





バージョン 7.1



## IBM Informix 4GL to EGL Conversion Utility ユーザーズ・ガイド

**ご注意:**

本書および本書で紹介する製品をご使用になる前に、L-1 ページの『特記事項』に記載されている情報をお読みください。

本書には、IBM の専有情報が含まれています。その情報は、使用許諾条件に基づき提供され、著作権により保護されています。本書に記載される情報には、いかなる製品の保証も含まれていません。また、本書で提供されるいかなる記述も、製品保証として解釈すべきではありません。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： G229-6339-01  
Rational® IBM Informix  
Version 7.1  
IBM Informix 4GL to EGL Conversion Utility User's Guide

発行： 日本アイ・ピー・エム株式会社

担当： ナショナル・ランゲージ・サポート

第1刷 2008.5

© Copyright International Business Machines Corporation 2005, 2008. All rights reserved.

---

# 目次

概要	vii
この概要について	vii
この資料について	vii
対象ユーザーのタイプ	vii
この製品のフィーチャー	vii
サポートされるプラットフォーム	viii
書体の規則	viii
資料	ix
IBM へのご意見	xi
第 1 章 変換プロセスの概要	1-1
本章の内容	1-1
I4GL の EGL 変換の概要	1-1
変換の利点	1-1
I4GL の EGL 変換の概要	1-2
変換前のステージ	1-2
変換ステージ	1-2
変換後のステージ	1-3
再変換ステージ	1-3
変換の制限	1-3
C インターフェースのサポートおよび制限	1-4
レポートのサポートおよび制限	1-4
画面書式のサポート	1-5
第 2 章 変換の準備	2-1
本章の内容	2-1
変換前のタスクの概要	2-1
変換の制限と次善策	2-2
4GL ソースの変換	2-2
C コード機能	2-2
レポート	2-3
既存の I4GL コンポーネント・プロジェクトの識別	2-5
I4GL ソース・ファイルの生成	2-5
I4GL アプリケーションのコンパイル	2-5
クライアント・ロケールの識別	2-5
共用ライブラリーの識別と分離	2-5
Rapid Development System (RDS) で使用する C コードの変更	2-6
ユーザー定義メッセージ・ファイルの識別	2-7
Informix データベース・スキーマ情報の識別	2-7
EGL 宛先ディレクトリーの識別	2-7
I4GL ソース・ファイル・ディレクトリーの準備	2-8
第 3 章 変換タスク	3-1
本章の内容	3-1
変換ユーティリティー・ステージ	3-1
Informix データベース・スキーマ抽出	3-1
Informix データベース・スキーマ抽出の変換ユーティリティー処理	3-3
I4GL 共用ライブラリー変換	3-4
I4GL 共用ライブラリーの変換ユーティリティー処理	3-7
I4GL アプリケーション変換	3-7

I4GL アプリケーション変換の変換ユーティリティー処理	3-10
変換ユーティリティーのコマンド行モード	3-10
変換ログ	3-11

## 第 4 章 変換後のタスク 4-1

本章の内容	4-1
変換後のタスク	4-2
変換中に行われる変更点	4-2
変換中に生成される成果物	4-3
構成ファイル	4-5
マニフェスト・ファイル	4-5
ソース・ファイル変換マッピング	4-6
コマンド行変換: プロジェクトのワークスペースへのインポート	4-7
変換エラーの訂正	4-7
変換ログの内容	4-9
EGL プログラムでの C 共用ライブラリーの使用	4-10
EGL ネイティブ・ライブラリー	4-11
関数テーブル	4-11
アプリケーション・レベルの共用ライブラリーの作成	4-12
プロパティ・ファイル	4-13
変換済み EGL ファイルの検証とコンパイル	4-15
EGL の Java への生成	4-15
エラー・メッセージ変換の理解	4-16
レポート変換の理解	4-17
EGL レポート・ドライバ関数	4-18
I4GL レポート・セクション	4-19
EGL プロジェクト、パッケージ、およびファイルの理解	4-32
EGL プロジェクト	4-32
パッケージ	4-33
EGL ファイル	4-34
推奨事項	4-35
インフォメーション・センターのヘルプ・システムと EGL チュートリアル	4-36

## 第 5 章 再変換プロセスおよびタスク 5-1

本章の内容	5-1
I4GL 共用ライブラリーを再変換する場合	5-1
I4GL 共用ライブラリーの再変換の方法	5-1
変換ウィザードの再変換	5-2
コマンド行の再変換	5-2
失敗した再変換の原因と次善策	5-3

## 付録 A. I4GL から EGL への構文のマッピング A-1

## 付録 B. I4GL レポート変換コード例 B-1

## 付録 C. I4GL 書式コードから EGL 書式コードへの例 C-1

## 付録 D. 構成ファイルのテンプレート D-1

## 付録 E. マニフェスト・ファイルの例 E-1

## 付録 F. DTD 例 F-1

## 付録 G. 変換ログの例 G-1

## 付録 H. EGL ビルド記述子の例 H-1

付録 I. EGL 予約語 . . . . .	I-1
用語集 . . . . .	J-1
エラー・メッセージ . . . . .	K-1
特記事項 . . . . .	L-1
索引 . . . . .	X-1



---

## 概要

この概要について . . . . .	vii
この資料について . . . . .	vii
対象ユーザーのタイプ . . . . .	vii
この製品のフィーチャー . . . . .	vii
サポートされるプラットフォーム . . . . .	viii
書体の規則 . . . . .	viii
資料 . . . . .	ix
IBM へのご意見 . . . . .	xi

---

### この概要について

この概要では、この資料に含まれる情報の概要と、使用される規則について説明します。

---

### この資料について

この資料は、IBM Informix 4GL (I4GL) アプリケーションを Enterprise Generation Language (EGL) アプリケーションに変換する方法に関する情報を提供します。拡張 Rational Business Developer で EGL の開発テクノロジーを使用することによって、Java™ 環境、および z/OS 上で稼働する全機能アプリケーションを迅速に作成することができます。

---

### 対象ユーザーのタイプ

この資料は、Web アプリケーションを柔軟に作成したい、書式を含む手持ちのアプリケーションの使用可能度を拡張したい、新規の EGL レポート機能を使用したい、EGL プログラムでメッセージ・キューを使用したい、また、あらゆる EGL 機能をフルに使用したいなどの希望を持つ I4GL アプリケーション開発者を対象に記述されています。

この資料のユーザーは、I4GL コードをコンパイルした経験などの I4GL プログラミングの経験があることを前提とします。さらに、I4GL プログラムで C コードを使用している場合は、C 言語プログラミングの知識が必要です。

---

### この製品のフィーチャー

このセクションの情報には、I4GL から EGL への変換ユーティリティー製品と、EGL に含まれる I4GL に類似した拡張機能の両方が含まれています。

I4GL から EGL への変換ユーティリティーは、ご使用の共用ライブラリーおよび I4GL アプリケーションを EGL パッケージおよびプログラムに変換します。ご使用の共用ライブラリーと I4GL アプリケーションを変換した後に、ご使用の C 関数を変換済みの EGL コードにリンクすることができます。

EGL には、次の I4GL 類似機能があります。

**画面書式。**変換ユーティリティーは、既存の I4GL 画面書式を EGL と同等のコンソール・ユーザー・インターフェースに変換します。

**C 関数の呼び出し。**C 関数は、I4GL プログラムから呼び出すことができます。同様に、EGL も C 関数を呼び出すことができます。

**レポート。**変換ユーティリティーは、ご使用の I4GL レポート・ファイルを同等の EGL ファイルおよび JasperReports ファイルに変換します。

**注:** この製品には、Teodor Danciu (<http://jasperreports.sourceforge.net>) により開発されたソフトウェアが含まれています。

---

## サポートされるプラットフォーム

I4GL プログラムを EGL パッケージに正常に変換するには、以下の開発環境プラットフォームのいずれかにアクセスする必要があります。

- AIX
- HP-UX
- Linux
- Solaris
- Windows

さらに、共用ライブラリーおよび新規の **.egl** ソース・ファイルを作成するためのソース・コードおよびディレクトリー構造への適切なシステム・アクセスとアクセス権が必要です。

---

## 書体の規則

このセクションでは、この資料で使用する書体の規則について説明します。これらの規則によって、この資料および資料セット内のその他資料からの情報の収集が容易になります。

規則	意味
キーワード	プログラミング言語のステートメントの基本要素 (キーワード) は、すべてセリフ・フォントの大文字で表示されます。
イタリック体	テキストでは、新規の語句および強調語はイタリック体で表示されます。構文およびコードの例では、指定する変数値がイタリック体で表示されます。
太字体	プログラム・エンティティーの名前 (クラス、イベント、およびテーブルなど)、環境変数、ファイルとパス名、およびインターフェース・エレメント (アイコン、メニュー項目、およびボタンなど) は、太字体で表示されます。
モノスペース	製品が表示する情報およびユーザーが入力する情報はモノスペース書体で表示されます。
キー・ストローク	ユーザーが押すキーは、sans serif フォントの大文字で表示されます。

規則	意味
>	このシンボルは、メニュー項目を示します。例えば、“「ツール」>「オプション」を選択”は、「ツール」メニューから「オプション」項目を選択するという意味です。

## 資料

このセクションでは、I4GL 変換に役立ち、また EGL の使用法についてさらに学ぶために使用することができる資料について説明します。

この「*IBM Informix 4GL to EGL Conversion Utility ユーザーズ・ガイド*」(ユーザーズ・ガイド) では、I4GL ユーザーを対象に、EGL への変換パスを使用する方法について説明します。この資料は以下の章で構成されます。

- この概要
- 『第 1 章 変換プロセスの概要』では、I4GL の EGL 変換プロセスの利点と制限について説明し、変換前、変換、変換後、および再変換のステージについて概要を提供します。
- 『第 2 章 変換の準備』では、変換ウィザードを使用する前に、ユーザーが実行する必要のあるタスクについて説明します。これらのタスクには、データベース・スキーマおよび I4GL アプリケーションのコンポーネントの識別、C 共用ライブラリーの作成、およびレポートのフォント指定の理解などがあります。
- 『第 3 章 変換タスク』では、変換ウィザードを使用して Informix データベース・スキーマを抽出し、I4GL 共用ライブラリーおよびアプリケーションを変換する方法の詳細を記載しています。
- 『第 4 章 変換後のタスク』では、変換ユーティリティーによって I4GL アプリケーションに対して行なわれた変更点、変換中に生成された成果物、変換エラーの訂正方法、および EGL アプリケーションで C 共用ライブラリーを使用する方法について説明します。この章では、EGL Information Center のオンライン・ヘルプ・システムの使用法についても説明します。
- 『第 5 章 再変換プロセスおよびタスク』では、I4GL 共用ライブラリーを再変換する場合とその方法について説明します。
- A-1 ページの『付録 A. I4GL から EGL への構文のマッピング』では、I4GL と EGL 構文の間のマッピングを提供します。
- B-1 ページの『付録 B. I4GL レポート変換コード例』では、I4GL レポート・コードおよび、それと同等の EGL ドライバー関数の例を提供します。
- C-1 ページの『付録 C. I4GL 書式コードから EGL 書式コードへの例』では、I4GL 書式コードおよび、それと同等の EGL コードの例を提供します。
- D-1 ページの『付録 D. 構成ファイルのテンプレート』では、データベース・スキーマ抽出、ライブラリー、およびアプリケーション・プロジェクトのテンプレート構成ファイルを提供します。
- E-1 ページの『付録 E. マニフェスト・ファイルの例』では、データベース・スキーマ抽出、ライブラリー、およびアプリケーション・プロジェクトのマニフェスト・ファイルの例を提供します。

- F-1 ページの『付録 F. DTD 例』では、データベース・スキーマ抽出、ライブラリー、およびアプリケーション・プロジェクトの構成ファイルおよびマニフェスト・ファイルに使用される DTD の例を提供します。
- G-1 ページの『付録 G. 変換ログの例』では、**.txt** フォーマットの変換ログの例を提供します。
- H-1 ページの『付録 H. EGL ビルド記述子の例』では、変換ユーティリティーによって生成された EGL ビルド記述子ファイルの例を提供します。
- I-1 ページの『付録 I. EGL 予約語』には、『EGL 予約語』がリストされています。
- これらの章に続いて、関連する語句の用語集とエラー・メッセージ・セクションがあります。
- 特記事項の付録では、IBM 製品、フィーチャー、およびサービスについて説明しています。
- 索引からは、特に関心のある項目の記載場所を調べることができます。

I4GL アプリケーションを変換する前に本書をすべて読み、I4GL アプリケーションの変換中にも本書を参照してください。「IBM Redbook *Transitioning: Informix 4GL to Enterprise Generation Language (EGL)*」でも、4GL アプリケーションの変換方法に関する情報と考察が記載されています。Redbook は、[www.redbooks.ibm.com/redbooks.nsf/redbooks/](http://www.redbooks.ibm.com/redbooks.nsf/redbooks/) にあり、オンラインでアクセスできます。

Rational 製品には、**readme.html** ファイルが含まれています。**readme.html** ファイルには、ご使用の Rational 製品の資料に対するすべての制限と変更点に関する最新情報が記載されています。このファイルは、Rational 製品のトップ・ディレクトリーにあります。

さらに、変換ユーティリティーには、**readme004FGL.html** ファイルが含まれています。**readme004FGL.html** ファイルには、変換ユーティリティーの制限およびこのユーザー・ガイドが完成してから行われた資料への変更点についての情報が含まれています。I4GL アプリケーションを変換する前に **readme004FGL.html** ファイルの内容を検討してください。ファイルは、変換ユーティリティー・プラグインのトップ・ディレクトリーにあります。

変換ウィザードを使用しているときは、**F1** を押して使用中の特定のパネルのヘルプをアクティブにすることができます。変換ウィザードの各パネルに、それに関連する専用のヘルプ・トピックがあります。それぞれのヘルプ・トピックで、そのウィザード・パネルで必要な情報について説明しています。ヘルプ・トピックをアクティブにするには、ウィザード・パネル上にカーソルを移動して **F1** キーを押します。

変換済みの EGL コードの使用方法についての情報は、次の場所にあります。

- Rational オンライン・インフォメーション・センター。インフォメーション・センターには、メインメニューから、「ヘルプ」 > 「**Rational ヘルプ (Rational Help)**」と選択してアクセスできます。
- EGL ホーム・ページおよび IBM developerWorks のフォーラム。EGL ホーム・ページには、EGL に関するさまざまなトピックに関する資料が提供されています。EGL ホーム・ページは、[www.ibm.com/developerworks/rational/products/egl/](http://www.ibm.com/developerworks/rational/products/egl/)

にあります。 [www.ibm.com/developerworks/forums/dw\\_rforums.jsp](http://www.ibm.com/developerworks/forums/dw_rforums.jsp) にある developerWorks コミュニティーから、EGL フォーラムにアクセスすることができます。

- EGL チュートリアルでは、EGL を使用して簡単な動的 Web サイトをビルドする方法について学びます。チュートリアルには、メインメニューから「ヘルプ」>「チュートリアル・ギャラリー」を選択してアクセスすることができます。チュートリアルについて詳しくは、4-36 ページの『インフォメーション・センターのヘルプ・システムと EGL チュートリアル』を参照してください。

---

## IBM へのご意見

この資料内で、役に立つと思われる修正や説明がございましたらお知らせください。今後のバージョンの参考にさせていただきます。その際、次の情報をご記入ください。

- ご使用の資料の名前およびバージョン
- セクションおよびページ番号
- 資料に関するお客様のご提案

次の E メール・アドレスにお客様のご意見をお寄せ下さい。

[docinf@us.ibm.com](mailto:docinf@us.ibm.com)

この E メール・アドレスは、資料内の間違いや欠落部分を報告するために使用します。技術的な問題について直ちにヘルプが必要な場合は、IBM テクニカル・サポートにご連絡ください。



---

## 第 1 章 変換プロセスの概要

本章の内容	1-1
I4GL の EGL 変換の概要	1-1
変換の利点	1-1
I4GL の EGL 変換の概要	1-2
変換前のステージ	1-2
変換ステージ	1-2
変換後のステージ	1-3
再変換ステージ	1-3
変換の制限	1-3
C インターフェースのサポートおよび制限	1-4
レポートのサポートおよび制限	1-4
画面書式のサポート	1-5

---

### 本章の内容

この章では、I4GL の EGL 変換プロセスの利点と制限について説明し、変換前、変換、変換後、および再変換のステージについて概要を説明します。

---

### I4GL の EGL 変換の概要

IBM Informix 4GL の EGL 変換ユーティリティ (変換ユーティリティ) によって、Informix 4GL (I4GL) 言語で書かれたアプリケーションを Enterprise Generation Language (EGL) に変換できます。EGL によって、Java™ 環境で稼働する全機能アプリケーションを迅速に作成することができます。EGL によって、Web テクノロジーに関する経験が少ないユーザーでも、企業データをブラウザに提供できます。EGL を使用して、他のインターネット・ベースのプログラムがアクセスできる Web サービスをコーディングすることができます。

変換ユーティリティは、特定の IBM Informix Dynamic Server (IDS) データベースに接続している I4GL プログラムを、同じ IDS データベースに接続する同等の EGL プログラムに変換します。変換ユーティリティは、I4GL 言語構文を変換して、成果物 (I4GL 共用ライブラリーなど) を同等の EGL コンポーネントにビルドします。実行時に、EGL ビルドが Java 実行可能プログラムを生成します。EGL アプリケーションは、IDS データベースを処理することができます。現在、EGL アプリケーションは、IBM Informix XPS、オンライン、または SE データベースではサポートされません。

変換ユーティリティには、1 回のセッションでプログラムを変換、必要なときに変換を開始および終了、または必要に応じて共用ライブラリーを再変換する機能があります。

---

### 変換の利点

I4GL ユーザーがアプリケーションを EGL に変換すると、以下の作業が実行できるようになります。

- I4GL 書式インターフェース (EGL ではコンソール・ユーザー・インターフェースと呼ばれます) などの新規の EGL コードを、I4GL とまったく同様に使用します。EGL コードは、C 関数を呼び出したり、制限付きで ESQL/C 関数を呼び出すこともできます。最後に、EGL ユーザーは I4GL と同様のレポートを生成することができます。
- 新規 EGL コードを使用して Web アプリケーションを作成します。

---

## I4GL の EGL 変換の概要

I4GL の EGL への変換は、一連の 3 ステージが完了することによって正常に行なわれます。以下にそれぞれについて説明します。

1. 変換前のステージ
2. 変換ユーティリティ・ステージ
3. 変換後のステージ

注: I4GL 共用ライブラリーを再変換する必要がある場合があります。再変換のステージについても以下で説明します。

### 変換前のステージ

変換を開始する前に、I4GL プログラムおよびデータベース・スキーマのコンポーネントを識別し、I4GL プログラム・ファイルおよびライブラリーの準備をする必要があります。これについては、2-1 ページの『第 2 章 変換の準備』に詳しく記載されています。

### 変換ステージ

I4GL アプリケーションの変換は、変換ユーティリティ・ウィザードで実行します。ウィザードの画面で、変換前のステージで編成した情報を挿入するようにプロンプトが出されます。ウィザードの使用方法については、3-1 ページの『第 3 章 変換タスク』に詳しく記載されています。

I4GL プログラムの変換は、1-5 ページの図 1-2 に示すような多層プロセスです。この図では、変換される I4GL ファイルのタイプ、ファイルが変換される順序、および変換後のファイルのタイプが要約されています。また、**.c** ファイルは変換されないことも示されています。

I4GL プログラムのコンポーネントは、以下の順序で抽出や変換を行う必要があります。

- データベース・スキーマ情報
- I4GL 共用ライブラリーまたはライブラリー
- I4GL アプリケーション

注: C 共用ライブラリーは、EGL で使用するために変換する必要がないので、ウィザードを使用しません。

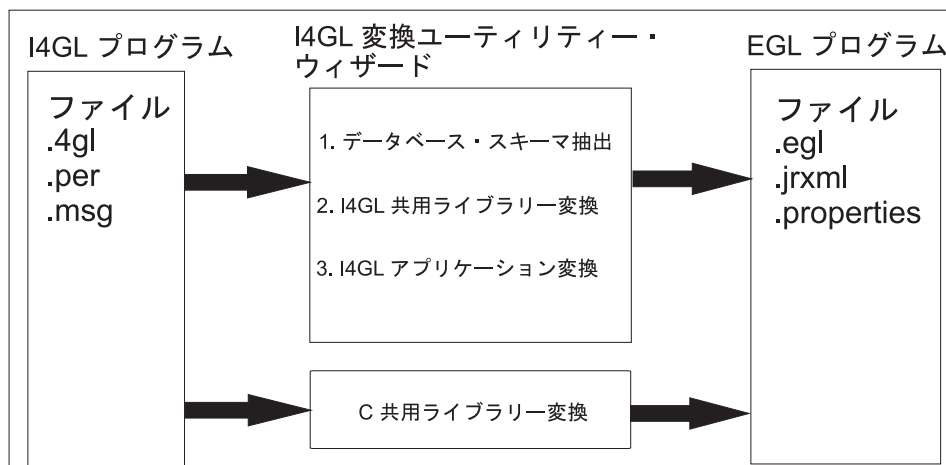


図 1-1. I4GL プログラムを EGL プログラムに変換する方法

## 変換後のステージ

変換ログで、変換が正常に完了したかどうかを確認します。変換が正常に実行され、変換ログにエラーがないことを確認したら、EGL ソース・ファイルをコンパイルして変換されたアプリケーションを使用することができます。変換ログにエラーが特定されている場合は、EGL ファイルを使用する前にそれらのエラーを修正する必要があります。これについては、4-1 ページの『第 4 章 変換後のタスク』に詳しく記載されています。

## 再変換ステージ

共用ライブラリーが誤った順序で変換されたり、他の共用ライブラリーの関数を参照すると、C 関数と想定された関数参照の一部が後で EGL に定義されます。想定が誤っていた場合、生成された共用ライブラリーをすべて再変換しなければなりません。また、複数の共用ライブラリーを含むアプリケーション内の共用ライブラリーは、再変換する必要があることがあります。共用ライブラリーの再変換について詳しくは、5-1 ページの『第 5 章 再変換プロセスおよびタスク』を参照してください。

## 変換の制限

I4GL から EGL への変換ユーティリティ、バージョン 7.1 には、以下の制限があります。

1. I4GL 関数を呼び出す C 関数はサポートされません。
2. I4GL モジュールで定義されたグローバル変数に、C ソース・コードからアクセスすることはできません。
3. ESQL/C または C 共用ライブラリーと、マイグレーション済み EGL アプリケーションまたはパッケージとの間のデータベース接続の共用はサポートされません。
4. 動的 4GL はサポートされません。

5. 変換済みレポートの再設計や新規レポートの設計を行うには、サード・パーティーのレポート設計製品を使用する必要があります。
6. EGL に変換された I4GL レポートは、Java にのみ生成されます。COBOL に生成することはできません。
7. EGL に変換された I4GL アプリケーションのパフォーマンスは、やや低下します。
8. Ace レポートおよび ISQL などの I4GL 補助製品の変換はサポートされません。
9. EGL は、同じアプリケーション内または共用ライブラリー内の、グローバルおよびローカルのカーソル有効範囲の両方をサポートしません。例えば、ローカルの有効範囲のあるライブラリーおよびグローバルの有効範囲のあるアプリケーションを変換することはできません。
10. I4GL プログラム・データベース **syspgm4gl** は、変換プロセスでサポートされません。
11. I4GL では、関数は異なる数および型の値を戻すことができます。一方、EGL では 1 つの関数は単一の値のみを戻します。EGL 関数は 1 つ以上の OUT パラメーターを持つことができ、これを使用して複数値の戻りをシミュレートすることが可能です。ただし、いずれの場合でも型は EGL に固定され、一度 I4GL が EGL に変換されると、ユーザーは、例えばある場所で "return 5;" を使用し、他の場所で "return "foo";" を使用することはできません。さらに、ユーザーは、1 つの場所で (1) の関数を戻し、他の場所で (1, "foo") を戻すことはできません。

注: 変換の制限に関するタスク指向の情報およびそれらの次善策については、2-2 ページの『変換の制限と次善策』に詳しく記載されています。

---

## C インターフェースのサポートおよび制限

多くの I4GL ユーザーは、C 関数を呼び出す I4GL プログラムを持っています。わずかな修正を行うことで、または修正することなく既存の C コードを I4GL 変換済みコードで使うことができるようになります。変換ユーティリティを使用する前に、C コードを準備するのに必要なタスクについて具体的な情報は、2-6 ページの『Rapid Development System (RDS) で使用する C コードの変更』に記載されています。C 共用ライブラリーと変換済み EGL アプリケーションとをリンクする方法については、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。C の機能を EGL で使用する方法については、オンライン・インフォメーション・センターを参照してください。

---

## レポートのサポートおよび制限

変換ユーティリティは、I4GL レポート・ファイルを、ドライバーおよび ReportHandler については EGL ファイルに変換し、設計の値については JasperReports ファイルに変換します。これらの新規ファイルを作成することにより、EGL ユーザーはレポート設計機能や選択したテキスト・エディターを使用して、レポートのレイアウトを設計することができます。

注: ご使用の Rational 製品には、サード・パーティーによるレポート設計ソフトウェアは含まれていません。サード・パーティーのレポート設計ソフトウェアを使用したい場合は、それを別に購入する必要があります。詳しくは、『EGL レポートの概要』のヘルプ・トピックを参照してください。

図 1-2 は、I4GL レポートを EGL に変換する方法を示します。まず、既存の I4GL レポート・プログラムのプロパティを定義し、それを変換ユーティリティ・ウィザードに入力します。次に、ユーティリティ・ウィザードが、JasperReport XML 設計文書、EGL レポート・ドライバー機能、および EGL レポート・ハンドラーの出力を作成します。

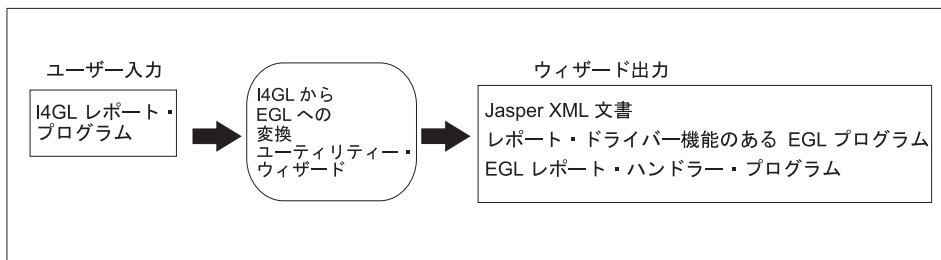


図 1-2. I4GL レポートを EGL に変換する方法

EGL への変換のために I4GL レポートを準備する方法については、2-2 ページの『変換の制限と次善策』を参照してください。レポートの変換について理解する方法については、4-17 ページの『レポート変換の理解』を参照してください。I4GL レポート、EGL および JasperReports の間の構文マッピングについては、A-1 ページの『付録 A. I4GL から EGL への構文のマッピング』を参照してください。EGL レポート・フィーチャーを EGL で使用する方法については、『EGL レポートの概要』のヘルプ・トピックを参照してください。

## 画面書式のサポート

I4GL から EGL への変換により、ユーザーは Web ベース・アプリケーションをグラフィカル・ユーザー・インターフェースで実装できます。また、変換ユーティリティによって、ユーザーは I4GL 画面書式やテキスト・ベースのユーザー・インターフェースの外観や機能を保持することもできます。ご使用の I4GL アプリケーションを変換している間に、I4GL 画面書式は、EGL コンソール・ユーザー・インターフェース (CUI) に変換されます。CUI には、画面書式とまったく同じ機能が備わります。変換済み CUI の実行に加えて、新規の CUI を作成することもできます。

I4GL 書式と CUI の間の構文マッピングを確認するには、A-9 ページの『I4GL 書式から EGL コンソール・ユーザー・インターフェースへ』を参照してください。EGL アプリケーションでコンソール・ユーザー・インターフェースを使用する方法については、『コンソール・ユーザー・インターフェース』のヘルプ・トピックを参照してください。



---

## 第 2 章 変換の準備

本章の内容	2-1
変換前のタスクの概要	2-1
変換の制限と次善策	2-2
4GL ソースの変換	2-2
C コード機能	2-2
レポート	2-3
既存の I4GL コンポーネント・プロジェクトの識別	2-5
I4GL ソース・ファイルの生成	2-5
I4GL アプリケーションのコンパイル	2-5
クライアント・ロケールの識別	2-5
共用ライブラリーの識別と分離	2-5
Rapid Development System (RDS) で使用する C コードの変更	2-6
ユーザー定義メッセージ・ファイルの識別	2-7
Informix データベース・スキーマ情報の識別	2-7
EGL 宛先ディレクトリーの識別	2-7
I4GL ソース・ファイル・ディレクトリーの準備	2-8

---

### 本章の内容

この章では、変換ウィザードを使用する前に実行する必要があるタスクについて説明します。これらのタスクには、データベース・スキーマおよび I4GL アプリケーションのコンポーネントの識別、C 共用ライブラリーの作成、およびレポートのフォント指定の理解などがあります。

---

### 変換前のタスクの概要

I4GL から EGL への変換を正常に実行するために、変換ユーティリティー・ウィザードは、I4GL アプリケーションに固有の情報を必要とします。以下にリストした変換前のステップは、変換するファイルを準備してウィザードが必要とする情報を収集するのに役立ちます。

この章では、以下のステップについて詳しく説明します。

1. 変換の制限について理解します。
2. 変換したい I4GL プロジェクトを識別し、そのプロジェクト内にある各 I4GL アプリケーションの名前とソース・ファイルのロケーションを記録します。  
I4GL アプリケーションにリンクされているすべての従属 I4GL 共用ライブラリーと C 共用ライブラリーの名前とロケーションを識別して記録します。必要であれば、すでに変換済みの EGL パッケージの共用ライブラリーもリストしてください。
3. 必要に応じて、**.4gl** および **.per** ファイルを生成します。
4. すべての I4GL ソース・ファイルが、I4GL 7.32 コンパイラーを使用して正常にコンパイルされることを確認します。
5. I4GL プログラムの中に英語以外のロケールでコンパイルされるものがある場合は、クライアント・ロケールを識別します。

6. 共用ライブラリーの I4GL ビルド環境に **.4gl** および **.c** ソース・ファイルの両方が含まれているかどうかを識別します。それら両方が含まれている場合は、**.4gl** ファイルを分離して、それらを I4GL 共用ライブラリー・プロジェクトとして変換してください。**.c** ファイルは共用ライブラリーにコンパイルします。
7. ユーザー定義のメッセージ・ファイルおよびメッセージをエンコードしたコード・ページを識別します。
8. IBM Informix データベース・インスタンスを開始して、I4GL モジュールが使用するデータベース・スキーマを作成し、データベース接続情報 (データベース名、サーバー名、ホスト名、ポート番号、クライアント・ロケール、およびデータベース・ロケール) を記録します。変換の間に、ユーザー名とパスワードも知っておく必要があります。
9. EGL ファイルの宛先ディレクトリーを識別します。
10. I4GL ソース・ファイルを、変換ユーティリティーを実行するマシン上の I4GL ステージング・ディレクトリーに移動します。

---

## 変換の制限と次善策

以下に、変換の制限および次善策の適用が可能であればそれらをリストします。

### 4GL ソースの変換

4GL では、DATABASE ステートメントがデフォルトのデータベース接続を作成し、GLOBALS または MAIN ファイルのいずれかに置くことができます。しかし、EGL では、プログラムの開始時に **connect()** ステートメントを使用しなければなりません。GLOBALS ファイルが、アプリケーションまたは共用ライブラリー変換プロジェクトと同じプロジェクト内に存在する場合は、変換ユーティリティーがGLOBALS ファイルから DATABASE を抽出して **connect()** ステートメントを生成することができます。一方、GLOBALS ファイルが分離した変換プロジェクトとして変換されていると、DATABASE ステートメント情報が失われ、アプリケーション・プロジェクトの変換中に変換ユーティリティーが **connect()** ステートメントを生成するのに十分な情報を得ることができません。この後者のシナリオが発生した場合、ランタイム・エラーが生成されます。

これを回避するには、MAIN プログラム・ファイルを含むアプリケーション・プロジェクト・ファイルを編集して、GLOBALS 内に存在した DATABASE ステートメントを挿入する必要があります。この次善策によって、変換ユーティリティーが 4GL の DATABASE *databasename* ステートメント情報を使用して EGL の **connect()** ステートメントを生成することができます。

### C コード機能

I4GL 言語では、以下に示すように C コードの 2 つの実装があります。

- C 関数は I4GL プログラムによって呼び出され、push および pop 外部関数を使用して C 関数が I4GL 関数を呼び出すことができます。
- C プログラムでは、**fgl\_start()**、**fgl\_call()**、**fgl\_exitfm()**、および **fgl\_end()** マクロを使用して、I4GL 関数を呼び出すことができます。

EGL 言語では、EGL 関数が C 関数を呼び出すことができますが、C 関数は EGL 関数を呼び出すことができません。I4GL アプリケーションを EGL に変換する前

に、I4GL 関数を呼び出している C 関数のすべてのインスタンスをコードから除去しなければなりません。さらに、I4GL 関数に値を渡すために使用する push 関数と I4GL 関数によって戻された値を取り出すために使用する pop 関数を除去する必要があります。

EGL に定義されたグローバル変数は C コードで使うことができません。変換の前に、I4GL および C コードを変更してこのようなグローバル変数を C コードに明示的に渡してから、C 呼び出しから戻った時点で値をリセットするようにする必要があります。

EGL は、実行時に Java に変換されるため、C 関数の呼び出し時に追加の JNI (Java Native Interface) の層が必要になります。従って、C 関数を呼び出している EGL アプリケーションのパフォーマンスは、C 関数を呼び出している I4GL アプリケーションのパフォーマンスよりも遅くなる可能性があります。

## レポート

変換ユーティリティは、I4GL レポート論理を次の 2 つのタイプのファイルに変換します。

- ReportHandler および ReportDriver 関数の複数 EGL (.egl) ファイル
- レポート設計の JasperReports (.jrxml) ファイル

ユーティリティは、.jrxml ファイルを作成するために、フォント形式の指定を必要とします。I4GL アプリケーションの変換中、ウィザードの「**カーソル有効範囲とレポート・フォント・ファイル**」画面でデフォルト・フォント指定を受け入れるか、代わりのフォント指定を指定するようにプロンプトが出されます。

デフォルト・フォント指定は以下のようになっています。

```
locale name = en_us
    name = Courier New
    size = 10
    height = 12
    width = 6
    pdfFont = Courier
    encoding = CP1252
```

さらに、

*product\installation\eclipse\plugins\com.ibm.etools.i4gl.conversion\_version\etc\examples\FontSpecification* ディレクトリーにある **FontInfo.xml** ファイルにもこれらのデフォルト値がリストされています。ウィザードの使用中に代替の非デフォルト・フォント設定を指定するには、**FontInfo.xml** ファイルを編集して、使用したいフォント情報を含める必要があります。

以下は、デフォルトの **FontInfo.xml** ファイルの例です。

```
<locale name="en_us">
    <name>Courier New</name>
    <size>10</size>
    <height>12</height>
    <width>6</width>
```

```
<pdfFont>Courier</pdfFont>
<encoding>CP1252</encoding>
</locale>
```

ファイル内で次の **.xml** ファイルを定義します。

- **ロケール名:** フォントで識別される言語および国。各ロケールに 1 つのフォント指定に制限されます。ただし、同じ **.xml** ファイル内に複数の異なるロケール指定を組み込むことができます。
- **名前:** フォントの名前。
- **サイズ:** フォントのサイズ。これは、「10」のような整数リテラルでなければなりません。
- **高さ:** ピクセルで表したフォントの高さ。これは、「12」のような整数リテラルでなければなりません。

高さの値は、静的 1 行テキスト領域、1 行テキスト領域、バンドおよびページの高さ、および上下マージンのサイズなど、垂直位置決めを必要とするレポート要素を計算する場合に使用されます。

- **幅:** ピクセルで表したフォントの幅。これは「6」のような整数リテラルでなければなりません。EGL レポート出力が I4GL レポートと同様の出力を作成するためにするため、フォントは固定幅であることが必要です。

幅の値は、静的 1 行テキスト領域、1 行テキスト領域、ページおよび列の幅、左右マージン幅など、水平位置決めを必要とするレポート要素を計算する際に使用されます。

- **pdfFont:** レポートを Adobe PDF フォーマットにエクスポートする場合に使用するフォントの名前です。
- **エンコード方式:** Java のコード・ページを指定する英数字コードです。

何かの理由で、指定されたフォント指定が破損した場合は、変換ツールがデフォルト情報を **.jrxml** ファイルにマップします。

**注:** 変換ユーティリティーは、システム上で選択されたフォント名、フォント・サイズ、PDF フォント、またはエンコード方式の可用性を検証しません。これらの値は、指定されたとおりにフォント仕様ファイルから JasperReport 設計 (**.jrxml**) ファイルに転送されます。誤ったフォント情報が原因で、EGL のコンパイルやランタイム・エラーが発生する可能性があります。

以下に示すフォント指定選択のガイドラインに従ってください。

- 幅と高さの両方が固定されているフォントを使用します。
- 一般的なレポートのエクスポート・オプションと PDF エクスポート・オプションで、同様のフォントを使用します。
- フォントの高さおよびフォントの幅は、指定されたフォント・タイプおよびサイズのピクセル寸法の数値を反映していなければなりません。
- エンコード・ストリングが、ロケールを正しく反映していることを検証してください。

---

## 既存の I4GL コンポーネント・プロジェクトの識別

変換する I4GL プログラムを選択したら、次の作業を実行する必要があります。

1. 各 I4GL アプリケーションの I4GL ソース・モジュールとロケーションを識別および記録します。
2. I4GL アプリケーションにリンクする各 I4GL 共用ライブラリーと C 共用ライブラリーのファイル名とロケーションを識別および記録します。
3. 該当する場合は、既に変換済みの I4GL 共用ライブラリーから、マニフェスト・ファイルの名前とロケーションをリストする必要があります。

注: 後の変換前のステージで、C 共用ライブラリーの再コンパイルが必要になります。

---

## I4GL ソース・ファイルの生成

必要に応じて、**.4gl** および **.per** ソース・ファイルを生成します。

I4GL モジュールから I4GL ソース・ファイルを生成するために必要な前処理は、この時点ですべて完了していなければなりません。

---

## I4GL アプリケーションのコンパイル

必要に応じて I4GL アプリケーション・コードをコンパイルし、それが正常にコンパイルされることを検証します。I4GL アプリケーション・コードが正常にコンパイルされない場合、I4GL から EGL への変換は失敗します。正常にコンパイルされないコードを修復してください。すべての I4GL アプリケーション・コードが正常にコンパイルされるまでは、I4GL から EGL への変換を試行しないでください。

---

## クライアント・ロケールの識別

変換ウィザードは、メッセージ・ファイルのデフォルト・ロケールとして **en\_US.8859-1** の英語指定を使用します。メッセージ・ファイルが英語でない場合は、メッセージ・ファイルの正しいロケールを確認し、その情報をウィザードに挿入できるようにしなければなりません。

---

## 共用ライブラリーの識別と分離

I4GL プログラムに共用ライブラリーがある場合、それぞれのライブラリーを検討して、**.4gl** か、**.c** か、あるいは両方のタイプのソース・ファイルからコンパイルされているかを識別する必要があります。

1. **.4gl** ソース・ファイルからのみコンパイルされた共用ライブラリーは、さらに変換の準備を行う必要はありません。これは、I4GL 共用ライブラリー・プロジェクトとして変換されなければなりません。
2. **.c** ソース・ファイルからのみコンパイルされた共用ライブラリーは、さらに変換の準備を行う必要はありません。これは、アプリケーション・レベルの共用ライブラリーを作成するときにリンクされなければなりません。
3. **.4gl** および **.c** ソース・ファイルの両方からコンパイルされた共用ライブラリーは、以下の修正が必要です。

- a. **.4gl** ファイルは分離して、I4GL 共用ライブラリー・プロジェクトとして変換する必要があります。
- b. **.c** ファイルは、共用ライブラリーにコンパイルする必要があります。この共用ライブラリーは、アプリケーション・レベルの共用ライブラリーを作成するときにリンクされなければなりません。共用ライブラリーをリンクする方法について詳しくは、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。

I4GL アプリケーションが変換された後に、アプリケーション・レベルの共用ライブラリーを作成し、それに、C 共用ライブラリーをリンクします。このアプリケーション・レベルの共用ライブラリーは、新規の EGL アプリケーションで使用されます。詳しくは、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。

注: プログラムが、以前から存在する **.c** ソース・ファイルからコンパイルされたオブジェクト・ファイルから構成される静的ライブラリーを使用する場合は、それらの静的ライブラリーもアプリケーション・レベルの共用ライブラリーを作成するときにリンクされなければなりません。

## Rapid Development System (RDS) で使用する C コードの変更

RDS は、C 関数を I4GL プログラムから呼び出すためにカスタマイズされたランナーを使用するので、RDS を使用する I4GL 開発者は、次に詳しく説明するように C コードを変更する必要があります。

注: カスタマイズされたランナー とは、C 関数を呼び出す I4GL プログラムを実行するためにユーザーが作成する実行可能プログラムのことです。

カスタマイズされたランナーは、以下の構文で示すように **cfglgo** コマンドで作成します。

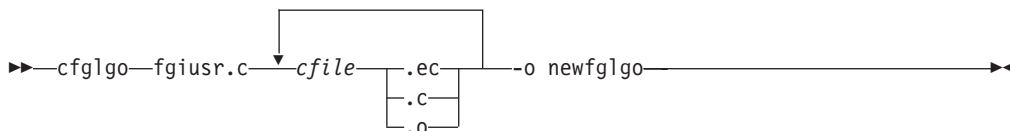


表 2-1. **cfglgo** 要素

要素	説明
fgiusr.c	C 関数および ESQ/C 関数が宣言されたファイルの名前。
cfile .ec.cl.o	コンパイルして新規のランナーにリンクする C または ESQ/C 関数を含むソース・ファイルの名前、あるいは以前に <b>.c</b> または <b>.ec</b> ファイルからコンパイルされたオブジェクト・ファイルの名前。
newfglgo	カスタマイズされたランナーの名前。

RDS を使用する場合は、すべての **.o**、**.c**、または **.ec** ファイルを 1 つの共用ライブラリーにコンパイルする必要があります。**.ec** ファイルがライブラリーにコンパイルされる場合は、そのライブラリーをコンパイルするときに ESQ/C 共用ライブラリーを使用する必要があります。この共用ライブラリーは、アプリケーション・レ

ベルの共用ライブラリーを作成するときにリンクされなければなりません。詳しくは、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。

**cfglgo** コマンドについて詳しくは、「*IBM Informix 4GL リファレンス・マニュアル*」の第 1 章を参照してください。

---

## ユーザー定義メッセージ・ファイルの識別

ユーザー定義のメッセージ・ファイルおよびメッセージをエンコードしたコード・ページをすべて識別します。

---

## Informix データベース・スキーマ情報の識別

変換プロセスの最初のステップは、変換アプリケーションのデータベース・スキーマを抽出することです。

1. I4GL プログラムのデータベースのインスタンスを作成します
2. スキーマ・ファイルを見つけて開きます。
3. 以下の情報を記録してください。
  - データベース名
  - サーバー名
  - ホスト名
  - ポート名
  - ユーザー名
  - パスワード
  - データベース・ロケール

デフォルトで、変換ユーティリティはシステム・テーブルを無視します。ただし、データベースを使用する I4GL アプリケーションが Informix システム・テーブルを参照する場合は、データベース内のすべてのシステム・テーブルのスキーマを抽出するオプションを選択できます。

---

## EGL 宛先ディレクトリーの識別

変換ユーティリティは、EGL 宛先ディレクトリーを使用して、Rational IDE が必要とする変換済み EGL ソース・ファイル、メッセージ・ファイル、およびプロジェクト固有のファイルを編成します。

ウィザードが提供する情報に基づいて、変換ユーティリティが EGL 宛先ディレクトリーを作成します。例えば、識別された EGL 宛先ディレクトリーが **C:\eglsrc** で、プロジェクト名が **MyProject** である場合、EGL 宛先ディレクトリー内で変換ユーティリティが **projectname\EGLSource\projectname** という新規サブディレクトリー構造を作成します。上記の例のパラメーターに従い、新規ディレクトリー構造は **C:\eglsrc\MyProject\EGLSource\MyProject** になります。

**project name** の最初のインスタンスが Rational IDE 固有ファイルである **.classpath**、**.eglPath**、および **.project** をホストして、変換されたプロジェクトを IDE にインポートするためのプロジェクト・コンテンツ・ディレクトリーの役割を果たします。

**EGLSource** サブディレクトリーは、すべての変換済み EGL ソース・ファイルのブレースホルダーの役割を果たし、すべての変換済み EGL ファイルのルート・パッケージ名を表す **projectname** サブディレクトリーをその中に含みます。変換済み EGL ファイルのディレクトリー階層は、I4GL ソース・ディレクトリー階層と同じで、**projectname** サブディレクトリー内にあります。

変換ユーティリティは、以下のディレクトリーを EGL 宛先ディレクトリー内に作成します。

- **projectname\EGLSource\projectname**。すべての変換済み EGL ソース・ファイルをホストします。
- **projectname\MessageSource**。メッセージ・ファイル・ロケールに基づいたすべての変換済み EGL メッセージ・ファイルをホストします。
- **project name\JavaSource**。IDE 内の EGL ソースから生成されたすべての Java ソース・ファイルをホストします。

注: 既存の I4GL ソース・ファイルと新規 EGL ソース・ファイルの両方で同じディレクトリーが識別される場合、変換ユーティリティは変換を開始する前に新規ディレクトリー構造を作成します。

変換ユーティリティは、システムに書き込み許可がなかったり、EGL 宛先ディレクトリーに十分なディスク・スペースがない場合には、変換を停止します。変換を正常に実行するためには、少なくとも既存の I4GL ディレクトリーと同等のディスク・スペースが必要です。

---

## I4GL ソース・ファイル・ディレクトリーの準備

変換ウィザードを起動する前に、すべての I4GL ソース・ファイル、共用ライブラリーおよびメッセージ・ファイルと一緒に 1 つのディレクトリーに配置されていることを確認しなければなりません。この I4GL ディレクトリーは、Rational 製品ワークスペース内か、Rational 製品から完全に外部の場所に配置できます。

さらに、I4GL プログラム・ソース・ファイルのストレージ用と EGL プロジェクト・ファイルの宛先ディレクトリーとして同じディレクトリーを指定することができます。このオプションによって、変換中に I4GL ソース・ファイルの保全性が保持され、I4GL および EGL ソース・ファイルのディレクトリーは 1 つという結果になります。

注: 変換ユーティリティによって、すべての I4GL プロジェクトを単一のディレクトリーに置き、そこから I4GL プロジェクトを変換することができますが、I4GL プロジェクトは、それぞれ個々のディレクトリー内に配置してください。そうすることで、複数の類似したファイルがあるために複雑な問題が発生する可能性が削減されます。

---

## 第 3 章 変換タスク

本章の内容	3-1
変換ユーティリティ・ステージ	3-1
Informix データベース・スキーマ抽出	3-1
Informix データベース・スキーマ抽出の変換ユーティリティ処理	3-3
I4GL 共用ライブラリー変換	3-4
I4GL 共用ライブラリーの変換ユーティリティ処理	3-7
I4GL アプリケーション変換	3-7
I4GL アプリケーション変換の変換ユーティリティ処理	3-10
変換ユーティリティのコマンド行モード	3-10
変換ログ	3-11

