングオプション	結果
		のマッピング規則については，「16.2.9　埋字と桁寄せ（出力フィールド）」を参照のこと。
－	物理マップ	データを表示しない。

（凡例）

－：該当しない。

初期値を指定した場合

AP が出力項目データ名の領域に代入した内容と結果

AP が出力項目データ名の領域に代入した内容	マッピングオプション	結果
先頭 1 文字がデータ有無コード	マージ，論理マップ	先頭 1 文字がデータ有無コードのときの結果を参照。
すべてデータ		代入されたデータを表示する。
データの後半にデータ有無コード		この場合，「桁寄せ」の指定に従って桁寄せをし，指定した埋字を埋めて表示する（数字編集項目の場合は，編集文字に従って編集してから表示）。埋字と桁寄せのマッピング規則については，「16.2.9　埋字と桁寄せ（出力フィールド）」を参照のこと。
－	物理マップ	先頭 1 文字がデータ有無コードのときの結果を参照。

（凡例）

－：該当しない。

先頭 1 文字がデータ有無コードのときの結果

初期値の内容	結果
すべて空白（$(20)_{16}$）	項目全体を空白で表示する。
すべて 0（$(30)_{16}$）	項目全体を 0 で表示する。
すべてヌル（$(00)_{16}$）	項目をヌルクリアして表示する。
繰り返し文字指定	項目全体を繰り返し文字で表示する。
初期値の長さと画面の表示長とが等しい	初期値を項目に表示する。
初期値の長さが画面の表示長より短い	定義で指定した「桁寄せ」，および埋字に従って表示する。埋字抑止を指定した場合は，左寄せで初期値を項目に表示する。
初期値の長さが画面の表示長より長い	「桁寄せ」に従って桁寄せし，余りを切り捨てて項目を表示する。

(3)　可変項目の入力論理マップ生成規則とマッピング規則

入出力フィールド･テキストを指定した場合の可変項目の入力論理マップ生成規則と論理マップ生成規則について説明します。

(a)　入力論理マップ生成規則

COBOL

COBOL の可変項目の生成規則は，使用目的によってデータ型が変わります。

論理マップ可変部の集団項目化を指定した場合

・数字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC 編集文字.
```

・文字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC X(長さ).
```

・漢字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC N(長さ÷2).
```

論理マップ可変部の集団項目化を指定していない場合

・数字項目の場合

```
02 マップ名-FIELDnnnn-I    PIC 9(長さ)  〔OCCURS 回数〕.
```

・文字項目の場合

```
02 マップ名-FIELDnnnn-I     PIC X(長さ)  〔OCCURS 回数〕.
```

・漢字項目の場合

```
02 マップ名-FIELDnnnn-I     PIC N(長さ÷2)  〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_I 〔[回数]〕[長さ];
```

(b)　マッピング規則

入力操作と入力項目データ名の領域の内容

使用目的が日本語以外の場合

入力操作	結果（入力項目データ名の領域の内容）
半角のデータを入力	データを入力項目データ名の領域に代入する。
全角，または半角／全角のデータを入力※	エラー通知文字に従ってエラー通知文字で入力項目データ名の領域をクリアする。コードエラーを検出したときのマッピング規則については，「16.2.13　コードエラーを検出したとき」を参照のこと。
フィールドキーを押す（すべて $(00)_{16}$ のデータ入力）	データ消去通知文字に従って入力項目データ名の領域をクリアする。フィールドキーを押したときのマッピング規則については，「16.2.10　フィールドキーを押したとき」を参照のこと。
入力操作なし	（初期値を指定している） 初期値で指定した定数を入力項目データ名の領域に代入する。 （初期値を指定していない） 初期クリア文字に従って入力データ名の領域をクリアする。入力操作がなかったときのマッピング規則については，「16.2.12　入力操作がなかったとき」を参照のこと。

注※

カナのときだけ該当します。数字または英数字のときは，キー入力時点で入力できません。

使用目的が日本語かつデータ型が漢字の場合

入力操作	結果（入力項目データ名の領域の内容）
全角のデータを入力または半角スペースを入力	データを入力項目データ名の領域に代入する。
フィールドキーを押す（すべて$(00)_{16}$のデータ入力）	データ消去通知文字に従って入力項目データ名の領域をクリアする。フィールドキーを押したときのマッピング規則については，「16.2.10　フィールドキーを押したとき」を参照のこと。
入力操作なし	（初期値を指定している） 初期値で指定した定数を入力項目データ名の領域に代入する。 （初期値を指定していない） 初期クリア文字に従って入力データ名の領域をクリアする。入力操作がなかったときのマッピング規則については，「16.2.12　入力操作がなかったとき」を参照のこと。

使用目的が日本語かつデータ型が文字の場合

入力操作	結果（入力項目データ名の領域の内容）
$(00)_{16}$以外のデータを入力	データを入力項目データ名の領域に代入する。
フィールドキーを押す（すべて$(00)_{16}$のデータ入力）	データ消去通知文字に従って入力項目データ名の領域をクリアする。フィールドキーを押したときのマッピング規則については，「16.2.10　フィールドキーを押したとき」を参照のこと。
入力操作なし	（初期値を指定している） 初期値で指定した定数を入力項目データ名の領域に代入する。 （初期値を指定していない） 初期クリア文字に従って入力データ名の領域をクリアする。入力操作がなかったときのマッピング規則については，「16.2.12　入力操作がなかったとき」を参照のこと。

16.2.6　入力数字編集項目

入力数字編集項目の論理マップ生成規則とマッピング規則について説明します。

(1)　入力数字編集項目の定義

入出力テキストボックス，または入出力フィールドで，データ型が数値，金額のフィールドに対して生成されます。

- 入力項目　　：マップ名-FIELDnnnn-I
  数字編集文字列：データ型で指定できる数字編集文字列は次のとおり。「9」，「V」，「S」
- 繰り返し回数　：縦，または横の反復回数（入出力フィールドだけ）

(2)　論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-H 〔OCCRUS 回数〕.
  {03|04} マップ名-FIELDnnnn-I  PIC 編集文字.
```

(3)　マッピング規則

入力操作と結果

入力操作	結果
フィールドキーを押した場合	データ消去通知文字に従って入力項目データ名の領域をクリアする。データ消去通知文字が埋字として指定してあるときは，0（$(30)_{16}$）でクリアする。
入力操作がない，または END キーを押した場合	（初期値が指定してある場合） 初期値で指定した定数を入力項目データ名の領域に代入する。 （初期値が指定していない場合） 初期クリア文字が埋字として指定してあるときは，0（$(30)_{16}$）でクリアする。
全桁に空白を入力した場合	空白（$(20)_{16}$）で入力項目データ名の領域をクリアする。また，NULL と空白を混在している場合も，すべて空白となる。
数字項目として正しいデータを入力した場合	数字編集項目の指定に従って入力項目データ名の領域に値を代入する。
数字項目として不正なデータを入力した場合※	入力チェックを指定した場合，入力時にエラーとなる。入力チェックを指定していない場合，エラー通知文字に従って入力項目データ名の領域をクリアする。エラー通知文字の標準は HIGH（X'FF'）である。

注※

不正なデータとは次のとおりです。

- 数字入力として不正な場合
  - ・コンマ，ピリオド，符号，¥が連続している。
  - ・数字の間に空白がある。
  - ・数字（0～9）が一つもない。
  - ・＊の直後が数字（0～9）でない。
- ピクチャと一致しない場合
  - ・オーバフロー，アンダフロー。
  - ・V なしのピクチャに対し，小数点がある。
  - ・S なしのピクチャに対し，マイナス符号がある。
- 「入力済み」属性を指定して，数字編集文字以外や桁数が合わないデータを表示して画面確定している場合

16.2.7　動的変更

動的変更の論理マップ生成規則とマッピング規則について説明します。動的変更は，入出力テキストボックスダイアログ，または入出力フィールドダイアログで，出力だけに関する制御項目は出力テキストボックスダイアログ，または出力フィールドダイアログで「動的変更（AP から表示属性を変更する）」を選んだときに生成されます。

(1)　動的変更の定義

- 制御項目のデータ名：マップ名-FIELDnnnn-A
  データ名は，各オブジェクトのダイアログの「データ名」で変更できます。
- 長さ：2
  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「キャラクタコントロール（キーエントリ／選択エントリ）」を選択し，「修飾名長」で長さを設定します。

- 繰り返し：フィールドの繰り返しと同じ

(2)　論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-A  PIC X(長さ) 〔OCCRUS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_A 〔[回数]〕[長さ];
```

(3)　マッピング規則

AP が動的変更データ名の領域に代入した内容と結果

上記ダイアログで「動的変更（AP から表示属性を変更する)」を選んだ場合

AP が動的変更データ名の領域に代入した内容	マッピングオプション	結果
修飾名と同じ	マージ	修飾名に対応する表示属性を使って項目を表示する。
先頭にデータ有無コード，または修飾名以外（上記以外）		標準の属性を使って項目を表示する。
修飾名と同じ	論理マップ	修飾名に対応する属性を使って項目を表示する。
先頭にデータ有無コード，または修飾名以外（上記以外）		(表示形態が「全面書換」の場合) 標準の属性を使って表示する。 (表示形態が「一部上書」で直前に表示したマップ名と同じ場合) 直前の画面の表示属性を変更しないでデータだけを書き換える。 (表示形態が「一部上書」で直前に表示したマップ名と異なる場合) 標準の属性を使って表示する。
－	物理マップ	標準の属性を使って項目を表示する。

(凡例)

－：該当しない。

注

修飾名と修飾名に対応したテキスト・フィールド変更属性情報は，ドローセットアップの「表示属性の動的変更」の「キャラクタコントロール」タブで変更できます。

上記ダイアログで「動的変更（AP から表示属性を変更する)」を選ばない場合

標準の属性を使って項目を表示します。

16.2.8　入力バイト数格納項目

入力バイト数格納項目の論理マップ生成規則とマッピング規則について説明します。ドローセットアップの論理マップ属性で「入力データ長格納領域の生成」を選んだ場合対象になります。対象となるオブジェクトは，入出力テキスト（フィールド）になります。

(1)　入力バイト数格納項目の定義

ドローセットアップの論理マップ属性で展開方式の「入力データ長格納領域の生成」を選んだときに生成されます。

(2)　論理マップ生成規則

COBOL

・Windows リトルエンディアン用

```
{02|03} マップ名-FIELDnnnn-L  PIC S9(4)  COMP-5 〔OCCURS 回数〕.
```

・Windows ビッグエンディアン用，UNIX 用

```
{02|03} マップ名-FIELDnnnn-L  PIC S9(4)  COMP 〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_L〔[回数]〕[2];
```

(3)　マッピング規則

入力操作と結果（制御項目の内容）

使用目的がカナ，英数の場合

入力操作	結果（制御項目の内容）
半角のデータを入力	入力データ長を制御項目に代入する（埋字を含まない）。
全角または半角／全角混在のデータを入力，またはフィールドキーを押す。	$(00)_{16}$ を制御項目に代入する。
入力操作なし	入出力テキストボックス・フィールドダイアログで「入力済み（入力しなくても表示データを AP に返す）」を選び，画面出力データが半角のときは画面上のデータ長を制御項目に代入する。それ以外は$(00)_{16}$ を代入する。

使用目的が日本語でデータ型が漢字の場合

入力操作	結果（制御項目の内容）
フィールドキーを押す	$(00)_{16}$ を制御項目に代入する。
全角のデータを入力	入力データ長（バイト数）を制御項目に代入する（埋字を含まない）。
入力操作なし	入出力テキストボックス・フィールドダイアログで「入力済み（入力しなくても表示データを AP に返す）」を選び，画面出力データが全角のときは画面上のデータ長を制御項目に代入する。それ以外は$(00)_{16}$ を代入する。

使用目的が日本語でデータ型が文字の場合

入力操作	結果（制御項目の内容）
$(00)_{16}$ 以外のデータを入力	入力データ長を制御項目に代入する（埋字を含まない）。
フィールドキーを押す	$(00)_{16}$ を制御項目に代入する。
入力操作なし	入出力テキストボックス・フィールドダイアログで「入力済み（入力しなくても表示データを AP に返す）」を選んだ場合，画面上のデータ長を制御項目に代入する。それ以外は$(00)_{16}$ を代入する。

使用目的が数字，数値，金額の場合

入力操作	結果（制御項目の内容）
数字項目として正しいデータを入力	論理項目長を制御項目に代入する。

入力操作	結果（制御項目の内容）
数字項目として不正なデータを入力， フィールドキーを押す， 入力操作なし，または空白を入力。	$(00)_{16}$ を制御項目に代入する。

16.2.9　埋字と桁寄せ（出力フィールド）

出力テキストボックス，または出力フィールドの埋字と桁寄せに関するマッピング規則について説明します。

定義	結果（画面表示結果）
埋字に「埋めない」以外を指定	桁寄せ向きに従って桁寄せし※，埋字に指定した文字で埋める。
埋字に「埋めない」を指定	初期値，および出力論理データは，桁寄せ向きに関係なく左寄せし，埋字を代入しない。ただし，埋字を代入しないことで画面上の次の項目位置はドロー画面で指定した位置に表示され，詰まることはない。表示形態が「全面書換」の場合，ヌルが入り，「一部上書」の場合，前画面の属性をそのまま引き継ぎデータだけを書き換える。

注※

使用目的が日本語の場合，左寄せ固定になります。

16.2.10　フィールドキーを押したとき

入出力テキストボックス・フィールドに対して，フィールドキーを押したときのマッピング規則について説明します。フィールドキーに関するマッピング規則は，ドローセットアップの論理マップ属性にある「データ消去通知文字」と「初期クリア文字」の指定で結果が異なります。

また，次の操作をした場合，フィールドキーを押したときと同じ扱いになります。

- $(00)_{16}$ でクリアされていて，入出力テキストボックス・フィールドダイアログで「入力済み（入力しなくても表示データを AP に返す）」を指定しているフィールドに何もデータを入力しないで確定した。
- ウィンドウ表示制御の表示形態が「全面書換」でデータを何も表示しないフィールドに何もデータを入力しないで確定した。

「データ消去通知文字」が「なし」の場合

初期クリア文字	各フィールドの初期値	結果（入出力項目データ名の領域の内容）
なし	あり	初期値を設定する。
	なし	マッピング制御機能は論理マップに何もセットしないため，マッピング制御機能に渡された時点の論理マップの内容のままになる。
埋字	あり	初期値を設定する。
	なし	各項目の埋字を設定する。
その他 （文字指定）	あり	初期値を設定する。
	なし	初期クリア文字を設定する。

「データ消去通知文字」が「埋字」の場合

各項目の埋字を設定します。

「データ消去通知文字」がその他の場合

データ消去通知文字を設定します。

16.2.11　データキーを押したとき

入出力テキストボックス・フィールドに対して，データキーを押したときのマッピング規則について説明します。データキーに関するマッピング規則は，ドローセットアップの論理マップ属性にある「データ消去通知文字」と「初期クリア文字」の指定および表示・印刷セットアップの「キー操作」タブの「データキーの動作」の指定で結果が異なります。

「データキーの動作」が「未入力状態」の場合

初期クリア文字	各フィールドの初期値	結果（入出力項目データ名の領域の内容）
なし	あり	初期値を設定する。
	なし	マッピング制御機能は論理マップに何もセットしないため，マッピング制御機能に渡された時点の論理マップの内容のままになる。
埋字	あり	初期値を設定する。
	なし	各項目の埋字を設定する。
その他（文字指定）	あり	初期値を設定する。
	なし	初期クリア文字を設定する。

「データキーの動作」が「ヌルクリア状態」の場合

各項目に対してフィールドキーを押したときと同じ扱いになります。フィールドキーを押したときのマッピング規則については，「16.2.10　フィールドキーを押したとき」を参照してください。

16.2.12　入力操作がなかったとき

入力操作がなかったときのマッピング規則について説明します。なお，入力操作をしていない場合でも，入力済み属性が有効となる画面を表示すると，表示データを入力した場合と同じ動作となります。

「初期クリア文字」が「なし」の場合

各フィールドの初期値	結果（入出力項目データ名の領域の内容）
あり	初期値を設定する。
なし	マッピング制御機能は論理マップに何もセットしないため，マッピング制御機能に渡された時点の論理マップの内容のままになる。

「初期クリア文字」が「埋字」の場合

各フィールドの初期値	結果（入出力項目データ名の領域の内容）
あり	初期値を設定する。
なし	各項目の埋字を設定する。

「初期クリア文字」がその他（文字指定）の場合

各フィールドの初期値	結果（入出力項目データ名の領域の内容）
あり	初期値を設定する。
なし	初期クリア文字を設定する。

16.2.13　コードエラーを検出したとき

コードエラーを検出したときのマッピング規則について説明します。コードエラーを検出したときのマッピング規則は，ドローセットアップの論理マップ属性にある「初期クリア文字」と「エラー通知文字」の指定で結果が異なります。

「初期クリア文字」が「なし」の場合

エラー通知文字	結果（入出力項目データ名の領域の内容）
なし	マッピング制御機能は論理マップ中に何もセットしないため，マッピング制御機能に渡された時点の論理マップの内容のままになる。
埋字	各項目の埋字を設定する。
その他（文字指定）	エラー通知文字を設定する。

「初期クリア文字」が「埋字」の場合

エラー通知文字	結果（入出力項目データ名の領域の内容）
なし	各項目の埋字を設定する。
埋字	各項目の埋字を設定する。
その他（文字指定）	エラー通知文字を設定する。

「初期クリア文字」がその他（文字指定）の場合

エラー通知文字	結果（入出力項目データ名の領域の内容）
なし	初期クリア文字を設定する。
埋字	各項目の埋字を設定する。
その他（文字指定）	エラー通知文字を設定する。

16.2.14　埋字と桁寄せ（入力フィールド）

入出力テキストボックス・フィールドの埋字と桁寄せに関するマッピング規則について説明します。

定義	結果（入出力項目データ名の領域）
埋字に「埋めない」以外を指定	桁寄せ向きに従って桁寄せし※，埋字に指定した文字で埋める。

定義	結果（入出力項目データ名の領域）
埋字に「埋めない」を指定	初期値，および入力論理データは，桁寄せ向きに従って桁寄せし，埋字を代入しないで初期クリア文字が入る。

注※

使用目的が日本語の場合，左寄せ固定になります。

16.2.15　MCR 入力

MCR 入力のマッピング規則について説明します。論理マップ生成規則については，「16.2.5(3)　可変項目の入力論理マップ生成規則とマッピング規則」を参照してください。MCR は入出力テキストボックス・フィールドの使用目的を MCR にしておく必要があります。

使用目的が MCR の場合

入力操作	結果（入力項目データ名領域）
半角のカードデータを入力※	データを論理項目に代入する。カードデータ長が論理項目長より短いときは桁寄せ向きに従って桁寄せし，埋字指定の内容で埋字を代入する。
入力操作なし，またはキー入力でデータを入力	初期値で指定した定数を論理項目に代入する。初期値が指定されていない場合，初期クリア文字に従って論理項目をクリアする。

注※

ID コードが「>」のカードデータの場合になります。「>」以外の ID コードを使った場合は，入力操作なしと同じ動作となります。また，「>」だけを入力した場合は，データ消去通知文字に従って論理項目をクリアします。

使用目的が MCR 以外の場合

可変項目の場合と同じです。「16.2.5(3)　可変項目の入力論理マップ生成規則とマッピング規則」を参照してください。

16.2.16　隠しフィールド

(1)　隠しフィールドの定義

隠しフィールドは，画面属性ダイアログで隠しフィールドを設定すると生成されます。

(2)　論理マップ生成規則

隠しフィールドの論理マップ生成規則について説明します。

COBOL

```
{02|03} マップ名-TRAN-I  PIC X(9).
```

C 言語

```
unsigned char マップ名_TRAN_I[9];
```

(3)　マッピング規則

入力操作と結果

入力操作	結果
画面からの入力	隠しフィールドデータ名の領域に，画面定義時に設定した値が左寄せで入り，残りをスペースで埋字する。

16.2.17　確定キーの制御情報

(1)　制御情報の標準値

- 確定キーの制御データ名：マップ名-INCO

  データ名は，ドローセットアップの「ドローの設定」から表示する，表示属性の動的変更ダイアログの「データ名」タブで変更できます。

- 長さ：2

  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定...］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「確定キー属性」を選択し，「修飾名長」で長さを設定します。

(2)　論理マップ生成規則

確定キーの制御情報の論理マップ生成例について説明します。

COBOL

```
{02|03} マップ名-INCO  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_INCO[長さ];
```

(3)　マッピング規則

確定キーの制御情報のマッピング規則について説明します。

AP が確定キーの制御情報のデータ名の領域に代入した内容とその結果

AP が確定キーの制御情報のデータ名の領域に代入した内容	結果
修飾名と同じ	確定キーの制御情報の修飾名に対応するウィンドウ制御情報を使って画面が表示される。ただし，標準属性が「キー押下無効」に設定されているキーは常にキー操作が無効となる。
データの先頭またはすべてがデータ有無コード，または修飾名以外（上記以外）	標準の確定キーの制御情報を使って画面が表示される。次に示す条件の場合，直前の表示画面と同じ画面のときは同じキー制御情報を使って表示し，直前の表示画面と異なるときは，標準のキー制御情報を使って表示する。 • 画面属性ダイアログの表示形態が「一部上書」で，かつマッピングオプションで論理マップを指定した場合。 • 画面属性ダイアログの表示形態が「自動」で，かつマッピングオプションで論理マップを指定した場合。

注

修飾名と修飾名に対応した確定キー制御情報は，ドローセットアップの「表示属性の動的変更」の「確定キー属性」タブで変更できます。

16.2.18　下位項目

(1)　下位項目の定義

下位項目は主にテキスト，フィールドおよびラベルに対して，データ型が「文字（XX)」のときに指定します。各オブジェクトの定義ダイアログで設定すると，生成されます。

(2)　下位項目の出力論理マップ生成規則とマッピング規則

下位項目の出力論理マップ生成規則について説明します。

(a)　出力論理マップ生成規則

COBOL

・数字項目の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC  9(長さ).
```

・文字項目の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC  X(長さ).
```

・1 文字の反復の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC  X(1) OCCURS 回数.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_nnn_O 〔[回数]〕[長さ];
```

(b)　マッピング規則

出力結果

上位項目単位にマッピングします。下位項目を二つ定義し，二つ目の下位項目にデータ有無コード以外のデータを設定し，一つ目の下位項目にデータ有無コードを設定しても，埋字の対象にはなりません。

(3)　下位項目の入力論理マップ生成規則とマッピング規則

下位項目の入力論理マップ生成規則について説明します。

(a)　入力論理マップ生成規則

COBOL

・数字項目の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  9(長さ).
```

・文字項目の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  X(長さ).
```

・1 文字の反復の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  X(1) OCCURS 回数.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_nnn_I 〔[回数]〕[長さ];
```

(b)　マッピング規則

入力結果

上位項目単位にマッピングします。

16.2.19　フレーム

(1)　フレームの定義

フレームをドローで定義し，フレーム中に可変項目を定義することで，生成されます。

- フレームのデータ名：マップ名-FRAMEnnnn-O
  データ名はフレームダイアログの「データ名」で変更できます。
- フレームの反復
  方向：縦，または横
  回数：反復回数
  間隔：反復間隔をます目単位で指定

オブジェクトに対応するデータ項目をまとめて集団項目にします。

COBOL

```
{02|03} マップ名-FRAMEnnnn-{O|I} 〔OCCURS n〕.
   {04|05}マップ名……オブジェクトに対応する項目
```

C 言語

```
struct {
   オブジェクトに対応する項目
      …} マップ名_FRAMEnnnn_{O|I}〔[n]〕;
```

16.2.20　予約テキスト・フィールド

予約テキスト・フィールドのマッピング規則を次に示します。なお，論理マップに予約テキスト・フィールドの情報は生成されません。

(1)　ドローで定義した予約項目名と結果

ドローで定義した項目名	データの表示結果
OpenTP1 がサポートしている予約項目名を指定	•「指定した項目長」＝「OpenTP1 で決められた項目長」 OpenTP1 が設定するデータが表示される。 •「指定した項目長」＞「OpenTP1 で決められた項目長」 データが左寄せで表示される。また，残りは空白が表示される。 •「指定した項目長」＜「OpenTP1 で決められた項目長」 データが左寄せで表示される。また，表示しきれないデータは切り捨てられる。
OpenTP1 がサポートしていない予約項目名を指定	項目には何も表示されない。

OpenTP1 がサポートしていない予約項目を指定した場合はエラーとなります。このとき詳細コードには $1036(040C)_{16}$ が返されます。詳細については「付録 F　リターンコードと詳細コード」を参照してください。

16.3　可変部（GUI 画面固有）

ここでは，GUI 画面固有の論理マップ生成規則について説明します。フィールドボックス内の可変項目の論理マップ生成規則については，「16.2　可変部（画面共通）」を参照してください。なお，この節では，COBOL の数字項目の記述を次のようにしています。

- PIC 9（長さ）と表記している項目
  実際に生成される論理マップは，PIC 99999 のように 9 が長さ分生成されます。

16.3.1　ポップアップメニュー

(1)　ポップアップメニューの定義

ポップアップメニューには，データを入力するテキストボックスと入力するデータを表示するメニューがあります。テキストボックスをポップアップフィールドといい，表示するメニューをポップアップメニューといいます。

- **ポップアップフィールドの定義**
  可変ポップアップに対して生成されます。定義については，「16.2.5　可変項目」，および「16.2.7　動的変更」と同じです。
- **ポップアップメニューの定義**
  可変ポップアップの「メニュー」で指定した内容で生成されます。
  - 可変用論理テーブル：マップ名-可変用論理テーブル名の指定値（標準のデータ名：マップ名-POPUPnnnn-O）
  - 回数：繰り返し回数の指定値
  - 可変メニューラベル：マップ名-可変メニューラベル名の指定値（標準のデータ名：マップ名-POPUP-LABELnnnn-O）
  - データ型：データ型の指定値
  - 長さ：ラベルのデータ長の指定値
  - 可変メニューコード：マップ名-可変メニューコード名の指定値（標準のデータ名：マップ名-POPUP-CODEnnnn-O）
  - コードの長さ：コード項目のデータ長の指定値
  - 可変メニュー選択ラベル：マップ名-可変メニュー選択ラベル名の指定値（標準のデータ名：マップ名-POPUP-TEXTnnnn-O）
  - 選択ラベルのデータ型：選択ラベルのデータ型の指定値
  - 選択ラベルの長さ：選択ラベルのデータ長の指定値
  - 可変メニューアクセスキー：マップ-可変メニューアクセスキー名の指定値（標準のデータ名：マップ名-POPUP-KEYnnnn-O）
  - メニューデータファイル：マップ名-メニューデータファイル項目名の指定値（標準のデータ名：マップ名-POPUP-FILEnnnn-O）
  - データ型：文字（XX）
  - 長さ：ファイル名指定：12
    フルパス 64：64
    フルパス 128：128

フルパス 259：259

また，使用目的は次のとおりです。なお，可変メニューラベル，可変メニュー選択ラベルには，使用目的に関係なく，文字，漢字，数字から選択できます。

使用目的	使用できる項目
POP（手動-数字）	文字
POP（手動-カナ）	文字
POP（手動-英数）	文字
POP（手動-日本語）	文字，漢字
POP（自動-英数）	文字
POP（自動-日本語）	文字，漢字

(2)　ポップアップフィールドの論理マップ生成規則とマッピング規則

ポップアップフィールドで生成される論理マップには，項目属性，出力，入力の三つの論理マップがあります。

(a)　ポップアップ項目属性の論理マップ生成規則とマッピング規則

ポップアップ項目属性の論理マップ生成規則とマッピング規則について説明します。

●論理マップ生成規則

- **「動的変更（AP から表示属性を変更する)」を指定した場合**

  COBOL

  ```
  {02|03} マップ名-FIELDnnnn-A  PIC X(長さ).
  ```

  C 言語

  ```
  unsigned char マップ名_FIELDnnnn_A[長さ];
  ```

- **「動的変更（AP から表示属性を変更する)」を指定しない場合**

  定義に関係なく標準の属性を使って表示します。

●マッピング規則

- **AP が制御項目に代入した内容と表示結果**

AP が制御項目に代入した内容	マッピングオプション	表示結果
ドローセットアップで指定した修飾名と同じ修飾名を指定	マージ	ポップアップ項目の修飾名に対応する属性を使って表示する。
先頭またはすべてにデータ有無コード，または定義で指定していない修飾名を指定		標準の属性を使って表示する。
ドローセットアップで指定した修飾名と同じ修飾名を指定	論理マップ	ポップアップ項目の修飾名に対応する属性を使って表示する。
先頭またはすべてにデータ有無コード，または定義で指定していない修飾名を指定		画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 標準の属性を使って表示する。

APが制御項目に代入した内容	マッピングオプション	表示結果
		•「一部上書」で直前に表示したマップと同じ場合 前回の属性のまま表示する。 •「一部上書」で直前に表示したマップと異なる場合 標準の属性を使って表示する。
−	物理マップ	標準の属性を使って表示する。

（凡例）
−：該当しない。

(b)　ポップアップ項目の出力論理マップ生成規則とマッピング規則

ポップアップ項目の出力論理マップ生成規則とマッピング規則について説明します。

●出力論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-O  PIC X(長さ)  〔OCCURS 回数〕.
```

C言語

```
unsigned char マップ名_FIELDnnnn_O〔[回数]〕[長さ];
```

●マッピング規則

- APが論理項目に代入した内容と表示結果

《初期値を指定しない場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ， 論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$のデータ		ヌルクリアして表示する。
すべて$(20)_{16}$のデータ		空白で表示する。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。 このとき，画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 ヌルクリアして表示する。 •「一部上書」で直前に表示したマップと同じ場合 前回のデータのまま表示する。 •「一部上書」で直前に表示したマップと異なる場合 ヌルクリアして表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。 論理項目の桁寄せ向きと，指定した埋字の内容に従って表示する。
画面の表示長より長いデータ		論理項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。

APが論理項目に代入した内容	マッピングオプション	表示結果
−	物理マップ	何も表示しない。

（凡例）

−：該当しない。

《初期値を指定した場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
先頭1文字がデータ有無コード	マージ，論理マップ	《先頭1文字がデータ有無コードのときの結果》を参照のこと。
すべてのデータ		代入されたデータを表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。論理項目の向きと，指定した埋字の内容に従って表示する。
−	物理マップ	《先頭1文字がデータ有無コードのときの結果》を参照のこと。

（凡例）

−：該当しない。

《先頭1文字がデータ有無コードのときの結果》

初期値の内容	結果
すべて空白（$(20)_{16}$）	項目全体を空白で表示する。
すべて0（$(30)_{16}$）	項目全体を0で表示する。
すべてヌル（$(00)_{16}$）	項目全体をヌルクリアして表示する。
繰り返し文字指定	項目全体を繰り返し文字で表示する。
初期値の長さと画面の表示長が等しい	初期値を項目に表示する。
初期値の長さが画面の表示長より短い	定義で指定した「桁寄せ」，および埋字に従って表示する。埋字なしを指定した場合は，左寄せで初期値を項目に表示する。
初期値の長さが画面の表示長より長い	「桁寄せ」に従って桁寄せし，余りを切り捨てて項目を表示する。

(c)　ポップアップ項目の入力論理マップ生成規則とマッピング規則

ポップアップ項目の入力論理マップ生成規則とマッピング規則について説明します。

●入力論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I   PIC {X(長さ)|N(長さ÷2)}  〔OCCURS 回数〕.
```

C言語

```
unsigned char マップ名_FIELDnnnn_I〔[回数]〕[長さ];
```

●マップピング規則

- 入力操作と結果（論理項目の内容）

《自動ポップアップの場合》

入力操作	結果（論理項目の内容）
ポップアップ項目にカーソルを位置づけ表示されたポップアップメニューの中からメニューを選ぶ	選んだメニュー（フォーカスを位置づけたラベル）に対応した定数を論理項目に代入する※。
ポップアップ項目にカーソルを位置づけ表示されたポップアップメニューの中からメニューを選ばない，またはポップアップ項目にカーソルを位置づけない	• 論理項目の初期値を指定している場合，指定した定数を論理項目に代入する。 • 論理項目の初期値を指定していない場合，初期クリア文字の指定に従って論理項目をクリアする。
メニューのアクセスキー項目に指定されているキーを入力	• アクセスキーに対応した通知コードを，論理項目に代入する※。
メニューのアクセスキー項目に指定されていないキーを入力	• 論理項目の初期値を指定している場合，初期値を論理項目に代入する。 • 論理項目の初期値を指定していない場合，初期クリア文字の指定に従って論理項目をクリアする。

注※

「メニューデータをファイルで指定する」を指定している場合，使用目的が日本語以外で，かつ通知コードが全角を含む場合には，エラー通知文字が返ります。

《手動ポップアップの場合》

入力操作	結果（論理項目の内容）
ポップアップ項目にカーソルを位置づけ表示されたポップアップメニューの中からメニューを選ぶ	選んだメニュー（フォーカスを位置づけたラベル）に対応した定数を論理項目に代入する※1。
ポップアップ項目にカーソルを位置づけ表示されたポップアップメニューの中からメニューを選ばない	• 論理項目の初期値を指定している場合，指定した定数を論理項目に代入する。 • 論理項目の初期値を指定していない場合，初期クリア文字の指定に従って論理項目をクリアする。
半角のデータを入力※2	データを論理項目に代入する。
全角，または半角／全角混在のデータを入力※2	
フィールドキーを押す（すべて$(00)_{16}$）データ入力※2	データ消去通知文字に従って論理項目をクリアする。
ポップアップ項目にカーソルを位置づけない	• 論理項目の初期値を指定している場合，指定した定数を論理項目に代入する。 • 論理項目の初期値を指定していない場合，初期クリア文字の指定に従って論理項目をクリアする。
メニューのアクセスキー項目に指定されているキーを入力	• アクセスキーに対応した通知コードを，論理項目に代入する※1。

入力操作	結果（論理項目の内容）
メニューのアクセスキー項目に指定されていないキーを入力	• 論理項目の初期値を指定している場合，初期値を論理項目に代入する。 • 論理項目の初期値を指定していない場合，初期クリア文字の指定に従って論理項目をクリアする。

注※1

「メニューデータをファイルで指定する」を指定している場合，使用目的が日本語以外で，かつ通知コードが全角を含む場合には，エラー通知文字が返ります。

注※2

ポップアップ項目にカーソルを位置づけてデータを入力する場合が該当します。

(3)　ポップアップメニューの論理マップ生成規則とマッピング規則

(a)　メニュー属性のマッピング規則

標準の属性を使ってメニューを表示します。

(b)　ポップアップメニューラベルの論理マップ生成規則とマッピング規則

ポップアップメニューラベルの論理マップ生成規則とマッピング規則について説明します。

●ラベルの論理マップ生成規則

・可変用論理テーブル名を指定した場合

COBOL

```
{02|03} マップ名-POPUPnnnn-O  〔OCCURS 回数〕.
{03|04} マップ名-POPUP-CODEnnnn-O  PIC {X(長さ)|N(長さ÷2)}.
{03|04} マップ名-POPUP-LABELnnnn-O  PIC {X(長さ)|N(長さ÷2)|編集文字}.
{03|04} マップ名-POPUP-KEYnnnn-O  PIC X(長さ).                    ……※1
{03|04} マップ名-POPUP-TEXTnnnn-O  PIC {X(長さ)|N(長さ÷2)|編集文字}.…※2
```

C 言語

```
struct {
    unsigned char マップ名_POPUP_CODEnnnn-O[長さ];
    unsigned char マップ名_POPUP_LABELnnnn-O[長さ];
    unsigned char マップ名_POPUP_KEYnnnn-O[長さ];     ……※1
    unsigned char マップ名_POPUP_TEXTnnnn-O[長さ];    ……※2
}マップ名_POPUPnnnn_O 〔[回数]〕 ;
```

注※1

アクセスキーを指定すると展開されます。

注※2

選択ラベルと通知コードを同じ値にしない場合に展開されます。

●マッピング規則

• AP が論理項目に代入した内容と表示結果

《ラベル項目を指定した場合》

AP が論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$※1		ヌルクリアして表示する。

APが論理項目に代入した内容	マッピングオプション	表示結果
すべて$(20)_{16}$※1		空白で表示する。
先頭1文字がデータ有無コード※2		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 コード項目のデータがあるときは，コード項目のデータを表示する。コード項目のデータがないときは，何も表示しない。 •「一部上書」の場合 前回のデータのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。指定したラベル項目の桁寄せ向き，および埋字の内容に従って表示する。
画面の表示長より長いデータ		ラベル項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 何も表示しない。 •「一部上書」の場合 前回のデータのまま表示する。

（凡例）

－：該当しない。

注※1

メニューの後半に該当する項目がある場合は，項目は圧縮されます。

注※2

ラベル項目のデータ型とコード項目のデータ型が同じ場合だけ，ラベル項目にデータ有無コードを設定できます。

《コード項目を指定した場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ※	マージ，論理マップ	代入されたデータをコードとして扱う。
すべて$(00)_{16}$※		ヌルクリアをコードとして扱う。
すべて$(20)_{16}$※		空白をコードとして扱う。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 コードなしとする。 •「一部上書」の場合

APが論理項目に代入した内容	マッピングオプション	表示結果
		前回のコードをそのまま扱う。
データの後半にデータ有無コード※		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。指定されたデータをコードとして扱う。
－	物理マップ	画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 何も表示しない。 •「一部上書」の場合 前回のコードをそのまま扱う。

（凡例）

－：該当しない。

注※

選択されたメニュー項目に選択ラベルの指定がない場合，対応するテキストまたはフィールドの桁寄せ，および埋字されたデータが入力時に論理項目に代入されます。選択ラベルの指定がある場合は，コードの桁寄せ，および埋字されたデータが入力時に論理項目に代入されます。ドローで定義した場合，対応するテキストまたはフィールドとコードの桁寄せ，および埋字は同じになります。

《選択ラベル項目を指定した場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$		ヌルクリアして表示する。
すべて$(20)_{16}$		空白で表示する。
先頭1文字がデータ有無コード※		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 コード項目のデータがあるときは，コード項目のデータを表示する。コード項目のデータがないときは，何も表示しない。 •「一部上書」の場合 前回のデータのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。指定された選択ラベル項目の桁寄せ向き，および埋字の内容に従って表示する。
画面の表示長より長いデータ		選択ラベル項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 何も表示しない。

APが論理項目に代入した内容	マッピングオプション	表示結果
		• 「一部上書」の場合 前回のデータのまま表示する。

（凡例）

－：該当しない。

注※

選択ラベル項目のデータ型と，コード項目のデータ型が同じ場合だけ，選択ラベル項目にデータ有無コードを設定できます。

《アクセスキーを指定した場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ， 論理マップ	• 指定されたデータがすべて半角英数の場合，指定されたデータがアクセスキーとして有効となる。 • 指定されたデータに半角英数以外の文字が含まれる場合，指定された項目のアクセスキーは無効となる。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，画面属性の表示形態によって次のように変わる。 • 「全面書換」の場合 対応するアクセスキーなしとなる。 • 「一部上書」の場合 前回のアクセスキーがそのまま有効となる。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。指定された項目のアクセスキーは無効となる。
－	物理マップ	画面属性の表示形態によって次のように変わる。 • 「全面書換」の場合 対応するアクセスキーなしとなる。 • 「一部上書」の場合 前回のアクセスキーがそのまま有効となる。

（凡例）

－：該当しない。

(c)　ポップアップメニューファイル使用時の論理マップ生成規則とマッピング規則

ポップアップメニューファイルを使用したときの，論理マップ生成規則とマッピング規則について説明します。

●出力論理マップ生成規則

COBOL

```
{02|03} マップ名-POPUP-FILEnnnn-O  PIC X(長さ).
```

C言語

```
unsigned char マップ名_ POPUP_FILEnnnn-O[長さ];
```

●マッピング規則

- AP が論理項目に代入した内容と表示結果

AP が論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ， 論理マップ	(指定されたファイルが存在した場合) 指定されたファイルを読み込み，メニューを表示する。 (指定されたファイルが存在しない場合) メニューを表示しない※。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例として X'1F414234'または X'1F1F1F'のようなデータが該当する。このとき，画面属性の表示形態によって次のように変わる。 •「全面書換」の場合 画面定義で可変ポップアップメニューの「メニューデータをファイルで指定する」が選択され，データ名の「初期値」にポップアップメニューファイル名が指定されている場合は，指定されたファイルを読み込み，メニューを表示する。画面定義で初期値が設定されていない場合は，メニューを表示しない※。 •「一部上書」の場合 前回のメニューデータをそのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例として X'4142431F1F'のようなデータが該当する。 (指定されたファイルが存在した場合) 指定されたファイルを読み込み，メニューを表示する。 (指定されたファイルが存在しない場合) メニューを表示しない※。

注※

1 階層の空のメニューを表示します。

●入力論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I   PIC {X(長さ)|N(長さ÷2)}  〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_I〔[回数]〕[長さ];
```

●入力論理マップのマッピング規則

- 入力操作と結果（論理項目の内容）

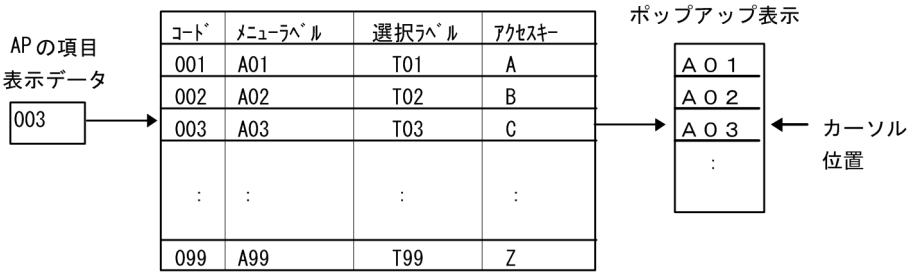
ポップアップメニューと同様です。詳細については，「16.3.1(2)(c)　ポップアップ項目の入力論理マップ生成規則とマッピング規則」を参照してください。

(d)　ポップアップメニューの表示

ポップアップメニューの表示に関するマッピング規則について説明します。項目の表示データは対応するポップアップメニューコードの値によって表示結果が異なります。

- メニューコード内に項目表示データの値があり，項目長とメニューコード長が等しい

  先頭のメニューから表示し，項目表示データと同じメニューコードに対応するメニューに，カーソルを位置づけます。



- メニューコード内に項目表示データの値がないか，項目長とメニューコード長が等しくない

  先頭メニューを表示し，カーソルを位置づけます。

16.3.2　メニューバー

(1)　メニュー制御項目の定義

メニュー定義ダイアログで「動的変更（AP から表示属性を変更する)」を選び，制御項目名を指定することで生成されます。

- メニュー制御項目のデータ名：マップ名-MENUnnnn-A（標準値）

  データ名は，画面属性ダイアログの「メニューバー定義」から表示する，メニューバー定義ダイアログの「制御項目データ名」で変更できます。

- 長さ：2

  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。

(2)　論理マップ生成規則

COBOL

```
{02|03} マップ名-MENUnnnn-A  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_MENUnnnn_A[長さ];
```

16.3.3　プッシュボタンボックス

プッシュボタンボックスの論理マップ生成規則とマッピング規則について説明します。

(1)　プッシュボタンの定義

- プッシュボタンの属性

  プッシュボタンダイアログの「動的変更（AP から表示属性を変更する)」を選ぶと生成されます。

  - ボタン制御項目のデータ名：マップ名-BUTTONnnnn-A

    データ名は，プッシュボタンダイアログの AP が渡す項目の「データ名」に従います。

  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。

- プッシュボタンのラベルテキスト

  プッシュボタンダイアログの「AP からテキストを変更する」を選ぶと生成されます。

  - ラベルテキスト項目のデータ名：マップ名-BUTTONnnnn-O

    データ名は，プッシュボタンダイアログの AP が渡す項目の「データ名」で変更できます。

  - 長さ：ラベルテキストのデータ長

(2)　プッシュボタン属性の論理マップ生成規則とマッピング規則

プッシュボタン属性の論理マップ生成例とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-BUTTONnnnn-A  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_BUTTONnnnn_A[長さ];
```

(b)　マッピング規則

- AP が制御項目に代入した内容と表示結果

  「動的変更（AP から表示属性を変更する)」を指定していない場合は，標準の属性を使って表示します。

AP が制御項目に代入した内容	マッピングオプション	表示結果
セットアップで指定してある修飾名と同じ	マージ	プッシュボタンのボタン属性の修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		標準の属性を使って表示する。
セットアップで指定してある修飾名と同じ	論理マップ	プッシュボタンのボタン属性の修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		(表示形態が「全面書換」のとき) 標準の属性を使って表示する。

APが制御項目に代入した内容	マッピングオプション	表示結果
		(表示形態が「一部上書」で直前に表示したマップと同じ場合) 前回の属性のまま表示する。 (表示形態が「一部上書」で直前に表示したマップと異なる場合) 標準の属性を使って表示する。
−	物理マップ	標準の属性を使って表示する。

(凡例)

−：該当しない。

注

修飾名と修飾名に対応したボタン属性情報は，ドローセットアップの「表示属性の動的変更」の「コマンドコントロール」タブで変更できます。

(3)　プッシュボタンのラベルテキストの論理マップ生成規則とマッピング規則

プッシュボタンのラベルテキストの論理マップ生成規則とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-BUTTONnnnn-0  PIC X(長さ).
```

C言語

```
unsigned char マップ名_BUTTONnnnn_0[長さ];
```

(b)　マッピング規則

- APが論理項目に代入した内容と表示結果

「APからテキストを変更する」を指定していない場合は，テキストに指定したデータを表示します。テキストに文字列を指定していない場合は，空白を表示します。

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	代入されたデータを表示する。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。この場合，テキストに指定したデータを表示する。テキストに文字列を指定していない場合，空白を表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。この場合，データ有無コード以前のデータを表示する。
画面の表示長より長いデータ		代入されたデータを左寄せで桁寄せし，余りを切り捨てて表示する。
−	物理マップ	テキストに指定したデータを表示する。テキストに文字列を指定していない場合，空白を表示する。

(凡例)

−：該当しない。

(4)　プッシュボタンの入力論理マップ生成規則とマッピング規則

プッシュボタンの入力論理マップ生成規則とマッピング規則について説明します。

(a)　入力論理マップ生成規則

COBOL

```
{02|03} マップ名-INCI  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_INCI [長さ];
```

(b)　マッピング規則

- 入力操作と結果（制御項目の内容）

入力操作	結果（制御項目の内容）
ボタンボックスの中からボタンを選ぶ	選んだボタンに対応するイベント通知コードを制御項目に代入する。
ボタンボックスの中からボタンを選ばない（イベント通知用のキーを押した場合）	イベント通知コードを指定したキーによって画面送信をしたときに，対応するイベント通知コードを制御項目に代入する。

16.3.4　ラジオボタンボックス

ラジオボタンボックスの論理マップ生成規則とマッピング規則について説明します。

(1)　ラジオボタンボックスの定義

- ラジオボタンの属性

  固定／可変ラジオボタンボックスダイアログで「動的変更（AP から表示属性を変更する）」を選ぶことで生成されます。

- 固定ラジオボタンの属性
  - 固定ボタン制御項目のデータ名：マップ名-RADIOnnnn-A

    データ名は，固定ラジオボタンボックスダイアログの「制御項目データ名」で変更できます。
  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。
- 可変ラジオボタンの属性
  - 可変ボタン制御項目のデータ名：マップ名-FIELDnnnn-A

