DB2 Content Manager OnDemand

IBM

Windows クライアント・カスタマイズ・ガイド

SC88-8840-04
(英文原典：SC27-0837-04)

DB2 Content Manager OnDemand

Windows クライアント・カスタマイズ・ガイド

SC88-8840-04
(英文原典：SC27-0837-04)

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、 295 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。

本書は、SC88-8840-03 の改訂版です。

本書は IBM DB2 Content Manager OnDemand for z/OS and OS/390 バージョン 8 リリース 4 (プロダクト番号 5697-N93)、IBM DB2 Content Manager OnDemand for Multiplatforms バージョン 8 リリース 4 (プロダクト番号 5724-J33)、i5/OS 共通サーバー用 IBM DB2 Content Manager OnDemand バージョン 5 (プロダクト番号 5722-RD1) および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

この製品には、Apache Software Foundation (http://www.apache.org/) により開発されたソフトウェアが含まれています。

IBM 発行のマニュアルに関する情報のページ

http://www.ibm.com/jp/manuals/

こちらから、日本語版および英語版のオンライン・ライブラリーをご利用いただけます。また、マニュアルに関するご意見やご感想を、上記ページよりお送りください。今後の参考にさせていただきます。

(URL は、変更になる場合があります)

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： SC27–0837–04
DB2® Content Manager OnDemand
Windows Client Customization Guide

発　行： 日本アイ・ビー・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 2007.5

目次

本書について

本書には、IBM® DB2® Content Manager OnDemand (OnDemand) Object Linking and Embedding (OLE) Control に関する情報と、OnDemand Windows クライアントをカスタマイズする方法が記載されています。カスタマイズは、コマンド行パラメーターを指定するか、動的データ交換 (DDE) インターフェースで別の Windows® アプリケーションから OnDemand Windows クライアントを呼び出して操作するか、Program Information File (PIF) を作成して、行います。 本書には、ネットワークを介して複数のユーザーに OnDemand Windows クライアント・ソフトウェアおよびファイルを配布するときに管理者が利用できる情報も記載されています。

注: *OnDemand Windows クライアント* (または単にクライアント) という用語は、Windows 2000、Windows XP、および Windows 2003 の下で実行される OnDemand エンド・ユーザー・クライアント・ソフトウェアを指します。 サーバー という用語は、OnDemand for i5/OS® バージョン 5、OnDemand for Multiplatforms バージョン 8.3、および OnDemand for z/OS® および OS/390® バージョン 7.1 ソフトウェアを実行しているシステムを指します。

対象読者

本書は、OnDemand と他の Windows アプリケーションとの統合を図るプログラマー、およびソフトウェア製品のインストールと配布を担当する管理者を主な対象読者としています。

本書の構成

本書には、次の情報があります。

- 1 ページの『第 1 部 OnDemand OLE Control』には、次のセクションがあります。
  - 3 ページの『第 1 章 OnDemand OLE Control の概要』では、OnDemand OLE Control の概要を説明します。
  - 5 ページの『第 2 章 メソッド』では、OnDemand OLE Control で使用できるメソッドについて説明します。
  - 121 ページの『第 3 章 OLE イベント』では、OnDemand OLE Control によって引き起こされるイベントを説明します。
- 123 ページの『第 2 部 Windows 32 ビット GUI カスタマイズ・ガイド』には、次のセクションがあります。
  - 125 ページの『第 4 章 OnDemand のカスタマイズの概要』では、クライアントのカスタマイズ方法について概説します。
  - 127 ページの『第 5 章 コマンド行』では、クライアントの開始方法、コマンド行パラメーターに使用されるパラメーター構文規則、およびクライアントで認識されるパラメーターについて説明します。