---

### 本章の内容

この章では、変換ウィザードを使用して Informix データベース・スキーマを抽出し、I4GL 共用ライブラリーおよびアプリケーションを変換する方法について、詳しく説明します。また、この章では変換ユーティリティのコマンド行オプションの使用方法について説明し、変換ログについても説明します。

---

### 変換ユーティリティ・ステージ

I4GL 変換ユーティリティは、『第 2 章 変換の準備』で必要に応じて収集し、識別した情報を入力するようにプロンプトを出すウィザードです。

変換を完了するには、I4GL プログラムのコンポーネントがこのウィザードを複数回使用する必要があります。さらに、次の順序で、ユーザーの情報をウィザードに入力しなければなりません。

1. Informix データベース・スキーマ
2. I4GL 共用ライブラリー
3. I4GL アプリケーション

C 共用ライブラリーは EGL に変換されないので、ウィザードを使用しません。C 共用ライブラリーの処理について詳しくは、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。

ix ページの『資料』で述べたように、ウィザードの各パネルに、それに関連するヘルプ・トピックがあります。このヘルプ・トピックには、**F1** を押すことでアクセスできます。

---

### Informix データベース・スキーマ抽出

変換プロセスの最初のステップは、プロジェクトのデータベース・スキーマを抽出することです。

注: 変換済みファイルのために生成されるインポートの数を制限するため、それぞれのデータベースを別々のプロジェクトとして抽出する必要があります。デー

データベース・スキーマ抽出プロジェクト内の各データベースは、別々のパッケージになりますが、すべてのデータベースが、データベース・スキーマ抽出プロジェクト用に生成された単一のマニフェスト・ファイルにリストされます。I4GL ファイルが参照しなくても、変換ユーティリティーはマニフェスト・ファイルにリストされているすべてのパッケージのインポートを挿入します。

ユーティリティーを開始して、Informix データベース・スキーマを抽出する手順は、次のとおりです。

1. Rational 製品の EGL パースペクティブから、「ファイル」>「新規作成」>「その他」>「EGL 変換に対する Informix 4GL」>「データベース・スキーマ抽出」を選択します。
2. 「データベース・スキーマ抽出」画面で、以下の情報を挿入します。

- プロジェクト名。例えば、**Mydbschema** です。

注: データベース・スキーマ抽出のプロジェクト名と共用ライブラリーおよびアプリケーション変換用のプロジェクト名は、異なるものでなければなりません。

- **新規プロジェクト。**データベース・スキーマ抽出は、既存の構成ファイルがない場合には、「新規」と見なされます。「ロケーション」については、参照して適切なディレクトリーを見つけるか、新規ディレクトリーを作成できます。
  - **プロジェクトを開く。**既存の構成ファイルがある場合、データベース・スキーマ抽出は「開く」と見なされます。このオプションを選択する場合は、既存構成ファイルの場所を指定する必要があります。参照してファイルを見つけることができます。
  - **変換成果物。**変換ユーティリティーは、構成ファイル、マニフェスト・ファイル、および変換ログ・ファイルなど、変換に関係する多くの成果物を生成します。デフォルトで、変換成果物は **EGLDestinationDirectory/ConversionArtifacts** ディレクトリーに置かれます。また、外部ディレクトリーに変換成果物を作成するように指定することもできます。適切なディレクトリーを参照することができますが、この時点では新規ディレクトリーを作成することはできません。
3. **データベース接続の詳細。**以下のような、Informix データベース・サーバー・インスタンスに接続するために必要な情報を追加、削除、または編集します。
    - データベース
    - サーバー名
    - ホスト名
    - ポート番号
    - ユーザー名
    - パスワード
    - クライアント・ロケール
    - データベース・ロケール
    - システム・テーブル。I4GL アプリケーションが Informix システム・テーブルの抽出を必要とする場合は、「はい」を選択します。

4. **変換プロジェクトの詳細。**前の画面で入力されたすべてのプロジェクトの詳細を統合した構成ファイルの **.xml** 文書を検討することができます。変換プロセスのこのステージでは、マニフェスト・ファイルと「変換レポート」画面には情報が入っていません。
5. 「完了」をクリックして、データベース・スキーマ抽出を開始します。
6. 「はい」をクリックしてデータベース・スキーマ抽出を確認し、開始してください。
7. 「スキーマ抽出状況 (Schema Extraction Status)」画面で、抽出が正常であったかどうかを確認します。抽出が正常に実行された場合は、情報が取り込まれたマニフェスト・ファイルを検討できます。このファイルには、選択されたデータベースの列とテーブルの情報が含まれます。ウィザードは **DATABASE** ステートメントを含む **I4GL** 共用ライブラリーやアプリケーションの変換中に、このマニフェスト・ファイルを参照します。必要に応じて、以下の 3-4 ページの『**I4GL** 共用ライブラリー変換』または 3-7 ページの『**I4GL** アプリケーション変換』に進みます。

変換済みプロジェクトの詳細な状況を示す変換ログ・ファイルが作成されます。

「スキーマ抽出状況 (Schema Extraction Status)」メッセージが正常でない場合、このデータベース・スキーマ抽出プロセスを最初から繰り返し、誤った情報が入力されていた場合にはそれを訂正してください。データベース・スキーマ抽出が正常に行われない場合がある理由については、4-7 ページの『変換エラーの訂正』を参照してください。

8. 「キャンセル」をクリックしてウィザードを閉じます。

## Informix データベース・スキーマ抽出の変換ユーティリティー処理

データベース・スキーマ抽出ステージは以下を開始します。

1. 変換ユーティリティーは、**I4GL** プログラムで使用されるそれぞれのデータベース用に個々の **EGL** パッケージを作成します。これらの **EGL** パッケージには、データベースに関連する **.egl** ファイルがあります。ファイルには、テーブルおよび各列に関連付けられたデータ項目にマッピングする **SQLRecord** 定義があります。
2. データベース内のそれぞれのテーブルに、対応する **EGL** ソース・ファイルが生成されます。
3. 収集したデータベース・スキーマは、他の **EGL** プロジェクトから参照される別々の **EGL** プロジェクト になります。
4. 構成ファイルとマニフェスト・ファイルが生成されます。
5. **IBM Informix 4GL 7.32** リリースでサポートされる **Informix** データ型のみが抽出されます。その他のデータ型は無視されます。

---

## I4GL 共用ライブラリー変換

I4GL アプリケーションが共用ライブラリーを使用している場合は、Informix データベース・スキーマを抽出した後、I4GL アプリケーションを変換する前に共用ライブラリーを変換する必要があります。I4GL アプリケーションに複数の共用ライブラリーがある場合は、変換ユーティリティーを使用して、各共用ライブラリーを別々に変換しなければなりません。基本的に、それぞれの共用ライブラリーが分離した変換プロジェクトになります。

変換済みの共用ライブラリーと I4GL アプリケーションとの間に依存関係があるため、I4GL アプリケーションは、すべての I4GL 共用ライブラリーが変換された後のみ変換することができます。

注: この I4GL 共用ライブラリー・ステージは C 共用ライブラリーでは使用されません。C 共用ライブラリーの変換後の処理については、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。

I4GL 共用ライブラリーを変換するには、次のようにします。

1. Rational 製品の EGL パースペクティブから、「ファイル」>「新規作成」>「その他」>「EGL 変換に対する Informix 4GL」>「I4GL 共用ライブラリー変換」を選択します。
2. 「I4GL 共用ライブラリー変換プロジェクト (I4GL Shared Library Conversion Project)」画面で次の情報を挿入してください。
  - プロジェクト名。例えば、Mysharedlibrary です。

注: データベース・スキーマ抽出のプロジェクト名と共用ライブラリーおよびアプリケーション変換用のプロジェクト名はすべて異なるものでなければなりません。

- プロジェクトの詳細。次のようなオプションがあります。
  - **新規プロジェクト。** 共用ライブラリー・プロジェクトは、既存の構成ファイルがない場合には、「新規」と見なされます。このオプションを選択する場合は、I4GL ソース・ディレクトリーと EGL ソース・ディレクトリーを指定する必要があります。参照してこれらのディレクトリーを見つけることができます。
  - **プロジェクトを開く。** 既存の構成ファイルがある場合、共用ライブラリー・プロジェクトは、「開く」と見なされます。このオプションを選択する場合は、既存構成ファイルのロケーションを指定する必要があります。参照してファイルを見つけることができます。
  - **再変換プロジェクト。** 以前に I4GL 共用ライブラリー・プロジェクトを実行し、共用ライブラリー・ファイルの再変換が必要であることが I4GL アプリケーション変換によって判別されている場合、共用ライブラリー・プロジェクトは、「再変換」と見なされます。このオプションを選択する場合は、既存構成ファイルおよびマニフェスト・ファイルのロケーションを指定する必要があります。参照してファイルを見つけることができます。

注: 再変換が必要になる場合について詳しくは、5-1 ページの『I4GL 共用ライブラリーを再変換する場合』を参照してください。

- **変換成果物**。変換ユーティリティーは、構成ファイル、マニフェスト・ファイル、および変換ログ・ファイルなど、変換に関係する多くの成果物を生成します。デフォルトで、変換成果物は **EGLDestinationDirectory/ConversionArtifacts** ディレクトリーに置かれます。外部ディレクトリーに変換成果物をするように指定することができます。適切なディレクトリーを参照することができますが、この時点では新規ディレクトリーを作成することはできません。
- 3. 「次へ」をクリックして先に進みます。
- 4. 「I4GL 変換プロジェクト・ファイル」画面に次の情報が表示されます。
  - 「クライアント・ロケール」に、データベース・スキーマ抽出ステージで入力した情報が表示され、これがメッセージ・ファイルを正しく変換するのに使用されます。この時点でロケールを変更することはできません。
  - 「プロジェクト・ファイル」には、前の「I4GL アプリケーション変換プロジェクト」画面で入力した I4GL ソース・ディレクトリー内にあるフォルダー、サブフォルダー、およびファイルが表示されます。以下のオプションがあります。
    - **フィルター・タイプ**。「プロジェクト・ファイル」画面で選択したファイルを限定することができます。「**タイプの選択**」ダイアログ・ボックスに、現在 Rational 製品でサポートされているファイル・タイプのリストが表示されます。**.4gl**、**.per** および **.msg** ファイルがデフォルトで選択されています。「**他の拡張子**」テキスト・ボックスで、**\*.msg** 以外のメッセージ・ファイル拡張子を入力します。
    - 「**すべて選択**」は、画面に表示されているルート・ディレクトリー内のすべてのファイルを選択します。
    - 「**選択をすべて解除**」は、「プロジェクト・ファイル」内のすべてのフォルダー、サブフォルダー、およびファイルの選択をすべて解除します。
  - 変換する I4GL フォルダーおよびファイルに関連するボックスをクリックします。
  - 「次へ」をクリックして先に進みます。
- 5. 「メッセージ・ファイルのロケール」画面には、前の「プロジェクト・ファイル」画面で選択したメッセージ・ファイルのみが表示されます。「メッセージ・ファイルのロケール」画面には、選択したメッセージ・ファイルのみが表示されます。
  - 「次へ」をクリックして先に進みます。
- 6. 「レポート・タイプおよびカーソル有効範囲」画面では、データベース・カーソルの有効範囲の状況と I4GL レポートのターゲット・レポート・エンジンを確認できます。レポート・タイプについては、「**テキスト・レポート**」を選択します。「**Jasper レポート**」オプションは、従来の変換用に維持されており、新しい変換には推奨されません。「データベース・カーソル有効範囲」のデフォルトは「local」です。カーソルの有効範囲を「global」に変更するには、「**変換用の SQL カーソルの有効範囲の設定**」チェック・ボックスを選択します。
  - 「次へ」をクリックして先に進みます。

7. 「データベース・スキーマの詳細」画面で、変換に必要なデータベースの存在とスキーマ情報を含むマニフェスト・ファイルを確認します。I4GL ライブラリーに DATABASE ステートメントが含まれていない場合は、この画面でのアクションは必要ありません。

この画面では、以下のアクションが必要です。

- データベース・スキーマの抽出ステージを正常に完了した場合、「**従属データベース・スキーマ (Dependent Database Schema)**」ボックスをクリックします。
  - 「**デフォルト・サーバー**」の名前を入力します。
  - 「**追加**」をクリックすると、「**マニフェスト・ファイルを従属プロジェクトから追加**」ボックスが表示されます。
  - データベース・スキーマの「**マニフェスト・ファイル**」のロケーションを入力します。このファイルは、データベース・スキーマ抽出ステージで作成した **ConversionArtifacts** ディレクトリーにあります。
  - 「**OK**」を選択して続行します。
  - 「**プロジェクト名**」と「**データベース・スキーマ抽出マニフェスト・ファイル**」のロケーションが画面に表示されています。この情報への追加、削除、または編集は、「**追加**」、「**削除**」または「**編集**」ボタンをクリックして、実行することができます。
  - 「**次へ**」をクリックして先に進みます。
8. **変換プロジェクトの依存関係 (Conversion Project Dependencies)** 画面で、変換済みアプリケーションが依存するすべての EGL パッケージ (前の I4GL 共用ライブラリー) を識別することができます。

必要に応じて、以下からボックスを選択します。

- 従属 I4GL 共用ライブラリー・プロジェクト
  - 「**追加**」をクリックすると、「**マニフェスト・ファイルを従属プロジェクトから追加**」ボックスが表示されます。
  - データベース・スキーマの「**マニフェスト・ファイル**」のロケーションを入力します。これは、I4GL 共用ライブラリー変換ステージで作成した **ConversionArtifacts** ディレクトリーにあります。
  - 「**OK**」を選択して続行します。
  - I4GL 共用ライブラリー・プロジェクト名およびマニフェスト・ファイルのロケーションが画面に表示されます。この情報への追加、削除、または編集は、「**追加**」、「**削除**」または「**編集**」ボタンをクリックして、実行することができます。
- 9. 「**次へ**」をクリックします。
- 10. **変換プロジェクトの詳細**。プロジェクトが「**新規プロジェクト**」であると、前の画面で入力されたすべてのプロジェクトの詳細を統合した構成ファイルの **.xml** 文書を検討することができます。変換プロセスのこのステージでは、マニフェスト・ファイルと「**変換レポート**」画面には情報が入っていません。プロジェクトが「**開く**」または「**再変換プロジェクト**」である場合は、この画面に情報が入力されたマニフェスト・ファイルと変換レポートが表示されます。
- 11. 「**完了**」をクリックして変換を開始します。

12. 「はい」をクリックすると I4GL2EGL 変換プログラムが起動します。

## I4GL 共用ライブラリーの変換ユーティリティー処理

I4GL 共用ライブラリー変換のステージでは、I4GL2EGL 変換ユーティリティーの起動によって次の作業が開始されます。

1. 変換ユーティリティーは、それぞれの I4GL 共用ライブラリー用に個々の EGL パッケージを作成します。
2. EGL パッケージのグループ全体が 1 つの EGL プロジェクト内に生成され、その他の EGL アプリケーションから参照することができます。
3. マニフェスト・ファイルが、その EGL プロジェクトに生成されます。

注: I4GL 共用ライブラリーはコマンド行モードでも変換することができます。ただし、このオプションは再変換作業の場合のみ推奨されます。コマンド行モードの使用方法について詳しくは、5-1 ページの『I4GL 共用ライブラリーの再変換の方法』を参照してください。

---

## I4GL アプリケーション変換

I4GL アプリケーションは、Informix データベース・スキーマが抽出され、すべての I4GL 共用ライブラリーが変換された後で変換可能になります。

I4GL アプリケーションを変換するには、次のようにします。

1. Rational 製品の EGL パースペクティブから、「ファイル」>「新規作成」>「その他」>「EGL 変換に対する Informix 4GL」>「I4GL アプリケーション変換」を選択します。
2. 「I4GL アプリケーション変換プロジェクト」画面で、以下の情報を挿入します。
  - プロジェクト名。例えば、**Myapplication** です。

注: データベース・スキーマ抽出のプロジェクト名と共用ライブラリーおよびアプリケーション変換用のプロジェクト名はすべて異なるものでなければなりません。

- **新規プロジェクト**。プロジェクトは、既存の構成ファイルがない場合には、「新規」と見なされます。このオプションを選択する場合は、I4GL ソース・ディレクトリーと EGL ソース・ディレクトリーを指定する必要があります。参照してこれらのディレクトリーを見つけることができます。
- **プロジェクトを開く**。既存の構成ファイルがある場合、プロジェクトは、「開く」と見なされます。このオプションを選択する場合は、既存構成ファイルのロケーションを指定する必要があります。参照してファイルを見つけることができます。

注: 共用ライブラリーを再変換することはできませんが、I4GL アプリケーションを再変換することはできません。I4GL アプリケーションの変換が失敗した場合は、変換ログで指摘されているエラーを解決し、I4GL アプリケーションを「新規」のプロジェクトまたは「開く」プロジェクトとして変換しなければなりません。

- **変換成果物。**変換ユーティリティーは、構成ファイル、マニフェスト・ファイル、および変換ログ・ファイルなど、変換に関係する多くの成果物を生成します。デフォルトで、変換成果物は **EGLDestinationDirectory/ConversionArtifacts** ディレクトリーに置かれます。また、外部ディレクトリーに変換成果物を作成するように指定することもできます。適切なディレクトリーを参照することができますが、この時点では新規ディレクトリーを作成することはできません。
3. 「次へ」をクリックして先に進みます。
  4. 「I4GL 変換プロジェクト・ファイル」画面に次の情報が表示されます。
    - 「クライアント・ロケール」に、データベース・スキーマ抽出ステージで入力した情報が表示され、これがメッセージ・ファイルを正しく変換するのに使用されます。この時点でロケールを変更することはできません。
    - 「プロジェクト・ファイル」には、前の「I4GL アプリケーション変換プロジェクト」画面で入力した I4GL ソース・ディレクトリー内にあるフォルダー、サブフォルダー、およびファイルが表示されます。以下のオプションがあります。
      - **フィルター・タイプ。**「プロジェクト・ファイル」画面で選択したファイルを限定することができます。「**タイプの選択**」ダイアログ・ボックスに、現在 Rational 製品でサポートされているファイル・タイプのリストが表示されます。**.4gl**、**.per** および **.msg** ファイルがデフォルトで選択されています。「**他の拡張子**」テキスト・ボックスで、**\*.msg** 以外のメッセージ・ファイル拡張子を入力します。
      - 「**すべて選択**」は、画面に表示されているルート・ディレクトリー内のすべてのファイルを選択します。
      - 「**選択をすべて解除**」は、「プロジェクト・ファイル」内のすべてのフォルダー、サブフォルダー、およびファイルの選択をすべて解除します。
    - 変換する I4GL フォルダーおよびファイルに関連するボックスをクリックします。
    - 「次へ」をクリックして先に進みます。
  5. 「メッセージ・ファイルのロケール」画面には、前の「プロジェクト・ファイル」画面で選択したメッセージ・ファイルのみが表示されます。「メッセージ・ファイルのロケール」画面には、選択したメッセージ・ファイルのみが表示されます。
    - 「次へ」をクリックして先に進みます。
  6. 「レポート・タイプおよびカーソル有効範囲」画面では、データベース・カーソルの有効範囲の状況と I4GL レポートのターゲット・レポート・エンジンを確認できます。レポート・タイプについては、「**テキスト・レポート**」を選択します。「**Jasper レポート**」オプションは、従来の変換用に維持されており、新しい変換には推奨されません。「データベース・カーソル有効範囲」のデフォルトは「local」です。カーソルの有効範囲を「global」に変更するには、「**変換用の SQL カーソルの有効範囲の設定**」チェック・ボックスを選択します。
    - 「次へ」をクリックして先に進みます。

7. 「データベース・スキーマの詳細」画面で、変換に必要なデータベースの存在とスキーマ情報を含むマニフェスト・ファイルを確認します。I4GL プログラムに DATABASE ステートメントが含まれていない場合は、この画面でのアクションは必要ありません。

この画面では、以下のアクションが必要です。

- データベース・スキーマの抽出ステージを正常に完了した場合、「**従属データベース・スキーマ (Dependent Database Schema)**」ボックスをクリックします。
  - 「**デフォルト・サーバー**」の名前を入力します。
  - 「**追加**」をクリックすると、「**マニフェスト・ファイルを従属プロジェクトから追加**」ボックスが表示されます。
  - データベース・スキーマの「**マニフェスト・ファイル**」のロケーションを入力します。このファイルは、データベース・スキーマ抽出ステージで作成した **ConversionArtifacts** ディレクトリーにあります。
  - 「**OK**」を選択して続行します。
  - 「**プロジェクト名**」と「**データベース・スキーマ抽出マニフェスト・ファイル**」のロケーションが画面に表示されています。この情報への追加、削除、または編集は、「**追加**」、「**削除**」または「**編集**」ボタンをクリックして、実行することができます。
  - 「**次へ**」をクリックして先に進みます。
8. **変換プロジェクトの依存関係 (Conversion Project Dependencies)** 画面で、変換済みアプリケーションが依存するすべての EGL パッケージ (前の I4GL 共用ライブラリー) を識別することができます。

必要に応じて、以下からボックスを選択します。

- 従属 I4GL 共用ライブラリー・プロジェクト
  - 「**追加**」をクリックすると、「**マニフェスト・ファイルを従属プロジェクトから追加**」ボックスが表示されます。
  - データベース・スキーマの「**マニフェスト・ファイル**」のロケーションを入力します。これは、I4GL 共用ライブラリー変換ステージで作成した **ConversionArtifacts** ディレクトリーにあります。
  - 「**OK**」を選択して続行します。
  - I4GL 共用ライブラリー・プロジェクト名およびマニフェスト・ファイルのロケーションが画面に表示されます。この情報への追加、削除、または編集は、「**追加**」、「**削除**」または「**編集**」ボタンをクリックして、実行することができます。
- 9. 「**次へ**」をクリックします。
- 10. **変換プロジェクトの詳細**。プロジェクトが「**新規プロジェクト**」であると、前の画面で入力されたすべてのプロジェクトの詳細を統合した構成ファイルの **.xml** 文書を検討することができます。変換プロセスのこのステージでは、マニフェスト・ファイルと「**変換レポート**」画面には情報が入っていません。プロジェクトが「**開く**」または「**再変換プロジェクト**」である場合は、この画面に情報が入力されたマニフェスト・ファイルと変換レポートが表示されます。
- 11. 「**完了**」をクリックして変換を開始します。

12. 「はい」をクリックすると I4GL2EGL 変換プログラムが起動します。

## I4GL アプリケーション変換の変換ユーティリティー処理

I4GL アプリケーション・ステージでは、I4GL2EGL 変換ユーティリティーの起動によって次の作業が開始されます。

1. パーサーが選択済みの I4GL アプリケーション・ファイルを変換します。
2. 関数呼び出し、関数引数、グローバル変数およびその他の情報を含むマニフェスト・ファイルが作成されます。
3. 変換済み I4GL 共用ライブラリー、および従属共用ライブラリー・プロジェクトからの I4GL ソース・コード・マニフェスト・ファイルの整合性と起こり得るエラーが検討されます。
4. 各メッセージ・ファイルが読み取られ、対応する Java コード・ページを使用して、適切な **.properties** ファイルに変換されます。
5. 現行プロジェクトが、以前に EGL 関数を外部関数と想定された名前で定義している場合、変換プログラムは従属 I4GL 共用ライブラリー・プロジェクトのマニフェスト・ファイルを更新および変更して変換を終了し、変換エラー・ログ内にエラーを報告します。この時点で、更新済みのマニフェスト・ファイルを使用して I4GL 共用ライブラリーを再変換する必要があります。
6. すべての **.4gl**、**.per**、**.msg**、およびカスタマイズされたエラー・メッセージ・ファイルが EGL に変換されます。
7. I4GL エラーが確認されると、変換プログラムは EGL 宛先ディレクトリー内に **.err** ファイルを生成し、エラー情報でログ・ファイルを更新します。
8. 変換が正常に行なわれた場合、デフォルトの EGL ビルド記述子ファイルが *EGLdestinationdirectory/project name/EGLSource* 内に生成されます。このファイルには *project name.eglbid* という名前が付けられます。この名前のファイルが既に EGL 宛先ディレクトリーに存在する場合、変換ユーティリティーは既存のファイルをオーバーライドする新規ファイルを作成しません。
9. 変換されているプログラムに C ライブラリーの依存関係がある場合、変換ユーティリティーは、それぞれの C 関数呼び出しを識別して関数テーブルを生成します。関数テーブルは **NativeFuncTab.c** という名前が付けられ、**ConversionArtifacts/NativeLibrary** ディレクトリーに配置されます。関数テーブルについて詳しくは、4-11 ページの『関数テーブル』を参照してください。
10. 変換ログが作成され、変換の統計および詳細が取り込まれます。
11. 変換ログ・ファイルは自動的に開いて表示されます。
12. 「キャンセル」をクリックしてウィザードを終了します。

---

## 変換ユーティリティーのコマンド行モード

変換ユーティリティー・ウィザードを使用して I4GL アプリケーションを EGL に変換する代わりに、変換ユーティリティーのコマンド行モードを使用することができます。ただし、最初の変換プロセスでコマンド行モードを使用するには、手動で構成ファイルを作成することが必要であるため、コマンド行モードは、既存の構成ファイルを使用することができる場合は共用ライブラリーの再変換で主に使用の方が簡単です。

コマンド行オプションは、***productinstallation/bin*** ディレクトリーにある **e4gl** という名前のスクリプトで起動します。

このスクリプトは以下のディレクトリーにあります。

- Linux:  
/opt/IBM/SDP71/bin
- Windows:  
C:\Program Files\IBM\SDP71\bin

コマンド行スクリプトを使用する前に、変換プロジェクト用の構成ファイルを作成する必要があります。構成ファイルは **.xml** フォーマットで、サンプル構成ファイルに提供されているデータ型定義 (DTD) に正確に従ったものでなければなりません。サンプル・ファイルをテンプレートとして使用しないで構成ファイルを作成しないでください。

次のようにして構成ファイルを作成します。

1. 以下のディレクトリーで、サンプル構成ファイルを見つけてください。  
plugins/com.ibm.etools.i4gl.conversion\_version/etc/dtd/Configuration
2. ファイルを **.xml** エディターで開き、ファイルを名前変更してから希望のディレクトリーに保管します。
3. 新規ファイルには、DTD が含まれ、これにより直感的に使用できる **.xml** タグが提供されます。これらのタグを使用して情報をファイルに挿入します。
4. ファイルを保管します。

次のようにしてコマンド行オプションを使用します。

1. コマンド行で、以下を入力します。ここで *configurationfile* は、新規に作成した構成ファイルの名前です。  
e4gl *configurationfile*

構成ファイルが現行作業ディレクトリーにない場合は、ファイルを絶対ディレクトリー・パスで修飾する必要があります。

コマンド行ユーティリティーを使用して、I4GL アプリケーションを変換する方法については、5-2 ページの『コマンド行の再変換』を参照してください。

**注:** スクリプトがデフォルトの ***productinstallation/bin*** ディレクトリーから移動されている場合は、RAD\_HOME 環境変数 (これを使用してスクリプトが変換 **.jar** ファイルを見つけます) が Rational IDE 製品インストールのルート・ディレクトリーをポイントするように設定されていなければなりません。

---

## 変換ログ

I4GL から EGL への変換の最後に、変換ログが開いて表示されます。ログには、警告、致命的エラー、共用ライブラリーおよびソース・ファイルの変換状況などの情報を含む変換の詳細があります。変換ログは、主に変換エラーを識別するために使用されます。変換ログを使用して変換エラーを解決する方法について詳しくは、4-7 ページの『変換エラーの訂正』を参照してください。



---

## 第 4 章 変換後のタスク

本章の内容	4-1
変換後のタスク	4-2
変換中に行われる変更点	4-2
変換中に生成される成果物	4-3
構成ファイル	4-5
マニフェスト・ファイル	4-5
ソース・ファイル変換マッピング	4-6
コマンド行変換: プロジェクトのワークスペースへのインポート	4-7
変換エラーの訂正	4-7
変換ログの内容	4-9
EGL プログラムでの C 共用ライブラリーの使用	4-10
EGL ネイティブ・ライブラリー	4-11
関数テーブル	4-11
アプリケーション・レベルの共用ライブラリーの作成	4-12
プロパティ・ファイル	4-13
変換済み EGL ファイルの検証とコンパイル	4-15
EGL の Java への生成	4-15
エラー・メッセージ変換の理解	4-16
レポート変換の理解	4-17
EGL レポート・ドライバ関数	4-18
I4GL レポート・セクション	4-19
I4GL DEFINE セクション	4-20
I4GL OUTPUT セクション	4-21
I4GL ORDER BY セクション	4-22
I4GL FORMAT セクション	4-23
EGL プロジェクト、パッケージ、およびファイルの理解	4-32
EGL プロジェクト	4-32
EGL ソース・フォルダー	4-33
EGL ビルド・パス	4-33
デフォルトのビルド記述子	4-33
パッケージ	4-33
EGL ファイル	4-34
ソース・ファイル	4-34
ビルド・ファイル	4-35
推奨事項	4-35
ビルド記述子の場合	4-35
パッケージの場合	4-35
パーツの割り当て	4-35
インフォメーション・センターのヘルプ・システムと EGL チュートリアル	4-36

---

### 本章の内容

この章では、変換ユーティリティが I4GL アプリケーションに対して行う変更点、変換中に生成される成果物、変換エラーを訂正する方法、および EGL アプリケーションで C 共用ライブラリーを使用する方法について説明します。この章では、EGL Information Center のオンライン・ヘルプ・システムの用法についても説明します。

---

## 変換後のタスク

変換済みの EGL プログラムを使用する前に、必要に応じて以下のタスクおよび情報を検討または実装してください。

- 『変換中に行われる変更点』
- 4-3 ページの『変換中に生成される成果物』
- 4-6 ページの『ソース・ファイル変換マッピング』
- 4-7 ページの『コマンド行変換: プロジェクトのワークスペースへのインポート』
- 4-7 ページの『変換エラーの訂正』
- 4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』
- 4-13 ページの『プロパティ・ファイル』
- 4-15 ページの『変換済み EGL ファイルの検証とコンパイル』
- 4-15 ページの『EGL の Java への生成』
- 4-16 ページの『エラー・メッセージ変換の理解』
- 4-17 ページの『レポート変換の理解』
- 4-32 ページの『EGL プロジェクト、パッケージ、およびファイルの理解』
- 4-36 ページの『インフォメーション・センターのヘルプ・システムと EGL チュートリアル』

---

## 変換中に行われる変更点

3 ステージにわたる I4GL から EGL への変換中に、以下のような変更が行われます。

- 変換ユーティリティーが、変換済み EGL ソース・ファイルをホストする新しいルート・ディレクトリー構造を EGL 宛先ディレクトリーに作成します。ワークスペース・リソース・フォルダーも作成されます。このフォルダーにはデフォルトでサブディレクトリー ***EGLDestinationDirectory/ProjectName/EGLSource/ProjectName*** があります。
- 変換済みアプリケーションがいずれかの C 共用ライブラリーに依存している場合は、EGL ネイティブ・ライブラリーおよび関数テーブルが生成されます。EGL ネイティブ・ライブラリーおよび関数テーブルを使用して C 共用ライブラリーと EGL プログラム間のリンクを作成する方法については、4-10 ページの『EGL プログラムでの C 共用ライブラリーの使用』を参照してください。
- EGL プロジェクト・ビルド記述子ファイルが作成され、EGL パースペクティブに変換済み **.egl** ファイルをビルドするために使用されます。
- I4GL 構文の構成は EGL 構文にマップされます。I4GL と EGL の完全なマッピングについては、A-1 ページの『付録 A. I4GL から EGL への構文のマッピング』を参照してください。
- **.4gl** ソース・ファイルはすべて **.egl** ソース・ファイルに変換されます。
- I4GL **.per** 書式仕様ファイルはすべて **.egl** コンソール・ユーザー・インターフェース仕様ファイルに変換されます。

注: EGL では、ファイルはすべて **.egl** ファイルです。I4GL ファイルと EGL ファイルの完全なマッピングについては、4-6 ページの『ソース・ファイル変換マッピング』を参照してください。

- I4GL カスタマイズ済みエラー・メッセージ・ファイルは EGL にマイグレーションされます。これらのエラー・メッセージ・ファイルについては、4-16 ページの『エラー・メッセージ変換の理解』を参照してください。
- I4GL レポート・ロジックは、同等の EGL および JasperReports フォーマットに変換されます。変換済みレポートについては、4-17 ページの『レポート変換の理解』を参照してください。I4GL と EGL 間のマッピングについて検討するには、A-12 ページの『4GL レポート実行ステートメント』と A-12 ページの『I4GL レポート・ドライバー・ステートメント』を参照してください。

注: I4GL プログラムにレポート機能が含まれていない場合でも、この機能を変換済みプログラムに追加できます。EGL プロジェクトに新規レポートを作成する方法については、インフォメーション・センターのヘルプ・トピック『EGL レポート』を参照してください。

## 変換中に生成される成果物

変換ユーティリティは、広義で 2 つのカテゴリの変換成果物を生成します。

- 構成ファイルやマニフェスト・ファイルなどの、プロジェクト固有の成果物。
- I4GL 入力ファイルの構文エラーから生成されるエラー・ファイル。

エラー・ファイルは、入力 I4GL ファイルで使用するディレクトリと同じディレクトリに置かれます。デフォルトでは、他の変換成果物はすべて **EGLDestinationDirectory/ConversionArtifacts** ディレクトリに置かれます。データベース・スキーマの抽出中、または変換ユーティリティの I4GL 共用ライブラリー変換ステージで、異なるディレクトリーのロケーションを指定できます。

変換成果物ディレクトリーには以下のサブディレクトリーが含まれます。

1. **/config**。変換ユーティリティが新規プロジェクトの構成ファイルを生成するためのプレースホルダー。
2. **/manifest**。変換ユーティリティが現行の変換プロジェクトのマニフェスト・ファイルを生成するためのプレースホルダー。
3. **/log**。変換ユーティリティが現行の変換プロジェクトの変換ログ・ファイルを生成するためのプレースホルダー。
4. **/NativeLibrary**。変換ユーティリティが C 共用ライブラリーと EGL のリンクに使用する関数テーブルを生成するためのプレースホルダー。

変換ユーティリティは以下の成果物を生成します。

表 4-1. 変換ユーティリティによって生成される成果物 (文書)

成果物	ファイル名	説明
構成ファイル	<b>ProjectNameProjectType Config.xml</b> 例えば、 <b>MyDatabaseSchema Config.xml</b>	プロジェクト、データベース・スキーマ、ディレクトリー、およびファイル情報を含みます。 詳しくは、4-5 ページの『構成ファイル』を参照してください。

表 4-1. 変換ユーティリティによって生成される成果物 (文書) (続き)

成果物	ファイル名	説明
変換ログ	<b>ProjectNameProjectType</b> <b>Log.txt</b> <b>ProjectNameProjectType</b> <b>Log.html</b>	変換エラーと警告、従属ライブラリーのリスト、およびソース・ファイルの変換状況などの、プロジェクト情報を含みます。このログは、失敗した変換についての主要な情報源です。致命的エラーによって変換プロセスが停止しない限り、変換が成功しても失敗しても、このログが生成されます。詳細については、4-7 ページの『変換エラーの訂正』を参照してください。
EGL ネイティブ・ライブラリー	4-11 ページの『EGL ネイティブ・ライブラリー』を参照してください。	I4GL 共用ライブラリーまたはアプリケーションの変換中に検出した C 関数のプロトタイプを含みます。
ERR ファイル	<b>sourcefilename.err</b> <b>sourcefilenameForm.err</b>	I4GL ソース構文の EGL への変換中に検出されたエラーを識別します。
関数テーブル	4-11 ページの『関数テーブル』を参照してください。	I4GL アプリケーションの変換中に検出したすべての C 関数の名前を含みます。
マニフェスト・ファイル	<b>ProjectNameProjectType</b> <b>Manifest.xml</b>	依存関係を解決するために必要なプロジェクト固有の情報を含みます。データベース・スキーマ抽出プロジェクトでは、マニフェスト・ファイルにテーブル/列の情報が含まれます。共用ライブラリーまたは I4GL アプリケーションのプロジェクトでは、マニフェスト・ファイルに、関数呼び出しと戻りの型、グローバル変数、フォーム名、レコード・タイプ、パッケージ名などの情報が含まれます。詳しくは、4-5 ページの『マニフェスト・ファイル』を参照してください。

注: 変換ユーティリティは、ファイル名を上記の表で表示したとおりに生成します。変換ユーティリティが、変換中に変換成果物ディレクトリーでこれらのファイル名を検出した場合は、既存のファイルを **filename.bak.num** に名前変更します。ここで、**num** は 1 から始まり、連続して増える数値です。変換ユーティリティは、上記の表 4-1 で説明したように新規ファイルに名前を付けます。

## 構成ファイル

3 ステージにわたる変換ユーティリティの各ステージで、ユーティリティは、特定のプロジェクト、データベース・スキーマ、ディレクトリー、およびファイル情報を、個別の構成ファイルにコンパイルします。したがって、構成ファイルには変換についての重大な情報がすべて保管されます。必要に応じて参照してください。

各ファイルは、ユーティリティのどのステージで生成されたかに従って、名前が付けられます。例えば、**MyDB** というデータベース・スキーマ抽出プロジェクトの場合は、変換ユーティリティのこのステージでの構成ファイルには **MyDBSchemaConfig.xml** という名前が付けられます。

I4GL プロジェクトを Rational 環境の外部で変換したい場合は、構成ファイルを手動で作成できます。サンプル構成ファイルと DTD が製品に含まれています。構成ファイルを手動で作成する場合は、製品構成ファイル DTD に準拠するようにしてください。

サンプル構成ファイルは **conversion.sample.xml**、DTD は **conversionconfig.dtd** という名前です。両方とも以下のディレクトリーにあります。

```
productinstallation/egl/eclipse/plugins/com.ibm.etools.i4gl.conversion_version/  
etc/dtd
```

データベース・スキーマ抽出プロジェクトと I4GL アプリケーション・プロジェクトのサンプル構成ファイルは、以下のディレクトリーにあります。

```
productinstallation/egl/eclipse/plugins/com.ibm.etools.i4gl.conversion_version/  
etc/examples/Configuration
```

**注:** 変換ユーティリティ・ウィザードを使用すると、変換ユーティリティが自動的に構成ファイルを生成します。 **etc** ディレクトリーにリストされるサンプル構成ファイルは、自分のプロジェクト仕様に従って変更できるテンプレート・ファイルです。テンプレート・ファイルを使用する場合は、DTD に従い、有効な整形式 XML 文書を作成してください。XML 仕様に合わない、手動で編集された構成ファイルがあると、変換は終了します。

構成ファイルのテンプレートの例については、『付録 D. 構成ファイルのテンプレート』を参照してください。構成ファイルおよびマニフェスト・ファイルに使用される DTD の例については、『付録 F. DTD 例』を参照してください。

## マニフェスト・ファイル

変換の 3 つのステージすべてで、ステージに固有のマニフェスト・ファイルが生成されます。

データベース・スキーマの抽出中に生成されたマニフェスト・ファイルには、選択されたデータベースのすべてのテーブル、列、およびデータ型についての情報が含まれます。このファイルは、I4GL と EGL 間の列の名前とデータ型を解決するために、変換ユーティリティによって使用されます。

I4GL 共用ライブラリーの変換中に生成されたマニフェスト・ファイルには、I4GL 変換プロジェクトで使用される、I4GL と、想定される C 関数呼び出しのリストが

含まれます。このファイルは、I4GL アプリケーション変換ステージで、従属 EGL パッケージと I4GL ソース・コードで使用される、関数呼び出しをすべて検証するために使用されます。

**注:** 変換ユーティリティーは、I4GL アプリケーション変換プロジェクトのためのマニフェスト・ファイルも生成します。ただし、アプリケーションのマニフェスト・ファイルは参照用にのみ生成され、プロジェクトの技術的な詳細をリストします。

マニフェスト・ファイルの例については、『付録 E. マニフェスト・ファイルの例』を参照してください。

## ソース・ファイル変換マッピング

変換プロセス中に、I4GL プログラム・ファイルが同等の EGL プロジェクト・ファイルにマップされました。これらのファイル間のマッピングについては、表 4-2 に記載されています。

表 4-2. I4GL ファイルの EGL ファイルへの変換方法

ファイルおよびライブラリー型	I4GL ファイル拡張子	EGL ファイル拡張子
ソース・ファイル	<b>.4gl</b>	<b>.egl</b>
書式仕様ファイル	<b>.per</b> I4GL フォーム・ファイルを変換する方法の例については、『付録 C. I4GL 書式コードから EGL 書式コードへの例』を参照してください。	<b>.egl</b> 定数ストリング <b>Form</b> が、変換されたすべての書式仕様ファイル名に追加されます。例えば、 <b>someFile.per</b> という名前の I4GL ファイルは、EGL では <b>someFileForm.egl</b> という名前になります。
レポート・ファイル	<b>.4gl</b>	<b>.egl</b> <b>注:</b> I4GL レポート設計ロジックは、JasperReports <b>.jrxml</b> ファイルに変換されます。レポート・ファイル変換についての詳細は、4-17 ページの『レポート変換の理解』を参照してください。
カスタマイズされたヘルプおよびエラー・メッセージ・ファイル	<b>.msg</b> またはその他の拡張子	<b>.properties</b>
I4GL ソース・ファイルから作成される共用ライブラリー	<b>.so</b> または、その他のプラットフォームで同等の拡張子	EGL パッケージ
ビルド・ユーティリティー・ファイル	Make ファイル	<b>.eglbld</b> 、EGL ビルド記述子ファイル

---

## コマンド行変換: プロジェクトのワークスペースへのインポート

変換ユーティリティ・ウィザードは、変換済みのプロジェクトを自動的に「**EGL プロジェクト**」カテゴリー内のワークスペースにインポートします。プロジェクトの変換にコマンド行オプションを使用した場合は、プロジェクトを手動で「**EGL プロジェクト**」にインポートする必要があります。 データベース・スキーマ、共用ライブラリー、およびアプリケーションのプロジェクトを個別にインポートしてください。

データベース・スキーマ、共用ライブラリー、およびアプリケーションのプロジェクトをインポートするには、次のようにします。

1. EGL パースペクティブで、「**プロジェクト・エクスプローラー**」ワークスペースの任意の場所を右クリックします。
2. メニューから「**インポート**」を選択します。
3. 「**インポート・ウィザード**」から「**既存プロジェクトをワークスペースへ**」を選択します。「**次へ**」を選択します。
4. 「**プロジェクトをファイル・システムからインポート**」ウィンドウの「**プロジェクト・コンテンツ**」テキスト・ボックスで、「**参照**」を選択します。
5. 「**フォルダーの参照**」ウィンドウで、変換済みプロジェクトの EGL 宛先ディレクトリーを選択します。「**OK**」を選択して「**フォルダーの参照**」ウィンドウを終了します。「**完了**」を選択して「**プロジェクトをファイル・システムからインポート**」ウィンドウを終了します。

注: 変換ユーティリティは、EGL 宛先ディレクトリーに **.project**、**.eglPath**、および **.classpath** ファイルを作成します。これらのファイルは、Rational 製品がプロジェクトを識別するために使用します。これらのファイルが欠落すると、プロジェクトをワークスペースにインポートできません。

6. 選択したディレクトリーが、「**プロジェクト・コンテンツ**」内の「**インポート・ウィザード**」に表示されるようになります。プロジェクト名は「**プロジェクト名**」に表示されます。

注: Rational 製品が、選択されたディレクトリー内に **.project** ファイルを見つけないと、エラーが戻されます。続行するには、ディレクトリーにそのファイルがあることを検証してください。

7. 「**終了**」をクリックします。プロジェクトがワークスペース内で表示されるようになります。

---

## 変換エラーの訂正

変換ユーティリティは、変換プロジェクトの状況の理解と、変換エラーの訂正に役立つ、変換ログを生成します。変換ユーティリティは、この変換ログを、指定された成果物のディレクトリーである **log** サブフォルダーに保管します。このファイルは **ProjectNameProjectTypeLog.html** という名前で識別できます。ここで、**ProjectType** は **Schema**、**Library**、または **Application** となります。**.html** ファイルに加え、**.txt** ファイルが生成されます。

変換プロジェクトは、**PASSED** か **FAILED** に分類されます。**PASSED** 分類は、変換に成功し、EGL コードを使用できるということです。

1 つ以上の EGL ソース・ファイルを生成できないと、プロジェクトが FAILED と分類されます。以下のいずれかの理由により、プロジェクトが失敗する可能性があります。

- 変換ユーティリティが、識別された EGL ソース・ディレクトリーまたは変換成果物ディレクトリーへの書き込み許可を持っていませんでした。
- XML 構成ファイルまたは従属マニフェスト・ファイルにエラーが含まれています。
- データベース・スキーマ抽出プロジェクトでは、データベースへの接続許可がない場合、パスワードを誤った場合、またはその他の例外に気付いた場合に、JDBC ドライバーがプロセスを終了します。
- 共用ライブラリーまたは I4GL アプリケーションのプロジェクトでは、構成ファイルにリストされたソース・ファイルが読み取り許可を持っていないか、欠落していると、変換は終了します。
- I4GL 構文を EGL 構文に変換できませんでした。

検証のステージでプロジェクトに障害が起こると、変換は終了します。検証のステージには、XML 構成ファイル、マニフェスト・ファイル、ソース・ファイル、JDBC 接続、および書き込み許可と読み取り許可の検証が含まれます。I4GL ソース・ファイル内の構文エラーのためにプロジェクトに障害が起こる場合、変換は続行しますが、プロジェクトは FAILED とマークを付けられます。その他のステージでプロジェクトに障害が起こると、変換は続行します。いずれの場合も、変換ログが生成されます。FAILED プロジェクトは再び最初から変換しなくてはなりません。

すべての変換エラーを訂正してから、すべての FAILED プロジェクトで再度変換プロセスを行う必要があります。このプロセスで変換ウィザードを使用する場合は、「プロジェクトを開く」オプションを選択してください。既存の構成ファイルを使用できるようになります。

**注:** FAILED プロジェクトの変換は、共用ライブラリーの再変換とは異なります。FAILED プロジェクトでは、3 ステージの変換ウィザードの手順、つまり、データベース・スキーマ抽出、共用ライブラリー変換、および I4GL アプリケーション変換を実行する必要があります。共用ライブラリーで再変換が必要な場合は、共用ライブラリーの手順を使用する必要があるだけです。

個別のファイルはさらに以下のいずれかに分類できます。

- **PASSED.** ファイルが正常に変換されたことを確認します。ユーザー・アクションは必要ありません。
- **ERROR.** 変換全体が失敗したことを識別します。多くの場合、どのように失敗したかを明示的に識別します。変換ウィザードを使用して I4GL プログラムを再変換する必要があります。
- **FIXME.** 変換の 1 つ以上のステージが失敗したことを識別しますが、変換ウィザードを使用した変換は必要ありません。この分類からの情報は、EGL パースペクティブで、手動で変換済みファイルの問題を修正するために十分な情報です。EGL コード内で、各 **FIXME** は、例えば **//FIXME:** のような、**FIXME** タグで識別されます。さらに、各 **FIXME** と **TODO** には、元の I4GL ファイルへの行および列の参照が含まれます。