    データ名は，可変ラジオボタンボックスダイアログの AP が受け取る項目の「データ名」に従います。
  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。
- 可変ラジオボタン用論理テーブル

- 可変用論理テーブルのデータ名：マップ名-RADIOnnnn-O
- 回数：ボタンの数の指定値
- 可変ボタンラベル項目のデータ名：マップ名-RADIO-LABELnnnn-O
  データ名は，可変ラジオボタンボックスダイアログのラベルの「データ名」で変更できます。
- データ型：ラベルのデータ型で指定
- 長さ：ラベルのデータ長で指定
- 可変ボタンコード項目のデータ名：マップ名-RADIO-CODEnnnn-O
  データ名は，可変ラジオボタンボックスダイアログの通知コードの「データ名」で変更できます。
- 長さ：2
  長さは，可変ラジオボタンボックスダイアログの AP が受け取る項目の「データ長」に従います。

- ラジオボタンボックス
  この項目は可変／固定ラジオボタンボックス共通です。
  - ボックス単位通知項目のデータ名：マップ名-FIELDnnnn-I
    データ名は，ラジオボタンボックスダイアログの AP が受け取る項目の「データ名」で変更できます。
  - 長さ：AP が受け取る項目のデータ長で指定

(2)　ラジオボタンボックス属性の論理マップ生成規則とマッピング規則

ラジオボタンボックス属性の論理マップ生成規則とマッピング規則について説明します。この項目は，ダイアログで「動的変更（AP から表示属性を変更する)」を指定したときだけ論理マップが生成されます。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-RADIOnnnn-A  PIC X(長さ)〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_RADIOnnnn_A〔[回数]〕[長さ];
```

(b)　マッピング規則

- AP が制御項目に代入した内容と表示結果

《「動的変更（AP から表示属性を変更する)」を指定した場合》

AP が制御項目に代入した内容	マッピングオプション	表示結果
セットアップで指定してある修飾名と同じ	マージ	ラジオボタンの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		標準の属性を使って表示する。
セットアップで指定してある修飾名と同じ	論理マップ	ラジオボタンの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		（表示形態が「全面書換」のとき） 標準の属性を使って表示する。

APが制御項目に代入した内容	マッピングオプション	表示結果
		(表示形態が「一部上書」で直前に表示したマップと同じ場合) 前回の属性のまま表示する。 (表示形態が「一部上書」で直前に表示したマップと異なる場合) 標準の属性を使って表示する。
−	物理マップ	標準の属性を使って表示する。

(凡例)

−：該当しない。

注

修飾名と修飾名に対応したボタン属性情報は，ドローセットアップの「表示属性の動的変更」の「候補選択コントロール」タブで変更できます。

《「動的変更（APから表示属性を変更する)」を指定しない場合》

標準の属性を使って表示します。

(3)　ラジオボタンボックス可変用論理テーブルの論理マップ生成規則とマッピング規則

ラジオボタンボックス可変用論理テーブルの論理マップ生成規則とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-RADIOnnnn-O 〔OCCURS 回数〕.
  {03|04} マップ名-RADIO-LABELnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
  {03|04} マップ名-RADIO-CODEnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
```

C言語

```
struct {
 unsigned char マップ名_RADIO_LABELnnnn_O[長さ];
 unsigned char マップ名_RADIO_CODEnnnn_O [長さ];
} マップ名-RADIOnnnn_O〔[回数]〕;
```

(b)　マッピング規則

- APが論理項目に代入した内容と表示結果

《ラベル項目を指定した場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ， 論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$		ヌルクリアして表示する。
すべて$(20)_{16}$		空白で表示する。
先頭1文字がデータ有無コード※		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。 (コード項目のデータがある) コード項目のデータを表示する。

APが論理項目に代入した内容	マッピングオプション	表示結果
		(コード項目のデータがない) ボタン図形だけでラベルは表示しない。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例として X'4142431F1F'のようなデータが該当する。 ラベル項目の桁寄せ向きに従って桁寄せし，埋字指定の内容で埋字をして表示する。
画面の表示長より長いデータ		ラベル項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	何も表示しない。

(凡例)

－：該当しない。

注※

ラベル項目のデータ型とコード項目のデータ型が同じ場合だけ，ラベル項目にデータ有無コードを設定できます。

(4)　ラジオボタンの入力論理マップ生成規則とマッピング規則

ラジオボタンの入力論理マップ生成例とマッピング規則について説明します。

(a)　入力論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H.
  04 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

C言語

```
unsigned char マップ名_FIELDnnnn_I[長さ];
```

(b)　マッピング規則

- 入力操作と結果（論理項目の内容）

入力操作	結果（論理項目の内容）
ボタンボックス中からボタンを選ぶ	選んだボタンに対応する定数を論理項目に設定する。
ボタンボックス中からボタンを選ばない	(固定ラジオボタンの場合) ボタン属性情報で「選択済みにする」を指定したボタンに対応する定数を論理項目に設定する。複数のボタン属性情報に「選択済みにする」を指定した場合，最初に指定したラジオボタンに対応する定数を論理項目に設定する。 「選択済みにする」を指定しなかった場合，初期クリア文字に従って論理項目をクリアする。 (可変ラジオボタンの場合) 初期クリア文字に従って論理項目をクリアする。

16.3.5　チェックボタンボックス

チェックボタンボックスのマッピング規則は，基本的にラジオボタンボックスと同じです。マッピング規則については，「16.3.4　ラジオボタンボックス」を参照してください。

ここでは，チェックボタンの定義，論理マップ生成規則，およびラジオボタンと異なるマッピング規則について説明します。

(1)　チェックボタンボックスの定義

- チェックボタンの属性

  固定／可変チェックボタンダイアログで「動的変更（AP から表示属性を変更する）」を選ぶことで生成されます。

  - ボタン制御項目のデータ名：マップ名-FIELDnnnn-A

    データ名は，チェックボタンダイアログの AP が受け取る項目の「データ名」に従います。

  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。

- 可変チェックボタン用論理テーブル

  - 可変用論理テーブルのデータ名：マップ名-CHECKnnnn-O
  - 回数：ボタンの数の指定値
  - 可変ボタンラベル項目のデータ名：マップ名-CHECK-LABELnnnn-O

    データ名は，可変チェックボタンボックスダイアログのラベルの「データ名」で変更できます。

  - データ型：ラベルのデータ型で指定
  - 長さ：ラベルのデータ長で指定
  - 可変ボタンコード項目のデータ名：マップ名-CHECK-CODEnnnn-O

    データ名は，可変チェックボタンボックスダイアログの通知コードの「データ名」で変更できます。

  - 長さ：2

    長さは，可変チェックボタンボックスダイアログの AP が受け取る項目の「データ長」に従います。

- チェックボタンボックス

  この項目は可変／固定チェックボタンボックス共通です。

  - ボックス単位通知項目のデータ名：マップ名-FIELDnnnn-I

    データ名は，チェックボタンダイアログの AP が受け取る項目の「データ名」で変更できます。

  - 長さ：AP が受け取る項目のデータ長で指定

(2)　チェックボタンボックス属性の論理マップ生成規則とマッピング規則

チェックボタンボックス属性の論理マップ生成規則とマッピング規則について説明します。この項目は，ダイアログで「動的変更（AP から表示属性を変更する）」を指定したときだけ論理マップが生成されます。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-A  PIC X(長さ)〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_A〔[回数]〕[長さ] ;
```

(b)　マッピング規則

- AP が制御項目に代入した内容と表示結果

  ラジオボタンボックスと同様です。詳細については,「16.3.4(2)(b)　マッピング規則」を参照してください。

(3)　チェックボタンボックス可変用論理テーブルの論理マップ生成規則とマッピング規則

チェックボタンボックス可変用論理テーブルの論理マップ生成規則とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-CHECKnnnn-O 〔OCCURS 回数〕.
  {03|04} マップ名-CHECK-LABELnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
  {03|04} マップ名-CHECK-CODEnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
```

C 言語

```
struct {
  unsigned char マップ名_CHECK_LABELnnnn_O[長さ];
  unsigned char マップ名_CHECK_CODEnnnn_O[長さ];
} マップ名-CHECKnnnn_O〔[回数]〕;
```

(b)　マッピング規則

- AP が論理項目に代入した内容と表示結果

  ラジオボタンボックスと同様です。詳細については,「16.3.4(3)(b)　マッピング規則」を参照してください。

(4)　チェックボタンの入力論理マップ生成規則とマッピング規則

チェックボタンの入力論理マップ生成例とマッピング規則について説明します。

(a)　入力論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H  〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_I〔[回数]〕[長さ];
```

(b)　マッピング規則

- 入力操作と結果（論理項目の内容）

入力操作	結果（論理項目の内容）
ボタンボックス中からボタンを選ぶ	選んだボタンに対応する定数を論理項目に代入する。複数のボタンを選んだ場合，選んだすべてのボタンに対応する定数を論理項目に代入する。

入力操作	結果（論理項目の内容）
ボタンボックス中からボタンを選ばない，またはボタンボックスにフォーカスを位置づけない	（固定チェックボタンの場合） ボタン属性情報で「フォーカス設定（フォーカスを本ボタンに位置づける）」を指定したボタンに対応する定数を論理項目に代入する。複数のボタン属性情報に初期選択を指定した場合，選んだすべてのボタンに対応する定数を論理項目に代入する。 ボタン属性情報を指定しなかった場合，初期値で指定した定数を論理項目に代入する。 初期値を指定しなかった場合，初期クリア文字に従って論理項目をクリアする。 （可変ラジオボタンの場合） 初期クリア文字に従って論理項目をクリアする。

16.3.6　リストボックス

リストボックスの論理マップ生成規則とマッピング規則について説明します。

(1)　リストボックスの定義

- **リストボックスの属性**

  リストボックスダイアログで「動的変更（AP から表示属性を変更する）」を選ぶことで生成されます。

  - リスト制御項目のデータ名：マップ名-FIELDnnnn-A

    データ名は，リストボックスダイアログの AP が受け取る項目の「データ名」に従います。
  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。
- **論理テーブル**
  - 可変用論理テーブルのデータ名：マップ名-LISTnnnn-O
  - 回数：指定したリスト項目数
  - 可変リストラベル項目のデータ名：マップ名-LIST-LABELnnnn-O

    データ名は，リストボックスダイアログのラベルの「データ名」で変更できます。
  - データ型：ラベルのデータ型で指定
  - 長さ：ラベルのデータ長で指定
  - 可変リストコード項目のデータ名：マップ名-LIST-CODEnnnn-O

    データ名は，リストボックスダイアログの通知コードの「データ名」で変更できます。
  - 長さ：2

    長さは，リストボックスダイアログの AP が受け取る項目の「データ長」に従います。
- **リストボックス**
  - ボックス単位通知項目のデータ名：マップ名-FIELDnnnn-I

    データ名は，リストボックスダイアログの AP が受け取る項目の「データ名」で変更できます。
  - 長さ：AP が受け取る項目のデータ長で指定

(2)　リストボックスのボックス属性の論理マップ生成規則とマッピング規則

リストボックスのボックス属性の論理マップ生成規則とマッピング規則について説明します。この項目は，ダイアログで「動的変更（AP から表示属性を変更する)」を指定したときだけ論理マップが生成されます。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-LISTnnnn-A  PIC X(長さ)〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_LISTnnnn_A〔[回数]〕[長さ];
```

(b)　マッピング規則

- AP が制御項目に代入した内容と表示結果

《「動的変更（AP から表示属性を変更する)」を指定した場合》

AP が制御項目に代入した内容	マッピングオプション	表示結果
セットアップで指定してある修飾名と同じ	マージ	リストボックスの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		標準の属性を使って表示する。
セットアップで指定してある修飾名と同じ	論理マップ	リストボックスの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		(表示形態が「全面書換」のとき) 標準の属性を使って表示する。 (表示形態が「一部上書」で直前に表示したマップと同じ場合) 前回の属性のまま表示する。 (表示形態が「一部上書」で直前に表示したマップと異なる場合) 標準の属性を使って表示する。
－	物理マップ	標準の属性を使って表示する。

(凡例)

－：該当しない。

注

修飾名と修飾名に対応したリスト属性情報は，ドローセットアップの「表示属性の動的変更」の「候補選択コントロール」タブで変更できます。

《「動的変更（AP から表示属性を変更する)」を指定しない場合》

標準の属性を使って表示します。

(3)　リストボックスの論理テーブルの論理マップ生成規則とマッピング規則

リストボックスの論理テーブルの論理マップ生成規則とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-LISTnnnn-O 〔OCCURS 回数〕.
  {03|04} マップ名-LIST-LABELnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
  {03|04} マップ名-LIST-CODEnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
```

C 言語

```
struct {
  unsigned char マップ名_LIST_LABELnnnn_O[長さ];
  unsigned char マップ名_LIST_CODEnnnn_O[長さ];
} マップ名-LISTnnnn_O〔[回数]〕;
```

(b)　マッピング規則

- AP が論理項目に代入した内容と表示結果

《ラベル項目を指定した場合》

AP が論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$※1		ヌルクリアして表示する。
すべて$(20)_{16}$※1		空白で表示する。
先頭 1 文字がデータ有無コード※2		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。 (コード項目の指定がある) 　コード項目のデータを表示する。 (コード項目の指定がない) 　何も表示しない。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。テキスト項目の桁寄せ向きに従って桁寄せし，埋字指定の内容で埋字をして表示する。
画面の表示長より長いデータ		テキスト項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	何も表示しない。

(凡例)

－：該当しない。

注※1

リストの後半に該当する項目がある場合，項目は圧縮されます。

注※2

ラベル項目のデータ型とコード項目のデータ型が同じ場合だけ，ラベル項目にデータ有無コードを設定できます。

(4)　リストボックスの入力論理マップ生成規則とマッピング規則

リストボックスの入力論理マップ生成例とマッピング規則について説明します。

(a)　入力論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H　〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I　PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I　PIC {X(長さ)|N(長さ÷2)|9(長さ)}　〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_I〔[回数]〕[長さ];
```

(b)　マッピング規則

- 入力操作と結果（論理項目の内容）

入力操作	結果（論理項目の内容）
リストボックス中からリストを選ぶ	選んだリストに対応する定数を論理項目に代入する。単一選択リストボックスは最後に選択したリストを，複数選択リストボックスで複数のリストを選んだ場合，選択したすべてのリストに対応する定数を論理項目に代入する。
リストボックスの中からリストを選ばない，またはリストボックスにフォーカスを位置づけない	リスト属性情報で初期選択を指定したリストに対応する定数を論理項目に代入する。ただし，単一選択リストボックスは最初に選択したリストを，複数選択リストボックスで複数のリスト属性情報に初期選択を指定した場合は指定したすべてのリストに対応する定数を論理項目に代入する。 リスト属性を指定しなかった場合，論理項目の初期値で指定した定数を論理項目に代入する。 論理項目の初期値を指定しなかった場合，ドローセットアップの論理マップ属性で指定した初期クリア文字に従って論理項目を初期化する。

16.3.7　コンボボックス

コンボボックスの論理マップ生成規則とマッピング規則について説明します。

(1)　コンボボックスの定義

コンボボックスは，選択されたデータや入力されたデータを表示するボックスと，入力するデータの候補を表示するメニューから成ります。

- コンボボックスの属性

  コンボボックスダイアログで「動的変更（AP から表示属性を変更する)」を選び，属性項目データ名を指定することで生成されます。

  - 制御項目のデータ名：マップ名-FIELDnnnn-A

    データ名は，コンボボックスダイアログの AP が受け取る通知コードの項目の「データ名」に従います。

  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「キャラクタコントロール（キーエントリ／選択エントリ)」を選択し，「修飾名長」で長さを設定します。

- ボックスの定義

固定コンボボックス，および可変コンボボックスに対して生成されます。

- ボックス項目のデータ名：マップ名-FIELDnnnn-I/O
  データ名は，コンボボックスダイアログの AP が受け取る通知コードの項目の「データ名」で変更できます。
- 長さ：AP が受け取る項目のデータ長で指定

- **可変コンボボックス用論理テーブルの定義**
  可変メニューダイアログで指定した内容で生成されます。
  - 可変用論理テーブルのデータ名：マップ名-POPUPnnnn-O
  - 回数：メニューの項目数の指定値
  - 可変メニューラベル項目のデータ名：マップ名-POPUP-LABELnnnn-O
    データ名は，可変コンボメニューダイアログのラベルの「データ名」で変更できます。
  - データ型：ラベルのデータ型で指定
  - 長さ：コンボボックスの長さの指定値-2
  - 可変メニューコード項目のデータ名：マップ名-POPUP-CODEnnnn-O
    データ名は，可変コンボメニューダイアログの AP に返す通知コードの「データ名」で変更できます。
  - 長さ：2
    長さは，コンボボックスダイアログの AP が受け取る通知コードの項目の「データ長」に従います。

(2)　コンボボックスの論理マップ生成規則とマッピング規則

(a)　項目属性の論理マップ生成規則とマッピング規則

コンボボックスの項目属性に関する論理マップ生成規則とマッピング規則について説明します。この論理マップは，コンボボックスダイアログで「動的変更（AP から表示属性を変更する)」を選び，属性項目データ名を指定することで生成されます。

- **論理マップ生成規則**

  COBOL

  ```
  {02|03} マップ名-FIELDnnnn-A  PIC X(長さ).
  ```

  C 言語

  ```
  unsigned char  マップ名_FIELDnnnn_A[長さ];
  ```

- **マッピング規則**
  - **AP が制御項目に代入した内容と表示結果**

  《「動的変更（AP から表示属性を変更する)」を指定した場合》

AP が制御項目に代入した内容	マッピングオプション	表示結果
修飾名と同じ	マージ	コンボボックスの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または 修飾名以外（上記以外）		標準の属性を使って表示する。
修飾名と同じ	論理マップ	コンボボックスの修飾名に対応する属性を使って表示する。

APが制御項目に代入した内容	マッピングオプション	表示結果
先頭にデータ有無コード，または 修飾名以外（上記以外）		（表示形態が「全面書換」の場合） 　標準の属性を使って表示する。 （表示形態が「一部上書」で直前に表示したマップと同じ場合） 　前回の属性をそのまま表示する。 （表示形態が「一部上書」で直前に表示したマップと異なる場合） 　標準の属性を使って表示する。
－	物理マップ	標準の属性を使って表示する。

（凡例）

－：該当しない。

注

修飾名と修飾名に対応したテキスト・フィールド変更属性情報は，ドローセットアップの「表示属性の動的変更」の「キャラクタコントロール」タブで変更できます。

《「動的変更（APから表示属性を変更する）」を指定しない場合》

標準の属性を使って表示します。

(b)　出力論理マップ生成規則とマッピング規則

コンボボックスの出力論理マップ生成規則とマッピング規則について説明します。

- 出力論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-O  PIC X(長さ).
```

C言語

```
unsigned char マップ名_FIELDnnnn_O[長さ];
```

- マッピング規則
  - APが論理項目に代入した内容と表示結果

《入力形式がキー入力可能で，初期値を指定しない場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ， 論理マップ	（コード項目に同一のデータが指定されている場合） 　対応するラベル項目のデータを表示する。 （コード項目に同一のデータが指定されていない場合） 　代入されたデータを表示する。
すべて$(00)_{16}$のデータ		ヌルクリアして表示する。
すべて$(20)_{16}$のデータ		空白で表示する。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例として'1F414234'またはX'1F1F1F'のようなデータが該当する。 （表示形態が「全面書換」の場合） 　何も表示しない。

APが論理項目に代入した内容	マッピングオプション	表示結果
		（表示形態が「一部上書」で前回と同じマップの場合） 直前のデータのまま表示する。 （表示形態が「一部上書」で前回と異なるマップの場合） 何も表示しない。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。 （データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されている場合） 対応するラベル項目のデータを表示する。 （データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されていない場合） 論理項目の桁寄せ向きに従って桁寄せする。また，埋字指定の内容に従って埋字をする。
画面の表示長より長いデータ		（コード項目に同一のデータが指定されている場合） 対応するラベル項目のデータを表示する。 （コード項目に同一のデータが指定されていない場合） 論理項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	何も表示しない。

（凡例）

－：該当しない。

《入力形式がキー入力可能で，初期値を指定する場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
先頭1文字がデータ有無コード	マージ，論理マップ	《先頭1文字がデータ有無コードのときの結果》を参照のこと。
すべてデータ		代入されたデータを表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。論理項目の向きと，指定した埋字の内容に従って表示する。
－	物理マップ	初期値を指定しない場合と同じになる。

（凡例）

－：該当しない。

《先頭1文字がデータ有無コードのときの結果》

初期値の内容	結果
すべて空白（$(20)_{16}$）	項目全体を空白で表示する。
すべて0（$(30)_{16}$）	項目全体を0で表示する。
すべてヌル（$(00)_{16}$）	項目全体をヌルクリアして表示する。

初期値の内容	結果
繰り返し文字指定	項目全体を繰り返し文字で表示する。
初期値の長さと画面の表示長が等しい	初期値を項目に表示する。
初期値の長さが画面の表示長より短い	定義で指定した「桁寄せ」，および埋字に従って表示する。埋字なしを指定した場合は，左寄せで初期値を項目に表示する。
初期値の長さが画面の表示長より長い	「桁寄せ」に従って桁寄せし，余りを切り捨てて項目を表示する。

《入力形式がメニュー選択だけで，初期値を指定しない場合》

AP が論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	(コード項目に同一のデータが指定されている場合) 対応するラベル項目のデータを表示する。 (コード項目に同一のデータが指定されていない場合) 先頭のラベル項目のデータを表示する。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'または X'1F1F1F'のようなデータが該当する。 (表示形態が「全面書換」の場合) 先頭のラベル項目のデータを表示する。 (表示形態が「一部上書」で前回と同じマップの場合) 直前のデータのまま表示する。 (表示形態が「一部上書」で前回と異なるマップの場合) 先頭のラベル項目のデータを表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。 (データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されている場合) 対応するラベル項目のデータを表示する。 (データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されていない場合) 先頭のラベル項目のデータを表示する。
画面の表示長より長いデータ		(コード項目に同一のデータが指定されている場合) 対応するラベル項目のデータを表示する。 (コード項目に同一のデータが指定されていない場合) 先頭のラベル項目のデータを表示する。
－	物理マップ	先頭のラベル項目のデータを表示する。

(凡例)

－：該当しない。

《入力形式がメニュー選択だけで，初期値を指定する場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
先頭1文字がデータ有無コード	マージ，論理マップ	《先頭1文字がデータ有無コードのときの結果》を参照のこと。
すべてデータ		(コード項目に同一のデータが指定されている場合) 対応するラベル項目のデータを表示する。 (コード項目に同一のデータが指定されていない場合) 先頭のラベル項目のデータを表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。 (データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されている場合) 対応するラベル項目のデータを表示する。 (データ有無コードを切り捨てたデータとメニューのコード項目に同一のデータが指定されていない場合) 先頭のラベル項目のデータを表示する。
－	物理マップ	《先頭1文字がデータ有無コードのときの結果》を参照のこと。

(凡例)

－：該当しない。

《先頭1文字がデータ有無コードのときの結果》

初期値の内容	結果
メニューのコード項目に指定されているデータと同じ	対応するラベル項目のデータを表示する。
メニューのコード項目に指定されているデータと異なる	先頭のラベル項目のデータを表示する。

(c)　入力論理マップ生成規則とマッピング規則

コンボボックスの入力論理マップ生成規則とマッピング規則について説明します。

- 論理マップ生成規則

  COBOL

  ・論理マップ可変部の集団項目化を指定した場合

  ```
  03 マップ名-FIELDnnnn-H.
    04 マップ名-FIELDnnnn-I  PIC  {X(長さ)|N(長さ÷2)|9(長さ)}.
  ```

  ・論理マップ可変部の集団項目化を指定しない場合

  ```
  02 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
  ```

  C言語

  ```
  unsigned char マップ名_FIELDnnnn_I[長さ];
  ```

- マッピング規則
  - 入力操作と結果（論理項目の内容）

《入力形式がキー入力可能の場合》

入力操作	結果（論理項目の内容）
コンボボックスにカーソルを位置づけ，表示された候補の中からメニューを選択※1 する	選んだメニュー（フォーカスを位置づけたラベル）に対応したコードを論理項目に代入する。
コンボボックスにカーソルを位置づけ表示された候補からメニューを選択※1 しない，またはコンボボックスにカーソルを位置づけない	論理項目の初期値で指定したコードを論理項目に代入する。論理項目の初期値が指定されていない場合，初期クリア文字の指定に従って論理項目をクリアする。
半角のデータを入力※2	データを論理項目に代入する。
全角，または半角／全角混在のデータを入力※2	
フィールドキーを押す（すべて$(00)_{16}$）データ入力※2	データ消去通知文字に従って論理項目をクリアする。

注※1

選択とは，カーソルでメニューを選んで，復改キーを押す操作です。この操作で，コードが論理項目に代入されます。

注※2

コンボボックスにカーソルを位置づけてデータを入力する場合が該当します。

《入力形式がメニュー選択だけの場合》

入力操作	結果（論理項目の内容）
コンボボックスにカーソルを位置づけ，表示された候補の中からメニューを選択※する	選んだメニュー（フォーカスを位置づけたラベル）に対応したコードを論理項目に代入する。
コンボボックスにカーソルを位置づけ表示された候補からメニューを選択※しない，またはコンボボックスにカーソルを位置づけない	論理項目の初期値で指定したコードを論理項目に代入する。論理項目の初期値が指定されていない場合，初期クリア文字の指定に従って論理項目をクリアする。

注※

選択とは，カーソルでメニューを選んで，スペースキーまたは復改キーを押す操作です。この操作で，コードが論理項目に代入されます。

(3)　コンボボックスのメニューの論理マップ生成規則とマッピング規則

(a)　メニュー属性のマッピング規則

メニュー属性のマッピング規則は，標準の属性を使ってメニューを表示します。

(b)　ラベルの論理マップ生成規則とマッピング規則

可変用論理テーブル名を指定した場合のコンボボックスの論理マップ生成規則とマッピング規則について説明します。

- 論理マップ生成例（入力形式がメニュー選択だけの場合）

COBOL

```
03 マップ名-POPUPnnnn-O  OCCURS 回数.
  04 マップ名-POPUP-LABELnnnn-O  PIC {9(長さ)|X(長さ)|N(長さ÷2)|編集文字}.
```

```
04マップ名-POPUP-CODEnnnn-O  PIC X(長さ).
```

C言語

```
struct {
  unsigned char マップ名_POPUP_LABELnnnn_O[長さ];
  unsigned char マップ名_POPUP_CODEnnnn_O[長さ];
} マップ名_POPUPnnnn_O[回数];
```

- 論理マップ生成例（入力形式がキー入力可能の場合）

COBOL

```
03 マップ名-POPUPnnnn-O  OCCURS 回数.
  04 マップ名-POPUP-CODEnnnn-O  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

C言語

```
struct {
  unsigned char マップ名_POPUP_CODEnnnn_O[長さ];
} マップ名_POPUPnnnn_O[回数];
```

- マッピング規則
  - APが論理項目に代入した内容と表示結果

《入力形式がメニュー選択だけの場合》

APが論理項目に代入した内容	マッピングオプション	表示結果
すべてデータ	マージ，論理マップ	代入されたデータを表示する。
すべて$(00)_{16}$※1		ヌルクリアして表示する。
すべて$(20)_{16}$※1		空白で表示する。
先頭1文字がデータ有無コード※2		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。 (コード項目の指定がある場合) コード項目のデータを表示する。 (コード項目の指定がない場合) 何も表示しない。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する。 指定したラベル項目の向き，および埋字の内容に従って表示する。
画面の表示長より長いデータ		ラベル項目の桁寄せ向きに従って桁寄せし，余りを切り捨てて表示する。
－	物理マップ	何も表示しない。

(凡例)

－：該当しない。

注※1

リストの後半に該当する項目がある場合，項目は圧縮されます。

注※2

ラベル項目のデータ型とコード項目のデータ型が同じ場合だけ，ラベル項目にデータ有無コードを設定できます。

《入力形式がキー入力可能の場合》

マッピングオプション	表示結果
マージ，論理マップ	コード項目のデータを表示する。
物理マップ	何も表示しない。

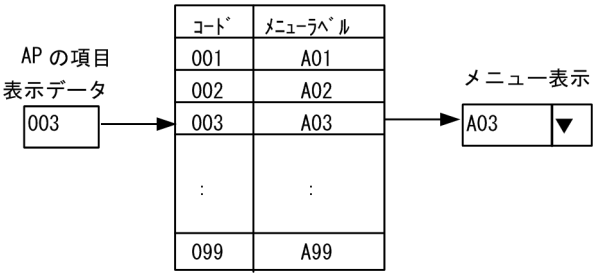
(c)　コンボボックスのメニューの表示

コンボボックスのメニューの表示に関するマッピング規則について説明します。項目の表示データに対応するコンボボックスのメニューコードの値によって表示結果が異なります。

- 入力形式がメニュー選択だけの場合

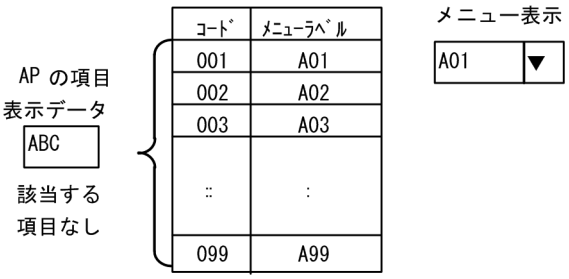
《メニューコード内に項目表示データの値があり，項目長とメニューコード長が等しい》

先頭のメニューから表示し，項目表示データと同じメニューコードに対応するメニューに，カーソルを位置づけます。



《メニューコード内に項目表示データの値がないか，項目長とメニューコード長が等しくない》

先頭メニューを表示し，カーソルを位置づけます。



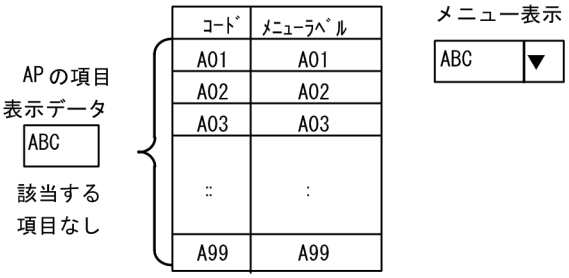
- 入力形式がキー入力可能の場合

《メニューコード内にキー入力したデータの値があり，項目長とメニューコード長が等しい》

入力形式がメニュー選択だけの場合と同じです。キー入力の場合，「AP の項目表示データ」は「キー入力したデータ」となります。

《メニューコード内にキー入力したデータの値がないか，項目長とメニューコード長が等しくない》

キー入力したデータを表示します。



16.3.8　フォーカス・カーソル位置

フォーカス・カーソル位置の論理マップ生成規則について説明します。

(1)　フォーカス・カーソル項目の定義

(a)　フォーカス項目の定義

- 出力フォーカス項目の定義
  - 出力フォーカス項目：マップ名-OUTFOCUS-O
  - 長さ：2

データ名は，ドローセットアップの「カーソルとフォーカス」ダイアログで変更できます。

- 入力フォーカス項目の定義
  - 入力フォーカス項目：マップ名-INFOCUS-I
  - 長さ：2

データ名は，ドローセットアップの「カーソルとフォーカス」ダイアログで変更できます。

(b)　カーソル項目の定義

フィールドボックスを複数定義した場合，それぞれに割り当てるために，カーソル項目を複数定義できます。ここでは，2 個目以降の定義について示します。

1 個目の定義については，「16.2.2(1)　出力カーソル項目の定義」，「16.2.3(1)　入力カーソル項目の定義」を参照してください。

- 行列（2 進）カーソルの標準値
  - 出力カーソル項目データ名：（行）マップ名-OUTCURSnL，（列）マップ名-OUTCURSnC
  - 入力カーソル項目データ名：（行）マップ名-INCURSnN，（列）マップ名-INCURSnM
  - 長さ：4

データ名は，ドローセットアップの「カーソルとフォーカス」ダイアログで変更できます。

- 行列（10 進）カーソルの標準値
  - 出力カーソル項目データ名：（行）マップ名-OUTCURSnY，（列）マップ名-OUTCURSnX
  - 入力カーソル項目データ名：（行）マップ名-INCURSnY，（列）マップ名-INCURSnX
  - 長さ：8

データ名は，ドローセットアップの「カーソルとフォーカス」ダイアログで変更できます。

- 論理カーソルの標準値
  - 出力カーソル項目データ名：マップ名-OUTCURSnO
  - 入力カーソル項目データ名：マップ名-INCURSnI
  - 長さ：2

データ名は，ドローセットアップの「カーソルとフォーカス」ダイアログで変更できます。

(2)　出力フォーカス・カーソル位置の論理マップ生成規則とマッピング規則

(a)　フォーカスとカーソルを同時に制御する場合

論理マップにはカーソル制御項目だけが生成されます。フォーカス制御項目は生成されません。

- **論理マップ生成例**
  「16.2.2(2)　論理マップ生成規則」を参照してください。
- **マッピング規則**
  - **AP がカーソル項目データ名の領域に代入した内容と結果**

《フィールドボックスの定義がある場合》

AP がカーソル項目データ名の領域に代入した内容	結果
$(00)_{16}$ クリア，またはマップ生成機能が生成するフォーカス・カーソル定数以外の値	《初期フォーカス，初期カーソルの指定がある》 • フォーカスは，初期フォーカスを指定したオブジェクトに位置づく。 • カーソルは，初期カーソルを指定したフィールドの先頭に位置づく。 《初期フォーカスの指定があり，初期カーソルの指定がない》 • フォーカスは，初期フォーカスを指定したオブジェクトに位置づく。 • カーソルは，フィールドボックス内の先頭の行列位置にある入力できるフィールドに位置づく。カーソルはフィールドの先頭に位置づく。 《初期フォーカスの指定がなく，初期カーソルの指定がある》 • フォーカスは，ウィンドウ上の先頭の行列位置にあるオブジェクトに位置づく。 • カーソルは，初期カーソルを指定したフィールドの先頭に位置づく。 《初期フォーカス，初期カーソルの指定がない》 • フォーカスは，ウィンドウ上の先頭の行列位置にあるオブジェクトに位置づく。 • カーソルは，フィールドボックス内の先頭の行列位置にある入力できるフィールドの先頭に位置づく。
マップ生成機能が生成するフィールドボックス内のフィールドのフォーカス・カーソル定数	フォーカスはフィールドボックスに位置づく。カーソルはフォーカス・カーソル定数に対応したフィールドの先頭に位置づく。 フィールドボックスを複数定義している場合，フォーカスの位置づかないフィールドボックスのカーソルは，初期カーソルを指定したフィールドの先頭に位置づく。初期カーソルの指定がない場合，フィールドボックス内の先頭の行列位置にある入力できるフィールドの先頭に位置づく。
マップ生成機能が生成するフィールドボックス外のオブジェクトのフォーカス・カーソル定数	フォーカスはフォーカス・カーソル定数に対応したオブジェクトに位置づく。カーソルは初期カーソルの指定に従う。

APがカーソル項目データ名の領域に代入した内容	結果
	フィールドボックスを複数定義している場合，フォーカスの位置づかないフィールドボックスのカーソルは，初期カーソルの指定に従う。 《初期カーソルの指定がある》 初期カーソルを指定したフィールドの先頭にカーソルが位置づく。 《初期カーソルの指定がない》 フィールドボックス内の先頭の行列位置にある入力できるフィールドの先頭にカーソルが位置づく。

《フィールドボックスの定義がない場合》

APがカーソル項目データ名の領域に代入した内容	結果
$(00)_{16}$ クリア，または マップ生成機能が生成するフォーカス・カーソル定数以外の値	《初期フォーカスの指定がある》 初期フォーカスを指定したオブジェクトにフォーカスが位置づく。 《初期フォーカスの指定がない》 ウィンドウ上の先頭の行列位置にあるオブジェクトにフォーカスが位置づく。
マップ生成機能が生成するフォーカス・カーソル定数	フォーカス・カーソル定数に対応したオブジェクトにフォーカスが位置づく。

(b)　フォーカスとカーソルを別々に制御する場合

論理マップには，フォーカス制御項目とカーソル制御項目が生成されます。フォーカス制御の論理マップ生成例とマッピング規則について説明します。カーソル制御のマッピング規則については「16.2.2(3)　マッピング規則」を参照してください。フォーカス定数については「16.5.3　フォーカス定数の論理マップ生成規則」を参照してください。

・論理マップ生成規則

COBOL

・行列（2進）カーソルを指定した場合

Windowsリトルエンディアン用

```
{02|03} マップ名-OUTCURSL      PIC S9(4) COMP-5.
{02|03} マップ名-OUTCURSC      PIC S9(4) COMP-5.
{02|03} マップ名-OUTCURSnL     PIC S9(4) COMP-5.        ……※
{02|03} マップ名-OUTCURSnC     PIC S9(4) COMP-5.        ……※
{02|03} マップ名-OUTFOCUS-O    PIC X(2).
```

Windowsビッグエンディアン用，UNIX用

```
{02|03} マップ名-OUTCURSL      PIC S9(4) COMP.
{02|03} マップ名-OUTCURSC      PIC S9(4) COMP.
{02|03} マップ名-OUTCURSnL     PIC S9(4) COMP.          ……※
{02|03} マップ名-OUTCURSnC     PIC S9(4) COMP.          ……※
{02|03} マップ名-OUTFOCUS-O    PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

・行列（10進）カーソルを指定した場合

```
{02|03} マップ名-OUTCURSY      PIC 9(4).
{02|03} マップ名-OUTCURSX      PIC 9(4).
{02|03} マップ名-OUTCURSnY     PIC 9(4).              ……※
```

```
{02|03} マップ名-OUTCURSnX     PIC 9(4).              ……※
{02|03} マップ名-OUTFOCUS-O    PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

・論理カーソルを指定した場合

```
{02|03} マップ名-OUTCURS-LOCO  PIC X(2).
{02|03} マップ名-OUTCURSnO     PIC X(2).              ……※
{02|03} マップ名-OUTFOCUS-O    PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

C 言語

・行列（2 進）カーソルを指定した場合

```
unsigned char マップ名_OUTCURSL[2];
unsigned char マップ名_OUTCURSC[2];
unsigned char マップ名_OUTCURSnL[2];              ……※
unsigned char マップ名_OUTCURSnC[2];              ……※
unsigned char マップ名_OUTFOCUS_0[2];
```

注※

フィールドボックスを複数定義している場合に展開されます。

・行列（10 進）カーソルを指定した場合

行列（2 進）カーソルに変換されます。

・論理カーソルを指定した場合

```
unsigned char マップ名_OUTCURS_LOCO[2];
unsigned char マップ名_OUTCURSnO[2];              ……※
unsigned char マップ名_OUTFOCUS_O[2];
```

注※

フィールドボックスを複数定義している場合に展開されます。

・マッピング規則

- AP がフォーカス項目データ名の領域に代入した内容と結果

AP がフォーカス項目データ名の領域に代入した内容	結果
$(00)_{16}$ クリア，またはマップ生成機能が生成するフォーカス定数以外の値	《初期フォーカスの指定がある》 初期フォーカスを指定したオブジェクトにフォーカスが位置づく。 《初期フォーカスの指定がない》 ウィンドウ上の先頭の行列位置にあるオブジェクトにフォーカスが位置づく。
マップ生成機能が生成するフォーカス定数	フォーカス定数に対応したオブジェクトにフォーカスが位置づく。

(3)　入力フォーカス・カーソル位置の論理マップ生成規則とマッピング規則

(a)　フォーカスとカーソルを同時に制御する場合

論理マップにはカーソル制御項目だけが生成されます。フォーカス制御項目は生成されません。フィールドボックスを定義した画面で，複数のプッシュボタンボックスのフォーカス位置を識別して制御したい場合には，フォーカスとカーソルを別々に制御してください。

- 論理マップ生成規則

「16.2.3(2)(c)　論理カーソルを指定した場合」を参照してください。

- マッピング規則
  - AP がカーソル項目データ名の領域に代入される内容

《フィールドボックスの定義がある場合》

入力操作	結果
画面送信時のフォーカス位置がプッシュボタンに位置づいている (ほかのオブジェクトにフォーカスが位置づいている場合で, マウスによるプッシュボタン選択は該当しない)	通知コードデータ名 T に対応するフォーカス・カーソル定数を代入する。
画面送信時のフォーカス位置がフィールドボックスに位置づいていて, カーソル位置が入力・入出力フィールド内に収まっていない	$(00)_{16}$ を代入する。
画面送信時のフォーカス位置がフィールドボックスに位置づいていて, カーソル位置が入力・入出力フィールド内に収まっている	該当するフィールドに対応するカーソル定数を代入する。
画面送信時のフォーカス位置がプッシュボタン, フィールドボックス以外のオブジェクトに位置づいている	該当するオブジェクトに対応するフォーカス・カーソル定数を代入する。

注

GUI 画面で入力単位が「フィールド単位」または「イベント単位」の場合，キー操作をしたオブジェクトのフォーカス定数，またはカーソル定数を代入します。

《フィールドボックスの定義がない場合》

入力操作	結果
画面送信時のフォーカス位置がプッシュボタンに位置づいている (ほかのオブジェクトにフォーカスが位置づいている場合で, マウスによるプッシュボタン選択は該当しない)	《プッシュボタンにボックス名称の指定がある》 ボックス名-T に対応するフォーカス・カーソル定数を代入する。 《プッシュボタンにボックス名称の指定がない》 通知コードデータ名 T に対応するフォーカス・カーソル定数を代入する。
画面送信時のフォーカス位置がプッシュボタン以外のオブジェクトに位置づいている	該当するオブジェクトに対応するフォーカス・カーソル定数を代入する。

注

入力単位が「フィールド単位」または「イベント単位」の場合，キー操作をしたオブジェクトの定数を代入します。

(b)　フォーカスとカーソルを別々に制御する場合

論理マップにはフォーカス制御項目とカーソル制御項目が生成されます。フォーカス制御の論理マップ生成例とマッピング規則について説明します。カーソル制御のマッピング規則については「16.2.3(3)　マッピング規則」を参照してください。フォーカス定数については「16.5.3　フォーカス定数の論理マップ生成規則」を参照してください。

・論理マップ生成規則

COBOL

・行列（2 進）カーソルを指定した場合

Windows リトルエンディアン用

```
{02|03} マップ名-INCURSN      PIC S9(4) COMP-5.
{02|03} マップ名-INCURSM      PIC S9(4) COMP-5.
{02|03} マップ名-INCURSnN     PIC S9(4) COMP-5.        ……※
{02|03} マップ名-INCURSnM     PIC S9(4) COMP-5.        ……※
{02|03} マップ名-INFOCUS-I    PIC X(2).
```

Windows ビッグエンディアン用，UNIX 用

```
{02|03} マップ名-INCURSN      PIC S9(4) COMP.
{02|03} マップ名-INCURSM      PIC S9(4) COMP.
{02|03} マップ名-INCURSnN     PIC S9(4) COMP.         ……※
{02|03} マップ名-INCURSnM     PIC S9(4) COMP.         ……※
{02|03} マップ名-INFOCUS-I    PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

・行列（10 進）カーソルを指定した場合

```
{02|03} マップ名-INCURSY      PIC 9(4).
{02|03} マップ名-INCURSX      PIC 9(4).
{02|03} マップ名-INCURSnY     PIC 9(4).            ……※
{02|03} マップ名-INCURSnX     PIC 9(4).            ……※
{02|03} マップ名-INFOCUS-I    PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

・論理カーソルを指定した場合

```
{02|03} マップ名-INCURS-LOCI    PIC X(2).
{02|03} マップ名-INCURSnI       PIC X(2).            ……※
{02|03} マップ名-INFOCUS-I      PIC X(2).
```

注※

フィールドボックスを複数定義している場合に展開されます。

C 言語

・行列（2 進）カーソルを指定した場合

```
unsigned char マップ名_INCURSN[2];
unsigned char マップ名_INCURSM[2];
unsigned char マップ名_INCURSnN[2];            ……※
unsigned char マップ名_INCURSnM[2];            ……※
unsigned char マップ名_INFOCUS_I[2];
```

注※

フィールドボックスを複数定義している場合に展開されます。

・行列（10 進）カーソルを指定した場合

行列（2 進）カーソルに変換されます。

・論理カーソルを指定した場合