- 133 ページの『第 6 章 動的データ交換 (DDE) および DDE Management Library』 では、OnDemand で動的データ交換 (DDE、Dynamic Data Exchange) を使用する方法を説明し、OnDemand DDE コマンドをリストします。
- 139 ページの『第 7 章 OnDemand DDE コマンド』では、OnDemand DDE コマンドについて説明します。
- 177 ページの『第 8 章 戻りコード』では、OnDemand 戻りコードについて説明します。
- 179 ページの『第 9 章 DDEML Advise Loop』では、特定のイベントが発生した時に通知を行うために、クライアント・アプリケーションが作成する可能性のある、DDEML アドバイス・ループについて説明します。
- 181 ページの『第 10 章 外部アプリケーションとダイナミック・リンク・ライブラリー』では、エンド・ユーザーが別の Windows アプリケーションを呼び出したり、ダイナミック・リンク・ライブラリー (DLL) の関数を実行することができるメニューおよびツールバー拡張機能について説明します。
- 189 ページの『第 11 章 関連文書』では、エンド・ユーザーが現在表示中の文書に関連した文書の取り出しおよび表示を行うことができるメニューおよびツールバー拡張機能について説明します。
- 195 ページの『第 12 章 Program Information File』では、Product Information File (PIF) を使用して「製品情報」ダイアログ・ボックスの OnDemand アプリケーションの表題と外観をカスタマイズする方法について説明します。
- 197 ページの『第 13 章 Document Audit Facility』では、Document Audit Facility (DAF) について説明します。DAF を使用して、クライアントで文書を監査することができます。
- 203 ページの『第 14 章 レジストリー (Registry) によるクライアントの動作の変更』では、レジストリーを使ってクライアントの振る舞いを変更する方法を説明します。
- 205 ページの『第 15 章 Monarch バージョン 5 との統合』では、Monarch データ・マイニング・ソフトウェアを扱うためのクライアントの構成方法について説明します。
- 217 ページの『第 16 章 ネットワーク上のクライアント・ソフトウェアのインストール』では、複数のユーザーがネットワークを介して共用するクライアント・ソフトウェアのインストールについて説明します。
- 223 ページの『第 17 章 ユーザー定義ファイルの配布』では、ユーザー定義ファイルを配布するためのクライアント・インストール・プログラムの構成方法について説明します。
- 227 ページの『第 18 章 応答ファイルの使用』では、クライアント・インストール・プロセスを自動化するときに役立つ情報を提供します。
- 231 ページの『第 19 章 AFP フォントのマッピング』では、文書の作成時に使用した 拡張機能表示™ (AFP™) フォントを、クライアントが表示できるフォントにマップするときに役立つ情報を提供します。
- 245 ページの『第 20 章 トラブルシューティング』には、トラブルシューティング・シナリオ、および『ヒント』セクションが含まれています。

• 『付録』には、次のサンプル・プログラムが含まれています。
  247 ページの『付録 A. Microsoft Visual Basic 5.0 DDE サンプル・プログラム』

本書の追加情報の入手先

製品パッケージには、情報が完全なセットで含まれており、システムの計画、インストール、管理、および使用の際に役立ちます。すべての製品資料は、PDF 形式で提供されています。オペレーティング・システムに合った Adobe® Acrobat® Reader を使用して、PDF ファイルをオンラインで表示することができます。 Acrobat Reader をまだインストールしていない場合は、Adobe Web サイト www.adobe.com からダウンロードできます。

製品資料は、OnDemand Web サイト (www.ibm.com/software/data/ondemand/) および IBM 資料オーダー・システム (www.ibm.com/shop/publications/order) から入手することもできます。

次の表に、プラットフォームごとの OnDemand 資料をリストします。

表 1. IBM DB2 Content Manager OnDemand for z/OS and OS/390 バージョン 8 資料

資料名	資料番号
Administration Guide	SC19-1213
Configuration Guide	SC19-1212
Indexing Reference	SC19-1214
Introduction and Planning Guide	SC19-1217
メッセージとコード	SC88-9781
Migration Guide	SC19-1216
OnDemand Distribution Facility Installation and Reference Guide	GC19-1218
ユーザーズ・ガイド	SC88-8839
Web Enablement Kit Implementation Guide	SC19-1215
Windows クライアント・カスタマイズ・ガイド	SC88-8840

「*IBM DB2 Content Manager OnDemand for z/OS: Introduction and Planning Guide*」には、OnDemand ライブラリーの用語集が含まれています。「*IBM DB2 Content Manager OnDemand ユーザーズ・ガイド*」に掲載されているのは、OnDemand 管理者ではなく OnDemand ユーザーを対象とした、比較的小さな用語集です。「*IBM DB2 Content Manager OnDemand for z/OS: OnDemand Distribution Facility Installation and Reference Guide*」には、OnDemand Distribution Facility に特化した用語集が含まれています。

表 2. IBM DB2 Content Manager OnDemand for Multiplatforms バージョン 8 資料

資料名	資料番号
管理ガイド	SD88-6411
索引付けリファレンス	SD88-6409
インストールと構成ガイド	SD88-6407
紹介および計画ガイド	GD88-6410
メッセージとコード	SC88-9781
レポート配布: インストール、使用およびリファレンス	SD88-6408
ユーザーズ・ガイド	SC88-8839
Web イネーブルメント・キット インプリメンテーション・ガイド	SD88-6406
Windows クライアント・カスタマイズ・ガイド	SC88-8840