EGL コード内で FIXME を識別するだけでなく、各 FIXME は「EGL タスク (EGL Tasks)」ビュー内にリストされます。各 FIXME の情報を使用して、コードを訂正してください。

- **TODO。** 変換はパスしたが、警告があったことを識別します。警告は、無視することも、EGL パースペクティブで、それぞれを手動で対処することもできます。各警告を検証してください。

## 変換ログの内容

変換ログには以下の情報が含まれます。

- **変換状況。** このセクションには以下のものが含まれます。
  - プロジェクト状況
  - 変換日
  - ユーザー名
  - ホスト
  - OS のバージョン
- **プロジェクトの詳細。** このセクションには以下のものが含まれます。
  - プロジェクト名
  - ディレクトリー情報
  - 変換タイプ
  - デフォルト Informix サーバー・インスタンス
- **変換成果物。** このセクションには、変換成果物のファイル名が含まれます。
- **I4GL ソース・ファイル変換要約。** このセクションには、入力として提供される I4GL ソース・ファイルの数と、出力として EGL に変換されるファイルの数がリストされます。EGL ファイルの数が I4GL ファイルの数を超える場合があります。
- **ソース・ファイル変換の詳細。** このセクションは、共用ライブラリーおよびアプリケーションのプロジェクトについてのみ表示され、以下を識別します。
  - 入力 I4GL ソース・ファイルと出力 EGL ソース・ファイルの名前。
  - 各ファイルの変換状況。
  - 必要に応じて、ERR ファイルのロケーション。
- **例外。** このオプションのセクションは、サポートされない構文の警告、重大なエラーと重大でないエラー、起こりうる変換ユーティリティの内部機能障害の例外を含む、あらゆる変換の問題を識別します。このセクションでは、アプリケーション・プロジェクトのマニフェスト・ファイルの整合性検査中に識別される、再変換の要件を記録できます。
- **データベース接続の詳細。** このセクションは、データベース・スキーマ抽出のプロジェクトについてのみ表示され、以下を識別します。
  - サーバー名
  - データベース名
  - テーブル名
  - このテーブルのために抽出される、対応する EGL ソース・ファイル

注: データベース抽出と、上記のソース・ファイル変換の詳細のセクションの両方で、変換ログの **.html** バージョンに、入出力ファイルとディレクトリーへのハイパーリンクがあります。

変換が完了すると、ログ・ファイルは「**レポート**」タブに表示されます。デフォルトのブラウザがシステムにある場合は、変換ユーティリティーが自動的にブラウザと変換ログを開いて表示します。変換が完了したらすぐにログ・ファイルの内容を検討してください。変換エラーや警告を訂正するには、ログ・ファイルの内容を理解している必要があります。

コマンド行モードを使用する場合、変換状況は、コンソールか、変換ユーティリティーが起動されるシェル・ウィンドウに表示されます。ログ・ファイルは **EGLDestinationDirectory/ConversionArtifacts/log** ディレクトリーに生成されます。表示するにはこのファイルを開く必要があります。

注: 変換ログは、JDBC ドライバーによって戻されるエラーも表示します。ご使用のドライバーがエラーの説明を提供する場合は、その説明はドライバーから抽出され、変換ログに表示されます。ご使用のドライバーがエラーの説明を提供しない場合は、JDBC ドライバーが提供するエラー番号のみが変換ログに表示されます。

変換ログ内に含まれるファイル情報に加え、変換ユーティリティーは **ERR** ファイルも生成して、EGL に正常に変換されなかった **I4GL** 構文ステートメントを識別します。ERR ファイル情報を使用して、**I4GL** コード内の構文エラーを訂正してから、**I4GL** アプリケーションを再変換できます。**.4gl** ファイルでは、ERR ファイルは **sourcefile.err** という名前です。**.per** ファイルでは、ERR ファイルは **sourcefileForm.err** という名前です。

変換ログの例については、『付録 G. 変換ログの例』を参照してください。変換ユーティリティーのエラー・メッセージのリストを検討するには、エラー・メッセージを参照してください。

---

## EGL プログラムでの C 共用ライブラリーの使用

**I4GL** プログラムに共用ライブラリーがあった場合、変換前のタスクの 1 つとして、各ライブラリーを検討し、**.4gl** ソース・ファイル、**.c** ソース・ファイル、またはこの両方の、いずれからコンパイルされたものを識別します。**.4gl** ソース・ファイルと **.c** ソース・ファイル両方からコンパイルされた共用ライブラリーについては、**.c** ファイルを分離して C 共用ライブラリーにコンパイルしました。さらに、元は **.c** ソース・ファイルからのみコンパイルされた共用ライブラリーや、**.c** ソース・ファイルからコンパイルされたオブジェクト・ファイルからなる既存の static ライブラリーを識別した可能性があります。このセクションでは、アプリケーション・レベルの共用ライブラリーを作成し、識別されている C ライブラリーの種類とリンクさせる方法について説明します。

**I4GL** 共用ライブラリーまたはアプリケーションの変換中に、このプロジェクトで検出した C 関数のプロトタイプは、このプロジェクトのために作成された EGL ネイティブ・ライブラリーで定義されました。**I4GL** アプリケーション変換の最後に、すべての C 関数を識別する関数テーブルが生成されました。これらのコンポーネントについてもこのセクションで説明します。

## EGL ネイティブ・ライブラリー

I4GL 共用ライブラリーまたはアプリケーションの変換中に、この変換プロジェクトで検出した C 関数のプロトタイプは、このプロジェクトのために作成された EGL ネイティブ・ライブラリーで定義されます。このライブラリーは、関数本体のない関数プロトタイプで構成されます。C 関数は C ライブラリーの内側で定義され、EGL ネイティブ・ライブラリーは EGL IDE 内の検証エラーを避けるために使用されます。ネイティブ・ライブラリーの名前は **CExternals** で、**EGLDestinationDirectory/ProjectName/EGLSource/CExternals** ディレクトリーに作成されます。

EGL ネイティブ・ライブラリーの詳細については、『*nativeLibrary* 型のライブラリー・パーツ』ヘルプ・トピックを参照してください。

## 関数テーブル

関数テーブルは、I4GL アプリケーション変換の最終ステージで生成されます。EGL プログラムから呼び出されるすべての C 関数の名前を含むものが C ソース・ファイルです。関数テーブルの名前は **NativeFuncTab.c** で、**EGLDestinationDirectory/ConversionArtifacts/NativeLibrary** ディレクトリーにあります。

次の関数テーブル例では、**c\_fun1** と **c\_fun2** が C 関数の名前です。

```
#include <stdio.h>
struct func_table {

    char *fun_name;
    int (*fptr)(int);
};

extern int c_fun1(int);
extern int c_fun2(int);
/* Similar prototypes for other functions */

struct func_table ftab[] =
{
    "c_fun1", c_fun1,
    "c_fun2", c_fun2,
    /* Similarly for other functions */
    "", NULL
};
```

以下の場合、関数テーブルと EGL アプリケーション・コードを両方とも変更する必要があります。

1. 既存の C 関数の名前が変更される場合。EGL コード内の関数呼び出しステートメントを変更するだけでなく、以下も変更する必要があります。
  - ・ 関数テーブル内の C 関数の名前。
  - ・ 関数プロトタイプは、関数が呼び出される各プロジェクトに対応する EGL ネイティブ・ライブラリーで定義する必要があります。
2. 新しい C 関数が追加される場合。以下の両方を変更します。
  - ・ 新しい C 関数の名前を関数テーブルに追加する必要があります
  - ・ 関数が含まれる各プロジェクトに対応する EGL ネイティブ・ライブラリーで定義される関数プロトタイプ。

注: 2 つの C 関数が同じ名前を共有している場合、実際の関数呼び出しは共用ライブラリーのリンク順序に応じて決まります。

## アプリケーション・レベルの共用ライブラリーの作成

このセクションでは、アプリケーション・レベルの共用ライブラリーを作成する方法と、これを、変換前の間に識別または作成された C ライブラリーとリンクさせる方法について説明します。

アプリケーション・レベルの共用ライブラリーの作成には 2 つの部分があります。

1. EGL スタック・ライブラリーおよびアプリケーション・オブジェクト・ファイルを、IBM Web サイトからコンピューターにダウンロードします。
2. この関数テーブルと適切なプラットフォーム固有のアプリケーション・オブジェクト・ファイルをアプリケーション・レベルの共用ライブラリーにコンパイルし、この共用ライブラリーを、適切なプラットフォーム固有のスタック・ライブラリーと変換前の間に識別または作成された C ライブラリーにリンクします。

**EGL スタック・ライブラリーとアプリケーション・オブジェクト・ファイルをダウンロードする手順は、次のとおりです。**

1. スタック・ライブラリーは、EGL コードと C コードとの間で値を受け渡したり戻したりするために使用する。C コードで使用されるポップおよび戻り API は、スタック・ライブラリーで解決されます。アプリケーション・オブジェクト・ファイルは、EGL プログラムと C コードの間のインターフェースとして機能します。スタック・ライブラリーとアプリケーション・オブジェクト・ファイルは両方とも、プラットフォーム固有のコンポーネントです。

- a. 次の Web サイトにある **EGLRuntimesV70.zip** という名前のファイルを見つけてください。

<http://download.boulder.ibm.com/ibmdl/pub/software/rationalsdp/v7/rbd/70/redist/EGLRuntimes/>

注: EGL スタック・ライブラリーは、次のロケーションのインストール・メディアから入手可能です。

`disk1/redist/EGLRuntimes/platform/`

- b. **EGLRuntimesV70.zip** ファイルを適切なディレクトリーにダウンロードする。
- c. **EGLRuntimesV70.zip** を unzip して、以下のファイルがあることを確認する。

プラットフォーム固有のスタック・ライブラリーの場合

- AIX 32 ビット: **EGLRuntimes/Aix/bin/libstack.so**
- AIX 64 ビット: **EGLRuntimes/Aix64/bin/libstack.so**
- Linux: **EGLRuntimes/Linux/bin/libstack.so**
- Win32:
  - **EGLRuntimes/Win32/bin/stack.dll**
  - **EGLRuntimes/Win32/bin/stack.lib**
- Solaris 32 ビット: **EGLRuntimes/Solaris/bin/libstack.so**
- Solaris 64 ビット: **EGLRuntimes/Solaris64/bin/libstack.so**

- HPUX 32 ビット: **EGLRuntimes/HPUX/bin/libstack.sl**
- HPUX 64 ビット: **EGLRuntimes/HPUX64/bin/libstack.sl**

プラットフォーム固有のアプリケーション・オブジェクト・ファイルの場合

- AIX: **EGLRuntimes/Aix/bin/application.o**
- AIX 64 ビット: **EGLRuntimes/Aix64/bin/application.o**
- Linux: **EGLRuntimes/Linux/bin/application.o**
- Win32: **EGLRuntimes/Win32/bin/application.obj**
- Solaris 32 ビット: **EGLRuntimes/Solaris/bin/application.o**
- Solaris 64 ビット: **EGLRuntimes/Solaris64/bin/application.o**
- HPUX 32 ビット: **EGLRuntimes/HPUX/bin/application.o**
- HPUX 64 ビット: **EGLRuntimes/HPUX64/bin/application.o**

### アプリケーション・レベルの共用ライブラリーを作成する手順

1. 次の 2 つの成果物をアプリケーション・レベルの共用ライブラリーにコンパイルし、スタック・ライブラリーおよび変換前の間に識別または作成したライブラリーにリンクします。

- 変換ユーティリティーによって作成される関数テーブル
- アプリケーション・オブジェクト・ファイル

2. 次のサンプルを使用して新規の共用ライブラリーをコンパイルします。ここで **NativeFuncTab.c** は関数テーブルの名前、**lib1/lib2** などの変換前の間に識別または作成された C ライブラリーの名前、**lib\_dir1/lib\_dir2** などこれらのライブラリーそれぞれのディレクトリーのロケーションです。

- AIX の場合:

```
cc -c NativeFuncTab.c
ld -G -b32 -bexpall -bnoentry -brtl NativeFuncTab.o application.o
-Lstack_lib_dir -lstack -Llib_dir1 -llib1 -Llib_dir2 -llib2 ... .. -o
app_lib_name -lc
```

- Linux の場合:

```
cc -c NativeFuncTab.c
gcc -shared NativeFuncTab.o application.o -Lstack_lib_dir -lstack
-Llib_dir1 -llib1 -Llib_dir2 -llib2 ... .. -o app_lib_name
```

- Windows の場合:

```
cl /c NativeFuncTab.c
link /DLL NativeFuncTab.obj application.obj /LIBPATH:stack_lib_dir
/DEFAULTLIB:stack.lib /LIBPATH:lib_dir1 /DEFAULTLIB:lib1.lib
/LIBPATH:lib_dir2 /DEFAULTLIB:lib2.lib ... .. /OUT:app_lib_name
```

EGL アプリケーションの実行中に、このアプリケーション・レベルの共用ライブラリーは **vgj.defaultI4GLNativeLibrary** Java ランタイム・プロパティーを使用して指定されます。

---

## プロパティー・ファイル

変換済み EGL プロジェクトのプロパティーは、それぞれが **.properties** ファイル拡張子を持つ、3 つのファイルのいずれかに定義されます。必要に応じて I4GL 環境変数を以下のいずれかのファイルに入力する必要があります。

- ホーム・ディレクトリーにある、**user.properties**。このファイルは、**user** や **password** のようなユーザー固有のデータを識別します。
- CLASSPATH にある、**programname.properties**。変換済みの I4GL プロジェクトそれぞれが、この名前のファイルを持つことができます。このファイル内のプロパティーと値は、すべてのユーザーの値を含むようにカスタマイズされます。
- CLASSPATH にある、**rununit.properties**。このファイルは、実行時に必要な情報を識別します。

いずれかのファイルで同じ名前のプロパティーが設定される場合は、ファイル **user.properties** のプロパティーが使用されます。プロパティーの検索順序は、**user.properties**、**programname.properties**、**rununit.properties** です。

I4GL 環境変数、EGL プロパティー、および JDBC プロパティーのマッピングについては、A-22 ページの『環境変数』を参照してください。

以下の番号なしリストの値は、少なくとも 1 つの **.properties** ファイルで定義する必要があります。この例の値は、次の項目を前提としています。EGL アプリケーションが **stores7** という名前のデータベースを参照していること、INFORMIXSERVER 値に基づくサーバーの切り替えがサポートされていること、パスワードがなくても **.rhosts** を介してサーバー接続を使用可能にできること、アプリケーションとデータベース・サーバーが同じ UNIX ホストで実行していないこと、です。

- **INFORMIXSERVER=myserver**
- **DEFAULT\_USER=id**
- **DEFAULT\_PASSWORD=pw**
- **stores7@myserver=jdbc:informix-sqli://host:port/  
database:INFORMIXSERVER=myserver;DBDATE=MDY4/**

アプリケーションとデータベース・サーバーが同じホストで実行している場合、**DEFAULT\_USER** または **DEFAULT\_PASSWORD** は必要ありません。

アプリケーションとデータベース・サーバーが異なる UNIX ホストで動作している場合、データベース・サーバー・ホストの **.rhosts** ファイルにアプリケーション・ホストを指定できます。 **DEFAULT\_USER** 値と **DEFAULT\_PASSWORD** 値を指定する必要はありません。

アプリケーションで **INFORMIXSERVER** の値が必要ない場合、**INFORMIXSERVER** は別の値として除外できます。アプリケーションが **CONNECT** ステートメントを介してユーザー ID とパスワード値を直接指定する場合、**DEFAULT\_USER** 値と **DEFAULT\_PASSWORD** 値は指定しなくてもかまいません。

**.properties** ファイルの詳細については、ヘルプ・トピック『*genProperties*』および『*Java ランタイム・プロパティー*』を参照してください。IBM Informix DBMS サーバーに JDBC URL を指定する際の詳細については、<http://www.ibm.com/software/data/informix/pubs/library/> から使用できる IBM Informix JDBC 資料を参照してください。

---

## 変換済み EGL ファイルの検証とコンパイル

変換済み EGL ソース・ファイルは検証してコンパイルする必要があります。これらのステップは、I4GL アプリケーションと共用ライブラリーのプロジェクトでのみ必要で、データベース・スキーマ抽出のプロジェクトでは必要ありません。このプロセスを試行する前に、システムに IBM Informix JDBC ドライバーがインストールされていなければなりません。

注: 変換ユーティリティは、各変換済みプロジェクトについて、ビルド記述子ファイルのテンプレートを生成します。このファイルは **EGLSource** ディレクトリにあり、**Projectname.eglbuild** の命名規則に従っています。

変換済み EGL ソース・ファイルを検証およびコンパイルする手順は、次のとおりです。

注: ステップ 4、5、および 6 はオプションです。

1. 「プロジェクト・エクスプローラー」ビューから、「**Projectname**」を選択して右クリックします。
2. 表示されるメニューから、「プロパティ」を選択します。
3. 「**Projectname のプロパティ (Properties for Projectname)**」ウィンドウの左のボックスで、「**EGL ビルド・パス**」を選択します。タブ付きの「**プロジェクト**」セクションで、リストされている従属プロジェクトをすべて選択します。
4. 左のボックスで、「**EGL デフォルト・ビルド記述子**」を選択します。
5. 「**EGL デフォルト・ビルド記述子**」セクションの「**ターゲット・システムのビルド記述子**」フィールドで、下矢印を選択します。これで、ワークスペースで使用可能な EGL ビルド記述子ファイルがすべて表示されます。
6. **Projectname.eglbuild** の命名規則に従うファイルである、ビルド記述子を選択します。「**OK**」を選択します。EGL 自動ビルド・プロセスが開始します。
7. プロジェクトでテスト用のデータベース接続が必要な場合は、左のボックスで、「**Java のビルド・パス**」を選択します。「**ライブラリー**」タブを選択します。
8. 「**外部 JAR の追加...**」ボタンを選択します。「**JAR の選択**」ウィンドウでファイル・システムを参照し、適切な JDBC ドライバー JAR ファイルを見つけます。「**OK**」をクリックします。選択した JAR ファイルが「**ライブラリー**」タブに表示されます。

注: EGL ビルド記述子の例については、『付録 H. EGL ビルド記述子の例』を参照してください。

---

## EGL の Java への生成

EGL ファイルを Java に生成する前に、以下を行う必要があります。

- TODO メッセージおよび FIXME メッセージをすべて識別して、解決します。詳細については、4-7 ページの『変換エラーの訂正』を参照してください。
- 検証エラーをすべて識別して、解決します。検証エラーをすべて識別するまで、EGL ファイルを Java に生成できません。
- 必要に応じて、データベース接続情報を **Projectname.eglbuild** ファイルに追加してください。

データベース接続情報を **Projectname.egl** ファイルに追加するには、以下を行います。

1. 「プロジェクト・エクスプローラー」ビューで、「**Projectname**」を右クリックします。
2. 「**EGLSource**」ディレクトリー内で、「**Projectname.egl**」ファイルをダブルクリックします。
3. タブ付きのウィンドウに「**Projectname.egl**」ファイルが開きます。「指定したオプションのみを表示」ボックスにチェック・マークを付けます。
4. 必要に応じて、既存のオプション「値」をクリックして、以下のオプションについての接続情報を入力します。
  - sqlDB
  - sqlID
  - sqlPassword
  - sqlValidationConnectionURL
5. ウィンドウ・タブを閉じて、変更を受け入れます。

EGL を Java に生成する手順は、次のとおりです。

1. 「プロジェクト・エクスプローラー」ビューで、「**Projectname**」を右クリックします。
2. メニューから「生成」を選択します。
3. 「**EGL データ・パーツの生成中**」ウィンドウが表示され、生成が完了すると閉じられます。

生成された Java プログラムをデータベース接続で実行するには、4-13 ページの『プロパティ・ファイル』で説明されているように、**.property** ファイルを作成する必要があります。

Java ソース・ファイルは、プロジェクト・ワークスペースの **JavaSource** フォルダにあります。EGL ファイルの生成についての詳細は、ヘルプ・トピック『EGL 出力の生成、準備、および実行』を参照してください。

---

## エラー・メッセージ変換の理解

変換ユーティリティは、I4GL エラー・メッセージを Java プロパティ・ファイルに変換して、変換済みのメッセージ・ファイルをすべてディレクトリー **EGLDestinationDirectory/MessageSource/locale/** に配置します。

I4GL エラー・メッセージ・ファイルと、結果として生じる Java プロパティのフォーマットの比較を、以下の表 4-3 に示します。

表 4-3. I4GL メッセージと Java プロパティのファイル・フォーマットの比較

I4GL メッセージ・ファイル・フォーマット	EGL メッセージ・ファイル・フォーマット
. message-number message-text	message-number=message-text

I4GL メッセージ・ファイルは Java **.properties** ファイルに変換され、ロケール名を使用してサブディレクトリに編成されます。例えば、**MyMessage.msg** という名前の I4GL メッセージ・ファイルは以下のように変換されます。

英語の場合:

**EGLDestinationDirectory/MessageSource/en\_us/8859-1/MyMessage.properties**

中国語の場合:

**EGLDestinationDirectory/MessageSource/zh\_tw/big5/MyMessage\_zh.properties**

---

## レポート変換の理解

EGL レポートは、JasperReports の機能である、Java で作成されるオープン・ソースのレポート・ライブラリーを使用します。このため、EGL への変換中に、一部の I4GL コード、関数、およびファイルが変更されます。

I4GL アプリケーションの変換中に、I4GL レポート・コードに以下のような変更が起きました。

- I4GL プログラム・ファイルが EGL プログラム・ファイルに変換されました。例えば、**myreport.4gl** という I4GL ファイルは **myreport.egl** という EGL ファイルになります。
- 単一の I4GL REPORT 関数が 4 つの EGL 関数に変換されました。例えば、**myreport.4gl** という名前で **REPORT** 関数 **REPORT my4glReport(a, b, c)** を持つ I4GL レポート・ファイルは、**myreport.egl** という名前で以下の関数呼び出しを持つ EGL レポート・ファイルに変換されました。

- **my4glReport\_START( )**
- **my4glReport\_OUTPUT(a, b, c)**
- **my4glReport\_FINISH( )**
- **my4glReport\_TERMINATE( )**

この 4 つの EGL 関数を一緒に使用すると、EGL で単一のレポート・ドライバ関数のように機能します。

- I4GL レポートのビジネス・ロジックは、変換されて、EGL レポート・ハンドラー・ファイルに移動しました。レポート・ハンドラーとは、レポートを埋める際に起こるイベントを処理するロジックを提供する、EGL レポートのコンポーネントです。

I4GL レポートの名前が **my4glreport** であった場合、EGL レポート・ハンドラー・ファイルの名前は **my4glreport\_handler.egl** になります。

- I4GL レポートのプレゼンテーション・ロジックは、変換されて、JasperReports XML 設計ファイルに移動します。この変換済みファイルには、I4GL レポートから変換されたレポート・レイアウトのテンプレートが含まれます。

I4GL レポートの名前が **my4glreport** である場合、JasperReports XML ファイルの名前は **my4glreport\_XML.jrxml** になります。I4gl レポートに、出力ロジックを含む複数のループ構成体が含まれる場合、各ループ構成体ブロックは後続のサ

ブレポートに移動します。後続の各レポートのファイル名は、**SUB** という語とシーケンス番号で識別されます。例えば、最初の後続レポートの名前は **my4glreport\_XML\_SUB1.jrxml** です。

## EGL レポート・ドライバー関数

上述したように、I4GL から EGL への変換中に、単一の I4GL REPORT 関数が 4 つの EGL 関数で置き換えられます。各 EGL 関数の I4GL 構文へのマップについて、以下の表 4-4 に示します。

表 4-4. EGL レポート・ドライバーの振る舞い

EGL 関数	対応する I4GL ステートメント	EGL での振る舞い
<i>reportname</i> _START( )	START REPORT	変換済みの EGL プログラムの実行中に、この関数は EGL アプリケーションのデフォルトまたはアクティブなデータベースに一時テーブルを作成します。 I4GL レポート関数によって受け取られるパラメーターは、一時テーブルで列として追加されます。さらに、配列変数を除いて、I4GL レポートが使用するグローバル変数がすべてこの一時テーブルに列として追加されます。
<i>reportname</i> _OUTPUT( ) <i>reportname</i> _OUTPUT(a, b, c)	OUTPUT TO REPORT	この関数は、渡されるすべての引数と、I4GL レポート関数がレポートで使用するその他のグローバル変数をすべて収集し、それらの現行値を行として一時テーブルに追加します。一般に、すべてのレポート・データを受け渡すには多数の呼び出しを行う必要があります。 <b>注:</b> I4GL レポートの変換で引数を受け取らない場合や、(配列変数以外の) グローバル変数を使用しない場合は、この関数は一時テーブルにグローバル変数のためのプレースホルダー行を挿入します。

表 4-4. EGL レポート・ドライバーの振る舞い (続き)

EGL 関数	対応する I4GL ステートメント	EGL での振る舞い
<b>reportname_FINISH( )</b>	FINISH REPORT	<p>この関数は以下のことを行います。</p> <ul style="list-style-type: none"> <li>• レポート・ライブラリーを初期化します。</li> <li>• 一時テーブルを参照する SELECT ステートメントを EGL レポート・ライブラリーに受け渡します。生成される結果セットは JasperReports エンジンのデータ・ソースとして使用されます。</li> <li>• EGL API を使用して JasperReports を埋めます。</li> <li>• EGL JasperReports テキスト・エクスポーターを使用して、JasperReports をテキスト・フォーマットにエクスポートします。</li> </ul> <p>レポートが生成された後、この関数はレポート・データの保管に使用された一時テーブルをドロップします。</p>
<b>reportname_TERMINATE( )</b>	TERMINATE REPORT	<p>この関数は一時テーブルをドロップし、レポートの処理を停止します。</p>

注: レポートを処理する際、生成済みの EGL レポート・ドライバー関数は常に、アクティブなデータベースが使用可能であることを前提としています。アクティブなデータベースを使用できない場合は、レポート・ドライバー関数のコードを変更して、レコードの EGL 動的配列を使用してデータを集め、JasperReports エンジンに受け渡す必要があります。

グローバル配列変数は一時テーブルに追加されないため、EGL に変換される際に、グローバル配列変数を使用する I4GL レポート・コードが期待される結果を出さないことがあります。期待される結果を達成するには、EGL レポートを変更してください。

I4GL レポートは 1 行ずつ処理されましたが、EGL ではレポート・データはすべて一時テーブルに収集されてから処理されました。レポート・データが処理されるのは、データがすべて一時テーブルに収集されていて、**reportname\_FINISH( )** が呼び出される場合のみです。したがって、レポート内のデータと I4GL レポート・ドライバー関数に依存関係がある場合は、正しい結果を出さないレポートもあります。

## I4GL レポート・セクション

このセクションでは、以下の 4 つの I4GL レポート・セクションの EGL への変換について説明します。

- DEFINE
- OUTPUT
- ORDER BY

- FORMAT

## I4GL DEFINE セクション

I4GL では、DEFINE セクションは、REPORT プロトタイプ内の仮引数ごとのデータ型と、REPORT プログラム・ブロック内からのみ参照できる任意の追加のローカル変数のデータ型を宣言します。

**パラメーター変換:** DEFINE セクションで宣言されるパラメーターはすべて、レポート・フィールドとして XML 設計文書に変換されてから、EGL レポート・ハンドラーで変数として宣言されます。さらに、パラメーターは一時テーブルの列としてレポート・ドライバー関数の一部に変換されるため、OUTPUT TO レポート呼び出しを反復するたびに、その値を一時テーブルに保管できます。一時テーブルに保管される値は、レポートのデータ・ソースとして使用されます。

変換中に、I4GL レコードの型パラメーターは、JasperReports フィールドで単一レベルにフラット化されます。以下の例は、元の I4GL コードです。

```
Define rec Record
    i, j Integer,
    k, l Integer,
    si Smallint,
End Record
```

この例は、XML 設計文書内の、フラット化された同等のコードです。

```
rec_i
rec_j
rec_k
rec_l
rec_si
```

変換ユーティリティは、I4GL レコード・パラメーターの名前が固有であること、つまり、別のフラット化されたか宣言された JasperReports フィールドまたは変数の名前が同じではないことを、検証しようとしません。変換後に、2 つのフラット化されたか宣言された JasperReports フィールドまたは変数の名前が同じであることを識別した場合は、いずれかの名前を変更して、それぞれの名前を固有にしてください。

EGL レポート・ハンドラーに変換される際、I4GL レポートの DEFINE セクションのパラメーターは、通常の変数として宣言されます。EGL レポート・ハンドラーは、内部関数 **init** を使用してビジネス・ロジックを処理し、レポート・フィールド値を使用してロケール変数を初期化します。I4GL ではレポート・フィールドや受け取ったパラメーターに値を割り当てることが可能でしたが、EGL ではできません。式が矛盾していると、誤った結果のレポートとなることがあります。期待した結果のレポートを得るには、変換済みコードを手動で変更してください。

変換ユーティリティは、I4GL 変数を、EGL レポート・ハンドラー変数および一時テーブルの列に変換します。変換ユーティリティは、レポートのステートメントと式で使用する各変数を検討し、グローバル変数と非配列変数をレポート・パラメーターと同様に処理します。変換ユーティリティは、EGL が一時テーブルから生成されたデータ・ソースを処理する際に、受け取った値が受け取った I4GL 値と同じになるように、反復のたびに各変数の値を保持しようとします。ただし、値を渡すこのメソッドは、必ずしも I4GL プログラム・ロジックに準拠しているわ

けではなく、誤った結果が起こることがあります。期待した結果のレポートを得るには、変換済みコードを手動で変更してください。

注: 1 つの例外を除いて、I4GL レポートが使用するグローバル配列変数の性質はグローバルのままです。I4GL コードにグローバル配列変数があり、その値が OUTPUT TO レポートのステートメントが反復すると変更される場合は、その変数はグローバル変数とは見なされず、変換されません。期待した結果のレポートを得るには、変換済みコードを手動で変更してください。

**ローカル変数の変換:** DEFINE セクションのローカル変数はすべて EGL レポート・ハンドラー変数に変換されます。

## I4GL OUTPUT セクション

I4GL では、OUTPUT セクションが、レポートやプリンターのページ替えのシーケンスからの出力用の宛先および次元を指定します。

元の I4GL OUTPUT セクション文節と、XML 設計文書への変換方法について、以下の表 4-5 に示します。

表 4-5. I4GL 出力文節の変換

I4GL 出力文節	変換先	変換の特性
PAGE LENGTH	XML 設計文書	<ul style="list-style-type: none"><li>変換中に PAGE LENGTH 文節が存在する場合は、XML 設計文書 <b>pageHeight</b> ページ属性が以下のように設定されます。 <math display="block">\text{pageHeight} = (\text{Font height for the locale}) * (\text{value of PAGE LENGTH clause})</math></li><li>変換中に PAGE LENGTH 文節が存在しない場合は、PAGE LENGTH 文節のデフォルト値が、I4GL のデフォルト・ページ・サイズである 66 になります。</li><li><b>pageHeight</b> 値を設定する前に、変換ユーティリティーは、すべてのバンドの高さと <b>topMargin</b> と <b>bottomMargin</b> の合計が、<b>pageHeight</b> 値内に収まることを検証します。必要に応じて、変換ユーティリティーは <b>pageHeight</b> を調整して、正しく収まるようにします。</li></ul>
BOTTOM MARGIN	XML 設計文書	<ul style="list-style-type: none"><li>変換中に BOTTOM MARGIN 文節が存在する場合は、XML 設計文書 <b>bottomMargin</b> ページ属性が以下のように設定されます。 <math display="block">\text{bottomMargin} = (\text{Font height for the locale}) * (\text{value of BOTTOM MARGIN})</math></li><li>変換中に BOTTOM MARGIN 文節が存在しない場合は、<b>bottomMargin</b> ページ属性が以下のように設定されます。 <math display="block">\text{bottomMargin} = (\text{Font height for the locale}) * 3 \text{ (default I4GL value for BOTTOM MARGIN)}</math></li></ul>

表 4-5. I4GL 出力文節の変換 (続き)

I4GL 出力文節	変換先	変換の特性
LEFT MARGIN	XML 設計文書	<ul style="list-style-type: none"> <li>変換中に LEFT MARGIN 文節が存在する場合は、JasperReports XML 設計 <b>leftMargin</b> ページ属性が以下のように設定されます。  <math>\text{leftMargin} = (\text{Font width for the locale}) * (\text{value of LEFT MARGIN})</math></li> <li>変換中に LEFT MARGIN 文節が存在しない場合は、<b>leftMargin</b> ページ属性が以下のように設定されます。  <math>\text{leftMargin} = (\text{Font width for the locale}) * 5</math> (default I4GL value for LEFT MARGIN)</li> </ul>
RIGHT MARGIN	XML 設計文書	<ul style="list-style-type: none"> <li>変換中に RIGHT MARGIN 文節が存在する場合は、XML 設計文書 <b>pageWidth</b> 属性が以下のように設定されます。  <math>\text{pageWidth} = (\text{Font width for the locale}) * (\text{value of right margin})</math></li> <li>変換中に RIGHT MARGIN 文節が存在しない場合は、<b>pageWidth</b> 属性が以下のように設定されます。  <math>\text{pageWidth} = (\text{Font width for the locale}) * 132</math> (default I4GL value for RIGHT MARGIN)</li> </ul>
TOP MARGIN	XML 設計文書	<ul style="list-style-type: none"> <li>変換中に TOP MARGIN 文節が存在する場合は、XML 設計文書 <b>topMargin</b> 属性が以下のように設定されます。  <math>\text{topMargin} = (\text{Font height for the locale}) * (\text{value of TOP MARGIN})</math></li> <li>変換中に TOP MARGIN 文節が存在しない場合は、<b>topMargin</b> 属性が以下のように設定されます。  <math>\text{topMargin} = (\text{Font height for the locale}) * 3</math> (default I4GL value for TOP MARGIN)</li> </ul>
REPORT TO	直接変換しません	<ul style="list-style-type: none"> <li>REPORT TO FILE <i>filename</i> または REPORT TO <i>filename</i> 文節が I4GL レポートに存在する場合、変換ユーティリティーは、エクスポートされたテキスト出力と指定された <i>filename</i> を保管します。</li> </ul>
TOP OF PAGE	直接変換しません	EGL では、EGL レポート・テキスト・エクスポーターが使用する文字。

## I4GL ORDER BY セクション

I4GL では、ORDER BY セクションは入力レコードのソート・リストを指定し、グループが評価される順序を識別します。ORDER BY セクションは、XML 設計文書 (拡張子 **.jrxml** 付き) および EGL レポート・ドライバー関数に変換されます。

ORDER BY を EGL に変換すると、以下のようになります。

- XML 設計文書で、その順序に従うグループ・セクションが、 I4GL ORDER BY セクションに設定されます。
- ORDER EXTERNAL BY が I4GL レポートで使用されないと、一時テーブルは昇順または降順の列の順序を指定されたままにし、レコードはそのソート順でテーブルから検索されます。

### I4GL FORMAT セクション

I4GL は 2 つのタイプの FORMAT セクションをサポートします。両方ともレポートの外観から、出力の外観を決定します。最も単純なレポートには、FORMAT キーワードと END REPORT キーワードの間に EVERY ROW キーワードしか含まれません。より複雑な I4GL レポートには、レポートの処理中に実行されるステートメントを含む、(ON EVERY ROW や BEFORE GROUP OF のような) 制御ブロックを含む FORMAT セクションがあります。このセクションでは、単純な I4GL レポートと複雑な I4GL レポート両方についてと、そのコンポーネントの EGL および JasperReports への変換方法について説明します。

**単純なレポート:** I4GL では、FORMAT EVERY ROW ステートメントが、I4GL レポート関数に渡されるすべてのデータのデフォルトのレポートを作成します。このステートメントではその他のフォーマットのブロック・ステートメントは許可されていません。

単純な I4GL レポートを EGL の同等のレポートに変換する際に、変換ユーティリティーはレポート・フィールド名のサイズとフィールド値のサイズを比較し、大きいサイズを列幅に使用します。行幅の合計が、XML 設計文書内のタグの **columnWidth** 属性より小さい場合は、変換ユーティリティーが XML 設計文書に以下を追加します。

- 1 つのページ・グループ・ヘッダー・セクション
- 1 つの詳細セクション

ヘッダー・セクションでは、フィールド名がすべて 1 行に表示されます。フィールド名間の間隔は、上記で決定された列幅に従って決定されます。以下の表は、ヘッダー・セクションの例です。

表 4-6. ヘッダー・セクションの例

FieldName1	FieldName2	FieldName3
Value1	LongValue2	Value3
Value1b	LongValue2b	Value3b

行幅の合計が、XML 設計文書内の **columnWidth** 属性より大きい場合は、変換ユーティリティーが XML 設計文書に 1 つの詳細セクションを作成します。詳細セクションは、1 行に 1 つずつレポート・フィールドを表示します。フィールド内では各フィールド名の後にデータが入ります。以下の表は、詳細セクションの例です。

表 4-7. 詳細セクションの例

FieldName1	Value1
FieldName2	LongValue2
FieldName3	Value3

表 4-7. 詳細セクションの例 (続き)

FieldName1	Value1b
FieldName2	LongValue2b
FieldName3	Value3b

単純な I4GL レポートのデータ値は、以下の 3 つの EGL レポート・ハンドラー関数の文字列値のいずれかに変換されます。

- **beforeDetailEvals。** この JasperReports デフォルト・メソッドは、詳細バンド要素の処理よりも前に呼び出されます。
- **init。** このメソッドは、レポート・フィールド値をすべてローカル変数にコピーします。
- **getPrintString。** このメソッドは XML 設計文書によって使用され、印刷ストリングの配列からレポート・フィールド値を取り出します。

**複雑なレポート:** I4GL では、レポートの特定の部分が処理される際に実行される 1 つ以上のステートメントを指定することによって、制御ブロックがレポートの構造を定義します。出力としてレポートに送信されるデータ・レコードがないと、これらのブロックのステートメントはまったく実行されません。基本的に、複雑な I4GL レポートには FORMAT セクション内に複数のサブセクションがあります。

I4GL レポートでは中央のロケーションにビジネス・ロジックとプレゼンテーション・ロジックが含まれますが、EGL レポートではビジネス・ロジックとプレゼンテーション・ロジックは分かれています。変換ユーティリティは、I4GL ビジネス・ロジックを EGL レポート・ハンドラー文書に、I4GL プレゼンテーション・ロジックを XML 設計文書に変換します。

I4GL FORMAT サブセクションのデータ・プレゼンテーション・ステートメントはすべて、対応する JasperReports バンド、汎用レポート・セクションに変換または移動されます。元の I4GL FORMAT サブセクションのデータ・プレゼンテーション・ステートメントと変換されたバンド間のマッピングについて、以下の表 4-8 に示します。

表 4-8. I4GL データ・プレゼンテーション・ステートメントと JasperReports バンドのマッピング

データ・プレゼンテーション・ステートメント	バンド
First Page Header	title
Page Header	pageHeader
Before Group Of <i>variable</i>	groupHeader (指定された <i>variable</i> のグループ・セクションのもの)
On Every Row	detail
After Group Of <i>variable</i>	groupFooter (指定された <i>variable</i> のグループ・セクションのもの)
Page Trailer	pageFooter
On Last Row	要約

元の I4GL FORMAT サブセクションのデータと EGL レポート・ハンドラー・メソッド間のマッピングについて、以下の 表 4-9 に示します。

**注: JasperReports** メソッド列見出しで、メソッドがデフォルトのレポート・イベント処理メソッドであるかどうかも識別します。

表 4-9. I4GL FORMAT サブセクションの EGL と JasperReports へのマッピング

I4GL レポートの FORMAT サブセクション	EGL レポート・ハンドラー・メソッド	JasperReports メソッド
First Page Header	firstPageHeader	なし
Page Header	pageHeader	なし
Before Group Of <i>variable</i>	beforeGroupOf <i>variable</i>	なし
On Every Row	beforeDetailEval	あり
After Group Of <i>variable</i>	afterGroupOf	なし
Page Trailer	pageTrailer	なし
On Last Row	onLastRow	なし

上記の表でリストされたメソッドに加えて、EGL レポート・ハンドラーではさらに 3 つのメソッドが定義されます。

- **init**。 レポート・ハンドラー・メソッドは、他のメソッドから呼び出されて、レポート・フィールド値を使用して、ローカル・レポート・ハンドラー変数を初期化します。 さらにこのメソッドは、指定されたバンドに関連付けられた印刷フラグおよび印刷ストリングを初期化します。
- **getPrintString**。 このメソッドは XML 設計文書から呼び出され、1 行テキスト領域に表示される値を戻します。
- **getPrintFlag**。 このメソッドは XML 設計文書から呼び出され、静的テキストまたは 1 行テキスト領域の印刷状況を判別します。このメソッドは値 1 (true) または 0 (false) を戻します。

以下の I4GL レポート・ステートメントは EGL や JasperReports に変換されませんが、構文的に正しいコードを維持するために、変換ユーティリティーがこれらのステートメントのレポート・ハンドラー内に冗長コードを生成します。

- EXIT REPORT
- NEED
- PAUSE

以下の I4GL 要素は条件付きで EGL に変換されます。

- **SKIP TO TOP OF PAGE**。 このステートメントは、FORMAT セクションの BEFORE GROUP OF サブセクション内で出現する場合にのみ、変換されます。**SKIP TO TOP OF PAGE** がその他のインスタンスで出現する場合は、ステートメントは変換されず、結果としてレポートが誤ったページにデータを作成することがあります。
- **LINENO**。 EGL は、I4GL LINENO 演算子の変換をサポートしません。ただし、変換ユーティリティーは、ビジネス・ロジックを介して実行される print ス

ステートメントごとに、ローカル・レポート・ハンドラー変数を増やすことで  
LINENO 演算をシミュレートする、EGL レポート・ハンドラー・コードを生成し  
ます。

注: 上記にリストした要素を含むレポートを変換した場合は、変換されたコードを  
検討して、必要に応じて訂正してください。

**I4GL PRINT ステートメント:** I4GL では、PRINT ステートメントはレポート定  
義から出力を作成します。変換ユーティリティーはすべての I4GL レポートの  
PRINT ステートメントからデータを分析および収集し、ほとんどのレポートのプレ  
ゼンテーション・レイアウトを XML 設計文書に変換します。変換ユーティリティー  
は、以下の PRINT ステートメント・データを分析および収集します。

1. レポート内の PRINT ステートメントの数。
2. PRINT ステートメントを含むループ構成体の数と、各ループ構成体内の PRINT  
ステートメントの数。
3. PRINT ステートメント内の式の数と、各式の属性。
4. 各 PRINT ステートメントのサイズ。サイズは、PRINT ステートメントの式に  
よって戻されるすべてのデータのサイズを加算して計算されます。PRINT ステ  
ートメントのサイズが **columnWidth** より大きい場合は、**pageWidth** と  
**columnWidth** を調整して、すべての印刷フィールドがページ内に入るようにし  
ます。

上記の情報が収集されたら、変換ユーティリティーは EGL レポート・ハンドラー  
または XML 設計文書について、以下を定義します。

1. 各 I4GL FORMAT サブセクションの設計バンドの高さ。
2. 必要な EGL サブレポートの数、および各サブレポートの静的テキストおよび 1  
行テキスト領域の構造。
3. レコード配列の数、および EGL レポート・ハンドラーにサブレポート・データ  
を保持するために必要な構造。
4. EGL レポート・ハンドラーで必要な印刷フラグと印刷ストリングの数。
5. テキスト要素の数と、静的テキストまたは 1 行テキスト領域に合うかどうか。
6. 指定された 1 行テキスト領域が、右揃えであるため数値フィールドであるの  
か、左揃えであるため非数値フィールドであるのか。

ループ内で出現しない各 PRINT ステートメントについて、変換ユーティリティー  
は以下を行います。

1. 静的テキストまたは 1 行テキスト領域の **.xml** タグについて座標を調整し、  
XML 設計文書内で正しい座標を生成します。
2. **egl4gl\_printFlag** 内の 1 つの配列指標を、指定された PRINT ステートメントと  
関連付けます。
3. XML 設計文書でコードを生成し、XML 設計文書のすべての静的テキストまた  
は 1 行テキスト領域の **printWhenExpression** タグについて EGL レポート・ハ  
ンドラー・プログラムから戻される、関連する **egl4gl\_printFlag** 値を取得して、  
その要素をレポートに表示する必要があるかどうかを決定します。
4. 関連する **egl4gl\_printflag** について適切な値を設定する、EGL レポート・ハン  
ドラー・コードを生成します。

5. **egl4gl\_printString** 内の 1 つの配列指標を、**PRINT** ステートメント内の各式と関連付けます。
6. XML 設計文書内の 1 行テキスト領域にコードを追加して、**egl4gl\_printString** の指定された配列指標について、EGL レポート・ハンドラーからのレポートに配置される文字列値を戻します。
7. EGL レポート・ハンドラーにコードを生成して、**egl4gl\_printString** 配列指標に、**PRINT** 属性を考慮して出力文字列値を取り込みます。
8. **egl4gl\_lineNumber** 変数を 1 つ増加します。この増加は **I4GL LINENO** 演算子と同等です。

**I4GL PRINT ステートメントの式:** **I4GL PRINT** ステートメントの式は、印刷可能文字として表示できる 1 つ以上の値を戻します。変換ユーティリティーは各式を分析して、XML 設計文書内の **staticText** フィールドか 1 つ以上の 1 行テキスト領域に変換します。さらにこれらの **print** 式は、XML 設計文書内の **staticText** フィールドか 1 行テキスト領域の、垂直方向の配置およびサイズまたは幅を決定するために使用されます。幅は以下のパラメーターによって決定されます。

- **print** 式の属性。
- **PRINT** 式を作成する変数のデータ・サイズ。

**注:** **I4GL SPACE** または **SPACES AND COLUMN** 演算子で整数リテラルではなく整数式を使用する場合は、変換ユーティリティーが式の値をデフォルトの 1 に設定します。正しいレポート・レイアウトを得るには、レポート設計およびコードを調整してください。

**ループ構成体の I4GL PRINT ステートメント:** 他の **PRINT** ステートメントと同様に、ループ構成体の **PRINT** ステートメントも、同じ数の **print** 式に変換され、それぞれが XML 設計文書サブレポートに追加されます。EGL レポート・ハンドラーでは、1 つのレコード配列が、出力のために生成されるデータをすべて累積するよう宣言されます。ループ構成体で使用する各 **PRINT** ステートメントについて、変換ユーティリティーは以下を行います。

1. **PRINT** ステートメントの式をすべて識別します。
2. ループのネスト構造を識別します。
3. **PRINT** ステートメントのネスト構造、印刷ストリング、および印刷フラグと一致する、レコード配列構造を宣言します。
4. EGL レポート・ハンドラーにコードを生成してレコード配列を宣言し、レコード配列フィールドに印刷ストリングおよび印刷フラグ・データを取り込みます。
5. XML 設計文書にサブレポート・タグを追加します。
6. 別の XML 設計文書を生成し、詳細セクションにレコード配列のすべてのフィールドを含めます。この詳細セクションは、ループ構造で識別された **PRINT** ステートメントのテンプレートを表示します。

**注:** 変換ユーティリティーは **I4GL LABEL** および **GOTO** ループ構造ステートメントを、1 つのループ構造に変換しようとします。ただし、**LABEL** および **GOTO** ループ構造の識別と変換は常に成功するわけではないので、正しい結果を得るには、レポート設計およびコードを調整してください。