```
unsigned char マップ名_INCURS_LOCI[2];
unsigned char マップ名_INCURSnI[2];            ……※
unsigned char マップ名_INFOCUS_I[2];
```

注※

フィールドボックスを複数定義している場合に展開されます。

・マッピング規則

- AP がフォーカス項目データ名の領域に代入した内容と結果

入力操作	結果
画面送信時のフォーカス位置がプッシュボタンに位置づいている （ほかのオブジェクトにフォーカスが位置づいている場合で，マウスによるプッシュボタン選択は該当しない）	《プッシュボタンにボックス名称の指定がある》 ボックス名-T に対応するフォーカス定数を代入する。 《プッシュボタンにボックス名称の指定がない》 通知コードデータ名 T に対応するフォーカス定数を代入する。
画面送信時のフォーカス位置がプッシュボタン以外のオブジェクトに位置づいている	該当するオブジェクトに対応するフォーカス定数を代入する。
画面送信時のフォーカス位置がフィールドボックスに位置づいている	フィールドボックス内の先頭の行列位置にあるフィールドに対応するフォーカス定数を代入する。

注

入力単位が「フィールド単位」または「イベント単位」の場合，キー操作をしたオブジェクトの定数を代入します。

16.3.9　二次ウィンドウの表示

二次ウィンドウのマッピング規則について説明します。

(1)　初期表示（ウィンドウが何も表示されていない）の場合

出力するマップ	表示形態	画面表示結果※	フォーカス位置
一次ウィンドウ定義のマップ	一部上書， 全面書換， 自動	一次ウィンドウ定義時のウィンドウの表示位置と大きさで一次ウィンドウを表示する	出力した一次ウィンドウ
二次ウィンドウ定義のマップ	－	表示されない（XMAP3 でエラー）	－

（凡例）

－：該当しない。

注※

ウィンドウが表示できない（実画面からウィンドウがはみ出る）場合は，ウィンドウが表示できるように XMAP3 が位置を補正します。

(2)　一次ウィンドウだけが表示されている場合

出力するマップ		表示形態	画面表示結果※1	フォーカス位置
一次ウィンドウ定義のマップ	表示されているものと同じマップ名	一部上書	直前の一次ウィンドウを表示していた位置にそのときと同じ大きさで一次ウィンドウを表示する。	出力した一次ウィンドウ
		全面書換	表示・印刷セットアップの一次ウィンドウの位置を「ウィンドウ操作を優先する」にした場合，直前に一次ウィンドウを表示していた位置に今回表示する大きさで表示する。「画面定義を優先する」を指定した場合，定義した位置に表示する。	
	表示されているものと異なるマップ名	一部上書， 全面書換， 自動		

出力するマップ		表示形態	画面表示結果※1	フォーカス位置
二次ウィンドウ定義のマップ	一次ウィンドウ定義中に，同じ ID（ウィンドウ種別番号）の定義がある。	一部上書，全面書換，自動	**（一次ウィンドウ定義で二次ウィンドウ位置を指定した場合）** 一次ウィンドウ定義中の対応する ID（二次ウィンドウ用 ID）の表示位置※2 に表示する。 **（二次ウィンドウ定義でウィンドウ位置を指定した場合）** 二次ウィンドウ定義時の表示位置に二次ウィンドウを表示する。	出力した二次ウィンドウ
	一次ウィンドウ定義中に，二次ウィンドウの定義なし，または同じ ID（ウィンドウ識別番号）の二次ウィンドウの定義なし。		**（一次ウィンドウ定義で二次ウィンドウ位置を指定した場合）** 一次ウィンドウの位置に二次ウィンドウを表示する。 **（二次ウィンドウ定義でウィンドウ位置を指定した場合）** 二次ウィンドウ定義時の表示位置に二次ウィンドウを表示する。	

注※1

ウィンドウが表示できない（実画面からウィンドウがはみ出る）場合は，ウィンドウが表示できるように XMAP3 が位置を補正します。

注※2

端末オペレータによって，一次ウィンドウが移動されている場合，二次ウィンドウも同様に移動された位置に出力されます。

(3)　一次ウィンドウと二次ウィンドウが表示されている場合

出力するマップ		表示形態	画面表示結果※1	フォーカス位置
一次ウィンドウ定義のマップ	表示されているものと同じマップ名	全面書換	二次ウィンドウは消去される。	出力した一次ウィンドウ
		一部上書，自動	二次ウィンドウはそのまま表示される。	画面を表示した時点で対話権があった二次ウィンドウ
	表示されているものと異なるマップ名	一部上書，全面書換，自動	一次ウィンドウを表示する。二次ウィンドウは消去される。	出力した一次ウィンドウ
二次ウィンドウ定義のマップ	表示されているものと同じマップ	全面書換	**（一次ウィンドウ定義で二次ウィンドウ位置を指定した場合）** 一次ウィンドウ定義中の対応する ID（二次ウィンドウ用 ID）の表示位置※2 に，直前の二次ウィンドウの大きさで表示する。 **（一次ウィンドウ定義で二次ウィンドウ位置を指定しない場合）** 一次ウィンドウの位置に直前の二次ウィンドウの大きさで表示する。	出力した二次ウィンドウ

出力するマップ		表示形態	画面表示結果※1	フォーカス位置
			(二次ウィンドウ定義でウィンドウ位置を指定した場合) 二次ウィンドウ定義時の表示位置に直前の二次ウィンドウの大きさで表示する。	
		一部上書	直前の二次ウィンドウを表示していた位置※3にそのときと同じ大きさで二次ウィンドウを表示する。	
	表示されているものと異なるマップで，二次ウィンドウのID（ウィンドウ種別番号）は現在表示されているものと同じ	一部上書，全面書換， 自動	**(一次ウィンドウ定義で二次ウィンドウ位置を指定した場合)** 一次ウィンドウ定義中の対応するID（二次ウィンドウ用ID）の表示位置※2に表示する。このとき，直前の一次ウィンドウはそのまま表示され，二次ウィンドウは出力したウィンドウの内容と置き換えられる。 **(一次ウィンドウ定義で二次ウィンドウ位置を指定しない場合)** 一次ウィンドウの表示位置に二次ウィンドウを表示する。このとき，直前の一次ウィンドウはそのまま表示され，二次ウィンドウは出力したウィンドウの内容と置き換えられる。 **(二次ウィンドウ定義でウィンドウ位置を指定した場合)** 二次ウィンドウ定義時の表示位置に二次ウィンドウを表示する。このとき，直前の一次ウィンドウはそのまま表示され，二次ウィンドウは出力したウィンドウの内容と置き換えられる。	出力した二次ウィンドウ
	表示されているものと異なるマップで，二次ウィンドウのID（ウィンドウ種別番号）も異なる	一部上書， 全面書換， 自動	**(一次ウィンドウ定義で二次ウィンドウ位置を指定した場合)** 一次ウィンドウ定義中の対応するID（二次ウィンドウ用ID）の表示位置※2に表示する。このとき，直前の二次ウィンドウと一次ウィンドウは表示したままになる。 **(一次ウィンドウ定義で二次ウィンドウ位置を指定しない場合)** 一次ウィンドウの表示位置に二次ウィンドウを表示する。このとき，直前の二次ウィンドウと一次ウィンドウは表示したままになる。 **(二次ウィンドウ定義でウィンドウ位置を指定した場合)** 二次ウィンドウ定義時の表示位置に二次ウィンドウを表示する。このと	

出力するマップ		表示形態	画面表示結果※1	フォーカス位置
			き，直前の二次ウィンドウと一次ウィンドウは，表示したままになる。	

注※1

ウィンドウが表示できない（実画面からウィンドウがはみ出る）場合は，ウィンドウが表示できるように XMAP3 が位置を補正します。

注※2

端末オペレータによって，一次ウィンドウが移動されている場合，二次ウィンドウも同様に移動された位置に出力されます。

注※3

端末オペレータによってウィンドウが移動されている場合は，その位置に表示されます。

16.3.10　出力グラフィック

(1)　出力グラフィックの定義

出力グラフィックをドローで定義することで生成されます。

- 出力グラフィック項目のデータ名：マップ名-GRAPHnnnn-O
  データ名は，出力グラフィックダイアログの「データ名」で変更できます。
- 長さ
  ファイル名指定：12
  フルパス 64：64
  フルパス 128：128
  フルパス 259：259

(2)　出力グラフィックの論理マップ生成規則とマッピング規則

出力グラフィックの論理マップ生成規則とマッピング規則について説明します。

(a)　可変項目の出力論理マップの生成規則

COBOL

```
{02|03} マップ名-GRAPHnnnn-O PIC X(長さ).
```

C 言語

```
unsigned char マップ名_GRAPHnnnn_O[長さ];
```

(b)　可変項目のマッピング規則

- AP が出力項目データ名の領域に代入した内容と結果

AP が出力項目データ名の領域に代入した内容	結果
すべてデータの場合	（指定されたファイルが存在した場合） 　指定されたファイルを読み込み，グラフィックを表示する※。 （指定されたファイルが存在しない場合） 　表示されない。

APが出力項目データ名の領域に代入した内容	結果
先頭1文字がデータ有無コード	データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，表示形態が「全面書換」か「一部上書」によって次のように異なる。 全面書換の場合：何も表示しない。 一部上書の場合：直前に表示したデータのまま表示する。
データの後半にデータ有無コード，ヌルまたは空白	データ有無コード，ヌル（$(00)_{16}$），または空白（$(20)_{16}$）を$(xx)_{16}$と仮定した場合，例としてX'414243xxxx'のようなデータが該当する。 この場合，後半のX'xxxx'を切り捨てたデータのX'414243'が指定されたファイル名となる。 (指定されたファイルが存在した場合) 　指定されたファイルを読み込み，グラフィックを表示する※。 (指定されたファイルが存在しない場合) 　表示されない。

注※

データが「*CLIP」の場合，クリップボード中のグラフィックを表示します。

16.3.11　日付・時刻項目

(1)　日付・時刻項目の定義

GUI画面とGUI画面内のフィールドボックスおよびCUI画面では，各項目の名称が次の表のように異なります。

表16-4　可変項目の名称の違い

日付・時刻項目	GUI画面	フィールドボックスおよびCUI画面
出力項目	出力日付・時刻テキストボックス	出力日付・時刻フィールド
入出力項目	入出力日付・時刻テキストボックス	入出力日付・時刻フィールド

各項目の名称は，それぞれの画面での名称に置き換えてください。

- 出力項目

  出力日付・時刻テキストボックス，または出力日付・時刻フィールドに対して生成されます。

  - 出力項目：マップ名-FIELDnnnn-O
  - 繰り返し回数：縦，または横の反復回数
  - 長さ：論理項目の長さ。指定がないときはフィールドのデータ長
  - データ型：テキストボックス，フィールドのデータ型で設定

  表16-5　日付・時刻の出力項目のデータ型と使用できる項目

データ型	使用できる項目
99999	数字編集項目 (COBOLで桁数に満たない値を代入すると，COBOLの仕様によって「0」が設定されます)

データ型	使用できる項目
文字（XX）	文字

注

桁数に満たない値を代入し，その残りにデータ有無コードを指定している場合，「桁寄せ：左」，「埋字：スペース」で埋字されます。その他のマッピング規則については，「16.3.11(2)　日付・時刻項目の出力論理マップ生成規則とマッピング規則」を参照してください。

- 入力項目

入出力日付・時刻テキストボックス，または入出力日付・時刻フィールドに対して生成されます。次に示す入力項目以外の定義内容は出力項目と同じになります。

- 入力項目：マップ名-FIELDnnnn-I

日付，時刻入力項目では，入力する桁数をキー入力時にチェックするため，桁数に満たないデータの入力ができません。しかし，「入力済み」属性が設定されている場合，画面表示上のデータがそのまま入力データとなり，日付・時刻の出力項目のデータ型と同じデータ型指定に合ったデータで処理されます。日付・時刻の出力項目のデータ型については，「表 16-5　日付・時刻の出力項目のデータ型と使用できる項目」を参照してください。

- 入出力項目

入出力日付・時刻テキストボックス，または入出力日付・時刻フィールドに対して生成されます。論理マップは，入力論理マップと出力論理マップに分けて出力されます。定義内容は出力項目と同じになります。

(2)　日付・時刻項目の出力論理マップ生成規則とマッピング規則

日付・時刻項目の出力論理マップ生成規則と論理マップ生成規則について説明します。

(a)　可変項目の出力論理マップの生成規則

COBOL

・文字項目の場合

```
{02|03} マップ名-FIELDnnnn-O  PIC X(長さ)〔OCCURS 回数〕.
```

・数字編集項目の場合

```
{02|03} マップ名-FIELDnnnn-O  PIC 編集文字〔OCCURS 回数〕.
```

C 言語

C 言語使用時は，数字編集項目は指定できません。

```
unsigned char マップ名_FIELDnnnn_O  〔[回数]〕[長さ];
```

(b)　可変項目のマッピング規則

- AP が出力項目データ名の領域に代入した内容と結果

AP が出力項目データ名の領域に代入した内容	マッピングオプション	結果
すべてデータの場合	マージ，論理マップ	代入されたデータを表示する。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例として X'1F414234'または X'1F1F1F'のようなデータが該当する。このとき，表示形態が「全面書換」か「一部上書」によって次のように異なる。 全面書換の場合：何も表示しない。

APが出力項目データ名の領域に代入した内容	マッピングオプション	結果
		一部上書の場合：直前に表示したデータのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例として X'4142431F1F'のようなデータが該当する。この場合，代入されたデータを表示する（不足分は表示されない）。
－	物理マップ	データを表示しない。

（凡例）

－：該当しない。

(3)　日付・時刻項目の入力論理マップ生成規則とマッピング規則

日付・時刻項目の入力論理マップ生成規則と論理マップ生成規則について説明します。

(a)　入出力日付・時刻フィールドおよび入出力日付・時刻テキストを指定した場合

・入力論理マップの生成規則

COBOL

●論理マップ可変部の集団項目化を指定した場合

・数字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC 編集文字.
```

・文字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC X(長さ).
```

●論理マップ可変部の集団項目化を指定していない場合

・文字項目の場合

```
02 マップ名-FIELDnnnn-I     PIC X(長さ)  〔OCCURS 回数〕.
```

C言語

```
unsigned char マップ名_FIELDnnnn_I 〔[回数]〕[長さ];
```

・マッピング規則

- 入力操作と入力項目データ名の領域の内容

《使用目的が日本語以外の場合》

入力操作	結果（入力項目データ名の領域の内容）
データを入力	データを入力項目データ名の領域に代入する。
フィールドキーを押す（すべて$(00)_{16}$のデータ入力）	データ消去通知文字に従って入力項目データ名の領域をクリアする。フィールドキーを押したときのマッピング規則については，「16.2.10　フィールドキーを押したとき」を参照のこと。
入力操作なし	初期クリア文字に従って入力データ名の領域をクリアする。入力操作がなかったときのマッピング規則については，「16.2.12　入力操作がなかったとき」を参照のこと。

16.3.12　トグルフィールド

トグルフィールドの論理マップ生成規則とマッピング規則について説明します。

(1)　トグルフィールドの定義

- トグルフィールドの属性

  トグルフィールドダイアログで「動的変更（AP から表示属性を変更する)」を選ぶことで生成されます。

  - 制御項目のデータ名：マップ名-FIELDnnnn-A

    データ名は，トグルフィールドダイアログの「データ名」に従います。

  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「候補選択コントロール／コマンドコントロール」を選択し，「修飾名長」で長さを設定します。

- トグルフィールドの定義

  - ボックス項目のデータ名：マップ名-FIELDnnnn-I

    データ名は，トグルフィールドダイアログの「データ名」で変更できます。

  - 長さ：AP が受け取る項目のデータ長で指定

(2)　トグルフィールド属性の論理マップ生成規則とマッピング規則

トグルフィールド属性の論理マップ生成規則とマッピング規則について説明します。この項目は，ダイアログで「動的変更（AP から表示属性を変更する)」を指定したときだけ論理マップが生成されます。

(a)　論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-A  PIC X(長さ)〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_A〔[回数]〕[長さ];
```

(b)　マッピング規則

- AP が制御項目に代入した内容と表示結果

《「動的変更（AP から表示属性を変更する)」を指定した場合》

AP が制御項目に代入した内容	マッピングオプション	表示結果
セットアップで指定してある修飾名と同じ	マージ	トグルフィールドの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		標準の属性を使って表示する。
セットアップで指定してある修飾名と同じ	論理マップ	トグルフィールドの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または設定していない修飾名		(表示形態が「全面書換」のとき) 標準の属性を使って表示する。

APが制御項目に代入した内容	マッピングオプション	表示結果
		（表示形態が「一部上書」で直前に表示したマップと同じ場合） 前回の属性のまま表示する。 （表示形態が「一部上書」で直前に表示したマップと異なる場合） 標準の属性を使って表示する。
－	物理マップ	標準の属性を使って表示する。

（凡例）

－：該当しない。

注

修飾名と修飾名に対応したフィールド属性情報は，ドローセットアップの「表示属性の動的変更」の「候補選択コントロール」タブで変更できます。

《「動的変更（APから表示属性を変更する）」を指定しない場合》

標準の属性を使って表示します。

(3)　トグルフィールドの入力論理マップ生成規則とマッピング規則

トグルフィールドの入力論理マップ生成例とマッピング規則について説明します。

(a)　論理マップ生成規則

COBOL

・論理マップ可変部の集団項目化を指定した場合

```
03 マップ名-FIELDnnnn-H.
  04 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

・論理マップ可変部の集団項目化を指定しない場合

```
02 マップ名-FIELDnnnn-I  PIC {X(長さ)|N(長さ÷2)|9(長さ)}.
```

C言語

```
unsigned char マップ名_FIELDnnnn_I[長さ];
```

(b)　マッピング規則

- 入力操作と結果（論理項目の内容）

入力操作	結果（論理項目の内容）
フィールド中の文字を変更する	選んだ文字に対応する定数を論理項目に設定する。
フィールド中の文字を変更しない，またはフィールドにカーソルを位置づけない	初期クリア文字に従って論理項目をクリアする。

16.3.13　スピンボックス

スピンボックスの論理マップ生成規則とマッピング規則について説明します。

(1)　スピンボックスの定義

- スピンボックスの属性

スピンボックスダイアログで「動的変更（AP から表示属性を変更する)」を選ぶと生成されます。

- 制御項目のデータ名：マップ名-FIELDnnnn-A
  データ名は，スピンボックスダイアログの AP が渡す項目の「データ名」に従います。
- 長さ：2
  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「キャラクタコントロール（キーエントリ／選択エントリ)」を選択し，「修飾名長」で長さを設定します。

- スピンボックスの定義
  - ボックス項目のデータ名：マップ名-FIELDnnnn-I/O
    データ名は，スピンボックスダイアログの AP が受け取る項目の「データ名」で変更できます。
  - 長さ：AP が受け取る項目のデータ長で指定

(2)　スピンボックスの論理マップ生成規則とマッピング規則

(a)　項目属性の論理マップ生成規則とマッピング規則

スピンボックスの項目属性に関する論理マップ生成規則とマッピング規則について説明します。この論理マップは，スピンボックスダイアログで「動的変更（AP から表示属性を変更する)」を選び，属性項目データ名を指定することで生成されます。

- 論理マップ生成規則

  COBOL

  ```
  {02|03} マップ名-FIELDnnnn-A  PIC X(長さ).
  ```

  C 言語

  ```
  unsigned char  マップ名_FIELDnnnn_A[長さ];
  ```

- マッピング規則
  - AP が制御項目に代入した内容と表示結果

《「動的変更（AP から表示属性を変更する)」を指定した場合》

AP が制御項目に代入した内容	マッピングオプション	表示結果
修飾名と同じ	マージ	スピンボックスの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または 修飾名以外（上記以外）		標準の属性を使って表示する。
修飾名と同じ	論理マップ	スピンボックスの修飾名に対応する属性を使って表示する。
先頭にデータ有無コード，または 修飾名以外（上記以外）		(表示形態が「全面書換」の場合) 　標準の属性を使って表示する。 (表示形態が「一部上書」で直前に表示したマップと同じ場合) 　前回の属性をそのまま表示する。 (表示形態が「一部上書」で直前に表示したマップと異なる場合) 　標準の属性を使って表示する。

APが制御項目に代入した内容	マッピング オプション	表示結果
−	物理マップ	標準の属性を使って表示する。

（凡例）

−：該当しない。

注

修飾名と修飾名に対応したテキスト・フィールド変更属性情報は，ドローセットアップの「表示属性の動的変更」の「キャラクタコントロール」タブで変更できます。

《「動的変更（APから表示属性を変更する）」を指定しない場合》

標準の属性を使って表示します。

(b)　可変項目の出力論理マップ生成規則とマッピング規則

可変項目の出力論理マップ生成規則と論理マップ生成規則について説明します。

- 出力論理マップの生成規則

COBOL

COBOLの可変項目の生成規則は，使用目的によってデータ型が変わります。

・数字項目の場合

```
{02|03} マップ名-FIELDnnnn-O  PIC 9(長さ)〔OCCURS 回数〕.
```

・文字項目の場合

```
{02|03} マップ名-FIELDnnnn-O  PIC X(長さ)〔OCCURS 回数〕.
```

・数字編集項目の場合

```
{02|03} マップ名-FIELDnnnn-O  PIC 編集文字〔OCCURS 回数〕.
```

C言語

C言語使用時は，数字編集項目は指定できません。

```
unsigned char マップ名_FIELDnnnn_O  〔[回数]〕[長さ];
```

- マッピング規則
  - APが出力項目データ名の領域に代入した内容と結果

APが出力項目データ名の領域に代入した内容	マッピング オプション	結果
すべてデータの場合	マージ，論理マップ	代入されたデータを表示する※。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を設定した場合，例としてX'1F414234'またはX'1F1F1F'のようなデータが該当する。このとき，表示形態が「全面書換」か「一部上書」によって次のように異なる。 全面書換の場合：最小値を表示する。 一部上書の場合：直前に表示したデータのまま表示する。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合，例としてX'4142431F1F'のようなデータが該当する※。
−	物理マップ	最小値を表示する。

（凡例）

－：該当しない。

注※

代入されたデータを表示します。ただし，値が範囲外の場合には，最初の操作によって最大値，最小値のどちらか近い値を表示します。

(c)　スピンボックスの入力論理マップ生成規則とマッピング規則

スピンボックスの入力論理マップ生成規則と論理マップ生成規則について説明します。

- 入力論理マップの生成規則

COBOL

COBOL の可変項目の生成規則は，使用目的によってデータ型が変わります。

・論理マップ可変部の集団項目化を指定した場合

・数字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC 編集文字.
```

・文字項目の場合

```
03 マップ名-FIELDnnnn-H 〔OCCURS 回数〕.
  04 マップ名-FIELDnnnn-I    PIC X(長さ).
```

・論理マップ可変部の集団項目化を指定していない場合

・数字項目の場合

```
02 マップ名-FIELDnnnn-I    PIC 9(長さ)  〔OCCURS 回数〕.
```

・文字項目の場合

```
02 マップ名-FIELDnnnn-I     PIC X(長さ)  〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_I 〔[回数]〕[長さ];
```

- マッピング規則
  - 入力操作と入力項目データ名の領域の内容

《使用目的が日本語以外の場合》

入力操作	結果（入力項目データ名の領域の内容）
スピンボックスにフォーカスを位置づけ，ボタン操作で表示データを変更する。	選択したデータを入力項目データ名の領域に代入する。
入力操作なし	初期クリア文字に従って入力データ名の領域をクリアする。入力操作がなかったときのマッピング規則については，「16.2.12　入力操作がなかったとき」を参照のこと。

16.3.14　数値・金額項目

数値・金額項目の論理マップ生成規則とマッピング規則について説明します。

(1)　数値・金額項目の定義

- 入力項目　　：マップ名-FIELDnnnn-I
  数字編集文字列：データ型で指定できる数字編集文字列は次のとおり。「9」，「V」，「S」
- 繰り返し回数　：縦，または横の反復回数（入出力フィールドだけ）

(2)　論理マップ生成規則

COBOL

```
{02|03} マップ名-FIELDnnnn-H 〔OCCRUS 回数〕.
  {03|04} マップ名-FIELDnnnn-I  PIC 編集文字.
```

注

使用目的が金額の場合，編集文字に小数点を示す「V」を使用できます。

(3)　マッピング規則

- 入力操作と結果

入力操作	結果
フィールドキーを押した場合	データ消去通知文字に従って入力項目データ名の領域をクリアする。データ消去通知文字が埋字として指定してあるときは，0（$(30)_{16}$）でクリアする。
入力操作がない，または END キーを押した場合。	（初期値が指定してある場合） 初期値で指定した定数を入力項目データ名の領域に代入する。 （初期値が指定していない場合） 初期クリア文字が埋字として指定してあるときは，0（$(30)_{16}$）でクリアする。
全桁に空白を入力した場合	空白（$(20)_{16}$）で入力項目データ名の領域をクリアする。また，NULL と空白を混在している場合も，すべて空白となる。
数字項目として正しいデータを入力した場合	数字編集項目の指定に従って入力項目データ名の領域に値を代入する。
数字項目として不正なデータを入力した場合※	入力チェックを指定した場合，入力時にエラーとなる。入力チェックを指定していない場合，エラー通知文字に従って入力項目データ名の領域をクリアする。エラー通知文字の標準は HIGH（X'FF'）である。

注※

「入力済み」属性を指定して，整数桁や小数桁が合わないデータを表示して画面確定している場合，不正なデータとなります。

16.3.15　ウィンドウ表示位置

(1)　ウィンドウ表示位置の制御情報

- 制御情報の標準値
  - ウィンドウ位置制御データ名：マップ名-WINDOWW

    データ名は，ドローセットアップの「ドローの設定」から表示する，表示属性の動的変更ダイアログの「データ名」タブで変更できます。
  - 長さ：2

    長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定...］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。変更するときには，「動的変更の種別」から「位置属性」を選択し，「修飾名長」で長さを設定します。

(2)　論理マップ生成規則

ウィンドウ表示位置の制御情報の論理マップ生成例について説明します。

COBOL

```
{02|03} マップ名-WINDOWW  PIC X(長さ).
```

C 言語

```
unsigned char マップ名_WINDOWW[長さ];
```

(3)　マッピング規則

ウィンドウ表示位置の制御情報のマッピング規則について説明します。

- AP がウィンドウ表示位置の制御情報のデータ名の領域に代入した内容とその結果

AP がウィンドウ表示位置の制御情報のデータ名の領域に代入した内容	結果
修飾名と同じ	ウィンドウ表示位置の制御情報の修飾名に対応するウィンドウ制御情報を使って画面が表示される※。
データの先頭またはすべてがデータ有無コード，または修飾名以外（上記以外）。	前回表示したウィンドウ制御情報のまま画面が表示される。

注

修飾名と修飾名に対応したウィンドウ表示位置の制御情報は，ドローセットアップの「表示属性の動的変更」の「位置属性」タブで変更できます。

注※

1 回目の画面表示，または「全面書換」で画面が表示される場合にだけ，指定が有効になります。

16.4　可変部（帳票）

ここでは，帳票の論理マップ生成規則とマッピング規則について説明します。

なお，論理項目に指定する文字データとして，LOW$(00)_{16}$ およびデータ有無コード$(1F)_{16}$ を除く制御コード（JIS：$(00)_{16}$～$(1F)_{16}$，$(7F)_{16}$）は指定できません。

16.4.1　可変項目

ここでは，可変部の出力に関する論理マップ生成規則とマッピング規則について説明します。なお，この項では，COBOL の数字項目の記述を次のようにしています。

- PIC 9（長さ）と表記している項目
  実際に生成される論理マップは，PIC 99999 のように 9 が長さ分生成されます。

(1)　可変項目の定義

- 出力項目
  出力フィールドに対して生成されます。
  - 出力項目のデータ名：マップ名-FIELDnnnn-O
    データ名は，出力フィールドダイアログの AP が渡す項目の「データ名」で変更できます。
  - 繰り返し回数：縦，または横の反復回数
  - 長さ：論理項目の長さ。指定がないときはフィールドのデータ長
  - データ型：フィールドのデータ型で設定
  - 使用目的：フィールドの使用目的で設定
    使用目的には，次の 3 項目があります。

使用目的	使用できる項目
数字	文字，数字編集
日本語	文字，漢字
英数	文字

(2)　出力項目の論理マップ生成規則

出力項目の論理マップ生成規則について説明します。

(a)　初期値を指定した場合

COBOL

COBOL の可変項目の生成規則は，使用目的によってデータ型が変わります。

・数字項目の場合

```
{02|03} マップ名-FIELDnnnn-O PIC 9(長さ)〔OCCURS 回数〕.
```

・文字項目の場合

```
{02|03} マップ名-FIELDnnnn-O PIC X(長さ)〔OCCURS 回数〕.
```

・漢字項目の場合

```
{02|03} マップ名-FIELDnnnn-O PIC N(長さ÷2)〔OCCURS 回数〕.
```

・数字編集項目の場合

{02|03} マップ名-FIELDnnnn-O PIC 編集文字〔OCCURS 回数〕.

C 言語

unsigned char マップ名_FIELDnnnn_O　〔[回数]〕[長さ];

C 言語使用時は，数字編集項目は指定できません。

日本語では，「桁寄せ」は左寄せになります。

(b)　初期値を指定しない場合

初期値を指定しない場合の論理マップ生成例は初期値を指定したときと同じです。

(3)　出力項目のマッピング規則

出力項目のマッピング規則について説明します。

(a)　初期値を指定した場合

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて空白（$(20)_{16}$）	項目全体を印字しません。
	すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
	すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
	上記以外のデータ	代入されたデータを印字します。
先頭 1 文字がデータ有無コード		次表を参照してください。
データの後半にデータ有無コード		「桁寄せ」の指定に従って桁寄せをし，埋字に指定した文字で埋めて印字します（数字編集項目の場合は，編集文字に従って編集してから印字）。埋字と桁寄せのマッピング規則については，「16.4.4　埋字と桁寄せ（出力フィールド）」を参照してください。

- 先頭 1 文字がデータ有無コードのときの結果

初期値の内容	結果
すべて空白（$(20)_{16}$）	項目全体を印字しせん。
すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
初期値の長さと帳票の印刷長とが等しい	初期値を項目に印字します。
初期値の長さが帳票の印刷長より短い	定義で指定した「桁寄せ」，および埋字に従って印字します。埋字抑止を指定した場合は，左寄せで初期値を項目に印字します。
初期値の長さが帳票の印刷長より長い	「桁寄せ」に従って桁寄せし，余りを切り捨てて項目を印字します。

(b)　初期値を指定しない場合

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて空白（$(20)_{16}$）	項目全体を印字しません。
	すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
	すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
	上記以外のデータ	代入されたデータを印字します。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414234'または X'1F1F1F'のようなデータが該当します。このときは，データを印字しません。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，「桁寄せ」の指定に従って桁寄せをし，埋字に指定した文字で埋めて印字します。埋字と桁寄せのマッピング規則については，「16.4.4　埋字と桁寄せ（出力フィールド）」を参照してくだい。

16.4.2　印刷枚数

印刷枚数の論理マップ生成規則について説明します。

(1)　印刷枚数の定義

ドローセットアップの「網掛け／グラフィック帳票」の印刷枚数動的変更のデータ名で，「アプリケーションでの印刷枚数指定領域の生成」を指定することで生成されます。

- 印刷枚数制御項目のデータ名：マップ名-COPIESO
  データ名は，ドローセットアップの「網掛け／グラフィック帳票」の「コピー枚数動的変更のデータ名」で変更できます。
- 長さ：2

(2)　印刷枚数の論理マップ生成規則

(a)　制御項目による印刷枚数を指定した場合

COBOL

・Windows リトルエンディアン用

```
{02|03} マップ名-COPIESO  PIC S9(4)  COMP-5.
```

・Windows ビッグエンディアン用

```
{02|03} マップ名-COPIESO  PIC S9(4)  COMP.
```

C 言語

```
unsigned char マップ名_COPIESO [2];
```

(b)　制御項目による印刷枚数を指定しない場合

論理マップは生成されません。

(3)　印刷枚数のマッピング規則

印刷枚数のマッピング規則について説明します。

- AP が印刷枚数制御項目データ名の領域に代入した内容とその結果

AP が印刷枚数制御項目データ名の領域に代入した内容	結果
不正な枚数（0，または 33 以上）	1 枚印刷します。
正しい枚数（1～32）	制御項目で指定した枚数を印刷します。
データ有無コードクリア	帳票属性ダイアログの「印刷部数」で指定した枚数を印刷します。

16.4.3　印刷ドキュメント名

印刷ドキュメント名の論理マップ生成規則とマッピング規則について説明します。

(1)　印刷ドキュメント名の定義

ドローセットアップの「けい線／プレプリント帳票」または「網掛け／グラフィック帳票」の印刷ドキュメント名動的変更のデータ名で，「アプリケーションでの印刷ドキュメント名指定領域の生成」を指定することで生成されます。

- 印刷ドキュメント名項目のデータ名：マップ名-DOCNAMEO
- 長さ：1～607
  データ名および長さは，ドローセットアップの「けい線／プレプリント帳票」または「網掛け／グラフィック帳票」の「印刷ドキュメント名動的変更のデータ名」で変更できます。

(2)　印刷ドキュメント名の論理マップ生成規則

COBOL

```
{02|03} マップ名-DOCNAMEO PIC X（長さ）.
```

C 言語

```
unsigned char マップ名_DOCNAMEO［長さ］;
```

(3)　印刷ドキュメント名のマップピング規則

印刷ドキュメント名のマップピング規則について説明します。

(a)　プリンタ出力の場合

- 「印刷ドキュメント名を AP で変更する」を指定した場合

AP が出力項目データ名の領域に代入した内容		結果（スプールに表示されるドキュメント名）
すべてデータ	すべて空白（$(20)_{16}$）	空白で出力します。
	すべて NULL（$(00)_{16}$）	「XMAP3」を出力します。

AP が出力項目データ名の領域に代入した内容		結果（スプールに表示されるドキュメント名）
の場合	上記以外のデータ	指定したデータを出力します。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414243'または X'1F1F1F'のようなデータが該当します。 • ドロー定義時に「印刷ドキュメント名」を指定した場合 指定した印刷ドキュメント名を出力します。 • ドロー定義時に「印刷ドキュメント名」を指定しない場合 「XMAP3」を出力します。
先頭 1 文字が NULL（$(00)_{16}$）		「XMAP3」を出力します。
データの途中に NULL（$(00)_{16}$）		先頭から NULL の直前までのデータを出力します。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，データ有無コードのデータ（X'414243'）を出力します。

注

表示・印刷セットアップの「プリンタ」タブの「用紙の確認通知」で，「物理マップ名で確認する」を指定している場合，この表で示す出力内容の末尾に「FNAM-物理マップ名」が付加されます。物理マップ名に拡張子（.pmp）は含まれません。

- 「印刷ドキュメント名を AP で変更する」を指定しない場合
  - ドロー定義時に「印刷ドキュメント名」を指定した場合
    指定した印刷ドキュメント名を出力します。
  - ドロー定義時に「印刷ドキュメント名」を指定しない場合
    「XMAP3」を出力します。

注

表示・印刷セットアップの「プリンタ」タブの「用紙の確認通知」で，「物理マップ名で確認する」を指定している場合，出力内容の末尾に「△△△FNAM-物理マップ名」が付加されます。物理マップ名に拡張子（.pmp）は含まれません。

(b)　PDF ファイル出力の場合

- 「印刷ドキュメント名を AP で変更する」を指定した場合

AP が出力項目データ名の領域に代入した内容		結果（スプールに表示されるドキュメント名）
すべてデータの場合	すべて空白（$(20)_{16}$）	物理マップ名に「.pdf」※を付加したファイル名で出力します。
	すべて NULL（$(00)_{16}$）	物理マップ名に「.pdf」※を付加したファイル名で出力します。
	上記以外のデータ	指定したデータに「.pdf」※を付加したファイル名で出力します。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414243'または X'1F1F1F'のようなデータが該当します。 • ドロー定義時に「印刷ドキュメント名」を指定した場合 指定した印刷ドキュメント名に「.pdf」※を付加したファイル名で出力します。

AP が出力項目データ名の領域に代入した内容	結果（スプールに表示されるドキュメント名）
	• ドロー定義時に「印刷ドキュメント名」を指定しない場合 物理マップ名に「.pdf」※を付加したファイル名で出力します。
先頭 1 文字が NULL（$(00)_{16}$）	物理マップ名に「.pdf」※を付加したファイル名で出力します。
データの途中に NULL（$(00)_{16}$）	先頭から NULL の直前までのデータに「.pdf」※を付加したファイル名で出力します。
データの後半にデータ有無コード	データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，データ有無コードのデータ（X'414243'）に「.pdf」※を付加したファイル名で出力します。
データの後半に空白（$(20)_{16}$）	例として，X'4142432020'のようなデータが該当します。この場合，空白以前のデータ（X'414243'）に「.pdf」※を付加したファイル名で出力します。

注
　物理マップ名に拡張子（.pmp）は含まれません。

注※
　データの末尾にすでに「.pdf」が設定されている場合は付加しません。

- **「印刷ドキュメント名を AP で変更する」を指定しない場合**
  - ドロー定義時に「印刷ドキュメント名」を指定した場合
    指定した印刷ドキュメント名に「.pdf」を付加したファイル名で出力します。
  - ドロー定義時に「印刷ドキュメント名」を指定しない場合
    物理マップ名に「.pdf」を付加したファイル名で出力します。

注
　物理マップ名に拡張子（.pmp）は含まれません。

16.4.4　埋字と桁寄せ（出力フィールド）

出力フィールドの埋字と桁寄せに関するマッピング規則について説明します。

定義	結果（印刷結果）
埋字に「埋めない」以外を指定	桁寄せ向きに従って桁寄せし※，埋字に指定した文字で埋めます。
埋字に「埋めない」を指定	初期値，および出力論理データは，桁寄せ向きに関係なく左寄せし，埋字を代入しません。ただし，埋字を代入しないことで詰まることはありません。帳票上の次の項目位置はドロー画面で指定した位置に印字されます。

注※
　使用目的が日本語の場合，左寄せになります。

16.4.5　バーコード

(1)　バーコードの定義

バーコードをドローで定義することで生成されます。

- 出力バーコード項目のデータ名：
  - 出力バーコードの場合：マップ名-BARCODEnnnn-O
    データ名は，出力バーコードダイアログの「データ名」で変更できます。
  - 連結出力バーコードの場合：マップ名-BARCODEnnnn-nnn-O
    データ名は，連結出力バーコードダイアログの分類の「データ名（部分)」で変更できます。
- 長さ：
  - JAN13 の場合：13
  - JAN8 の場合：8
  - CODE39 の場合：3～76
  - NW-7 の場合：3～99
  - ITF(6)の場合：6
  - ITF(14)の場合：14
  - ITF(16)の場合：16
  - カスタマの場合：22
  - GS1-128 の場合：4～240

(2)　バーコードの論理マップ生成規則

バーコードの論理マップ生成規則について説明します。

COBOL

・出力バーコードの場合

```
{02|03} マップ名-BARCODEnnnn-O PIC X(長さ)〔OCCURS 回数〕.
```

・連結出力バーコードの場合

```
{02|03} マップ名-BARCODEnnnn-O〔OCCURS 回数〕.
  {03|04} マップ名-BARCODEnnnn-nnn-O PIC X(長さ).
```

C 言語

・出力バーコードの場合

```
unsigned char マップ名_BARCODEnnnn_O  〔[回数]〕[長さ];
```

・連結出力バーコードの場合

```
struct{
  unsigned char マップ名_BARCODEnnnn_nnn_O[長さ];
}unsigned char マップ名_BARCODEnnnn_O〔[回数]〕;
```

(3)　バーコードのマッピング規則

バーコードのマッピング規則について説明します。

《AP が出力項目データ名の領域に代入した内容と結果》

- JAN，CODE39，NW-7，ITF の場合

AP が出力項目データ名の領域に代入した内容	結果
すべてデータの場合	代入されたデータに対応するバーコードを印字します。
先頭 1 文字がデータ有無コード	データ有無コード$(1F)_{16}$を仮定した場合

APが出力項目データ名の領域に代入した内容	結果
	例として，X'1F414234'またはX'1F1F1F'のようなデータが該当します。このときは，何も印字しません。
データの後半にデータ有無コード	データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，データ有無コード以前のデータに対応するバーコードを印字します。 バーコードの種類によって，何も印字されない場合があります。

- カスタマバーコードの場合

APが出力項目データ名の領域に代入した内容		結果
すべてデータの場合（22文字）※	すべて正しいデータ	代入されたデータに対するバーコードを印字します。
	すべて空白（$(20)_{16}$），NULL（$(00)_{16}$）	すべて制御コード（CC4）※で印字します。
	データの途中が空白（$(20)_{16}$）またはNULL（$(00)_{16}$）	空白（$(20)_{16}$）またはNULL（$(00)_{16}$）の部分は，制御コード（CC4）※で印字します。
	データの途中およびすべて不当データ	何も印字しません。
先頭1文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414243'またはX'1F1F1F'のようなデータが該当します。この場合，何も印字しません。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，何も印字しません。

注※

カスタマバーコードのデータについては，マニュアル「XMAP3 開発ガイド」を参照してください。

- GS1-128の場合

一つのバーコードデータが複数の項目から成る場合，項目単位にマッピングし，その結果を連結して一つのバーコードを印字します。次の表では，項目単位のマッピング規則について説明します。

APが出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて正しいデータ	コードセットの指定がBまたはCの場合 代入されたデータの先頭にコードセットの制御コード（CodeBまたはCodeC）※1を付けて，対応するバーコードを印字します。 コードセットの指定が継続の場合 代入されたデータに対応するバーコードを印字します。
	すべて空白（$(20)_{16}$）またはNULL（$(00)_{16}$）	コードセットの指定がBまたはCの場合 コードセットの制御コード（CodeBまたはCodeC）※1に対応するバーコードを印字します※2。 コードセットの指定が継続の場合

AP が出力項目データ名の領域に代入した内容		結果
		何も印字しません※3。
	データの途中およびすべて不当データ	何も印字しません※4。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414243'のようなデータが該当します。 コードセットの指定が B または C の場合 コードセットの制御コード（CodeB または CodeC）※1 に対応するバーコードを印刷します※2。 コードセットの指定が継続の場合 何も印字しません※3。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。 コードセットの指定が B または C の場合 代入されたデータの先頭にコードセットの制御コード（CodeB または CodeC）※1 を，データ有無コード以前のデータ末尾に制御コード（FNC1）※1 を付けて，対応するバーコードを印字します。 コードセットの指定が継続の場合 データ有無コード以前のデータ末尾に制御コード（FNC1）※1 を付けて，対応するバーコードを印字します。

注※1

制御コードは一つだけ付けます。データ文字には出力されません。

注※2

すべての項目に対して，すべて空白，すべて NULL または先頭 1 文字にデータ有無コードが指定されている場合，何も印字しません。

注※3

ほかの項目データのマッピング結果でバーコードを印字します。

注※4

不当データが含まれている場合，ほかの項目データの内容に関係なくバーコードを印字しません。

なお，GS1-128 バーコードとして印字するバーコードは，AP から指定されたバーコードの桁数の総和が 2 以上の場合だけ印字されます。ここで，バーコードの桁数とは，AP から指定されたデータのうち，バーコードパターンとならないデータ※1 を除いた次のデータの総和です。

- コードセット C のデータ桁数の半分※2
- コードセット B のデータ桁数
- 項目のコードセットの属性が C から B，または B から C に切り替わった回数
- データ後半にデータ有無コードがある項目の数

注※1

コードセット C の項目に指定された(,)，-，空白。

注※2

各項目で連続するコードセット C の項目のデータ桁数総和が偶数でない（2 で割り切れない）場合，データ不正となり，バーコードは破棄されます。

AP からの指定データによって印字されないで，破棄されるケースの例を次に示します。

- ケース 1：有効なバーコード桁数の不足

  定義項目リストボックスの定義内容

データ名（部分）	桁	コードセット	データ文字
BARCODE0001-001-	4	C（継続）	改行無し

  AP からの指定

  BARCODE0001-001 に「(91)」を設定

  破棄理由

  BARCODE0001-001 に設定された値は「(91)」であるため，有効なバーコードデータは「91」となり，バーコードの桁数が 1 となるため。

- ケース 2：データ桁の総数が偶数でない

  定義項目リストボックスの定義内容

データ名（部分）	桁	コードセット	データ文字
BARCODE0001-001-	1	C（継続）	改行無し
BARCODE0001-002-	5	C（継続）	改行無し

  AP からの指定

  BARCODE0001-001 に「-」を設定

  BARCODE0001-002 に「12345」を設定

  破棄理由

  BARCODE0001-001 に設定された「-」はバーコードパターンの対象外であるため，有効なバーコードデータは「12345」（奇数）となり，コードセット C では表現できないため。

16.4.6　OCR

(1)　OCR 可変項目の定義

- OCR 項目

  OCR 出力フィールドに対して生成されます。

  - OCR 項目のデータ名：マップ名-FIELDnnnn-O

    データ名は，出力 OCR ダイアログの「データ名」で変更できます。
  - 繰り返し回数：縦，または横の反復回数
  - 長さ：論理項目の長さ。指定がないときはフィールドのデータ長
  - データ型：文字，数字編集項目のどちらかを選択

(2)　OCR 項目の出力論理マップ生成規則

OCR の出力論理マップ生成規則について説明します。

(a)　初期値を指定した場合

COBOL

COBOL の OCR の生成規則は，データ型によって変わります。

・文字項目の場合

```
03 マップ名-FIELDnnnn-O PIC X(長さ)〔OCCURS 回数〕.
```

・数字編集項目の場合

```
03 マップ名-FIELDnnnn-O PIC 編集文字〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_O　〔[回数]〕[長さ];
```

C 言語使用時は，数字編集項目は指定できません。

日本語では，「桁寄せ」は左寄せになります。

(b)　初期値を指定しない場合

初期値を指定しない場合の論理マップ生成規則は初期値を指定したときと同じです。

(3)　OCR 項目のマッピング規則

OCR のマッピング規則について説明します。

(a)　初期値を指定した場合

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて空白（$(20)_{16}$）	項目全体を空白で印字します。
	すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
	すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
	上記以外のデータ	代入されたデータを印字します。
先頭 1 文字がデータ有無コード		次表を参照してください。
データの後半にデータ有無コード		「桁寄せ」の指定に従って桁寄せをし，埋字に指定した文字で埋めて印字します(数字編集項目の場合は，編集文字に従って編集してから印字)。埋字と桁寄せのマッピング規則については，「16.4.4　埋字と桁寄せ（出力フィールド)」を参照してください。

《先頭 1 文字がデータ有無コードのときの結果》

初期値の内容	結果
すべて空白（$(20)_{16}$）	項目全体を空白で印字します。
すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
初期値の長さと帳票の印刷長とが等しい	初期値を項目に印字します。
初期値の長さが帳票の印刷長より短い	定義で指定した「桁寄せ」，および埋字に従って印字します。埋字抑止を指定した場合は，左寄せで初期値を項目に印字します。
初期値の長さが帳票の印刷長より長い	「桁寄せ」に従って桁寄せし，余りを切り捨てて項目を印字します。

(b)　初期値を指定しない場合

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて空白（$(20)_{16}$）	項目全体を空白で印字します。
	すべて 0（$(30)_{16}$）	項目全体を 0 で印字します。
	すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
	上記以外のデータ	代入されたデータを印字します。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414234'または X'1F1F1F'のようなデータが該当します。このときは，データを印字しません。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'4142431F1F'のようなデータが該当します。この場合，「桁寄せ」の指定に従って桁寄せをし，埋字に指定した文字で埋めて印字します。埋字と桁寄せのマッピング規則については，「16.4.4　埋字と桁寄せ（出力フィールド）」を参照してください。

16.4.7　出力グラフィック

(1)　出力グラフィックの定義

出力グラフィックをドローで定義することで生成されます。

- 出力グラフィック項目のデータ名：マップ名-GRAPHnnnn-O
  データ名は，出力グラフィックダイアログの「データ名」で変更できます。
- 長さ：
  - ファイル名指定：12
  - フルパス 64：64
  - フルパス 128：128
  - フルパス 259：259

(2)　出力グラフィックの論理マップ生成規則

出力グラフィックの論理マップ生成規則について説明します。

COBOL

```
{02|03} マップ名-GRAPHnnnn-O PIC X(長さ).
```

C 言語

```
unsigned char マップ名_GRAPHnnnn_O[長さ];
```

(3)　出力グラフィックのマッピング規則

出力グラフィックのマッピング規則について説明します。

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容	結果
すべてデータの場合	•「指定されたファイルが存在した場合」 指定されたファイルを読み込み，グラフィックを印刷します※。 •「指定されたファイルが存在しない場合」 何も印刷しません。
先頭 1 文字がデータ有無コード	データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414234'または X'1F1F1F'のようなデータが該当します。このときは，何も印刷しません。
データの後半にデータ有無コード，NULL または空白	データ有無コード，NULL（$((00)_{16}$），または空白（$(20)_{16}$）を$(xx)_{16}$と仮定した場合，例として X'414243xxxx'のようなデータが該当します。この場合，後半の X'xxxx'を切り捨てたデータの X'414243'が指定されたファイル名となります。 •「指定されたファイルが存在した場合」 指定されたファイルを読み込み，グラフィックを印刷します※。 •「指定されたファイルが存在しない場合」 何も印刷しません。

注※

データが「*CLIP」の場合，クリップボード中のグラフィックを印刷します。

16.4.8　制御項目

(1)　制御項目の定義

- 制御項目のデータ名：マップ名-FIELDnnnn-A

  データ名は，各オブジェクトのダイアログで指定している「データ名」に従います。ただし，けい線の場合は，「制御項目データ名」で変更できます。
- 長さ：2

  長さは，ドローセットアップの「運用管理者用の設定」の［修飾名の設定...］ボタンをクリックして表示される，修飾名の設定ダイアログで変更できます。出力フィールド用の長さを変更するときには「動的変更の種別」から「[帳票] フィールド表示属性」を，けい線用の長さを変更するときには「[帳票] けい線表示属性」を選択し，「修飾名長」で長さを設定します。
- 繰り返し：フィールドの繰り返しと同じ

(2)　制御項目の論理マップ生成規則

制御項目の論理マップ生成規則について説明します。

COBOL

```
{02|03} マップ名-FIELDnnnn-A PIC X(長さ) [OCCURS 回数].
```

C 言語

```
unsigned char マップ名_FIELDnnnn_A [[回数]] [長さ];
```

(3)　制御項目のマッピング規則

制御項目のマッピング規則について説明します。

《AP が制御項目データ名の領域に代入した内容と結果》

- 出力フィールド，日付／時刻フィールドおよびけい線ダイアログで「動的変更（AP から表示属性を変更します）」を選んだ場合

AP が出力項目データ名の領域に代入した内容	マッピングオプション	結果
修飾名と同じ	マージ	修飾名に対応する表示属性を使って項目を表示します。
先頭にデータ有無コード，または修飾名以外（上記以外）		標準の属性を使って項目を表示します。