「*IBM DB2 Content Manager OnDemand for Multiplatforms 紹介および計画のガイド*」には、OnDemand ライブラリーの用語集が含まれています。「*IBM DB2 Content Manager OnDemand ユーザーズ・ガイド*」に掲載されているのは、OnDemand 管理者ではなく OnDemand ユーザーを対象とした、比較的小さな用語集です。「*IBM DB2 Content Manager OnDemand for Multiplatforms レポート配布: インストール、使用およびリファレンス*」には、特に OnDemand レポート配布機能に関する用語集が含まれています。

表 3. IBM DB2 Content Manager OnDemand for i5/OS Common Server バージョン 5 資料

資料名	資料番号
管理ガイド	SD88-5029
Common Server 管理ガイド	SC88-4011
Common Server 索引付けリファレンス	SC88-4010
Common Server 計画とインストール	SC88-4008
Common Server Web Enablement Kit インストールおよび構成ガイド	SC88-4013
導入の手引き	SD88-5070
メッセージとコード	SC88-9781
ユーザーズ・ガイド	SC88-8839
Windows クライアント・カスタマイズ・ガイド	SC88-8840

OnDemand のアクセシビリティー情報

本製品でサポートされるアクセシビリティー機能の詳細情報は、「OnDemand *管理ガイド*」を参照してください。

Web から入手可能なサポート

IBM では、最新の製品情報をオンラインで提供しています。よくある質問、ヒント、および技術情報については、以下のプラットフォーム固有 Web サイトの 1 つからサポート・リンクをたどってください。

- Multiplatforms: http://www.ibm.com/software/data/ondemand/mp/
- i5/OS: http://www.ibm.com/software/data/ondemand/400/
- z/OS と OS/390: http://www-306.ibm.com/software/data/ondemand/390/

教育研修

IBM では、OnDemand 管理者を対象としたいくつかの研修を開催しています。コースの説明および価格については、以下のプラットフォーム固有 Web サイトの 1 つからトレーニングおよび認証リンクをたどってください。

- Multiplatforms: http://www.ibm.com/software/data/ondemand/mp/
- i5/OS: http://www.ibm.com/software/data/ondemand/400/
- z/OS と OS/390: http://www-306.ibm.com/software/data/ondemand/390/

マニュアル・出版物に関するご意見・ご質問

マニュアル・出版物に関するご意見・ご質問は、次の URL からお送りください。
http://www.ibm.com/jp/manuals/main/mail.html

変更の要約

本書には、IBM DB2 Content Manager OnDemand *Windows クライアント・カスタマイズ・ガイド* (SC88-8840-03) に記載されていた情報への追加、変更が含まれています。技術的な追加および変更のあった個所には、左マージンにリビジョン・バー (|) が示されています。

- 8 枚の画面取りを更新しました。

第 1 部 OnDemand OLE Control

第 1 章 OnDemand OLE Control の概要

注: 『OLE Control の概要』を利用するには、OLE Control をアプリケーションに組み込む方法を知っておく必要があります。

OnDemand によって OLE (Object Linking and Embedding) Control が使用可能になり、OnDemand データベースの文書が表示できるようになります。OLE Control は、ARSOLE.OCX 内でインプリメントされます。OnDemand のインストール時に、このファイルは、他の OnDemand 実行可能ファイルと同じディレクトリーに入れられ、OLE Control は Windows システムに登録されます。 OnDemand がインストールされたディレクトリー以外のディレクトリーからコンテナー・アプリケーションを実行するには、OnDemand ディレクトリーをパスに追加する必要があります。

OLE Control を使用するときは、以下の規則が適用されます。

- 各コントロールが一度に表示できる文書は 1 つだけです。表示されている文書をクローズしてからでないと、別の文書を表示できません。
- 文書データのスクロールを制御するスクロール・バーは、コンテナー・アプリケーションによって制御される対象です。スクロール・バーは、必ず OLE Control ウィンドウの外側に表示されていなければなりません。OLE Control には、文書データのスクロールを指示するためのメソッドがあります。スクロール・バーの範囲を ARS_OLE_SCROLL_RANGE に設定すると、そのメソッドが比較的使用しやすくなります。
- 複数のフォルダーが同時にオープン状態になることはあっても、アクティブ・フォルダーになるのは、それらのフォルダーのうちの 1 つだけです。OLE Control には、フォルダーのオープン、クローズ、およびアクティブ化を行うためのメソッドがあります。
- コンテナー・アプリケーションは、ログオン、フォルダー・オープン、フォルダー検索、フォルダー・クローズ、および文書オープンの各操作の制御をすべて実行できますが、これらの操作を実行するために、通常の OnDemand ダイアログ・ボックスを使用できるようにする場合もあります。