**セミコロンで終了する I4GL PRINT ステートメント:** セミコロンで終了する I4GL PRINT ステートメントは、次の PRINT ステートメントが同じ行で継続することを示します。変換ユーティリティーは、セミコロンで終了する各 PRINT ステートメントを識別し、同様のレイアウトの XML 設計文書を作成します。ただし、セミコロンで終了する PRINT ステートメントは、以下の条件が起こる場合は成功しません。

1. 次の PRINT ステートメントの前に IF 条件ブロックが実装される場合。
2. PRINT ステートメントの後にループ構成体が続く場合。
3. PRINT ステートメントが I4GL FORMAT セクションの最後の PRINT ステートメントである場合。

**I4GL レポート演算子:** I4GL レポート演算子の変換についての詳細を、以下の表 4-10 に示します。

表 4-10. I4GL レポート演算子の変換

I4GL レポート演算子	変換の特性
CLIPPED	XML 設計文書で、PRINT ステートメントの式の後に、CLIPPED 属性を持つ PRINT ステートメントの式が続く場合、2 番目の PRINT ステートメントは最初のステートメントと同じ 1 行テキスト領域に配置されます。変換ユーティリティーが、COLUMN print 式または print ステートメントの終わりを識別すると、1 つの 1 行テキスト領域への 2 つの print ステートメントの配置は終了します。PRINT ステートメントが配置される 1 行テキスト領域のサイズは、1 行テキスト領域に配置されるすべての PRINT フィールドの合計です。
WORDWRAP	WORDWRAP を変換するために、変換ユーティリティーは XML 設計文書の textField タグ <b>isStretchWithOverflow</b> 属性を <b>true</b> に設定します。
USING	EGL レポート・ハンドラー・プログラムで出力 <b>egl4gl_printString</b> を作成する際、EGL フォーマット関数が、I4GL USING 属性と同様にフォーマット設定される、戻りストリングに呼び出されます。
SPACE または SPACES	変換ユーティリティーは SPACE または SPACES 演算子を引用符で囲まれた空のストリングとして分析し、これらの演算子を Jasper XML 設計文書の staticText フィールドまたは textField に変換します。SPACES の数が非整数リテラルによって定義される場合は、変換ユーティリティーは SPACE を 1 と想定します。
PAGENO	変換ユーティリティーは、PAGENO 演算子を JasperReport <b>PAGE_NUMBER</b> 変数に変換します。EGL レポート・ハンドラーは <b>PAGE_NUMBER</b> 変数にアクセスするメソッドを維持します。
LINENO	I4GL LINENO 演算子については、直接の変換マッピングはありません。代わりに、変換済みの I4GL レポートは、ユーザー定義変数 <b>egl4gl_lineNumber</b> を使用して、LINENO をシミュレートします。 <b>注:</b> LINENO について正しい値を維持できないので、期待される結果のレポートを得るには、レポート・コードを調整してください。
FILE	この演算子は EGL に変換されません。FILE によって生成される結果を得るには、変換済みのコードを手動で訂正してください。

**I4GL 集約レポート関数:** I4GL では、**SUM**、**MAX**、**MIN**、**COUNT**、**PERCENT**、および **AVG** の集約レポート関数が、1 つのレポートの複数のレコードからデータを要約するために使用されました。I4GL レポートの変換中に、集約レポート関数の使用法がそれぞれ、1 つ以上の XML 設計レポート変数と、1 つ以上のレポート・ハンドラー関数に変換されます。レポート・ハンドラー関数で計算の処理が可能になるように、XML 設計レポート変数の計算タイプは **SYSTEM** に設定されます。

このセクションでは、各 I4GL 集約レポート関数の EGL への変換について記載します。以下の I4GL ステートメントのすべての例で、*variable* は有効な I4GL レポート識別子です。

**SUM:** I4GL **SUM** 集約関数は、1 つの JasperReports XML 設計変数と 1 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである `PRINT SUM(variable)`

は、EGL では以下の 2 つの要素に変換されます。

1. JasperReports XML 設計変数

`variable_SUM_number`

2. JasperReports XML 設計変数を計算する、EGL レポート・ハンドラー関数

`Function update variable_SUM_number`

上記の EGL レポート・ハンドラー関数で、*number* は I4GL **SUM** 集約関数の各オカレンスを区別する通し番号で、固有の変数名を作成するために使用されます。

**注:** レコードのメンバーである場合は、変数名が変わります。

変換中に、`getReportVariableValue()` レポート・ライブラリー API を使用してレポート設計変数を呼び出すために、コードが生成されます。したがって、上記の **PRINT SUM(variable)** ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] = getReportVariableValue("<variable>_SUM_<number>");
```

**MAX:** I4GL **MAX** 集約関数は、1 つの XML 設計変数と 1 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである

`PRINT MAX(variable)`

は、EGL では以下の 2 つの要素に変換されます。

1. XML 設計変数

`variable_MAX_number`

2. 変数の最大値を識別する、EGL レポート・ハンドラー関数

`Function update variable_MAX_number`

上記の EGL レポート・ハンドラー関数で、*number* は I4GL **MAX** 集約関数の各オカレンスを区別する通し番号で、固有の変数名を作成するために使用されます。

**注:** レコードのメンバーである場合は、変数名が変わります。

変換中に、**getReportVariableValue()** レポート・ライブラリー API を使用して XML 設計レポート変数を呼び出すために、コードが生成されます。したがって、上記の **I4GL PRINT MAX(variable)** ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] = getReportVariableValue("variable_MAX_number");
```

**MIN:** I4GL **MIN** 集約関数は、1 つの XML 設計変数と 1 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである

```
PRINT MIN(variable)
```

は、EGL では以下の 2 つの要素に変換されます。

1. JasperReports XML 設計変数
  - *variable\_MIN\_number*
2. JasperReports 変数の最小値を識別する、EGL レポート・ハンドラー関数
  - Function update *variable\_MIN\_number*

上記の EGL レポート・ハンドラー関数で、*number* は I4GL **MIN** 集約関数の各オカレンスを区別する通し番号で、固有の変数名を作成するために使用されます。

**注:** レコードのメンバーである場合は、変数名が変わります。

変換中に、**getReportVariableValue()** レポート・ライブラリー API を使用して JasperReports XML 設計レポート変数を呼び出すために、コードが生成されます。したがって、上記の **I4GL PRINT MIN(variable)** ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] = getReportVariableValue("variable_MIN_number");
```

**COUNT:** I4GL **COUNT** 集約関数は、1 つの JasperReports XML 設計変数と 1 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである

```
PRINT COUNT(*)
```

は、EGL では以下の 2 つの要素に変換されます。

1. JasperReports XML 設計変数
  - *COUNT\_number*
2. JasperReports 変数の最大値を識別する、EGL レポート・ハンドラー関数
  - Function update *variable\_COUNT\_number*

上記の EGL レポート・ハンドラー関数で、*number* は I4GL **COUNT** 集約関数の各オカレンスを区別する通し番号で、固有の *variable* 名を作成するために使用されます。

**注:** レコードのメンバーである場合は、変数名が変わります。

変換中に、**getReportVariableValue()** レポート・ライブラリー API を使用して XML 設計レポート変数を呼び出すために、コードが生成されます。したがって、上記の **I4GL PRINT COUNT(\*)** ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] = getReportVariableValue("variable_COUNT_number");
```

**PERCENT:** I4GL **PERCENT** 集約関数は、2 つの JasperReports XML 設計変数と、2 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである

```
PRINT PERCENT(variable)
```

は、EGL では以下の 4 つの要素に変換されます。

1. 2 つの JasperReports XML 設計変数
  - `variable_PERCENT_number_PART`
  - `variable_PERCENT_number_WHOLE`
2. それぞれが JasperReports 変数の最大値を識別する、2 つの EGL レポート・ハンドラー関数
  - Function update `variable_PERCENT_number_PART`
  - Function update `variable_PERCENT_number_WHOLE`

XML 設計変数と EGL レポート・ハンドラー関数の両方で、以下が適用されます。

- Function update `variable_PERCENT_number_PART` では
  - `variable` が、条件付き文節が満たされるすべての変数の値を累積します。
- Function update `variable_PERCENT_number_WHOLE` では
  - `variable` が、レコード・セット内のすべての変数の値を累積します。

上記の両方の EGL レポート・ハンドラー関数で、`number` は I4GL **PERCENT** 集約関数の各オカレンスを区別する通し番号で、固有の `variable` 名を作成するために使用されます。

**注:** レコードのメンバーである場合は、変数名が変わります。

変換中に、`getReportVariableValue()` レポート・ライブラリー API を使用して XML 設計レポート変数を呼び出すために、コードが生成されます。したがって、上記の I4GL **PRINT PERCENT(variable)** ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] =  
getReportVariableValue("variable_PERCENT_number_PART") /  
getReportVariableValue("variable_PERCENT_number_WHOLE") * 100;
```

**AVG:** I4GL **AVG** 集約関数は、2 つの JasperReports XML 設計変数と、2 つのレポート・ハンドラー関数を生成します。例えば、単一の I4GL ステートメントである

```
PRINT AVG(variable)
```

は、EGL では以下の 4 つの要素に変換されます。

1. 2 つの JasperReports XML 設計変数
  - `variable_AVG_number_SUM`
  - `variable_AVG_number_COUNT`
2. それぞれが JasperReports 変数の最大値を識別する、2 つの EGL レポート・ハンドラー関数
  - Function update `variable_AVG_number_SUM`
  - Function update `variable_AVG_number_COUNT`

JasperReports XML 設計変数と EGL レポート・ハンドラー関数の両方で、以下が適用されます。

- Function `update variable_AVG_number_SUM` では
  - `variable` が、条件付き文節が満たされるすべての変数の値を累積します。
- Function `update variable_AVG_number_COUNT` では
  - `variable` が、レコード・セット内のすべての変数の値を累積します。

上記の両方の EGL レポート・ハンドラー関数で、`number` は I4GL **AVG** 集約関数の各オカレンスを区別する通し番号で、固有の `variable` 名を作成するために使用されます。

注: レコードのメンバーである場合は、変数名が変わります。

変換中に、`getReportVariableValue()` レポート・ライブラリー API を使用して XML 設計レポート変数を呼び出すために、コードが生成されます。したがって、上記の I4GL **PRINT AVG**(`variable`) ステートメントは、EGL では次のようになります。

```
egl4gl_printString[1] =  
getReportVariableValue("variable_AVG_number_SUM") /  
getReportVariableValue("variable_AVG_number_COUNT");
```

EGL レポートを実装する方法や、新規レポートを作成する方法については、インフォメーション・センターのトピック「EGL レポート」を参照してください。

---

## EGL プロジェクト、パッケージ、およびファイルの理解

EGL プロジェクトには、ソース・フォルダーがゼロから多数まで含まれています。それぞれのフォルダーには、パッケージがゼロから多数まで含まれています。それぞれのパッケージには、ファイルがゼロから多数まで含まれています。それぞれのファイルには、パーツがゼロから多数まで含まれています。

注: 以下の情報は Rational 製品のインフォメーション・センターでも表示され、他のインフォメーション・センターのトピックと相互参照できます。

### EGL プロジェクト

EGL プロジェクトは、一連のプロパティによって特徴付けられています。EGL プロジェクトのコンテキストでは、特定のタスク実行時 (例えば、EGL ファイルまたは EGL ビルド・ファイルの保管時) に、EGL が自動的に検証を実行し、パーツ参照を解決します。また、ページ・ハンドラー・パーツの操作を行う場合、EGL は、以下の場合にかぎり自動的に出力を生成します。

- 次のオプション「ウィンドウ」>「設定」>「ワークベンチ」>「リソース変更時にビルドを自動的に実行」の選択後に、自動ビルド・プロセスを設定した場合
- デフォルトのビルド記述子が設定またはプロパティとして設定されている場合

EGL プロジェクトを構成するには、新規プロジェクトの作成時にプロジェクト・タイプとして **EGL** または **EGL Web** を選択します。プロジェクト作成ステップを実行中に、プロパティを割り当てます。これらのステップの完了後に選択を変更するには、プロジェクト名を右マウス・ボタン・クリックし、表示されたコンテキスト・メニューで、「プロパティ」をクリックします。

EGL プロパティについて、以下のセクションで説明します。

## EGL ソース・フォルダー

プロジェクトのパッケージのルートである 1 つ以上のプロジェクト・フォルダー。各フォルダーは、サブディレクトリーの集合です。ソース・フォルダーは、Java ファイルとは区別して EGL ソースを保持したり、Web デプロイメント・ディレクトリー以外の場所に EGL ソース・ファイルを保持する場合に役立ちます。どのような場合でも、EGL ソース・フォルダーを指定してください。ただし、ソース・フォルダーが指定されていない場合には、唯一のソース・フォルダーはプロジェクト・ディレクトリーになります。

このプロパティの値は、プロジェクト・ディレクトリーの **.eglp** という名前のファイルに格納され、EGL ファイルの保管に使用するリポジトリ (存在する場合) に保管されます。

それぞれの EGL プロジェクト・ウィザードは、EGLSource という名前の 1 つのソース・フォルダーを作成します。

## EGL ビルド・パス

現行プロジェクトに存在しない任意のパーツで検索されるプロジェクト・リスト。

このプロパティの値は、プロジェクト・ディレクトリーの **.eglp** という名前のファイルに格納され、EGL ファイルの保管に使用するリポジトリ (存在する場合) に保管されます。

次の **.eglp** ファイルの例では、EGLSource は現行プロジェクト内のソース・フォルダーで、**AnotherProject** は EGL パス内のプロジェクトです。

```
<?xml version="1.0" encoding="UTF-8"?>
<eglp>
  <eglpentry kind="src" path="EGLSource"/>
  <eglpentry kind="src" path="AnotherProject"/>
</eglp>
```

**AnotherProject** のソース・フォルダーは、そのプロジェクトの **.eglp** ファイルによって判別されます。

## デフォルトのビルド記述子

出力を迅速に生成できるビルド記述子 (インフォメーション・センターの「ワークベンチでの生成」トピックを参照)

## パッケージ

パッケージは、関連するソース・パーツの名前付きコレクションです。

ビルド・パーツを作成するときに、パッケージは使用されません。

規則に従って、パッケージ名の最初の部分を組織のインターネット・ドメイン名を語順反転したものにすることによって、パッケージ名の一意性を実現します。例えば、IBM ドメイン名は **ibm.com** なので、EGL パッケージは「**com.ibm**」で始まるものになります。この規則を使用することによって、組織が開発した Web プログラム名は、他の組織が開発したプログラム名と重複しないということが保証され、名前が衝突することなく同一サーバーにインストールできます。

個々のパッケージのフォルダーは、パッケージ名によって識別されます。これは、ID をピリオド (.) で区切ったシーケンスです。例を次に示します。

`com.mycom.mypack`

それぞれの ID は、EGL ソース・フォルダー内のサブフォルダーに対応しています。例えば、**com.mycom.mypack** のディレクトリー構造は `¥com¥mycom¥mypack` で、ソース・ファイルは最下位のフォルダー (この場合には **mypack**) に格納されます。ワークスペースが `c:¥myWorkspace`、プロジェクトが `new.project`、ソース・フォルダーが `EGLSource` の場合には、このパッケージのパスは次のようになります。

`c:¥myWorkspace¥new.project¥EGLSource¥com¥mycom¥mypack`

EGL ファイル内のすべてのパーツは同一のパッケージに属します。ファイルの `package` 文がある場合、その文でそのパッケージ名を指定します。 `package` 文を指定しない場合には、パーツはソース・フォルダーに直接格納されます。つまり、**デフォルト・パッケージ**に格納された状態になります。 `package` ステートメントは、常に指定してください。これは、デフォルト・パッケージ内のファイルは、他のパッケージまたはプロジェクト内のパーツによって共有できないためです。

同一の ID を持つ 2 つのパーツは、同じパッケージに定義することはできません。

**注:** 同じパッケージ名を異なるプロジェクトまたは異なるフォルダーで使わないでください。

生成された Java 出力のパッケージは、ほとんどの場合 EGL ファイル・パッケージと同じです。

## EGL ファイル

すべての EGL ファイルは、以下のいずれかのカテゴリーに含まれます。

### ソース・ファイル

EGL ソース・ファイル (拡張子 `.egl`) には、ロジック、データ、およびユーザー・インターフェース・パーツが含まれ、EGL ソース形式で作成されています。以下の生成可能パーツは、コンパイル可能単位に変換できます。

- `DataTable`
- `FormGroup`
- `Handler` (レポート・ハンドラーの基礎)
- ライブラリー
- `PageHandler`
- プログラム

その他のパーツはサブパーツと呼ばれます。

EGL ソース・ファイルには、サブパーツをゼロ個以上いくつでも組み込むことができますが、生成可能パーツは 1 つだけ組み込むことができます。生成可能パーツ (存在する場合) は、ファイルの最上位にある必要があり、そのファイルと同じ名前になっている必要があります。

## ビルド・ファイル

EGL ビルド・ファイル (拡張子 **.eglbld**) には、ビルド・パーツをいくつでも組み込むことができ、Extensible Markup Language (XML)、EGL ビルド・ファイル形式で作成されています。関連する DTD (以下のディレクトリーにあります) を検討することができます。

```
installationDir%egl%eclipse%plugins%com.ibm.etools.egl_version
```

## 推奨事項

このセクションでは、ユーザーの開発プロジェクトの設定に関する推奨事項について説明します。

### ビルド記述子の場合

プロジェクト・チームでは、ビルド記述子開発者として 1 人を指名する必要があります。開発者の作業は以下のとおりです。

- ・ ソース・コード開発者用のビルド記述子を作成する
- ・ これらのビルド記述子をソース・コード・プロジェクトとは別のプロジェクトに配置する。さらに、リポジトリまたはその他の方法で、その別のプロジェクトを利用可能にします。
- ・ ソース・コード開発者に、プロジェクト内にプロパティー・デフォルトのビルド記述子を設定するように依頼する。これによって、このプロパティーは、適切なビルド記述子を参照します。
- ・ ビルド記述子オプションの小さなサブセット (ユーザー ID およびパスワードなど) が、1 人のソース・コード開発者と次のソース・コード開発者で異なる場合には、それぞれのソース・コード開発者に以下を行うよう依頼します。
  - **nextBuildDescriptor** オプションを使用してグループ・ビルド記述子を指す個人用ビルド記述子をコード化する。
  - ソース・コード開発者に、ファイル、フォルダー、またはパッケージ内にプロパティーを、**デフォルトのビルド記述子**に設定するように依頼する。これにより、このプロパティーは、個人用ビルド記述子を参照します。開発者は、プロパティーをプロジェクト・レベルでは設定しません。これは、プロジェクト・レベルのプロパティーが、他のプロジェクト情報とともに、リポジトリによって制御されるためです。

追加情報については、「ビルド記述子パーツ」を参照してください。

### パッケージの場合

パッケージに関する推奨事項は、次のとおりです。

- ・ 異なるプロジェクトまたは異なるソース・ディレクトリーで、同じパッケージ名は使用しない
- ・ デフォルト・パッケージは使用しない

### パーツの割り当て

パーツに関する推奨事項の多くはベスト・プラクティスに言及したものであり、必須要件とは異なります。別の方法で行う正当な理由がない場合には、以下のオプションの推奨事項も実行してください。

- 要件 は、関連付けられているページ・ハンドラーと同じプロジェクトに JSP を配置することです。
- サブパーツ (レコード・パーツなど) が、1 つのプログラム、ライブラリー、またはページ・ハンドラーでのみ使用される場合は、パーツと同じファイル内にそのサブパーツを配置してください。
- パーツが同じパッケージ内の異なるファイルから参照される場合は、そのパッケージ内の別のファイルにそのパーツを配置します。
- パーツが単一プロジェクト内のパッケージ間で共用される場合は、そのプロジェクト内の別のパッケージにそのパーツを配置します。
- 完全に無関係なアプリケーションのコードは、異なるプロジェクトに配置します。プロジェクトは、ローカル・ディレクトリー構造とリポジトリ間でコード転送を行う単位です。開発者が開発システムにロードする必要があるコードの量を最小化できるようにプロジェクト構造を設計します。
- プロジェクト、パッケージ、およびファイルに、それらに含まれるパーツの使用法を反映した名前を付けます。
- プロセスで開発者のコード所有権が主張されている場合は、所有者が異なるパーツを同じファイルに割り当てないでください。
- パッケージの目的を明確に理解してから、パッケージにパーツを割り当てます。また、パーツ間の関係の緊密度に応じてパーツをグループ化します。以下の特徴は重要です。
  - 同一のパッケージ内でファイルからファイルへパーツを移動するために、他のファイルの `import` 文を変更する必要はありません。
  - 1 つのパッケージから他のパッケージへパーツを移動する場合は、移動するパーツを参照するすべてのファイルに、`import` ステートメントを追加するか、または変更する必要がある場合があります。

---

## インフォメーション・センターのヘルプ・システムと EGL チュートリアル

インフォメーション・センターには、EGL と一連の製品機能の両方を使用する方法についての、幅広い資料が含まれています。メインメニューから「ヘルプ」>「**Rational ヘルプ**」と選択して、インフォメーション・センターにアクセスしてください。

EGL チュートリアルでは、EGL を使用して簡単な動的 Web サイトをビルドする方法について学びます。チュートリアルには、メインメニューから「ヘルプ」>「チュートリアル・ギャラリー」を選択し、ギャラリーの左側のペインから「実践」を選択すると、アクセスできます。チュートリアルは、以下を学習する際に役に立ちます。

- EGL プロジェクトのセットアップと構成
- EGL ソース・コードの作成
- リレーショナル・データベースのデータにアクセスする 2 つの単純な Web ページの作成
- 1 つの Web ページから別の Web ページへのパラメーターの受け渡し
- Web アプリケーション・サーバーの構成およびそのサーバー上でのアプリケーションの実行

チュートリアルの演習には、以下のものが含まれます。

- EGL のセットアップ
- EGL プロジェクトの作成と構成
- WebSphere Application Server の始動と構成
- EGL データ・パーツの作成
- EGL ライブラリーの作成
- Web ページの作成
- データのページへの追加
- 別のページへのリンク
- 更新ページの作成



---

## 第 5 章 再変換プロセスおよびタスク

本章の内容	5-1
I4GL 共用ライブラリーを再変換する場合	5-1
I4GL 共用ライブラリーの再変換の方法	5-1
変換ウィザードの再変換	5-2
コマンド行の再変換	5-2
失敗した再変換の原因と次善策	5-3

---

### 本章の内容

この章では、I4GL 共用ライブラリーをいつ、どのように再変換するかを説明します。

---

#### I4GL 共用ライブラリーを再変換する場合

I4GL 共用ライブラリーの変換中に、ライブラリー固有のマニフェスト・ファイルがディレクトリー **EGLDestinationDirectory/ConversionArtifacts/manifest** で生成されます。このマニフェスト・ファイルは、**ProjectnameProjecttypeManifest.xml** という命名規則があります。例えば、**MyLibrary** という共用ライブラリー・プロジェクトの場合、マニフェスト・ファイル名は **MyLibraryLibraryManifest.xml** になります。マニフェスト・ファイルは、各ライブラリーで使用される関数、グローバル変数、および書式を判別します。

I4GL アプリケーション変換中に、変換ユーティリティーは、ソース・ファイル内の関数呼び出し参照と、従属共用ライブラリーのマニフェスト・ファイルを比較します。ソース・ファイル内の関数呼び出し参照と従属マニフェスト・ファイル内の関数参照に整合性がない場合、両方とも調整する必要があります。従属マニフェスト・ファイルは、正しい関数呼び出し参照で更新し、共用ライブラリー・プロジェクトは、更新されたマニフェスト・ファイルを使用して再変換する必要があります。

---

#### I4GL 共用ライブラリーの再変換の方法

I4GL アプリケーション変換中に、変換ユーティリティーは、従属共用ライブラリー・マニフェスト・ファイルを調整します。このマニフェスト・ファイルは、従属共用ライブラリー再変換用に使用してください。

変換ユーティリティーがこの調整されたマニフェスト・ファイルを書き込む際に、ファイル名を **filename.bak.num** として元の従属マニフェスト・ファイルのバックアップをとり、所定のマニフェスト・ファイルを調整されたマニフェスト・ファイルに置き換えます。正しいマニフェスト・ファイルが従属共用ライブラリーの再変換に使用されていることを確認してください。

以下の 2 つの方式で共用ライブラリーを再変換できます。

- 変換ウィザードを使用
- コマンド行を使用

注: 初期の変換プロセスでコマンド行モードを使用すると、ユーザーが手動で構成ファイルを作成する必要があるため、既存の構成ファイルが使用できる場合は、再変換のみにコマンド行オプションを使用するほうが簡単です。

---

## 変換ウィザードの再変換

共用ライブラリーを再変換するには、次のように実行します。

1. Rational 製品の EGL パースペクティブから、「ファイル」>「新規作成」>「その他」>「EGL 変換に対する Informix 4GL」>「共用ライブラリー変換ウィザード」を選択します。
2. 「I4GL 共用ライブラリー変換プロジェクト (I4GL Shared Library Conversion Project)」画面で次の情報を挿入してください。
  - プロジェクトの詳細。
    - 再変換プロジェクト。このオプションを選択すると、既存の構成ファイルと調整されたマニフェスト・ファイルのロケーションがわかります。参照してファイルを見つけることができます。
  - 変換成果物。変換ユーティリティーは、構成ファイル、マニフェスト・ファイル、および変換ログ・ファイルなど、変換に関係する多くの成果物を生成します。デフォルトでは、変換成果物は EGL 宛先ディレクトリーに配置されます。また、外部ディレクトリーに変換成果物を作成するように指定することもできます。適切なディレクトリーを参照することはできますが、この時点で新規のディレクトリーを作成することはできません。
3. 変換プロジェクトの詳細。構成ファイルおよびマニフェスト・ファイルの内容を含む、プロジェクトの詳細を検討します。
4. 「完了」をクリックして、再変換を起動します。

---

## コマンド行の再変換

コマンド行から共用ライブラリーを再変換することができます。

コマンド行を使用して共用ライブラリーを再変換する手順は、次のとおりです。

1. コマンド行を開きます。
2. プロンプトで、以下を入力します。ここで、*configurationfile* は前の変換で使用された構成ファイルの名前、*manifestfile* は、変換ユーティリティーによって作成された更新済みマニフェスト・ファイルの名前です。
  - a. Windows の場合:  
`e4gl.bat configurationfile -reconversion manifestfile`
  - b. Linux の場合:  
`e4gl.sh configurationfile -reconversion manifestfile`

注: コマンド行ユーティリティーを使用して、I4GL アプリケーションを変換する方法については、3-10 ページの『変換ユーティリティーのコマンド行モード』を参照してください。

## 失敗した再変換の原因と次善策

共用ライブラリーの再変換は、下記の 表 5-1 で記されている理由で失敗する場合があります。

表 5-1. 共用ライブラリー再変換失敗の理由

再変換失敗の理由	次善策
構成ファイルの破損	構成ファイルを手動で訂正し編集するか、変換ユーティリティー・ウィザードを使用して新規の破損されていない構成ファイルを再生成します。
マニフェスト・ファイルの破損	変換ユーティリティー・ウィザードを使用して、共用ライブラリーを <b>新規プロジェクト</b> として変換し、新規のマニフェスト・ファイルを作成します。新規マニフェスト・ファイルを使用してアプリケーション・プロジェクトを変換し、アプリケーション・プロジェクトの調整されたマニフェスト・ファイルを使用して、共用ライブラリーを再変換します。
不十分なディスク・スペース	十分なディスク・スペースを作成します。
ファイル・システムの書き込み許可がない	ファイル・システムの書き込み許可を取得します。
リストされた I4GL ソース・ファイルの読み取り不能	各ソース・ファイルの名前とパスを検証します。



---

## 付録 A. I4GL から EGL への構文のマッピング

---

### 本付録の内容

この付録では、I4GL および EGL 構文の対応を示し、以下の表を記載しています。

- 『データ型』、下記
- A-3 ページの『特殊なデータのキャスト』
- A-4 ページの『定義および宣言ステートメント』
- A-5 ページの『ストレージ操作ステートメント』
- A-6 ページの『プログラム・フロー制御ステートメント』
- A-8 ページの『コンパイラ・ディレクティブ』
- A-9 ページの『I4GL 書式から EGL コンソール・ユーザー・インターフェースへ』
- A-12 ページの『4GL レポート実行ステートメント』
- A-13 ページの『4GL ビルトイン関数、変数、および定数』
- A-14 ページの『ビルトインおよび外部 SQL 関数とプロシージャ』
- A-15 ページの『演算子』
  - A-15 ページの『キーワード・ベースの演算子』
  - A-16 ページの『非英字の記号によって表される演算子』
- A-17 ページの『SQL カーソル操作ステートメント』
- A-18 ページの『SQL データ定義ステートメント』
- A-19 ページの『SQL データ操作ステートメント』
- A-19 ページの『SQL 動的管理ステートメント』
- A-19 ページの『SQL 照会最適化ステートメント』
- A-20 ページの『SQL データ・アクセス・ステートメント』
- A-20 ページの『SQL データ保全性ステートメント』
- A-20 ページの『SQL ストアード・プロシージャ・ステートメント』
- A-21 ページの『SQL クライアント/サーバー接続ステートメント』
- A-21 ページの『SQL オプティカル・サブシステム・ステートメント』
- A-22 ページの『環境変数』

---

### データ型

すべての EGL プリミティブ型が記述され、その振る舞いはオンライン・ヘルプ・トピックの「プリミティブ型」で説明されています。

注: RECORD 定義は、生成されたファイルの最後に付加され、その型の定義と最初の変数を含むファイルおよび関数に基づいた名前を取得します。

表 A-1. 4GL データ型を EGL プリミティブ型へマップする方法

I4GL	EGL
ARRAY OF:  DEFINE <i>x</i> array[10] of integer  DEFINE <i>myrec</i> ARRAY[10,2] of RECORD INT <i>x</i> , INT <i>y</i> END RECORD	x int[10];  Myrec recordtype_filename_myrec[10][2];  Record recordtype_filename_myrec type SqlRecord x Int; y Int; End
BYTE	BLOB
BIGINT	BIGINT
CHAR ( <i>size</i> ) CHAR	UNICODE (SIZE) UNICODE(1)
CHARACTER	UNICODE(1)
DATE	DATE
DATETIME YEAR TO FRACTION(3) DATETIME YEAR TO HOUR	Timestamp ("yyyyMMddhhmmssfff"); Timestamp ("yyyyMMddhh");
DEC	DECIMAL
DECIMAL( <i>p</i> , <i>s</i> )	DECIMAL( <i>p</i> , <i>s</i> )
DECIMAL( <i>p</i> ) NUMERIC( <i>p</i> )	DECIMAL( <i>p</i> ) (ANSI データベースの場合) DOUBLE (ANSI 以外のデータベースの場合) FLOAT (ANSI 以外のデータベースの場合) • ANSI データベースはデータベース・スキーマ・マ ニフェスト・ファイルで検出される値から決定され ます。
DOUBLE PRECISION	FLOAT
DYNAMIC ARRAY DEFINE <i>myA</i> DYNAMIC ARRAY WITH 3 DIMENSIONS of <i>xxxx</i>	myA xxx[][][];
FLOAT	FLOAT
INT INTEGER	INT INT
INT8	BIGINT
INTERVAL YEAR(9) TO MONTH INTERVAL DAY(7) TO FRACTION(3)	Interval("yyyyyyyyMM" /* YEAR(9) TO MONTH */) Interval("ddddddhhmmssfff")
MONEY	MONEY
NCHAR( <i>size</i> )	UNICODE( <i>size</i> )
NVARCHAR( <i>size</i> )	String( <i>size</i> )
REAL	SMALLFLOAT

表A-1. 4GL データ型を EGL プリミティブ型へマップする方法 (続き)

I4GL	EGL
RECORD	<pre> /* RECORD は、データベース・スキーマの抽出 プロジェクト中に EGL ファイルによって定義 されます。*/ Package IfmxDatabaseSchema.Svrname.Dbname; Dataitem like_tabname_code UNICODE(10) { ... properties ... }; Dataitem like_tabname_quantity INT { ... properties ... }; Record rec_like_tabname type SQLRecord { tablenames=["table"]} code like_tabname_code { column="code"} quantity like_tabname_quantity { columnname="code"} endImport IfmxDatabaseSchema.Svrname.Dbname.*; Var like_table_code; Recvar rec_like_table; </pre>
SMALLFLOAT	SMALLFLOAT
SMALLINT	SMALLINT
TEXT	CLOB
VARCHAR( <i>size</i> )	String( <i>size</i> )

## 特殊なデータのキャスト

表 A-2. 特殊なデータのキャスト

I4GL	EGL
<u>DEFINE <i>i</i> INTEGER, <i>d</i> DATE;</u> LET <i>i</i> = <i>d</i> ; LET <i>d</i> = <i>I</i> ;	<i>i</i> = <i>d</i> ; <i>d</i> = <i>DateTimeLib.dateFromInt</i> ( <i>i</i> );;
<u>DEFINE <i>c</i> CHAR(80), <i>d</i> DATE;</u> LET <i>c</i> = <i>d</i> ;	<i>C</i> = <i>d</i> ; <i>/* システムが、ロケールに基づいて d のフォーマット済みストリングを作成します */</i>
<u>DEFINE <i>ds</i> DATETIME YEAR</u> <u>TO SEC;</u> DISPLAY "now is", <i>ds</i> ;	<i>displayLineMode</i> ("now is "+ <i>ds</i> ); <i>/* システムが、ロケールに基づいて ds のフォーマット済みストリングを作成します */</i>
<u>DEFINE <i>inv</i> INTERVAL DAY</u> <u>TO SEC;</u> DISPLAY "remaining time is", <i>inv</i> ;	<i>displayLineMode</i> ("remaining time is "+ <i>inv</i> ); <i>/* システムがロケールに基づいて inv のフォーマット済みストリングを作成します */</i>

## 定義および宣言ステートメント

表 A-3. 4GL 定義および宣言ステートメントを EGL ヘマップする方法

I4GL	EGL
DEFINE <i>x</i> INTEGER	x int
FUNCTION <i>fn1(a,b)</i> DEFINE <i>a</i> INTEGER DEFINE <i>b</i> INTEGER DEFINE <i>c</i> INTEGER // ... RETURN <i>c</i> END FUNCTION	function fn1 (a INT in, b INT in) returning (INT) <i>c</i> int;  // ... return ( <i>c</i> ); end //function
Multiple returns: FUNCTION <i>fn1(a,b)</i> DEFINE <i>a</i> INTEGER DEFINE <i>b</i> INTEGER DEFINE <i>c</i> INTEGER DEFINE <i>d</i> INTEGER  RETURN <i>c,d</i> END FUNCTION	function fn1 (a INT in, b INT in \$_retvar1 INT out, \$_retvar2 INT out) <i>c</i> int; <i>d</i> int;  \$_retvar1 = <i>c</i> \$_retvar2 = <i>d</i> ; return; end//Function
DEFINE <i>r</i> RECORD <i>ii</i> int, <i>jj</i> int END RECORD  CALL <i>foo( )</i> RETURNING <i>r.*</i> ;  FUNCTION <i>foo( )</i> DEFINE <i>c, d</i> INT RETURN <i>c, d</i> END FUNCTION	<b>Call:</b> foo (/*returning*/ rr.ii, rr.jj);  <b>Function:</b> FUNCTION FOO ( /*returning*/ \$_retvar_1 INT OUT, \$_retvar_2 INT OUT) <i>c</i> INT; <i>d</i> INT; \$_retvar_1 = <i>c</i> ; \$_retvar_2 = <i>d</i> ; return ; END
Record.mem1 THRU Record.mem2: FUNCTION <i>fn1(r.a THRU r.c)</i> DEFINE <i>r</i> LIKE <i>tablename.*</i> ;  END FUNCTION	// メンバー・リストを個々の値に展開します FUNCTION fun1 (a INT IN, b UNICODE(20) IN, c INT IN) ... ... END // FUNCTION

表 A-3. 4GL 定義および宣言ステートメントを EGL ヘマッピングする方法 (続き)

I4GL	EGL
GLOBALS ... END GLOBALS  DEFINE <i>fvar</i> INT; GLOBALS DEFINE <i>gvar</i> INT; END GLOBALS  GLOBALS <i>filename</i>	GLOBALS セクションではなく、ファイルの上部に定義された変数は、ファイルに対して <code>private</code> になります。 GLOBALS セクションで定義された変数は、他のファイルやパッケージから参照できます。  private <i>fvar</i> int; <i>gvar</i> int;  import packageOfFilename.*; // ライブラリー宣言 use <i>filename</i> ;
LABEL LABEL xxx:	xxx: • GOTO ステートメントと共に使用されます
MAIN	FUNCTION \$_filename_main()
REPORT	External tool

## ストレージ操作ステートメント

表 A-4. 4GL ストレージ操作ステートメントを EGL ヘマッピングする方法

I4GL	EGL
DEFINE <i>var</i> BLOB, <i>var2</i> TEXT; LOCATE <i>var</i> , <i>var</i> , <i>var2</i> IN MEMORY  LOCATE <i>var</i> , <i>var2</i> IN FILE  LOCATE <i>var</i> , <i>var2</i> IN FILE " <i>filename</i> ";  LOCATE <i>var</i> , <i>var2</i> IN FILE <i>filevar</i> ;	Var Blob; Var2 Clob; //LOCATE <i>var</i> , <i>var</i> , <i>var</i> IN MEMORY;  attachBlobToTempFile( <i>var</i> ); attachBlobToTempFile( <i>var2</i> );  attachBlobToFile( <i>var</i> , " <i>filename</i> "); attachBlobToFile( <i>var2</i> , " <i>filename</i> ");  attachBlobToFile( <i>var</i> , <i>filevar</i> ); attachBlobToFile( <i>var2</i> , <i>filevar</i> );
FREE <i>preparedStatement</i> FREE <i>Cursor</i> FREE <i>bytevar</i> FREE <i>textvar</i>	FreeSql ( <i>preparedStatement</i> ); // フリー・カーソルは必須ではありません。 FreeBlob( <i>blobvar</i> ); FreeClob( <i>clobvar</i> );
INITIALIZE	SET
LET <i>x</i> = 10;	<i>x</i> = 10;
VALIDATE	サポートされていません。
VALIDATE LIKE	

## プログラム・フロー制御ステートメント

表 A-5. 4GL プログラム・フロー制御ステートメントを EGL ヘマッピングする方法

I4GL	EGL
<pre>/* 14GL では、foreach は open、 fetch、close を行います。*/ DECLARE cursorname FOR stmt: FOREACH cursorname USING a, b INTO x,y,z WITH REOPTIMIZE END FOREACH</pre>	<pre>FOREACH EGL foreach ステートメントは fetch、および close を行います。 //stmt のカーソル名を宣言します Open cursorname using a, b Foreach (from cursorname into x,y,z) /* REOPTIMIZE はサポートされていません */ End</pre>
<pre>CALL fn1(a,b) RETURNING c;  LET c = fn1(a,b)  CALL fn1(a,b) RETURNING a  CALL fn1(a,b) RETURNING a,b  CALL fn1(rec.*) RETURNING rec2.*;</pre>	<pre>c = fn1(a,b);  c = fn1(a,b);  a = fn1(a,b);  fn1(a,b, /* returning */ a,b);  fn1(rec.fld1, rec.fld2, ..., /*returning*/ rec2.fld1, rec2.fld2, ;</pre>
CASE	SWITCH CASE ... CASE ... CASE
CONTINUE	CONTINUE
<pre>DATABASE mydb@SERVER2;  DEFINE rec RECORD LIKE table.*  /*14GL は、内部的に \$database ステートメントを .ec ファイルの main 関数に put します。 これは MIN で生成されます*/</pre>	<pre>//DATABASE mydb@SERVER2 import IfmxSchema.server2.mydb.*;  rec IfmxSchema.server2.mydb.rec_like_table;  DefineDatabaseAlias("DEFAULT",getProperty ("mydb@SERVER2")); connect("DEFAULT", getProperty("DEFAULT_USERID"), getProperty("DEFAULT_PASSWORD"), type1, explicit, repeatableRead, noAutoCommit); // ANSI DB 設定</pre>
<pre>EXIT CASE EXIT CONSTRUCT EXIT DISPLAY EXIT FOR EXIT FOREACH EXIT INPUT EXIT MENU EXIT PROGRAM EXIT REPORT EXIT WHILE</pre>	<pre>Exit Exit openUI Exit openUI Exit for Exit foreach Exit openUI Exit openUI Exit openUI Exit rununit Call reportName_TERMINATE(); Exit while</pre>
FOR	FOR

表 A-5. 4GL プログラム・フロー制御ステートメントを EGL ヘマッピングする方法 (続き)

I4GL	EGL
GLOBALS <i>filename</i>	//GLOBALS "filename"  このステートメントは無視されます。 アプリケーション変換プロジェクトと外部プロジェクトへの個々のライブラリー参照に含まれる、 ライブラリー・マニフェストとデータベース・スキーマに基づいて、必須の Imports ステートメントと Use ステートメントが生成されます。
GOTO	GOTO
IF	IF
IF <cond> THEN ELSE END IF	if (<cond>) else end
OUTPUT TO REPORT	外部ツールを介して間接的に
RETURN; RETURN <i>a</i> ;  RETURN <i>rec.*</i> ;  RETURN <i>a, b</i> ;  RETURN <i>rec.col_a</i> THRU <i>rec.col_c</i> ;	RETURN; RETURN ( <i>a</i> );  \$_retvar_1 = rec.fld1; \$_retvar_2 = rec.fld2; ... RETURN;  \$_retvar1 = <i>a</i> ; \$_retvar2 = <i>b</i> ; return;  function ( \$_retvar_1 type OUT, \$_retvar_2 type OUT, \$_retvar_3 type OUT) ... \$_retvar_1 = <i>rec.col_a</i> ; \$_retvar_2 = <i>rec.col_b</i> ; \$_retvar_3 = <i>rec.col_c</i> ; return; end
RUN " <i>cmd</i> " LINE RUN " <i>cmd</i> " FORM  RUN " <i>cmd</i> " WITHOUT WAITING  RUN .. RETURNING <i>x</i> ;	CallCmd("cmd", Line); CallCmd("cmd", Form);  StartCmd("cmd", Line);  CallCmd("cmd",Line); <i>x</i> = sysVar.returnValue;
SQL ... END SQL	execute #sql {...}
WHILE <i>cond</i> ... END WHILE	while ( <i>cond</i> ) ... end

注: FINISH REPORT、OUTPUT TO REPORT、START REPORT および  
 TERMINATE REPORT の各 4GL プログラム・フロー制御ステートメントのマ  
 ッピングについては、A-12 ページの『4GL レポート実行ステートメント』に記  
 載されています。

## コンパイラー・ディレクティブ

表 A-6. 4GL コンパイラー・ディレクティブを EGL ヘマッピングする方法

I4GL	EGL
DEFER INTERRUPT DEFER QUIT	consoleLib.deferInterrupt=YES; consoleLib.deferQuit=YES;
GLOBALS <i>filename</i>	Import packageOfFilename.*; Use <i>filename</i> ; // library name.
SQL ... END SQL	EXECUTE #sql{ ... } 注: プログラム変数は、コロン (:) 接頭部で識 別する必要があります。
WHenever SQLERROR CONTINUE; SQL ... END SQL SQL ... END SQL	//WHenever SQLERROR CONTINUE; Try EXECUTE #sql{ ... } onException end; Try EXECUTE #sql{ ... } onException end;
WHenever SQLERROR CALL XYZ; SQL ... END SQL  SQL ... END SQL	//WHenever SQLERROR CALL; try EXECUTE #sql{ ... } onException /* ERROR */ xzy() endtry EXECUTE #sql{ ... } onException /* ERROR */ xzy() end
WHenever SQLERROR GOTO :ABC;  SQL ... END SQL SQL ... END SQL	//WHenever SQLERROR GOTO; try execute #sql{ ... }; execute #sql( ... ); onException /* ERROR */ gotoABC; end
WHenever SQLERROR GOTO :ABC Whenever WARNING STOP  SQL ... END SQL  SQL ... END SQL	/*whenever sqlerror goto :ABC*/ /*whenever warning stop */;  try execute #sql{ ... }; if (SQLCODE > 0 && SQLCODE !=100) then exit program; end /* WARNING */ onException /* ERROR */ gotoABC; endtry execute #sql{ ... }; if (SQLCODE > 0 && SQLCODE !=100) then exit program; end /* WARNING */ onException /* ERROR */ gotoABC; end

表 A-6. 4GL コンパイラー・ディレクティブを EGL ヘマッピングする方法 (続き)

I4GL	EGL
WHENEVER ERROR CALL; SQL ... END SQL  INPUT ... END INPUT	//WHENEVER ERROR CALL abc; try Execute #sql{ ... } onException call abc(); endtry openUI ... End onException call abc(); end
WHENEVER SQLWARNING CALL abc; SQL ... END SQL  SQL ... END SQL	//WHENEVER SQLWARNING CALL abc; try Execute #sql{ ... } onException end if sqlcode > 0 then call abc(); endtry Execute #sql{ ... } onException end if sqlcode > 0 then call abc(); end

## I4GL 書式から EGL コンソール・ユーザー・インターフェースへ

表 A-7. 4GL 書式ステートメントから EGL コンソール・ユーザー・インターフェース・ステートメントへマッピングする方法

I4GL	EGL
CLEAR SCREEN	activateWindow (consoleLib.screen); --または-- clearActiveWindow();
CLEAR WINDOW SCREEN	clearWindow(consolelib.screen);
CLEAR WINDOW <i>windowName</i>	clearWindow([windowObject]); --または-- clearWindowByName({ windowName})
CLEAR FORM	clearActiveForm();
CLEAR <i>listOfFieldNames</i>	clearFields() --または-- clearConsoleFields(fieldName { , fieldName});
CLOSE FORM	NA
CLOSE WINDOW <i>windowName</i>	closeWindow(windowObject); --または-- closeWindowByName(windowName);
CONSTRUCT	openUI {isConstruct=yes} formObject end
CURRENT WINDOW IS <i>windowName</i>	activateWindow(windowObject); --または-- activateWindowByName(windowName)
CURRENT WINDOW IS SCREEN	activateWindow(consoleLib.screen)
DEFER INTERRUPT	consoleLib.deferInterrupt=yes;
DEFER QUIT	consoleLib.deferQuit=yes;
DISPLAY	displayLineMode()

表 A-7. 4GL 書式ステートメントから EGL コンソール・ユーザー・インターフェース・ステートメントへマップする方法 (続き)

I4GL	EGL
DISPLAY AT x,y;	displayAtPosition()
DISPLAY AT x;	displayAtLine():
DISPLAY a, b TO field1, field2	openUI {displayOnly=yes, bindingbyName=no} activeForm.field1, activeForm.field2 bind a,b end;
DISPLAY BY NAME fname, lname;	openUI {displayOnly=yes, bindingbyName=yes} activeForm bind fname, lname end;
DISPLAY BY NAME Re.*	openUI {displayOnly=yes, bindingbyName=yes} activeForm bind <each rec element>;
DISPLAY ARRAY	openUI {displayOnly=yes} formArrayDictionary bind programRecordArray end
DISPLAY FORM	clearForm() openUI {displayOnly=yes} consoleForm end
ERROR	displayError (message);
INPUT	openUI consoleForm bind program Variables end --または-- openUI consoleFieldList bind program Variables end
INPUT ARRAY	openUI activeForm.arrayDictionary bind programRecordArray end
MENU	openUI new Menu { labelText="Menu1", menuItems=[new MenuItem {accelerators={"F1"}, name="Cmd1", labelText="Command1" } //Repeat for other commands --separate each with a comma ] } OnEvent (MENU_ACTION:"Cmd1") ...egl statements... end;
MESSAGE	displayMessage (message);
NEXT FIELD fieldName	gotoFieldByName(fieldIdentifier)
OPEN FORM formName from fileName	formNameConsoleformType;

表 A-7. 4GL 書式ステートメントから EGL コンソール・ユーザー・インターフェース・ステートメントへマップする方法 (続き)

I4GL	EGL
OPEN WINDOW <i>windowName</i>	<pre>the Window {name="theWindow", size=[rr,cc]. Position=[zz,yy]}; openWindow (<i>theWindow</i>); --または-- openWindowByName("theWindow");</pre> <p>Migration Mapping:</p> <pre>openWindow(new Window{     name="theWindow", size=[rr,cc],     position=[xx,yy]});</pre>
OPTIONS	consoleLib.property を使用
PROMPT	<pre>openUI new Prompt {message="Do you want to continue?", isChar=yes} bind userAnswer end;</pre> <p>--または--</p> <pre>myPrompt Prompt {message="What is your name?"; openUI myPrompt bind usersName end;</pre> <p>ライン・モード操作の場合、promptLineMode(): を使用</p> <pre>userAnswer Char(1); userAnswer=promptLineMode ("Continue (y/n)?");</pre>
SCROLL	<pre>scrollDownPage() -または- scrollDownLines(<i>integerCount</i>) -または- scrollUpPage() -または- scrollUpLines (<i>integerCount</i>)</pre>
ARR_COUNT()	currentArrayCount()
ARR_CURR()	currentArrayDataLine()
SCR_LINE()	currentArrayScreenLine()
FGL_DRAWBOX()	<pre>drawBox() --または-- drawBoxWithColor()</pre>
FGL_GETKEY()	getKey()
FGL_KEYVAL()	getKeyCode()
NEXT OPTION <i>fieldName</i>	<pre>gotoMenuItem() --または-- gotoMenuItemByName()</pre>
INFIELD()	<pre>isCurrentField() --または-- isFieldModifiedByName()</pre>

表 A-7. 4GL 書式ステートメントから EGL コンソール・ユーザー・インターフェース・ステートメントへマップする方法 (続き)

I4GL	EGL
<i>FIELD_TOUCHED()</i>	isFieldModified() --または-- isFieldModifiedByName()
FGL_LASTKEY	lastKeyTyped()
NEXT_FIELD	nextField()
PREVIOUS_FIELD	previousField()
FGL_SETCURRLINE	setArrayLine()
<i>FGL_SCR_SIZE()</i>	syslib.size(screenArray)
<i>SHOWHELP()</i>	showHelp()

## 4GL レポート実行ステートメント

変換ユーティリティは、I4GL レポート・ファイルを EGL (.egl) および JasperReport (.jrxml) ファイルの両方に変換します。I4GL レポート構文の変換方法については、4-17 ページの『レポート変換の理解』を参照してください。

以下の I4GL レポート・ステートメントは、EGL や JasperReports に変換されませんが、文法的に正しいコードを維持するために、変換ユーティリティは、以下のステートメントのレポート・ハンドラーに冗長なコードを生成します。

- EXIT REPORT
- NEED
- PAUSE

表 A-8. 4GL レポート実行ステートメントを EGL/JasperReport へマップする方法

I4GL	EGL / JasperReport
PRINT	4-23 ページの『I4GL FORMAT セクション』を参照してください。
SKIP	外部ツールを介して間接的にマップされます。

## I4GL レポート・ドライバー・ステートメント

I4GL レポート・ドライバー・ステートメントは、EGL 関数へマップします。I4GL レポート・コードの例および EGL ドライバー関数における同等の例については、『付録 B. I4GL レポート変換コード例』を参照してください。

表 A-9. 4GL レポート・ドライバー・ステートメントを EGL へマップする方法

I4GL	EGL
FINISH REPORT <i>reportname</i>	<i>reportname</i> _FINISH( )
OUTPUT TO REPORT <i>reportname</i>	<i>reportname</i> _OUTPUT ( )
OUTPUT TO REPORT <i>reportname</i> (a, b, c)	<i>reportname</i> _OUTPUT (a, b, c)

表 A-9. 4GL レポート・ドライバー・ステートメントを EGL ヘマップする方法 (続き)

I4GL	EGL
START REPORT <i>reportname report options</i>	<i>reportname</i> _START( ) 注: START REPORT レポート・オプションは、EGL ヘマップしません。また、EGL 関数呼び出しは、引数も戻り値もありません。
TERMINATE REPORT <i>reportname</i>	<i>reportname</i> _TERMINATE ( )

## 4GL ビルトイン関数、変数、および定数

表 A-10. 4GL ビルトイン関数、変数、および定数を EGL ヘマップする方法

I4GL	EGL
ARG_VAL ( <i>int-expr</i> )	getCmdLineArg( <i>int-expr</i> )
ARR_COUNT ( )	currentArrayCount()
ASCII ( <i>int-expr</i> )	integerAsChar( <i>int-expr</i> )
COLUMN <i>int-expr</i>	直接的なマッピングはありません。 外部ツールを介して間接的にマップされます。
COLUMN <i>int-const</i> , PAGENO, LINENO	
CURSOR_NAME (" <i>Identifier</i> ")	サポートされていません。
DATE( <i>char</i> )	dateValue( <i>char</i> )
Date( <i>integer</i> )	dateValue( <i>integer</i> )
DAY( <i>datevalOrDatetime</i> )	dayOf( <i>dateexpr</i> );
DOWNSHIFT ( <i>char-expr</i> )	lowercase( <i>charexpr</i> )
ERR_PRINT ( <i>int-expr</i> )	displayError( <i>err_get</i> ("number"));
ERR_QUIT ( <i>int-expr</i> )	displayError( <i>err_get</i> ("number")); exit program;
ERRORLOG ( <i>int-expr</i> )	errorLog()
EXTEND ( <i>value, qualifier</i> )	extend ( <i>value, qualifierpattern</i> )
FALSE	0/*FALSE*/
FGL_DRAWBOX ( <i>nlines, ncols, begy, begx, color</i> )	drawBox( <i>int, int, int, int</i> ) drawBoxWithColor( <i>int, int, int, int, colorKind</i> )
FGL_GETENV ( <i>char-expr</i> )	getProperty( <i>charexpr</i> )
FGL_GETKEY ( ) // INPUT KEYSTROKE	getKey()
FGL_KEYVAL ( <i>char-expr</i> )	getKeyCode( <i>String</i> )
FGL_LASTKEY() //Doesn't wait,	lastKeyTyped()
FGL_SCR_SIZE ( <i>arrayname</i> )	syslib.size( <i>activeForm[arrayname]</i> );
FGL_SCR_SIZE (" <i>arrayname</i> ");	syslib.size( <i>activeForm."arrayname"</i> );
FGL_SETCURRLINE()	setArrayLine()
<i>Intval</i> UNITS <i>dateTlmeUnit</i>	IntervalValueWithPattern( <i>intval, "UnitsString"</i> );
LENGTH ( <i>char-expr</i> )	StrLeng( <i>charexpr</i> )

表 A-10. 4GL ビルトイン関数、変数、および定数を EGL ヘマッピングする方法 (続き)

I4GL	EGL
LET <i>a</i> = ERR_GET (SQLCODE) // <i>Get Message</i> LET <i>a</i> = ERR_GET ( <i>int-expr</i> ) // <i>Get Message</i>	err_get(SQLCODE); err_get( <i>int-expr</i> );
MONTH ( <i>datevalOrDatetime</i> )	monthOf( <i>dateexpr</i> );
MDY ( <i>intmonth, intdays, intyear</i> )	MDY( <i>intmonth, intdays, intyear</i> )
NEXT FIELD NEXT FIELD <i>fieldname</i>	nextField() gotoFieldByName (" <i>fieldname</i> ");
NEXT FIELD PREVIOUS	previousField()
NEXT OPTION <i>name</i>	gotoMenuItemByName(" <i>name</i> ");
NUM_ARGS ()	getCmdLineArgCount()
ORD ( <i>char-expr</i> )	characterAsInteger( <i>char-expr</i> )
SCR_LINE()	currentArrayScreenLine()
SET_COUNT ( <i>int-expr</i> )	setCurrentArrayCount()
SHOWHELP ( <i>int-expr</i> )	showHelp(" <i>int-expr</i> ");
STARTLOG (" <i>filename.filetype</i> ")	startLog()
TIME TIME ( <i>DateTimeValue</i> )	timeStamp(" <i>hhmmss</i> "); currentTime();
TODAY	currentDate();
TRUE	I*TRUE*/
UPSHIFT ( <i>char-expr</i> )	uppercase( <i>char-expr</i> )
WEEKDAY ( <i>datevalOrDatetime</i> )	weekdayOf( <i>dateexpr</i> );
YEAR ( <i>datevalOrDatetime</i> )	yearOf( <i>dateexpr</i> );

## ビルトインおよび外部 SQL 関数とプロシージャ

表 A-11. 4GL ビルトインおよび外部 SQL 関数とプロシージャを EGL ヘマッピングする方法

I4GL	EGL
CREATE FUNCTION	Execute #sql{ create function ... }
CREATE FUNCTION FROM	非サポート
CREATE PROCEDURE FROM	非サポート
CREATE ROUTINE FROM	非サポート
EXECUTE FUNCTION	Execute #sql{ execute function ... }
EXECUTE PROCEDURE	Execute #sql{ execute procedure ... }

## 演算子

### キーワード・ベースの演算子

表 A-12. 4GL キーワード・ベースの演算子を EGL ヘマップする方法

I4GL	EGL
ASCII <i>int-expr</i>	integerAsCharacter( <i>int-expr</i> )
AND	&&
<i>Value</i> BETWEEN <i>expr1</i> AND <i>expr2</i>	<i>Value</i> >= <i>expr1</i> && <i>Value</i> <= <i>expr2</i>
<i>char-expr</i> CLIPPED	Clip( <i>char_expr</i> )
CURRENT	currentTimeStamp() extend (currentTimeStamp(), "yyyMMddhhmmss";
CURRENT <i>qualifier</i>	
FIELD_TOUCHED ( <i>field-list</i> )	isFieldModified( <i>consoleField</i> ) --または-- isFieldModifiedByName( <i>String</i> )
GET_FLDBUF ( <i>field</i> )	activeForm.field
GET_FLDBUF ( <i>field-list</i> )	対応する field.Value について、個々に割り当て 注: GET_FLDBUF ( <i>field</i> ) パラメーターは、 table-reference.field、screen-record.field、または screen-array.field の 3 つの書式を受け入 れます。さらに、GET_FLDBUF ( <i>field</i> ) は、 プログラム・レコード書式も受け入れます が、それによって変換エラーが発生します。 例えば、 I4GL コード bb = get_fldbbuf(pr.aa) は、誤った EGL コード bb = activeForm.pr.aa.value; に変換され ます。 変換後に <b>pr.</b> フィールドを除去し、EGL コ ードを手動で訂正する必要があります。上記 の変換例の場合、訂正されたコードは bb = activeForm.aa.value; となります。
INFIELD ( <i>field</i> )	isCurrentField( <i>ConsoleField</i> ) isCurrentFieldByName( <i>String</i> )
INT_FLAG	interruptRequested
<i>Xyz</i> IS NULL	<i>Xyz</i> IS NULL
<i>Xyz</i> IS NOT NULL	<i>Xyz</i> NOT NULL
LENGTH ( <i>char-expr</i> )	StrLen ( <i>char-expr</i> )
LIKE	like !( <i>x</i> like <i>y</i> )
<i>x</i> NOT LIKE <i>y</i>	
LINENO	外部ツールを介して間接的にサポートされま す

表 A-12. 4GL キーワード・ベースの演算子を EGL ヘマップする方法 (続き)

I4GL	EGL
MATCHES <i>expr</i>	matches !( <i>x</i> matches <i>y</i> )
<i>x</i> NOT MATCHES <i>y</i>	
<i>int-expr</i> MOD <i>int-expr</i>	Int-expr % int-expr
NOT	!
NOT FOUND	100/*NOTFOUND*/
OR	
PAGENO	外部ツールを介して間接的にサポートされます
QUIT_FLAG	quitRequested
<i>int-expr</i> SPACE	Spaces( <i>int-expr</i> )
<i>int-expr</i> SPACES	Spaces( <i>int-expr</i> )
STATUS	SQLCODE
TIME	A-13 ページの『4GL ビルトイン関数、変数、および定数』を参照してください。
TODAY	A-13 ページの『4GL ビルトイン関数、変数、および定数』を参照してください。
<i>int-expr</i> UNITS <i>time-keyword</i>	A-13 ページの『4GL ビルトイン関数、変数、および定数』を参照してください。
<i>int-expr</i> USING <i>format-string</i>	Format( <i>expression</i> , <i>formatString</i> )  formatNumber(intval, "#####")  formatDate(mydate, "mm/dd/yy")
<i>char-expr</i> USING <i>format-string</i>	format( <i>char-exp</i> , <i>formatString</i> )
<i>datetime</i> USING <i>format-string</i>	formatDate( <i>datetime</i> , <i>formatString</i> )
<i>char-expr</i> WORDWRAP	JasperReports によって処理されます

## 非英字の記号によって表される演算子

表 A-13. 非英字の記号によって表される演算子を EGL ヘマップする方法

英字	4GL	EGL
Addition (加算)	+	+
コメント	-- 単一行コメント # 単一行コメント  { 複数行コメント }	// 単一行コメント  /* 複数行コメント */
Division (除法)	/	/
Exponentiation (指数)	**	**
より大きい	>	>

表 A-13. 非英字の記号によって表される演算子を EGL ヘマップする方法 (続き)

英字	4GL	EGL
Greater than or equal to (より大または等しい)	>=	>=
未満	<	<
Less than or equal to (より小または等しい)	<=	<=
Membership (メンバーシップ)	.	.
Modulus (係数)	MOD	%
Multiplication (乗算)	*	*
等しくない	!= or <>	!=
Sub string (サブストリング)	[first,last]	[first:last]
Subtraction (減算)	-	-
Unary negative (単項負)	-	-
Unary positive (単項正)	+	+

## SQL カーソル操作ステートメント

表 A-14. 4GL カーソル操作ステートメントを EGL ヘマップする方法

I4GL	EGL
CLOSE <i>cursorname</i>	try close <i>cursorname</i> ; onException end
DECLARE <i>c</i> CURSOR FOR <i>stmt</i> ; DECLARE C CURSOR FOR SELECT * FROM SYSTABLES;	/* <i>stmt</i> の <i>c</i> カーソルを宣言します */ try PREPARE \$_STMT_C FROM "SELECT * FROM SYSTABLES"; onException end 注: DECLARE からの Scroll および Hold 属性は、カーソルのオープン時に指定されます。
// WHENEVER SQLERROR STOP;  FETCH <i>c</i> ; FETCH NEXT C; FETCH PREVIOUS C; FETCH RELATIVE 1 C; FETCH RELATIVE -1 C; FETCH FIRST C; FETCH LAST C;	/* FYI: WHENEVER SQLERROR STOP */ / get next from <i>c</i> ; get next from C; get previous from C get relative( 1) from C; get relative( -1) from C; get first from C; get last from C;
FLUSH	NO-OP
FREE	freeSQL

表 A-14. 4GL カーソル操作ステートメントを EGL ヘマップする方法 (続き)

I4GL	EGL
OPEN	OPEN OPEN cursor WITH <i>statementid</i> ; OPEN cursor WITH <i>statementid</i> USING <i>param1</i> , <i>param2</i> ; OPEN cursor SCROLL WITH <i>statementid</i> ;
OPEN ... WITH REOPTIMIZE	OPEN /* REOPTIMIZE はサポートされていません */
PREPARE	PREPARE
PUT	EXECUTE
SET AUTOFREE	NO-OP
SQL ... END	execute #sql { ... }

## SQL データ定義ステートメント

表 A-15. 4GL SQL データ定義ステートメントを EGL ヘマップする方法

I4GL	EGL
ALTER INDEX	Execute #sql{ ... }
ALTER FRAGMENT	Execute #sql{alter fragment ... }
ALTER TABLE	Execute #sql{ ... }
CLOSE DATABASE	Execute #sql{ close database }
CREATE DATABASE	Execute #sql{ ... }
CREATE EXTERNAL TABLE	Execute #sql{ ... }
CREATE INDEX	Execute #sql{ ... }
CREATE PROCEDURE FORM	Execute #sql{ ... }
CREATE ROLE	Execute #sql{ ... }
CREATE SCHEMA	Execute #sql{ ... }
CREATE SYNONYM	Execute #sql{ ... }
CREATE TABLE	Execute #sql{ ... }
CREATE TRIGGER	Execute #sql{ ... }
CREATE VIEW	Execute #sql{ ... }
CONNECT	defineDatabaseHandle(getProperty(dbname)); connect(getProperty("DEFAULT_USER"), getProperty("DEFAULT_PASSWORD"), explicit, autoCommit);
DATABASE	
DROP DATABASE	Execute #sql{ ... }
DROP INDEX	Execute #sql{ ... }
DROP PROCEDURE	Execute #sql{ ... }
DROP ROLE	Execute #sql{ ... }

表 A-15. 4GL SQL データ定義ステートメントを EGL ヘマップする方法 (続き)

I4GL	EGL
DROP SYNONYM	Execute #sql{ ... }
DROP TABLE	Execute #sql{ ... }
DROP TRIGGER	Execute #sql{ ... }
DROP VIEW	Execute #sql{ ... }
RENAME COLUMN	Execute #sql{ ... }
RENAME DATABASE	Execute #sql{ ... }
RENAME TABLE	Execute #sql{ ... }

## SQL データ操作ステートメント

表 A-16. 4GL データ操作ステートメントを EGL ヘマップする方法

I4GL	EGL
INSERT	Execute #sql{ INSERT ... }
DELETE	Execute #sql{ DELETE ... }
LOAD	loadTable(filename, sql, delimiter);
OUTPUT	外部ツールによって間接的に
SELECT	Execute #sql{ select } into ...
UNLOAD	unloadTable(filename, sql, delimiter);
UPDATE	Execute #sql{ UPDATE ... }

## SQL 動的管理ステートメント

表 A-17. 4GL SQL 動的管理ステートメントを EGL ヘマップする方法

I4GL	EGL
EXECUTE	EXECUTE statement
EXECUTE IMMEDIATE	Execute #SQL{ ... }
FREE	FREE
PREPARE	PREPARE
SET DEFERRED_PREPARE	NO-OP

## SQL 照会最適化ステートメント

表 A-18. 4GL SQL 照会最適化ステートメントを EGL ヘマップする方法

I4GL	EGL
SET OPTIMIZATION	Execute #sql{set optimization ... }
SET EXPLAIN	Execute #sql{set explain ... }
SET PDQPRIORITY	Execute #sql{ ... }
SET RESIDENCY	Execute #sql{ ... }
SET SCHEDULE LEVEL	Execute #sql{ ... }

表 A-18. 4GL SQL 照会最適化ステートメントを EGL ヘマッピングする方法 (続き)

I4GL	EGL
UPDATE STATISTICS	Execute #sql{ ... }

## SQL データ・アクセス・ステートメント

表 A-19. 4GL SQL データ・アクセス・ステートメントを EGL ヘマッピングする方法

I4GL	EGL
GRANT	Execute #sql{ ... }
GRANT FRAGMENT	Execute #sql{ ... }
LOCK TABLE	Execute #sql{ ... }
REVOKE	Execute #sql{ ... }
REVOKE FRAGMENT	Execute #sql{ ... }
SET ISOLATION	Execute #sql{ ... };
SET LOCK MODE	Execute #sql{ ... }
SET ROLE	Execute #sql{ ... }
SET SESSION	Execute #sql{ ... }
SET TRANSACTION	Execute #sql{ ... }
UNLOCK TABLE	Execute #sql{ ... }

## SQL データ保全性ステートメント

表 A-20. 4GL SQL データ保全性ステートメントを EGL ヘマッピングする方法

I4GL	EGL
BEGIN WORK	<i>beginTransaction();</i>
COMMIT WORK	<i>commit();</i>
ROLLBACK WORK	<i>rollback();</i>
SET Database Object Mode	Execute #sql{... }
SET LOG	Execute #sql{... }
SET PLOAD FILE	Execute #sql{... }
SET TRANSACTION MODE	Execute #sql{... }
START VIOLATIONS TABLE	Execute #sql{... }
STOP VIOLATIONS TABLE	Execute #sql{... }

## SQL ストアド・プロシージャ・ステートメント

表 A-21. SQL ストアド・プロシージャ・ステートメントを EGL ヘマッピングする方法

I4GL	EGL
EXECUTE PROCEDURE	Execute #sql{ ... }
SET DEBUG FILE TO	Execute #sql{ ... }

---

## SQL クライアント/サーバー接続ステートメント

表 A-22. 4GL 接続ステートメントを EGL ヘマッピングする方法

I4GL	EGL
SET CONNECTION 'conname';	setCurrentDatabase("conname");
SET CONNECTION convar;	setCurrentDatabase (convar)
SET CONNECTION dbname;	setCurrentDatabase("dbname")
SET CONNECTION DEFAULT;	setCurrentDatabase ("DEFAULT");
SET CONNECTION ... DORMANT	// set connection ... dormant
CONNECT TO db@server	setDatabaseHandle(getProperty("db@server")); connect(... );
CONNECT TO db AS 'conname'	setDatabaseHandle(getProperty("db@server")); connect( ... )
CONNECT TO DEFAULT	try try disconnect(); onException end  try DefineDatabaseAlias( getProperty("@myserver"); getProperty("DEFAULT_USERID"), getProperty("DEFAULT_PASSWORD"), type1, explicit, repeatableRead, noAutoCommit); onException end onException end
DISCONNECT CURRENT	disconnect()
DISCONNECT DEFAULT	disconnect("DEFAULT");
DISCONNECT ALL	disconnectAll();
DISCONNECT 'conname'	disconnect("conname");
DISCONNECT convar	disconnect(convar);

---

## SQL オプティカル・サブシステム・ステートメント

表 A-23. SQL オプティカル・サブシステム・ステートメントを EGL ヘマッピングする方法

I4GL	EGL
ALTER OPTICAL CLUSTER	Execute #sql{ ... }
CREATE OPTICAL CLUSTER	Execute #sql{ ... }
DROP OPTICAL CLUSTER	Execute #sql{ ... }
RELEASE	Execute #sql{ ... }
RESERVE	Execute #sql{ ... }
SET MOUNTING TIMINOUT	Execute #sql{ ... }

## 環境変数

表 A-24. I4GL 環境変数を EGL および JDBC プロパティーへマップする方法

I4GL 環境変数	EGL プロパティー	JDBC プロパティー
C4GLFLAGS	NO-OP	NO-OP
C4GLNOPARAMCHK	NO-OP	NO-OP
CC	NO-OP	NO-OP
COLUMNS	NO-OP	NO-OP
CLIENT_LOCALE	CLIENT_LOCALE	CLIENT_LOCALE
COLLCHAR	NO-OP	NO-OP
DBANSIWARN	NO-OP	NO-OP
DBCENTURY	NO-OP	DBCENTURY
DBDATE	defaultDateFormat	DBDATE
DBDELIMITER	defaultDbDelimiterFormat	NO-OP
DBEDIT	NO-OP	NO-OP
DBESCWT	NO-OP	NO-OP
DBFORM	NO-OP	NO-OP
DBFORMAT	NO-OP	NO-OP
DBLANG	NO-OP	NO-OP
DBMONEY	defaultMoneyFormat defaultNumericFormat	NO-OP
DBPATH	NO-OP	NO-OP
DBPRINT	NO-OP	NO-OP
DBREMOTECMD	NO-OP	NO-OP
DBSPACETEMP	NO-OP	DBSPACETEMP
DBSRC	NO-OP	NO-OP
DBTEMP	NO-OP	DBTEMP
DBTIME	NO-OP	NO-OP
DBUPSPACE	NO-OP	DBUPSPACE
DBAPICODE	NO-OP	NO-OP
DB_LOCALE	DB_LOCALE	DB_LOCALE
DBNLS	NO-OP	NO-OP
ENVIGNORE	NO-OP	NO-OP
FET_BUF_SIZE	NO-OP	FET_BUF_SIZE
FGLPCFLAGS	NO-OP	NO-OP
FGLSKIPNXTPG	NO-OP	NO-OP
GL_DATE	defaultDateFormat	NO-OP
GL_DATETIME	defaultTimeStampFormat	NO-OP
INFORMIXC	NO-OP	NO-OP
INFORMIXONRETRY	NO-OP	NO-OP
INFORMIXCONTINUE	NO-OP	NO-OP
INFORMIXDIR	NO-OP	NO-OP

表 A-24. I4GL 環境変数を EGL および JDBC プロパティーマップする方法 (続き)

I4GL 環境変数	EGL プロパティ	JDBC プロパティ
INFORMIXSERVER	INFORMIXSERVER	INFORMIXSERVER
INFORMIXSHMBASE	NO-OP	NO-OP
INFORMIXTERM	NO-OP	NO-OP
IXOLDFLDScope	NO-OP	NO-OP
LANG	NO-OP	NO-OP
LINES	NO-OP	NO-OP
ONCONFIG	NO-OP	NO-OP
PATH	NO-OP	NO-OP
PDQPRIORITY	NO-OP	PDQPRIORITY
PROGRAM_DESIGN_DBS	NO-OP	NO-OP
PSORT_DBTEMP	NO-OP	PSORT_DBTEMP
PSORT_NPROCS	NO-OP	PSORT_NPROCS
SQLEXEC	NO-OP	NO-OP
SQLRM	NO-OP	NO-OP
SQLRMDIR	NO-OP	NO-OP
SUPOUTPIPEMSG	NO-OP	NO-OP
SERVER_LOCALE	NO-OP	NO-OP
TERM	NO-OP	NO-OP
TERMCAP	NO-OP	NO-OP
TERMINFO	NO-OP	NO-OP



---

## 付録 B. I4GL レポート変換コード例

---

### 本付録の内容

この付録では、I4GL レポート・コードの例および EGL ドライバー関数における同等の例を記載します。

---

### 4GL レポート・コード