注

修飾名と修飾名に対応した変更属性情報は，ドローセットアップの表示属性の動的変更で変更できます。

- 出力フィールド，日付／時刻フィールドおよびけい線ダイアログで「動的変更（AP から表示属性を変更します）」を選ばない場合

標準の属性を使用して項目を表示します。

16.4.9　日付／時刻

(1)　日付／時刻項目の定義

- 日付／時刻項目

出力フィールドに対して生成されます。

- 日付／時刻項目のデータ名：マップ名-FIELDnnnn-O

データ名は，日付／時刻フィールドダイアログの「データ名」で変更できます。

- 繰り返し回数：縦，または横の反復回数
- 長さ：論理項目の長さ。指定がないときはフィールドのデータ長
- データ型：文字，数字編集項目のどちらかを選択

データ型	使用できる項目
99999	数字編集項目 COBOL で桁数に満たない値を代入すると，COBOL の仕様によって「0（ゼロ）」が設定されます。
文字（XX）	文字

注

桁数に満たない値を代入した残りにデータ有無コードを指定している場合は，「桁寄せ：左」「埋字：スペース」で埋字されます。そのほかのマッピング規則については，「16.4.9(3)　日付／時刻項目のマッピング規則」を参照してください。

(2)　日付／時刻項目の出力論理マップ生成規則

日付／時刻の出力論理マップ生成規則について説明します。

COBOL

COBOL の日付／時刻の生成規則は，データ型によって変わります。

・文字項目の場合

```
03 マップ名-FIELDnnnn-O PIC X(長さ)〔OCCURS 回数〕.
```

・数字編集項目の場合

```
03 マップ名-FIELDnnnn-O PIC 編集文字〔OCCURS 回数〕.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_O　〔[回数]〕[長さ];
```

C 言語使用時は，数字編集項目は指定できません。

(3)　日付／時刻項目のマッピング規則

日付／時刻の論理マップ生成規則について説明します。

《AP が出力項目データ名の領域に代入した内容と結果》

AP が出力項目データ名の領域に代入した内容		結果
すべてデータの場合	すべて空白（$(20)_{16}$）	項目全体を空白で印字します。
	すべて 0（$(30)_{16}$）	0 をデータとしてフォーマットに合わせて印字します。
	すべて NULL（$(00)_{16}$）	項目全体を空白で印字します。
	上記以外のデータ	代入されたデータをフォーマットに合わせて印字します。
先頭 1 文字がデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として，X'1F414234'または X'1F1F1F'のようなデータが該当します。このときは，データを印字しません。
データの後半にデータ有無コード		データ有無コード$(1F)_{16}$を仮定した場合 例として， X'4142431F1F'のようなデータが該当します。この場合，データ有無コード以前のデータに対応するデータを，フォーマットに合わせて印字します。

16.4.10　下位項目

(1)　下位項目の定義

下位項目は主にテキスト，フィールドおよびラベルに対して，データ型が「文字（XX)」のときに指定します。各オブジェクトの定義ダイアログで設定すると，生成されます。

(2)　下位項目の出力論理マップ生成規則とマッピング規則

下位項目の出力論理マップ生成規則について説明します。

(a)　出力論理マップ生成規則

COBOL

・数字項目の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC　9(長さ).
```

・文字項目の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC　X(長さ).
```

・1 文字の反復の場合

```
04 マップ名-FIELDnnnn-nnn-O PIC　X(1) OCCURS 回数.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_nnn_O 〔［回数］〕［長さ］;
```

(b)　マッピング規則

出力結果

上位項目単位にマッピングします。下位項目を二つ定義し，二つ目の下位項目にデータ有無コード以外のデータを設定し，一つ目の下位項目にデータ有無コードを設定しても，埋字の対象にはなりません。

(3)　下位項目の入力論理マップ生成規則とマッピング規則

下位項目の入力論理マップ生成規則について説明します。

(a)　入力論理マップ生成規則

COBOL

・数字項目の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  9(長さ).
```

・文字項目の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  X(長さ).
```

・1 文字の反復の場合

```
04 マップ名-FIELDnnnn-nnn-I PIC  X(1) OCCURS 回数.
```

C 言語

```
unsigned char マップ名_FIELDnnnn_nnn_I 〔［回数］〕［長さ］;
```

(b)　マッピング規則

入力結果

上位項目単位にマッピングします。

16.4.11　フレーム

フレームをドローで定義し，フレーム中に可変項目を定義することで，生成されます。

- フレームのデータ名：マップ名-FRAMEnnnn-O
  データ名はフレームダイアログの「データ名」で変更できます。
- フレームの反復
  方向：縦，または横
  回数：反復回数
  間隔：反復間隔をます目単位で指定

オブジェクトに対応するデータ項目をまとめて集団項目にします。

COBOL

```
{02|03} マップ名-FRAMEnnnn-{O|I} 〔OCCURS n〕.
   {04|05}マップ名……オブジェクトに対応する項目
```

C 言語

```
struct {
   オブジェクトに対応する項目
       …} マップ名_FRAMEnnnn_{O|I} 〔[n]〕;
```

16.4.12　予約フィールド

ドローで定義した予約項目名と印刷結果に関するマッピング規則について説明します。なお，論理マップには，予約フィールドの情報は生成されません。

定義	結果
OpenTP1 がサポートしている予約項目名を指定	ドロー定義で指定した項目長が OpenTP1 で定められた項目長と等しい場合 OpenTP1 が設定するデータが印刷されます。 ドロー定義で指定した項目長が OpenTP1 で定められた項目長より大きい場合 データが左寄せで印刷されます。残りには空白が埋められます。 ドロー定義で指定した項目長が OpenTP1 で定められた項目長より小さい場合 データが左寄せで印刷されます。余りは切り捨てられます。
OpenTP1 がサポートしていない予約項目名を指定	項目には何も印刷されません。

16.5　定数部

定数部の論理マップ生成規則について説明します。

16.5.1　定数部の論理マップ生成規則

(1)　定義部の定義

- 出力論理マップ長の出力

  ドローセットアップの論理マップ属性ダイアログ中の展開方式の「定数部への論理マップ長出力」を指定したときに生成されます。

- カーソル定数の出力

  カーソル定数は無条件に生成されます。

(2)　論理マップ生成規則

(a)　出力論理マップ長の出力の指定

出力論理マップの論理マップ生成例を示します。また，ここで説明する出力論理マップ長とは，定数部だけの論理マップ長です。

COBOL

・Windows リトルエンディアン用

```
01 マップ名T  PIC S9(4)  COMP-5  VALUE ＋長さ.
```

・Windows ビッグエンディアン用，UNIX 用

＜論理マップ長が 10,000 バイト未満の場合＞

```
01 マップ名T  PIC  S9(4)  COMP  VALUE ＋長さ.
```

＜論理マップ長が 10,000 バイト以上の場合＞

```
01 マップ名U  PIC  X(2)  VALUE X'長さ'.
01 マップ名T  REDEFINES  マップ名U  PIC  S9(4)  COMP.
```

C 言語

・Windows リトルエンディアン用，UNIX 用

```
short  マップ名T=長さ;
```

・Windows ビッグエンディアン用

```
unsigned char  マップ名T[2]={0x長さ};
```

(b)　カーソル定数の出力の指定

カーソル定数を指定したときの論理マップ生成例を示します。

・繰り返し項目がある場合

COBOL

・Windows 用

```
01 マップ名D.
  02 項目データ名D.
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
       :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

・UNIX 用

＜カーソル定数の値が 10,000 未満の場合＞

```
01 マップ名D.
  02 項目データ名D.
    03 FILLER  PIC S9(4) COMP VALUE +nnnn.
          :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

＜カーソル定数の値が 10,000 以上の場合＞

```
01 マップ名D.
  02 項目データ名D.
          :
    03 FILLER  PIC X(2) VALUE X'nnnn'.
          :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

C 言語

```
struct {
    unsigned  char  項目データ名T  [回数][2];
      :
} マップ名D={0xnn,0xnn〔,…〕};
```

・繰り返し項目がない場合

COBOL

・Windows 用

```
01 マップ名D.
  02 項目データ名T.
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
          :
```

・UNIX 用

＜カーソル定数の値が 10,000 未満の場合＞

```
01 マップ名D.
  02 項目データ名T.
    03 FILLER  PIC S9(4) COMP VALUE +nnnn.
          :
```

＜カーソル定数の値が 10,000 以上の場合＞

```
01 マップ名D.
  02 項目データ名T.
          :
    03 FILLER  PIC X(2) VALUE X'nnnn'.
          :
```

C 言語

```
struct {
    unsigned  char  項目データ名T [2];
      :
} マップ名D={0xnn,0xnn};
```

16.5.2　イベント定数の論理マップ生成規則

イベント定数の論理マップ生成規則について説明します。

(1)　イベント定数の定義

イベント定数は，画面属性ダイアログの「入力単位」で「イベント」を指定すると生成されます。ドローセットアップの「イベント通知コード」ダイアログの，「通知コードの長さ」が「4」のときだけ指定できます。

(2)　論理マップ生成規則

COBOL

```
01 マップ名D.
  02 項目データ名V.
    03 FILLER  PIC X(4)  VALUE X'nnnnnnnn'.
          :
```

C言語

```
struct {
    unsigned  char  項目データ名V [4];
          :
} マップ名D={0xnn,0xnn,0xnn,0xnn 〔,…〕 };
```

16.5.3　フォーカス定数の論理マップ生成規則

フォーカス定数の論理マップ生成規則について説明します。

(1)　フォーカス定数の定義

フォーカス定数は，無条件に生成されます。

(2)　フォーカス定数の論理マップ生成例

COBOL

・繰り返し項目がある場合

Windows用

```
01 マップ名D.
  02 項目データ名D.
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
          :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

UNIX用

＜フォーカス定数の値が10,000未満の場合＞

```
01 マップ名D.
  02 項目データ名D.
    03 FILLER  PIC S9(4) COMP VALUE +nnnn.
          :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

＜フォーカス定数の値が10,000以上の場合＞

```
01 マップ名D.
  02 項目データ名D.
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
          :
  02項目データ名T  REDEFINES  項目データ名D  OCCURS 回数.
```

・繰り返し項目がない場合

Windows用

```
01 マップ名D.
  02 項目データ名T.
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
          :
```

UNIX用

＜フォーカス定数の値が10,000未満の場合＞

```
01 マップ名D.
  02 項目データ名T.
    03 FILLER  PIC S9(4) COMP VALUE +nnnn.
          :
```

＜フォーカス定数の値が 10,000 以上の場合＞

```
01 マップ名D.
  02 項目データ名T.
    03 FILLER  PIC X(2) VALUE X'nnnn'.
          :
```

・プッシュボタンボックスの場合

```
01 マップ名D.
  02 ボックス名T.   ……※
    03 FILLER  PIC X(2)  VALUE X'nnnn'.
          :
```

注※

プッシュボタンボックスダイアログで定義したボックス名称が展開されます。

C 言語

・繰り返し項目がある場合

```
struct {
    unsigned  char  項目データ名T  〔[回数]〕[2];
          :
} マップ名D={0xnn,0xnn〔,…〕};
```

・繰り返し項目がない場合

```
struct {
    unsigned  char  項目データ名T [2];
          :
} マップ名D={0xnn,0xnn};
```

・プッシュボタンボックスの場合

```
struct {
    unsigned  char  ボックス名T [2]. ……※
          :
} マップ名D={0xnn, 0xnn};
```

注※

プッシュボタンボックスダイアログで定義したボックス名称が展開されます。

16.5.4　動的変更テーブル

実行時に色などの属性を AP から変更するときには，XMAP3 が用意する属性変更用の定数テーブルの修飾名を表示属性の動的変更制御項目に代入します。この属性変更用の定数テーブルを動的変更テーブルといいます。動的変更テーブルは，COBOL 用，C 言語用の二つを用意しています。

17 チューニングとトラブルの対処方法

この章では，XMAP3 実行時の性能を向上させるためのポイント，およびトラブル発生時の対処方法について説明します。

17.1　性能向上のポイント

ここでは，AP 実行の性能向上のためのテクニックについて説明します。

17.1.1　定義時のポイント

作成する画面や帳票は，必要最小限のオブジェクトにし，マップを小さくすることで実行性能が向上します。マップを小さくするには，次のことに注意してください。

(1)　けい線の見直し

- むだなけい線は削除する。
- つながるけい線は 1 本で定義する。
- 矩形で囲んだフィールドは反転表示にして枠を取る。

(2)　テキストボックスやフィールドの見直し

(a)　テキストなどの集約

テキストボックスやフィールドは，まとめて定義してください。例えば，「年」「月」「日」は三つに分けるのではなく，一つにまとめることでマップが小さくなります。

(b)　フィールドボックスの利用（1）

テキストボックスで作成されている表は，フィールドボックスのフィールドとして定義してください。表示性能が向上します。

(c)　フィールドボックスの利用（2）

GUI 画面でテキストボックスが多く繰り返されている場合には，フィールドボックス中にフレームを配置して，その中のフィールドとして定義してください。フィールドボックスのフレーム機能を使用すると，1 フィールドの定義で複数フィールドに相当する定義ができます。また，論理マップ上では OCCURS 展開されるため，AP のロジックも繰り返し指定で実現でき，効率が良くなります。

(3)　網掛けの見直し

- むだな網掛けは削除する。
- 網掛けの種類を減らす。

(4)　グラフィックデータの見直し

(a)　不要なグラフィックデータの削除

グラフィックデータは，データ量が多くマップが大きくなってしまいます。グラフィックデータは，必要な個所にだけ使用するようにしてください。

(b)　グラフィックデータの縮小化

固定グラフィックの場合はファイルサイズの総和が 31,000 バイト以内に収まるように作成してください。出力グラフィックの場合は 100KB 以内を目安にして作成することをお勧めします。容量の大きいグラフィックデータを小さくするには，次の方法で対処できます。

ビットマップの縦横ドット数を小さくする

ビットマップは，中身の図柄がどのようなものであっても，同じ縦横サイズならばファイルサイズとして，常に同じバイト数を必要とします。したがって，縦横ドット数を小さくすることで，ファイルサイズが小さくなります。実際に絵を描いていない部分もビットマップの情報として格納されますので，必要最小限の領域を取るようお勧めします。

次に既存のビットマップの必要な部分だけをペイント（またはペイントブラシ）を用いて切り出す方法を説明します。

1. ペイント（またはペイントブラシ）で基になるビットマップを表示する。
2. 四角形選択ツールで（または自動選択ツール）で必要な部分を選択する。
3. メニューバーから［編集］の［ファイルへコピー］を選択する。
4. ファイル名を指定して保存する。

使用する色を減らす

ビットマップの形式として，2 色，16 色，256 色，1,600 万色（フルカラー）の色数のものがあります。色数が増えると情報量が増えるので，それだけビットマップサイズが大きくなります。また，その表示色数をサポートしていないディスプレイに表示した場合，色化けが起きたり，表示が遅くなったりするなどの問題が起こります。

既存のビットマップの色数を変更するには，画像フォーマットを変換する機能を持った市販のグラフィックソフトを使用する方法があります。しかし，色数を減らすことでビットマップが見にくくなることがあります。したがって，初めから 16 色でビットマップを作成することをお勧めします。

(5)　プレーンサイズの見直し

ボックスに隠れているプレーンは，データがなくても表示しています。不要なサイズになっていないか見直してください。

(6)　画面サイズの見直し

画面サイズが小さい方が表示性能は高くなります。画面サイズを見直してください。

17.1.2　AP 作成時のポイント

(1)　同一マップを再表示するときの方法の見直し

同一マップの画面を再表示する場合は，一部上書を指定しないと，画面中の一つ（または一部）のオブジェクトだけを書き換えるときでも全画面の再表示となり，表示性能が低下します。

一部上書と全面書換とを使い分けることをお勧めします。特に一部上書は，フィールド単位にデータを入力し，AP でチェックおよび表示する場合に効果的です。指定方法については「11.1.1(1)　通信記述項」を参照してください。

- 前提となる定義

1. ドローで，画面属性の「入力・選択状態の扱い」に「未入力・未選択」または「状態を維持」を指定します。また，該当するテキスト・フィールドの属性の「初期値」に「なし」を指定します。
2. 1.の指定によって，データ未入力を埋字以外で判定する必要が生じた場合には，ドローセットアップの論理マップ属性ダイアログで，「初期クリア文字」に「埋字」以外を指定します。さらに，支援ツールのセットアップ情報の反映で，マップを再生成します。

1 回目の表示時

- プログラム中のマッピングオプションに，「マージ」を指定します。
- TRANSCEIVE 文，SEND 文などで画面を表示します。

同じ画面の再表示時

1. プログラム中のマッピングオプションに，「論理マップだけ」を指定します。
2. 画面表示時に変更しない項目にデータ有無コードを格納します。

- 動的変更属性の指定があれば，プログラム中の制御項目（修飾名を格納する論理マップのエリア）に格納します。
- 項目データは，先頭 1 バイト目，またはすべての項目データに対して格納します。
- 出力論理マップがデータ有無コードでクリアされるため，表示中のデータを記憶しておく場合には別のエリアに画面情報をバックアップしておくような対処が必要です。

1. 画面表示時に変更する項目だけにデータを格納します。
2. 画面の表示形態を指定する制御項目（マップ名-CNTRLO）に，動的変更用の定数テーブル中の「XMAP-CNTRL1（標準の場合)」を格納します。
3. TRANSCEIVE 文，SEND 文などで 1 回目と同じマップ名を再表示します。

このコーディングのひな型として，AP パターン CRLINP01.CBL を利用できます。

(2)　データ量の見直し

AP でむだなデータを送っていないか，見直してください。例えば，常にスペースやゼロを送ってしまうと実行性能が劣ってしまいます。したがって，出力時の性能を上げるにはデータの量を減らすことが重要です。次の 2 点に注意してください。

- データ有無コードで論理マップをクリアする
- むだな初期値は指定しない

データ有無コードについては，「16.　論理マップ生成規則とマッピング規則」を参照してください。

(3)　AP 分割の見直し

XMAP3 のオープン要求がどのタイミングで出されているかを見直してください。コンパイル単位で，TRANSCEIVE 文または，SEND 文が発行されると，AP ごとにオープンが実行されるため，処理が遅くなります。不要にオープンを要求しないためには次の点に注意してください。

- 一つのコンパイル単位に TRANSCEIVE 文，SEND 文をまとめ，オープンの実行は必要最低限にします。
- 共通のルーチンにまとめて TRANSCEIVE 文，SEND 文を発行するように AP の構造を見直します。
- COBOL の場合，AP 間でオープンを引き継ぐ「CBLTERMSHAR=YES」を指定すると，AP 間でオープンの引き継ぎが行われます。

詳細については「3.1.2　AP 間でオープンを引き継ぐ場合」を参照してください。

COBOL では，明示的に CALL 文でオープンを要求する場合を除き，一つのコンパイル単位で最初の SEND 文または TRANSCEIVE 文が発行されると，オープンが要求されます。そのため，1 画面が 1 実行ファイル（.EXE）のような構成にすると，実行ファイルごとにオープンを要求することになります。この場合，オープン・クローズ，SEND 文，または TRANSCEIVE 文を発行する実行ファイルとビジネス処理をする実行ファイルを分けた方が，実行性能が高くなります。

また，複数のコンパイル単位のものを合わせて一つの実行ファイルにするときは，各コンパイル単位でオープンを発行しないようにするため，COBOL の実行支援の環境変数で，「CBLTERMSHAR=YES」を指定します。「CBLTERMSHAR=YES」は，SEND／RECEIVE／TRANSCEIVE 文で AP を作成したときだけ有効です。CALL 文で AP を作成したときは無効になります。

AP 間のオープン引き継ぎたいときは，XMAP3 インタフェースエリアの情報を引き継ぐようなコーディングをする必要があります。画面用 AP のコーディングのひな型として，AP パターンの GENDSP02 および GENDSP03 が利用できます。

(4)　グラフィックデータの渡し方の見直し

クリップボード中のグラフィックを指定（「*CLIP」指定）すると，AP でファイルを指定して呼び出すよりも処理が早くなります。ただし，指定できるファイルは，一つだけのため注意が必要です。

(5)　画面クローズのタイミングの見直し

画面のクローズは，明示的に CALL 文でクローズするか，DISABLE 文を発行したタイミングです。ただし，クローズすると次に画面を表示するときには必ずオープンしなければなりません。画面を素早く消したり，再表示したりする場合は，画面属性ダイアログの「Z 位置」で「一時非表示」を指定して，見た目上は表示されていないようにしておき，次に画面を表示するときに動的変更で「Z 位置」の指定を「標準表示」または「手前に表示」に変更します。このようにすると，画面をクローズ，オープンするよりも早く画面を切り替えられます。

(6)　画面用 AP での制御のやり取りの見直し

画面属性の定義で入力単位を「画面」にしているにもかかわらず，オブジェクトの属性で「自動送信」を指定して，オブジェクト単位で AP に制御を返すような AP 構造は，画面単位に AP に制御を返す AP 構造と比べ オーバーヘッドが大きくなり，オブジェクト遷移（フォーカス移動）が遅くなる場合があります。このため，AP との制御のやり取りは画面単位で行うことをお勧めします。オブジェクト単位で AP と制御のやり取りをする場合は，入力単位を「フィールド」や「イベント」に設定し，できるだけ画面中のオブジェクトの数を少なくすることがポイントです。

17.1.3　実行時のポイント

(1)　マップの常駐化サイズの見直し

使用する物理マップを常駐するバッファサイズは変更できます。詳細については，マニュアル「XMAP3 開発ガイド」を参照してください。

(2)　スプール書き出し単位の見直し

AP から出力するデータを，1 ページごとに印字するか，AP の終了またはクローズするごとに印字するかを選べます。複数の帳票や書式を出力する場合は，表示・印刷セットアップの「スプール書き出し単位」の設定を見直してください。設定については，マニュアル「XMAP3 実行ガイド」の帳票環境のセットアップに関する記述を参照してください。

(3)　プリンタドライバの出力タイミングの設定

プリンタドライバの設定で，Windows スプーラの設定，および印字開始タイミングの設定ができます。印字開始のタイミングは，すぐに印刷を開始するか，データがすべてスプールされてから印刷を開始するかを選べます。

(4)　GDI か PDL スルーかの見直し

デフォルトは GDI です。LIPS または ESC/P スルーを利用すると，データ量が少なくなり，AP 実行時および印刷時の性能が向上します。

(5)　C/S 構成時の通信データ削減

C/S 構成の場合，XMAP3 の通信データを圧縮する機能が利用できます。通信データの圧縮機能を利用すると，C/S 間の通信量が削減できるため，画面応答のレスポンス向上が期待できます。詳細については，マニュアル「XMAP3 実行ガイド」を参照してください。

(6)　PDF ファイルの圧縮

XMAP3 が出力する PDF ファイルを圧縮できます。出力する PDF ファイルを小さくすることで，実行性能が向上します。

17.2　トラブルの対処方法

ここでは，COBOL 使用時，画面の表示時，および帳票の印刷時のトラブルの対処方法について説明します。

17.2.1　COBOL 使用時のトラブル

(1)　コンパイル時の主なエラー

表 17–1　コンパイル時の主なエラー

エラー	意味	対処方法
KCCC1076C-S	COPY 文を取り込もうとしているファイルが, AP のソースが格納されているフォルダ中にないことが考えられる。	ファイルが，AP のソースが格納されているフォルダ中にあるか見直し, ない場合はファイルをコピーしてコンパイルし直す。
	COPY 文を取り込もうとしているファイルの名称が誤っていることが考えられる。	XMAP3 で作成した論理マップファイルの名称であるか見直し, ファイルの名称が誤っているときは正しい名称に修正してコンパイルし直す。
KCCC3015C-S	データ名, または変数名に不当な文字を指定していることが考えられる。	データ名，および変数名を見直す。
	必要なテーブル「X3MODTBL」が AP を格納しているフォルダにないことが考えられる。	AP を格納しているフォルダに次のファイルをコピーして，コンパイルし直す。 *XMAP3 インストールフォルダ*¥INCLUDE¥X3MODTBL
Unsatisfied Symbols: jsvwadrv(code)	ccbl2002 コマンドによるコンパイルで XMAP3 Server Runtime のライブラリ設定（-lxpdrv）をしていないことが考えられる。	-lxpdrv および-lxpw オプションを指定してコンパイルし直す。
Line n：KCCB1099C-U ソースファイルの形式に誤りがあります	ccbl2002 コマンドによるコンパイル時に使用する論理マップおよび動的変更テーブルが使用環境に合っていないことが考えられる。	EUC 環境で使用する場合は，EUC へコード変換したかを見直す。

(2)　リンク（MAKE）時の主なエラー

表 17–2　リンク（MAKE）時の主なエラー

エラー	意味	対処方法
LNK1181 LNK1104	TRANSCEIVE 文, SEND 文，および RECEIVE 文で XMAP3 を使用する場合に，必要な XMAP3 のライブラリを参照できない。	コンパイル環境のマシンに XMAP3 の開発モデルがインストールされているか確認する（CALL 文で作成した AP の場合はリンケージ環境）。
LNK2001 LNK1120	CALL 文で XMAP3 を使用する場合に，リンケージ時に必	COBOL のリンケージオプションとして，次の手順で設定する。

エラー	意味	対処方法
	要な XMAP3 のライブラリを参照できない。	1. COBOL 開発マネージャの［プロジェクト］－［プロジェクトの設定］コマンドを選ぶ。 2.「リンク」タブの「ライブラリの指定」にチェックを入れる。 表示されるダイアログの設定内容に，次の内容を追加する。 *XMAP3 インストールフォルダ*¥LIB¥X3MWDR32.LIB

付録

付録 A　標準提供動的変更テーブル

ここでは，XMAP3 が標準で提供する動的変更テーブルの値について説明します。

付録 A.1　画面の表示属性

画面の表示属性を AP から動的に変更する場合，ドローセットアップの表示属性の動的変更ダイアログで定義した修飾名を使用します。表示属性の動的変更ダイアログは定義するオブジェクトの系統別に，次のように分けられています。

- キャラクタコントロール
- 候補選択コントロール
- コマンドコントロール
- ウィンドウ属性
- 位置属性
- 確定キー属性
- データ名

ここでは，XMAP3 が標準に提供する修飾名について説明します。

(1)　キャラクタコントロールの表示属性の標準値（GUI／CUI 共通）

キャラクタコントロール（入出力テキスト・フィールド・ポップアップ・コンボボックス，ポップアップテキスト・フィールド）の表示属性の標準値を次に示します。これらの変更属性は，使用目的別に設定値を変更する必要があります。

また，使用目的が出力専用のときは，設定値が異なります。出力専用のときの，表示属性の標準値について次に示します。

表 A-1　キャラクタコントロールの表示属性の標準値（数字系・日本語系・カナ系・英数系キーエントリおよび日本語系選択エントリ）（1／3）

動的変更テーブルのデータ名	修飾名	表示方法・入力可否属性	表示方法・非表示属性	遷移条件	詳細目的※2（モジュラスチェック）
XMAP-IN-ATTR1	ER	入力可能	標準表示※1	変更なし	変更なし
XMAP-IN-ATTR2	RV	入力可能	標準表示※1	変更なし	変更なし
XMAP-IN-ATTR3	PT	入力不可	標準表示※1	変更なし	指定できない
XMAP-IN-ATTR4	NP	入力可能	標準表示※1	変更なし	変更なし
XMAP-IN-BK	BK※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-GR	GR※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-BL	BL※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-LR	LR※3	変更なし	変更なし	指定できない	指定できない

動的変更テーブルのデータ名	修飾名	表示方法・入力可否属性	表示方法・非表示属性	遷移条件	詳細目的※2（モジュラスチェック）
XMAP-IN-LB	LB※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-LG	LG※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-LY	LY※3	変更なし	変更なし	指定できない	指定できない
XMAP-IN-IV	IV※3	入力不可	全体非表示	指定できない	指定できない
XMAP-IN-GO	GO※3	入力不可	グレーアウト	指定できない	指定できない

注※1

非表示にしません。

注※2

「詳細目的」は適用オブジェクトグループが「数字系キーエントリ」の場合だけ有効です。

注※3

定義対象が CUI 画面の場合，これらの修飾名は「設定する修飾名」として選択されていません。選択した場合，修飾名の属性はすべて「変更なし」となります。

表 A-2　キャラクタコントロールの表示属性の標準値（数字系・日本語系・カナ系・英数系キーエントリおよび日本語系選択エントリ）（2／3）

動的変更テーブルのデータ名	修飾名	反転表示	背景色※1	文字色	入力済み
XMAP-IN-ATTR1	ER	変更なし	変更なし	赤	変更なし
XMAP-IN-ATTR2	RV	反転する	変更なし	黒	変更なし
XMAP-IN-ATTR3	PT	変更なし	変更なし	黒	指定できない
XMAP-IN-ATTR4	NP	変更なし	変更なし	黒	変更なし
XMAP-IN-BK	BK※2	変更なし	変更なし	黒	指定できない
XMAP-IN-GR	GR※2	変更なし	変更なし	緑	指定できない
XMAP-IN-BL	BL※2	変更なし	変更なし	青	指定できない
XMAP-IN-LR	LR※2	変更なし	ライトレッド	変更なし	指定できない
XMAP-IN-LB	LB※2	変更なし	ライトブルー	変更なし	指定できない
XMAP-IN-LG	LG※2	変更なし	ライトグリーン	変更なし	指定できない
XMAP-IN-LY	LY※2	変更なし	ライトイエロー	変更なし	指定できない
XMAP-IN-IV	IV※2	変更なし	変更なし	変更なし	指定できない
XMAP-IN-GO	GO※2	変更なし	変更なし	変更なし	指定できない

注※1

CUI 画面では指定できません。

注※2

定義対象が CUI 画面の場合，これらの修飾名は「設定する修飾名」として選択されていません。選択した場合，修飾名の属性はすべて「変更なし」となります。

表 A-3　キャラクタコントロールの表示属性の標準値（数字系・日本語系・カナ系・英数系キーエントリおよび日本語系選択エントリ）（3／3）

動的変更テーブルのデータ名	修飾名	自動送信	入力必須	ワンタッチクリア※1	印字
XMAP-IN-ATTR1	ER	変更なし	変更なし	変更なし	変更なし
XMAP-IN-ATTR2	RV	変更なし	変更なし	変更なし	変更なし
XMAP-IN-ATTR3	PT	指定できない	指定できない	指定できない	変更なし
XMAP-IN-ATTR4	NP	変更なし	変更なし	変更なし	変更なし
XMAP-IN-BK	BK※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-GR	GR※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-BL	BL※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-LR	LR※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-LB	LB※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-LG	LG※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-LY	LY※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-IV	IV※2	指定できない	指定できない	指定できない	変更なし
XMAP-IN-GO	GO※2	指定できない	指定できない	指定できない	変更なし

注※1

CUI 画面では指定できません。

注※2

定義対象が CUI 画面の場合，これらの修飾名は「設定する修飾名」として選択されていません。選択した場合，修飾名の属性はすべて「変更なし」となります。

表 A-4　キャラクタコントロールの表示属性の標準値（ラベル）

動的変更テーブルのデータ名	修飾名	表示方法・非表示属性	反転表示	文字色	印字
XMAP-OUT-ATTR1	RE	標準表示※1	変更なし	赤	変更なし
XMAP-OUT-ATTR2	GR	標準表示※1	変更なし	緑	変更なし
XMAP-OUT-ATTR3	BL	標準表示※1	変更なし	青	変更なし
XMAP-OUT-ATTR4	YE	標準表示※1	変更なし	黄	変更なし
XMAP-OUT-ATTR5	CY	標準表示※1	変更なし	空	変更なし
XMAP-OUT-ATTR6	MA	標準表示※1	変更なし	紫	変更なし

動的変更テーブルのデータ名	修飾名	表示方法・非表示属性	反転表示	文字色	印字
XMAP-OUT-ATTR7	WH	標準表示※1	変更なし	白	変更なし
XMAP-OUT-BK	BK※2	変更なし	変更なし	黒	変更なし
XMAP-OUT-DR	DR※2	変更なし	変更なし	ダークレッド	変更なし
XMAP-OUT-DB	DB※2	変更なし	変更なし	ダークブルー	変更なし
XMAP-OUT-DG	DG※2	変更なし	変更なし	ダークグリーン	変更なし
XMAP-OUT-DY	DY※2	変更なし	変更なし	ダークイエロー	変更なし
XMAP-OUT-IV	IV※2	全体非表示	変更なし	変更なし	変更なし
XMAP-OUT-GO	GO※2	グレーアウト表示	変更なし	変更なし	変更なし

注※1

非表示にしません。

注※2

定義対象が CUI 画面の場合，これらの修飾名は「設定する修飾名」として選択されていません。選択した場合，修飾名の属性はすべて「変更なし」となります。

(2)　候補選択コントロールの表示属性の標準値（GUI）

ラジオボタン，チェックボタン，リスト，トグルフィールドの表示属性の標準値を次の表に示します。

表 A-5　候補選択コントロールの表示属性の標準値

動的変更テーブルのデータ名	修飾名	表示方法（リスト項目以外）	印字（トグルのみ）	不活性（選択できない状態にする）	フォーカス設定（フォーカスをボタン／リストに位置づける）※	選択済みにする
XMAP-BUTTON-SEL1	PT	標準表示	変更なし	不活性	設定しない	選択前
XMAP-BUTTON-SEL2	ON	標準表示	変更なし	活性	設定しない	選択済み
XMAP-BUTTON-SEL3	OF	標準表示	変更なし	活性	設定しない	選択前
XMAP-BUTTON-SEL4	ST	標準表示	変更なし	活性	設定する	選択前
XMAP-BUTTON-SEL5	IV	全体非表示	変更なし	指定できない	設定しない	選択前

注※

トグルフィールド以外です。

(3)　コマンドコントロールの表示属性の標準値（GUI）

プッシュボタン，メニューバーの表示属性の標準値を次の表に示します。

表 A-6　コマンドコントロールの表示属性の標準値

動的変更テーブルのデータ名	修飾名	表示方法（プッシュボタンだけ）	不活性（選択できない状態にする）	フォーカス設定（フォーカスをボタンに位置づける）
XMAP-BUTTON-SEL1	PT	標準表示	不活性	設定しない
XMAP-BUTTON-SEL2	ON	標準表示	活性	設定しない
XMAP-BUTTON-SEL3	OF	標準表示	活性	設定しない
XMAP-BUTTON-SEL4	ST	標準表示	活性	設定する
XMAP-BUTTON-SEL5	IV	全体非表示	指定できない	設定する

(4)　ウィンドウ属性の表示属性の標準値（GUI／CUI 共通）

プッシュボタン，メニューバーの表示属性の標準値を次の表に示します。

表 A-7　ウィンドウ属性の表示属性の標準値

動的変更テーブルのデータ名	修飾名	表示形態	入力・選択状態の扱い	キーボードのロック状態を解除する	ウィンドウを表示した時にアラームを鳴らす
XMAP-CNTRL1	WR	一部上書	状態を維持	解除する	鳴らさない
XMAP-CNTRL2	EW	全面書換	初期状態	解除する	鳴らさない
XMAP-CNTRL3	CL	一部上書	初期状態	解除する	鳴らさない
XMAP-CNTRL4	BL	一部上書	状態を維持	解除する	鳴らす

(5)　位置属性の表示属性の標準値（GUI）

表 A-8　位置属性の表示属性の標準値

動的変更テーブルのデータ名	修飾名	XY 位置	Z 位置
XMAP-WINDOW-RT	RT	右上	標準表示
XMAP-WINDOW-RD	RD	右下	標準表示
XMAP-WINDOW-LT	LT	左上	標準表示
XMAP-WINDOW-LD	LD	左下	標準表示
XMAP-WINDOW-IV	IV	変更なし	一時非表示
XMAP-WINDOW-FR	FR	中央	手前に表示
XMAP-WINDOW-VC	VC	中央	標準表示

(6)　確定キー属性の表示属性の標準値（GUI）

表 A–9　確定キー属性の表示属性の標準値

動的変更テーブルのデータ名	修飾名	キー押下無効にするキー
XMAP-EVENT-AL	AL	なし
XMAP-EVENT-PC	PC	PF13～PF84，PA1～PA3，スクリーン消去

(7)　データ名の表示属性の標準値（GUI／CUI 共通）

表 A–10　データ名の表示属性の標準値

属性種別	データ名	論理マップ中でのマップ名
ウィンドウ属性の動的変更	マップ名－CNTRL	マップ名－CNTRLO
位置属性の動的変更	マップ名－WINDOW	マップ名－WINDOWW
確定キー属性の動的変更のデータ名	マップ名－INC	マップ名－INCO

付録 A.2　帳票の表示属性

帳票の表示属性を AP から動的に変更する場合，ドローセットアップの次に示す機能で定義した修飾名を使用します。

- フィールド表示属性の動的変更
- けい線表示属性の動的変更

ここでは，XMAP3 が標準に提供する修飾名について説明します。

(1)　フィールド表示属性の動的変更

フィールド表示属性の動的変更の修飾名を次の表に示します。

表 A–11　フィールド表示属性の動的変更

動的変更テーブルのデータ名	修飾名	表示属性
XMAP-FIELD-BL	BL	強調（太字）
XMAP-FIELD-SH	SH	網掛け
XMAP-FIELD-CR	CR	抹消線
XMAP-FIELD-RE	RE	赤色表示

(2)　けい線表示属性の動的変更

けい線表示属性の動的変更の修飾名を次の表に示します。

表 A–12　けい線表示属性の動的変更

動的変更テーブルのデータ名	修飾名	表示属性
XMAP-LINE-DB	DB	二重線

動的変更テーブルのデータ名	修飾名	表示属性
XMAP-LINE-IV	IV	非表示（線を引かない）
XMAP-LINE-SL	SL	実線（細線）
XMAP-LINE-DS	DS	破線（細線）
XMAP-LINE-DT	DT	点線（細線）

付録 B　書式オーバレイの 1 ページデータ量の制限

ここでは，書式オーバレイでの 1 ページのデータ量の制限を開発言語別に説明します。

ページデータの上限値を超えるデータが AP から指定された場合，XMAP3 が改ページを行い，上限値を超えるデータは次のページに出力する動作となります。

付録 B.1　COBOL の場合

概算式で使用する用語について説明します。

全角／半角切り替わり回数

WRITE 文で出力する行データ項目ごとに，半角文字から全角文字に切り替わる回数，および全角文字から半角文字に切り替わる回数を数えます。このとき，全角空白は半角空白 2 バイトに置き換えて数えます。

例えば，行データが「123 あい　う」の場合，「3」のあと，「い」のあとおよび，「う」の前で切り替わりが発生するので，切り替わり回数は 3 回となります。

文字サイズ／字間値／書体／横倍の切り替わり回数

POINT／INTERVAL／FORMAT／WIDE の属性ごとに属性の切り替わり回数を数えます。属性値が同じ値の場合は，切り替わりが発生しません。属性の指定が省略されている場合は，デフォルト（ドローの書式定義で指定した値）に切り替わるものとして数えます。

例えば，次に示す例の場合，文字サイズ切り替わり回数は 2 回，字間値切り替わり回数は 2 回となります。

```
03  DATA1   PIC  X(5).                        …切り替わりなし
03  DATA2   PIC  X(5)  POINT-9 INTERVAL-0.    …文字サイズ9ポ，字間値0ポに切り替わる
03  DATA3   PIC  X(5)  POINT-9 INTERVAL-0.    …切り替わりなし
03  DATA4   PIC  X(5)  POINT-9.               …字間値がデフォルトに切り替わる
03  DATA5   PIC  X(5).                        …文字サイズがデフォルトに切り替わる
03  DATA6   PIC  X(5).                        …切り替わりなし
```

(1)　「WRITE 行データ FROM データ 1」形式の処理で，CHARACTER TYPE 句指定なしの場合

COBOL 記述例

定義部

```
01 行データ.
  02 項目Ａ PIC X(10).
  02 項目Ｂ PIC X(10).

01 データ１.
  02 項目１ PIC X(10).
  02 項目２ PIC X(5).
  02 項目３ PIC X(5).
```

処理部

```
WRITE 行データ FROM データ１.
```

概算式

1 ページデータ量の上限

印刷ドキュメント名を指定していない場合

32767≧行データの和※＋ 62

印刷ドキュメント名を指定している場合

32767≧行データの和※＋ 印刷ドキュメント名の長さ ＋ 70

注※

行データの和＝｛行データの領域長＋（データ中の平均全角／半角切り替わり回数×2）＋ 10｝×行数

(2) 「WRITE 行データ FROM データ 1」形式の処理で，データ 1 に CHARACTER TYPE 句指定ありの場合

COBOL 記述例

定義部

```
01 行データ.
  02 項目Ａ PIC X(10).
  02 項目Ｂ PIC X(10).

01 データ１.
  02 項目１ PIC X(10) CHARACTER TYPE POINT-9 INTERVAL-0.
  02 項目２ PIC X(5).
  02 項目３ PIC X(5).
```

処理部

```
WRITE 行データ FROM データ１.
```

概算式

1 ページデータ量の上限

印刷ドキュメント名を指定していない場合

32767≧行データの和※＋ 62

印刷ドキュメント名を指定している場合

32767≧行データの和※＋ 印刷ドキュメント名の長さ ＋ 70

注※

行データの和＝｛データ 1 の各項目の長さの和
＋（データ 1 の各項目内での平均全角／半角切り替わり回数の和×2）
＋（データ 1 の各項目への平均文字サイズ切り替わり回数の和×3）
＋（データ 1 の各項目への平均字間値切り替わり回数の和×3）
＋（データ 1 の各項目への平均書体切り替わり回数の和×3）
＋（データ 1 の各項目への平均横倍切り替わり回数の和×4）
＋（データ 1 の項目の数×2）
＋ 8｝×行数

(3) 「WRITE 行データ」形式の処理で，CHARACTER TYPE 句指定なしの場合

COBOL 記述例

定義部

```
01 行データ.
  02 項目１ PIC X(10).
  02 項目２ PIC X(5).
  02 項目３ PIC X(5).
```

処理部

```
WRITE 行データ.
```

概算式

1 ページデータ量の上限

印刷ドキュメント名を指定していない場合

32767≧行データの和※＋ 62

印刷ドキュメント名を指定している場合

32767≧行データの和※＋ 印刷ドキュメント名の長さ ＋ 70

注※

行データの和＝｛行データの領域長＋（データ中の平均全角／半角切り替わり回数×2）＋ 10｝×行数

(4) 「WRITE 行データ」形式の処理で，CHARACTER TYPE 句指定ありの場合

COBOL 記述例

定義部

```
01 行データ.
  02 項目１ PIC X(10) CHARACTER TYPE POINT-9 INTERVAL-0.
  02 項目２ PIC X(5).
  02 項目３ PIC X(5).
```

処理部

```
WRITE 行データ.
```

概算式

1 ページデータ量の上限

印刷ドキュメント名を指定していない場合

32767≧行データの和※＋ 62

印刷ドキュメント名を指定している場合

32767≧行データの和※＋ 印刷ドキュメント名の長さ ＋ 70

注※

行データの和＝｛行データの各項目の長さの和
＋（行データの各項目内での平均全角／半角切り替わり回数の和×2）
＋（行データの各項目への平均文字サイズ指定回数の和×3）
＋（行データの各項目への平均字間値切り替わり回数の和×3）
＋（行データの各項目への平均書体切り替わり回数の和×3）
＋（行データの各項目への平均横倍切り替わり回数の和×4）
＋（行データの項目の数×2）
＋ 8｝×行数

付録 B.2　C 言語の場合

概算式で使用する用語について説明します。

全角／半角切り替わり回数

jstqldat 関数，または XmapFrmSetData 関数で指定する項目データごとに，半角文字から全角文字に切り替わる回数，および全角文字から半角文字に切り替わる回数を数えます。このとき，全角空白は半角空白 2 バイトに置き換えて数えます。

jstqllpt 関数，または XmapFrmSetLine 関数を発行するまでを一つの行データと考えます。

例えば，行データが「123 あい　う」の場合，「3」の後ろ，「い」の後ろ，「う」の前で切り替わりが発生するので，切り替わり回数は 3 回となります。

文字サイズ／字間値／書体／横倍の切り替わり回数

jstqlctp 関数，XmapFrmSetPoint 関数，XmapFrmSetInterval 関数，XmapFrmSetFont 関数，または XmapFrmSetWidth 関数で指定した属性の数をそれぞれ数えます。

概算式

1 ページデータ量の上限

- **印刷ドキュメント名を指定していない場合**

  32767≧行データの和※＋ 62

- **印刷ドキュメント名を指定している場合**

  32767≧行データの和※＋ 印刷ドキュメント名の長さ ＋ 70

注※

行データの和＝｛行データの各項目データの長さの和
＋（行データの各項目データ内での平均全角／半角切り替わり回数の和×2）
＋（行データの各項目データへの平均文字サイズ指定回数の和×3）
＋（行データの各項目データへの平均字間値指定回数の和×3）
＋（行データの各項目データへの平均書体指定回数の和×3）
＋（行データの各項目データへの平均横倍指定回数の和×4）
＋（行データの各項目データ内での平均項目データ数×2）
＋ 8｝×行数

付録 C　AP パターンの一覧と使用例

XMAP3 では，COBOL と C 言語用の AP のひな型として，AP パターンと AP 部品を提供しています。これらを使用することで，AP を効率良く作成できます。また，コーディングの統一を図れます。AP パターンと AP 部品について次に示します。

- AP パターン

  COBOL の場合

  COBOL ソースプログラムの見出し部，データ部，手続き部など，プログラム全体の標準的な骨組みが記述されていますので，メインプログラムとして利用できます。

  C 言語の場合

  C 言語ソースプログラムの標準的な骨組みが記述されていますので，メインプログラムとして利用できます。

- AP 部品

  AP パターン共通に，頻繁に使用すると考えられる処理の手続きが記述されていますので，サブルーチンとして利用できます。

AP パターンと AP 部品は次のフォルダに格納されています。

Windows の場合

- COBOL

  XMAP3 インストールフォルダ¥PATTERNS¥COBOL

- C 言語

  XMAP3 インストールフォルダ¥PATTERNS¥C

UNIX の場合

/opt/HIXMAP/patterns

ここでは，XMAP3 で提供している AP パターンの一覧を COBOL と C 言語別に示します。

COBOL

COBOL 用の AP パターンの一覧を次に示します。

表 C-1　COBOL 用の AP パターン一覧

ファイル名	処理の概要	XMAP3 Developer	UNIX 版 XMAP3 Server Runtime
Atrcrs01.cbl	エラー項目の色の変更，およびカーソルの位置づけ	○	○
Btmenu01.cbl Btmenu01.cet	TRANSCEIVE 文を使用したメニュー画面の表示	○	○
		○	×
Btprot01.cbl	ボタン属性の変更（ボタンの不活性）	○	×
Clrinp01.cbl	データエントリ画面の入力項目のクリア	○	○

ファイル名	処理の概要	XMAP3 Developer	UNIX 版 XMAP3 Server Runtime
Dspprt01.cbl Dspprt01.cet	TRANSCEIVE 文を使用した画面表示と SEND 文を使用した帳票印刷	○	○
		○	×
Fldhlp01.cbl	ヘルプ画面の表示	○	○
gendsp01.cbl gendsp01.cet	汎用画面入出力	○	○
		○	×
Gendsp02.cbl Gendsp02.cet	AP 間のオープン引継ぎ	○	○
		○	×
Gendsp03.cbl Gendsp03.cet	AP 間のオープン引継ぎ	○	○
		○	×
GENEVN01.cbl GENEVN01.cet	GUI 画面用のイベント単位入力	○	×
		○	×
Genfld01.cbl	GUI 画面用のフィールド単位入力	○	×
Genfld01.cet		○	×
genovl01.cbl genovl01.cet	WRITE 文を使用した書式付き帳票印刷	○	×
		○	×
genrep01.cbl genrep01.cet	同一帳票の複数枚印刷	○	○
		○	×
Genrep02.cbl Genrep02.cet	CALL 文を使用した帳票印刷	○	×
		○	×
Genrep03.cbl Genrep03.cet	汎用帳票出力（追加帳票）	○	×
		○	×
mcfdp01.cbl	OLTP サーバ構成での RECEIVE 文と SEND 文を使用した画面表示と帳票印刷	○	×
Modatr01.cbl	フィールド属性変更	○	○
Modatr02.cbl	フィールド属性の変更，およびけい線種別の動的変更	○	×
Nxtdsp01.cbl	次画面の入出力	○	○
Nxtevn01.cbl	次画面の入出力およびイベント単位入力	○	×
Nxtfld01.cbl	次画面の入出力およびフィールド単位入力	○	×
Nxtrep01.cbl	標準帳票の出力	○	○
Patwrt01.cbl	同一画面に対し，1 項目だけ書き換える処理	○	○

ファイル名	処理の概要	XMAP3 Developer	UNIX 版 XMAP3 Server Runtime
Setcrs01.cbl	フォーカス・カーソルの変更	○	○
Slmenu01.cbl	任意のプログラム（EXE）をメニュー画面から CALL 文で呼び出す	○	○
Slmenu01.cet		○	×

（凡例）

○：ファイルを提供している。

×：ファイルを提供していない。

C 言語

C 言語用の AP パターンの一覧を次に示します。

表 C-2　C 言語用の AP パターン一覧

ファイル名	処理の概要	XMAP3 Developer	UNIX 版 XMAP3 Server Runtime
Atrcrs01.c	エラー項目の色の変更，およびカーソルの位置づけ	○	×
Btmenu01.c	jsvwadrv 関数を使用した画面表示	○	×
Btprot01.c	ボタン属性の変更（ボタンの不活性）	○	×
Clrinp01.c	データエントリ画面の入力項目のクリア	○	×
Dspprt01.c	jsvwadrv 関数を使用した画面表示と帳票印刷	○	×
Fldhlp01.c	ヘルプ画面表示の呼び出し	○	×
GENDSP01.C	汎用画面入出力	○	○
GENEVN01.c	GUI 画面イベント単位入力 AP	○	×
GENFLD01.c	GUI フィールド単位入力	○	×
Genovl01.c	書式オーバレイ用の API（jstq～）を使用した書式付き帳票印刷	○	×
GENREP01.C	帳票用	○	○
mcfdp01.c	TP1/NET/XMAP3 の API（dc_mcf_～）を使用した画面表示と帳票印刷	○	×
Modatr01.c	項目属性の変更	○	×
Modatr02.c	フィールド属性の変更，およびけい線種別の動的変更	○	×
Nxtdsp01.c	次画面の入出力	○	×
Nxtevn01.c	次画面の入出力およびイベント単位入力	○	×
Nxtfld01.c	次画面の入出力およびフィールド単位入力	○	×

ファイル名	処理の概要	XMAP3 Developer	UNIX 版 XMAP3 Server Runtime
nxtrep01.c	標準帳票の出力	○	×
Patwrt01.c	同一画面に対し，1 項目だけ書き換える処理	○	×
Setcrs01.c	フォーカス・カーソルの変更	○	×
Slmenu01.c	任意のプログラム（EXE）をメニュー画面から呼び出す	○	×

（凡例）

○：ファイルを提供している。

×：ファイルを提供していない。

AP パターンの使用例

ここでは，XMAP3 で提供している COBOL 用 AP パターンの使用例を説明します。使用例を説明する AP パターンの一覧を次の表に示します。

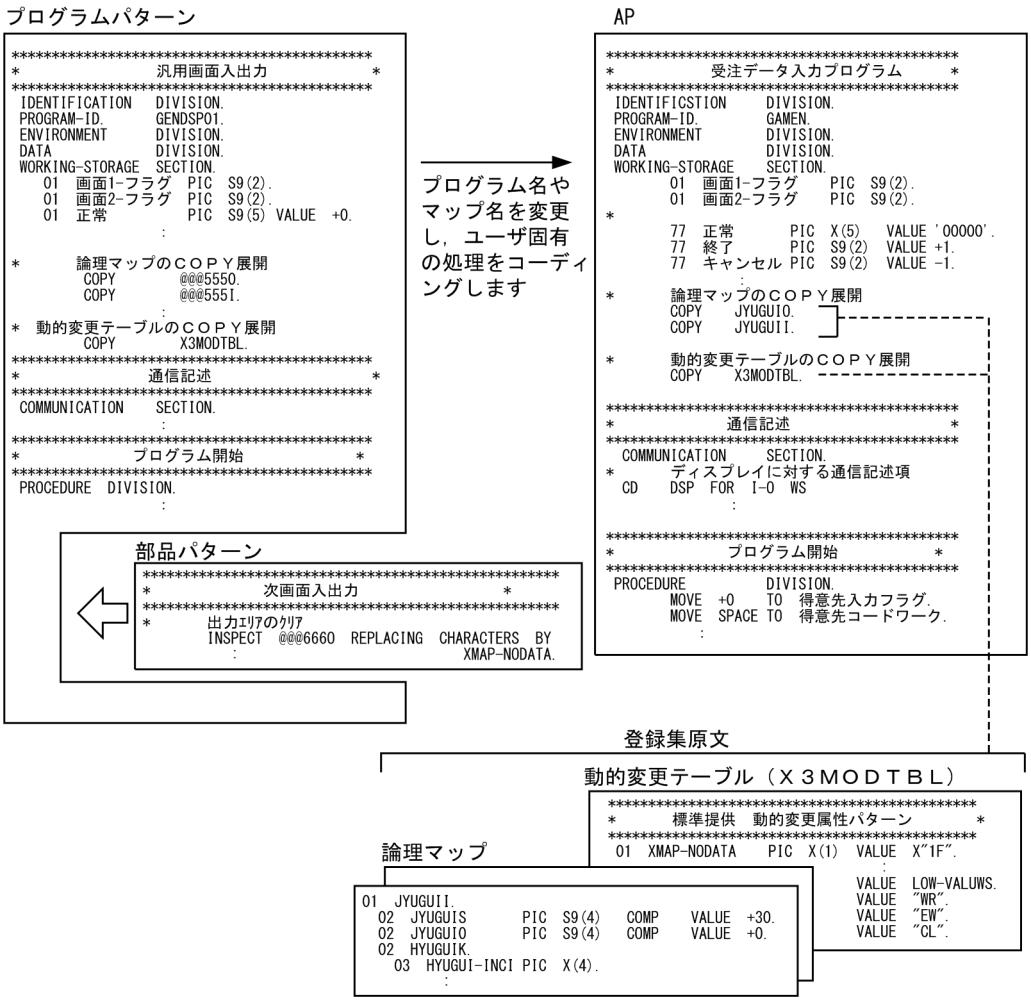
表 C-3　使用例を説明する AP パターンの一覧

パターン名	使用目的	処理の概要	例題の説明個所
BTMENU01	メニュー画面の表示	TRANSCEIVE 文を使用したメニュー画面の表示	付録 C.2
DSPPRT01	画面表示と帳票印刷	TRANSCEIVE 文を使用した画面表示と SEND 文を使用した帳票印刷	付録 C.3
GENREP02	同一帳票の複数枚印刷	CALL 文を使用した帳票印刷	付録 C.4
GENOVL01	書式付き帳票印刷	WRITE 文を使用した書式付き帳票印刷	付録 C.5
MCFDP01	OLTP 環境での画面表示と帳票印刷	RECEIVE 文と SEND 文を使用した画面表示と帳票印刷	付録 C.6

付録 C.1　AP パターンを利用した AP の作成手順

プログラムパターンと部品パターンを編集したり，コーディングしたりする作業は通常，エディタを使用します。AP パターンの利用イメージを次の図に示し，AP の作成手順を説明します。

図 C-1　AP パターンの利用イメージ



1. 画面や帳票のレイアウト定義とあわせて，プログラムの仕様を決定します。

2. プログラムパターンを決定します。

   プログラムの仕様を基に，コーディングに利用するプログラムパターンを選びます。プログラムパターンを参考にしながら，処理の流れ，データ入出力の流れを詳細に検討します。

3. 部品パターンを組み込みます。

   処理内容に合わせて利用する部品パターンを決めます。2.で検討した処理の流れに従って部品パターンを組み込み，コーディングのベースを作ります。

4. コーディングします。

   プログラム名や，マップ名，ファイルなどを定義し，業務固有の手続きをコーディングします。このとき，コンパイルで取り込む論理マップや動的変更テーブルを COPY 文に指定します。

   コーディングを終えたソースプログラムは，コンパイルしたあと，実行してテスト・デバッグができるようになります。

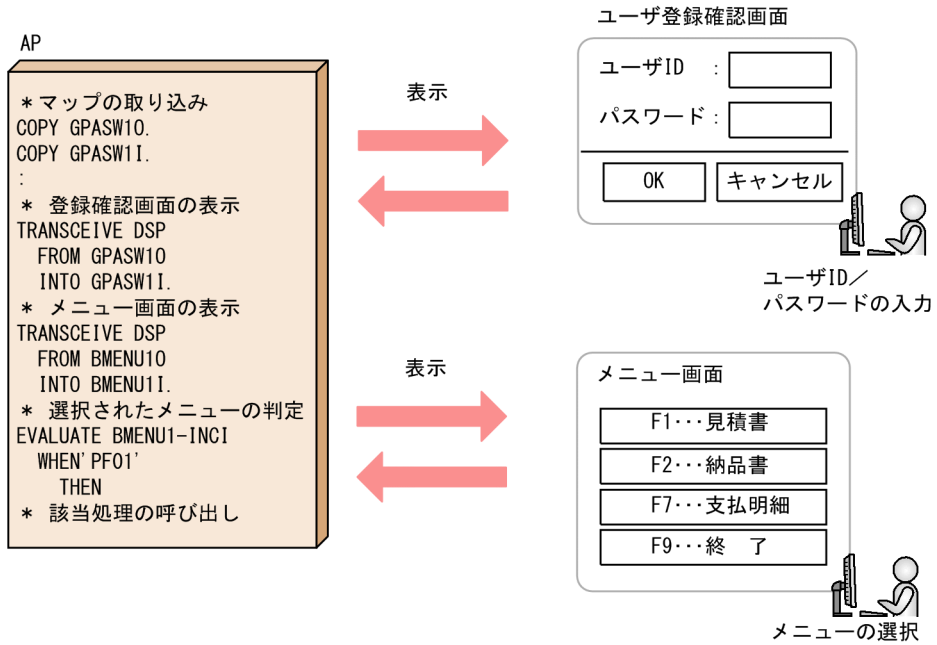
付録 C.2　BTMENU01 パターンを使用したメニュー画面の表示例

(1)　処理の概要

ユーザ ID とパスワードを入力するためのユーザ登録確認画面を表示します。ユーザ ID とパスワードを照合して登録があれば，次にメニュー画面を表示します。このメニュー画面で，プッシュボタンまたはファンクションキーを使用して次の処理を選ぶと，選ばれたメニューに対応した使用ごとの AP を起動します。

メニュー画面の表示処理の概要を次の図に示します。

図 C-2　メニュー画面の表示処理



(2)　画面定義上のポイント

ユーザ登録の確認画面には次の要素を定義することを推奨します。

- パスワードを入力する項目の使用目的を「パスワード」と指定すると，パスワード入力時は入力データを「＊」や「空白」で隠せる。
- ユーザ ID またはパスワードに誤りがあった場合に利用できるエラーメッセージ用の項目を用意している。

(3)　AP 作成上のポイント

作成する AP の基になる AP パターンを次に示します。

- メインプログラムで使用する AP パターン名：BTMENU01

メインプログラムには，次の特長があります。

- 対応しないファンクションキーや［Enter］が押されたときの処理が用意されている。
- メニュー画面からは，［F1…見積書］ボタンまたは［F1］を選んだときの業務の AP を CALL 文で呼び出す仕様となっている。
- メニュー画面から業務ごとの AP を呼び出すとき，メニュー画面を DISABLE 文でクローズしている。

なお，この AP パターンを使用してのメインプログラム作成時には次の注意が必要です。

- ユーザ ID およびパスワードの照合処理は含まれていないため，ユーザ独自のコーディングが必要となる。

(4)　コーディング例

コーディング上のポイントを次に示します。

```
WORKING-STORAGE               SECTION.
* 論理マップのCOPY展開
* ユーザ登録確認画面用
   COPY    GPASW10.           ……… 1.
   COPY    GPASW1I.
*メニュー画面用
   COPY    BMENU10.           ……… 1.
   COPY    BMENU1I.
   COPY    X3MODTBL.
   01 EXEC-NAME1.
      02 FILLER    PIC X(12)  ……… 2.
                  VALUE 'DSPCLEAR.exe'
PROCEDURE          DIVISION.
* マップ名の設定
         MOVE 'GPASW1ND' TO 画面マップ名
* ユーザ登録確認画面の表示
         TRANSCEIVE DSP FROM GPASW10 INTO GPASW1I.
:
* メニュー画面の表示
* マップ名の設定
         MOVE 'BMENU1ND' TO 画面マップ名.
* 画面の表示と入力
         TRANSCEIVE DSP FROM BMENU10 INTO BMENU1I.
* ボタン・ファンクションキーのチェックと該当プログラムの呼び出し
         EVALUATE BMENU1-INCI
           WHEN    'PF01'
             DISABLE DSP
             MOVE '0' TO PROC-IND
             CALL 'CBLEXEC' USING      EXEC-NAME-LEN…… 3.
                                       EXEC-NAME1 EXEC-PARM
         END-EVALUATE.
```

1. マップ名を変更する。

   プログラム中の'@@@111'を'GPASW1'に，'@@@444'を'BMENU1'に変更します。

2. 実行ファイル名を変更する。

   プログラム中の'###EXE01.exe'を'DSPCLEAR.exe'に変更します。

3. CALL 文を使用して AP を呼び出す。

   CALL 'DSPCLEAR.exe'

付録 C.3　DSPPRT01 パターンを使用した画面表示と帳票印刷

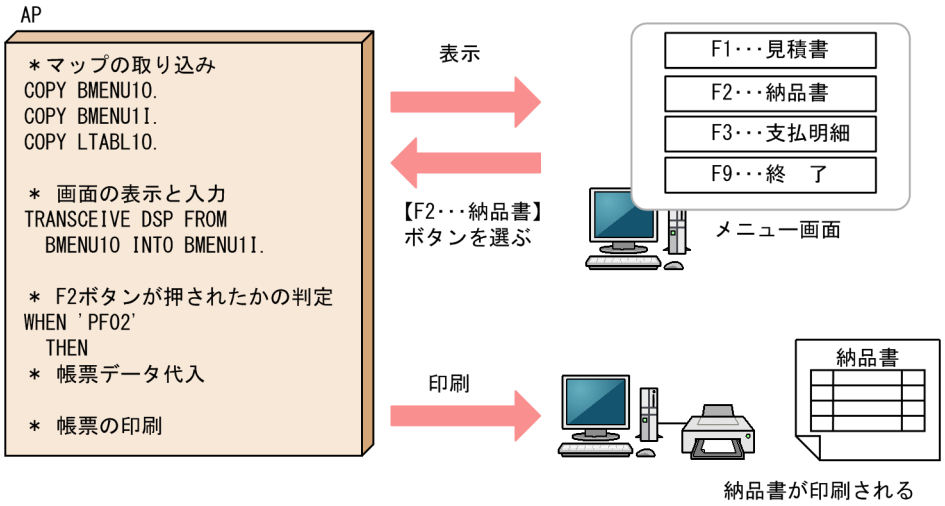
一つの AP で画面の表示と帳票の印刷をする業務を想定した例を説明します。

(1)　処理の概要

各帳票を印刷するためのメニュー画面を表示します。メニューから［F2...納品書］のプッシュボタンを選ぶと，納品書の帳票を印刷します。このとき，帳票はシリアルインパクトプリンタへ印刷されます。

メニュー画面の表示と帳票印刷の処理の概要を次の図に示します。

図 C-3　メニュー画面の表示と帳票印刷



(2)　AP 作成上のポイント

作成する AP の基になる AP パターンを次に示します。

- メインプログラムで使用する AP パターン：DSPPRT01

この AP では，［F2...納品書］ボタンが押されたときの処理だけを記述しています。「納品書」を 1 ページ印刷すると，プログラムが終了します。

この AP 作成時には，次の点に注意してください。

- ［F2...納品書］以外のボタンが押されたときの処理を追加する。
- 印刷データの代入処理を追加する。
- メニュー画面でのエラーチェックを必要に応じて追加する。

(3)　コーディング例

コーディング上のポイントを次に示します。

```
WORKING-STORAGE                 SECTION.
＊ 論理マップのCOPY展開
＊ メニュー画面用
  COPY    BMENU1O.  ･･･････････････････････ 1.
  COPY    BMENU1I.
＊ ダイアログボックス用
  COPY    LTABL1O.  ･･･････････････････････ 1.
PROCEDURE                       DIVISION.
＊ マップ名の設定
MOVE 'BMENU1ND'     TO 画面マップ名
＊ 画面の表示と入力
TRANSCEIVE DSP FROM BMENU1O INTO BMENU1I
＊ ボタンの判定と該当処理の呼び出し
EVALUATE BMENU1-INCI  ･･･････････････････ 2.
  WHEN 'PF02'
         PERFORM 帳票処理
  WHEN 'PF09'
         MOVE キャンセル TO 画面1-フラグ
END-EVALUATE
    :
帳票処理                        SECTION.
  PERFORM 出力データ代入. ･･･････････････････ 3.
  SEND PRT FORM LTABL1O WITH EMI.
```

```
出力データ代入                    SECTION.
  MOVE ’D0012345’                 TO LTABL1-SHEETNO-0.
  MOVE ’○×A1家電’                 TO LTABL1-CUSTOMER-0.
  MOVE ’K0012345’                 TO LTABL1-USERNO-0.
出力データ代入-END.
```

1. マップ名を変更する。
   プログラム中の'@@@888'を BMENU1 に変更し，'¥¥¥222'を'LTABL1'に変更します。
2. ファンクションキーの判定処理。
3. 印刷データの代入処理。

付録 C.4　GENREP02 パターンを使用した同一帳票の複数枚印刷

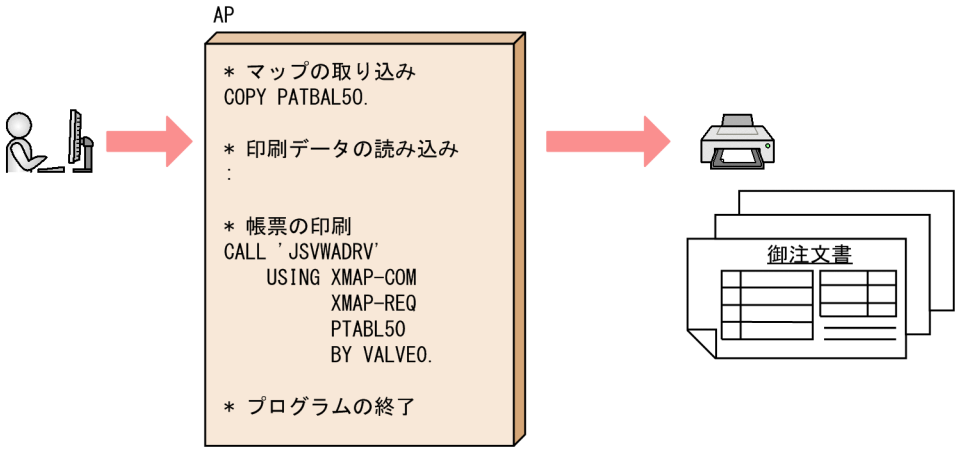
COBOL の CALL 文を使用して，1 種類の帳票を一つの AP から複数枚印刷する方法について説明します。

(1)　処理の概要

注文書 3 ページ分のデータを読み込み，帳票を無条件に 3 部印刷します。

注文書を印刷するときの処理概要を次の図に示します。

図 C-4　注文書の印刷



(2)　AP 作成上のポイント

作成する AP の基になる AP パターンを次に示します。

- メインプログラムで使用する AP パターン名：GENREP02
  この AP パターンでは，帳票に出力するデータは固定データを代入して，無条件に帳票を 1 部印刷する仕様になっています。ファイルからデータを読み込む仕様にする場合は，別にコーディングを追加する必要があります。

このほかに，AP 作成時には次の注意が必要です。

- 出力先がページプリンタ　網掛け帳票であるため，物理マップ名は「マップ名 6B」で指定します。
- 印刷データがない場合に出力論理マップをクリアする処理を追加します。
- 印刷処理の繰り返し方法，および印刷を終了するタイミングを指定します。

(3)　コーディング例

コーディング上のポイントを次に示します。

```
 WORKING-STORAGE      SECTION.
 77 回数    PIC 9(2).
* 論理マップのCOPY展開
  COPY    PTABL50.                                  ……1.
  PROCEDURE       DIVISION.
  MOVE 0 TO 回数.
  :
* マップ名の設定
  MOVE 'PTABL56B' TO 帳票マップ名.                  ……1.
* 出力データ代入処理，帳票印刷処理を３回呼び出す
 PERFORM VARYING 回数 FROM 1 BY 1 UNTIL 回数 > 3
         PERFORM 出力データ代入                     ……2.
         CALL 'jsvwadrv' USING XMAP-COM XMAP-REQ
         PTABL50 BY VALUE0.                        ……3.
 END-PERFORM.
* 出力データ代入処理
 出力データ代入 SECTION.
  MOVE    回数          TO PTABL5-NO-0.
  MOVE    '○×A1電器'    TO PTABL5-KOKYAKU-0.
  MOVE    'K0012345'    TO PTABL5-KCODE-0.
 出力データ代入-END.
```

1. マップ名の変更

   プログラム中の'¥¥¥111'を'PTABL5'に，'XX'を'6B'に変更します。

2. 出力データの代入

3. 帳票の印刷

付録 C.5　GENOVL01 パターンを使用した書式付き帳票の複数枚印刷

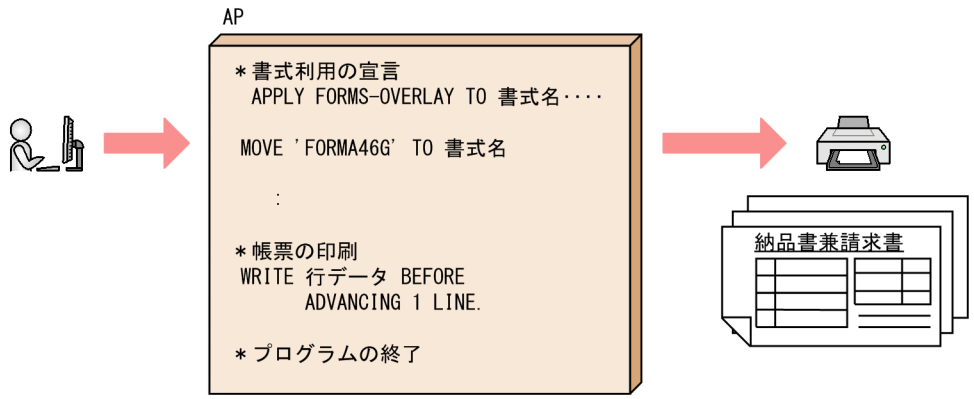
COBOL の WRITE 文を使用して，書式オーバレイで帳票を複数枚印刷するときの方法を説明します。

(1)　処理の概要

書式付き帳票を 3 部印刷します。納品書兼請求書の書式に 3 ページ分の出力データを重ねてページプリンタに印刷します。

書式付き帳票を複数枚印刷する処理の概要を次の図に示します。

図 C-5　書式付き帳票の複数枚印刷



(2)　AP 作成上のポイント

作成する AP の基になる AP パターンを次に示します。

- メインプログラムで使用する AP パターン名：GENOVL01

このプログラムでは，帳票に出力するデータは固定データを代入して，帳票を無条件に 3 部印刷する仕様になっています。

AP 作成時には，ヘッダ，ボディ（明細），トレイラ部分を印刷するタイミングを考慮する必要があります。また，ファイルからデータを読み込む仕様にする場合は，ファイルの入出力処理を追加してください。

(3)　コーディング例

コーディング上のポイントを次に示します。

```
*
 ENVIRONMENT        DIVISION.
*
 INPUT-OUTPUT       SECTION.
 FILE-CONTROL.
   SELECT  プリンタ  ASSIGN TO PRT001……プリンタの宣言
           ORGANIZATION IS SEQUENTIAL
           FILE STATUS IS ファイル状態
*
   SELECT テキストファイル ASSIGN TO ファイル名
           ORGANIZATION IS LINE SEQUENTIAL
           FILE STATUS IS ファイル状態
************************************************************
*        書式印刷の定義                                    *
************************************************************
 I-O-CONTROL
   APPLY FORMS-OVERLAY TO 書式名 ON プリンタ.      …書式利用の宣言

 DATA            DIVISION.
 FILE            SECTION.
* 一行分のデータエリア
  01 行データ    PIC X(122).
 WORKING-STORAGE SECTION.
* 印刷データの明細行数
  77 明細行数    PIC S9(4) VALUE 25.  …書式に合わせた行データと明細行数を設定
 PROCEDURE       DIVISION.
 印刷処理
  IF プリンタ状態 = 正常
         THEN
* 1ページ分の印刷処理（３回繰り返し）          ……帳票を繰り返し印刷する処理
                PERFORM VARYING 回数 FROM 1 BY 1
                        UNTIL 回数 > 3
                        PERFORM 1ページ印刷
                END-PERFORM
  END-IF.
  1ページ印刷   SECTION.

* 印刷するオーバレイ名の設定
  MOVE 'FORMA46G' TO 書式名.         …書式名を変更（書式名を'FORMA46G'に変更）

 ヘッダ印刷        SECTION.
  MOVE SPACE      TO 番号行.         …出力するデータを代入する処理
  MOVE 回数       TO 番号行-番号.
  MOVE 行番号     TO 行データ.
  WRITE 行データ   TO BEFORE ADVANCING 3 LINE.
```

付録 C.6　MCFDP01 パターンを使用した画面表示と帳票印刷

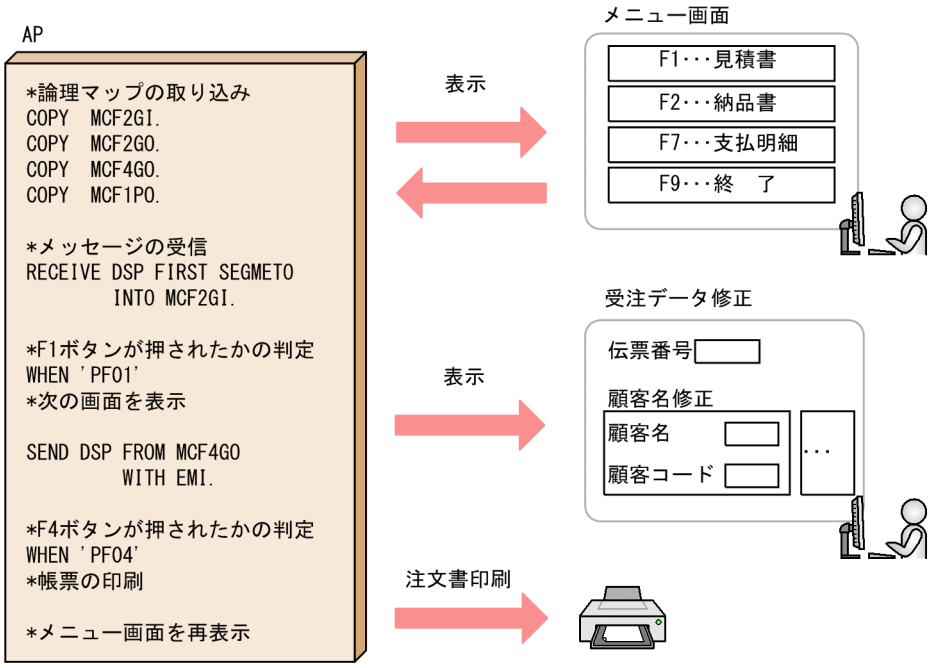
(1)　処理の概要

利用者名入力画面でログインしたあと，業務を選択するためのメニュー画面を表示します。この画面で各メニューのプッシュボタンを選ぶと，選ばれたメニューに対応する業務の画面を表示します。

なお，［注文書印刷］プッシュボタンを選ぶと納品書が印刷されます。

メニュー画面から次画面（この例では「受注データ修正画面」）を表示する処理の概要を次の図に示します。

図 C-6　メニュー画面からの次画面表示処理



(2)　AP 作成上のポイント

作成する AP の基になる AP パターンを次に示します。

- メインプログラムで使用する AP パターン名：MCFDP01

この AP パターンを使用した AP 作成時には次の注意が必要です。

- ［F1：受注データ修正］ボタン，［F4：注文書印刷］ボタンが押されたときの処理（［F1：受注データ修正］ボタンを押したときは次の画面を表示し，［F4：注文書印刷］ボタンを押したときは注文書を印刷したあと，メニュー画面を再表示する）を追加する。
- 次の画面，帳票に出力するデータの代入処理を追加する。
- エラー処理を必要に応じて追加する。

(3)　コーディング例

コーディング上のポイントを次に示します。

```
IDENTIFICATION          DIVISION.
PROGRAM-ID.             MCF2G.              ・・・・・・・・・・1.
        :
```

```
   WORKING-STORAGE         SECTION.
     COPY                  MCF2GI.           ・・・・・・・・・・・2.
     COPY                  MCF2GO.           ・・・・・・・・・・・2.
     COPY                  MCF3GO.           ・・・・・・・・・・・2.
     COPY                  MCF1PO.           ・・・・・・・・・・・2.
           :
   PROCEDURE               DIVISION.
*  メッセージの受信
     RECEIVE  DSP  FIRST  SEGMENT  INTO  MCF2GI.
           :
     EVALUATE  MCF2G-INCI                    ・・・・・・・・・・・3.
     WHEN  'PF01'
*      次の画面を表示します
       MOVE  'MCF3G'      TO  DSP-MAP-NAME
       SEND  DSP  FROM  MCF3GO  WITH  EMI
           :
     WHEN  'PF04'
*      帳票を印刷したあと，受信した画面を再表示します
*      帳票を印刷します
       MOVE  'XPNLE201'    TO  PRT-TERM       ・・・・・・・・・・・4.
       MOVE  'MCF1P'       TO  PRT-MAP-NAME
       SEND  PRT  FROM  MCF1PO  WITH  EMI
*      画面を再表示します
       MOVE  'MCF2G'      TO  DSP-MAP-NAME
       SEND  DSP  FROM  MCF2GO  WITH  EMI
     WHEN  OTHER
*      受信した画面を再表示します
       MOVE  'MCF2G'      TO  DSP-MAP-NAME
       SEND  DSP  FROM  MCF2GO  WITH  EMI
     END-EVALUATE
*  プログラムの終了
   EXIT  PROGRAM.
```

1. 入り口名を変更する。

   プログラム中の'@@@111'を'MCF2G'に変更します。

2. マップ名を変更する。

   プログラム中の次の項目をそれぞれ変更します。

   '@@@222'を'MCF2G'に変更

   '@@@333'を'MCF3G'に変更

   '@@@444'を'MCF1P'に変更

3. ボタンの判定処理。

4. 論理端末名を変更する。

   プログラム中の'@@@@555'を'XPNLE201'（プリンタに割り当てた論理端末名）に変更します。

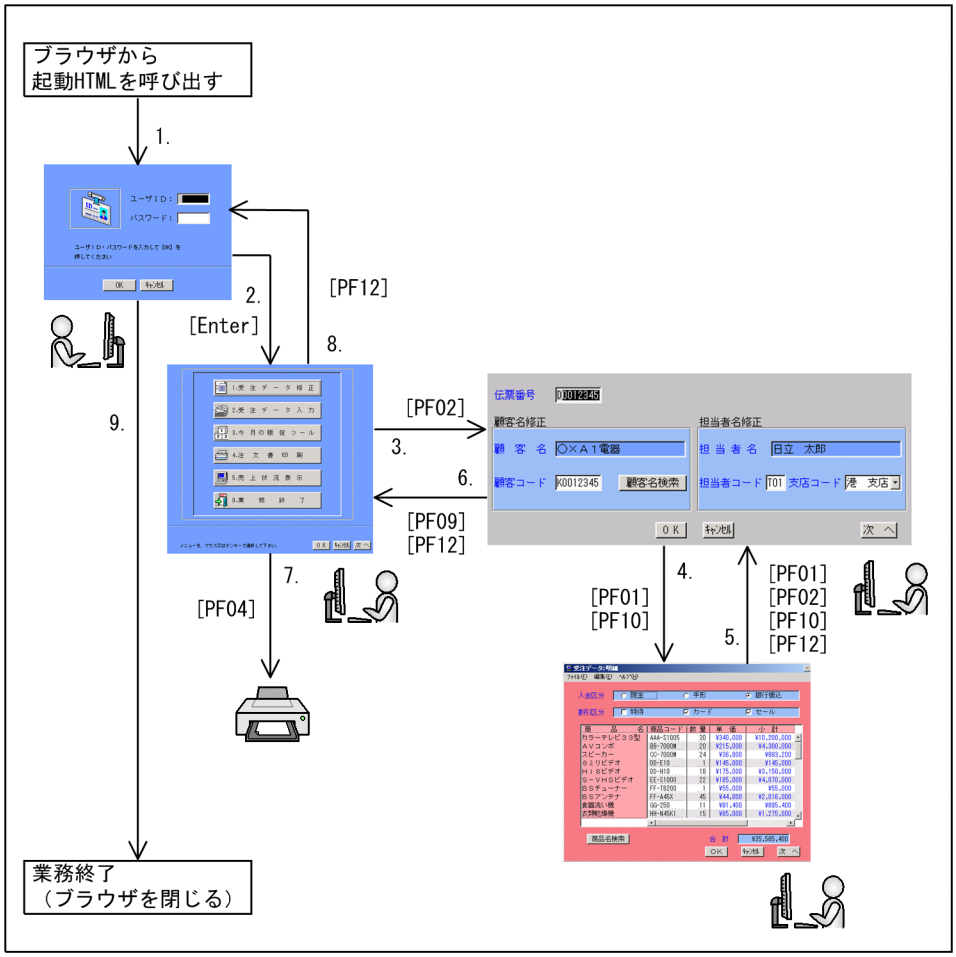
付録 D　サンプルプログラム（XMAP3 Cosminexus 連携）

サンプルプログラムのコンパイルから実行までの手順を説明します。

付録 D.1　サンプルプログラムの概要

サンプルプログラムの動作概要を次の図に示します。

図 D-1　サンプルプログラムの動作概要図



1. 業務開始 URL を指定すると，初期画面（「利用者名入力画面」）が表示される。
2. 初期画面で［OK］ボタンを選ぶか［Enter］キーを押すと，メニュー画面（「受注伝票発行業務」画面）が表示される。
3. メニュー画面で［受注データ修正］ボタンを選ぶか［PF02］キーを押すと，修正ヘッダ画面（「受注データ修正：ヘッダ」画面）が表示される。
4. 修正ヘッダ画面の［ファイル］－［明細］コマンドを選ぶか，［PF01］または［PF10］キーを押すと，修正明細画面（「受注データ：明細」画面）が表示される。
5. 修正明細画面で［ファイル］－［終了］コマンド，または［ファイル］－［取消］コマンドを選ぶか，［PF01］［PF02］［PF10］または［PF12］キーを押すと，修正ヘッダ画面に戻る。

6. 修正ヘッダ画面で［ファイル］－［終了］コマンドを選ぶか，［PF9］または［PF12］キーを押すと，メニュー画面に戻る。
7. メニュー画面で［注文書印刷］ボタンを選ぶか［PF04］キーを押すと，帳票が印刷される。
8. メニュー画面で［業務終了］ボタンを選ぶか［PF12］キーを押すと，初期画面に戻る。
9. 初期画面で［キャンセル］ボタンを選ぶか，［PF12］キーを押して，初期画面を閉じる。

付録 D.2　サンプルプログラムのコンパイル方法（Java）

XMAP3/Web for Cosminexus では，XMAP3 の提供するサンプルプログラム中のファイルを利用した，Java のサンプルプログラムを提供しています。

(1)　サンプルプログラム一覧

XMAP3/Web for Cosminexus が提供するサンプルプログラムの一覧を次の表に示します。

表 D-1　サンプルプログラム一覧（Java）

ファイル名	内容	格納場所
x3xwbfrm.htm	起動 HTML	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥Sample
x3xwbfrm.js	起動 HTML 用スクリプトファイル	
web.xml	Web アプリケーション用の DD（業務サーブレットを呼び出す URL 定義情報）	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥JAVA
xmap3.properties	環境管理ファイル	
AppInterface.java	共通 Java AP：入出力処理を行うクラスのためのインタフェース	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥JAVA¥sample¥appmanager
AppManager.java	共通 Java AP：画面遷移の情報を基に入出力処理を行うクラスを呼び出すクラス	
AppMappingValue.java	共通 Java AP：AppTable の補助クラス	
AppTable.java	共通 Java AP：画面遷移の情報を指定するクラス	
InciMappingValue.java	共通 Java AP：AppTable の補助クラス	
KAD1GHND_ENTR.java	業務 Java AP：図 D-1 の 2.の処理	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥JAVA¥sample¥apps
KAD1GHND_OTHER.java	業務 Java AP：図 D-1 の 9.の処理	
KAD1PH6G_OTHER.java	業務 Java AP：帳票出力後処理	

ファイル名	内容	格納場所
KAD2GHND_OTHER.java	業務 Java AP：KAD2GHND 画面での PF02，PF04，PF12 以外のキーを押したときの処理	
KAD2GHND_PF02.java	業務 Java AP：図 D-1 の 3.の処理	
KAD2GHND_PF04.java	業務 Java AP：図 D-1 の 7.の処理	
KAD2GHND_PF12.java	業務 Java AP：図 D-1 の 8.の処理	
KAD4GHND_OTHER.java	業務 Java AP：KAD4GHND 画面での PF01，または PF09 以外のキーを押したときの処理	
KAD4GHND_PF01.java	業務 Java AP：図 D-1 の 4.の処理	
KAD4GHND_PF09.java	業務 Java AP：図 D-1 の 6.の処理	
KAD6GHND_OTHER.java	KAD6GHND 画面での PF01 以外のキーを押したときの処理	
KAD6GHND_PF01.java	業務 Java AP：図 D-1 の 5.の処理	
START.java	業務 Java AP：図 D-1 の 1.の処理	
SampleServlet.java	業務サーブレットのソースプログラム	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥JAVA¥sample¥servlet
xmap3server.jar	XMAP3 実行クラスライブラリ	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥Lib
X3XCOMTBL.xml	通信制御用 XML 文書	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥Etc
X3MODTBL.H	動的変更テーブル	*XMAP3 インストールフォルダ*¥INCLUDE
KAD1GH.imp	マップ定義ファイル初期画面（KAD1GH 画面）	*XMAP3 インストールフォルダ*¥SAMPLES¥C
kad1ph.imp	マップ定義ファイルKAD1PH 帳票印刷	
kad2gh.imp	マップ定義ファイルKAD2GH 画面	
KAD4GH.imp	マップ定義ファイルKAD4GH 画面	
KAD6GH.imp	マップ定義ファイルKAD6GH 画面	

(2)　コンパイル前に必要な準備作業

業務 JavaAP（apps¥*.java），共通 JavaAP（appmanager¥*.java），および業務サーブレットのソースプログラム（SampleServlet.java）をコンパイルする前に，物理マップと XML 文書を作成します。

マップ定義ファイル（拡張子は「.imp」）から物理マップファイル（拡張子は「.pmp」）と論理マップファイル（拡張子は「.h」）を生成します。また，Java 言語用ツールを使用して，論理マップファイルから入力／出力データ用 XML 文書と定数用 XML 文書を生成します。

KAD4GH.imp，KAD6GH.imp のメニューバー定義で，プルダウンメニューが定義されていないメニューバーは削除してください。

(3)　ユーザプログラムおよびサーブレットのコンパイル

MyEclipse, Eclipse または JBuilder を使用して，業務 JavaAP（apps¥*.java），共通 JavaAP（appmanager¥*.java），および業務サーブレットのソースプログラム（SampleServlet.java）をコンパイルし，それぞれの class ファイルを生成します。

コンパイルするときには，次の必須ライブラリを追加してコンパイルしてください。

- xmap3server.jar
- javax.servlet および javax.servlet.http パッケージを含むライブラリ※

注※

バージョンに合ったライブラリ（servlet.jar または javax.servlet）をライブラリパスに指定して「ユーザーホーム」に新規ライブラリを定義してください。

コンパイルの手順については，Cosminexus アプリケーションサーバのドキュメントを参照してください。

付録 D.3　環境設定例（Java）

XMAP3/Web for Cosminexus および Cosminexus がインストールされているサーバ環境と，クライアントとなる Web ブラウザの環境設定について設定例を示します。

環境設定例は，次の環境を想定しています。Web アプリケーション名は「XmapWebApp」としています。

XMAP3/Web for Cosminexus インストールフォルダ

```
c:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus
```

スクリプトファイルの格納場所

```
c:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus¥Script
```

環境ファイルの格納場所

```
c:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus¥Etc
```

起動 HTML（X3XVWBFRM.htm）および起動 HTML 用スクリプトファイル（X3XVWBFRM.js）の格納場所

```
c:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus¥SAMPLE
```

環境管理ファイル（xmap3.properties）の格納場所

```
c:¥Cosminexus¥CC¥Web¥containers¥サーバ名¥webapps¥XmapWebApp¥WEB-INF¥classes
```

共通 JavaAP（appmanager¥*.java）から生成した class ファイルの格納場所

```
c:¥Cosminexus¥CC¥Web¥containers¥サーバ名¥webapps¥XmapWebApp¥WEB-INF¥classes¥sample
¥appmanager
```

業務 JavaAP（apps¥*.java）から生成した class ファイルの格納場所

```
c:¥Cosminexus¥CC¥Web¥containers¥サーバ名¥webapps¥XmapWebApp¥WEB-INF¥classes¥sample¥apps
```

SampleServlet.class の格納場所

c:¥Cosminexus¥CC¥Web¥containers¥*サーバ名*¥webapps¥XmapWebApp¥WEB-INF¥classes¥sample¥servlet

xmap3server.jar

c:¥Cosminexus¥CC¥Web¥containers¥*サーバ名*¥webapps¥XmapWebApp¥WEB-INF¥lib

XML 文書の格納場所

c:¥Cosminexus¥CC¥Web¥containers¥*サーバ名*¥webapps¥XmapWebApp¥WEB-INF¥XMAP3

Web アプリケーション用の DD（web.xml）の格納場所

c:¥Cosminexus¥CC¥Web¥containers¥*サーバ名*¥webapps¥XmapWebApp¥WEB-INF

物理マップの格納場所

c:¥Program Files¥HITACHI¥XMAP3¥Web for Cosminexus¥SAMPLE¥JAVA

注

webapps 下のサブフォルダについては，ユーザが手動で作成する必要があります。

(1)　サーバ環境の設定

サーバ環境では次の環境設定が必要です。

- Web サーバのディレクトリ割り当て
- XMAP3/Web for Cosminexus 定義ファイルの設定
- Cosminexus の設定

(a)　Web サーバのディレクトリ割り当て

Cosminexus HTTP Server（Hitachi Web Server）または IIS で，次のディレクトリ名を割り当ててください。

URL に使用するディレクトリ名	割り当てフォルダ	内容
xmap3	c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus	XMAP3 Cosminexus 連携機能インストールフォルダ

(b)　XMAP3/Web for Cosminexus 定義ファイルの設定

サンプル動作に必要な定義ファイルの設定例を示します。ここでは，XMAP3/Web for Cosminexus の提供時のファイルを前提として変更点だけを示します。

- 起動 HTML 定義ファイル（X3XWBFRM.htm）および起動 HTML 用スクリプトファイル（X3XWBFRM.js）

  起動 HTML 定義ファイルおよび起動 HTML 用スクリプトファイルは，サンプル提供の「X3XWBFRM.htm」および「X3XWBFRM.js」を複写して作成してください。次に示すフォルダに提供しています。

  XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE

- 起動 HTML 定義ファイル

  ＜ CLASSID 属性および CODEBASE 属性の設定値＞

  この個所は，原則として修正を禁止します。この個所を修正した場合，Web ブラウザの実行については保証しません。

  ＜ PARAM タグに対応した設定値＞

Web ブラウザ実行時の情報を，起動 HTML 用スクリプトファイルの PARAM タグに対して，起動 HTML 定義ファイルに設定します。