```
REPORT r_invoice (c, x)
    DEFINE c RECORD
        customer_num int,
        fname         char(15),
        lname         char(15),
        company       char(20),
        address1      char(20),
        address2      char(20),
        city          char(15),
        state         char(2),
        zipcode       char(5),
        phone         char(18),
    END RECORD

    DEFINE x RECORD
        order_num INT,
        order_date      date,
        ship_instruct   Char(40),
        backlog         char,
        po_num          char(10),
        ship_date date,
        ship_weight     decimal(8,2),
        ship_charge     money(6,2),
        item_num smallint,
        stock_num smallint,
        manu_code char(3),
        quantity smallint,
        total_price     money(8,2),
        description     char(15),
        unit_price      money(6,2),
        unit            char(4),
        unit_descr      char(15),
        manu_name      char(10)
    END RECORD

...
...
...
END REPORT
```

---

### 4GL コードから生成された EGL ドライバー関数

```
Function r_invoice_START()
execute #sql {
    create temp table r_invoice_report_table (
        c_customer_num int ,
        c_fname char (15),
        c_lname char (15) ,
        c_company char (20) ,
        c_address1 char (20) ,
```

```

        c_address2 char (20) ,
        c_city char (15) ,
        c_state char (2) ,
        c_zipcode char (5) ,
        c_phone char (18) ,
        x_order_num int ,
        x_order_date date ,
        x_ship_instruct char (40) ,
        x_backlog char ,
        x_po_num char (10) ,
        x_ship_date date ,
        x_ship_weight decimal (8, 2),
        x_ship_charge_money (6, 2),
        x_item_num smallint ,
        x_stock_num smallint ,
        x_manu_code char (3) ,
        x_quantity smallint ,
        x_total_price money (6, 2),
        x_description char (15) ,
        x_unit_price money (6, 2),
        x_unit char (4) ,
        x_unit_descr char (15) ,
        x_manu_name char (10)
    )
}
end

Function r_invoice_OUTPUT (c recordtype_v4_c IN, x recordtype_v4_x IN)

    Prepare insert_stmt from "insert into r_invoice_report_table +
        "values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
    Execute insert_stmt using (
        c.customer_num,
        c.fname,
        c.lname,
        c.company,
        c.address1,
        c.address2,
        c.city,
        c.state,
        c.zipcode,
        c.phone,
        x.order_num,
        x.order_date,
        x.ship_instruct,
        x.backlog,
        x.po_num,
        x.ship_date,
        x.ship_weight,
        x.ship_charge,
        x.item_num,
        x.stock_num,
        x.manu_code,
        x.quantity,
        x.total_price,
        x.description,
        x.unit_price,
        x.unit,
        x.unit_descr,
        x.manu_name
    )

End

Function r_invoice_FINISH()
    egl4glReport      Report;
    egl4glReportData ReportData;

```

```

    egl4glReport.reportDesignFile      = "r_invoice_XML.jasper";
    egl4glReport.reportDestinationFile = "r_invoice.jrprint";
    egl4glReport.reportExportFile      = "r_invoice.txt";

    egl4glReportData.sqlStatement = "Select * From" +
                                     " r_invoice_report_table";

    egl4glReport.reportData = egl4glReportData;

    ReportLib.fillReport(egl4glReport, DATASOURCE_SQL_STATEMENT );
    ReportLib.exportReport(egl4glReport, EXPORT_TEXT);

    Execute #SQL {
        Drop Table r_invoice_report_table
    }
End

Function r_invoice_TERMINATE()
    Execute #SQL {
        Drop Table r_invoice_report_table
    }
End

...
...

Record recordtype_v4_c type BasicRecord
    customer_num INT;
    fname          UNICODE(15);
    lname          UNICODE(15);
    company         UNICODE(20);
    address1        UNICODE(20);
    address2        UNICODE(20);
    city            UNICODE(15);
    state           UNICODE(2);
    zipcode         UNICODE(5);
    phone           UNICODE(18);
END

RECORD recordtype_v4_x type BasicRecord
    order_num INT;
    order_date DATE;
    ship_instruct UNICODE(40);
    ship_instruct UNICODE(10);
    backlog       UNICODE(1);
    po_num        UNICODE(10);
    ship_date DATE;
    ship_weight   DECIMAL;
    ship_charge   DECIMAL(6,2);
    item_num      smallint;
    stock_num     smallint;
    manu_code     UNICODE(3);
    quantity      smallint;
    total_price   DECIMAL(8,2);
    description   UNICODE(15);
    unit_price    DECIMAL(6, 2);
    unit          UNICODE(4);
    unit_descr    UNICODE(15);
    manu_name     UNICODE(10);
END

```



---

## 付録 C. I4GL 書式コードから EGL 書式コードへの例

---

### 本付録の内容

この付録では、I4GL 書式コードおよび生成された EGL 書式コードの例を示しています。

---

### 4GL 書式コード

```
----- customer.4gl -----
GLOBALS
  DEFINE p_customer RECORD
    customer_num int,
    fname         char(15),
    lname         char(15),
    company       char(20),
    address1      char(20),
    address2      char(20),
    city          char(15),
    state         char(2),
    zipcode       char(5),
    phone         char(18)
  END RECORD
END GLOBALS

MAIN

  DISPLAY "Starting form Customer "
  SLEEP 2

  CALL input_cust()

END MAIN

FUNCTION input_cust()

  OPEN FORM customer FROM "customer"
  DISPLAY FORM customer
  ATTRIBUTE(BLUE)

  DISPLAY "Press ESC to enter new customer data" AT 1,1
  INPUT BY NAME p_customer.*
  AFTER FIELD state
    DISPLAY "In field state "
    DISPLAY "Press ESC to enter new customer data", "" AT 1,1
  ON KEY (F1, CONTROL-F)
    DISPLAY "Control-F pressed"
  ON KEY (F2, CONTROL-Y)
    LET int_flag = TRUE
    EXIT INPUT
  END INPUT
  IF int_flag
  THEN
    LET int_flag = FALSE
    RETURN(FALSE)
  END IF

  DISPLAY BY NAME p_customer.fname ATTRIBUTE(MAGENTA)

END FUNCTION
```

```

----- customer.per -----
DATABASE FORMONLY
SCREEN
{

                                Customer Form

    Number      :[f000          ]
    Owner Name   :[f001          ][f002          ]
    Company      :[f003          ]
    Address      :[f004          ]
                  [f005          ]
    City         :[f006          ] State:[a0] Zipcode:[f007 ]
    Telephone    :[f008          ]

}

ATTRIBUTES
f000 = FORMONLY.customer_num TYPE INT, NOENTRY;
f001 = FORMONLY.fname TYPE CHAR, UPSHIFT ;
f002 = FORMONLY.lname TYPE CHAR, COLOR=RED;
f003 = FORMONLY.company TYPE CHAR, COMMENTS="Company name",
COLOR=MAGENTA;
f004 = FORMONLY.address1 TYPE CHAR, AUTONEXT, COLOR=MAGENTA;
f005 = FORMONLY.address2 TYPE CHAR, COLOR=MAGENTA;
f006 = FORMONLY.city TYPE CHAR, REQUIRED;
a0 = FORMONLY.state TYPE CHAR, UPSHIFT;
f007 = FORMONLY.zipcode TYPE CHAR;
f008 = FORMONLY.phone TYPE CHAR, PICTURE = "###-###-#### XXXXX";

INSTRUCTIONS
SCREEN RECORD customer (FORMONLY.customer_num THRU FORMONLY.fname)

```

---

## EGL コード

```

-----customer.egl-----
//Program
LIBRARY customer{localSQLScope=YES}

use typeconv.typeconvConversionGlobals;

p_customer recordtype_p_customer;

FUNCTION $_customer_i4glmain()
displayLineMode( "Starting form Customer ");
wait( 2);

input_cust();
END

FUNCTION input_cust()
returns (Int)
$_FORM_customer customerForm{ name="$_FORM_customer" };
$__open_form_name= "$_FORM_customer";

$_open_form_name= "$_FORM_customer";
ConsoleLib.CurrentDisplayAttrs{ color=BLUE };
displayFormByName( $__open_form_name );
displayAtPosition( "Press ESC to enter new customer data" , 1,1);

OpenUI{setInitial=YES, bindingByName=YES} activeForm
bind p_customer.customer_num, p_customer.fname, p_customer.lname,

```

```

p_customer.company, p_customer.address1, p_customer.address2,
p_customer.city, p_customer.state, p_customer.zipcode,
p_customer.phone

onEvent(AFTER_FIELD:"state")
displayLineMode( "In field state ");
displayAtPosition( "Press ESC to enter new customer data" + " " , 1,1);

onEvent(ON_KEY:"F1","CONTROL_F")
displayLineMode( "Control-F pressed");

onEvent(ON_KEY:"F2","CONTROL_Y")
interruptRequested = YES;
EXIT OpenUI;
end;
IF ((YES == interruptRequested))
interruptRequested = NO;
return( (0/*FALSE*/) );
END

ConsoleLib.CurrentDisplayAttrs{ color=MAGENTA };
OpenUI { displayOnly=YES, bindingByName=YES } activeForm
bind p_customer.fname
End

END

END // PROGRAM

record recordtype_p_customer type SqlRecord
customer_num INT{isNullable=yes};
fname String(15){isNullable=yes};
lname String(15){isNullable=yes};
company String(20){isNullable=yes};
address1 String(20){isNullable=yes};
address2 String(20){isNullable=yes};
city String(15){isNullable=yes};
state String(2){isNullable=yes};
zipcode String(5){isNullable=yes};
phone String(18){isNullable=yes};
END

-----customerForm.egl-----

Record customerForm type ConsoleForm { formSize = [12,60],
showBrackets = yes }

*ConsoleField { position = [2,26], value = "Customer Form" };
*ConsoleField { position = [4,11], value = "Number :" };
*ConsoleField { position = [5,19], value = "Owner Name :" };
*ConsoleField { position = [6,19], value = "Company :" };
*ConsoleField { position = [7,19], value = "Address :" };
*ConsoleField { position = [9,19], value = "City :" };
*ConsoleField { position = [9,33], value = "State:" };
*ConsoleField { position = [9,44], value = "Zipcode:" };
*ConsoleField { position = [10,19], value = "Telephone :" };

customer_num ConsoleField { position = [4,20], fieldLen = 5,
dataType = "int", protect = yes, name="customer_num" };
fname ConsoleField { position = [5,32], fieldLen = 5,
dataType = "unicode", caseFormat = upper, name="fname" };
lname ConsoleField { position = [5,39], fieldLen = 5,
dataType = "unicode", color=RED, name="lname" };
company ConsoleField { position = [6,29], fieldLen = 5,
dataType = "unicode", comment = "Company name", color=MAGENTA,
name="company" };
address1 ConsoleField { position = [7,29], fieldLen = 5,

```

```

        dataType = "unicode", autonext = yes, color=MAGENTA,
        name="address1" };
address2 ConsoleField { position = [8,20], fieldLen = 5,
        dataType = "unicode", color=MAGENTA, name="address2" };
city ConsoleField { position = [9,26], fieldLen = 5,
        dataType = "unicode", inputRequired = yes, name="city" };
state ConsoleField { position = [9,40], fieldLen = 2,
        dataType = "unicode", caseFormat = upper, name="state" };
zipcode ConsoleField { position = [9,53], fieldLen = 5,
        dataType = "unicode", name="zipcode" };
phone ConsoleField { position = [10,31],fieldLen = 5, dataType
        = "unicode", pattern = "###-###-#### XXXXX", name="phone" };

customer Dictionary { customer_num=customer_num, fname=fname};

end

```

---

## 付録 D. 構成ファイルのテンプレート

---

### 本付録の内容

この付録では、データベース・スキーマ抽出、共用ライブラリー、および I4GL アプリケーション・プロジェクトのテンプレート構成ファイルについて説明します。

注: 構成ファイルを手動で作成する場合、XML パーサーが XML 予約文字と値を正しく区別できないような文字は使用しないでください。XML パーサーが誤った解釈をしないように、XML 要素の値は `<![CDATA[value]>` タグで組み込むようにしてください。

---

### データベース・スキーマ抽出プロジェクトの構成ファイル・テンプレート

```
<?xml version="1.0" encoding="utf-8"?>
<!--Internal DTD for Configuration file-->
<!DOCTYPE conversion [
  <!ELEMENT conversion (rootdir,dbconnection*)>
  <!ATTLIST conversion project CDATA #REQUIRED >
  <!ATTLIST conversion type CDATA #FIXED "schema">
  <!ELEMENT rootdir (egldir,artifactsdir?)>
  <!ELEMENT egldir (#PCDATA)>
  <!ELEMENT artifactsdir (#PCDATA)>
  <!ELEMENT dbconnection (database,server,host,port,user,password)+>
  <!ATTLIST dbconnection extractSystemTables (yes|no) "no" >
  <!ATTLIST dbconnection client_locale CDATA #IMPLIED >
  <!ATTLIST dbconnection db_locale CDATA #IMPLIED >
  <!ELEMENT database (#PCDATA)>
  <!ELEMENT server (#PCDATA)>
  <!ELEMENT host (#PCDATA)>
  <!ELEMENT port (#PCDATA)>
  <!ELEMENT user (#PCDATA)>
  <!ELEMENT password ANY>
]>

<conversion project="stores7" type="schema">

  <rootdir>
    <egldir>C:\%egl\%src%stores7</egldir>
    <artifactsdir>C:\%tmp%stores7\ConversionArtifacts</artifactsdir>
  </rootdir>

  <dbconnection extractSystemTables="no" client_locale="en_US.8859-1"
db_locale="en_US.8859-1">
    <database><![CDATA[stores7]]></database>
    <server><![CDATA[myserver]]></server>
    <host>mymachine.location.company.com</host>
    <port>1999</port>
    <user>jdoe</user>
    <password><![CDATA[password]]></password>
  </dbconnection>
</conversion >
```

---

## 共用ライブラリー・プロジェクトの構成ファイル・テンプレート

```
<?xml version="1.0" encoding="utf-8"?>
<!--Internal DTD for Configuration file-->
<!DOCTYPE conversion [
<!ELEMENT conversion (rootdir, manifestfiles*, fglfiles?,
    formfiles?, msgfiles*, fontconfigfile?)>
    <!--ATTLIST conversion project CDATA #REQUIRED -->
    <!--ATTLIST conversion type CDATA #FIXED "library" -->
    <!--ATTLIST conversion locale CDATA #IMPLIED -->
    <!--ATTLIST conversion cursor (local | global) #IMPLIED-->
    <!--ATTLIST conversion defaultserver CDATA #IMPLIED-->
<!ELEMENT rootdir (fgldir?,egldir,artifactsdir?)>
    <!--ELEMENT fgldir (#PCDATA)-->
<!--ELEMENT egldir (#PCDATA)-->
<!--ELEMENT artifactsdir (#PCDATA)-->
<!--ELEMENT manifestfiles (file)+-->
    <!--ATTLIST manifestfiles type ( schema | library) #REQUIRED-->
<!--ELEMENT fglfiles (file)*-->
<!--ELEMENT formfiles (file)*-->
<!--ELEMENT fontconfigfile (file)*-->
    <!--ELEMENT file (#PCDATA)-->
<!--ELEMENT msgfiles (file)+-->
<!--ATTLIST msgfiles locale CDATA #IMPLIED -->
]>
<conversion project="SharedLibraryProject" type="library"
    locale="en_US.8859-1" cursor="local" defaultserver="demo_on">

    <rootdir>
    <fgldir>C:\i4gl\SharedLibraryProject\fgl\fgldir>
    <egldir>C:\i4gl\SharedLibraryProject\egl\egldir>
    <artifactsdir>C:\ConversionArtifacts\artifactsdir>
    </rootdir>

    <manifestfiles type="schema">
    <file>C:\ConversionArtifacts\manifest\Stores7Manifest.xml</file>
    </manifestfiles>

    <fglfiles>
    <file>fgl\d4_cust.4gl</file>
    <file>fgl\d4_demo.4gl</file>
    <file>fgl\d4_globals.4gl</file>
    </fglfiles>

    <!--report font file, optional-->
    <fontconfigfile>
    <file><![CDATA[C:\Documents and Settings\myfont.xml]]</file>
    </fontconfigfile>
    </conversion>
```

---

## アプリケーション・プロジェクトの構成ファイル・テンプレート

```
<?xml version="1.0" encoding="utf-8"?>
<!--Internal DTD for Configuration file-->
<!DOCTYPE conversion [
<!--ELEMENT conversion ( rootdir, manifestfiles*, fglfiles?,
    formfiles?, msgfiles* , fontconfigfile? )-->
    <!--ATTLIST conversion project CDATA #REQUIRED -->
    <!--ATTLIST conversion type CDATA #FIXED "application" -->
    <!--ATTLIST conversion locale CDATA #IMPLIED -->
    <!--ATTLIST conversion cursor (local | global) #IMPLIED-->
    <!--ATTLIST conversion defaultserver CDATA #IMPLIED-->
<!--ELEMENT rootdir (fgldir?,egldir,artifactsdir?)-->
    <!--ELEMENT fgldir (#PCDATA)-->
```

```

<!ELEMENT egldir (#PCDATA)>
<!ELEMENT artifactsdir (#PCDATA)>
<!ELEMENT manifestfiles (file)+>
  <!ATTLIST manifestfiles type ( schema | library) #REQUIRED>
<!ELEMENT fglfiles (file)*>
<!ELEMENT formfiles (file)*>
<!ELEMENT fontconfigfile (file)*>
  <!ELEMENT file (#PCDATA)>
<!ELEMENT msgfiles (file)+>
<!ATTLIST msgfiles locale CDATA #IMPLIED >
]>

  <!--Sample Conversion configuration file for i4gldemo application conversion-->
<conversion project="i4gldemo" type="application"
  locale="en_US.8859-1" defaultserver="egltest">
<rootdir>
<fgldir>C:\i4gl\src\i4gldemo</fgldir>
<egldir>C:\egl\src\i4gldemo</egldir>
<artifactsdir></artifactsdir>
</rootdir>

<manifestfiles type="schema">
  <file>C:\ConversionArtifacts\manifest\Stores7Manifest.xml</file>
<fglfiles>
<file>d4_cust.4gl</file>
<file>d4_demo.4gl</file>
<file>d4_globals.4gl</file>
<file>d4_load.4gl</file>
<file>d4_main.4gl</file>
<file>d4_orders.4gl</file>
<file>d4_report.4gl</file>
<file>d4_stock.4gl</file>
</fglfiles>
<formfiles>
<file>forms\cust.per</file>
<file>forms\custcur.per</file>
<file>forms\custform.per</file>
<file>forms\customer.per</file>
<file>forms\ordcur.per</file>
<file>forms\order.per</file>
<file>forms\orderform.per</file>
<file>forms\p_ordcur.per</file>
<file>forms\state_list.per</file>
<file>forms\stockl.per</file>
<file>forms\stock_sel.per</file>
</formfiles>

<!--example for converting UJIS message files-->
<msgfiles locale="ja_jp.UJIS">
<file>C:\i4gl\src\msg\jp\ujis/orders.msg</file>
<file>C:\i4gl\src\msg\jp\ujis/customer.msg</file>
</msgfile>
</conversion>

```





```

    fgltype="char" egltype="unicode(32)" size="32" />
<column name="aggid" dataitem="dataitem_like_sysaggregates_aggid"
    fgltype="serial" egltype="int" />
<column name="init_func"
    dataitem="dataitem_like_sysaggregates_init_func"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="iter_func"
    dataitem="dataitem_like_sysaggregates_iter_func"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="combine_func" dataitem
    ="dataitem_like_sysaggregates_combine_func"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="final_func" dataitem="dataitem_like_sysaggregates_final_func"
    fgltype="varchar" egltype="string(128)" size="128" />
</table>
<table name="sysams" egltype="rec like_sysams">
<column name="am_name" dataitem="dataitem_like_sysams_am_name"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="am_owner" dataitem="dataitem_like_sysams_am_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="am_id" dataitem="dataitem_like_sysams_am_id"
    fgltype="int" egltype="int" />
<column name="am_type" dataitem="dataitem_like_sysams_am_type"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="am_sptype" dataitem="dataitem_like_sysams_am_sptype"
    fgltype="char" egltype="unicode(3)" size="3" />
<column name="am_defopclass" dataitem
    ="dataitem_like_sysams_am_defopclass"
    fgltype="int" egltype="int" />
<column name="am_keyscan" dataitem="dataitem_like_sysams_am_keyscan"
    fgltype="int" egltype="int" />
<column name="am_unique" dataitem="dataitem_like_sysams_am_unique"
    fgltype="int" egltype="int" />
<column name="am_cluster" dataitem="dataitem_like_sysams_am_cluster"
    fgltype="int" egltype="int" />
<column name="am_rowids" dataitem="dataitem_like_sysams_am_rowids"
    fgltype="int" egltype="int" />
<column name="am_readwrite" dataitem="dataitem_like_sysams_am_readwrite"
    fgltype="int" egltype="int" />
<column name="am_parallel" dataitem="dataitem_like_sysams_am_parallel"
    fgltype="int" egltype="int" />
<column name="am_costfactor" dataitem
    ="dataitem_like_sysams_am_costfactor"
    fgltype="smallfloat" egltype="smallfloat" />
<column name="am_create" dataitem="dataitem_like_sysams_am_create"
    fgltype="int" egltype="int" />
<column name="am_drop" dataitem="dataitem_like_sysams_am_drop"
    fgltype="int" egltype="int" />
<column name="am_open" dataitem="dataitem_like_sysams_am_open"
    fgltype="int" egltype="int" />
<column name="am_close" dataitem="dataitem_like_sysams_am_close"
    fgltype="int" egltype="int" />
<column name="am_insert" dataitem="dataitem_like_sysams_am_insert"
    fgltype="int" egltype="int" />
<column name="am_delete" dataitem="dataitem_like_sysams_am_delete"
    fgltype="int" egltype="int" />
<column name="am_update" dataitem="dataitem_like_sysams_am_update"
    fgltype="int" egltype="int" />
<column name="am_stats" dataitem="dataitem_like_sysams_am_stats"
    fgltype="int" egltype="int" />
<column name="am_scancost" dataitem="dataitem_like_sysams_am_scancost"
    fgltype="int" egltype="int" />
<column name="am_check" dataitem="dataitem_like_sysams_am_check"
    fgltype="int" egltype="int" />
<column name="am_beginscan" dataitem="dataitem_like_sysams_am_beginscan"
    fgltype="int" egltype="int" />
<column name="am_endscan" dataitem="dataitem_like_sysams_am_endscan"

```