```
                    :
"http://server_url※1/xmap3/Script/X3XWBJSC.JS",
"http://server_url※1※2/XmapWebApp/servlet/kaden※3",
"http://server_url※1/xmap3/Etc",
"http://server_url※1/xmap3/SAMPLE/Java",
"1F",
"IGNORE")
```

注※1

「server_url」には，サーバの IP アドレスを記載してください。

注※2

リダイレクタの設定を行っていない場合は，ポート番号を指定する必要があります。次にポート番号の指定形式を示します。

server_url：*ポート番号*

注※3

業務サーブレットを呼び出す URL 定義情報が web.xml に定義されているため，「kaden」で呼び出せます。

- 環境管理ファイル（xmap3.properties）

  定義項目「LogFileDir」にログの出力先フォルダを指定します。インストール直後のファイルには出力先フォルダの指定がないため，必ず指定してください。

  環境管理ファイルは，次の場所に格納されます。

  XMAP3インストールフォルダ¥Web for Cosminexus¥SAMPLE¥JAVA

(c)　Cosminexus の設定

サーバ側の Cosminexus の実行環境を設定してください。

(2)　クライアント環境の設定

クライアントの Web ブラウザの，セキュリティ，一時ファイルを設定してください。

付録 D.4　サンプルプログラムの実行（Java）

サンプルプログラムを実行する前に，次に示すサーバを起動しておいてください。

J2EE サーバモードの場合

J2EE サーバおよび Web サーバ

サーブレットエンジンモードの場合

Web コンテナサーバおよび Web サーバ

Web ブラウザから起動 HTML 定義ファイルを呼び出すと，業務サーブレットが呼び出され，用意した画面がブラウザ上に表示されます。

付録 D.5　サンプルプログラムのコンパイル方法（COBOL）

XMAP3/Web for Cosminexus では，XMAP3 の提供するサンプルプログラム中のファイルを利用した，COBOL のサンプルプログラムを提供しています。

(1)　サンプルプログラム一覧

XMAP3/Web for Cosminexus および XMAP3 が提供するサンプルプログラムの一覧を次の表に示します。

表 D-2　サンプルプログラム一覧（COBOL）

ファイル名	内容	格納場所
X3XWBFRM.htm	起動 HTML	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE
X3XWBFRM.js	起動 HTML 用のスクリプトファイル	
CBLIOTBL.cbl	データ送受信用の登録集原文	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥COBOL
KADCNTRL.cbl	COBOL AP（共通）	
KADSTART.cbl	COBOL AP（業務処理）初期画面（KAD1GC 画面）の表示処理	
KAD1GC.cbl	COBOL AP（業務処理）KAD1GC 画面の入力判定処理	
KAD2GC.cbl	COBOL AP（業務処理）KAD2GC 画面の入力判定処理	
KAD4GC.cbl	COBOL AP（業務処理）KAD4GC 画面の入力判定処理	
KAD6GC.cbl	COBOL AP（業務処理）KAD6GC 画面の入力判定処理	
KAD1PC.cbl	COBOL AP（業務処理）KAD1PC 帳票印刷の後処理（次画面表示）	
CBLBEAN.java	COBOL アクセス用 Bean の生成ソース	
CBLSRV.java	通信制御サーブレットのソースプログラム	
web.xml	Web アプリケーション用の DD（通信制御サーブレットを呼び出す URL 定義情報）	
kaden.hmf	COBOL プロジェクトマスタファイル	
CBLIOTBL.cbl	データ送受信用の登録集原文（業務 MPP 用）	*XMAP3 インストールフォルダ*¥Web for Cosminexus¥SAMPLE¥COBOL_KEIS
CBLIOTBL_IN_BEAN.cbl	入力データ用の登録集原文（業務 MPP の Bean 生成用）	
CBLIOTBL_OUT_BEAN.cbl	出力データ用の登録集原文（業務 MPP の Bean 生成用）	
CBLBEAN_KEIS.java	TP1/COBOL アクセス用 Bean の生成ソース（業務 MPP 用）	

ファイル名	内容	格納場所
CBLSRV_KEIS.java	通信制御サーブレットのソースプログラム（業務 MPP 用）	
web.xml	Web アプリケーション用の DD（通信制御サーブレットを呼び出す URL 定義情報）	
X3MODTBL.CBL	動的変更テーブル	*XMAP3 インストールフォルダ*¥INCLUDE
KAD1GC.imp	マップ定義ファイル初期画面（KAD1GC 画面）	*XMAP3 インストールフォルダ*¥SAMPLES¥COBOL
KAD2GC.imp	マップ定義ファイルKAD2GC 画面	
KAD4GC.imp	マップ定義ファイルKAD4GC 画面	
KAD6GC.imp	マップ定義ファイルKAD6GC 画面	
KAD1PC.imp	マップ定義ファイルKAD1PC 帳票印刷	

(2)　コンパイル前に必要な準備作業

XMAP3 が提供するサンプルプログラム中のマップ定義ファイルおよび動的変更テーブルをユーザプログラムの COBOL プロジェクトマスタファイルと同じ場所に配置してください。

COBOL プロジェクトマスタファイルをコンパイルすると，物理マップファイル（.PMP），論理マップファイル（.CBL），生成情報ファイル（.LSX）が自動生成されます。

KAD4GC.imp，KAD6GC.imp のメニューバー定義で，プルダウンメニューが定義されていないメニューバーは削除してください。

(3)　サンプルプログラムのコンパイル

(a)　ユーザプログラムのコンパイル

Cosminexus ベースのシステムの場合

COBOL2002 の開発マネージャ for COBOL2002 を使用して，ユーザプログラムの COBOL プロジェクトマスタファイル（kaden.hmf）を開いてビルドし，「kaden.dll」を作成します。

(b)　Bean およびサーブレットのコンパイル

MyEclipse，Eclipse または JBuilder を使用して，COBOL アクセス用 Bean の生成ソースおよび通信制御サーブレットのソースプログラムをコンパイルし，それぞれ class ファイルを生成します。

コンパイル時には次の必須ライブラリを追加してコンパイルしてください。

- 「J2cb2k」ライブラリ※1
- javax.servlet および javax.servlet.http パッケージを含むライブラリ※2

注※1

JBuilder を使用してコンパイルする場合，COBOL2002 の Cosminexus 連携機能の COBOL adapter JBuilder への部品組み込みツールによって JBuilder に組み込んでおく必要があります。

注※2

バージョンに合ったライブラリをライブラリパスに指定して「ユーザーホーム」に新規ライブラリを定義してください。

コンパイルの手順については，Cosminexus アプリケーションサーバのドキュメントを参照してください。

付録 D.6　環境設定例（COBOL）

XMAP3/Web for Cosminexus および Cosminexus がインストールされているサーバ環境と，クライアントとなる Web ブラウザの環境設定について設定例を示します。

環境設定例は，次の環境を想定しています。Web アプリケーション名は「XmapWebApp」としています。

MyEclipse でのファイル定義

COBOL アクセス用 Bean（CBLBEAN.class）

通信制御サーブレット（CBLSRV.class）

Web アプリケーション用の DD（web.xml）

XMAP3/Web for Cosminexus インストールフォルダ

```
c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus
```

スクリプトファイルの格納場所

```
c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus¥Script
```

環境ファイルの格納場所

```
c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus¥Etc
```

起動 HTML および起動 HTML 用スクリプトファイルの格納場所

```
c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus¥SAMPLE
```

ユーザプログラム（kaden.dll）の格納場所※

```
c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus¥SAMPLE¥COBOL¥kaden
```

物理マップの格納場所

```
c:¥Program files¥HITACHI¥XMAP3¥WEB for Cosminexus¥SAMPLE¥COBOL
```

論理マップの格納場所

```
c:¥Cosminexus¥CC¥Web¥containers¥サーバ名¥webapps¥XmapWebApp
```

注※

Cosminexus を操作して，ユーザプログラム（kaden.dll）の格納場所を追加してください。

(1)　サーバ環境の設定

サーバ環境では次の環境設定が必要です。

- Web サーバのディレクトリ割り当て
- XMAP3/Web for Cosminexus 定義ファイルの設定
- Cosminexus の設定

(a) Web サーバのディレクトリ割り当て

Cosminexus HTTP Server（Hitachi Web Server）または IIS で，次のディレクトリ名を割り当ててください。

URL に使用するディレクトリ名	割り当てフォルダ	内容
xmap3	c:¥Program files¥HITACHI¥XMAP3¥Web for Cosminexus	XMAP3 Cosminexus 連携機能インストールフォルダ

(b) XMAP3/Web for Cosminexus 定義ファイルの設定

サンプル動作に必要な定義ファイルの設定例を示します。ここでは，XMAP3/Web for Cosminexus の提供時のファイルを前提として変更点だけを示します。

起動 HTML 定義ファイルおよび起動 HTML 用スクリプトファイルは，サンプル提供の「X3XWBFRM.htm」および「X3XWBFRM.js」を複写して作成してください。次に示すフォルダに提供しています。

XMAP3/Web for Cosminexusインストールフォルダ¥SAMPLE

- 起動 HTML 定義ファイル

  ＜ PARAM タグに対応した設定値＞

  Web ブラウザ実行時の情報を，起動 HTML 用スクリプトファイルの PARAM タグに対して，起動 HTML 定義ファイルに設定します。

  ```
  "http://server_url※1/xmap3/Script/X3XWBJSC.JS",
  "http://server_url※1※2/XmapWebApp/servlet/kaden※3",
  "http://server_url※1/xmap3/Etc",
  "http://server_url※1/xmap3/SAMPLE/COBOL",
  "1F"
  "NORMAL")
  ```

  注※1

  「server_url」には，サーバの IP アドレスを記載してください。

  注※2

  リダイレクタの設定を行っていない場合は，ポート番号を指定する必要があります。次にポート番号の指定形式を示します。

  server_url：*ポート番号*

  注※3

  通信制御サーブレットを呼び出す URL 定義情報が web.xml に定義されているため，「kaden」で呼び出せます。

(c) Cosminexus の設定

サーバ側の Cosminexus の実行環境を設定してください。

(2) クライアント環境の設定

クライアントの Web ブラウザの，セキュリティ，一時ファイルを設定してください。

付録 D.7　サンプルプログラムの実行（COBOL）

サンプルプログラムを実行する前に，次に示すサーバを起動しておいてください。

J2EE サーバモードの場合

　J2EE サーバおよび Web サーバ

サーブレットエンジンモードの場合

　Web コンテナサーバおよび Web サーバ

Web ブラウザから起動 HTML 定義ファイルを呼び出すと，通信制御サーブレットから COBOL アクセス用 Bean を経由して，サンプルプログラムが呼び出され，用意した画面がブラウザ上に表示されます。

付録 E　サンプルプログラム（XMAP3 TP1/Web 連携）

サンプルプログラムのコンパイルから実行までの手順を説明します。提供サンプルは，画面をブラウザ上に 1 回だけ表示し，応答を返すとブラウザを閉じて終了します。帳票印刷用のサンプルは提供していません。

付録 E.1　サンプルプログラムのコンパイル方法

XMAP3 TP1/Web 連携機能と TP1/Web の関数を呼び出して画面表示する COBOL および C 言語用のサンプルプログラムを提供しています。

(1)　サンプルプログラムの格納場所

サンプルプログラムは次に示すフォルダに格納されています。

(a)　COBOL の場合

- 格納場所
  XMAP3 インストールフォルダ¥Web for TP1¥SAMPLE
- 格納ファイル
  WEBPSTRT.CBL，WEBPSTOP.CBL，WEBUSRV1.CBL，WEBUSRV2.CBL

(b)　C 言語の場合

- 格納場所
  XMAP3 インストールフォルダ¥Web for TP1¥SAMPLE
- 格納ファイル
  WEBSAMPL.C

(2)　コンパイル前に必要な準備作業

サンプルプログラムをコンパイルする前に，次の準備が必要です。

- 論理マップの作成
- サンプルソースの変更
- モジュール定義ファイルの作成

(a)　論理マップの作成

サンプルプログラムをコンパイルするには論理マップが必要です。MAP001.imp をドローで作成し，論理マップを生成してください。実行時には，同時に生成される物理マップが必要となります。

- COBOL を使用する場合
  生成される論理マップは，MAP001O.cbl，MAP001I.cbl です。
- C 言語を使用する場合
  生成される論理マップは，MAP001O.h，MAP001I.h です。

注

サンプルプログラムでは入出力データを参照・設定しないので，任意の画面定義で問題ありません。

(b)　サンプルプログラムの変更

サンプルプログラムは，動作環境に合わせて指定する URL 名が異なるため，そのままコンパイルしても動作しません。コンパイル前に次の点を変更してください。

- COBOL を使用する場合

  webusrv1.cbl を次の手順で変更してください。

  1. NEXT-DATA1 文字列中の「server_url」にサーバマシンの IP アドレスを設定する。
  2. NEXT-DATA2 文字列中の「group.service」を「smpl.usr_service2」に置き換える。
  3. NEXT-DATA1，NEXT-DATA2 の PIC の長さを変更後の文字列の長さに合わせる。
  4. NEXT-LNG の VALUE 値を 3.で変更した文字列の総和に変更する。

- C 言語を使用する場合

  websampl.c を次の手順で変更してください。

  1. URL0 の「server_url」にサーバマシンの IP アドレスを設定する。
  2. URL1 の「group.service」を「smpl.usr_service2」に置き換える。

(c)　モジュール定義ファイルの作成

C 言語を使用する場合，生成した DLL の関数を外部から呼び出せるようにするため，リンク時にモジュール定義ファイル（.def）を取り込む必要があります。COBOL を使用する場合は作成する必要はありません。

C 言語のサンプルプログラムに対応したモジュール定義ファイルの内容を次に示します。ファイルを作成し，リンク時に取り込んでください。

- モジュール定義ファイル（smpl.def）

```
EXPORTS
    exam1_process_start
    exam1_process_stop
    exam1_usr_service1
    exam1_usr_service2
```

(3)　サンプルプログラムのコンパイル

サーバ環境に合わせて変更したサンプルプログラムと作成した論理マップを使って，コンパイルおよびリンケージをしてください。

以降の環境設定の説明では，ここで作成する DLL の名称を「smpl.dll」として説明しています。異なる名称で DLL を作成した場合は，置き換えてお読みください。

付録 E.2　環境設定例

XMAP3/Web for Cosminexus，TP1/Web がインストールされているサーバ環境とクライアントとなる Web ブラウザの環境設定について設定例を示します。

環境設定例は，次の環境を想定しています。

TP1/Web インストールフォルダ

```
c:¥tp1web
```

XMAP3/Web for Cosminexus インストールフォルダ

```
c:¥program files¥hitachi¥xmap3¥Web for TP1
```

サービスプログラム（smpl.dll）の格納場所

```
c:¥tp1web¥aplib
```

物理マップ（MAP001ND.pmp）の格納場所

```
c:¥program files¥hitachi¥xmap3¥user¥maps
```

(1)　サーバ環境の設定

サーバ環境では次の環境設定が必要です。

- TP1/Web 定義ファイルの設定
- IIS での仮想ディレクトリの割り当て
- XMAP3 TP1/Web 連携機能定義ファイルの設定

(a)　TP1/Web 定義ファイルの設定

サンプル動作に必要な定義ファイルの設定例を示します。ここでは，TP1/Web の提供ファイルを前提として変更点だけを示します。

- WWW セション管理機能定義ファイル（wbp.cnf）

```
SessionAssignType    dynamic
MaxSession           20
MaxProcess           10
```

- TP1/Web マネージャサービス定義ファイル（webconf）

```
set web_cltin_to_server  = NONE
set web_usr_conf = smpl
```

- ユーザサービス定義ファイル（smpl）

＜ COBOL 用のサンプルを使用する場合＞

```
set service_group         = smpl
set module                = c:¥tp1web¥aplib¥smpl.dll
set p_start_function      = USR_PROCESS_START
set p_stop_function       = USR_PROCESS_STOP
set service= ”usr_service1=USR_SERVICE_1”,¥
             ”usr_service2=USR_SERVICE_2”
```

＜ C 言語用のサンプルを使用する場合＞

```
set service_group         = smpl
set module                = c:¥tp1web¥aplib¥smpl.dll
set p_start_function      = exam1_process_start
set p_stop_function       = exam1_process_stop
set service= ”usr_service1=exam1_usr_service1”,¥
             ”usr_service2=exam1_usr_service2”
```

(b)　IIS での仮想ディレクトリの割り当て

IIS で表 E-1 および表 E-2 に示すフォルダに仮想ディレクトリを割り当ててください。仮想ディレクトリには，用途に応じたアクセス権を設定してください。

表 E-1　TP1/Web の仮想ディレクトリ定義

項番	仮想ディレクトリ名	割り当てフォルダ
1	tp1web	c:¥tp1web¥wsession¥cgi-bin

表 E-2　XMAP3/Web for Cosminexus の仮想ディレクトリ定義

項番	仮想ディレクトリ名	割り当てフォルダ
1	xmap3	c:¥program files¥hitachi¥xmap3

(c)　XMAP3 TP1/Web 連携機能定義ファイルの設定

サンプル動作に必要な定義ファイルの設定例を示します。ここでは，XMAP3 TP1/Web 連携機能の提供時のファイルを前提として変更点だけを示します。

起動 HTML 定義ファイルおよび起動 HTML 用スクリプトファイルは，サンプル提供の「x3webfrm.htm」および「x3webfrm.js」を複写して作成してください。次に示すフォルダに提供しています。

```
c:¥program files¥hitachi¥xmap3¥web for TP1¥sample
```

- 起動 HTML 定義ファイル（smpl.htm）

  ＜業務開始 URL の設定値＞

  ```
  "http://server_url/tp1web/dcwcgi.exe/DC_USR?DC_RPCCALL.
  smpl.usr_service1.DCNOFLAGS.DCRAP_CON=dmy",
  ```

  注

  「server_url」には，サーバマシンの IP アドレスを記載してください。

- サーバ環境定義ファイル（x3websrv）