```

    fgltype="int" egltype="int" />
<column name="am_rescan" dataitem="dataitem_like_sysams_am_rescan"
    fgltype="int" egltype="int" />
<column name="am_getnext" dataitem="dataitem_like_sysams_am_getnext"
    fgltype="int" egltype="int" />
<column name="am_getbyid" dataitem="dataitem_like_sysams_am_getbyid"
    fgltype="int" egltype="int" />
<column name="am_build" dataitem="dataitem_like_sysams_am_build"
    fgltype="int" egltype="int" />
<column name="am_init" dataitem="dataitem_like_sysams_am_init"
    fgltype="int" egltype="int" />
<column name="am_truncate" dataitem="dataitem_like_sysams_am_truncate"
    fgltype="int" egltype="int" />
</table>
<table name="sysattrtypes" egltype="rec_like_sysattrtypes">
<column name="extended_id" dataitem
    ="dataitem_like_sysattrtypes_extended_id"
    fgltype="int" egltype="int" />
<column name="seqno" dataitem="dataitem_like_sysattrtypes_seqno"
    fgltype="smallint" egltype="smallint" />
<column name="levelno" dataitem="dataitem_like_sysattrtypes_levelno"
    fgltype="smallint" egltype="smallint" />
<column name="parent_no" dataitem
    ="dataitem_like_sysattrtypes_parent_no"
    fgltype="smallint" egltype="smallint" />
<column name="fieldname" dataitem
    ="dataitem_like_sysattrtypes_fieldname"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="fieldno" dataitem="dataitem_like_sysattrtypes_fieldno"
    fgltype="smallint" egltype="smallint" />
<column name="type" dataitem="dataitem_like_sysattrtypes_type"
    fgltype="smallint" egltype="smallint" />
<column name="length" dataitem="dataitem_like_sysattrtypes_length"
    fgltype="smallint" egltype="smallint" />
<column name="xtd_type_id" dataitem
    ="dataitem_like_sysattrtypes_xtd_type_id"
    fgltype="int" egltype="int" />
</table>
<table name="sysblobs" egltype="rec_like_sysblobs">
<column name="spacename" dataitem="dataitem_like_sysblobs_spacename"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="type" dataitem="dataitem_like_sysblobs_type"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="tabid" dataitem="dataitem_like_sysblobs_tabid"
    fgltype="int" egltype="int" />
<column name="colno" dataitem="dataitem_like_sysblobs_colno"
    fgltype="smallint" egltype="smallint" />
</table>
<table name="syscasts" egltype="rec_like_syscasts">
<column name="owner" dataitem="dataitem_like_syscasts_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="argument_type" dataitem
    ="dataitem_like_syscasts_argument_type"
    fgltype="smallint" egltype="smallint" />
<column name="argument_xid" dataitem
    ="dataitem_like_syscasts_argument_xid"
    fgltype="int" egltype="int" />
<column name="result_type" dataitem="dataitem_like_syscasts_result_type"
    fgltype="smallint" egltype="smallint" />
<column name="result_xid" dataitem="dataitem_like_syscasts_result_xid"
    fgltype="int" egltype="int" />
<column name="routine_name" dataitem
    ="dataitem_like_syscasts_routine_name"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="routine_owner" dataitem
    ="dataitem_like_syscasts_routine_owner"
    fgltype="char" egltype="unicode(32)" size="32" />

```

```

    <column name ="class" dataitem ="dataitem_like_syscasts_class"
      fgltype ="char" egltype ="unicode(1)" size ="1" />
  </table>
  <table name ="syschecks" egltype ="rec_like_syschecks">
    <column name ="constrid" dataitem ="dataitem_like_syschecks_constrid"
      fgltype ="int" egltype ="int" />
    <column name ="type" dataitem ="dataitem_like_syschecks_type"
      fgltype ="char" egltype ="unicode(1)" size ="1" />
    <column name ="seqno" dataitem ="dataitem_like_syschecks_seqno"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="checktext" dataitem ="dataitem_like_syschecks_checktext"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
  </table>
  <table name ="syscolattrs" egltype ="rec_like_syscolattrs">
    <column name ="tabid" dataitem ="dataitem_like_syscolattrs_tabid"
      fgltype ="int" egltype ="int" />
    <column name ="colno" dataitem ="dataitem_like_syscolattrs_colno"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="extentsize" dataitem
      ="dataitem_like_syscolattrs_extentsize"
      fgltype ="int" egltype ="int" />
    <column name ="flags" dataitem ="dataitem_like_syscolattrs_flags"
      fgltype ="int" egltype ="int" />
    <column name ="flags1" dataitem ="dataitem_like_syscolattrs_flags1"
      fgltype ="int" egltype ="int" />
    <column name ="sbspace" dataitem ="dataitem_like_syscolattrs_sbspace"
      fgltype ="varchar" egltype ="string(128)" size ="128" />
  </table>
  <table name ="syscolauth" egltype ="rec_like_syscolauth">
    <column name ="grantor" dataitem ="dataitem_like_syscolauth_grantor"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="grantee" dataitem ="dataitem_like_syscolauth_grantee"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="tabid" dataitem ="dataitem_like_syscolauth_tabid"
      fgltype ="int" egltype ="int" />
    <column name ="colno" dataitem ="dataitem_like_syscolauth_colno"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="colauth" dataitem ="dataitem_like_syscolauth_colauth"
      fgltype ="char" egltype ="unicode(3)" size ="3" />
  </table>
  <table name ="syscoldepend" egltype ="rec_like_syscoldepend">
    <column name ="constrid" dataitem ="dataitem_like_syscoldepend_constrid"
      fgltype ="int" egltype ="int" />
    <column name ="tabid" dataitem ="dataitem_like_syscoldepend_tabid"
      fgltype ="int" egltype ="int" />
    <column name ="colno" dataitem ="dataitem_like_syscoldepend_colno"
      fgltype ="smallint" egltype ="smallint" />
  </table>
  <table name ="syscolumns" egltype ="rec_like_syscolumns">
    <column name ="colname" dataitem ="dataitem_like_syscolumns_colname"
      fgltype ="varchar" egltype ="string(128)" size ="128" />
    <column name ="tabid" dataitem ="dataitem_like_syscolumns_tabid"
      fgltype ="int" egltype ="int" />
    <column name ="colno" dataitem ="dataitem_like_syscolumns_colno"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="coltype" dataitem ="dataitem_like_syscolumns_coltype"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="collength" dataitem ="dataitem_like_syscolumns_collength"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="colmin" dataitem ="dataitem_like_syscolumns_colmin"
      fgltype ="int" egltype ="int" />
    <column name ="colmax" dataitem ="dataitem_like_syscolumns_colmax"
      fgltype ="int" egltype ="int" />
    <column name ="extended_id" dataitem
      ="dataitem_like_syscolumns_extended_id"
      fgltype ="int" egltype ="int" />
  </table>

```

```

<table name ="sysconstraints" egltype ="rec_like_sysconstraints">
  <column name ="constrid" dataitem ="dataitem_like_sysconstraints_constrid"
    fgltype ="serial" egltype ="int" />
  <column name ="constrname" dataitem
    ="dataitem_like_sysconstraints_constrname"
    fgltype ="varchar" egltype ="string(128)" size ="128" />
  <column name ="owner" dataitem ="dataitem_like_sysconstraints_owner"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="tabid" dataitem ="dataitem_like_sysconstraints_tabid"
    fgltype ="int" egltype ="int" />
  <column name ="constrtype" dataitem
    ="dataitem_like_sysconstraints_constrtype"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="idxname" dataitem ="dataitem_like_sysconstraints_idxname"
    fgltype ="varchar" egltype ="string(128)" size ="128" />
  <column name ="collation" dataitem ="dataitem_like_sysconstraints_collation"
    fgltype ="char" egltype ="unicode(36)" size ="36" />
</table>
<table name ="sysdefaults" egltype ="rec_like_sysdefaults">
  <column name ="tabid" dataitem ="dataitem_like_sysdefaults_tabid"
    fgltype ="int" egltype ="int" />
  <column name ="colno" dataitem ="dataitem_like_sysdefaults_colno"
    fgltype ="smallint" egltype ="smallint" />
  <column name ="type" dataitem ="dataitem_like_sysdefaults_type"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="default" dataitem ="dataitem_like_sysdefaults_default"
    fgltype ="char" egltype ="unicode(256)" size ="256" />
  <column name ="class" dataitem ="dataitem_like_sysdefaults_class"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
</table>
<table name ="sysdepend" egltype ="rec_like_sysdepend">
  <column name ="btbid" dataitem ="dataitem_like_sysdepend_btbid"
    fgltype ="int" egltype ="int" />
  <column name ="btype" dataitem ="dataitem_like_sysdepend_btype"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="dtbid" dataitem ="dataitem_like_sysdepend_dtbid"
    fgltype ="int" egltype ="int" />
  <column name ="dtype" dataitem ="dataitem_like_sysdepend_dtype"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
</table>
<table name ="sysdirectives" egltype ="rec_like_sysdirectives">
  <column name ="id" dataitem ="dataitem_like_sysdirectives_id"
    fgltype ="serial" egltype ="int" />
  <column name ="query" dataitem ="dataitem_like_sysdirectives_query"
    fgltype ="text" egltype ="clob" />
  <column name ="directive" dataitem ="dataitem_like_sysdirectives_directive"
    fgltype ="text" egltype ="clob" />
  <column name ="directivecode" dataitem
    ="dataitem_like_sysdirectives_directivecode"
    fgltype ="byte" egltype ="blob" />
  <column name ="active" dataitem ="dataitem_like_sysdirectives_active"
    fgltype ="smallint" egltype ="smallint" />
  <column name ="hashcode" dataitem ="dataitem_like_sysdirectives_hashcode"
    fgltype ="int" egltype ="int" />
</table>
<table name ="sysdistrib" egltype ="rec_like_sysdistrib">
  <column name ="tabid" dataitem ="dataitem_like_sysdistrib_tabid"
    fgltype ="int" egltype ="int" />
  <column name ="colno" dataitem ="dataitem_like_sysdistrib_colno"
    fgltype ="smallint" egltype ="smallint" />
  <column name ="seqno" dataitem ="dataitem_like_sysdistrib_seqno"
    fgltype ="int" egltype ="int" />
  <column name ="constructed" dataitem ="dataitem_like_sysdistrib_constructed"
    fgltype ="date" egltype ="date" />
  <column name ="mode" dataitem ="dataitem_like_sysdistrib_mode"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="resolution" dataitem ="dataitem_like_sysdistrib_resolution"

```

```

    fgltype="smallfloat" egltype="smallfloat" />
<column name="confidence" dataitem="dataitem_like_sysdistrib_confidence"
    fgltype="smallfloat" egltype="smallfloat" />
<column name="encdat" dataitem="dataitem_like_sysdistrib_encdat"
    fgltype="stat" egltype="string(0)" size="0" />
<column name="type" dataitem="dataitem_like_sysdistrib_type"
    fgltype="char" egltype="unicode(1)" size="1" />
</table>
<table name="syserrors" egltype="rec_like_syserrors">
<column name="sqlstate" dataitem="dataitem_like_syserrors_sqlstate"
    fgltype="char" egltype="unicode(5)" size="5" />
<column name="locale" dataitem="dataitem_like_syserrors_locale"
    fgltype="char" egltype="unicode(36)" size="36" />
<column name="level" dataitem="dataitem_like_syserrors_level"
    fgltype="smallint" egltype="smallint" />
<column name="seqno" dataitem="dataitem_like_syserrors_seqno"
    fgltype="smallint" egltype="smallint" />
<column name="message" dataitem="dataitem_like_syserrors_message"
    fgltype="varchar" egltype="string(255)" size="255" />
</table>
<table name="sysfragauth" egltype="rec_like_sysfragauth">
<column name="grantor" dataitem="dataitem_like_sysfragauth_grantor"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="grantee" dataitem="dataitem_like_sysfragauth_grantee"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="tabid" dataitem="dataitem_like_sysfragauth_tabid"
    fgltype="int" egltype="int" />
<column name="fragment" dataitem="dataitem_like_sysfragauth_fragment"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="fragauth" dataitem="dataitem_like_sysfragauth_fragauth"
    fgltype="char" egltype="unicode(6)" size="6" />
</table>
<table name="sysfragments" egltype="rec_like_sysfragments">
<column name="fragtype" dataitem="dataitem_like_sysfragments_fragtype"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="tabid" dataitem="dataitem_like_sysfragments_tabid"
    fgltype="int" egltype="int" />
<column name="indexname" dataitem="dataitem_like_sysfragments_indexname"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="colno" dataitem="dataitem_like_sysfragments_colno"
    fgltype="int" egltype="int" />
<column name="partn" dataitem="dataitem_like_sysfragments_partn"
    fgltype="int" egltype="int" />
<column name="strategy" dataitem="dataitem_like_sysfragments_strategy"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="location" dataitem="dataitem_like_sysfragments_location"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="servername" dataitem="dataitem_like_sysfragments_servername"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="evalpos" dataitem="dataitem_like_sysfragments_evalpos"
    fgltype="int" egltype="int" />
<column name="exprrtext" dataitem="dataitem_like_sysfragments_exprrtext"
    fgltype="text" egltype="clob" />
<column name="exprbin" dataitem="dataitem_like_sysfragments_exprbin"
    fgltype="byte" egltype="blob" />
<column name="exprarr" dataitem="dataitem_like_sysfragments_exprarr"
    fgltype="byte" egltype="blob" />
<column name="flags" dataitem="dataitem_like_sysfragments_flags"
    fgltype="int" egltype="int" />
<column name="dbspace" dataitem="dataitem_like_sysfragments_dbspace"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="levels" dataitem="dataitem_like_sysfragments_levels"
    fgltype="smallint" egltype="smallint" />
<column name="npused" dataitem="dataitem_like_sysfragments_npused"
    fgltype="int" egltype="int" />
<column name="nrows" dataitem="dataitem_like_sysfragments_nrows"
    fgltype="int" egltype="int" />

```

```

<column name ="clust" dataitem ="dataitem_like_sysfragments_clust"
  fgltype ="int" egltype ="int" />
<column name ="partition" dataitem ="dataitem_like_sysfragments_partition"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
</table>
<table name ="sysindices" egltype ="rec_like_sysindices">
  <column name ="idxname" dataitem ="dataitem_like_sysindices_idxname"
    fgltype ="varchar" egltype ="string(128)" size ="128" />
  <column name ="owner" dataitem ="dataitem_like_sysindices_owner"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="tabid" dataitem ="dataitem_like_sysindices_tabid"
    fgltype ="int" egltype ="int" />
  <column name ="idxtype" dataitem ="dataitem_like_sysindices_idxtype"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="clustered" dataitem ="dataitem_like_sysindices_clustered"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="levels" dataitem ="dataitem_like_sysindices_levels"
    fgltype ="smallint" egltype ="smallint" />
  <column name ="leaves" dataitem ="dataitem_like_sysindices_leaves"
    fgltype ="int" egltype ="int" />
  <column name ="nunique" dataitem ="dataitem_like_sysindices_nunique"
    fgltype ="int" egltype ="int" />
  <column name ="clust" dataitem ="dataitem_like_sysindices_clust"
    fgltype ="int" egltype ="int" />
  <column name ="nrows" dataitem ="dataitem_like_sysindices_nrows"
    fgltype ="float" egltype ="float" />
  <column name ="indexkeys" dataitem ="dataitem_like_sysindices_indexkeys"
    fgltype ="indexkeyarray" egltype ="string(0)" size ="0" />
  <column name ="amid" dataitem ="dataitem_like_sysindices_amid"
    fgltype ="int" egltype ="int" />
  <column name ="amparam" dataitem ="dataitem_like_sysindices_amparam"
    fgltype ="lvarchar" egltype ="string" size ="2048" />
  <column name ="collation" dataitem ="dataitem_like_sysindices_collation"
    fgltype ="char" egltype ="unicode(36)" size ="36" />
  <column name ="pagesize" dataitem ="dataitem_like_sysindices_pagesize"
    fgltype ="int" egltype ="int" />
</table>
<table name ="sysinherits" egltype ="rec_like_sysinherits">
  <column name ="child" dataitem ="dataitem_like_sysinherits_child"
    fgltype ="int" egltype ="int" />
  <column name ="parent" dataitem ="dataitem_like_sysinherits_parent"
    fgltype ="int" egltype ="int" />
  <column name ="class" dataitem ="dataitem_like_sysinherits_class"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
</table>
<table name ="syslangauth" egltype ="rec_like_syslangauth">
  <column name ="grantor" dataitem ="dataitem_like_syslangauth_grantor"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="grantee" dataitem ="dataitem_like_syslangauth_grantee"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="langid" dataitem ="dataitem_like_syslangauth_langid"
    fgltype ="int" egltype ="int" />
  <column name ="langauth" dataitem ="dataitem_like_syslangauth_langauth"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
</table>
<table name ="syslogmap" egltype ="rec_like_syslogmap">
  <column name ="tabloc" dataitem ="dataitem_like_syslogmap_tabloc"
    fgltype ="int" egltype ="int" />
  <column name ="tabid" dataitem ="dataitem_like_syslogmap_tabid"
    fgltype ="int" egltype ="int" />
  <column name ="fragid" dataitem ="dataitem_like_syslogmap_fragid"
    fgltype ="int" egltype ="int" />
  <column name ="flags" dataitem ="dataitem_like_syslogmap_flags"
    fgltype ="int" egltype ="int" />
</table>
<table name ="sysobjstate" egltype ="rec_like_sysobjstate">
  <column name ="objtype" dataitem ="dataitem_like_sysobjstate_objtype"

```

```

    fgltype="char" egltype="unicode(1)" size="1" />
<column name="owner" dataitem="dataitem_like_sysobjstate_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="name" dataitem="dataitem_like_sysobjstate_name"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="tabid" dataitem="dataitem_like_sysobjstate_tabid"
    fgltype="int" egltype="int" />
<column name="state" dataitem="dataitem_like_sysobjstate_state"
    fgltype="char" egltype="unicode(1)" size="1" />
</table>
<table name="sysopclasses" egltype="rec_like_sysopclasses">
<column name="opclassname" dataitem="dataitem_like_sysopclasses_opclassname"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="owner" dataitem="dataitem_like_sysopclasses_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="amid" dataitem="dataitem_like_sysopclasses_amid"
    fgltype="int" egltype="int" />
<column name="opclassid" dataitem="dataitem_like_sysopclasses_opclassid"
    fgltype="serial" egltype="int" />
<column name="ops" dataitem="dataitem_like_sysopclasses_ops"
    fgltype="lvarchar" egltype="string" size="2048" />
<column name="support" dataitem="dataitem_like_sysopclasses_support"
    fgltype="lvarchar" egltype="string" size="2048" />
</table>
<table name="sysopclstr" egltype="rec_like_sysopclstr">
<column name="owner" dataitem="dataitem_like_sysopclstr_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
<column name="clstrname" dataitem="dataitem_like_sysopclstr_clstrname"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="clstrsize" dataitem="dataitem_like_sysopclstr_clstrsize"
    fgltype="int" egltype="int" />
<column name="tabid" dataitem="dataitem_like_sysopclstr_tabid"
    fgltype="int" egltype="int" />
<column name="blobcol1" dataitem="dataitem_like_sysopclstr_blobcol1"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol2" dataitem="dataitem_like_sysopclstr_blobcol2"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol3" dataitem="dataitem_like_sysopclstr_blobcol3"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol4" dataitem="dataitem_like_sysopclstr_blobcol4"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol5" dataitem="dataitem_like_sysopclstr_blobcol5"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol6" dataitem="dataitem_like_sysopclstr_blobcol6"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol7" dataitem="dataitem_like_sysopclstr_blobcol7"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol8" dataitem="dataitem_like_sysopclstr_blobcol8"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol9" dataitem="dataitem_like_sysopclstr_blobcol9"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol10" dataitem="dataitem_like_sysopclstr_blobcol10"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol11" dataitem="dataitem_like_sysopclstr_blobcol11"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol12" dataitem="dataitem_like_sysopclstr_blobcol12"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol13" dataitem="dataitem_like_sysopclstr_blobcol13"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol14" dataitem="dataitem_like_sysopclstr_blobcol14"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol15" dataitem="dataitem_like_sysopclstr_blobcol15"
    fgltype="smallint" egltype="smallint" />
<column name="blobcol16" dataitem="dataitem_like_sysopclstr_blobcol16"
    fgltype="smallint" egltype="smallint" />
<column name="clstrkey1" dataitem="dataitem_like_sysopclstr_clstrkey1"
    fgltype="smallint" egltype="smallint" />

```

```

<column name ="clstrkey2" dataitem ="dataitem_like_sysopclstr_clstrkey2"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey3" dataitem ="dataitem_like_sysopclstr_clstrkey3"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey4" dataitem ="dataitem_like_sysopclstr_clstrkey4"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey5" dataitem ="dataitem_like_sysopclstr_clstrkey5"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey6" dataitem ="dataitem_like_sysopclstr_clstrkey6"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey7" dataitem ="dataitem_like_sysopclstr_clstrkey7"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey8" dataitem ="dataitem_like_sysopclstr_clstrkey8"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey9" dataitem ="dataitem_like_sysopclstr_clstrkey9"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey10" dataitem ="dataitem_like_sysopclstr_clstrkey10"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey11" dataitem ="dataitem_like_sysopclstr_clstrkey11"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey12" dataitem ="dataitem_like_sysopclstr_clstrkey12"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey13" dataitem ="dataitem_like_sysopclstr_clstrkey13"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey14" dataitem ="dataitem_like_sysopclstr_clstrkey14"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey15" dataitem ="dataitem_like_sysopclstr_clstrkey15"
  fgltype ="smallint" egltype ="smallint" />
<column name ="clstrkey16" dataitem ="dataitem_like_sysopclstr_clstrkey16"
  fgltype ="smallint" egltype ="smallint" />
</table>
<table name ="sysprocauth" egltype ="rec_like_sysprocauth">
  <column name ="grantor" dataitem ="dataitem_like_sysprocauth_grantor"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="grantee" dataitem ="dataitem_like_sysprocauth_grantee"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="procid" dataitem ="dataitem_like_sysprocauth_procid"
    fgltype ="int" egltype ="int" />
  <column name ="procauth" dataitem ="dataitem_like_sysprocauth_procauth"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
</table>
<table name ="sysprocbody" egltype ="rec_like_sysprocbody">
  <column name ="procid" dataitem ="dataitem_like_sysprocbody_procid"
    fgltype ="int" egltype ="int" />
  <column name ="datakey" dataitem ="dataitem_like_sysprocbody_datakey"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="seqno" dataitem ="dataitem_like_sysprocbody_seqno"
    fgltype ="int" egltype ="int" />
  <column name ="data" dataitem ="dataitem_like_sysprocbody_data"
    fgltype ="char" egltype ="unicode(256)" size ="256" />
</table>
<table name ="sysprocedures" egltype ="rec_like_sysprocedures">
  <column name ="procname" dataitem ="dataitem_like_sysprocedures_procname"
    fgltype ="varchar" egltype ="string(128)" size ="128" />
  <column name ="owner" dataitem ="dataitem_like_sysprocedures_owner"
    fgltype ="char" egltype ="unicode(32)" size ="32" />
  <column name ="procid" dataitem ="dataitem_like_sysprocedures_procid"
    fgltype ="serial" egltype ="int" />
  <column name ="mode" dataitem ="dataitem_like_sysprocedures_mode"
    fgltype ="char" egltype ="unicode(1)" size ="1" />
  <column name ="retsize" dataitem ="dataitem_like_sysprocedures_retsize"
    fgltype ="int" egltype ="int" />
  <column name ="symsize" dataitem ="dataitem_like_sysprocedures_symsize"
    fgltype ="int" egltype ="int" />
  <column name ="datasize" dataitem ="dataitem_like_sysprocedures_datasize"
    fgltype ="int" egltype ="int" />
  <column name ="codesize" dataitem ="dataitem_like_sysprocedures_codesize"

```

```

    fgltype="int" egltype="int" />
<column name="numargs" dataitem="dataitem_like_sysprocedures_numargs"
    fgltype="int" egltype="int" />
<column name="isproc" dataitem="dataitem_like_sysprocedures_isproc"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="specificname" dataitem
    ="dataitem_like_sysprocedures_specificname"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="externalname" dataitem
    ="dataitem_like_sysprocedures_externalname"
    fgltype="varchar" egltype="string(255)" size="255" />
<column name="paramstyle" dataitem
    ="dataitem_like_sysprocedures_paramstyle"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="langid" dataitem="dataitem_like_sysprocedures_langid"
    fgltype="int" egltype="int" />
<column name="paramtypes" dataitem
    ="dataitem_like_sysprocedures_paramtypes"
    fgltype="rtnparamtypes" egltype="string(0)" size="0" />
<column name="percallcost" dataitem
    ="dataitem_like_sysprocedures_perccallcost"
    fgltype="int" egltype="int" />
<column name="commutator" dataitem="dataitem_like_sysprocedures_commutator"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="negator" dataitem="dataitem_like_sysprocedures_negator"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="selfunc" dataitem="dataitem_like_sysprocedures_selfunc"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="class" dataitem="dataitem_like_sysprocedures_class"
    fgltype="char" egltype="unicode(18)" size="18" />
<column name="stack" dataitem="dataitem_like_sysprocedures_stack"
    fgltype="int" egltype="int" />
<column name="costfunc" dataitem="dataitem_like_sysprocedures_costfunc"
    fgltype="varchar" egltype="string(128)" size="128" />
<column name="selconst" dataitem="dataitem_like_sysprocedures_selconst"
    fgltype="smallfloat" egltype="smallfloat" />
<column name="collation" dataitem="dataitem_like_sysprocedures_collation"
    fgltype="char" egltype="unicode(36)" size="36" />
</table>
<table name="sysprocplan" egltype="rec_like_sysprocplan">
<column name="procid" dataitem="dataitem_like_sysprocplan_procid"
    fgltype="int" egltype="int" />
<column name="planid" dataitem="dataitem_like_sysprocplan_planid"
    fgltype="int" egltype="int" />
<column name="datakey" dataitem="dataitem_like_sysprocplan_datakey"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="seqno" dataitem="dataitem_like_sysprocplan_seqno"
    fgltype="int" egltype="int" />
<column name="created" dataitem="dataitem_like_sysprocplan_created"
    fgltype="date" egltype="date" />
<column name="datasize" dataitem="dataitem_like_sysprocplan_datasize"
    fgltype="int" egltype="int" />
<column name="data" dataitem="dataitem_like_sysprocplan_data"
    fgltype="char" egltype="unicode(256)" size="256" />
</table>
<table name="sysreferences" egltype="rec_like_sysreferences">
<column name="constrid" dataitem="dataitem_like_sysreferences_constrid"
    fgltype="int" egltype="int" />
<column name="primary" dataitem="dataitem_like_sysreferences_primary"
    fgltype="int" egltype="int" />
<column name="ptabid" dataitem="dataitem_like_sysreferences_ptabid"
    fgltype="int" egltype="int" />
<column name="updrule" dataitem="dataitem_like_sysreferences_updrule"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="delrule" dataitem="dataitem_like_sysreferences_delrule"
    fgltype="char" egltype="unicode(1)" size="1" />
<column name="matchtype" dataitem="dataitem_like_sysreferences_matchtype"

```

```

    fgltype="char" egltype="unicode(1)" size="1" />
    <column name="pendant" dataitem="dataitem_like_sysreferences_pendant"
    fgltype="char" egltype="unicode(1)" size="1" />
</table>
<table name="sysroleauth" egltype="rec_like_sysroleauth">
    <column name="rolename" dataitem="dataitem_like_sysroleauth_rolename"
    fgltype="char" egltype="unicode(32)" size="32" />
    <column name="grantee" dataitem="dataitem_like_sysroleauth_grantee"
    fgltype="char" egltype="unicode(32)" size="32" />
    <column name="is_grantable" dataitem
    ="dataitem_like_sysroleauth_is_grantable"
    fgltype="char" egltype="unicode(1)" size="1" />
</table>
<table name="sysroutinelangs" egltype="rec_like_sysroutinelangs">
    <column name="langid" dataitem="dataitem_like_sysroutinelangs_langid"
    fgltype="serial" egltype="int" />
    <column name="langname" dataitem="dataitem_like_sysroutinelangs_langname"
    fgltype="char" egltype="unicode(30)" size="30" />
    <column name="langinitfunc" dataitem
    ="dataitem_like_sysroutinelangs_langinitfunc"
    fgltype="varchar" egltype="string(128)" size="128" />
    <column name="langpath" dataitem="dataitem_like_sysroutinelangs_langpath"
    fgltype="char" egltype="unicode(255)" size="255" />
    <column name="langclass" dataitem="dataitem_like_sysroutinelangs_langclass"
    fgltype="char" egltype="unicode(18)" size="18" />
</table>
<table name="syssequences" egltype="rec_like_syssequences">
    <column name="seqid" dataitem="dataitem_like_syssequences_seqid"
    fgltype="serial" egltype="int" />
    <column name="tabid" dataitem="dataitem_like_syssequences_tabid"
    fgltype="int" egltype="int" />
    <column name="start_val" dataitem="dataitem_like_syssequences_start_val"
    fgltype="int8" egltype="bigint" />
    <column name="inc_val" dataitem="dataitem_like_syssequences_inc_val"
    fgltype="int8" egltype="bigint" />
    <column name="min_val" dataitem="dataitem_like_syssequences_min_val"
    fgltype="int8" egltype="bigint" />
    <column name="max_val" dataitem="dataitem_like_syssequences_max_val"
    fgltype="int8" egltype="bigint" />
    <column name="cycle" dataitem="dataitem_like_syssequences_cycle"
    fgltype="char" egltype="unicode(1)" size="1" />
    <column name="restart_val" dataitem
    ="dataitem_like_syssequences_restart_val"
    fgltype="int8" egltype="bigint" />
    <column name="cache" dataitem="dataitem_like_syssequences_cache"
    fgltype="int" egltype="int" />
    <column name="order" dataitem="dataitem_like_syssequences_order"
    fgltype="char" egltype="unicode(1)" size="1" />
</table>
<table name="syssynonyms" egltype="rec_like_syssynonyms">
    <column name="owner" dataitem="dataitem_like_syssynonyms_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
    <column name="synname" dataitem="dataitem_like_syssynonyms_synname"
    fgltype="varchar" egltype="string(128)" size="128" />
    <column name="created" dataitem="dataitem_like_syssynonyms_created"
    fgltype="date" egltype="date" />
    <column name="tabid" dataitem="dataitem_like_syssynonyms_tabid"
    fgltype="int" egltype="int" />
</table>
<table name="syssyntable" egltype="rec_like_syssyntable">
    <column name="tabid" dataitem="dataitem_like_syssyntable_tabid"
    fgltype="int" egltype="int" />
    <column name="servername" dataitem
    ="dataitem_like_syssyntable_servername"
    fgltype="varchar" egltype="string(128)" size="128" />
    <column name="dbname" dataitem="dataitem_like_syssyntable_dbname"
    fgltype="varchar" egltype="string(128)" size="128" />

```

```

<column name ="owner" dataitem ="dataitem_like_syssyntable_owner"
  fgltype ="char" egltype ="unicode(32)" size ="32" />
<column name ="tabname" dataitem ="dataitem_like_syssyntable_tabname"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
<column name ="btabid" dataitem ="dataitem_like_syssyntable_btabid"
  fgltype ="int" egltype ="int" />
</table>
<table name ="systabamdata" egltype ="rec_like_systabamdata">
<column name ="tabid" dataitem ="dataitem_like_systabamdata_tabid"
  fgltype ="int" egltype ="int" />
<column name ="am_param" dataitem ="dataitem_like_systabamdata_am_param"
  fgltype ="char" egltype ="unicode(256)" size ="256" />
<column name ="am_space" dataitem ="dataitem_like_systabamdata_am_space"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
</table>
<table name ="systabauth" egltype ="rec_like_systabauth">
<column name ="grantor" dataitem ="dataitem_like_systabauth_grantor"
  fgltype ="char" egltype ="unicode(32)" size ="32" />
<column name ="grantee" dataitem ="dataitem_like_systabauth_grantee"
  fgltype ="char" egltype ="unicode(32)" size ="32" />
<column name ="tabid" dataitem ="dataitem_like_systabauth_tabid"
  fgltype ="int" egltype ="int" />
<column name ="tabauth" dataitem ="dataitem_like_systabauth_tabauth"
  fgltype ="char" egltype ="unicode(9)" size ="9" />
</table>
<table name ="systables" egltype ="rec_like_systables">
<column name ="tabname" dataitem ="dataitem_like_systables_tabname"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
<column name ="owner" dataitem ="dataitem_like_systables_owner"
  fgltype ="char" egltype ="unicode(32)" size ="32" />
<column name ="partnum" dataitem ="dataitem_like_systables_partnum"
  fgltype ="int" egltype ="int" />
<column name ="tabid" dataitem ="dataitem_like_systables_tabid"
  fgltype ="serial" egltype ="int" />
<column name ="rowsize" dataitem ="dataitem_like_systables_rowsize"
  fgltype ="smallint" egltype ="smallint" />
<column name ="ncols" dataitem ="dataitem_like_systables_ncols"
  fgltype ="smallint" egltype ="smallint" />
<column name ="nindexes" dataitem ="dataitem_like_systables_nindexes"
  fgltype ="smallint" egltype ="smallint" />
<column name ="nrows" dataitem ="dataitem_like_systables_nrows"
  fgltype ="int" egltype ="int" />
<column name ="created" dataitem ="dataitem_like_systables_created"
  fgltype ="date" egltype ="date" />
<column name ="version" dataitem ="dataitem_like_systables_version"
  fgltype ="int" egltype ="int" />
<column name ="tabtype" dataitem ="dataitem_like_systables_tabtype"
  fgltype ="char" egltype ="unicode(1)" size ="1" />
<column name ="locklevel" dataitem ="dataitem_like_systables_locklevel"
  fgltype ="char" egltype ="unicode(1)" size ="1" />
<column name ="npused" dataitem ="dataitem_like_systables_npused"
  fgltype ="int" egltype ="int" />
<column name ="fextsize" dataitem ="dataitem_like_systables_fextsize"
  fgltype ="int" egltype ="int" />
<column name ="nextsize" dataitem ="dataitem_like_systables_nextsize"
  fgltype ="int" egltype ="int" />
<column name ="flags" dataitem ="dataitem_like_systables_flags"
  fgltype ="smallint" egltype ="smallint" />
<column name ="site" dataitem ="dataitem_like_systables_site"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
<column name ="dbname" dataitem ="dataitem_like_systables_dbname"
  fgltype ="varchar" egltype ="string(128)" size ="128" />
<column name ="type_xid" dataitem ="dataitem_like_systables_type_xid"
  fgltype ="int" egltype ="int" />
<column name ="am_id" dataitem ="dataitem_like_systables_am_id"
  fgltype ="int" egltype ="int" />
<column name ="pagesize" dataitem ="dataitem_like_systables_pagesize"

```

```

    fgltype="int" egltype="int" />
</table>
<table name="systraceclasses" egltype="rec_like_systraceclasses">
  <column name="name" dataitem="dataitem_like_systraceclasses_name"
    fgltype="char" egltype="unicode(18)" size="18" />
  <column name="classid" dataitem="dataitem_like_systraceclasses_classid"
    fgltype="serial" egltype="int" />
</table>
<table name="systracemsgs" egltype="rec_like_systracemsgs">
  <column name="name" dataitem="dataitem_like_systracemsgs_name"
    fgltype="varchar" egltype="string(128)" size="128" />
  <column name="msgid" dataitem="dataitem_like_systracemsgs_msgid"
    fgltype="serial" egltype="int" />
  <column name="locale" dataitem="dataitem_like_systracemsgs_locale"
    fgltype="char" egltype="unicode(36)" size="36" />
  <column name="seqno" dataitem="dataitem_like_systracemsgs_seqno"
    fgltype="smallint" egltype="smallint" />
  <column name="message" dataitem="dataitem_like_systracemsgs_message"
    fgltype="varchar" egltype="string(255)" size="255" />
</table>
<table name="systrigbody" egltype="rec_like_systrigbody">
  <column name="trigid" dataitem="dataitem_like_systrigbody_trigid"
    fgltype="int" egltype="int" />
  <column name="datakey" dataitem="dataitem_like_systrigbody_datakey"
    fgltype="char" egltype="unicode(1)" size="1" />
  <column name="seqno" dataitem="dataitem_like_systrigbody_seqno"
    fgltype="int" egltype="int" />
  <column name="data" dataitem="dataitem_like_systrigbody_data"
    fgltype="char" egltype="unicode(256)" size="256" />
</table>
<table name="systriggers" egltype="rec_like_systriggers">
  <column name="trigid" dataitem="dataitem_like_systriggers_trigid"
    fgltype="serial" egltype="int" />
  <column name="trigname" dataitem="dataitem_like_systriggers_trigname"
    fgltype="varchar" egltype="string(128)" size="128" />
  <column name="owner" dataitem="dataitem_like_systriggers_owner"
    fgltype="char" egltype="unicode(32)" size="32" />
  <column name="tabid" dataitem="dataitem_like_systriggers_tabid"
    fgltype="int" egltype="int" />
  <column name="event" dataitem="dataitem_like_systriggers_event"
    fgltype="char" egltype="unicode(1)" size="1" />
  <column name="old" dataitem="dataitem_like_systriggers_old"
    fgltype="varchar" egltype="string(128)" size="128" />
  <column name="new" dataitem="dataitem_like_systriggers_new"
    fgltype="varchar" egltype="string(128)" size="128" />
  <column name="mode" dataitem="dataitem_like_systriggers_mode"
    fgltype="char" egltype="unicode(1)" size="1" />
  <column name="collation" dataitem="dataitem_like_systriggers_collation"
    fgltype="char" egltype="unicode(36)" size="36" />
</table>
<table name="sysusers" egltype="rec_like_sysusers">
  <column name="username" dataitem="dataitem_like_sysusers_username"
    fgltype="char" egltype="unicode(32)" size="32" />
  <column name="usertype" dataitem="dataitem_like_sysusers_usertype"
    fgltype="char" egltype="unicode(1)" size="1" />
  <column name="priority" dataitem="dataitem_like_sysusers_priority"
    fgltype="smallint" egltype="smallint" />
  <column name="password" dataitem="dataitem_like_sysusers_password"
    fgltype="char" egltype="unicode(16)" size="16" />
  <column name="defrole" dataitem="dataitem_like_sysusers_defrole"
    fgltype="char" egltype="unicode(32)" size="32" />
</table>
<table name="sysviews" egltype="rec_like_sysviews">
  <column name="tabid" dataitem="dataitem_like_sysviews_tabid"
    fgltype="int" egltype="int" />
  <column name="seqno" dataitem="dataitem_like_sysviews_seqno"
    fgltype="smallint" egltype="smallint" />

```

```

    <column name ="viewtext" dataitem ="dataitem_like_sysviews_viewtext"
      fgltype ="char" egltype ="unicode(64)" size ="64" />
  </table>
  <table name ="sysviolations" egltype ="rec_like_sysviolations">
    <column name ="targettid" dataitem ="dataitem_like_sysviolations_targettid"
      fgltype ="int" egltype ="int" />
    <column name ="viotid" dataitem ="dataitem_like_sysviolations_viotid"
      fgltype ="int" egltype ="int" />
    <column name ="diatid" dataitem ="dataitem_like_sysviolations_diatid"
      fgltype ="int" egltype ="int" />
    <column name ="maxrows" dataitem ="dataitem_like_sysviolations_maxrows"
      fgltype ="int" egltype ="int" />
  </table>
  <table name ="sysxadasources" egltype ="rec_like_sysxadasources">
    <column name ="xa_datasrc_owner" dataitem
      ="dataitem_like_sysxadasources_xa_datasrc_owner"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="xa_datasrc_name" dataitem
      ="dataitem_like_sysxadasources_xa_datasrc_name"
      fgltype ="varchar" egltype ="string(128)" size ="128" />
    <column name ="xa_datasrc_rmid" dataitem
      ="dataitem_like_sysxadasources_xa_datasrc_rmid"
      fgltype ="serial" egltype ="int" />
    <column name ="xa_source_typeid" dataitem
      ="dataitem_like_sysxadasources_xa_source_typeid"
      fgltype ="int" egltype ="int" />
  </table>
  <table name ="sysxasourcetypes" egltype ="rec_like_sysxasourcetypes">
    <column name ="xa_source_typeid" dataitem
      ="dataitem_like_sysxasourcetypes_xa_source_typeid"
      fgltype ="serial" egltype ="int" />
    <column name ="xa_source_owner" dataitem
      ="dataitem_like_sysxasourcetypes_xa_source_owner"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="xa_source_name" dataitem
      ="dataitem_like_sysxasourcetypes_xa_source_name"
      fgltype ="varchar" egltype ="string(128)" size ="128" />
    <column name ="xa_flags" dataitem
      ="dataitem_like_sysxasourcetypes_xa_flags"
      fgltype ="int" egltype ="int" />
    <column name ="xa_version" dataitem
      ="dataitem_like_sysxasourcetypes_xa_version"
      fgltype ="int" egltype ="int" />
    <column name ="xa_open" dataitem
      ="dataitem_like_sysxasourcetypes_xa_open"
      fgltype ="int" egltype ="int" />
    <column name ="xa_close" dataitem ="dataitem_like_sysxasourcetypes_xa_close"
      fgltype ="int" egltype ="int" />
    <column name ="xa_start" dataitem
      ="dataitem_like_sysxasourcetypes_xa_start"
      fgltype ="int" egltype ="int" />
    <column name ="xa_end" dataitem
      ="dataitem_like_sysxasourcetypes_xa_end"
      fgltype ="int" egltype ="int" />
    <column name ="xa_rollback" dataitem
      ="dataitem_like_sysxasourcetypes_xa_rollback"
      fgltype ="int" egltype ="int" />
    <column name ="xa_prepare" dataitem
      ="dataitem_like_sysxasourcetypes_xa_prepare"
      fgltype ="int" egltype ="int" />
    <column name ="xa_commit" dataitem
      ="dataitem_like_sysxasourcetypes_xa_commit"
      fgltype ="int" egltype ="int" />
    <column name ="xa_recover" dataitem
      ="dataitem_like_sysxasourcetypes_xa_recover"
      fgltype ="int" egltype ="int" />
    <column name ="xa_forget" dataitem

```

```

        = "dataitem_like_sysxasourcetypes_xa_forget"
        fgltype = "int" egltype = "int" />
<column name = "xa_complete" dataitem
        = "dataitem_like_sysxasourcetypes_xa_complete"
        fgltype = "int" egltype = "int" />
</table>
<table name = "sysxtddesc" egltype = "rec_like_sysxtddesc">

    <column name = "extended_id" dataitem = "dataitem_like_sysxtddesc_extended_id"
        fgltype = "int" egltype = "int" />
    <column name = "seqno" dataitem = "dataitem_like_sysxtddesc_seqno"
        fgltype = "smallint" egltype = "smallint" />
    <column name = "description" dataitem = "dataitem_like_sysxtddesc_description"
        fgltype = "char" egltype = "unicode(256)" size = "256" />
</table>
<table name = "sysxtdtypeauth" egltype = "rec_like_sysxtdtypeauth">
    <column name = "grantor" dataitem = "dataitem_like_sysxtdtypeauth_grantor"
        fgltype = "char" egltype = "unicode(32)" size = "32" />
    <column name = "grantee" dataitem = "dataitem_like_sysxtdtypeauth_grantee"
        fgltype = "char" egltype = "unicode(32)" size = "32" />
    <column name = "type" dataitem = "dataitem_like_sysxtdtypeauth_type"
        fgltype = "int" egltype = "int" />
    <column name = "auth" dataitem = "dataitem_like_sysxtdtypeauth_auth"
        fgltype = "char" egltype = "unicode(2)" size = "2" />
</table>
<table name = "sysxtdtypes" egltype = "rec_like_sysxtdtypes">
    <column name = "extended_id" dataitem = "dataitem_like_sysxtdtypes_extended_id"
        fgltype = "serial" egltype = "int" />
    <column name = "domain" dataitem = "dataitem_like_sysxtdtypes_domain"
        fgltype = "char" egltype = "unicode(1)" size = "1" />
    <column name = "mode" dataitem = "dataitem_like_sysxtdtypes_mode"
        fgltype = "char" egltype = "unicode(1)" size = "1" />
    <column name = "owner" dataitem = "dataitem_like_sysxtdtypes_owner"
        fgltype = "char" egltype = "unicode(32)" size = "32" />
    <column name = "name" dataitem = "dataitem_like_sysxtdtypes_name"
        fgltype = "varchar" egltype = "string(128)" size = "128" />
    <column name = "type" dataitem = "dataitem_like_sysxtdtypes_type"
        fgltype = "smallint" egltype = "smallint" />
    <column name = "source" dataitem = "dataitem_like_sysxtdtypes_source"
        fgltype = "int" egltype = "int" />
    <column name = "maxlen" dataitem = "dataitem_like_sysxtdtypes_maxlen"
        fgltype = "int" egltype = "int" />
    <column name = "length" dataitem = "dataitem_like_sysxtdtypes_length"
        fgltype = "int" egltype = "int" />
    <column name = "byvalue" dataitem = "dataitem_like_sysxtdtypes_byvalue"
        fgltype = "char" egltype = "unicode(1)" size = "1" />
    <column name = "cannothash" dataitem = "dataitem_like_sysxtdtypes_cannothash"
        fgltype = "char" egltype = "unicode(1)" size = "1" />
    <column name = "align" dataitem = "dataitem_like_sysxtdtypes_align"
        fgltype = "smallint" egltype = "smallint" />
    <column name = "locator" dataitem = "dataitem_like_sysxtdtypes_locator"
        fgltype = "int" egltype = "int" />
</table>
<table name = "call_type" egltype = "rec_like_call_type">
    <column name = "call_code" dataitem = "dataitem_like_call_type_call_code"
        fgltype = "char" egltype = "unicode(1)" size = "1" />
    <column name = "code_descr" dataitem = "dataitem_like_call_type_code_descr"
        fgltype = "char" egltype = "unicode(30)" size = "30" />
</table>
<table name = "catalog" egltype = "rec_like_catalog">
    <column name = "catalog_num" dataitem = "dataitem_like_catalog_catalog_num"
        fgltype = "serial" egltype = "int" />
    <column name = "stock_num" dataitem = "dataitem_like_catalog_stock_num"
        fgltype = "smallint" egltype = "smallint" />
    <column name = "manu_code" dataitem = "dataitem_like_catalog_manu_code"
        fgltype = "char" egltype = "unicode(3)" size = "3" />
    <column name = "cat_descr" dataitem = "dataitem_like_catalog_cat_descr"

```

```

    fgltype="text" egltype="clob" />
    <column name="cat_picture" dataitem="dataitem_like_catalog_cat_picture"
      fgltype="byte" egltype="blob" />
    <column name="cat_advert" dataitem="dataitem_like_catalog_cat_advert"
      fgltype="varchar" egltype="string(255)" size="255" />
  </table>
  <table name="cust_calls" egltype="rec_like_cust_calls">
    <column name="customer_num" dataitem="dataitem_like_cust_calls_customer_num"
      fgltype="int" egltype="int" />
    <column name="call_dtime" dataitem="dataitem_like_cust_calls_call_dtime"
      fgltype="datetime year to minute"
      egltype="timestamp(&quot;yyyyMMddhhmm&quot;)"
      start="year" end="minute" />
    <column name="user_id" dataitem="dataitem_like_cust_calls_user_id"
      fgltype="char" egltype="unicode(32)" size="32" />
    <column name="call_code" dataitem="dataitem_like_cust_calls_call_code"
      fgltype="char" egltype="unicode(1)" size="1" />
    <column name="call_descr" dataitem="dataitem_like_cust_calls_call_descr"
      fgltype="char" egltype="unicode(240)" size="240" />
    <column name="res_dtime" dataitem="dataitem_like_cust_calls_res_dtime"
      fgltype="datetime year to minute"
      egltype="timestamp(&quot;yyyyMMddhhmm&quot;)"
      start="year" end="minute" />
    <column name="res_descr" dataitem="dataitem_like_cust_calls_res_descr"
      fgltype="char" egltype="unicode(240)" size="240" />
  </table>
  <table name="customer" egltype="rec_like_customer">
    <column name="customer_num" dataitem="dataitem_like_customer_customer_num"
      fgltype="serial" egltype="int" />
    <column name="fname" dataitem="dataitem_like_customer_fname"
      fgltype="char" egltype="unicode(15)" size="15" />
    <column name="lname" dataitem="dataitem_like_customer_lname"
      fgltype="char" egltype="unicode(15)" size="15" />
    <column name="company" dataitem="dataitem_like_customer_company"
      fgltype="char" egltype="unicode(20)" size="20" />
    <column name="address1" dataitem="dataitem_like_customer_address1"
      fgltype="char" egltype="unicode(20)" size="20" />
    <column name="address2" dataitem="dataitem_like_customer_address2"
      fgltype="char" egltype="unicode(20)" size="20" />
    <column name="city" dataitem="dataitem_like_customer_city"
      fgltype="char" egltype="unicode(15)" size="15" />
    <column name="state" dataitem="dataitem_like_customer_state"
      fgltype="char" egltype="unicode(2)" size="2" />
    <column name="zipcode" dataitem="dataitem_like_customer_zipcode"
      fgltype="char" egltype="unicode(5)" size="5" />
    <column name="phone" dataitem="dataitem_like_customer_phone"
      fgltype="char" egltype="unicode(18)" size="18" />
  </table>
  <table name="items" egltype="rec_like_items">
    <column name="item_num" dataitem="dataitem_like_items_item_num"
      fgltype="smallint" egltype="smallint" />
    <column name="order_num" dataitem="dataitem_like_items_order_num"
      fgltype="int" egltype="int" />
    <column name="stock_num" dataitem="dataitem_like_items_stock_num"
      fgltype="smallint" egltype="smallint" />
    <column name="manu_code" dataitem="dataitem_like_items_manu_code"
      fgltype="char" egltype="unicode(3)" size="3" />
    <column name="quantity" dataitem="dataitem_like_items_quantity"
      fgltype="smallint" egltype="smallint" />
    <column name="total_price" dataitem="dataitem_like_items_total_price"
      fgltype="money" egltype="money(8,2)" scale="2" precision="8" />
  </table>
  <table name="manufact" egltype="rec_like_manufact">
    <column name="manu_code" dataitem="dataitem_like_manufact_manu_code"
      fgltype="char" egltype="unicode(3)" size="3" />
    <column name="manu_name" dataitem="dataitem_like_manufact_manu_name"
      fgltype="char" egltype="unicode(15)" size="15" />

```

```

    <column name ="lead_time" dataitem ="dataitem_like_manufact_lead_time"
      fgltype ="interval_day(3) to day" egltype ="interval(&quot;ddd&quot;);"
precision ="3" start ="day" end ="day" />
  </table>
  <table name ="msgs" egltype ="rec_like_msgs">
    <column name ="lang" dataitem ="dataitem_like_msgs_lang"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="number" dataitem ="dataitem_like_msgs_number"
      fgltype ="int" egltype ="int" />
    <column name ="message" dataitem ="dataitem_like_msgs_message"
      fgltype ="nchar" egltype ="unicode(255)" size ="255" />
  </table>
  <table name ="orders" egltype ="rec_like_orders">
    <column name ="order_num" dataitem ="dataitem_like_orders_order_num"
      fgltype ="serial" egltype ="int" />
    <column name ="order_date" dataitem ="dataitem_like_orders_order_date"
      fgltype ="date" egltype ="date" />
    <column name ="customer_num" dataitem ="dataitem_like_orders_customer_num"
      fgltype ="int" egltype ="int" />
    <column name ="ship_instruct" dataitem ="dataitem_like_orders_ship_instruct"
      fgltype ="char" egltype ="unicode(40)" size ="40" />
    <column name ="backlog" dataitem ="dataitem_like_orders_backlog"
      fgltype ="char" egltype ="unicode(1)" size ="1" />
    <column name ="po_num" dataitem ="dataitem_like_orders_po_num"
      fgltype ="char" egltype ="unicode(10)" size ="10" />
    <column name ="ship_date" dataitem ="dataitem_like_orders_ship_date"
      fgltype ="date" egltype ="date" />
    <column name ="ship_weight" dataitem ="dataitem_like_orders_ship_weight"
      fgltype ="decimal" egltype ="decimal(8,2)" scale ="2" precision ="8" />
    <column name ="ship_charge" dataitem ="dataitem_like_orders_ship_charge"
      fgltype ="money" egltype ="money(6,2)" scale ="2" precision ="6" />
    <column name ="paid_date" dataitem ="dataitem_like_orders_paid_date"
      fgltype ="date" egltype ="date" />
  </table>
  <table name ="state" egltype ="rec_like_state">
    <column name ="code" dataitem ="dataitem_like_state_code"
      fgltype ="char" egltype ="unicode(2)" size ="2" />
    <column name ="sname" dataitem ="dataitem_like_state_sname"
      fgltype ="char" egltype ="unicode(15)" size ="15" />
  </table>
  <table name ="stock" egltype ="rec_like_stock">
    <column name ="stock_num" dataitem ="dataitem_like_stock_stock_num"
      fgltype ="smallint" egltype ="smallint" />
    <column name ="manu_code" dataitem ="dataitem_like_stock_manu_code"
      fgltype ="char" egltype ="unicode(3)" size ="3" />
    <column name ="description" dataitem ="dataitem_like_stock_description"
      fgltype ="char" egltype ="unicode(15)" size ="15" />
    <column name ="unit_price" dataitem ="dataitem_like_stock_unit_price"
      fgltype ="money" egltype ="money(6,2)" scale ="2" precision ="6" />
    <column name ="unit" dataitem ="dataitem_like_stock_unit"
      fgltype ="char" egltype ="unicode(4)" size ="4" />
    <column name ="unit_descr" dataitem ="dataitem_like_stock_unit_descr"
      fgltype ="char" egltype ="unicode(15)" size ="15" />
  </table>
  <table name ="sysdomains" egltype ="rec_like_sysdomains">
    <column name ="id" dataitem ="dataitem_like_sysdomains_id"
      fgltype ="serial" egltype ="int" />
    <column name ="owner" dataitem ="dataitem_like_sysdomains_owner"
      fgltype ="char" egltype ="unicode(32)" size ="32" />
    <column name ="name" dataitem ="dataitem_like_sysdomains_name"
      fgltype ="varchar" egltype ="string(128)" size ="128" />
    <column name ="type" dataitem ="dataitem_like_sysdomains_type"
      fgltype ="smallint" egltype ="smallint" />
  </table>
  <table name ="sysindexes" egltype ="rec_like_sysindexes">
    <column name ="idxname" dataitem ="dataitem_like_sysindexes_idxname"
      fgltype ="varchar" egltype ="string(128)" size ="128" />

```

```

<column name ="owner" dataitem ="dataitem_like_sysindexes_owner"
  fgltype ="char" egltype ="unicode(32)" size ="32" />
<column name ="tabid" dataitem ="dataitem_like_sysindexes_tabid"
  fgltype ="int" egltype ="int" />
<column name ="idxtype" dataitem ="dataitem_like_sysindexes_idxtype"
  fgltype ="char" egltype ="unicode(1)" size ="1" />
<column name ="clustered" dataitem ="dataitem_like_sysindexes_clustered"
  fgltype ="char" egltype ="unicode(1)" size ="1" />
<column name ="part1" dataitem ="dataitem_like_sysindexes_part1"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part2" dataitem ="dataitem_like_sysindexes_part2"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part3" dataitem ="dataitem_like_sysindexes_part3"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part4" dataitem ="dataitem_like_sysindexes_part4"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part5" dataitem ="dataitem_like_sysindexes_part5"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part6" dataitem ="dataitem_like_sysindexes_part6"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part7" dataitem ="dataitem_like_sysindexes_part7"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part8" dataitem ="dataitem_like_sysindexes_part8"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part9" dataitem ="dataitem_like_sysindexes_part9"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part10" dataitem ="dataitem_like_sysindexes_part10"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part11" dataitem ="dataitem_like_sysindexes_part11"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part12" dataitem ="dataitem_like_sysindexes_part12"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part13" dataitem ="dataitem_like_sysindexes_part13"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part14" dataitem ="dataitem_like_sysindexes_part14"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part15" dataitem ="dataitem_like_sysindexes_part15"
  fgltype ="smallint" egltype ="smallint" />
<column name ="part16" dataitem ="dataitem_like_sysindexes_part16"
  fgltype ="smallint" egltype ="smallint" />
<column name ="levels" dataitem ="dataitem_like_sysindexes_levels"
  fgltype ="smallint" egltype ="smallint" />
<column name ="leaves" dataitem ="dataitem_like_sysindexes_leaves"
  fgltype ="int" egltype ="int" />
<column name ="nunique" dataitem ="dataitem_like_sysindexes_nunique"
  fgltype ="int" egltype ="int" />
<column name ="clust" dataitem ="dataitem_like_sysindexes_clust"
  fgltype ="int" egltype ="int" />
</table>
</package>
</manifest>

```

---

## 共用ライブラリーまたはアプリケーション・プロジェクトのマニフェスト・ファイル例

この例は、共用ライブラリーのマニフェスト・ファイルを示しています。共用ライブラリーおよびアプリケーション・プロジェクトには、同一のマニフェスト・ファイルが存在し、以下の例外があります。

- 共用ライブラリー・プロジェクトのマニフェスト・ファイルで、属性 **type** が **library** に等しくなります。
- アプリケーション・プロジェクトのマニフェスト・ファイルで、属性 **type** が **application** に等しくなります。

```

<?xml version="1.0" encoding="utf-8"?>
<!--
    Manifest file generated by I4GL to EGL Conversion Utility
    Project Name   :I4gldemoSharedLibrary
    Generated on  :Fri Jan 28 14:01:36 CST 2005
-->

<!--DTD for Manifest file-->
<!DOCTYPE manifest [
<!ELEMENT manifest (package)>
    <!--ATTLIST manifest project CDATA #REQUIRED>
    <!--ATTLIST manifest type CDATA #FIXED "library">
<!ELEMENT package (rectype*, variables*,
function*,cfunc*,cursor*,
preparedStatements*)>
    <!--ATTLIST package name CDATA #REQUIRED>
<!ELEMENT function (parameter*,return*)>
    <!--ATTLIST function name CDATA #REQUIRED>
    <!--ATTLIST function package CDATA #IMPLIED>
    <!--ATTLIST function library CDATA #REQUIRED>
    <!--ATTLIST function type CDATA #REQUIRED>
<!ELEMENT parameter EMPTY>
    <!--ATTLIST parameter name CDATA #REQUIRED>
    <!--ATTLIST parameter egltype CDATA #REQUIRED>
    <!--ATTLIST parameter fgltype CDATA #IMPLIED>
    <!--ATTLIST parameter size CDATA #IMPLIED>
    <!--ATTLIST parameter precision CDATA #IMPLIED>
    <!--ATTLIST parameter scale CDATA #IMPLIED>
    <!--ATTLIST parameter start CDATA #IMPLIED>
    <!--ATTLIST parameter end CDATA #IMPLIED>
    <!--ATTLIST parameter isrectype (t|f) "f">
    <!--ATTLIST parameter library CDATA #REQUIRED>
<!ELEMENT return EMPTY>
    <!--ATTLIST return name CDATA #REQUIRED>
    <!--ATTLIST return egltype CDATA #REQUIRED>
    <!--ATTLIST return fgltype CDATA #IMPLIED>
    <!--ATTLIST return size CDATA #IMPLIED>
    <!--ATTLIST return precision CDATA #IMPLIED>
    <!--ATTLIST return scale CDATA #IMPLIED>
    <!--ATTLIST return start CDATA #IMPLIED>
    <!--ATTLIST return end CDATA #IMPLIED>
    <!--ATTLIST return isrectype (t|f) "f" >
    <!--ATTLIST return library CDATA #REQUIRED>
<!ELEMENT cfunc (dependentPackage*) >
    <!--ATTLIST cfunc name CDATA #REQUIRED>
    <!--ATTLIST cfunc package CDATA #REQUIRED>
    <!--ATTLIST cfunc library CDATA #REQUIRED>
    <!--ATTLIST cfunc argcount CDATA #REQUIRED>
    <!--ATTLIST cfunc retcount CDATA #REQUIRED>
<!ELEMENT dependentPackage EMPTY>
    <!--ATTLIST dependentPackage package CDATA #REQUIRED>
<!ELEMENT cursor EMPTY>
    <!--ATTLIST cursor name CDATA #REQUIRED>
    <!--ATTLIST cursor ishold (t|f) "f" >
    <!--ATTLIST cursor isscrolling (t|f) "f" >
    <!--ATTLIST cursor library CDATA #REQUIRED>
<!ELEMENT variables (variable*)>
<!ELEMENT variable EMPTY>
    <!--ATTLIST variable name CDATA #REQUIRED>
    <!--ATTLIST variable egltype CDATA #REQUIRED>
    <!--ATTLIST variable fgltype CDATA #IMPLIED>
    <!--ATTLIST variable size CDATA #IMPLIED>
    <!--ATTLIST variable precision CDATA #IMPLIED>
    <!--ATTLIST variable scale CDATA #IMPLIED>
    <!--ATTLIST variable start CDATA #IMPLIED>
    <!--ATTLIST variable end CDATA #IMPLIED>
    <!--ATTLIST variable isrectype (t|f) "f" >

```

```

    <!--ATTLIST variable library CDATA #REQUIRED-->
  <!--ELEMENT rectype (field*)-->
    <!--ATTLIST rectype name CDATA #REQUIRED-->
    <!--ATTLIST rectype library CDATA #REQUIRED-->
  <!--ELEMENT field EMPTY-->
    <!--ATTLIST field name CDATA #REQUIRED-->
    <!--ATTLIST field egltype CDATA #REQUIRED-->
    <!--ATTLIST field fgltype CDATA #IMPLIED-->
    <!--ATTLIST field size CDATA #IMPLIED-->
    <!--ATTLIST field precision CDATA #IMPLIED-->
    <!--ATTLIST field scale CDATA #IMPLIED-->
    <!--ATTLIST field start CDATA #IMPLIED-->
    <!--ATTLIST field end CDATA #IMPLIED-->
    <!--ATTLIST field isrectype (t|f) "f" -->
    <!--ATTLIST field library CDATA #REQUIRED-->
  <!--ELEMENT preparedStatements (statement*)-->
  <!--ELEMENT statement EMPTY-->
    <!--ATTLIST statement name CDATA #REQUIRED-->
    <!--ATTLIST statement library CDATA #REQUIRED-->
]>

<manifest project="I4gldemoSharedLibrary" type="library" >
<package name="I4gldemoSharedLibrary">
<!--Record Declarations-->
<rectype name="recordtype_d4_orders_invoice_x_invoice"
library="d4_orders">
  <field name="order_num" library="d4_orders" egltype="int"
fgltype="serial" isrectype="f" />
  <field name="order_date" library="d4_orders" egltype="date"
fgltype="date" isrectype="f" />
  <field name="ship_instruct" library="d4_orders"
egltype="unicode(40)"
fgltype="char" size="40" isrectype="f" />
  <field name="backlog" library="d4_orders" egltype="unicode(1)"
fgltype="char" size="1" isrectype="f" />
  <field name="po_num" library="d4_orders" egltype="unicode(10)"
fgltype="char" size="10" isrectype="f" />
  <field name="ship_date" library="d4_orders" egltype="date"
fgltype="date" isrectype="f" />
  <field name="ship_weight" library="d4_orders"
egltype="decimal(8,2)" fgltype="decimal" precision="8"
scale="2" isrectype="f" />
  <field name="ship_charge" library="d4_orders"
egltype="money(6,2)" fgltype="money" precision="6"
scale="2" isrectype="f" />
  <field name="item_num" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="stock_num" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="manu_code" library="d4_orders" egltype="unicode(3)"
fgltype="char" size="3" isrectype="f" />
  <field name="quantity" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="total_price" library="d4_orders" egltype="money(8,2)"
fgltype="money" precision="8" scale="2" isrectype="f" />
  <field name="description" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
  <field name="unit_price" library="d4_orders" egltype="money(6,2)"
fgltype="money" precision="6" scale="2" isrectype="f" />
  <field name="unit" library="d4_orders" egltype="unicode(4)"
fgltype="char" size="4" isrectype="f" />
  <field name="unit_descr" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
  <field name="manu_name" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
</rectype>
<rectype name="recordtype_d4_orders_x" library="d4_orders">

```

```

        <field name="order_num" library="d4_orders" egltype="int"
fgltype="serial" isrectype="f" />
        <field name="order_date" library="d4_orders" egltype="date"
fgltype="date" isrectype="f" />
        <field name="ship_instruct" library="d4_orders"
fgltype="unicode(40)" fgltype="char" size="40" isrectype="f" />
        <field name="backlog" library="d4_orders" egltype="unicode(1)"
fgltype="char" size="1" isrectype="f" />
        <field name="po_num" library="d4_orders" egltype="unicode(10)"
fgltype="char" size="10" isrectype="f" />
        <field name="ship_date" library="d4_orders" egltype="date"
fgltype="date" isrectype="f" />
        <field name="ship_weight" library="d4_orders" egltype="decimal(8,2)"
fgltype="decimal" precision="8" scale="2" isrectype="f" />
        <field name="ship_charge" library="d4_orders" egltype="money(6,2)"
fgltype="money" precision="6" scale="2" isrectype="f" />
        <field name="item_num" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
        <field name="stock_num" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
        <field name="manu_code" library="d4_orders" egltype="unicode(3)"
fgltype="char" size="3" isrectype="f" />
        <field name="quantity" library="d4_orders" egltype="smallint"
fgltype="smallint" isrectype="f" />
        <field name="total_price" library="d4_orders" egltype="money(8,2)"
fgltype="money" precision="8" scale="2" isrectype="f" />
        <field name="description" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="unit_price" library="d4_orders" egltype="money(6,2)"
fgltype="money" precision="6" scale="2" isrectype="f" />
        <field name="unit" library="d4_orders" egltype="unicode(4)"
fgltype="char" size="4" isrectype="f" />
        <field name="unit_descr" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="manu_name" library="d4_orders" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
</rectype>
<rectype name="recordtype_d4_report_print_ar_r" library="d4_report">
        <field name="customer_num" library="d4_report" egltype="int"
fgltype="serial" isrectype="f" />
        <field name="fname" library="d4_report" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="lname" library="d4_report" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="company" library="d4_report" egltype="unicode(20)"
fgltype="char" size="20" isrectype="f" />
        <field name="order_num" library="d4_report" egltype="int"
fgltype="serial" isrectype="f" />
        <field name="order_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
        <field name="ship_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
        <field name="paid_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
        <field name="total_price" library="d4_report" egltype="money(8,2)"
fgltype="money" precision="8" scale="2" isrectype="f" />
</rectype>
<rectype name="recordtype_d4_report_r" library="d4_report">
        <field name="customer_num" library="d4_report" egltype="int"
fgltype="serial" isrectype="f" />
        <field name="fname" library="d4_report" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="lname" library="d4_report" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
        <field name="company" library="d4_report" egltype="unicode(20)"
fgltype="char" size="20" isrectype="f" />
        <field name="order_num" library="d4_report" egltype="int"

```

```

fgltype="serial" isrectype="f" />
  <field name="order_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
  <field name="ship_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
  <field name="paid_date" library="d4_report" egltype="date"
fgltype="date" isrectype="f" />
  <field name="total_price" library="d4_report" egltype="money(8,2)"
fgltype="money" precision="8" scale="2" isrectype="f" />
</rectype>
<rectype name="recordtype_p_items" library="d4_globals">
  <field name="item_num" library="d4_globals" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="stock_num" library="d4_globals" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="manu_code" library="d4_globals" egltype="unicode(3)"
fgltype="char" size="3" isrectype="f" />
  <field name="description" library="d4_globals" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
  <field name="quantity" library="d4_globals" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="unit_price" library="d4_globals" egltype="money(6,2)"
fgltype="money" precision="6" scale="2" isrectype="f" />
  <field name="total_price" library="d4_globals" egltype="money(8,2)"
fgltype="money" precision="8" scale="2" isrectype="f" />
</rectype>
<rectype name="recordtype_p_orders" library="d4_globals">
  <field name="order_num" library="d4_globals" egltype="int"
fgltype="serial" isrectype="f" />
  <field name="order_date" library="d4_globals" egltype="date"
fgltype="date" isrectype="f" />
  <field name="po_num" library="d4_globals" egltype="unicode(10)"
fgltype="char" size="10" isrectype="f" />
  <field name="ship_instruct" library="d4_globals"
egltype="unicode(40)" fgltype="char" size="40" isrectype="f" />
</rectype>
<rectype name="recordtype_p_stock" library="d4_globals">
  <field name="stock_num" library="d4_globals" egltype="smallint"
fgltype="smallint" isrectype="f" />
  <field name="manu_code" library="d4_globals" egltype="unicode(3)"
fgltype="char" size="3" isrectype="f" />
  <field name="manu_name" library="d4_globals" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
  <field name="description" library="d4_globals" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
  <field name="unit_price" library="d4_globals" egltype="money(6,2)"
fgltype="money" precision="6" scale="2" isrectype="f" />
  <field name="unit_descr" library="d4_globals" egltype="unicode(15)"
fgltype="char" size="15" isrectype="f" />
</rectype>

<!--Global Variable Declarations-->
<variables>
  <variable name="p_customer" library="d4_globals"
egltype="rec_like_customer" isrectype="t" />
  <variable name="p_items" library="d4_globals"
egltype="recordtype_p_items" isrectype="t" />
  <variable name="p_orders" library="d4_globals"
egltype="recordtype_p_orders" isrectype="t" />
  <variable name="p_state" library="d4_globals"
egltype="rec_like_state" isrectype="t" />
  <variable name="p_stock" library="d4_globals"
egltype="recordtype_p_stock" isrectype="t" />
  <variable name="print_option" library="d4_globals"
egltype="UNICODE(1)" fgltype="CHAR(1)" size="1"
isrectype="f" />
  <variable name="state_cnt" library="d4_globals" egltype="INT"

```

```

    fgltype="INTEGER" isrectype="f" />
    <variable name="stock_cnt" library="d4_globals" egltype="INT"
    fgltype="INTEGER" isrectype="f" />
</variables>

<!--Function Declarations-->

<function name="add_customer" package="i4gldemo" library="d4_cust"
type="void" >
    <parameter name="repeat" library="d4_cust" egltype="INT"
    fgltype="INTEGER" size="" start="" end="" precision="" scale=""
    isrectype="f" />
</function>

<function name="add_order" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="add_stock" package="i4gldemo" library="d4_stock"
type="void" >
</function>

<function name="bang" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="clear_menu" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="customer" package="i4gldemo" library="d4_cust"
type="void" >
</function>

<function name="customer_help" package="i4gldemo" l
ibrary="d4_cust"
type="void" >
</function>

<function name="delete_customer" package="i4gldemo"
library="d4_cust" type="void" >
</function>

<function name="delete_order" package="i4gldemo"
library="d4_orders" type="void" >
</function>

<function name="delete_stock" package="i4gldemo"
library="d4_stock"
type="void"> </function>

<function name="demo" package="i4gldemo"
library="d4_demo" type="void" >
</function>

<function name="get_item" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="get_order" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="get_states" package="i4gldemo" library="d4_main"
type="void" >
</function>

```

```

<function name="get_stock" package="i4gldemo" library="d4_orders"
type="void" >
  <return name="stock_num" egltype="smallint"
library="d4_globals" fgltype="smallint" size="" start="" end=""
precision="" scale="" isrectype="f" />
  <return name="manu_code" egltype="unicode(3)"
library="d4_globals" fgltype="char" size="3" start=""
end="" precision="" scale="" isrectype="f" />
  <return name="description" egltype="unicode(15)"
library="d4_globals" fgltype="char" size="15" start=""
end="" precision="" scale="" isrectype="f" />
  <return name="unit_price" egltype="money(6,2)"
library="d4_globals" fgltype="money" size="" start=""
end="" precision="6" scale="2" isrectype="f" />
</function>

<function name="get_stocks" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="input_cust" package="i4gldemo" library="d4_cust"
type="Int" >
  <return name="" egltype="Int" library="" fgltype="Int" size="1"
start="TRUE" end="TRUE" precision="" scale="" isrectype="f" />
</function>

<function name="insert_items" package="i4gldemo" library="d4_orders"
type="Int" >
  <return name="" egltype="Int" library="" fgltype="Int" size="1"
start="TRUE" end="TRUE" precision="" scale="" isrectype="f" />
</function>

<function name="invoice" package="i4gldemo" library="d4_orders"
type="void" >
  <parameter name="file_name" library="d4_orders"
egltype="UNICODE(20)" fgltype="CHAR(20)" size="20" start=""
end="" precision="" scale="" isrectype="f" />
</function>

<function name="item_total" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="MAIN" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="mess" package="i4gldemo" library="d4_main"
type="void" > <parameter name="str" library="d4_main"
egltype="UNICODE(80)" fgltype="CHAR(80)" size="80" start=""
end="" precision="" scale="" isrectype="f" />
  <parameter name="mrow" library="d4_main" egltype="SMALLINT"
fgltype="SMALLINT" size="" start="" end="" precision=""
scale="" isrectype="f" />
</function>

<function name="order_total" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="orders" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="print_ar" package="i4gldemo" library="d4_report"
type="void" >
</function>

```

```

<function name="print_backlog" package="i4gldemo" library="d4_report"
type="void" >
</function>

<function name="print_labels" package="i4gldemo" library="d4_report"
type="void" >
</function>

<function name="print_stock" package="i4gldemo" library="d4_report"
type="void" >
</function>

<function name="query_customer" package="i4gldemo" library="d4_cust"
type="Int" >
  <parameter name="mrow" library="d4_cust" egltype="SMALLINT"
fgltype="SMALLINT" size="" start="" end="" precision="" scale=""
isrectype="f" />
  <return name="" egltype="Int" library="" fgltype="Int" size="1"
start="TRUE" end="TRUE" precision="" scale="" isrectype="f" />
</function>

<function name="query_stock" package="i4gldemo" library="d4_stock"
type="void" >
</function>

<function name="renum_items" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="REPORT" package="i4gldemo" library="d4_report"
type="void" >
  <parameter name="r" library="d4_report" egltype="recordtype_d4_report_r"
fgltype="RECORD customer_num LIKE customer.customer_num, fname LIKE
customer.fname, lname LIKE customer.lname,
company LIKE customer.company, order_num LIKE orders.order_num,
order_date LIKE orders.order_date, ship_date LIKE orders.ship_date,
paid_date LIKE orders.paid_date,
total_price LIKE items.total_price END RECORD" size="" start=""
end="" precision="" scale="" isrectype="t" />
</function>

<function name="reports" package="i4gldemo" library="d4_report"
type="void" >
</function>

<function name="ring_menu" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="statehelp" package="i4gldemo" library="d4_cust"
type="void" >
</function>

<function name="stock" package="i4gldemo" library="d4_stock"
type="void" >
</function>

<function name="unring_menu" package="i4gldemo" library="d4_main"
type="void" >
</function>

<function name="update_customer" package="i4gldemo"
library="d4_cust" type="void" > </function>

<function name="update_options" package="i4gldemo" library="d4_report"
type="void" >

```

```

</function>

<function name="update_order" package="i4gldemo" library="d4_orders"
type="void" >
</function>

<function name="update_stock" package="i4gldemo" library="d4_stock"
type="void" >
</function>
</package>
<forms>
    <form name="custForm.egl" package="i4gldemo.forms" />
    <form name="custcurForm.egl" package="i4gldemo.forms" />
    <form name="custformForm.egl" package="i4gldemo.forms" />
    <form name="customerForm.egl" package="i4gldemo.forms" />
    <form name="ordcurForm.egl" package="i4gldemo.forms" />
    <form name="orderForm.egl" package="i4gldemo.forms" />
    <form name="orderformForm.egl" package="i4gldemo.forms" />
    <form name="p_ordcurForm.egl" package="i4gldemo.forms" />
    <form name="state_listForm.egl" package="i4gldemo.forms" />
    <form name="stocklForm.egl" package="i4gldemo.forms" />
    <form name="stock_selForm.egl" package="i4gldemo.forms" />
</forms>
</manifest>

```

---

## 付録 F. DTD 例

---

### 本付録の内容

この付録では、データベース・スキーマ抽出、共用ライブラリー、およびアプリケーション・プロジェクトの構成ファイルとマニフェスト・ファイルで使用する DTD の例について説明します。

---

### DTD 例の概要

これらの例で使用される DTD は、EGL 変換に対する Informix 4GL ユーティリティー、バージョン 7.1 に特有です。これらの例は、ディレクトリー **plugininstallation/etc/dtd** に配置されています。

以下の項目の DTD 例が記載されています。

- 構成ファイル
  - スキーマ
  - ライブラリー
  - アプリケーション
- マニフェスト・ファイル
  - スキーマ
  - ライブラリー
  - アプリケーション

---

### 構成ファイル

#### データベース・スキーマ抽出プロジェクト

以下の例では、要素 **artifactsdir** で指定された成果物ディレクトリーはオプションです。値が指定されないと、ディレクトリーはデフォルトの **egldir/ConversionArtifacts** ディレクトリーになります。また、**client\_locale** および **db\_locale** 属性は、Informix 固有のロケールであり、デフォルトでは **English** および **en\_US.8859-1** に設定されます。

```
<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT conversion (rootdir,dbconnection*)>
<!--ATTLIST conversion project CDATA #REQUIRED -->
<!--ATTLIST conversion type CDATA #FIXED "schema" -->

<!ELEMENT rootdir (egldir,artifactsdir?)>
<!--ELEMENT egldir (#PCDATA)-->
<!--ELEMENT artifactsdir (#PCDATA)-->

<!--ELEMENT dbconnection (database,server,host,port,user,password)+-->
<!--ATTLIST dbconnection extractSystemTables (yes|no) "no" -->
<!--ATTLIST dbconnection client_locale CDATA #IMPLIED -->
<!--ATTLIST dbconnection db_locale CDATA #IMPLIED -->
<!--ELEMENT database (#PCDATA)-->
```

```

<!ELEMENT server (#PCDATA)>
<!ELEMENT host (#PCDATA)>
<!ELEMENT port (#PCDATA)>
<!ELEMENT user (#PCDATA)>
<!ELEMENT password ANY>

```

## 共用ライブラリー・プロジェクト

以下の例では、**conversion** および **msgfiles** 要素で使用される **locale** 属性は、両方ともデフォルトで **en\_US.8859-1** に設定されます。ライブラリー・プロジェクトがデータベース・スキーマに依存している場合、**defaultserver** 属性は、型 **schema** の要素 **manifestfiles** で指定された従属マニフェスト・ファイルに指定されたサーバー名を含みます。

```

<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT conversion ( rootdir, manifestfiles*, fglfiles?,
    formfiles?, msgfiles*,fontconfigfile? )>
<!--ATTLIST conversion project CDATA #REQUIRED -->
<!--ATTLIST conversion type CDATA #FIXED "library" -->
<!--ATTLIST conversion locale CDATA #IMPLIED -->
    <!--ATTLIST conversion cursor (local | global) #IMPLIED-->
<!--ATTLIST conversion defaultserver CDATA #IMPLIED-->

<!ELEMENT rootdir (fgldir?,egldir,artifactsdir?)>
<!ELEMENT fgldir (#PCDATA)>
<!ELEMENT egldir (#PCDATA)>
<!ELEMENT artifactsdir (#PCDATA)>

<!ELEMENT manifestfiles (file)+>
<!--ATTLIST manifestfiles type ( schema | library) #REQUIRED-->

<!ELEMENT fglfiles (file)+>
<!ELEMENT formfiles (file)+>
<!ELEMENT fontconfigfile (file)>
    <!--ELEMENT file (#PCDATA)-->

<!ELEMENT msgfiles (file)+>
<!--ATTLIST msgfiles locale CDATA #IMPLIED -->
<!--ATTLIST msgfiles encoding CDATA #IMPLIED -->

```

## アプリケーション・プロジェクト

```

<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT conversion ( rootdir, manifestfiles*, fglfiles?,
    formfiles?, msgfiles*,fontconfigfile? )>
<!--ATTLIST conversion project CDATA #REQUIRED -->
<!--ATTLIST conversion type CDATA #FIXED "application" -->
<!--ATTLIST conversion locale CDATA #IMPLIED -->
    <!--ATTLIST conversion cursor (local | global) #IMPLIED-->
<!--ATTLIST conversion defaultserver CDATA #IMPLIED-->

<!ELEMENT rootdir (fgldir?,egldir,artifactsdir?)>
<!ELEMENT fgldir (#PCDATA)>
<!ELEMENT egldir (#PCDATA)>
<!ELEMENT artifactsdir (#PCDATA)>

<!ELEMENT manifestfiles (file)+>
<!--ATTLIST manifestfiles type ( schema | library) #REQUIRED-->
<!--ELEMENT fglfiles (file)+>
<!--ELEMENT formfiles (file)+>
<!--ELEMENT fontconfigfile (file)>
    <!--ELEMENT file (#PCDATA)-->

```

```

<!ELEMENT msgfiles (file)+>
<!--ATTLIST msgfiles locale CDATA #IMPLIED -->
<!--ATTLIST msgfiles encoding CDATA #IMPLIED -->

```

---

## マニフェスト・ファイル

### データベース・スキーマ抽出プロジェクト

```

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE manifest [
  <!--ELEMENT manifest (package*)-->
  <!--ATTLIST manifest project CDATA #REQUIRED -->
  <!--ATTLIST manifest type CDATA #FIXED "schema"-->
  <!--ATTLIST manifest path CDATA #REQUIRED-->
  <!--ATTLIST manifest version CDATA #REQUIRED -->
  <!--ELEMENT package (table*)-->
  <!--ATTLIST package
    name CDATA #REQUIRED
    server CDATA #REQUIRED
    database CDATA #REQUIRED
    mode (ANSI) #IMPLIED
    isolationLevel CDATA #REQUIRED
    commitControl CDATA #REQUIRED-->
  <!--ELEMENT table (column*)-->
  <!--ATTLIST table
    name CDATA #REQUIRED
    egltype CDATA #REQUIRED
    owner CDATA #IMPLIED-->
  <!--ELEMENT column EMPTY-->
  <!--ATTLIST column
    name CDATA #REQUIRED
    dataitem CDATA #REQUIRED
    fgltype CDATA #REQUIRED
    egltype CDATA #REQUIRED
    size CDATA #IMPLIED
    start CDATA #IMPLIED
    end CDATA #IMPLIED
    precision CDATA #IMPLIED
    scale CDATA #IMPLIED-->

```

### ライブラリー・プロジェクト

```

<?xml version="1.0" encoding="UTF-8"?>

<!--ELEMENT manifest (package,forms*)-->
<!--ATTLIST manifest project CDATA #REQUIRED-->
<!--ATTLIST manifest type CDATA #FIXED "library"-->

<!--ELEMENT package (rectype*, variables*,function*,
  cfunc*,cursor*,preparedStatements*)-->
<!--ATTLIST package name CDATA #REQUIRED-->

<!--ELEMENT function (parameter*,return*)-->
<!--ATTLIST function name CDATA #REQUIRED-->
<!--ATTLIST function library CDATA #REQUIRED-->
<!--ATTLIST function type CDATA #REQUIRED-->

<!--ELEMENT parameter EMPTY-->
<!--ATTLIST parameter name CDATA #REQUIRED-->
<!--ATTLIST parameter egltype CDATA #REQUIRED-->
<!--ATTLIST parameter fgltype CDATA #IMPLIED-->
<!--ATTLIST parameter size CDATA #IMPLIED-->
<!--ATTLIST parameter precision CDATA #IMPLIED-->

```

```

<!ATTLIST parameter scale CDATA #IMPLIED>
<!ATTLIST parameter start CDATA #IMPLIED>
<!ATTLIST parameter end CDATA #IMPLIED>
<!ATTLIST parameter isrectype (t|f) "f">
<!ATTLIST parameter library CDATA #REQUIRED>

<!ELEMENT return EMPTY>
  <!ATTLIST return name CDATA #REQUIRED>
  <!ATTLIST return egltype CDATA #REQUIRED>
  <!ATTLIST return fgltype CDATA #IMPLIED>
  <!ATTLIST return size CDATA #IMPLIED>
  <!ATTLIST return precision CDATA #IMPLIED>
  <!ATTLIST return scale CDATA #IMPLIED>
  <!ATTLIST return start CDATA #IMPLIED>
  <!ATTLIST return end CDATA #IMPLIED>
  <!ATTLIST return isrectype (t|f) "f" >
  <!ATTLIST return library CDATA #REQUIRED>

<!ELEMENT cfunc EMPTY>
  <!ATTLIST cfunc name CDATA #REQUIRED>
  <!ATTLIST cfunc library CDATA #REQUIRED>
  <!ATTLIST cfunc argcount CDATA #REQUIRED>
  <!ATTLIST cfunc retcount CDATA #REQUIRED>

<!ELEMENT cursor EMPTY>
  <!ATTLIST cursor name CDATA #REQUIRED>
  <!ATTLIST cursor ishold (t|f) "f" >
  <!ATTLIST cursor isscrolling (t|f) "f" >
  <!ATTLIST cursor library CDATA #REQUIRED>

<!ELEMENT variables (variable*)>

<!ELEMENT variable EMPTY>
  <!ATTLIST variable name CDATA #REQUIRED>
  <!ATTLIST variable egltype CDATA #REQUIRED>
  <!ATTLIST variable fgltype CDATA #IMPLIED>
  <!ATTLIST variable size CDATA #IMPLIED>
  <!ATTLIST variable precision CDATA #IMPLIED>
  <!ATTLIST variable scale CDATA #IMPLIED>
  <!ATTLIST variable start CDATA #IMPLIED>
  <!ATTLIST variable end CDATA #IMPLIED>
  <!ATTLIST variable isrectype (t|f) "f" >
  <!ATTLIST variable library CDATA #REQUIRED>
  <!ATTLIST variable numsubscripts CDATA #IMPLIED>

<!ELEMENT rectype (field*)>
  <!ATTLIST rectype name CDATA #REQUIRED>
  <!ATTLIST rectype library CDATA #REQUIRED>

<!ELEMENT field EMPTY>
  <!ATTLIST field name CDATA #REQUIRED>
  <!ATTLIST field egltype CDATA #REQUIRED>
  <!ATTLIST field fgltype CDATA #IMPLIED>
  <!ATTLIST field size CDATA #IMPLIED>
  <!ATTLIST field precision CDATA #IMPLIED>
  <!ATTLIST field scale CDATA #IMPLIED>
  <!ATTLIST field start CDATA #IMPLIED>
  <!ATTLIST field end CDATA #IMPLIED>
  <!ATTLIST field isrectype (t|f) "f" >
  <!ATTLIST field library CDATA #REQUIRED>
  <!ATTLIST field numsubscripts CDATA #IMPLIED>

```

```

<!ELEMENT preparedStatements (statement*)>
<!ELEMENT statement EMPTY>
  <!--ATTLIST statement name CDATA #REQUIRED-->
  <!--ATTLIST statement library CDATA #REQUIRED-->

<!ELEMENT forms (form*)>
<!ELEMENT form EMPTY>
  <!--ATTLIST form name CDATA #REQUIRED-->
  <!--ATTLIST form package CDATA #REQUIRED-->

```

## アプリケーション・プロジェクト

```

<?xml version="1.0" encoding="UTF-8"?>

<!ELEMENT manifest (package, forms*)>
  <!--ATTLIST manifest project CDATA #REQUIRED-->
  <!--ATTLIST manifest type CDATA #FIXED "application"-->

<!ELEMENT package (rectype*, variables*, function*,
  cfunc*, cursor*, preparedStatements*)>
  <!--ATTLIST package name CDATA #REQUIRED-->

<!ELEMENT function (parameter*, return*)>
  <!--ATTLIST function name CDATA #REQUIRED-->
  <!--ATTLIST function library CDATA #REQUIRED-->
  <!--ATTLIST function type CDATA #REQUIRED-->

<!ELEMENT parameter EMPTY>
  <!--ATTLIST parameter name CDATA #REQUIRED-->
  <!--ATTLIST parameter egltype CDATA #REQUIRED-->
  <!--ATTLIST parameter fgltype CDATA #IMPLIED-->
  <!--ATTLIST parameter size CDATA #IMPLIED-->
  <!--ATTLIST parameter precision CDATA #IMPLIED-->
  <!--ATTLIST parameter scale CDATA #IMPLIED-->
  <!--ATTLIST parameter start CDATA #IMPLIED-->
  <!--ATTLIST parameter end CDATA #IMPLIED-->
  <!--ATTLIST parameter isrectype (t|f) "f"-->
  <!--ATTLIST parameter library CDATA #REQUIRED-->

<!ELEMENT return EMPTY>
  <!--ATTLIST return name CDATA #REQUIRED-->
  <!--ATTLIST return egltype CDATA #REQUIRED-->
  <!--ATTLIST return fgltype CDATA #IMPLIED-->
  <!--ATTLIST return size CDATA #IMPLIED-->
  <!--ATTLIST return precision CDATA #IMPLIED-->
  <!--ATTLIST return scale CDATA #IMPLIED-->
  <!--ATTLIST return start CDATA #IMPLIED-->
  <!--ATTLIST return end CDATA #IMPLIED-->
  <!--ATTLIST return isrectype (t|f) "f" -->
  <!--ATTLIST return library CDATA #REQUIRED-->

<!ELEMENT cfunc EMPTY>
  <!--ATTLIST cfunc name CDATA #REQUIRED-->
  <!--ATTLIST cfunc library CDATA #REQUIRED-->
  <!--ATTLIST cfunc argcount CDATA #REQUIRED-->
  <!--ATTLIST cfunc retcount CDATA #REQUIRED-->

<!ELEMENT cursor EMPTY>
  <!--ATTLIST cursor name CDATA #REQUIRED-->
  <!--ATTLIST cursor ishold (t|f) "f" -->
  <!--ATTLIST cursor isscrolling (t|f) "f" -->
  <!--ATTLIST cursor library CDATA #REQUIRED-->

```

```

<!ELEMENT variables (variable*)>

<!ELEMENT variable EMPTY>
  <!--ATTLIST variable name CDATA #REQUIRED-->
  <!--ATTLIST variable egltype CDATA #REQUIRED-->
  <!--ATTLIST variable fgltype CDATA #IMPLIED-->
  <!--ATTLIST variable size CDATA #IMPLIED-->
  <!--ATTLIST variable precision CDATA #IMPLIED-->
  <!--ATTLIST variable scale CDATA #IMPLIED-->
  <!--ATTLIST variable start CDATA #IMPLIED-->
  <!--ATTLIST variable end CDATA #IMPLIED-->
  <!--ATTLIST variable isrectype (t|f) "f" -->
  <!--ATTLIST variable library CDATA #REQUIRED-->
  <!--ATTLIST variable numsubscripts CDATA #IMPLIED-->

<!ELEMENT rectype (field*)>
  <!--ATTLIST rectype name CDATA #REQUIRED-->
  <!--ATTLIST rectype library CDATA #REQUIRED-->

<!ELEMENT field EMPTY>
  <!--ATTLIST field name CDATA #REQUIRED-->
  <!--ATTLIST field egltype CDATA #REQUIRED-->
  <!--ATTLIST field fgltype CDATA #IMPLIED-->
  <!--ATTLIST field size CDATA #IMPLIED-->
  <!--ATTLIST field precision CDATA #IMPLIED-->
  <!--ATTLIST field scale CDATA #IMPLIED-->
  <!--ATTLIST field start CDATA #IMPLIED-->
  <!--ATTLIST field end CDATA #IMPLIED-->
  <!--ATTLIST field isrectype (t|f) "f" -->
  <!--ATTLIST field library CDATA #REQUIRED-->
  <!--ATTLIST field numsubscripts CDATA #IMPLIED-->

<!ELEMENT preparedStatements (statement*)>
<!ELEMENT statement EMPTY>
  <!--ATTLIST statement name CDATA #REQUIRED-->
  <!--ATTLIST statement library CDATA #REQUIRED-->

<!ELEMENT forms (form*)>
<!ELEMENT form EMPTY>
  <!--ATTLIST form name CDATA #REQUIRED-->
  <!--ATTLIST form package CDATA #REQUIRED-->

```

---

## 付録 G. 変換ログの例

---

### 本付録の内容

この付録では、**.txt** フォーマットの変換ログの例を示します。

---

### アプリケーション・プロジェクト: PASSED 例

```
-----
                          i4gldemo Conversion Log
-----

Conversion Status:
-----
Project Status           : PASSED
Conversion date          : Wed Feb 02 22:22:53 CST 2005
User                     : jdoe
Host                      : boom
OS version                : Windows XP

Project Details:
-----
Project Name              : i4gldemo
Conversion Type            : application
4GL root directory        : C:\i4gl\i4gldemo
EGL destination directory : C:\workspace\i4gldemo
Conversion artifacts directory : C:\temp\i4gldemo\ConversionArtifacts
Configuration file         : C:\i4gl\config\i4gldemoConfig.xml
Default Informix server instance: myserver
Informix cursor scope      : local

Conversion Artifacts:
-----
Manifest File Generated   : C:\temp\i4gldemo\ConversionArtifacts\manifest\
i4gldemoApplicationManifest.xml
EGL Build descriptor file: C:\temp\i4gldemo\i4gldemo\EGLSources\i4gldemo.egl.bld