  ```
  EtcPath = "http://server_url/xmap3/Web for TP1/etc"
  DataPath = "http://server_url/xmap3/user/maps"
  ```

  注

  「server_url」には，サーバマシンの IP アドレスを記載してください。

(2)　クライアント環境の設定

クライアント環境を設定します。

(3)　サンプルの実行

Web ブラウザから起動 HTML 定義ファイルを呼び出すと，TP1/Web の CGI を呼び出し，サンプルプログラムのサービスが呼び出され，用意した画面がブラウザ上に表示されます。表示された画面で応答を返すと，ブラウザを閉じてプログラムは終了します。

付録 F　リターンコードと詳細コード

リターンコードと詳細コードについて説明します。

付録 F.1　XMAP3 のリターンコードと詳細コード

リターンコード，詳細コードとエラー内容を次の表に示します。

16 進表示のコードは，COBOL のコンソール画面に表示されるエラーコードに対応しています。COBOL のテストデバッガなどで，メモリ中のリターンコードを直接参照する場合は，1 バイト目と 2 バイト目が逆になりますので注意してください。表中で使用する記号の意味を次に示します。

(S) システムの処理

(P) プログラマの処理

なお，次の表のリターンコード詳細に示すエラーコードが，20480～20516（16 進数では$(5000)_{16}$～$(5024)_{16}$）の場合，すでにオープン状態であっても強制的にクローズ要求されたものとして処理します。

クローズ処理によって，表示されている画面が消えたり，その後の要求が 24577 エラー（16 進数では$(6001)_{16}$ エラー）となったりする場合がありますので，エラー原因を解決して AP を再起動してください。

リターンコードと詳細コードを次の表に示します。

表 F-1　リターンコードと詳細コード

詳細コード	内容	リターンコード
1036 $(040C)_{16}$	定義している予約項目名が，OpenTP1 で提供している予約項目名と一致していない。 (S) 処理を続行する。 (P) 予約項目に定義している予約項目名を OpenTP1 で提供している予約項目名と一致することを確認して，物理マップを再作成したあとに，AP を再実行する。	4
1044 $(0414)_{16}$	定義されたイベント通知コードと対応しない確定キーが入力された。 (S) 処理を続行する。 (P) 入力された確定キーに対応するイベント通知コードを定義し，物理マップを再作成して AP を再実行する。	4
1060 $(0424)_{16}$	カーソル位置が不正である。AP で設定するカーソル位置の値が画面上に存在しない。 (S) 処理を続行する。 (P) 画面範囲内のカーソル位置を設定するように AP を修正し，再コンパイル後，再実行する。位置を設定しないで領域をクリアする場合には$(00)_{16}$ を設定する。	4
3072 $(0C00)_{16}$	物理マップの内容が破壊されている。原因として次のことが考えられる。 • デバイス種別とマップが一致していない。例えば，ディスプレイを表示するときに AP で指定したマップがプリンタ用である，またはプリンタで印刷するときに AP で指定したマップがディスプレイ用である。 • 物理マップファイルが破壊されている。 • UNIX へファイル転送するときに，テキスト形式で転送している。 (S) 処理を終了する。 (P) 次の対処をする。	8

詳細コード	内容	リターンコード
	• 物理マップを再作成して，AP を再実行する。 • ファイル転送の問題の場合，再度ファイルを転送してから，AP を実行する。	
3104 $(0C20)_{16}$	実行環境の XMAP3 より上位のバージョン（VV-RR）で作成した物理マップが使用された。 (S) 処理を終了する。 (P) 実行環境の XMAP3 のバージョン（VV-RR）を，物理マップを作成した XMAP3 と同じバージョン（VV-RR）にして，AP を再実行する。	8
3108 $(0C24)_{16}$	物理マップのエンディアンが実行環境と合っていない。 (S) 処理を終了する。 (P) 実行環境に合ったエンディアンとなるように，物理マップを再作成する。	8
4096 $(1000)_{16}$	XMAP3 サーバ実行環境および XMAP3 開発環境間のバージョン（VV-RR）が不整合である。 (S) 処理を終了する。 (P) XMAP3 サーバ実行環境および XMAP3 開発環境のバージョン（VV-RR）関係を調べて，正しいバージョン（VV-RR）の組み合わせで再インストールする。	8
4100 $(1004)_{16}$	何らかの原因でデータ化けが発生したことで，物理マップ中の入力物理マップ名情報が破壊されている。 (S) 処理を終了する。 (P) 物理マップを再作成して，AP を再実行する。メッセージとマップ名を正しく対応づける。	8
4104 $(1008)_{16}$	何らかの原因でデータ化けが発生したことで，物理マップ中の作成管理情報が破壊されている。 (S) 処理を終了する。 (P) 物理マップを再作成して，AP を再実行する。	8
5120 $(1400)_{16}$	指定した物理マップが見つからない。または，AP で指定した仮想端末と指定した物理マップのデバイス種別が一致しない。 (S) 処理を終了する。 (P) 次の対処をする。 • 表示・印刷セットアップの［アプリケーション 1］タブの「マップ」で指定しているフォルダ，AP 環境ファイル中の mapPath で指定しているディレクトリまたは AP を実行したカレントフォルダに，指定した物理マップ（拡張子は pmp）があることを確認し，AP を再実行する。 • 画面の場合，AP で指定した仮想端末名（例えば DSP001）に対して，物理マップのデバイス種別（例えばデバイス ID が ND／NC）が合っているか確認し，AP を修正後，再実行する。 • 帳票の場合，AP で指定した仮想端末名（例えば PRT001）に対して，物理マップのデバイス種別（例えばデバイス ID が 6A／6B／6G／6H）が合っているか確認し，AP を修正後，再実行する。 • UNIX の場合，物理マップに拡張子「.pmp」が付いているときは，cmapcp コマンドで拡張子を削除する。	8
5124 $(1404)_{16}$	物理マップのロード処理中に入出力エラーが発生した。 (S) 処理を終了する。 (P) 物理マップを再作成して，AP を再実行する。	8

詳細コード	内容	リターンコード
5148 $(141C)_{16}$	部品用物理マップのロードで必要なメモリが不足した。 (S) 部品用物理マップなしとして処理を続ける。 (P) 次に示すどちらかの処理をする。 • ほかの AP を終了するか，OS を再起動する。 • マシン上のリソースを見直し，メモリ増設などをする。	4
5152 $(1420)_{16}$	AP を実行したカレントディレクトリ，または AP 環境ファイル（X3MWDRV／XMAPdrv）中の mapPath で指定したディレクトリ中に，部品用物理マップが存在しない。 (S) 部品用物理マップなしとして処理を続ける。 (P) mapPath で指定したディレクトリに部品用物理マップを置く。	4
5156 $(1424)_{16}$	部品用物理マップのロード処理中に入出力エラーが発生した。 (S) 部品用物理マップなしとして処理を続ける。 (P) 部品用物理マップを再作成して，AP を再実行する。	4
5160 $(1428)_{16}$	部品用物理マップの内容が破壊されている。 (S) 部品用物理マップなしとして処理を続ける。 (P) 部品用物理マップを再作成して，AP を再実行する。	4
5164 $(142C)_{16}$	部品用物理マップのロードパスで必要なメモリが不足した。 (S) 処理を終了する。 (P) 次に示すどちらかの処理をする。 • ほかの AP を終了するか，OS を再起動する。 • マシン上のリソースを見直し，メモリ増設などをする。	8
16384 $(4000)_{16}$	次の要因が考えられる。 • ホスト名取得時にエラーが発生した。 • マッピング構成ファイル（X3MWCONF／XMAPconfig）が存在しない，または参照できない（read 権限がない）。 (S) 処理を終了する。 (P) 次の設定を見直す。 • 仮想端末名ファイル（X3MWHOST／XMAPhosts）またはマッピング構成ファイル（X3MWCONF／XMAPconfig）のホスト名を確認し，自ホストのホスト名を正しく指定する。 • マッピング構成ファイル（X3MWCONF／XMAPconfig）を作成する，または read 権限を付与する。	8
16388 $(4004)_{16}$	マッピング処理プログラムの実行環境が不正なため，エラーが発生した。原因としては，XMAP3 実行環境のインストールが正しくされていないことが考えられる。 (S) 処理を終了する。 (P) 次の対処をする。 • XMAP3 実行環境を再インストールする。 • 実行環境に必要なソフトウェアをインストールする。	8
16396 $(400C)_{16}$	次のファイルの内容が不正，またはファイルが破壊されている。 • 仮想端末名ファイル（X3MWHOST／XMAPhosts） • マッピング構成ファイル（X3MWCONF／XMAPconfig）	8

詳細コード	内容	リターンコード
	• AP 環境ファイル（X3MWDRV／XMAPdrv） (S) 処理を終了する。 (P) ファイルの内容を見直して正しく定義する。	
16400 $(4010)_{16}$	何らかの原因でデータ化けが発生したことによって，物理マップ中のサイズ管理情報が破損している。 (S) 処理を終了する。 (P) 物理マップを再作成して，AP を再実行する。	8
16428 $(402C)_{16}$	同時に動作しているほかの AP などの関係，またはリソースの物理的不足によって，XMAP3 に必要なメモリが不足した。 (S) 処理を終了する。 (P) 次の対処をする。 • ほかの AP を終了するか，OS を再起動する。 • マシン上のリソースの見直し，スワップ領域の見直しおよびメモリの増設などをする。	8
16436 $(4034)_{16}$	仮想端末と物理マップのデバイス種別が一致していない。 原因としては，次のことが考えられる。 • 画面用の仮想端末に帳票用の物理マップを指定した。 • 帳票用の仮想端末に画面用の物理マップを指定した。 • 物理マップのマップ名をエクスプローラなどで変更した。 • 物理マップが壊れている。 • XMAP3 インストールフォルダ下の ETC フォルダの X3MWCONF ファイルがない，または空ファイルになっている。 (S) 処理を終了する。 (P) 次の対処をする。 • AP で指定している仮想端末と物理マップのデバイス種別が一致することを確認したあとで，再コンパイルして，再実行する。 • 物理マップを再作成して，AP を再実行する。 • X3MWCONF ファイルがないまたは空ファイルの場合は，アンインストール後に再インストールする。	8
16456 $(4048)_{16}$	サーバサービス番号がライブラリマッピング指定(0)となっていない。 (S) 処理を終了する。 (P) サーバサービス番号として，「0」を指定する。	8
20480 $(5000)_{16}$	XMAP3 の定義ファイルの内容と C/S システム環境での通信関係の設定が不正である。 (P) OS でのネットワーク構成との不整合が生じたことによる障害が考えられる。サーバ，クライアントおよび OS をすべて終了し，再起動する。	8
	仮想端末名ファイル（X3MWHOST／XMAPhosts）や DISPLAY 環境変数で，誤ったサービス名を指定している。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECPARM のエラー内容を参照。 (P) 仮想端末名ファイル（X3MWHOST／XMAPhosts）中で画面用として定義したサービス名が，サービス名ファイル（X3PHOST／XPWhosts）中でプリンタに対応づけられていないか，または DISPLAY 環境変数の設定値とサービス名ファイルのサービス名が不一致でないかなど，ファイル設定の相互関係を見直し，正しく設定する。	

詳細コード	内容	リターンコード
20482 $(5002)_{16}$	二次ウィンドウが表示されていない状態で二次ウィンドウに対して入力要求を行ったためにエラーが発生した。AP 内での処理シーケンスが不正になっていることが考えられる。 (P) 画面の送受信のシーケンスを見直す。	8
20484 $(5004)_{16}$	同時に動作しているほかの AP などの関係，またはリソースの物理的不足によって，表示サービスまたは印刷サービスの実行時に必要なメモリが不足した。 (P) 次の対処をする。 • ほかの AP を終了するか，OS を再起動する。 • マシン上のリソースを見直し，メモリの増設などを行う。	8
20485 $(5005)_{16}$	C/S システム環境での通信処理で，回復不能エラーが発生した。同時に動作しているほかの AP などの関係で，マシン上のリソース（主にメモリ）が不足したことが考えられる。また，LAN の回線上で障害，データ化けが発生したことが考えられる。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECFERR のエラー内容を参照。 (P) 次の対処をする。 • ほかの AP を終了するか，OS を再起動する。 • ログファイルを出力して原因を調査する。	8
20487 $(5007)_{16}$	何らかの原因でデータ化けが発生したことで，物理マップ中のバージョン（VV-RR）情報が破壊されている。 (P) 物理マップを再作成して，AP を再実行する。	8
20488 $(5008)_{16}$	メインウィンドウ（一次ウィンドウ）が表示されていない状態で，サブウィンドウ（二次ウィンドウ）を表示すると発生する。一般的には AP 内での処理が不正になっていることが考えられる。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECIMSG のエラー内容を参照。 (P) 次の対処をする。 • 表示するマップ名および画面の表示処理シーケンスを見直す。 • 仮想端末名ファイル（X3MWHOST／XMAPhosts）とマッピング構成ファイル（X3MWCONF／XMAPconfig）の設定値の関係を見直す。	8
20489 $(5009)_{16}$	**(画面の場合)** XMAP3 内部処理で何らかの異常が発生したため，表示サービスのプログラムが異常終了した。 (P) システム的な要因が原因である（システムリソースの異常・ほかの AP での異常）ことが考えられる。 Windows を再起動・再実行する。 **(帳票の場合)** プリンタが利用できない，プリンタ／サービス名が定義されていない，または XMAP3 内部処理で何らかの異常が発生したことで，表示サービスまたは印刷サービスのプログラムが停止した。 または，TCP/IP 通信時，通信相手との接続でエラーが発生した。詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECDOWN のエラー内容を参照。 (P) 次の対処をする。 • プリンタの障害を取り除いてから再実行する。 • 表示・印刷セットアップまたはサービス名ファイル（X3PHOST／XPWhosts）で指定したプリンタ名が正しいか確認し，再実行する。	8

詳細コード	内容	リターンコード
	• ほかの AP がプリンタを使用していないかを確認し，AP を終了して再実行する。 • システム的な要因が原因であること（システムリソースの異常・ほかの AP での異常）が考えられる。OS を再起動・再実行する。 • サービスが起動されているか確認し，起動されていない場合は，サービス起動後，AP を再実行する。 • サービス名ファイル中のデバイス名を正しく指定し，XMAP3 サーバを再起動したあと，AP を再実行する。	
20490 $(500A)_{16}$	画面属性の入力単位に「表示直後」を定義した画面で，「入力必須」または「選択必須」を定義したオブジェクトが存在する。 (P) 画面属性およびオブジェクトに指定している属性を見直し，物理マップを再作成して，AP を再実行する。	8
20493 $(500D)_{16}$	XMAP3 内部処理で何らかの異常が発生したため，サーバ内のプログラムに論理矛盾が起きた。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECSVRC のエラー内容を参照。 (P) システム的な要因が原因である（システムリソースの異常・ほかの AP での異常）ことが考えられる。OS を再起動・再実行する。	8
20505 $(5019)_{16}$	別プログラムでプリンタを使用中のため，印刷できなかった。または，スタンドアロン印刷で，同時に複数のプリンタをオープンした。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECIEPR のエラー内容を参照。 (P) プリンタが使用されていないことを確認し，印刷要求を再実行する。スタンドアロン印刷で，複数のプリンタをオープンしている場合には，同時に複数のプリンタをオープンしないように AP を変更する。	8
20507 $(501B)_{16}$	用紙切れ，電源 OFF などのプリンタ障害，Windows 上で印刷に必要なサービスが停止するなどの障害が発生した。 (P) プリンタ，および Windows 上の印刷障害を取り除いてから，印刷要求を再実行する。	8
20508 $(501C)_{16}$	次の要因が考えられる。 • TCP/IP による通信処理で何らかの障害が発生した。 • インストールの失敗によって，印刷に必要なモジュールが格納されていない。 • XMAP Server Runtime 内部でシステムコールエラーが発生した。詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECSYTM のエラー内容を参照。 (P) 次の対処をする。 • 1 回も動作していない場合は，TCP/IP 関連の設定環境を見直す。動作中に発生した場合は，相手 UNIX・Windows の状態を確認し，AP を再実行する。 • 既存のモジュールをアンインストールし，再インストールをする。 • XPW_DAEMON_PORT_NO 環境変数の設定と SERVICES ファイルのサービス名を確認し，間違っている場合は修正し，XMAP3 サーバ，AP を再起動する。 • サービス名ファイル（X3PHOST／XPWhosts）の設定とホスト名，IP アドレス（/etc/hosts ファイル）を確認し，間違っている場合は修正し，XMAP3 サーバ，AP を再起動する。 • サービスが起動されているか確認し，起動されていない場合は，サービス起動後，AP を再実行する。	8

詳細コード	内容	リターンコード
	XMAP3 Cosminexus 連携機能または XMAP3 TP1/Web 連携機能の場合で，サーバの EtcPath にプリンタ構成ファイルが存在しない。 (P) サーバの EtcPath にプリンタ構成ファイルを格納し，ユーザプログラムを再実行する。	
20514 $(5022)_{16}$	印刷サービス，または表示サービスが利用できない。 • Windows 版で 1 台のプリンタの構成の場合，出力先プリンタを「通常使うプリンタ」に設定していない。 • 複数台の構成の場合，表示・印刷セットアップで設定したプリンタデバイス名と，仮想端末名ファイル（X3MWHOST／XMAPhosts）中のプリンタ仮想端末のサービス名が一致していない。 • 出力する仮想端末のマシンが起動していないため，XMAP3 が動作していない。 • 出力する仮想端末関連の定義が不正なため，XMAP3 の表示・印刷サービスが起動されていない。 • 仮想端末名ファイル（X3MWHOST／XMAPhosts）またはサービス名ファイル（X3PHOST／XPWhosts）と表示・印刷セットアップの定義情報が不一致。 • UNIX 版で仮想端末名ファイル（XMAPhosts）中でサービス名に「**」と指定している場合，AP を起動した端末の DISPLAY 環境変数に誤りがある。詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECNOSV のエラー内容を参照。 (P) 次の対処をする。 • Windows 版で出力先プリンタを「通常使うプリンタ」に設定する。 • Windows 版で表示・印刷セットアップで定義したプリンタデバイス名と仮想端末名ファイル中のサービス名を一致させる。 • Windows 版で出力先のマシンを起動する。 • Windows 版で C/S セットアップを実行し，表示・印刷セットアップで AP から指定している仮想端末名のサービス名を一致させてから，AP を再実行する。 • UNIX 版で「**」と指定している場合は，サービス名ファイル中で，表示したいディスプレイ名に対応づけられたサービス名を端末の DISPLAY 環境変数に正しく指定して，AP を再実行する。 • UNIX 版で仮想端末名ファイル（XMAPhosts）とサービス名ファイル（XPWhosts）のサービス名を一致させて XMAP3 サーバを再起動してから，AP を再実行する。	8
20515 $(5023)_{16}$	C/S システム環境の場合，表示サービスまたは印刷サービスが存在する環境の XMAP3 と，AP が動作している環境の XMAP3 のバージョン（VV-RR）が不整合である。 (P) それぞれインストールされている XMAP3 のバージョン（VV-RR）を確認し，整合性のあるバージョン（VV-RR）になるようにインストールし直す。基本的に XMAP3 間では同一バージョン（VV-RR）を利用する。また，UNIX との C/S 構成の場合，前提となる UNIX 側のバージョン（VV-RR）はリリースノートを参照する。	8
20516 $(5024)_{16}$	XMAP3 の出力処理中に，OS または XMAP3 の内部で回復不能なエラーが発生した。 詳細については，マニュアル「XMAP3 実行ガイド」の TX_CECERHD のエラー内容を参照。 (P) システム的な要因が原因である（システムリソース不足，プリンタマネージャの異常検知など）ことが考えられる。OS を再起動し，再実行する。または，印刷処理の場合は，出力プリンタを確認し，再実行する。	8
24576 $(6000)_{16}$	次の要因が考えられる。	8

詳細コード	内容	リターンコード
	• jsvwadrv 関数を実行時，指定したパラメタが不正である。または TRANSCEIVE 要求，RECEIVE 要求，jsvwadrv 関数実行時に誤った論理マップ名を指定しているか，リトルエンディアン／ビッグエンディアンが誤っている物理マップを指定している。 • XMAP3 開発環境でビッグエンディアン指定しないで作成したマップを UNIX で使用した。 (S) 処理を終了する。 (P) AP 中の jsvwadrv 関数を発行している処理で指定したパラメタが正しいかを見直し，再コンパイル後，再実行する。	
24577 $(6001)_{16}$	jsvwadrv 関数の発行順序が不正である（OPEN 要求を行わずにほかの要求を行った，または SEND 要求を行わずに RECEIVE 要求を行った）。 (S) 処理を終了する。 (P) AP 中の jsvwadrv 関数を発行している処理で発行順序が正しいかを見直し，再コンパイル後，再実行する。	8
24578 $(6002)_{16}$	仮想端末名ファイル（X3MWHOST／XMAPhosts）の内容に次の不正がある。 • AP で設定している仮想端末名が定義されていない。 • 1 行の終わりに改行が設定されていない。 • Windows 版 XMAP3 の場合に，1 行が 511 バイトを超えている。 • UNIX 版 XMAP3 の場合に，1 行が 255 バイトを超えている。 • ファイルサイズが 32,767 バイトを超えている。 • 次の各パラメタが形式外 仮想端末名：8 文字以内，先頭は英字，英数字を使用 デバイス：指定できる文字は固定 サーバホスト名：32 文字以内 サーバサービス番号：5 文字以内，数字を使用，1024～65535 の範囲 サービス名：40 文字以内，半角文字を使用 環境設定ファイル名：64 文字以内 (S) 処理を終了する。 (P) AP で設定した仮想端末名を見直す。または仮想端末定義ファイルの設定内容を見直し，再実行する。	8
24579 $(6003)_{16}$	ファイルシステム容量不足や，何らかの原因によって，ログファイルのアクセス中にエラーが発生した。 (S) 処理を終了する。 (P) 次の対処をする。 • ほかの AP を終了するか，OS を再起動する。 • ログファイルを出力するために指定したフォルダが書き込みできるかを確認する。	8
24580 $(6004)_{16}$	仮想端末名ファイル（X3MWHOST／XMAPhosts）のオープン処理または物理マップロード中に必要なメモリが不足した。 (S) 処理を終了する。 (P) ほかの AP を終了するか，OS を再起動する。	8
24581 $(6005)_{16}$	仮想端末名ファイル（X3MWHOST／XMAPhosts）または AP 環境ファイル（X3MWDRV／XMAPdrv）へのアクセス中に入出力エラーが発生した。 (S) 処理を終了する。	8

詳細コード	内容	リターンコード
	(P) 仮想端末名ファイル（X3MWHOST／XMAPhosts）および AP 環境ファイル（X3MWDRV／XMAPdrv）が参照できることを確認し，AP を再実行する。環境設定ファイルを参照できない場合は，アンインストール後に再インストールする。	
24584 $(6008)_{16}$	COBOL の CALL 文を使用して XMAP3 の画面をアクセスする場合，または C 言語でアクセスする場合，共通インタフェーステーブル中に指定した入力論理マップの大きさが，実際の論理マップの大きさより小さい。 (S) 処理を終了する。 (P) AP の処理を見直し，再コンパイル後，再実行する。	8

付録 F.2　書式オーバレイ印刷で使用する関数のリターンコードとエラー詳細コード

書式オーバレイ印刷で使用する関数のリターンコードとエラー詳細コードを次の表に示します。

表中の記号（P）は，プログラマの処置を意味します。

表 F-2　関数のリターンコードとエラー詳細コード

関数名	リターンコード (lcomrc)	エラー詳細コード（lcomerr）		要因および対処
		エラー詳細	値	
jstqlopn	8	JSTQ_MAPSVNAME	268435458 $(10000002)_{16}$	入出力制御関連エラー：指定されたサービス名が未登録である，または対応する印刷サービスが動作していない。 (P) サービス名を確認して再実行する。
	8	JSTQ_MAPSRVOFF	268435489 $(10000003)_{16}$	入出力制御関連エラー：印刷サービスが起動されていない。 (P) サービス名を確認して再実行する。
jstqlpag jstqlopn	8	JSTQ_NOPMOD	65538 $(00010002)_{16}$	行制御データファイルがない。 (P) • 指定したフォルダに行制御データファイルを格納する。 • 環境変数 XMAP3_FMP を指定している場合，環境変数に指定した名称と同じ行制御データファイルが存在するかを確認する。
	8	JSTQ_PMODERR	65539 $(00010003)_{16}$	行制御データファイルが不正である。 (P) 正しい行制御データファイルを指定する。
	8	JSTQ_IOERRP	65540 $(00010004)_{16}$	I/O エラーが発生した。 (P) ディスクおよび行制御データファイルの状態を調査する。
jstqldat jstqlctp	8	JSTQ_LDATL	16777221 $(01000005)_{16}$	指定できる行データの長さが制限を超えた。 (P) 行送りなどを指定して，1 行当たりのデータを少なくして再実行する。

関数名	リターンコード (lcomrc)	エラー詳細コード（lcomerr）		要因および対処
		エラー詳細	値	
		JSTQ_HDATL	1048581 $(00100005)_{16}$	
jstqlctp	8	JSTQ_CVALERR	16777220 $(01000004)_{16}$	文字列制御情報の指定が不正である。 (P) 文字サイズ，文字の間隔などの指定を確かめる。
各関数共通	8	JSTQ_PARMH	1048577 $(00100001)_{16}$	パラメタエラーが発生した。 (P) パラメタを確認して再実行する。
		JSTQ_PARML	16777217 $(01000001)_{16}$	
	8	JSTQ_LOGERR	1048578 $(00100002)_{16}$	ログファイルのアクセス中にエラーが発生した。 (P) ディスクおよびログファイルの状態を調査する。
	8	JSTQ_HDOFNOAR	1048579 $(00100003)_{16}$	エリアの確保時にメモリ不足が発生した。 (P) ほかの処理を終了させ，メモリを空けて再実行する。
		JSTQ_LDOFNOAR	16777219 $(01000003)_{16}$	
	8	JSTQ_ABNDPMOD	65541 $(00010005)_{16}$	ページ制御でエラーを検出した。 (P) ログファイルを出力して原因を調査する。
	8	JSTQ_NOFORM	65542 $(00010006)_{16}$	書式が存在しない。 (P) AP 環境ファイルで指定したフォルダに書式を作成する。
	8	JSTQ_HNOPN	1048582 $(00100006)_{16}$	書式機能がオープンされていない。 (P) 関数の呼び出し順序を確認する。
		JSTQ_LNOPN	16777222 $(01000006)_{16}$	
	8	JSTQ_FORMVERR	1048583 $(00100007)_{16}$	実行環境の XMAP3 より上位のバージョン（VV-RR）で作成された書式マップは使用できない。 (S) 処理を終了する。 (P) 実行環境の XMAP3 バージョン（VV-RR）を，書式マップの作成時に使用した XMAP3 と同じバージョン（VV-RR）にし，AP を再実行する。
	8	JSTQ_FORMEDNERR	1048588 $(0010000C)_{16}$	書式マップのエンディアンが実行環境と合っていない。 (S) 処理を終了する。 (P) 実行環境に合ったエンディアンとなるように，物理マップを再作成する。

関数名	リターンコード (lcomrc)	エラー詳細コード（lcomerr）		要因および対処
		エラー詳細	値	
	8	JSTQ_PMODVERR	65543 $(00010007)_{16}$	実行環境の XMAP3 より上位のバージョン（VV-RR）で作成された行制御データファイルは使用できない。 (S) 処理を終了する。 (P) 実行環境の XMAP3 バージョン（VV-RR）を，行制御データファイルの作成時に使用した XMAP3 と同じバージョン（VV-RR）にし，AP を再実行する。
	8	JSTQ_MAPNOAR	268435460 $(10000004)_{16}$	入出力制御関連エラー：エリア確保時にメモリ不足が発生した。 (P) ほかの処理を終了させ，メモリを空けて再実行する。
	8	JSTQ_MAPERR	268435461 $(10000005)_{16}$	入出力制御関連エラー：サービス名がない，または入出力制御内で論理矛盾などが発生した。 (P) • 表示・印刷セットアップまたはサービス名ファイル（XPWhosts）で指定したデバイス名およびプリンタ名が正しいか確認し，再実行する。 • ログファイルを出力して原因を調査する。
	8	JSTQ_MAPIERR	268435462 $(10000006)_{16}$	入出力制御関連エラー：プリンタがほかで使用中のため，排他エラーが発生した。または，スタンドアロン印刷の場合，同時に複数のプリンタをオープンした。 (P) プリンタの状態を確認して再実行する。スタンドアロン印刷の場合に複数のプリンタをオープンしている場合には，同時に複数のプリンタをオープンしないように AP を変更する。
	8	JSTQ_MAPERPR	268435466 $(1000000A)_{16}$	入出力制御関連エラー：プリンタで障害が発生した。 (P) プリンタの状態を確認して障害を回復後，再実行する。
	8	JSTQ_MAPSYSC	268435467 $(1000000B)_{16}$	入出力制御関連エラー：システムコールエラーが発生した。 (P) ログファイルを出力して原因を調査する。
	8	JSTQ_MAPDOWN	268435468 $(1000000C)_{16}$	入出力制御関連エラー：サービス名がない，または通信路が切断された。 (P) • 表示・印刷セットアップまたはサービス名ファイル（XPWhosts）で指定したデバイス名およびプリンタ名が正しいか確認し，再実行する。

関数名	リターンコード (lcomrc)	エラー詳細コード（lcomerr）		要因および対処
		エラー詳細	値	
				• 印刷サービスを起動し直して再実行する。

付録 F.3　WRITE 文使用時の COBOL メッセージと XMAP3 のエラーコードの対応

COBOL で作成した AP で WRITE 文を使用する場合に出力される，COBOL メッセージと XMAP3 のエラーコードの対応を次の表に示します。次の表から対応するエラーコードの要因と対処方法を確認してください。

なお，KBTQ523-E が出力された場合，KBTQ523-E のエラー出力より前に，ほかのエラーが発生していないかどうかを確認してください。

表 F-3　COBOL メッセージと XMAP3 のエラーコードの対応表

COBOL メッセージ		XMAP3 のエラーコード
KBTQ501-E	フィールド制御情報が不正です	$(01000004)_{16}$
KBTQ502-E	1 行内のデータが多すぎます	$(01000005)_{16}$
KBTQ503-E	パラメタ不正	$(01000001)_{16}$ $(00100001)_{16}$
KBTQ504-E	ログファイルアクセスエラー	$(00100002)_{16}$
KBTQ505-E	メモリ不足が発生しました	$(01000003)_{16}$ $(00100003)_{16}$
KBTQ506-E	ページ制御でエラーが発生しました (エラーコード,エラー詳細コード)	$(00010005)_{16}$
KBTQ508-E	行配置情報が存在しません	$(00010002)_{16}$
KBTQ509-E	行配置情報が不正です	$(00010003)_{16}$
KBTQ510-E	I/O エラーが発生しました	$(00010004)_{16}$
KBTQ513-E	入出力制御サーバが起動されていません	$(10000003)_{16}$
KBTQ514-E	入出力制御でメモリ不足が発生しました	$(10000004)_{16}$
KBTQ515-E	入出力制御でエラーが発生しました (エラーコード,エラー詳細コード)	$(10000005)_{16}$
KBTQ516-E	指定されたプリンタは使用できません	$(10000006)_{16}$
KBTQ520-E	プリンタに障害が発生しました	$(1000000A)_{16}$
KBTQ521-E	入出力制御との通信路が切断されました	$(1000000C)_{16}$
KBTQ522-E	入出力制御でシステムコールエラーが発生しました	$(1000000B)_{16}$

COBOL メッセージ		XMAP3 のエラーコード
KBTQ523-E	書式機能がオープンされていません	$(01000006)_{16}$
KBTQ524-E	書式が存在しません	$(00010006)_{16}$
KBTQ526-E	上位のバージョン（VV-RR）で作成された書式マップは使用できません	$(00100007)_{16}$
KBTQ527-E	上位のバージョン(VV-RR)で作成された行制御データファイルは使用できません (行制御データファイル VR：xx-xx-xx)	$(00010007)_{16}$
KBTQ532-E	書式マップのエンディアンが実行環境と合っていません	$(0010000C)_{16}$
メッセージ ID なし	XMAP3:specified pmap cannot be used(file version:xx)	$(0C20)_{16}$
メッセージ ID なし	XMAP3:specified pmap cannot be used(endian)	$(0C24)_{16}$

付録 G　各バージョンの変更内容

各バージョンの変更内容を示します。

変更内容（3020-7-513-50）XMAP3 Developer Version 5　05-04

追加・変更内容
適用 OS から Windows XP，Windows Vista，Windows Server 2003，Windows Server 2003 R2 を削除した。これらの OS のサポート中止に伴い，OS に関する製品の記述を削除した。
カーソルを位置づける行・列の位置を 10 進数で指定できるようにした。
ドローセットアップで，ドローセットアップ情報を格納するフォルダを指定できるようにした。
印刷ドキュメント名を環境変数「XMAP3_PRINT_DOCNAME」で指定できるようにした。

変更内容（3020-7-513-40）XMAP3 Developer Version 5　05-03

追加・変更内容
Windows 版 XMAP3 サーバ/クライアント実行環境（64 ビット）の説明を追加した。

変更内容（3020-7-513-30）XMAP3 Developer Version 5　05-02

追加・変更内容
OpenTP1 サーバと連携する場合のソースプログラムの記述についての説明を変更した。
TP1/Web 連携用の AP の構成で，COBOL を使用する場合の説明を変更した。
出力カーソル制御のマッピング規則で，行列（2 進）カーソルを指定した場合に，カーソル項目データ名の領域に正しいカーソル位置を代入したときの結果の説明を変更した。
帳票の可変部の出力に関する出力項目のマッピング規則で，出力項目データ名の領域にすべて空白を代入した場合の結果の説明を変更した。
次に示すキャラクタコントロールの表示属性の標準値，および候補選択コントロールの表示属性の標準値に印字属性の説明を追加した。 • 数字系・日本語系・カナ系・英数系キーエントリおよび日本語系選択エントリ • ラベル

変更内容（3020-7-513-20）XMAP3 Server Runtime Version 5　05-01

追加・変更内容
UNIX 版 XMAP3 Server Runtime に HP-UX 版 XMAP3 Server Runtime を追加した。

変更内容（3020-7-513-10）XMAP3 Developer Version 5　05-01，XMAP3 Server Runtime Version 5　05-01

追加・変更内容
UNIX 版 XMAP3 Server Runtime Version 5 をサポートした。
画面用の AP 開発について，UNIX 版 XMAP3 の説明を追加した。
帳票用の AP 開発について，UNIX 版 XMAP3 の説明を追加した。

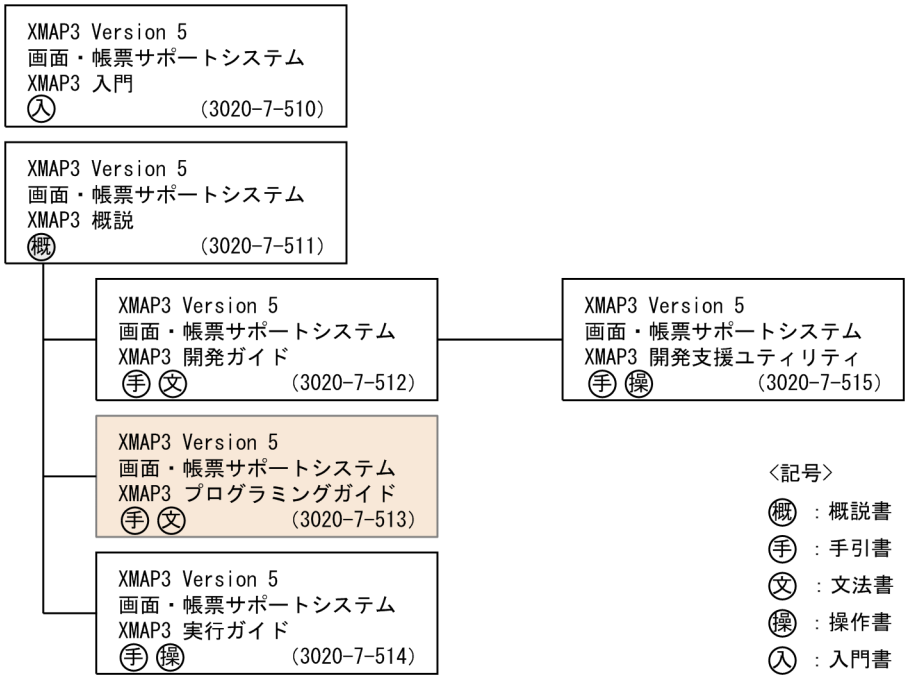
付録 H　このマニュアルの参考情報

このマニュアルを読むに当たっての参考情報を示します。

付録 H.1　関連マニュアル

関連マニュアルを次に示します。必要に応じてお読みください。

XMAP3 Version 5



Cosminexus Version 9

Cosminexus V9 アプリケーションサーバ & BPM/ESB 基盤 概説（3020-3-Y01）
Cosminexus V9 アプリケーションサーバ システム構築・運用ガイド（3020-3-Y02）
Cosminexus V9 アプリケーションサーバ システム設計ガイド（3020-3-Y04）
Cosminexus V9 アプリケーションサーバ アプリケーション開発ガイド（3020-3-Y20）
Cosminexus V9 アプリケーションサーバ リファレンス API 編（3020-3-Y21）

Cosminexus Version 8

Cosminexus アプリケーションサーバ V8 概説（3020-3-U01）
Cosminexus アプリケーションサーバ V8 システム設計ガイド（3020-3-U03）
Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド（3020-3-U04）
Cosminexus アプリケーションサーバ V8 アプリケーション開発ガイド（3020-3-U25）
Cosminexus アプリケーションサーバ V8 リファレンス API 編（3020-3-U26）

Cosminexus Version 7

Cosminexus 機能解説（3020-3-M03）
Cosminexus システム構築ガイド（3020-3-M06）
Cosminexus システム運用ガイド（3020-3-M07）
Cosminexus アプリケーション開発ガイド（3020-3-M41）

COBOL

COBOL2002 Professional 製品 導入ガイド（3020-3-C00）

COBOL2002 操作ガイド（3020-3-D41）

COBOL2002 ユーザーズガイド（3020-3-D42）

COBOL2002 言語 標準仕様編（3020-3-D44）

COBOL2002 言語 拡張仕様編（3020-3-D45）

COBOL2002 操作ガイド（3020-3-D47）

COBOL2002 ユーザーズガイド（3020-3-D48）

COBOL2002 操作ガイド（3020-3-D61）

COBOL2002 ユーザーズガイド（3020-3-D62）

COBOL2002 Cosminexus 連携機能ガイド（3020-3-D90）

SEWB+ Version 4

SEWB+/REPOSITORY 運用ガイド（3020-3-B81）

SEWB+/CONSTRUCTION アプリケーション開発ガイド（3020-3-B83）

SEWB+ Version 3

SEWB+/REPOSITORY 運用ガイド（3020-3-N81）

SEWB+/CONSTRUCTION アプリケーション開発ガイド（3020-3-N84）

Web サーバ

Hitachi Web Server（3020-3-M15）

Hitachi Web Server（3020-3-U17）

OpenTP1 Version 7

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 解説（3000-3-D50）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 プログラム作成の手引（3000-3-D51）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 システム定義（3000-3-D52）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 プログラム作成リファレンス COBOL 言語編（3000-3-D55）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 メッセージ（3000-3-D56）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/W，TP1/Client/P 編（3000-3-D58）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/J 編（3000-3-D59）

OpenTP1 インターネットゲートウェイ機能 TP1/Web 使用の手引（3000-3-D62）

OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 プロトコル TP1/NET/XMAP3 編（3000-3-D74）

OpenTP1 Version 6

TP1/COBOL adapter for Cosminexus ユーザーズガイド（3020-3-B08）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 解説（3000-3-941）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 プログラム作成の手引（3000-3-942）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 システム定義（3000-3-943）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 プログラム作成リファレンス COBOL 言語編（3000-3-946）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 メッセージ（3000-3-947）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/W，TP1/Client/P 編（3000-3-949）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/J 編（3000-3-950）

OpenTP1 インターネットゲートウェイ機能 TP1/Web 使用の手引（3000-3-953）

OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 プロトコル TP1/NET/XMAP3 編（3000-3-965）

注

このマニュアルの本文中の説明では，XMAP3，JP1 および OpenTP1 のマニュアル名のバージョン表記を省略しています。これらのマニュアルを参照するときは，該当するバージョンのマニュアルをご利用ください。

このマニュアルでのマニュアル名の表記

このマニュアルでは，XMAP3 の関連マニュアルを次のように表記しています。

表記	マニュアル名
XMAP3 入門	XMAP3 Version 5 画面・帳票サポートシステム XMAP3 入門
XMAP3 概説	XMAP3 Version 5 画面・帳票サポートシステム XMAP3 概説
XMAP3 開発ガイド	XMAP3 Version 5 画面・帳票サポートシステム XMAP3 開発ガイド
XMAP3 実行ガイド	XMAP3 Version 5 画面・帳票サポートシステム XMAP3 実行ガイド
XMAP3 開発支援ユティリティ	XMAP3 Version 5 画面・帳票サポートシステム XMAP3 開発支援ユティリティ

付録 H.2　このマニュアルでの表記

このマニュアルでは，製品名を次のように表記しています。

表記	製品名
ActiveX	ActiveX
Adobe Acrobat	Acrobat Reader
	Adobe Reader
COBOL2002 または COBOL	COBOL2002 Developer Professional
	COBOL2002 Developer Professional(64)
	COBOL2002 Net Client Suite
	COBOL2002 Net Client Runtime
	COBOL2002 Net Developer
	COBOL2002 Net Developer(64)
	COBOL2002 Net Server Suite
	COBOL2002 Net Server Suite(64)

表記		製品名
		COBOL2002 Net Server Runtime
		COBOL2002 Net Server Runtime(64)
Cosminexus アプリケーションサーバまたは Cosminexus		uCosminexus Application Server Standard
		uCosminexus Application Server Enterprise
		uCosminexus Application Service Platform
		uCosminexus Application Architect
		uCosminexus Developer Standard
		uCosminexus Developer Professional
Eclipse		Eclipse 3.1.1 または Eclipse 3.3.2 以降
Internet Explorer		Internet Explorer
		Windows Internet Explorer
Internet Information Services，または IIS		Microsoft Internet Information Services
IPF		Itanium Processor Family
JP1/AJS	JP1/AJS2	JP1/Automatic Job Management System 2 - Manager
		JP1/Automatic Job Management System 2 - Agent
		JP1/Automatic Job Management System 2 - View
		JP1/Automatic Job Management System 2 - Client Toolkit
	JP1/AJS3	JP1/Automatic Job Management System 3 - Manager
		JP1/Automatic Job Management System 3 - Agent
		JP1/Automatic Job Management System 3 - View
MyEclipse		MyEclipse for Cosminexus
OpenTP1	TP1/Client/J	uCosminexus TP1/Client/J
		TP1/Client/J
	TP1/Client/P	uCosminexus TP1/Client/P
		TP1/Client/P

表記			製品名
	TP1 Connector		uCosminexus TP1 Connector
			Cosminexus TP1 Connector
	TP1/LiNK		uCosminexus TP1/LiNK
			TP1/LiNK
	TP1/MCF		uCosminexus TP1/Message Control
			TP1/Message Control
			uCosminexus TP1/NET/Library
			TP1/NET/Library
			uCosminexus TP1/NET/XMAP3
			TP1/NET/XMAP3
	TP1/Server Base		uCosminexus TP1/Server Base
			TP1/Server Base
	TP1/Web		uCosminexus TP1/Web
			TP1/Web
PDF			Adobe PDF
Servlet，またはサーブレット			Java Servlet
UNIX	AIX		AIX V6.1
			AIX V7.1
			AIX V7.2
	HP-UX		HP-UX 11i V2(IPF)
			HP-UX 11i V3(IPF)
Visual C++	Visual C++ 2005		Microsoft Visual C++ 2005
	Visual C++ 2008		Microsoft Visual C++ 2008
	Visual C++ 2010		Microsoft Visual C++ 2010
	Visual C++ 2012		Microsoft Visual C++ 2012
	Visual C++ 2015		Microsoft Visual C++ 2015
VOS3			VOS3/AS
			VOS3/FS
			VOS3/LS
			VOS3/US
Windows	Windows 7	Windows 7 x86	Microsoft Windows 7 Professional 日本語版(32 ビット版)

表記			製品名
			Microsoft Windows 7 Enterprise 日本語版(32 ビット版)
			Microsoft Windows 7 Ultimate 日本語版(32 ビット版)
		Windows 7 x64	Microsoft Windows 7 Professional 日本語版(64 ビット版)
			Microsoft Windows 7 Enterprise 日本語版(64 ビット版)
			Microsoft Windows 7 Ultimate 日本語版(64 ビット版)
	Windows 8.1	Windows 8.1 x86	Windows 8.1 Pro 日本語版(32 ビット版)
			Windows 8.1 Enterprise 日本語版(32 ビット版)
		Windows 8.1 x64	Windows 8.1 Pro 日本語版(64 ビット版)
			Windows 8.1 Enterprise 日本語版(64 ビット版)
	Windows 10	Windows 10 x86	Windows 10 Pro 日本語版(32 ビット版)
			Windows 10 Enterprise 日本語版(32 ビット版)
		Windows 10 x64	Windows 10 Pro 日本語版(64 ビット版)
			Windows 10 Enterprise 日本語版(64 ビット版)
	Windows Server 2008	Windows Server 2008 x86	Microsoft Windows Server 2008 Standard 32-bit 日本語版
			Microsoft Windows Server 2008 Enterprise 32-bit 日本語版
		Windows Server 2008 x64	Microsoft Windows Server 2008 Standard 日本語版
			Microsoft Windows Server 2008 Enterprise 日本語版
	Windows Server 2008 R2		Microsoft Windows Server 2008 R2 Standard 日本語版
			Microsoft Windows Server 2008 R2 Enterprise 日本語版
			Microsoft Windows Server 2008 R2 Datacenter 日本語版

表記				製品名
	Windows Server 2012			Microsoft Windows Server 2012 Standard 日本語版
				Microsoft Windows Server 2012 Datacenter 日本語版
	Windows Server 2012 R2			Microsoft Windows Server 2012 R2 Standard 日本語版
				Microsoft Windows Server 2012 R2 Datacenter 日本語版
	Windows Server 2016			Windows Server 2016 Standard 日本語版
				Windows Server 2016 Datacenter 日本語版
	Windows Server 2019			Windows Server 2019 Standard 日本語版
				Windows Server 2019 Datacenter 日本語版
XMAP3 Version 5，または XMAP3	XMAP3 Developer または XMAP3 開発環境			XMAP3 Developer Version 5
	XMAP3 開発支援ユティリティ			XMAP3 Developer 開発支援ユティリティ Version 5
	XMAP3 実行環境	XMAP3 Server Runtime または XMAP3 サーバ実行環境	Windows x86 版 XMAP3 Server Runtime※1 または Windows 版 XMAP3 サーバ実行環境（32 ビット）※2	XMAP3 Server Runtime Version 5
			UNIX 版 XMAP3 Server Runtime または UNIX 版 XMAP3 サーバ実行環境	
			Windows x64 版 XMAP3 Server Runtime※1 または Windows 版 XMAP3 サーバ実行環境（64 ビット）※2	XMAP3 Server Runtime Version 5 with 64bit Adapter
		XMAP3 Client Runtime または XMAP3 クライアント実行環境	Windows x86 版 XMAP3 Client Runtime または Windows 版 XMAP3 クライアント実行環境（32 ビット）	XMAP3 Client Runtime Version 5

表記				製品名
			Windows x64 版 XMAP3 Client Runtime または Windows 版 XMAP3 クライアント実行環境（64 ビット）	XMAP3 Client Runtime Version 5 with 64bit Adapter
		XMAP3 Web 実行環境		XMAP3/Web for Cosminexus
		XMAP3 Cosminexus 連携		XMAP3/Web for Cosminexus の Cosminexus 連携
		XMAP3 TP1/Web 連携		XMAP3/Web for Cosminexus の TP1/Web 連携
XMAP3 Version 4	XMAP3/Enterprise Edition			XMAP3/Enterprise Edition Version 4
				XMAP3/Enterprise Edition Run Time System Version 4
	XMAP3/NET			XMAP3/NET Version 4
				XMAP3/NET Run Time System Version 4
	XMAP3/REPORT3			XMAP3/REPORT3 Version 4
				XMAP3/REPORT3 Run Time System Version 4
	XMAP3/Web			XMAP3/Web Version 4
				XMAP3/Web for Cosminexus (Version 4)

注 1

XMAP3 サーバ実行環境と XMAP3 クライアント実行環境を併記する場合は，XMAP3 サーバ/クライアント実行環境と表記します。

注 2

物理マップ，書式イメージファイルおよび行制御データファイルを定義体と表記します。

注※1

Windows x86 版 XMAP3 Server Runtime と，Windows x64 版 XMAP3 Server Runtime を総称して，Windows 版 XMAP3 Server Runtime と表記します。

注※2

Windows 版 XMAP3 サーバ実行環境（32 ビット）と，Windows 版 XMAP3 サーバ実行環境（64 ビット）を総称して，Windows 版 XMAP3 サーバ実行環境と表記します。

付録 H.3　KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

付録 I　用語解説

(英字)

AP（Application Program）

業務プログラムのことです。XMAP3 の AP は，画面・帳票の入出力処理を実行します。

API（Application Programming Interface）

関数，ライブラリなど，AP から製品の機能を呼び出すときのインタフェースのことです。XMAP3 の AP では，画面の入出力，および帳票の出力を実行するときに，API で XMAP3 の機能を呼び出します。

AP が受け取る項目

入力論理マップのデータ項目で，AP が受け取るデータを格納します。

AP が受け取るデータとその優先順位は次のとおりです。

1. キー入力データ／選択に応じた通知コード
2. [入力済み] 属性のオブジェクトの表示データ／［選択済み］属性の候補の通知コード
3. エラー通知文字
4. 初期値
5. 初期クリア文字

AP が渡す項目

出力論理マップのデータ項目で，AP が渡すデータ（AP 実行時に毎回変更する内容）を格納します。

AP 環境ファイル（X3MWDRV／XMAPdrv）

マッピングライブラリ実行時の各種情報を取得するための環境設定ファイルです。デバッグやチューニングのときに必要に応じて，ログの取得の有無と種類，物理マップのロードパス，物理マップの常駐サイズを設定します。

AP 環境ファイルの内容は，Windows 版 XMAP3 の場合は表示・印刷セットアップまたはロギング支援のプロパティで設定します。UNIX 版 XMAP3 の場合はテキストエディタで編集します。

AP 環境ファイルは，XMAP3 Server Runtime または XMAP3 Client Runtime で提供します。

格納場所とファイル名：

［Windows 版］*XMAP3 インストールフォルダ*¥ETC¥X3MWDRV

［UNIX 版］/etc/opt/HIXMAP/XMAPdrv

AP パターン・AP 部品

AP の定型的な型として XMAP3 が提供している COBOL 用と C 言語用のパターンです。AP パターンは，プログラムの標準的な骨組みです。AP 部品は，処理の手続きです。これらを組み合わせて利用すると，AP を効率良く開発でき，定型的なコーディングの統一も図れます。

格納場所：

［Windows 版］*XMAP3 インストールフォルダ*¥PATTERNS¥COBOL（または C）

［UNIX 版］/opt/HIXMAP/patterns

Bean

Java の用語で，部品化されたプログラムのことです。

C/S システム（Client / Server System）

サーバとクライアントで処理を分散する業務形態です。また，一つの処理をネットワーク上のサーバとクライアントで分担する機器構成モデル，または処理方式です。

C/S セットアップ

Windows サーバと Windows クライアントに設定された TCP/IP のホスト名，AP で指定する仮想端末名，および表示・印刷サービス名など，XMAP3 の C/S 構成での C/S システム環境を設定するファイルです。

C/S セットアップは，Windows 版 XMAP3 Server Runtime で提供します。

CD 項

通信記述項の略称です。XMAP3 で画面の入出力または帳票の出力をするために COBOL プログラム中にコーディングする論理端末定義ブロックです。物理マップ名や端末名などを指定します。

COBOL アクセス用 Bean

通信制御サーブレットから呼び出される Bean です。ユーザプログラムを呼び出します。

COBOL アクセス用 Bean は，XMAP3/Web for Cosminexus がサンプルプログラムとして提供しているデータ送受信用の登録集原文を基に，COBOL2002 の COBOL アクセス用 Bean 生成ツールを使用して作成します。

CPI（Character Per Inch）

文字間隔の単位で，25.4mm（1 インチ）当たり何文字印字できるかを表します。例えば，10CPI は 25.4mm（1 インチ）当たりに 10 文字になります。

CSV ファイル（Comma Separated Values）

データの区切りをコンマ（,）や改行で表すテキストデータを格納するファイルです。表計算ソフトで入力して，ユーザ独自の形式に編集できます。

CUI 画面（Character User Interface）

キーボードからの文字入力が中心の画面です。従来のメインフレーム型オンライン業務か，キャラクタベースの業務で使用していた画面です。

XMAP3 では，画面の入出力を物理マップと論理マップを使って実現しています。

DD（Deployment Descriptor）

Java プログラムなどを Web システムに配置するために必要な事項を記述したファイルです。配置するファイルの名前，タイプ，動作環境，セキュリティの仕様などが XML で記述されています。

dpi（dots per inch）

プリンタの印刷精度を示す数値で，25.4mm（1 インチ）当たりのドット数を指します。この数値が大きいプリンタほど，精密な印刷ができます。

Eclipse

Eclipse プロジェクト（eclipse.org）が提供するオープンソースの統合開発環境です。ソースコードの編集やデバッグなど，Java アプリケーションの開発効率を向上させる各種機能を備えています。

Cosminexus が提供する Eclipse プラグインを Eclipse に組み込むと，Eclipse からアプリケーション開発ツールを起動したり，開発したアプリケーションを実行したりできます。

ESC/P スルー印刷，ESC/P スルーモード（EPSON Standard Code for / Page printer）

シリアルプリンタ固有の印刷モードです。エプソン社のインパクト型ドットプリンタが該当します。

EUC（Extended UNIX Code）

UNIX で使用するマルチバイトコードです。

FAX 宛先ファイル

FAX 通信プログラムである FAX コネクションと連携して帳票を FAX 送信する場合に使用するファイルで，送信先の FAX 番号を指定します。XMAP3 実行環境のプリンタ構成ファイル（X3PPINF）に，次に示す設定をしたときに有効になります。

- 印刷モード：PrintType=6（日立 FAXC/SPOOL 出力）
- FAX 送信情報の副走査線密度：PrintDPI=01（ファイル）または PrintDPI=00（ノーマル）

GDI モード（Graphical Device Interface）

Windows モードのプリンタドライバを使用して印刷することをいいます。Windows 用のプリンタドライバが提供されていれば，このモードで印刷できます。GDI 対応のプリンタを Windows 対応プリンタといいます。

getter

Java の用語で，プロパティから値を取得するメソッドです。

GS1-128 バーコードの印字幅調整

GS1-128 バーコードを印刷するとき，バーの印字幅を調整することで，読み取り精度を上げることができます。プリンタ構成ファイル（X3PPINF）に，GS1-128 バーコード解像度と GS1-128 バーコード調整ドット数を設定することで実現できます。

GUI 画面（Graphical User Interface）

キーボードからの入力のほかに，マウスによる操作ができる画面です。Windows の「ボタン」「スクロール」「プルダウン・カスケード」「ポップアップ」などが使用できます。XMAP3 では，画面の入出力を物理マップと論理マップを使って実現しています。

Java 言語用ツール

ドローおよびドローセットアップで生成された C 言語用のファイルから，Web アプリケーションに必要なファイル（入力/出力データ用 XML 文書，定数用 XML 文書および動的変更用 XML 文書）を生成する機能です。Java 言語用ツールは，XMAP3 Developer で開発した画面・帳票を Cosminexus アプリケーションサーバ上で実行させる Web システム構成の AP を開発する場合に使用します。

Java 言語用ツールは，XMAP3 Developer で提供します。

JBuilder

Java2 プラットフォームをサポートし，Java アプリケーション，アプレット，Servlet，Bean などのビジュアル開発を実現する製品です。

LIPS スルーモード（LBP Image Processing System）

ページプリンタ固有の印刷モードです。キヤノン社のページプリンタが該当します。LIPS には，LIPSII+，LIPSIII，LIPSIV などがあります。

LPI（Line Per Inch）

行の間隔の単位で，25.4mm（1 インチ）当たり何行印字できるかを表します。XMAP3 で「ます目の設定」を「自由な設定」にした場合，3，4，6，8，10，12LPI の中から選べます。例えば，8LPI は 25.4mm（1 インチ）当たりが 8 行になります。

MAPPING MODE 句（マッピングオプション（マッピングモード））

COBOL の通信記述項にマッピングオプションを指定する領域です。次に示すマッピングオプションに対応した値を設定します。

マージ（論理マップと物理マップをマージ）：空白または 0

論理マップだけ：2

物理マップだけ：3

MyEclipse

Java アプリケーションの開発環境です。Eclipse で Java アプリケーションを開発するためのプラグインがまとめられています。MyEclipse を使用すると，Java アプリケーションの作成，J2EE サーバへのデプロイ，J2EE サーバの起動・停止などができます。また，フレームワークを使用したアプリケーションの開発や，アプリケーションの設計もできます。

Cosminexus では，MyEclipse の日本語版環境である MyEclipse for Cosminexus を提供しています。

OCR（Optical Character Recognition）

出力 OCR 用の文字でデータを印字するオブジェクトです。データは AP から指定し，論理マップを介して帳票に印字します。字間値は 10CPI 相当が標準です。

PDF（Portable Document Format）

Adobe Acrobat が扱う文書のファイル形式で，特定のプラットフォームに依存しないで文書を表示できます。XMAP3 では，ページプリンタ用帳票を PDF ファイルに出力できます。

PDL スルーモード（Page Description Language）

プリンタが持つ固有の印刷データ形式のページ記述言語で，プリンタドライバを使用しない（スルーレス）で印刷することをいいます。プリンタ制御言語の一つで，LIPS や ESC/P 対応の印刷モードです。LIPS は主にページプリンタ専用の印刷モードです。ESC/P は主にシリアルプリンタ専用の印刷モードになります。

RECEIVE

画面からデータを入力するときに，COBOL プログラムにコーディングする命令文です。

SEND

画面または帳票を出力するときに，COBOL プログラムにコーディングする命令文です。

Servlet

サーバ側で Java を実行させる方法の一つです。Servlet は，Web サーバと連携して，Web サーバに HTML 文書や画像ファイルを送るだけではなく，アプリケーションを実行し，その結果を HTML 文書として送り返す機能を提供します。

setter

Java の用語で，プロパティに値を設定するメソッドです。

SSL（Secure Sockets Layer）

暗号化および認証機能が付いた，サーバとクライアント間のネットワーク用の通信プロトコルです。通信データの暗号化や通信相手の認証などを行います。

TP1/COBOL アクセス用 Bean

OpenTP1 と連携したシステムの Web アプリケーションシステムで，通信制御サーブレットから呼び出される Bean です。ユーザプログラムを呼び出します。

TP1/COBOL アクセス用 Bean は，XMAP3/Web for Cosminexus がサンプルプログラムとして提供しているデータ送受信用の登録集原文を基に，TP1/COBOL adapter for Cosminexus の TP1/COBOL アクセス用 Bean 生成ツールを使用して作成します。

TRANSCEIVE

画面を出力してデータを AP に入力するときに，COBOL プログラムの中にコーディングする命令文です。

WAR ファイル（Web Archive）

Web アプリケーションの構成要素を JAR 形式に圧縮したファイルです。

Web コンテナサーバ

サーブレットエンジンモードで動作するサーバのことです。

Web サーバ

Web ブラウザとの間で HTML ファイルなどの文書を転送するための，アプリケーションレベルのプロトコルを制御する製品，または製品が動作するマシンのことです。Web サーバを構築する製品には，Cosminexus HTTP Server（Hitachi Web Server）と Internet Information Services があります。

Windows 対応プリンタ

Windows 用にプリンタドライバが提供されているプリンタです。XMAP3 では，GDI モードで印刷できるプリンタを Windows 対応プリンタと呼びます。

XMAP3 インストールフォルダ

インストール時に任意のフォルダを指定できます。指定を省略すると，次のフォルダにインストールされます。

- 32 ビット版の Windows の場合
  Windows フォルダのドライブ¥Program Files¥Hitachi¥Xmap3
- 64 ビット版の Windows の場合
  Windows フォルダのドライブ¥Program Files(x86)¥Hitachi¥Xmap3

XMAP3 サーバ

C/S 構成で運用する場合に，サーバ側で起動するプログラムです。サーバの OS が Windows の場合には，サービスとして起動することもできます。

XMAP3 入出力制御

AP からのデータを受けて，ディスプレイに表示（またはプリンタへ印字）したり，ディスプレイからデータを入力したりする機能です。

（ア行）

アクセスキー

メニューバー，ポップアップメニュー（ポップアップメニューファイルを使用する場合も含める），およびプッシュボタンボックスで，候補の選択操作をするためのキーです。

これらのオブジェクトにフォーカスが位置づいているときにアクセスキーを押すと，該当するメニュー項目やボタンを選択できます。また，［Alt］＋アクセスキーを押すと，メニューバーのメニュー項目を選択（プルダウンメニューを表示）できます。

網掛け帳票

240dpi/300dpi のページプリンタ用の帳票です。各種の文字サイズ，けい線，網掛けなどが使えます。

一次ウィンドウ

メインとなる画面です。業務を選択するメニューや，各業務の主画面として使用します。

一つの AP から同時に複数表示されることはなく，すでに表示している一次ウィンドウの消去後に次の一次ウィンドウが表示されます。

イベント通知コード

コマンドコントロールオブジェクト（プッシュボタン，メニューバー）や確定キー（PF キーなど）に割り当てるコードです。オブジェクトやキーを操作するとイベントが発生し，通知コードに対応づけた動作（AP 通知など）が実行されます。これを「INC 定数」といいます。また，入力単位がイベントのとき通知される「イベント定数」もあります。

イベント通知コードの値はドローセットアップで変更できます。動作はドローで変更できます。

印刷サービス名を指定する環境変数

出力先の印刷サービス名を指定します。COBOL の環境変数として指定する場合，「CBLX_外部装置名」で設定します。

XMAP3 の環境変数「XMAP3_PSNAME」で印刷サービス名を設定します。ただし，COBOL の環境変数，または COBOL の AP での指定がある場合は，XMAP3 の環境変数での指定は無効となります。

印刷ドキュメント名

帳票印刷時，Windows のプリンタスプールに登録される XMAP3 の印刷データのドキュメント名です。印刷ドキュメント名は帳票属性または書式属性として指定できます。また，帳票を印刷するときの出力論理マップ中にマップ帳票の印刷ド

キュメント名を指定すると，AP から動的に変更できます。この機能を利用すると，印刷した帳票の種別や内容を判別しやすくなり，帳票印刷業務でのリカバリ処理などを効率良く実行できます。

なお，帳票を PDF ファイルに出力する場合，印刷ドキュメント名が PDF ファイルのファイル名になります。

印刷枚数を指定する環境変数

XMAP3 の環境変数「XMAP3_COPIES」に同一帳票の複数枚印刷を指定できます。印刷枚数は 1～32 の範囲で指定します。省略すると，「1」が仮定されます。

印字領域

定義したオブジェクトを，実際に先頭印字位置から印刷できる領域です。プリンタの機種やプリンタにセットする用紙の位置などによって異なります。レイアウト定義では，印字領域内にレイアウトが収まるようにする必要があります。

ウィンドウ種別

XMAP3 で定義する画面には，一次ウィンドウと二次ウィンドウがあります。

一次ウィンドウは，メインとなる画面で，何も表示されていない状態のときに表示します。業務を選択するメニューや，各業務の主画面として使用します。原則として，一つの AP から同時に複数表示されることはなく，すでに表示されている一次ウィンドウを消去してから次の一次ウィンドウが表示されます。

二次ウィンドウは，一次ウィンドウを表示したまま新しく表示する画面です。データ入力の入力補助，メッセージ，ヘルプなどに使用します。一次ウィンドウに重ねて複数（最大 3 個）の二次ウィンドウを表示できます。このときの操作対象は，最後に表示した二次ウィンドウだけになります。

埋字

入力（または出力）されたデータが，AP が受け取る項目（または AP が渡す項目）の長さより短いとき，残りの領域を埋める文字です。右側と左側のどちらに埋字を格納するかは，桁寄せの指定に従います。ドローで作成したオブジェクトには，ドローセットアップで指定した埋字が仮定されます。必要に応じて，ドローのダイアログで変更できます。なお，埋字に指定できるのは半角文字だけです。埋字を「スペース」で指定した場合，漢字専用で使うオブジェクトでも 1 文字につき半角スペース二つが埋字されます。

上書きモード

キーボードから文字を入力する際に，すでにある文字に上書きしていく入力方式です。

XMAP3 では，「上書きモード」がデフォルトになっています。文字をカーソル位置に挿入する「挿入モード」をデフォルトにしたい場合は，表示・印刷セットアップで設定できます。

エラー通知文字

XMAP3 が入力データにエラーを検出した場合に，AP が受け取るデータです。

使用目的が「カナ」で詳細目的が「カナ・半角」（詳細目的は GUI 画面だけ）のエントリ系オブジェクトに対して，全角文字を入力したときに，入力データのエラーとなります。

また，属性の「空白入力」が「一部&全桁（半角）」または「一部（半角）」のエントリ系オブジェクトに対して，全角の空白を入力した場合も入力データのエラーとなります。ただし，XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）の，全角スペースコードの扱い（*.COSPCD=）オプションに 4040（半角スペース 2 個に変換して返す）を指定している場合は該当しません。

オフセット

シリアルプリンタで，用紙に穴を開けることなどを想定して設定しておく余白のことです。ハードマージンを基点としてレイアウト開始位置をずらせます。XMAP3 では，1 ます目から，レイアウト領域の範囲内でオフセット値を指定できます。

（カ行）

カーソル制御

カーソル制御は，ドローセットアップのカーソル・フォーカス制御で設定します。カーソル制御には，論理カーソル，行列（2 進）カーソル，行列（10 進）カーソルの 3 種類があります。

論理カーソルは，カーソル位置をカーソル定数で制御します。カーソル定数は XMAP3 が生成するため，画面のオブジェクト（フィールド）の位置を変更しても AP を変更する必要はありません。

カーソル定数

入力可能なフィールドにカーソルを位置づけるための定数です。カーソル定数は，例えばドローで AP が受け取る項目のデータ名を MAP001-FIELD0001- に設定した場合は，MAP001-FIELD0001-T の名称で論理マップ中に生成されます。カーソル定数と入力カーソル項目の値が一致するオブジェクトの位置が，画面上に位置づいたカーソルの位置として AP で認識されます。

下位項目

オブジェクトの入出力となる一つのデータ項目を階層化して，複数の項目に細分化できます。この場合，AP は細分化された下位項目を使ってデータを参照・格納できます。下位項目は，データ型が「文字（XX)」の場合に指定できます。

隠しフィールド

実際の画面には表示されないフィールドで，入力固定項目として使います。

入力固定項目とは，入力論理マップに必ず固定の値を返すデータ項目で，論理マップ可変部の最初のデータ項目として展開されます。

OLTP サーバ構成の環境で，使用するアプリケーション名を XMAP3 上で定義するには，隠しフィールドとして定義します。

仮想画面

画面ごとのデータを AP とやり取りするために，XMAP3 が持っている仮想的な画面です。AP から返されたデータは，仮想画面に展開されてから，実際のウィンドウに表示されます。ユーザの入力データも，仮想画面を介して AP に返されます。

仮想端末名ファイル（X3MWHOST／XMAPhosts）

AP がプログラム中で使用する仮想端末名，デバイス名，サービス名などを設定するファイルです。仮想端末名ファイルの内容は，Windows 版 XMAP3 の場合は表示・印刷セットアップの「プリンタ」，C/S セットアップの「C/S 構成」で設定します。UNIX 版 XMAP3 の場合は，テキストエディタで編集します。なお，スタンドアロン環境でプリンタを 1 台だけ使用している場合は，設定する必要はありません。

仮想端末名ファイルは，XMAP3 Server Runtime，XMAP3 Client Runtime，または XMAP3/Web for Cosminexus で提供します。

格納場所とファイル名：

[Windows 版]

- XMAP3 Server Runtime または XMAP3 Client Runtime の場合
  XMAP3 インストールフォルダ¥ETC¥X3MWHOST
- XMAP3/Web for Cosminexus（XMAP3 Cosminexus 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for Cosminexus¥ETC¥X3MWHOST
- XMAP3/Web for Cosminexus（XMAP3 TP1/Web 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for TP1¥ETC¥X3MWHOST

[UNIX 版]

/etc/opt/HIXMAP/XMAPhosts

カット紙

カット紙は，A4 サイズや B5 サイズなど 1 枚の紙になっている用紙です。ページプリンタ帳票とシリアルインパクト帳票の両方で利用できます。

壁紙

指定するグラフィックをレイアウト領域の各オブジェクトの背後に表示します。壁紙のグラフィックには，ビットマップファイル（.bmp）が使えます。レイアウト領域より小さい壁紙はタイル状に並べられて表示されます。レイアウト領域より大きい壁紙は，はみ出し部分（下側・右側）は表示されません。なお，壁紙はレイアウト領域には表示されません。開発時に確認する場合は，ドローからテスト表示を利用してください。

画面属性

入力項目の扱いや表示形態などの画面全体に関する属性です。GUI 画面では［画面属性］ダイアログで属性を指定します。また，画面属性は AP 実行時に動的に変更することもできます。AP から動的に変更するには，ドローセットアップの［表示属性の動的変更］で変更したい表示属性を定義します。ここで指定した修飾名を制御項目に代入することで，指定した表示属性に変更できます。

環境管理ファイル（xmap3.properties）

Web アプリケーションの実行に必要なファイルです。Web アプリケーションシステムを実行する場合に必要なファイルの格納先や，ログの出力先などを設定します。

開発する Web アプリケーションのシステムに合わせて，環境管理ファイルを更新します。

環境管理ファイルは，XMAP3/Web for Cosminexus で提供します。

環境ファイル操作

バックアップファイル（.sbk）を使って，セットアップ内容などの各種の情報を保存・復元します。XMAP3 の環境をほかの Windows マシンに移行する場合などに使います。開発環境のバックアップファイルは開発環境に，実行環境のバックアップファイルは実行環境にリストアできます。

開発環境用の環境ファイル操作は，XMAP3 Developer で提供します。

実行環境用の環境ファイル操作は，XMAP3 Server Runtime または XMAP3 Client Runtime で提供します。

キーエントリ

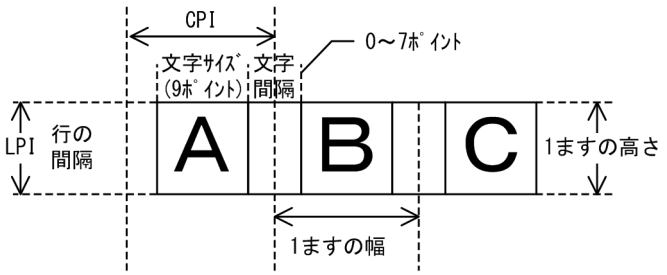
キーボードから文字列を入力するための各種オブジェクトの総称です。入出力テキスト・フィールド，入出力日付／時刻テキスト・フィールドが該当します。

基準ます目

帳票のレイアウト領域でオブジェクトを配置するときの単位となるます目です。ます目は，LPI と CPI の矩形を 1 ますとします。そのため，ドローでは，このます目を（LPI-CPI）の組み合わせで指定します。

LPI は，ます目の高さ（行間隔）です。XMAP3 で「ます目設定」を「自由な設定」にした場合，3，4，6，8LPI の中から選べます。例えば，8LPI は 25.4mm（1 インチ）当たりが 8 行になります。

CPI は，ます目の幅です。CPI を指定すると，自動的に文字サイズと文字間隔が決まります。例えば，10CPI と指定すると，25.4mm（1 インチ）に 10 文字格納できるように文字サイズと文字間隔が決まります。



上記，基準ます目によって，印字できる行列数が変わってきます。

帳票属性の定義では，「ます目設定」によって，配置できる領域が変わってきます。このます目設定は，レイアウト画面上に表示されているグリッドの間隔と対応します。

基準文字サイズ

GUI 画面の作業領域全体の基準となる文字サイズです。CUI 画面では変更できません。この値は，画面属性ダイアログの「基準文字サイズ」で設定します。

画面のレイアウト領域でオブジェクトを配置するときの単位となるグリッド（ます目）は，このサイズが基準になります。文字の縦幅の 1/2 がます目の縦幅，半角文字の幅がます目の横幅になります。

各オブジェクトでは，この基準文字サイズを基に「標準」「大」「小」の文字サイズ指定ができます。

起動 HTML

Web アプリケーションシステムの運用を開始するために必要な HTML ファイルです。起動 HTML は，起動 HTML 用スクリプトファイルと同じフォルダに格納します。

提供されている起動 HTML および起動 HTML 用スクリプトファイルのサンプルファイルを，カスタマイズして利用します。

行制御データファイル

書式オーバレイの使用時に，行データを印字する行の間隔，標準の文字サイズ，文字間隔などの情報を格納するファイルで，拡張子は.pci です。AP では，印刷するデータを 1 行（1 レコード）ずつ出力し，1 ページ分となった時点で書式がオーバレイされて印刷されます。

行データ

AP 中で設定する帳票の可変情報です。

業務サーブレット

通信処理および業務処理を記述した，Web サーバ側で動作するユーザの業務用 Java プログラムです。クライアントとの通信や，画面遷移，業務処理などを実行し，クライアントとの通信や受信・送信データの制御には，XMAP3 の実行クラスライブラリを利用します。

開発する Web アプリケーションのシステムに合わせて，画面遷移や業務処理用にクラスを利用するなど，任意の処理方法で作成できます。

空白入力

データ項目に半角の空白，全角の空白，またはそれらの混合の挿入を許すかどうかを指定します。選べる項目は，使用目的および詳細目的に応じて異なります。「禁止」を指定すると，空白の入力はできません。

グラフィック

グラフィックデータを出力するオブジェクトです。データには，ビットマップファイル（.bmp），メタファイル（.wmf），拡張メタファイル（.emf）が使えます。

出力するデータをドローで定義する「固定グラフィック」と，AP から指定する「出力グラフィック」があります。

固定グラフィックは，配置する前にあらかじめグラフィックファイルを用意しておきます。

出力グラフィックは，領域だけを配置します。グラフィックデータは，AP からファイル名またはクリップボード経由で渡します。

グラフィックコントロール

画面内のオブジェクトを整理・強調するための各種オブジェクトの総称です。

グラフィック，セパレータ，けい線，矩形，メッセージアイコン，塗りつぶしフィールドが該当します。

グラフィック帳票

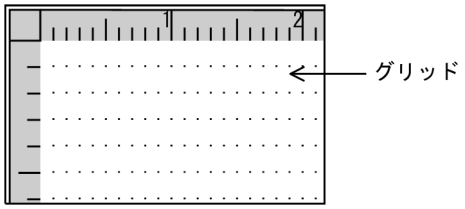
240dpi／300dpi のページプリンタ用の帳票です。網掛け帳票の機能に加え，オブジェクトやけい線などのバリエーションなどをより豊富に備えていて，文字のカラー印刷にも対応しています。また，複数のデータ項目を扱えるバーコード（GS1-128）を使用できます。

グリッド

ドローのレイアウト領域上の座標を示す格子状の線や点で，オブジェクト配置位置となります。格子一つ分の単位を「ます」といいます。

画面定義でのますのサイズは，画面属性で指定する基準文字サイズに従います。行方向は文字サイズの半分，列方向は半角文字の幅がそれぞれますの縦幅／横幅となります。

帳票定義でのますのサイズは，帳票属性で指定する基準ます目サイズに従う行／列単位と，ドット単位，mm 単位，ます目単位とで切り替えられます。ただし，シリアルインパクト帳票の場合は，行／列単位だけとなります。



グループコントロール

オブジェクト内に，ほかのオブジェクトを配置できる各種オブジェクトの総称です。グループボックス，フィールドボックス，フレームが該当します。

グループボックス

画面上のオブジェクト（セパレータを除く）を囲んでグループ化するオブジェクトです。必要に応じて，グループボックス単位にフォーカス遷移の順序を設定できます。

けい線帳票

180dpi のドットインパクトプリンタ用の帳票です。カット紙に対応するものと，連続紙に対応するものがあります。

OCR 文字も使えます。

桁

使用する文字数を半角換算で数えた単位です。例えば，出力テキストの属性設定用ダイアログで幅に「10 桁」とある場合は，そのオブジェクトに 10 文字（半角の場合）まで出力できることを示します。

なお，文字サイズを指定できるオブジェクトでは，現在の設定されている文字サイズでの文字数となります。

桁寄せ

テキストやフィールドに表示されるデータを右または左のどちらかに寄せるための項目です。次の場合に，データを右や左に寄せてデータを入出力します。

- 入力データ長が，AP が受け取る項目の長さより短い
- 出力データ長が，AP が渡す項目の長さより短い

後退キー（Back Space）

XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）の，後退キーの動作（表示サービス名.DCBKSP=）オプションの指定に対応して動作が次のように異なります。

- 直前の項目（BTAB）：テキスト・フィールドの先頭にカーソルを位置づけます。テキスト・フィールドの先頭では，前の位置にあるテキスト・フィールドにフォーカスが移動します。
- 1 文字削除（BSDEL）：直前の文字を削除し，間を詰めます。
- ヌル置換（BSNULL）：直前の文字をヌル（$(00)_{16}$）で置き換えます。

候補選択コントロール

画面に表示された各候補から対象を選ぶための各種オブジェクトの総称です。リストボックス，ラジオボタン，チェックボタン，トグルフィールドが該当します。

項目

AP と XMAP3 でやり取りする論理マップ中の各要素を項目といいます。画面や帳票に表示するユーザデータを扱う項目をデータ項目といい，表示属性を変更する目的で使用する項目を制御項目といいます。データ項目は，データ型によって次のように呼びます（COBOL の場合）。

- 文字項目：データ型が文字（XX）のときのデータ項目
- 数字項目：データ型が数字（99）のときのデータ項目
- 数字編集項目：データ型が数字編集（Z，9，¥などの組み合わせ）のときのデータ項目

- 漢字項目：データ型が日本語（NN）のときのデータ項目

固定テキスト・フィールド

固定の文字列を出力するオブジェクトです。文字列はドローで定義します。全角文字を使える「日本語」と，半角文字だけを使える「英数」があります。

固定バーコード

バーコードを印字するオブジェクトです。一つのバーコードで一つのデータ項目を表します。データはドロー上で指定して帳票に印字します。文字や長さに応じた各種があります。バーコードに付けるデータ文字の有無などを指定できます。

コマンドコントロール

選択操作で，イベントを発行して処理を実行するための各種オブジェクトの総称です。プッシュボタンとメニューバーが該当します。

コントロールメニュー

GUI 画面で，タイトルバー左端のアイコン（またはボタン）から表示される Windows のシステムメニューです。XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）の［閉じる］ボタン（表示サービス名.DCMPCS=）オプションの指定で，［閉じる］メニューを使用するかどうかを設定できます。

コンボボックス

候補（メニュー項目）を選択したり，データを直接，キー入力したりして，入力するデータを指定するオブジェクトです。必要な場合だけ入力するオプションを選択するときなどに使います。メニューは，付属しているボタンを選ぶと表示されます。メニュー項目をドローで定義する「固定コンボボックス」と，AP から指定する「可変コンボボックス」があります。また，GUI 画面では，メニュー選択とキーボードからの入力を併用する使い方ができる「キー入力固定コンボボックス」と「キー入力可変コンボボックス」もあります。

なお，ドローでは［コンボボックス］を使用するとメニュー選択するコンボボックスを定義でき，［キー入力コンボボックス］を使用すると，キー入力するコンボボックスが定義できます。これらを区別するため，メニュー選択するコンボボックスを「メニュー選択コンボボックス」，キー入力するコンボボックスを「キー入力コンボボックス」と表記し，これらの総称として「コンボボックス」と表記します。

（サ行）

サーバ AP 名ファイル（X3PAPL）

C/S 構成の場合に，クライアントの起動時に実行する AP の名前とサーバでの実際のファイル名との対応を設定するファイルです。サーバ AP 名ファイルの内容は，C/S セットアップの［アプリケーション］で設定します。

サーバ AP 名ファイルは，XMAP3 Server Runtime で提供します。

格納場所とファイル名：*XMAP3 インストールフォルダ*¥ETC¥X3PAPL

サーバ環境定義ファイル（X3WEBSRV）

Web 実行環境の場合に，各種の環境設定ファイルの格納先，実行時に Web クライアント側で必要なデータファイルの格納先などを定義するファイルです。

サーバ環境定義ファイルは，XMAP3/Web for Cosminexus の XMAP3 TP1/Web 連携機能で使用します。XMAP3 Cosminexus 連携機能の場合は不要です。

格納場所とファイル名：*XMAP3 インストールフォルダ*¥Web for TP1¥ETC¥X3WEBSRV

サーバ起動ファイル（X3PSERV）

Windows サービス上で動作する XMAP3 サーバの SERVICES ファイルのサービス名と，サービス名ファイル，および表示・印刷環境ファイルとの対応を設定するファイルです。サーバ起動ファイルの内容は，ファイルを開いてテキストエディタで直接設定します。

サーバ起動ファイルは，XMAP3 Server Runtime で提供します。

格納場所とファイル名：*XMAP3 インストールフォルダ*¥ETC¥X3PSERV

サービス名ファイル（X3PHOST／XPWhosts）

表示・印刷サービスに関連するサービス名やサービス種別などを設定するファイルです。この情報はサーバ側だけに設定します。サービス名ファイルには，Windows 版 XMAP3 の場合は C/S セットアップで設定した情報が反映されます。UNIX 版 XMAP3 の場合はテキストエディタで編集します。スタンドアロン環境の場合は，サービス名ファイルの設定は必要ありません。

サービス名ファイルは，XMAP3 Server Runtime または XMAP3 Client Runtime で提供します。

格納場所とファイル名：

［Windows 版］*XMAP3 インストールフォルダ*¥ETC¥X3PHOST

［UNIX 版］/etc/opt/HIXMAP/XPWhosts

再定義名

定義済みの論理マップを別定義として使用するときの名称です。半角で 30 字以内で定義します。

画面定義の場合，入力論理マップ用と出力論理マップ用とで別の名称にすることもできます。

削除キー（Delete）

入力できるテキスト・フィールドのデータを 1 文字ずつ削除するためのキーです。

支援ツール

次に示す XMAP3 の機能の総称です。

- 環境ファイル操作
- マップ生成
- セットアップ情報反映
- ポップアップメニューエディタ
- Java 言語用ツール

XMAP3 Developer では，すべての機能を提供します。

Windows 版 XMAP3 Server Runtime および XMAP3 Client Runtime では，環境ファイル操作を提供します。

時刻（入出力，出力）

時分秒を表示できます。時分秒は，時刻ダイアログ（出力／入出力）で指定します。「時」「分」「秒」の区切りには何を使うか，直前のゼロを表示するかしないかなど，どのような形式で表示するかは時刻ダイアログの選択肢で選べます。このような定義をしておくと，入力した数字データは，定義した形式になって表示されます。

下敷き

ドローでのプレプリント帳票，グラフィック帳票および書式オーバレイの定義で，用紙のフォーマットイメージをレイアウト領域の背後に表示しながら定義する機能です。用紙上のけい線などに合わせてオブジェクトの位置を調整する場合に使います。下敷きは，レイアウト時の目安となるだけで，実際の帳票印刷の対象にはなりません。

下敷きとして表示するデータは，あらかじめスキャナで読み込んで，256 色以下のビットマップファイル（.bmp）として用意しておきます。

修飾名

各画面およびグラフィック帳票で，表示属性の変更を AP から指示するための定数です。次の手順で利用します。

1. ドローセットアップで，修飾名と表示属性との対応を定義する。
2. ドローで，オブジェクトや画面の属性に「動的変更」を指定する。
   制御項目（修飾名を格納する領域）が論理マップに展開されます。
3. AP で，制御項目に修飾名を格納する。
   修飾名に設定された表示属性に従って，画面に表示（または帳票に印刷）されます。

出力 OCR

JIS 規格 OCR サブセット 2 の範囲の文字でデータを印字するオブジェクトです。データは AP から指定し，論理マップを介して帳票に印字します。字間値は 10CPI 相当が標準です。

出力テキスト・フィールド

出力データを AP から変更し，論理マップを介して帳票に出力するフィールドです。出力フィールドに出力できる文字には，数字だけを使用できる「数字」，全角文字を使用できる「日本語」，および半角文字だけを使用できる「英数」があります。

出力バーコード

バーコードを印字するオブジェクトです。一つのバーコードで一つのデータ項目を表します。データは AP から指定し，論理マップを介して帳票に印字します。文字や長さに応じた各種があります。バーコードに付けるコメント文字の有無などを指定できます。

バーコードの種類には，JAN8，JAN13，CODE39，ITF，NW-7，およびカスタマがあります。特別なハードウェアは必要ありません。

出力日付／時刻テキスト・フィールド

日付や時刻を表す文字列を出力するオブジェクトです。AP から指定する可変のデータを出力します。

GUI オブジェクトの「出力日付テキスト」と「出力時刻テキスト」，フィールドオブジェクトの「出力日付フィールド」と「出力時刻フィールド」があります。

ショートカットキー

メニューバーのメニュー項目の選択操作をするためのキーです。フォーカス・カーソルの位置に関係なく，［Ctrl］+半角英字 1 字を押すと該当するメニュー項目を選択できます。

なお，メニューバーのメニュー項目，またはカスケードメニューを持つプルダウンメニューにはショートカットキーを定義できません。

初期クリア文字

AP 実行時に，画面からの入力データを AP が受け取る前に，XMAP3 が入力論理マップの各項目をあらかじめクリアしておく文字です。標準値は埋字と同じ文字で，ドローセットアップで変更することもできます。

初期クリア文字に指定できるのは半角文字だけです。初期クリア文字に「スペース」を指定した場合は，漢字専用オブジェクトでも 1 文字につき半角スペース二つでクリアされます。

AP が受け取るデータの優先順位は次のとおりです。

1. キー入力データ／選択した通知コード（正常操作）
2. 入力済み／選択済みのとき，表示されているデータ／通知コード
3. エラー通知
4. データ消去通知文字
5. 初期値
6. 初期クリア文字

初期値

論理マップにあらかじめ設定しておく値です。

出力データの初期値は，出力論理マップの項目に設定します。AP 実行時に出力データの代わりにデータ有無コードを設定した場合はこの初期値が有効になります。

入力データの初期値は入力論理マップの項目に設定します。AP 実行時に画面からの入力データを AP が受け取る以前にはこの初期値が有効になります。

書式イメージファイル

書式オーバレイの文字列やけい線などの情報を格納するファイルで，拡張子は.fmp です。書式オーバレイの定義終了時に，行制御データファイルとともに生成されます。

書式オーバレイ

240dpi／300dpi のページプリンタ用の書式オーバレイです。定型帳票から書式（文字列やけい線などの固定項目）を分離し，アプリケーション（行データ）を印刷時に重ね合わせるソフトオーバレイ方式を採用しています。書式をプリンタ（ハードウェア側）に登録するメインフレーム環境の出力方式とは異なります。

書式定義ファイル

ドローで定義した書式の定義情報を格納したファイルで，拡張子は.ifm です。書式の定義終了時に生成され，書式イメージファイルと行制御データファイルはこの書式定義ファイルを基に生成されます。

書式名を指定する環境変数

XMAP3 の環境変数 XMAP3_FMP に，出力する書式名を指定できます。書式名は拡張子（.fmp）を除いた 8 文字以内の文字列で指定します。ただし，COBOL の AP で書式名が指定されている場合は，環境変数での指定は無効となります。

書体を指定する環境変数

XMAP3 の環境変数 XMAP3_FORMAT に，出力する行データの書体を指定できます。書体は，明朝体（1）またはゴシック体（2）で指定します。省略時にはドローで定義した書体が有効になります。

数字編集項目

指定した PICTURE 句に従って，数字を編集して入出力します。PICTURE 句は，次の文字で指定します。

入力：Ｓ Ｖ ９

出力：＊ ＋ - ￥ ９ . , Ｚ /

スクリプト環境ファイル（X3XSCONF）

スクリプト環境の設定ファイルです。スクリプト環境ファイルの定義には，テキストエディタを使用して直接編集します。スクリプトファイル環境ファイルは，起動 HTML の PARM タグの環境ファイルパスで指定したフォルダに格納します。

この環境設定ファイルは，XMAP3/Web for Cosminexus で提供します。

スクロール領域

ウィンドウやボックス内で，表示範囲を移動できる領域です。表示範囲の移動はスクロールバーで操作します。これに対し，表示範囲の移動対象外の領域を「スクロール禁止領域」といい，ウィンドウやボックス内に常に表示されます。

スピンボックス

入出力テキストの右端に，テキストに表示されている数値の増減をコントロールするステッパが付いているボックスです。増分値を指定すると，指定した間隔で表示されている数値が増減します。

セパレータ

画面上の各オブジェクトの間を区切るときに使う GUI 画面用の線オブジェクトです。縦線と横線が使えます。

選択エントリ

選択操作で文字列を入力するための各種オブジェクトの総称です。

コンボボックス，ポップアップ，スピンボックスが該当します。

挿入キー（Insert）

テキスト・フィールド上の文字列の間に，文字を挿入するための半角 1 文字分を空けます。挿入した空きに文字を書き込まないときは，AP が受け取るデータ上で空きが詰められます。

XMAP3 実行環境の環境設定時に，表示・印刷環境ファイル（X3PCONF／XPWconfig）の次に示すオプションで，挿入キーの動作を変更できます。

- 挿入キーの動作（表示サービス名.DCINMD=）オプション
- 挿入/上書きモードのデフォルト設定（表示サービス名.DCINST=）オプション
- 文字挿入の範囲（表示サービス名.DCINKY=）オプション

挿入モード

キーボードから文字を入力する際に，カーソルの位置に文字を挿入し，すでにある文字を入力文字数分だけ移動させる入力方式です。

XMAP3 では，すでにある文字に上書きする「上書きモード」がデフォルトになっています。「挿入モード」をデフォルトにしたい場合は，表示・印刷セットアップで設定できます。

（タ行）

ターゲット環境

XMAP3 で画面・帳票を定義するときに選択する環境です。定義した画面・帳票を運用する環境に応じて，次のターゲット環境があります。

- メインフレーム-PC 分散用の画面・帳票開発
- Windows 用の画面・帳票開発

チェックボタン

入力するデータを候補（ボタン）の選択によって指定するオブジェクトです。グループ内の複数のボタンから複数の選択ができます。ラベル長や選択肢の数が少ないときに使います。

ボタンラベルをドローで定義する「固定チェックボタン」と，AP から指定する「可変チェックボタン」があります。

チェックボタンのグループを表すボックスを「チェックボタンボックス」といいます。

通信記述項（CD 項）

XMAP3 で画面の入出力，および帳票の出力をするために COBOL プログラム中にコーディングする論理端末定義ブロックです。物理マップ名や仮想端末名などを指定します。

画面定義の例