I4GL Source File Conversion Summary:
-----
Total number of I4GL files given: 19
Total number of files converted successfully: 19

d4_cust.4gl -> PASSED
d4_demo.4gl -> PASSED
d4_globals.4gl -> PASSED
d4_load.4gl -> PASSED
d4_main.4gl -> PASSED
d4_orders.4gl -> PASSED
d4_report.4gl -> PASSED
d4_stock.4gl -> PASSED
cust.per -> PASSED
custcur.per -> PASSED
custform.per -> PASSED
customer.per -> PASSED
ordcur.per -> PASSED
order.per -> PASSED
orderform.per -> PASSED
p_ordcur.per -> PASSED
state_list.per -> PASSED
stock1.per -> PASSED
```

stock\_sel.per -> PASSED

Source File Conversion Details:

-----

4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_cust.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_cust.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_demo.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_demo.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_globals.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_globals.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_load.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_load.egl
	: C:\workspace\i4gldemo\d4_load_program.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_main.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_main.egl
	: C:\workspace\i4gldemo\d4_main_program.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_orders.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_orders.egl
	: C:\workspace\i4gldemo\r_invoice_handler.egl
	: C:\workspace\i4gldemo\r_invoice_XML.jrxml
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_report.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_report.egl
	: C:\workspace\i4gldemo\labels_report_handler.egl
	: C:\workspace\i4gldemo\labels_report_XML.jrxml
	: C:\workspace\i4gldemo\ar_report_handler.egl
	: C:\workspace\i4gldemo\ar_report_XML.jrxml
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_stock.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_stock.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\cust.per
EGL source file generated	: C:\workspace\i4gldemo\forms\custForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\custcur.per
EGL source file generated	: C:\workspace\i4gldemo\forms\custcurForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\custform.per
EGL source file generated	: C:\workspace\i4gldemo\forms\custformForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\customer.per
EGL source file generated	: C:\workspace\i4gldemo\forms\customerForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\ordcur.per
EGL source file generated	: C:\workspace\i4gldemo\forms\ordcurForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\order.per
EGL source file generated	: C:\workspace\i4gldemo\forms\orderForm.egl

Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\orderform.per
EGL source file generated	: C:\workspace\i4gldemo\forms\orderformForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\p_ordcur.per
EGL source file generated	: C:\workspace\i4gldemo\forms\p_ordcurForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\state_list.per
EGL source file generated	: C:\workspace\i4gldemo\forms\state_listForm.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\stock1.per
EGL source file generated	: C:\workspace\i4gldemo\forms\stock1Form.egl
Status	: PASSED
4GL source file given	: C:\i4gl\i4gldemo\fgl\forms\stock_sel.per
EGL source file generated	: C:\workspace\i4gldemo\forms\stock_selForm.egl
Status	: PASSED

---

## アプリケーション・プロジェクト: FAILED 例

---

### i4gldemo Conversion Log

---

#### Conversion Status:

---

Project Status	: FAILED
Conversion date	: Wed Feb 02 22:22:53 CST 2005
User	: jdoe
Host	: boom
OS version	: Windows XP

#### Project Details:

---

Project Name	: i4gldemo
Conversion Type	: application
4GL root directory	: C:\i4gl\i4gldemo
EGL destination directory	: C:\workspace\i4gldemo
Conversion artifacts directory	: C:\temp\i4gldemo\ConversionArtifacts
Configuration file	: C:\i4gl\config\i4gldemoConfig.xml
Default Informix server instance	: myserver
Informix cursor scope	: local

#### I4GL Source File Conversion Summary:

---

Total number of I4GL files given: 1  
Total number of files converted successfully: 0

d4\_cust.4gl -> FAILED

#### Source File Conversion Details:

---

4GL source file given	: C:\i4gl\i4gldemo\fgl\d4_cust.4gl
EGL source file generated	: C:\workspace\i4gldemo\d4_cust.egl
Status	: FAILED

---

## データベース・スキーマ抽出プロジェクト: FAILED 例

```
-----
                          Stores7 Conversion Log
-----

Conversion Status :
-----
Project Status           : FAILED
Conversion date          : Thu Feb 03 11:25:33 CST 2005
User                    : jdoe
Host                    : boom
OS version               : Windows XP

Project Details :
-----
Project Name             : Stores7
Conversion Type           : schema
EGL destination directory : C:\workspace\stores7
Conversion artifacts directory : C:\workspace\stores7\
ConversionArtifacts
Configuration file        : C:\workspace\stores7\
conversionArtifacts\config\Stores7SchemaConfig.xml

Database Connection details:
-----

Database                 : stores7
Server                   : myserver
Host                     : boom.antartica.world.com
Port                     : 2005
User                     : jdoe
CLIENT_LOCALE            : en_us.8859-1
DB_LOCALE                : en_us.8859-1

Exceptions:
-----

ERROR :   Server : "myserver" Database : "stores7"

Informix JDBC Exception :
java.sql.SQLException: User (com.informix.asf.IfxAASRemoteException jdoe)'s
password is not correct for the database server.
```

---

## 付録 H. EGL ビルド記述子の例

---

### 本付録の内容

この付録では、変換ユーティリティによって生成された EGL ビルド記述子ファイルの例を説明します。

---

### EGL ビルド記述子の概要

変換ユーティリティは、変換されたプロジェクトごとにビルド記述子ファイルを 1 つ生成します。生成されたファイルは、変換された I4GL ファイルに関連するプロパティのみに値を提供します。

---

### データベース・スキーマ抽出プロジェクト

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE EGL PUBLIC "-//IBM Corporation, Inc.//DTD EGL Build Parts
6.0//EN" "">
<EGL>
<BuildDescriptor
  name="Stores7JavaBuildOptions"
  genProject="Stores7"
  genDirectory="C:¥temp¥Stores7¥EGLSource¥Stores7"
  system="WIN"
  J2EE="NO"
  sqlCommitControl="AUTOCOMMIT"
  itemsNullable="YES"
  genProperties="GLOBAL"
  genDataTables="YES"
  dbms="INFORMIX"
  sqlValidationConnectionURL="jdbc:informix-sqli://
mymachine.loc.comp.com:2005/stores7:
INFORMIXSERVER=myserver;"
  sqlJDBCClass="com.informix.jdbc.IfxDriver" sqlID="jdoe"
  sqlPassword="password" sqlDB="jdbc:informix-sqli://
mymachine.loc.comp.com:2005/stores7:
INFORMIXSERVER=myserver;" >
</BuildDescriptor>
</EGL>
```

---

### ライブラリーまたはアプリケーション・プロジェクト

変換ユーティリティは、ライブラリーまたはアプリケーション・プロジェクトのユーザー名またはパスワードの値へのアクセス権を持っていません。これらの 2 つのプロジェクト・タイプの場合、変換ユーティリティは、ビルド記述子ファイルにプレースホルダー接続 URL を生成します。この接続 URL は、ビルド・ファイルを使用する前に編集する必要があります。また、**sqlID** および **sqlPassword** 値も編集する必要があります。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE EGL PUBLIC "-//IBM Corporation, Inc.//DTD EGL Build Parts
6.0//EN" "">
<EGL>
<BuildDescriptor
```

```

name="i4gldemoJavaBuildOptions"
genProject="i4gldemo"
genDirectory="C:\temp\i4gldemo\i4gldemo\EGLSources\i4gldemo"
system="WIN"
J2EE="NO"
sqlCommitControl="AUTOCOMMIT"
itemsNullable="YES"
genProperties="GLOBAL"
genDataTables="YES"
dbms="INFORMIX"
sqlValidationConnectionURL="jdbc:informix-sqli://host:port/
database:INFORMIXSERVER=server;"
sqlJDBCClass="com.informix.jdbc.IfxDriver" sqlID="user"
sqlPassword="password" sqlDB="jdbc:informix-sqli://host:port/
database:INFORMIXSERVER=server;" >
</BuildDescriptor>
</EGL>

```

---

## 付録 I. EGL 予約語

---

### 本付録の内容

この付録では、バージョン 7.1 の EGL 予約語をリストしています。EGL 予約語の最新リストについては、オンライン・ヘルプ・トピックの『EGL 予約語』を参照してください。

---

### EGL 予約語

以下の語が EGL で予約されています。

- absolute、add、all、any、as
- bigInt、bin、bind、blob、boolean、by、byName、byPosition
- call、case、char、clob、close、const、constructor、continue、converse、current
- dataItem、dataTable、date、dbChar、decimal、decrement、delegate、delete、display、dliCall
- else、embed、end、enumeration、escape、execute、exit、extends、externallyDefined、externalType
- false、field、first、float、for、forEach、form、formGroup、forUpdate、forward、freeSql、from、function
- get、goto
- handler、hex、hold
- if、import、implements、in、inOut、inparent、insert、int、interface、interval、into、is、isa
- label、languageBundle、last、library、like
- matches、mathlib、mbChar、money、move
- new、next、nil、no、noRefresh、not、null、nullable、num、number、numc
- onEvent、onException、open、openUI、otherwise、out
- pacf、package、pageHandler、passing、prepare、previous、print、private、program、psb
- record、ref、relative、replace、report、return、returning、returns、rununit
- scroll、self、service、serviceReferences、set、show、singleRow、smallFloat、smallInt、sql、sqlCondition、stack、static、string、strlib、syslib、sysvar
- this、time、timeStamp、to、transaction、transfer、true、try、type
- unicode、update、url、use、using、usingKeys、usingPCB
- when、while、with、withinParent
- yes



## 用語集

### 【ア行】

**インフォメーション・センター (Information Center).** Rational 製品および EGL に関する詳細情報を提供するオンライン・ヘルプ。「ヘルプ」 > 「Rational ヘルプ」を選択して、インフォメーション・センターにアクセスする。

**ウィザード (Wizard).** 変換ユーティリティ・ウィザードを使用することで、I4GL アプリケーションの変換に必要なステップを順に実行できる。「ファイル」 > 「新規作成」 > 「その他」 > 「EGL 変換に対する Informix 4GL」を選択して、ウィザードにアクセスする。

### 【カ行】

**構成ファイル (Configuration file).** *EGLDestinationDirectory/ConversionArtifacts/config* ディレクトリーに配置。このファイルによって構成の詳細が分かる。

**コマンド行変換 (Command line conversion).** 変換ユーティリティ・ウィザードと同等のコマンド行。このプログラムを実行するには、手動で構成ファイルを作成し、**e4GL** スクリプトを実行する必要がある。コマンド行変換は、主に再変換作業で推奨される。

**コンソール・ユーザー・インターフェース (Console User Interface (CUI)).** EGL で 4GL 書式に相当するものの。

### 【サ行】

**再変換 (Reconversion).** 共用ライブラリーが正常に変換されない場合は、変換ユーティリティ・ウィザードまたはコマンド行のどちらかで再変換する必要がある。

### 【ハ行】

**変換成果物 (Conversion artifacts).** 変換中に生成される構成ファイル、マニフェスト・ファイル、および変換ログ・ファイル。

**変換ユーティリティ・ウィザード (Conversion Utility Wizard).** このウィザードによって、データベース・スキーマ、共用ライブラリー、および I4GL アプリケーションの情報が収集され、変換プロセスを起動する。

**変換ログ (Conversion log).** *EGLDestinationDirectory/ConversionArtifacts*

*/log* ディレクトリーに配置。*ProjectName.log* という名前が付く。変換ログは、変換エラー、警告、およびファイル属性指定を判別する。このファイルは、変換エラーを訂正する方法を判別するために使用する。

### 【マ行】

**マニフェスト・ファイル (Manifest file).**

*EGLDestinationDirectory/ConversionArtifacts/manifest* ディレクトリーに配置。マニフェスト・ファイルは各変換ステージで生成される。データベース・スキーマのマニフェスト・ファイルには、選択されたデータベースのすべてのテーブル、列、およびデータ型に関する情報が含まれる。共用ライブラリーのマニフェスト・ファイルには、I4GL 変換プロジェクトで使用される I4GL および想定される C 関数呼び出しがリストされる。アプリケーション・マニフェスト・ファイルは、プロジェクトの技術詳細のリストを提供する。

### 【ワ行】

**ワークスペース (Workspace).** Rational 製品では、ワークスペースがデータ・ファイルの中心となるハブである。ワークスペースのリソースは、上位にプロジェクト、その下位にフォルダーやファイルのツリー構造で編成される。

## C

**ConversionArtifacts ディレクトリー (ConversionArtifacts directory).**

*EGLDestinationDirectory* に配置。このディレクトリーには、構成ファイル、マニフェスト・ファイル、およびログ・ファイルのサブディレクトリーが含まれる。

## E

**E4GL.** I4GL ファイルから EGL へのコマンド行変換のためのスクリプト・プログラム。**E4GL** は、**I4GL2EGL** 変換ユーティリティをアクティブにする。

**EGL.** 開発者が、Web を含めたいくつかの環境でデプロイする複雑なビジネス・アプリケーションを作成するときに、ビジネス・ロジックに焦点を置くことのできる高水準言語。この言語を使用することで、データベース、メッセージ・キュー・アクセス、および J2EE の使用が単純化される。

**EGL パッケージ (EGL package).** EGL パッケージは、関連ソース・パーツの名前付きコレクションであり、I4GL プロジェクトに相当する。I4GL から EGL への変換中、I4GL プロジェクト・コンポーネントは、EGL パッケージ・パーツに変換される。

**EGL プロジェクト (EGL project).** EGL プロジェクトには、ソース・フォルダーがゼロから多数まで含まれる。それぞれのフォルダーには、パッケージがゼロから多数まで含まれる。それぞれのパッケージには、ファイルがゼロから多数まで含まれる。それぞれのファイルには、パーツがゼロから多数まで含まれる。

## I

**I4GL2EGL.** I4GL ソース・ファイルを EGL ソース・ファイルに変換するプログラム。

## J

**JasperReports.** Java で作成され、EGL がレポートを作成するために使用するオープン・ソースのレポート作成ライブラリー。I4GL レポートは、**.egl** および JasperReport **.jrxml** ファイルの両方に変換される。

## R

**readme004FGL.html.** 変換ユーティリティのプラグイン・ディレクトリーに配置。**readme004FGL.html** ファイルには、変換ユーティリティ制限、プロシージャー、および本ユーザーズ・ガイドの完成後の資料への変更事項の情報が含まれる。

**readme.html.** Rational 製品のトップ・ディレクトリーに配置。**readme.html** ファイルには、製品制限の情報が含まれる。

---

## エラー・メッセージ

このエラー・メッセージ・セクションでは、変換ユーティリティーおよび FGL パーサーのエラー・メッセージの注釈説明とユーザー応答を解説します。

変換ユーティリティーが、JRE または Rational 製品の問題が原因で強制終了することがあります。そのような場合は、エラー・メッセージやログ・ファイルを作成せずに変換処理が終了します。JRE または Rational 製品で問題を解決した上で、変換処理を最初から開始する必要があります。

---

### 変換ユーティリティーのエラー・メッセージ

---

**構成ファイルの名前が提供されていません。**

**説明:** ウィザードまたはコマンド行で構成ファイルの名前が入力されていないか、正しく入力されていませんでした。

**ユーザーの処置:** 構成ファイルの正しい名前を入力します。

---

**構成ファイルが読み取りできないか、無効なフォーマットです。**

**説明:** 構成ファイルが読み取りできないか、無効なフォーマットです。

**ユーザーの処置:** 構成ファイルのフォーマットを訂正します。

---

**マニフェスト・ファイルが読み取りできないか、無効なフォーマットです。**

**説明:** マニフェスト・ファイルが読み取りできないか、無効なフォーマットです。

**ユーザーの処置:** マニフェスト・ファイルのフォーマットを訂正します。

---

**変換成果物ディレクトリーを作成できません。**

**説明:** 変換成果物ディレクトリーを作成できません。

**ユーザーの処置:** 成果物ディレクトリーを作成するために、ファイル・システムの書き込み許可をチェックします。

---

**I4GL ルート・ディレクトリーが存在しないか、または読み取れません。**

**説明:** I4GL ルート・ディレクトリーが存在しないか、または読み取れません。

**ユーザーの処置:** 正しい I4GL ルート・ディレクトリーを指定します。

---

**I4GL ソース・ファイルが存在しないか、または読み取れません。**

**説明:** I4GL ソース・ファイルが存在しないか、または読み取れません。

**ユーザーの処置:** I4GL ソース・ファイルのパスと読み取り許可をチェックします。

---

**I4GL フォーム・ファイルが存在しないか、または読み取れません。**

**説明:** I4GL フォーム・ファイルが存在しないか、または読み取れません。

**ユーザーの処置:** I4GL フォーム・ファイルのパスと読み取り許可をチェックします。

---

**I4GL メッセージ・ファイルが存在しないか、または読み取れません。**

**説明:** I4GL メッセージ・ファイルが存在しないか、または読み取れません。

**ユーザーの処置:** I4GL メッセージ・ファイルのパスと読み取り許可をチェックします。

---

**EGL 宛先ディレクトリーを作成できません。**

**説明:** EGL 宛先ディレクトリーを作成できません。

**ユーザーの処置:** 十分なディスク・スペースと書き込み許可があることを検証します。

---

データベース接続情報が構成ファイルで検出できません。

**説明:** 構成ファイルに Informix データベースへの接続方法に関する情報が含まれていません。

**ユーザーの処置:** 構成ファイルにデータベース接続情報が含まれていることを検証してください。構成ファイルのデータベース・セクションには、DTD に準拠した有効な XML 要素が存在しなくてはなりません。

---

**EGL ソース・ファイルを作成できません。**

**説明:** EGL ソース・ファイルを作成できません。

**ユーザーの処置:** EGL 宛先ディレクトリーの書き込み許可とディスク・スペースをチェックします。

---

**マニフェスト・ファイルを作成できません。**

**説明:** マニフェスト・ファイルを作成できません。

**ユーザーの処置:** **ConversionArtifacts/manifest** ディレクトリーのパスおよび書き込み許可をチェックします。

---

**I4GL フォーム・ファイルで構文エラーが検出されました。**

---

## FGL パーサー・メッセージ

---

配列ディメンションに **IDENTIFIER** が見つかりました。定義が必要である可能性があります。

**説明:** I4GL では、配列ディメンション宣言で変数を許可していません。したがって、I4GL ファイルはプリプロセッサ・ディレクティブを使用して、I4GL コンパイラーが I4GL ファイルを C に変換する際に発生するこの問題を解決する必要があります。

**ユーザーの処置:** **IDENTIFER** は、EGL で最後の INT 変数として作成され、値が割り当てられる必要があります。

---

**戻りの型が不明です。**

**説明:** 関数が戻した値を判別することができませんでした。

**ユーザーの処置:** 宣言された関数の戻りの型を EGL で訂正します。

---

**カーソル宣言が見つかりません。**

**説明:** カーソルは開きましたが、カーソルを、準備済みステートメントおよび **HOLD/SCROLL** 属性に関連付ける宣言が見つかりませんでした。

**説明:** I4GL フォーム・ファイルで構文エラーが検出されました。

**ユーザーの処置:** 構文エラーが検出された行および列番号情報のためにエラー・ファイルが作成されていることを検証してください。このファイルは、有効な I4GL フォーム・ファイルである必要があります。フォーム・ファイルを I4GL **form4gl** 書式コンパイラーでコンパイルして、エラー・メッセージの詳細を識別します。

---

**変換ログ・ファイルを作成できません。**

**説明:** 変換ログ・ファイルを作成できません。

**ユーザーの処置:** **ConversionArtifacts/log** サブディレクトリーのパスおよび書き込み許可をチェックします。

---

**ディレクトリーが存在しないか、書き込み許可がありません。**

**説明:** ディレクトリーが存在しないか、書き込み許可がありません。

**ユーザーの処置:** ディレクトリーのパスおよび書き込み許可をチェックします。

---

**ユーザーの処置:** I4GL プログラムからカーソルの **HOLD/SCROLL** 特性を判別し、いずれかを指定する場合、その属性を EGL オープン・ステートメントに追加します。

---

**書式の動的仕様はサポートされていません。**

**説明:** 書式は動的にロードできません。書式の Local/Global インスタンスを宣言する必要があります。変数を介して書式を参照するステートメントは、その変数の値と等しい名前を持つ書式の既存インスタンスがあれば機能します。

**ユーザーの処置:** Local/Global インスタンスを宣言します。

---

**非サポート・ステートメント・エラー。**

**説明:** 指定した行のステートメントが、I4GL から EGL への変換中、または EGL 言語によってサポートされていません。

**ユーザーの処置:** 変換は不可能です。必要であれば、機能を EGL で再書き込みします。

---

**C コンパイラー・ディレクティブが検出されました。**

**説明:** I4GL は、C コンパイラーで評価される中間 ESQL/C ファイルにインクルードされる I4GL ファイルに、C コンパイラー・ディレクティブを組み込むことを許可します。これらの C コンパイラー・ディレクティブは、I4GL から EGL への変換中はサポートされません。また、通常 EGL でもサポートされていません。

**ユーザーの処置:** 変換が、C コンパイラー・ディレクティブを無視しました。ディレクティブに関連するコードを検証し、訂正してください。

---

**想定された C コンパイラー・ディレクティブが検出されましたが、行の先頭にありません。**

**説明:** I4GL は、C コンパイラーに送信される中間ファイルにインクルードされる I4GL ファイルに、C コンパイラー・ディレクティブを組み込むことを許可します。これらの C コンパイラー・ディレクティブは、I4GL から EGL への変換中はサポートされません。また、通常 EGL でもサポートされていません。ディレクティブは、通常、列 1 から開始しますが、この場合はそうではありません。

**ユーザーの処置:** 変換が、C コンパイラー・ディレクティブを無視しました。ディレクティブに関連するコードを検証し、訂正してください。

---

**未定義の変数 "{0}"。**

**説明:** 指定された変数の定義が検出できませんでした。これにより、型情報が要求されるとき、変数を参照するコードを正しく生成できなくなります。

**ユーザーの処置:** 生成された変数名をチェックし、それが予約語または I4GL 変数からマップされたか、あるいはグローバル変数であったかを判別します。変換プロジェクト定義を訂正し、必要とするグローバル・ライブラリーまたは欠落ファイルを組み込みます。

---

**"WITH REOPTIMIZATION" はサポートされていません。**

**説明:** 4GL/ESQL/C は、カーソルを開いているときに 'WITH REOPTIMIZATION' の指定を許可しました。JDBC は、これをサポートしていないので、そのオプションは無視されます。ホスト変数パラメーターが指定されている SQL ステートメントは、通常、カーソルが開いているごとに、サーバーによって再度最適化されます。

**ユーザーの処置:** ターゲット・サーバーで発生する可能性があるパフォーマンス上の影響を考えます。影響が深刻な場合は、コードを再書き込みしてステートメントを再度準備すると、実行を再度最適化できます。

---

**"With Concurrent Transactions" がサポートされる唯一のモードです。**

**説明:** JDBC では、並行トランザクションを実行することに制限はありません。保留中のトランザクションがあるアプリケーションが、異なる接続で新規のトランザクションを開始する場合、実行時にエラーは生成されません。

**ユーザーの処置:** この状態を回避する必要がある場合は、追加のチェック事項をプログラムに書き込み、保留中のトランザクションがある場合に接続が切り替わらないようにします。**disconnect()** 関数は、トランザクションが保留中の場合、例外を戻します。

---

**非インプリメント。**

**説明:** ステートメント内の指定したロケーションのオプションが、I4GL から EGL への変換中、または EGL 言語によってサポートされていません。

**ユーザーの処置:** EGL 機能を使用してコードを再書き込みします。

---

**重複関数: "{0}" はローカルおよび外部プロジェクト "{1}" で定義されています。**

**説明:** インポートされたプロジェクトに、指定された関数名がすでに定義されていました。このエラーの原因として、I4GL ファイルが同じ変換プロジェクトに間違っ

**ユーザーの処置:** I4GL ファイルが同じ変換プロジェクトに間違っ

---

**定義される前に別のプロジェクトで参照された関数 "{0}"。プロジェクト "{1}" の再変換が必要です。**

**説明:** インポートされたプロジェクトで参照された、以前定義されなかった関数は、ここで定義されました。

**ユーザーの処置:** インポートされたプロジェクトを再変換し、コード生成が正しいかを確認します。

---

**変数 "{1}" の未定義の型 "{0}"**

**説明:** 指定したレコード・タイプが SCHEMA マニフェストで検出されませんでした。指定した変数には必要です。これは、アプリケーションが ANSI データベースを使用し、現行ユーザーが ANSI テーブルの所有者であ

ると想定して、ソースが書き込まれている場合に発生します。

**ユーザーの処置:** 所有者をテーブル参照に追加するか、SCHEMA マニフェストから所有者名を除去し、変換を再試行してください。

---

**エラー: 無効なマニフェスト・タイプ**

**説明:** マニフェスト・タイプが、マニフェストの既知の DTD 定義と一致しないか、定義を識別しません。

**ユーザーの処置:** 関連ライブラリーを再生成してマニフェスト・ファイルを再作成するか、DTD 参照を訂正します。

---

**エラー: マニフェスト・ファイルの再変換に失敗しました。**

**説明:** 従属プロジェクト・ファイルの更新中に内部例外が発生しました。

**ユーザーの処置:** 外部原因を訂正し、変換を再実行してください。外部原因が訂正できない場合は、IBM サポートにお問い合わせください。

---

**エラー: プロジェクト {0} は再変換の必要があります。マニフェスト {1} が更新されました。**

**説明:** 指定済みプロジェクトが、現行プロジェクトで定義された関数を参照しています。

**ユーザーの処置:** 指定済みプロジェクトを再変換し、コード生成が正しいことを確認します。

---

**エラー: マニフェスト・ファイルの生成に失敗しました。**

**説明:** プロジェクト・マニフェストを保管中に例外が発生しました。

**ユーザーの処置:** 外部原因を訂正し、変換を再実行してください。外部原因が訂正できない場合は、IBM サポートにお問い合わせください。

---

**エラー: ネイティブ・ライブラリーの生成ができません**

**説明:** C ネイティブ・ライブラリー・コードを書き込み中に例外が発生しました。

**ユーザーの処置:** 外部原因を訂正し、変換を再実行してください。外部原因が訂正できない場合は、IBM サポートにお問い合わせください。

---

## 特記事項

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-8711  
東京都港区六本木 3-2-12  
日本アイ・ビー・エム株式会社  
法務・知的財産  
知的財産権ライセンス渉外

**以下の保証は、国または地域の法律に沿わない場合は、適用されません。** IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation  
J46A/G4  
555 Bailey Avenue  
San Jose, CA 95141-1003  
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができませんが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

#### 著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。お客様は、IBM のアプリケーション・プログラミング・インターフェースに準拠し

たアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。© Copyright IBM Corp. (enter the year or years). All rights reserved.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

---

## 商標

AIX、DB2、DB2 Universal Database、Distributed Relational Database Architecture、NUMA-Q、OS/2、OS/390、OS/400、IBM Informix®、C-ISAM®、Foundation.2000™、IBM Informix® 4GL、IBM Informix®DataBlade®Module、Client SDK™、Cloudscape™、Cloudsync™、IBM Informix®Connect、IBM Informix®Driver for JDBC、Dynamic Connect™、IBM Informix®Dynamic Scalable Architecture™(DSA)、IBM Informix®Dynamic Server™、IBM Informix®Enterprise Gateway Manager (Enterprise Gateway Manager)、IBM Informix®Extended Parallel Server™、i.Financial Services™、J/Foundation™、MaxConnect™、Object Translator™、Red Brick™、IBM Informix®SE、IBM Informix®SQL、InformiXML™、RedBack®、SystemBuilder™、U2™、UniData®、UniVerse®、wintegrate® は、ed trademarks of International Business Machines Corporation の商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標です。

Windows、Windows NT、および Excel は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。



# 索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

## [ア行]

アプリケーション変換

    コマンド行モード 3-10

    ステップ 3-7, 3-10

    変換ユーティリティ処理 3-10

アプリケーション・レベルの共用ライブラリー

    作成 4-12, 4-13

アプリケーション・レベルの共用ライブラリーの作成 4-12, 4-13

インフォメーション・センター

    概要 4-36

ウィザード、変換ユーティリティ

    概要 3-1

エラー

    変換

        訂正 4-7, 4-10

        変換ログ 4-7, 4-10

エラー・メッセージ

    ファイル変換 4-16

    変換ユーティリティ K-1

演算子

    キーワード・ベースの

        EGL 同等 A-15

    非英字の記号

        EGL 同等 A-16

    EGL 同等 A-15

## [カ行]

書き込み許可

    欠落 4-8

環境変数 viii, A-22

関数

    レポート・ドライバー 4-18

        TERMINATE( ) 4-19

        \_FINISH( ) 4-19

        \_OUTPUT( ) 4-18

        \_OUTPUT(a, b, c ) 4-18

        \_START( ) 4-18

    EGL レポート・ドライバー 4-18

関数、ビルトイン

    EGL 同等 A-13

関数テーブル 4-11

    説明 4-4

    ファイル名 4-4

関数テーブル (続き)

    変更 4-11

    例 4-11

キーワード・ベースの演算子

    EGL 同等 A-15

規則

    書体 viii

共用ライブラリー

    再変換

        オプション 5-1

        概要 5-1

        コマンド行モード 5-2

        再変換する場合 5-1

        再変換の方法 5-1

        次善策 5-3

        失敗の理由 5-3

        プロセス 5-1

        変換ウィザードの再変換 5-2

    変換

        ステップ 3-4

        変換ユーティリティ処理 3-7

共用ライブラリーの再変換

    オプション 5-1

    概要 5-1

    コマンド行モード 5-2

    再変換する場合 5-1

    再変換の方法 5-1

    次善策 5-3

    失敗の理由 5-3

    プロセス 5-1

    変換ウィザードの再変換 5-2

共用ライブラリー変換

    マニフェスト・ファイル

        contents 4-5

グローバル配列変数

    変換 4-21

    変換の制限事項 4-19

構成ファイル

    サンプル 4-5

    手動作成 4-5

    手動で作成 3-11

    説明 4-5

    テンプレート例 D-1

破損

    次善策 5-3

    変換後のロケーション 4-3

    命名規則 4-5

    DTD 4-5

構成ファイルの作成 3-11

コマンド行

    アプリケーション変換 3-10

コマンド行 (続き)  
    共用ライブラリーの再変換 5-2  
コンソール・ユーザー・インターフェース  
    コード例 C-1  
    ステートメント A-9  
コンパイラー・ディレクティブ  
    EGL 同等 A-8

## [サ行]

再変換プロジェクト  
    定義  
        共用ライブラリー変換 3-4  
次善策  
    共用ライブラリーの再変換 5-3  
集約レポート関数  
    変換  
        AVG 4-31  
        COUNT 4-30  
        MAX 4-29  
        MIN 4-30  
        PERCENT 4-31  
        SUM 4-29

書式  
    コード例 C-1  
    EGL 同等 A-9  
    EGL への変換 C-1  
書体の規則 viii  
資料

    インフォメーション・センター  
        概要 4-36  
    変換ユーティリティ・ウィザード x  
    EGL x  
        オンライン・ヘルプ x  
    EGL チュートリアル  
        概要 4-36

新規プロジェクト  
    定義  
        アプリケーション変換 3-7  
        共用ライブラリー変換 3-4  
        データベース・スキーマ抽出 3-2

ストレージ操作ステートメント  
    EGL 同等 A-5

成果物  
    変換  
        デフォルト・ディレクトリー 4-3  
        変換中に生成される 4-3  
製品フィーチャー vii

## [タ行]

タスク、変換前の概要 2-1  
単純なレポート  
    変換 4-23

チュートリアル  
    EGL xi, 4-36  
抽出、データベース・スキーマ  
    ステップ 3-1, 3-3  
    変換ユーティリティ処理 3-3  
データ型  
    EGL 同等 A-1  
データ型定義 3-11  
データベース、アクティブ  
    レポート・ドライバ関数の前提 4-19  
データベース・スキーマ抽出  
    ステップ 3-1, 3-3  
    変換ユーティリティ処理 3-3  
    マニフェスト・ファイル  
        contents 4-5  
定義および宣言ステートメント  
    EGL 同等 A-4  
ディレクトリー  
    デフォルト  
        変換成果物 4-3  
特殊なデータのキャスト  
    EGL 同等 A-3

## [ハ行]

パラメーター変換  
    DEFINE セクション 4-20, 4-21  
非英字ベースの演算子  
    EGL 同等 A-16  
ビジネス・ロジック  
    レポート変換 4-17  
ビルトイン SQL 関数とプロシージャ  
    EGL 同等 A-14  
ビルトイン関数  
    EGL 同等 A-13  
複雑なレポート  
    変換 4-24, 4-26  
太字体 viii  
プレゼンテーション・ロジック  
    レポート変換 4-17  
プログラム・フロー制御ステートメント  
    EGL 同等 A-6  
プロシージャ、外部 SQL  
    EGL 同等 A-14  
プロシージャ、ビルトイン SQL  
    EGL 同等 A-14  
プロジェクト変換  
    エラー・メッセージ・ファイル 4-16  
4GL から EGL へのファイル・マッピング  
    エラー・メッセージ・ファイル 4-6  
    概要 4-6  
    共用ライブラリー・ファイル 4-6  
    書式仕様ファイル 4-6  
    ソース・ファイル 4-6  
    レポート・ファイル 4-6

プロジェクトを開く  
定義  
    アプリケーション変換 3-7  
    共用ライブラリー変換 3-4  
プロパティ・ファイル  
    ファイル  
        プロパティ 4-13  
変換  
    エラー、訂正 4-7, 4-10  
    共用ライブラリー  
        ステップ 3-4  
    コード例 4-20, B-1  
    コード例、レポート B-1  
    順序 3-1  
    成果物 4-3  
        関数テーブルの ロケーション 4-4  
        構成ファイル名 4-3  
        デフォルト・ディレクトリー 4-3  
        変換ログ・ファイル名 4-4  
        マニフェスト・ファイル 4-4  
        EGL ネイティブ・ライブラリーのファイル名 4-4  
        ERR ファイル 4-4, 4-10  
    制限と次善策 2-2  
    変換の順序 3-1  
    変更点  
        概要 4-2  
    application  
        コマンド行モード 3-10  
        ステップ 3-7, 3-10  
変換エラーの訂正 4-7, 4-10  
変換後  
    概要 4-1  
    タスク・リスト 4-1  
変換前  
    概要 2-1  
    タスク・リスト 2-1  
変換プロジェクト  
    失敗の理由 4-8  
        書き込み許可 4-8  
        JDBC ドライバーのエラー 4-8  
    レポート変換 4-17  
    FAILED 4-7, 4-8  
    PASSED 4-7  
変換ユーティリティー  
    ウィザード  
        概要 3-1  
    エラー・メッセージ K-1  
    サポートされるプラットフォーム viii  
    処理  
        アプリケーション変換 3-10  
        共用ライブラリー変換 3-7  
        データベース・スキーマ抽出 3-3  
    資料 ix  
    フィーチャー vii  
    PRINT ステートメントの分析 4-26

変換ユーティリティー・ウィザード  
    資料 x  
変換ログ 3-11  
    エラーの訂正 4-7, 4-10  
    使用法 4-7  
    説明 4-4  
    ファイル 分類  
        ERROR 4-8  
        FIXME 4-8  
        PASSED 4-8  
        TODO 4-9  
    変換後のロケーション 4-4  
    例 G-1  
    contents 4-9  
    JDBC ドライバーのエラー 4-8, 4-10  
    location 4-7

## [マ行]

マニフェスト・ファイル  
    共用ライブラリー変換  
        contents 4-5  
    説明 4-4  
    データベース・スキーマ抽出  
        contents 4-5  
    破損  
        次善策 5-3  
    ファイル名 4-4  
    例 E-1

## [ヤ行]

ユーザー、タイプ vii

## [ラ行]

リンク、C ライブラリーの EGL への 4-10, 4-13  
レポート  
    グローバル配列変数の 制限事項 4-19  
    変換  
        レポート演算子 4-28  
        DEFINE セクション 4-20, 4-21  
        DEFINE セクションの パラメーター変換 4-20, 4-21  
        DEFINE セクションのローカル変数の変換 4-21  
        FORMAT セクション 4-23, 4-24, 4-26, 4-28, 4-29  
        I4GL 集約 レポート関数 4-29, 4-32  
        I4GL レポート・セクション 4-19  
        LINENO 4-25  
        ORDER BY セクション 4-22  
        ORDER EXTERNAL BY 4-23  
        OUTPUT セクション 4-21, 4-22  
        SKIP TO TOP OF PAGE 4-25  
    EGL 関数呼び出し 4-17  
    EGL 処理 4-19  
    EGL への変換 4-17

レポート (続き)

例 B-1

I4GL コードへの変更 4-17

JasperReports への変換 4-20

レポート演算子

変換

CLIPPED 4-28

FILE 4-28

LINENO 4-28

PAGENO 4-28

SPACE 4-28

SPACES 4-28

USING 4-28

WORDWRAP 4-28

レポート実行ステートメント

EGL 同等 A-12

レポート変換

サブレポート 4-17

ビジネス・ロジック変換 4-17

ファイル名変換

サブレポート 4-17

ビジネス・ロジック 4-17

プレゼンテーション・ロジック 4-17

複数のループ構成体 4-17

プレゼンテーション・ロジック変換 4-17

EGL レポート・ドライバ関数 4-17

レポート・ドライバ・ステートメント

EGL 同等 A-12

レポート・ハンドラー・メソッド

afterGroup 4-25

beforeDetailEval 4-25

beforeGroupOf 4-25

firstPageHeader 4-25

getPrintFlag 4-25

getPrintString 4-25

init 4-25

onLastRow 4-25

pageHeader 4-25

pageTrailer 4-25

連絡先情報 xi

ローカル変数の変換

DEFINE セクション 4-21

ログ、変換

参照: 変換ログ

## [数字]

4GL

演算子

キーワード・ベースの A-15

非英字の記号 A-16

EGL 同等 A-15

外部 SQL 関数とプロシージャ

EGL 同等 A-14

環境変数

EGL 同等プロパティ A-22

4GL (続き)

環境変数 (続き)

JDBC 同等プロパティ A-22

関数、外部 SQL

EGL 同等 A-14

関数、ビルトイン

EGL 同等 A-13

関数、ビルトイン SQL

EGL 同等 A-14

コンパイラ・ディレクティブ

EGL 同等 A-8

書式

EGL 同等 A-9

ストレージ操作ステートメント

EGL 同等 A-5

データ型

EGL 同等 A-1

定義および宣言ステートメント

EGL 同等 A-4

特殊なデータのキャスト

EGL 同等 A-3

ビルトイン SQL 関数とプロシージャ

EGL 同等 A-14

ビルトイン関数

EGL 同等 A-13

プログラム・フロー制御ステートメント

EGL 同等 A-6

プロシージャ、外部 SQL

EGL 同等 A-14

プロシージャ、ビルトイン SQL

EGL 同等 A-14

レポート実行ステートメント

EGL 同等 A-12

レポート・ドライバ・ステートメント

EGL 同等 A-12

SQL オプティカル・サブシステム・ステートメント

EGL 同等 A-21

SQL カーソル操作ステートメント

EGL 同等 A-17

SQL クライアント/サーバー接続ステートメント

EGL 同等 A-21

SQL 照会最適化ステートメント

EGL 同等 A-19

SQL ストアド・プロシージャ・ステートメント

EGL 同等 A-20

SQL データ操作ステートメント

EGL 同等 A-19

SQL データ定義ステートメント

EGL 同等 A-18

SQL データ保全性ステートメント

EGL 同等 A-20

SQL データ・アクセス・ステートメント

EGL 同等 A-20

SQL 動的管理ステートメント

EGL 同等 A-19

4GL と同等 4-6

4GL ファイル拡張子  
EGL ヘマップ 4-6  
4GL 変換の順序 3-1

## A

After Group Of  
変換 4-24  
afterGroup  
レポート・ハンドラー・メソッド 4-25  
AVG 集約レポート関数  
変換 4-31

## B

Before Group Of  
変換 4-24  
beforeDetailEval  
レポート・ハンドラー・メソッド 4-25  
beforeGroupOf  
レポート・ハンドラー・メソッド 4-25  
BOTTOM MARGIN  
変換 4-21

## C

C 関数  
関数呼び出し 順序 4-12  
C ライブラリー  
EGL ヘ接続  
関数テーブル 4-11  
関数テーブル例 4-11  
EGL ネイティブ・ライブラリー 4-11  
EGL へのリンク 4-10, 4-13  
CLIPPED  
変換 4-28  
conversionconfig.dtd 4-5  
conversionssample.xml 4-5  
COUNT 集約レポート関数  
変換 4-30

## D

DEFINE セクション  
パラメーター変換 4-20, 4-21  
変換 4-20, 4-21  
ローカル変数の変換 4-21  
DTD 3-11  
例 F-1

## E

EGL  
資料 x

EGL (続き)  
オンライン・ヘルプ x  
EGL リファレンス・ガイド x  
チュートリアル xi  
プリミティブ型 A-1  
EGL リファレンス・ガイド x  
Web サイト x  
EGL チュートリアル  
概要 4-36  
EGL 同等  
特殊なデータのキャスト A-3  
ビルトイン SQL 関数 A-14  
4GL SQL オプティカル・サブシステム・ステートメント  
A-21  
4GL SQL クライアント/サーバー接続ステートメント  
A-21  
4GL SQL 照会最適化ステートメント A-19  
4GL SQL ストアード・プロシージャ・ステートメント  
A-20  
4GL SQL データ操作ステートメント A-19  
4GL SQL データ定義ステートメント A-18  
4GL SQL データ保全性ステートメント A-20  
4GL SQL データ・アクセス・ステートメント A-20  
4GL SQL 動的管理ステートメント A-19  
4GL 演算子 A-15  
4GL 演算子、キーワード・ベースの A-15  
4GL 演算子、非英字ベースの A-16  
4GL カーソル操作ステートメント A-17  
4GL 外部 SQL 関数 A-14  
4GL 環境変数 A-22  
4GL キーワード・ベースの演算子 A-15  
4GL コンパイラー・ディレクティブ A-8  
4GL 書式 A-9  
4GL ストレージ操作ステートメント A-5  
4GL データ型 A-1  
4GL 定義および宣言ステートメント A-4  
4GL 非英字ベースの演算子 A-16  
4GL ビルトイン関数 A-13  
4GL プログラム・フロー制御ステートメント A-6  
4GL レポート実行ステートメント A-12  
4GL レポート・ドライバー・ステートメント A-12  
EGL ネイティブ・ライブラリー 4-11  
説明 4-4  
変換後のロケーション 4-4  
EGL パッケージ  
概要 4-33  
EGL ビルド記述子  
ファイル 4-6  
例 H-1  
EGL ファイル  
概要 4-34  
ソース・ファイル 4-34  
ビルド・ファイル 4-35  
EGL ファイル拡張子  
4GL ヘマップ 4-6

EGL プロジェクト  
    概要 4-32  
    推奨事項 4-35, 4-36  
EGL プロパティ 4-32  
EGL メッセージ・ファイル・フォーマット 4-16  
EGL 予約語 1-1  
EGL リファレンス・ガイド x  
EGL レポートの関数呼び出し 4-17  
EGL レポート・ドライバ関数 4-17, 4-18  
    TERMINATE( ) 4-19  
    \_FINISH( ) 4-19  
    \_OUTPUT( ) 4-18  
    \_OUTPUT(a, b, c ) 4-18  
    \_START( ) 4-18  
ERR ファイル  
    説明 4-4, 4-10  
    ファイル名 4-4, 4-10  
ERROR  
    変換済みファイルの分類 4-8

## F

FAILED  
    変換プロジェクト 4-7, 4-8  
FILE  
    変換 4-28  
First Page Header  
    変換 4-24  
firstPageHeader  
    レポート・ハンドラー・メソッド 4-25  
FIXME  
    変換済みファイルの分類 4-8  
FORMAT サブセクション  
    After Group Of 4-24  
    Before Group Of 4-24  
    First Page Header 4-24  
    JasperReports バンドへの変換 4-24  
    On Every Row 4-24  
    On Last Row 4-24  
    Page Header 4-24  
    Page Trailer 4-24  
FORMAT セクション  
    単純なレポート  
        変換 4-23  
    複雑なレポート  
        変換 4-24, 4-26  
        変換 4-23, 4-29  
    PRINT ステートメント  
        変換 4-26, 4-28

## G

getPrintFlag  
    レポート・ハンドラー・メソッド 4-25

getPrintString  
    レポート・ハンドラー・メソッド 4-25

## I

Informix データベース  
    スキーマ抽出  
        ステップ 3-1, 3-3  
        変換ユーティリティ処理 3-3  
init  
    レポート・ハンドラー・メソッド 4-25

## J

JasperReports  
    バンド 4-24  
    変換コード 例 4-20  
Java プロパティ・ファイル  
    エラー・メッセージ変換 4-16  
JDBC ドライバのエラー 4-8, 4-10

## L

LEFT MARGIN  
    変換 4-22  
LINENO  
    変換 4-25, 4-28

## M

MAX 集約レポート関数  
    変換 4-29  
MIN 集約レポート関数  
    変換 4-30

## O

On Every Row  
    変換 4-24  
On Last Row  
    変換 4-24  
onLastRow  
    レポート・ハンドラー・メソッド 4-25  
ORDER BY セクション  
    変換 4-22  
ORDER EXTERNAL BY  
    変換 4-23  
OUTPUT セクション  
    変換 4-21, 4-22  
    BOTTOM MARGIN 4-21  
    LEFT MARGIN 4-22  
    PAGE LENGTH 4-21  
    REPORT TO 4-22  
    RIGHT MARGIN 4-22

OUTPUT セクション (続き)  
変換 (続き)  
TOP MARGIN 4-22  
TOP OF PAGE 4-22

## P

Page Header  
変換 4-24  
PAGE LENGTH  
変換 4-21  
Page Trailer  
変換 4-24  
pageHeader  
レポート・ハンドラー・メソッド 4-25  
PAGENO  
変換 4-28  
pageTrailer  
レポート・ハンドラー・メソッド 4-25  
PASSED  
変換済みファイルの分類 4-8  
変換プロジェクト 4-7  
PERCENT 集約レポート関数  
変換 4-31  
PRINT ステートメント  
式  
変換 4-27  
セミコロンで終了する  
変換 4-28  
変換 4-26, 4-28  
ループ構成体の 4-27

## R

REPORT TO  
変換 4-22  
RIGHT MARGIN  
変換 4-22

## S

SKIP TO TOP OF PAGE  
変換 4-25  
SPACE  
変換 4-28  
SPACES  
変換 4-28  
SUM 集約レポート関数  
変換 4-29

## T

TERMINATE( )  
EGL の振る舞い 4-19

TODO  
変換済みファイルの分類 4-9  
TOP MARGIN  
変換 4-22  
TOP OF PAGE  
変換 4-22

## U

USING  
変換 4-28

## W

WORDWRAP  
変換 4-28

## [特殊文字]

.eglbld 4-6  
.properties 4-13  
\_FINISH( )  
EGL の振る舞い 4-19  
\_OUTPUT( )  
EGL の振る舞い 4-18  
\_OUTPUT(a, b, c )  
EGL の振る舞い 4-18  
\_START( )  
EGL の振る舞い 4-18







Printed in Japan

GB88-4048-01



日本アイ・ビー・エム株式会社  
〒106-8711 東京都港区六本木3-2-12