```
CD DSP FOR I-O WS
MAP NAME          IS 画面マップ名      ：物理マップを格納する領域
SYMBOLIC TERMINAL IS 画面端末名        ：仮想端末名を格納する領域
MAPPING MODE      IS マッピングモード  ：マッピングオプションを格納する領域
STATUS KEY        IS 画面-RC.          ：画面の入出力が正しく実行されたかを判定する領域
```

帳票定義の例

```
CD PRT FOR OUTPUT WS
MAP NAME          IS 帳票マップ名      ：物理マップを格納する領域
SYMBOLIC TERMINAL IS 帳票端末名        ：仮想端末名を格納する領域
MAPPING MODE      IS マッピングモード  ：マッピングオプションを格納する領域
STATUS KEY        IS 帳票-RC.          ：帳票への出力が正しく実行されたかを判定する領域
```

通信制御サーブレット

Cosminexus 連携を利用した Web システム構成で，クライアントとユーザプログラム間のデータの受け渡しを制御する Servlet です。通信制御サーブレットは，COBOL アクセス用 Bean を経由して，Web ブラウザから受信したデータを AP に渡し，AP の処理結果を Web ブラウザへ返信します。

通信制御用 XML 文書（X3COMTBL.xml）

XMAP3/Web for Cosminexus で提供する実行クラスライブラリを使用するための参照ファイルです。受信・送信データの形式を定義する XMAP3 通信制御用領域で，仮想端末名や印刷完了通知，終了通知などの情報を取得・設定するために利用します。

Java でプログラミングするときに，XML ビューアなどで XML 文書のデータ名，形式，長さなどを参照できます。このファイルの形式や内容は変更できません。

通知コード（イベント通知コード）

コマンドコントロールオブジェクト（プッシュボタン，メニューバー）や確定キー（PF キーなど）に割り当てるコードです。オブジェクトやキーを操作するとイベントが発生し，通知コードに対応づけた動作（AP 通知など）が実行されます。

イベント通知コードの値はドローセットアップで変更できます。動作はドローで変更できます。

通知コード（データ入力用）

選択エントリ（スピンボックスを除く），候補選択コントロールオブジェクトの各候補（メニュー項目など）に割り当てるコードです。

メニュー項目の選択操作などで指定した候補に応じて，対応する通知コードが，AP が受け取る項目に格納されます。

定義体

画面や帳票のレイアウトに関する情報を格納した，物理マップ，書式イメージファイルおよび行制御データファイルの総称です。

定義パターン

ドローで画面を新規作成するときに，用途に応じて指定するパターンです。標準的なレイアウト領域のサイズやパターンなどがあらかじめ設定されていて，ドローでのレイアウト定義の基本フォーマットとなります。

XMAP3 が標準提供している定義パターンのほかに，ユーザが独自に作成・登録することもできます。

定数ファイル

画面表示でカーソルやフォーカスなどの定数を定義するファイルです。ドローセットアップの論理マップ属性で「定数部の別ファイル出力」を指定した場合，ドローで生成されます。定数ファイルは，定数用 XML 文書の生成に必要なファイルです。

データ有無コード

AP（出力論理マップのフィールドのデータ名の領域）にデータが設定されていないことを示す 1 バイトの文字コードです。$(00)_{16}$～$(FF)_{16}$ の任意の値で，通常は$(1F)_{16}$ が標準値です。

コードを変更する場合は，使用する言語に応じて次のとおりになります。

- COBOL の場合
  SEND／RECEIVE のとき，通信記述項の DATA ABSENCE CODE 句
  CALL のとき，オープン要求のインタフェース
- C 言語の場合
  jsvwadrv 関数のオープン要求インタフェース

データ型

COBOL の PICTURE 句で指定するデータの型です。文字型や数字型があります。C の場合は，char だけです。ドローで作成したオブジェクトには，ドローセットアップで指定したデータ型が仮定されます。必要に応じて，ドローで変更できます。

データ型は，文字用，数字用，数字編集用に分かれます。使用できるデータ型は次のとおりです。

入力用のデータ型

- 文字用：文字（XX），漢字（NN），漢字（XX）
- 数字用：99999
- 数字編集用：S99999，999V9 など

出力用のデータ型

- 文字用：文字（XX），漢字（NN），漢字（XX）
- 数字用：99999
- 数字編集用：Z，9，¥などを組み合わせた各種数字編集文字列

また，登録されていないデータ型を指定したい場合，［自由な設定］ボタンで表示されるダイアログで任意に設定できます。

データキー（Ctrl + End）

画面上のすべての入力テキスト（または入力フィールド）の文字を消去するためのキーです。データクリアキーともいいます。XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）の，データキーの動作（表示サービス名.DCDTTF=）オプションの指定で，データキーの動作を変更できます。

データ消去通知文字

AP 実行時に，画面からフィールドキーで入出力テキスト・フィールドのデータを消去したり，$(00)_{16}$ のデータを受信したりしたときに AP が受け取るデータです。したがって，AP が受け取ったデータがデータ消去通知文字であれば，画面操作でデータ消去されたと判断できます。

初期クリア文字や初期値とデータ消去通知文字とを分けておくことで，入力操作がなかったのか，データ消去されたのかが区別できます。

ただし，「入力済み」属性のオブジェクトで，画面確定時に入力データや表示データがない場合は，データ消去されていなくてもデータ消去通知文字が返ります。

また，データ消去通知文字に指定できるのは半角文字だけです。データ消去通知文字に「スペース」を指定した場合は，漢字専用オブジェクトでも 1 文字につき半角スペース二つが通知されます。

データ消去通知文字の標準は LOW$(00)_{16}$ で，ドローセットアップで変更することもできます。

データ長

論理マップの長さ（バイト数）です。通常は，桁を基に各ターゲットに応じた値が自動的に計算されます。ユーザ任意の値を設定できるターゲットもあります。

データ名，制御項目データ名

データ項目名（またはデータ名）は，データの入出力に関する各項目の領域の名称です。データ名のうち，修飾名（表示属性の変更を AP から指示するための定数）を格納する領域の名称を制御項目データ名といいます。

デバイス ID

マップ名に付けられる ID で，次の対応で画面・帳票の種別を表します。

6.は 6 桁のマップ名の場合，7.は 7 桁のマップ名の場合です。

画面・帳票の種別	6 桁のマップ名	7 桁のマップ名
GUI 画面	6.ND	7.O
CUI 画面	6.NC	7.S
けい線帳票	6.6A	7.P
プレプリント帳票	6.6H	7.L
網掛け帳票	6.6B	7.R
グラフィック帳票	6.6G	7.G
書式オーバレイ	6.6G	7.F

デリミタ線

複数の短い縦けい線をます目ごとに横に並べて配置するオブジェクトです。1 桁ごとに区切る目盛りを描画するときに使います。

動的変更テーブル

画面や帳票の実行時に色などの属性を AP から変更するときに，XMAP3 が用意する属性変更用の定数テーブルの修飾名を表示属性の動的変更制御項目に代入します。

登録集原文

COBOL のプログラム中でよく利用される標準化した手続き，ファイル記述，レコード記述，または完全な一つのプログラムなどをコンパイルするプログラムと別のファイルに登録したものです。

トグルフィールド

「YES/NO」などの印を付けるためのオブジェクトです。表の中に配置して，「オン／オフ」，「済／未」などを表す場合に使います。

ドロー

XMAP3 の画面や帳票を定義するエディタです。画面や帳票のレイアウトや，各オブジェクトの属性を定義します。ドローで定義した内容を基に，マップ定義ファイル（書式オーバレイでは書式定義ファイル），物理マップと論理マップ（書式オーバレイでは行制御データファイルと書式イメージファイル）が作成されます。

ドローセットアップ

XMAP3 の画面や帳票の定義に関する標準の値をカスタマイズする機能です。XMAP3 にはセットアップ項目の標準値があり，この標準値を変更する場合や，動的変更で修飾名を追加したい場合にドローセットアップを実行します。セットアップ項目はマップの形式に関係しているので，画面や帳票を作成する前にドローセットアップを済ませておく必要があります。

ドローセットアップは，XMAP3 Developer で提供します。

（ナ行）

二次ウィンドウ

データ入力補助，メッセージ，ヘルプなどのために，一次ウィンドウのデータを表示したまま新しく表示する画面です。一次ウィンドウに重ねて表示され，このとき操作できるのは二次ウィンドウだけとなります。

入出力テキスト・フィールド

キーボードなどから文字データを入力するオブジェクトです。また，AP から指定する文字データを表示できます。初期表示値（出力データ）を示し，書き換えが必要な場合だけ入力する使い方ができます。入出力するデータの意味に応じて，種類（使用目的）を選べます。

入出力日付／時刻テキスト・フィールド

日付や時刻を表す文字列を入出力するキーエントリオブジェクトです。キーボードなどから文字を入力します。また，AP から指定する可変のデータを表示できます。

表示されている出力データを必要に応じて書き換えて入力する使い方ができます。

GUI オブジェクトの「入出力日付テキスト」と「入出力時刻テキスト」，フィールドオブジェクトの「入出力日付フィールド」と「入出力時刻フィールド」があります。

入力単位（画面／フィールド／イベント／表示直後）

画面を確定させ，AP へ制御を渡すタイミングの種類です。通常は，画面の最後までデータを入力または選択したあとで，確定の機能を持つキーやボタンを押して，AP へ制御を渡します（画面）。「フィールド」では，フォーカスが位置づいているオブジェクトでデータの入力，選択を行い，そのオブジェクトからフォーカスが離脱した時点で AP へ制御が渡ります。「イベント」では，画面上のすべてのオブジェクトを対象にして，何かイベントが起これば AP に制御が渡ります。「表示直後」では，一定の待機時間が過ぎると AP に制御が渡されます。

入力データ長格納領域

入力テキストなどの画面上で入力可能な長さに対して，実際にどのくらいのデータが入力されたかの長さをオブジェクト単位に入力論理マップとして返す領域です。

塗りつぶしフィールド

フィールドボックス内の領域に色を付けるオブジェクトです。ほかのオブジェクトを重ねて配置できます。1 行おきに色のパターンを付けて行を見やすくする場合などに使います。

（ハ行）

ハードマージン

プリンタが紙を送るために必要な領域で，ハードウェア（プリンタ）に依存します。したがって，ユーザが任意に設定できません。また，GDI モードと LIPS スルーで次の違いがあります。

- GDI モードの場合，プリンタドライバが持つマージン
- LIPS スルーの場合，物理的に印字できない領域

ただし，プリンタドライバが持つマージンが，印刷するプリンタのハードマージンと同じかどうかは機種によって異なります。一般には，プリンタのハードマージンより大きく取られます。なお，ユーザがドローで指定できるマージンをソフトマージンまたは単にマージンといいます。

背景色

入出力テキストのボックス内の色やボタンボックス内の色です。背景色は，16 色から選べます。色の設定は，ドローで選択した各オブジェクトの属性ダイアログのほかに，ドローのツールバーの［文字色/背景色］ボタンをクリックして表示される文字色/背景色のツールボックスで変更できます。このツールボックスを使うと画面全体の配色を見ながら設定できます。

反転表示

この属性を指定すると文字色と背景色を反転させて表示します。したがって，何も入力されていない状態では，通常指定した文字色と背景色になり，入力が確定した時点で反転して表示されます。動的変更を利用して，反転表示を AP から指定することもできます。

反復定義

一つのオブジェクトを縦または横方向に繰り返す定義で，表形式で同じ属性のオブジェクトを並べる場合などに使います。

反復定義を使うと，同じ属性のオブジェクトを一つずつ配置するよりも効率良く定義できます。また，論理マップは配列（COBOL では OCCURS 句）で展開されるので，AP のロジックも反復指定で実現できます。

ビッグエンディアン

バイナリ形式のデータを扱う際，メモリ上に左から右側へ 1 バイトずつ格納する形式です。

（例）

$(000A)_{16}$ を 2 バイトのメモリに格納した場合は，$(000A)_{16}$ になります。

日付（入出力，出力）

年月日を表示できます。年月日は，日付ダイアログ（出力／入出力）で指定します。「年」を表すのに西暦にするか和暦にするか，「年」「月」「日」の区切りには何を使うか，直前のゼロを表示するかしないかなど，どのような形式で表示するかは日付ダイアログの選択肢で選べます。このような定義をしておくと，入力した数字データは，定義した形式になって表示されます。

表示・印刷環境ファイル（X3PCONF／XPWconfig）

XMAP3 での画面表示，および帳票印刷環境の設定ファイルです。表示・印刷環境ファイルの内容は，Windows 版 XMAP3 の場合は表示・印刷セットアップで設定します。UNIX 版 XMAP3 の場合は，テキストエディタで編集して設定します。

表示・印刷環境ファイルには，Windows 版 XMAP3 の表示・印刷セットアップで設定できない項目があります。これらの項目は，直接ファイルをエディタで開いて編集します。なお，エディタでパラメタを設定した場合でも，表示・印刷セットアップで設定できる項目については，セットアップでの設定が優先されます。

表示・印刷環境ファイルは，XMAP3 Server Runtime，XMAP3 Client Runtime，または XMAP3/Web for Cosminexus で提供します。

格納場所とファイル名：

［Windows 版］

- XMAP3 Server Runtime または XMAP3 Client Runtime の場合
  XMAP3 インストールフォルダ¥ETC¥X3PCONF
- XMAP3/Web for Cosminexus（XMAP3 Cosminexus 連携機能）の場合

XMAP3 インストールフォルダ¥Web for Cosminexus¥ETC¥X3PCONF

- XMAP3/Web for Cosminexus（XMAP3 TP1/Web 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for TP1¥ETC¥X3PCONF

[UNIX 版]

/etc/opt/HIXMAP/XPWconfig

表示・印刷セットアップ

XMAP3 が表示・印刷する画面・帳票の環境を必要に応じて設定するための機能です。ユーザ画面の外観や操作キーの割り当て，プリンタの印刷モードなどの各種設定があります。

表示・印刷セットアップは，Windows 版 XMAP3 Server Runtime，XMAP3 Client Runtime，および Windows 版 XMAP3/Web for Cosminexus で提供します。

表示形態

画面を表示するとき，直前に表示した画面に対して一部だけを書き換える（部分描画）か，全部を書き換える（全画面描画）かを指定する属性です。この値は，ドローのダイアログで値を設定します。また，ドローセットアップの「表示属性の動的変更」の「ウィンドウ属性」で変更したい「表示形態」の属性を指定した修飾名を用意しておくと，AP から動的に表示形態を変更できます。表示形態には，次の 3 種類があります。

- 一部上書
  直前に表示した画面の一部の項目だけを変更して表示します。書き換えに比べて表示時間を短縮できます。この場合，マッピングモードとデータ有無コードを合わせて使う必要があります。
- 全面書換
  表示中の画面を消去して，次の画面を全画面描画して表示します。
- 自動
  上書きと書き換えを XMAP3 に任せ，AP では特に意識しません。直前の画面と同じマップ名のときは部分描画になります。

表示属性

画面での，テキストやフィールドに出力する文字色やボタンの活性/不活性などを指定する属性，および帳票での，フィールドの文字の書体やけい線の種類などを指定する属性のことです。ドローでは，必要に応じて各ダイアログで変更できます。また，表示属性は定義上の指定のほかに，AP から動的に変更できる属性もあります。AP から動的に表示属性を変更するためには，ドローセットアップの動的変更を指定する修飾名で指定します。

フィールドキー（End）

テキスト・フィールドで，カーソル以降の文字を削除します（標準の場合）。

XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）のフィールドキーの動作（表示サービス名.DCFCLR=）オプションの指定で，割り当てるキーの動作を変更できます。

フィールドボックス

多量のデータをまとめて入出力するためのボックスオブジェクトです。次のような場合に適しています。

- けい線を使って，表形式でオブジェクトを並べる
- CUI 画面を基本にして一部だけを GUI 化（ファンクションキーによる選択部分だけのボタン化など）
- 文章データなど，複数行にわたる文字データの表示

フォーカス

GUI 画面を表示した際，操作対象となるオブジェクトに表示される枠です。AP から操作対象となるオブジェクトを選択することを，AP からフォーカスを位置づける，といいます。

フォーカス制御

フォーカス位置の情報を制御するための制御項目です。AP では，フォーカスを設定したいオブジェクトに対応するフォーカス定数を，制御項目に格納して渡します。また，画面上でフォーカスが位置づいているオブジェクトに対応したフォーカス

定数を，制御項目で受け取ります。なお，フォーカス定数は，XMAP3 が生成します。このため，画面のオブジェクトの位置を変更しても，AP を変更する必要はありません。

フォーカス定数

入力や選択ができるボックスにフォーカスを位置づけるための定数です。フォーカス定数は，例えばドローで AP が受け取る項目のデータ名を MAP001-FIELD0001- に設定した場合は，MAP001-FIELD0001-T の名称で論理マップ中に生成されます。フォーカス定数と入力フォーカス項目の値が一致するオブジェクトの位置が，画面上に位置づいたフォーカスの位置として AP で認識されます。

フォント構成ファイル（X3PFONT）

XMAP3 で作成した画面上に表示する文字のフォントやサイズの環境設定ファイルです。フォント構成ファイルの内容は，表示・印刷セットアップの「表示文字」で設定します。

フォント構成ファイルは，Windows 版 XMAP3 Server Runtime，XMAP3 Client Runtime，および XMAP3/Web for Cosminexus で提供します。

格納場所とファイル名：

- XMAP3 Server Runtime または XMAP3 Client Runtime の場合
  XMAP3 インストールフォルダ¥ETC¥X3PFONT
- XMAP3/Web for Cosminexus（XMAP3 Cosminexus 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for Cosminexus¥ETC¥X3PFONT
- XMAP3/Web for Cosminexus（XMAP3 TP1/Web 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for TP1¥ETC¥X3PFONT

不活性（選択できない状態にする）

候補（ボタンやメニュー）を選択できない状態にします。表示方法と組み合わせて，選べる候補を制限したい場合に使います。AP 実行中に選択可能な状態に戻したい場合は，該当するダイアログで「動的変更」を指定し，AP から制御項目に活性を指定した修飾名（ドローセットアップで指定）を指定することで実現できます。

複数行のフィールド

複数行にわたるサイズのフィールドです。フィールドの長さが 1 行に収まらない（レイアウト領域の右端からはみ出す）場合に，次の行に折り返されます。

GUI 画面では，フィールドボックス中のフィールド（固定/出力/入出力）とポップアップで使えます。

CUI 画面では，フィールド（固定/出力/入出力）で使えます。なお，反復定義や文字の拡大との併用はできません。

プッシュボタン

イベント（実行する処理）を候補（ボタン）の選択によって指定するオブジェクトです。イベントに対応する処理として「AP 通知」や「ポップアップ表示」などを指定できます。確定キーの代わりとしても使えます。

ボタンラベルはドローで定義し，AP 実行時に変更できます。ボタンラベルには，文字列のほかにグラフィックを表示することもできます。

プッシュボタンのグループを表すボックスを「プッシュボタンボックス」といいます。

物理画面

実際のディスプレイ画面です。各ウィンドウは，最終的にこの物理画面に表示されます。

物理マップ

画面のオブジェクトの位置などを格納した情報ファイルです。ファイル名はマップ名にデバイス ID（マップの定義対象を示す 1 文字または 2 文字の英数字）を付けた名前で，拡張子は.pmp です。画面の定義終了時に，論理マップとともに生成されます。ディスプレイやプリンタの入出力データを論理マップデータに変換したり，論理マップデータを入出力データに変換したりするときに XMAP3 が参照します。

プリンタ構成ファイル（X3PPINF）

プリンタ構成を設定する環境設定ファイルです。このファイルの内容は，次の方法で設定します。

- Windows 版 XMAP3 Server Runtime または XMAP3 Client Runtime の表示・印刷セットアップの「プリンタ」タブ
- Windows 版 XMAP3 Server Runtime で提供する印刷拡張セットアップまたは XMAP3 Client Runtime のオプション製品

プリンタ構成ファイルは，Windows 版 XMAP3 Server Runtime，XMAP3 Client Runtime，および XMAP3/Web for Cosminexus で提供します。

格納場所とファイル名：

- XMAP3 Server Runtime または XMAP3 Client Runtime の場合
  XMAP3 インストールフォルダ¥ ETC¥X3PPINF
- XMAP3/Web for Cosminexus（XMAP3 Cosminexus 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for Cosminexus¥ETC¥X3PPINF
- XMAP3/Web for Cosminexus（XMAP3 TP1/Web 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for TP1¥ETC¥X3PPINF

フレーム

複数のオブジェクトの組み合わせを縦方向または横方向に反復して並べるためのオブジェクトです。テキスト・フィールドなどを組み合わせて表形式で並べる場合に使います。論理マップには配列として展開されます。

プレーン

フィールドボックス，リストボックス，ボタンボックスが持つ領域で，ボックス内に表示するフィールドやボタンなどを配置するための平面です。

プレーンの属性の文字サイズは，プレーン内で共通の配置単位や文字サイズとなり，中のすべてのオブジェクトに適用されます。

フィールドボックスとリストボックスでは，ボックスよりも大きいプレーンを指定できます。画面上では，ボックスサイズの範囲だけが表示され，範囲外の部分はスクロールをして表示します。

プレプリント帳票

180dpi のドットインパクトプリンタ用の帳票です。カット紙に対応するものと，連続紙に対応するものとがあります。プレプリント用紙（あらかじめ，けい線や標題などが印刷されている用紙）に対して可変データを印字するときに使います。バーコードや OCR 文字も使えます。

ポップアップ

入力するデータを候補（メニュー項目）の選択によって指定するオブジェクトです。また，キーボードからの入力を併用する使い方もできます。

メニュー項目をドローで定義する「固定ポップアップ」と，AP から指定する「可変ポップアップ」があります。

ポップアップメニューファイル

XMAP3 では，ポップアップテキストに表示するメニューデータを AP 実行時にファイルとして渡せます。このファイルをポップアップメニューファイルといいます。ポップアップメニューファイルは，あらかじめ作成して用意しておく必要があります。

ポップアップメニューファイルは，メニュー項目が大量にある場合や，メニューを大分類，小分類にして表示したい場合に使用します。ポップアップメニューファイルは，ポップアップメニューエディタで編集できます。ポップアップメニューエディタは，XMAP3 Developer で提供します。

（マ行）

マージン

ページプリンタで，用紙に穴を開けることを想定して設定しておく余白のことです。用紙の左上を基点にしてマージンを設定し，レイアウト開始位置をずらせます。このようにユーザが指定するマージンをソフトマージンといいます。これに対し，プリンタドライバおよびプリンタ装置ごとに，ハードマージンと呼ばれるハード機構上，印刷できない領域があります。このハードマージンの値よりも小さいマージンを設定すると，ハードマージンが有効になるので，注意してください。

なお，ハードマージンは，GDI モード使用時にはプリンタドライバが持つ値となり，PDL スルーモード使用時にはプリンタ装置が持つ値となります。また，PDF ファイル出力の場合は，ハードマージンがないため，常にソフトマージンが有効なレイアウトとなります。

ます（GUI／CUI 画面の単位）

レイアウト定義に使う単位で，オブジェクトのサイズなどを表します。

ますのサイズは，画面属性で指定する基準文字サイズに従います。縦幅は文字サイズの半分，横幅は半角文字 1 字分となります。

レイアウト領域のグリッドは，1 ます刻みで表示されます。

ます（帳票の単位）

レイアウト定義に使う単位で，オブジェクトのサイズなどを表します。

ますのサイズは，帳票属性で指定する基準ます目に対応しています。

レイアウト領域のグリッドは，1 ます刻みで表示されます。ただし，けい線帳票以外の各帳票では，ドット単位や mm 単位のグリッドを表示することもできます。

マッピングインタフェース領域

COBOL の CALL 文および C 言語を使用したときにマッピングオプションを指定する領域です。

次に示すマッピングオプションに対応しています。ただし，C 言語の名称は - （ハイフン）が，_ （アンダースコア）になります。

マージ：XMAP-MDO-MAPFLD

論理マップだけ：XMAP-MDO-LOGFLD

物理マップだけ：XMAP-MDO-PHFLD

マッピングオプション（マッピングモード）

「マージ」，「論理マップだけ」，「物理マップだけ」の 3 種類があります。マッピングオプションの指定先には，COBOL の SEND，RECEIVE，TRANSCEIVE 文を使用したときに指定する MAPPING MODE 句と，COBOL の CALL 文および C 言語のときに指定するマッピングインタフェース領域があります。

マージは，論理マップと物理マップをマージして，同一画面をすべて書き換えるときに指定します。ただし，同じ画面に対し，2 回目以降の表示では固定部分（タイトルやけい線）は再描画しません。

論理マップだけは，一般的に 2 回目以降の表示で，入力した状態をそのままにして部分書き換えをするときに指定します。

物理マップだけは，メニュー表示など，物理マップだけで初期表示するときに指定します。

マッピング構成ファイル（X3MWCONF／XMAPconfig）

マッピング構成の環境設定ファイルです。

Windows 版 XMAP3 の場合，マッピング構成ファイルは編集できません。XMAP3 実行環境が提供しているファイルをそのまま使用します。UNIX 版 XMAP3 の場合は，テキストエディタで編集します。

マッピング構成ファイルは，XMAP3 Server Runtime，XMAP3 Client Runtime，または XMAP3/Web for Cosminexus で提供します。

格納場所とファイル名：

[Windows 版]

- XMAP3 Server Runtime または XMAP3 Client Runtime の場合
  XMAP3 インストールフォルダ¥ETC¥X3MWCONF
- XMAP3/Web for Cosminexus（XMAP3 Cosminexus 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for Cosminexus¥ETC¥X3MWCONF
- XMAP3/Web for Cosminexus（XMAP3 TP1/Web 連携機能）の場合
  XMAP3 インストールフォルダ¥Web for TP1¥ETC¥X3MWCONF

[UNIX 版]

/etc/opt/HIXMAP/XMAPconfig

マッピング属性ファイル（xps）

1 バイトコードで使用する日本語漢字コードを設定する環境設定ファイルです。マッピング属性ファイルは，UNIX 版 XMAP3 Server Runtime で使用します。

格納場所とファイル名：/etc/opt/HIXMAP/XMAPsrv/xps

マッピング方式

画面・帳票の固定データや AP の入出力データの配置を物理マップとして定義しておき，AP から入出力データを論理マップとして指定することで画面・帳票の入出力ができる方式です。画面・帳票のレイアウトとプログラムの処理を独立して設計できるので，業務開発の効率が向上します。

マップ

XMAP3 では，画面や帳票の様式に関する情報を AP から切り離して，マップと呼ばれる入れ物に保管します。マップにはマップ定義ファイル，物理マップ，および論理マップがあります。

マップ生成

マップ定義ファイルから，論理マップファイルと物理マップファイルを生成します。

また，書式定義ファイルから書式イメージファイルと行制御データファイルを生成します。

マップ生成は，XMAP3 Developer で提供します。

マップ帳票

けい線帳票，プレプリント帳票，網掛け帳票，およびグラフィック帳票の総称です。

マップ定義ファイル

ドローで定義した画面や帳票の定義情報であるソースマップを格納したファイルで，拡張子は.imp です。画面・帳票の定義終了時に，論理マップとともに生成されます。物理マップと論理マップは，このマップ定義ファイルを基に生成されます。

マップ展開形式

マップ生成時に出力される論理マップの数値領域の展開形を指定します。展開形式には，次に示す 2 種類があります。

リトルエンディアン用マップを展開

　Windows の場合で，スタンドアロンまたは C/S システム構成のときに使用できる展開方式で論理マップを生成します。

ビッグエンディアン用マップを展開

　AIX 版 XMAP3 を使った C/S 構成のときに使用できる展開方式で論理マップを生成します。

数値$(10)_{10}$ ＝$(0A)_{16}$ の場合，次のようになります。

リトルエンディアン：$(0A00)_{16}$

ビッグエンディアン：$(000A)_{16}$

メッセージアイコン

操作者に対して何らかの注意を促すときに表示します。メッセージアイコンは，メッセージと一緒に出力することをお勧めします。アイコンには，インフォメーション，ワーニング，クエスチョン，エラーの 4 種類があります。

メニュー形式

可変ポップアップテキストでポップアップメニューファイルを使用する場合，表示するポップアップメニューの形式を選択できます。メニュー形式には，標準のメニュー形式とリスト形式があり，リスト形式にはさらに「ポップアップ型」と「ダイアログ型」があります。

メニューバー

イベント（実行する処理）を候補（メニュー項目）の選択によって指定するオブジェクトです。イベントに対応する処理として「AP 通知」や「ポップアップ表示」などを指定できます。確定キーの代わりとしても使えます。メニューの内容はドローで定義し，AP からの指定はできません。

プルダウンメニューやカスケードメニューを使って階層化でき，処理内容に応じたグループ分けができます。

文字色

画面の場合，入出力テキスト・フィールドの文字やボタンのラベルなどの色になります。画面の文字色は，12 色から選べます。また，帳票の場合は，グラフィック帳票のフィールドや OCR の文字の色になります。帳票の文字色は 8 色から選べます。

色の設定は，ドローで選択した各オブジェクトの属性ダイアログのほかに，ドローのツールバーの［文字色/背景色］ボタンをクリックして表示される文字色/背景色のツールボックスで変更できます。このツールボックスを使うと，画面または帳票全体の配色を見ながら設定できます。

モジュラスチェック

「使用目的」が「数字」の場合に，モジュラスチェックをするかどうかを指定できます。指定した場合，画面の属性ダイアログで指定するモジュラスアルゴリズム（チェック 10 または 11）に従って，入力文字列（数字）がチェックされます。チェック条件に合った場合は AP へイベントは返しません。合わない場合はエラーになり，テキスト・フィールドからカーソル・フォーカスが移動できなくなります。

チェック 10

次のチェックをし，一致する（合格）かを判定します。

1. 入力したデータの最後の 1 バイト（チェックデジット）・入力したデータのバイト数（入力したデータの最後の 1 バイトを除く）を取り出す。
2. 入力したデータのバイト数の右側より 2，1，2，1，...を乗数とし桁ごとに掛け算をする。
3. 掛け算の結果が 2 桁となったものについては 10 で割り算し，商と余りに分解する。
4. 「2」「3」での掛け算と商および余りの結果をすべて加え合わせる。
5. 加算の結果を 10 で割り算し，余りを求める。
6. 10 から余りを減算し，差を求め，この差を入力したデータの最後の 1 バイトと比較する。

チェック 11

次のチェックをし，一致する（合格）かを判定します。

1. 入力したデータのバイト数（入力したデータの最後の 1 バイトを除く）を取り出す。
2. 入力したデータのバイト数の右側から 2，3，4，5，6，7，2，3，4，5，6，7，...の順に乗数を割り当て桁ごとに掛け算をする。
3. 結果をすべて加え合わせる。
4. 和を 11 で割り算し，余りを求める。
5. 11 から余りを減算し，差を求める。この差を入力したデータの最後の 1 バイトと比較する。

（ヤ行）

予約テキスト・フィールド

OLTP サーバ構成で OpenTP1 が提供する予約項目を出力するオブジェクトです。

予約項目に日付や時刻などを指定すると，AP 実行時に日付や時刻などの情報が OpenTP1 から動的に出力されます。

（ラ行）

ラジオボタン

入力するデータを候補（ボタン）の選択によって指定するオブジェクトです。グループ内の複数のボタンから一つだけ選択できます。ラベル長や選択肢の数が少ないとき，また，「YES／NO」などの単純な選択をするときに使います。

ボタンラベルをドローで定義する「固定ラジオボタン」と，AP から指定する「可変ラジオボタン」があります。

ラジオボタンのグループを表すボックスを「ラジオボタンボックス」といいます。

ラベル

画面に文字列を表示するための各種オブジェクトの総称です。定義時に指定した固定の文字列を表示する「固定テキスト・フィールド」と，AP から指定する可変の文字列を表示する「出力テキスト・フィールド」,「出力日付／時刻テキスト・フィールド」が該当します。

リストボックス

入力するデータを候補（リスト項目）の選択によって指定するオブジェクトです。リストの内容は AP から指定します。ラベル長や選択肢が多いときに使います。必要に応じて，スクロールバーを付けられます。

リストから一つだけ項目を選べる「単一選択リストボックス」と，複数の項目を選べる「複数選択リストボックス」があります。

リトルエンディアン

バイナリ形式のデータを扱う際，メモリ上に右から左側へ 1 バイトずつ格納する形式です。

（例）

$(000A)_{16}$ を 2 バイトのメモリに格納した場合は，$(0A00)_{16}$ になります。

レイアウトパターン

ドローで画面／帳票を新規作成するときに，用途に応じて指定するパターンです。標準的なレイアウト領域のサイズやパターンなどがあらかじめ設定されていて，ドローでのレイアウト定義の基本フォーマットとなります。

XMAP3 が標準提供しているレイアウトパターンのほかに，ユーザが独自に作成・登録することもできます。

レイアウト領域

ドローのレイアウト領域です。XMAP3 のレイアウト定義の範囲は，この領域になります。画面では，タイトルバー，メニューバー，オペレータインジケータを除く領域のことです。帳票では，オフセットとユーザが設定したマージン値（ソフトマージン）を除いた領域になります。

連結出力バーコード

バーコードを印字するオブジェクトです。連結出力バーコードは，従来の JAN コードなどのような単一データ（商品コードなど）をバーコードとして印字するオブジェクトに対して，複数データ（メーカーコード，支払い期限，支払い金額など）を一つのバーコードとして印字するオブジェクトです。データは AP から指定し，論理マップを介して帳票に印字します。このとき，一つのバーコードに対して複数の論理項目（分類項目）を使用します。

バーコードの種類には，GS1-128 があります。GS1-128 バーコードを印刷する場合は，300dpi（推奨は 600dpi）以上のプリンタを使用してください。

連続紙

連続紙は，各ページの境がミシン目になってつながっている用紙です。シリアルインパクト帳票でだけ利用できます。

ロギング支援

XMAP3 の AP 実行時に，ログ情報を採取してログファイルに出力したり，ログ情報を表示したりする機能です。ログ情報は，実行時に発生した問題の解決に利用します。

ロギング支援は，XMAP3 Server Runtime および XMAP3 Client Runtime で提供します。

論理ハードコピー

スタンドアロン構成，C/S 構成，および OLTP サーバ構成で，画面に表示されている CUI 画面および GUI 画面のフィールドボックスを対象に，画面そのものの印刷ではなく，画面に配置した文字，けい線などの画面オブジェクトを印刷する機能です。

論理ハードコピー機能は，XMAP3 実行環境で提供します。

論理マップ

論理マップは，プログラムの可変データを格納する領域です。画面データの入出力や帳票データの出力時に，AP が XMAP3 に対するパラメタとして利用します。COBOL で AP を開発する場合は，論理マップは登録集原文として生成され，COPY

文で AP に取り込みます。AP の実行時には，AP に取り込まれた論理マップのデータ領域を介して，データがやり取りされます。ファイルの拡張子は，COBOL の場合は.cbl，C 言語の場合は.h になります。

書式オーバレイ印刷の場合は，論理マップを使用しません。画面，帳票の物理マップに相当する書式イメージファイルと行制御データファイルを使用します。

（ワ行）

割込キー（Break），（Ctrl）＋（Pause）

送信要求の発生と割込のイベントを AP に通知するキーです。

［Break］キー，または［Ctrl］＋［Pause］キーです。

XMAP3 実行環境の環境設定で，表示・印刷環境ファイル（X3PCONF／XPWconfig）の，強制確定キーの動作（表示サービス名.DCSRED=）オプションの指定で，強制確定キーの設定ができます。

索引

数字

A

B

C

D

E

F

G

I

J

L

M

O

P

R

S

T

W

X

あ

い

う

え

お

か

き

く

け

こ

さ

し

す

せ

そ

た

ち

つ

て

と

に

ぬ

は

ひ

ふ

ほ

ま

め

も

ゆ

よ

ら

り

れ

ろ

わ