単一フォルダーの複数文書の表示

それぞれの OnDemand OLE Control に固有の実行時コントロール ID があります。このコントロール ID は、GetControlId メソッドで取り出すことができます。

コントロール ID によって、複数の OnDemand OLE Control が単一フォルダーの文書リストの文書を同時に表示できるようになります。このため、ログオン、フォルダー・オープン、フォルダー検索の操作を複数回行うことによるオーバーヘッドがなくなります。

1 つのアプリケーションに複数の OnDemand OLE Control を組み込むことができます。そのアプリケーションは、複数のコントロールの 1 つを使用してログオン、フォルダーのオープン、およびフォルダーの検索を行って、文書リストを作成できます。そのコントロール ID が使用可能になると、それ以外のコントロールは、

OpenDoc メソッドを使用するときに、そのコントロール ID を参照して単一の文書リストの文書を表示することができます。

ヘッダー・ファイル

ARSOLEEX.H ヘッダー・ファイルには、このあと説明する OLE コントロール・メソッドで使用される記号値の定義が含まれています。このファイルは、C/C++ インプリメンテーションに組み込むことができ、その他の言語では参照ファイルとして使用することができます。

このヘッダー・ファイルは、OnDemand インストール・ディレクトリーの INC サブディレクトリーにインストールされます。そのディレクトリーを組み込みファイル・パスに追加することも、このヘッダー・ファイルを別のディレクトリーにコピーすることもできます。

戻りコード

ほとんどの OnDemand OLE Control メソッドでは、short 値が戻ります。ARS_OLE_RC_SUCCESS などの戻りコード値のリストは、ARSOLEEX.H の中にあります。

第 2 章 メソッド

これ以降の項では、OnDemand OLE Control で使用できるメソッドについて説明します。

AboutBox

メソッド:

void **AboutBox( )**

説明: 「OnDemand About Box」を表示します。

リターン値:

なし

例: 以下の例では、「OnDemand About Box」を表示します。

C/C + + の例

```
CArsOle * pArsCtrl;

  .
  .

pArsCtrl->AboutBox( );

  .
  .
```

Visual Basic の例

```
 .
 .
 .

ArsOle.AboutBox

 .
 .
 .
```

ActivateFolder

メソッド:

short ActivateFolder(

char * **pFolderName**)

パラメーター:

pFolderName

アクティブにするフォルダーの名前を含むヌル終了文字ストリングを指します。

説明: 指定したフォルダーがアクティブ・フォルダーになります。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: OpenFolder, CloseFolder, CloseAllFolders

例 以下の例では、フォルダーをアクティブにします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

 rc = pArsCtrl->ActivateFolder( "Henry's Folder" );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .
 .

rc = ArsOle.ActivateFolder("Henry's Folder")
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
 .
```

AnnotateDoc

メソッド:

short AnnotateDoc(

long **Index**,

char * **pText**,

long **page**,

boolean **Public**,

boolean **CanBeCopied**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。この値がゼロ未満であれば、開いている文書に注釈が関連付けられます。

pText

注釈のテキストを含むヌル終了文字ストリングを指します。テキストの文字数が 32,700 を超える場合は切り捨てられます。

page

注釈に関連付けられるページ番号を指定します。

Public

注釈が共通かどうかを指示します。

CanBeCopied

注釈を他のサーバーにコピーできるかどうかを指示します。

説明: 注釈がデータベース内に作成され、指定した文書に関連付けられます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenDoc

例 以下の例では、文書の注釈を作成します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

 rc = pArsCtrl->AnnotateDoc( 3, "This is the text.", 5, TRUE, FALSE );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .
 .

rc = ArsOle.AnnotateDoc(3, "This is the text.", 5, True, False)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
 .
```

CancelOperation

メソッド:

short CancelOperation( )

説明: SearchFolder、OpenDoc または RetrieveDoc メソッドで開始した操作を取り消します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: SearchFolder, OpenDoc, WasOperationCancelled, ShowWaitCursorDuringCancelableOperation

例: 以下の例では、操作を取り消します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
 rc = pArsCtrl->CancelOperation( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.CancelOperation ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

ChangePassword

メソッド:

short **ChangePassword**(

char * **pCurrentPassword**,

char * **pNewPassword1**,

char * **pNewPassword2**)

パラメーター:

pCurrentPassword

ユーザーの現行パスワードを指定します。

pNewPassword1

ユーザーの新規パスワードを指定します。

pNewPassword2

ユーザーの新規パスワードをもう一度指定します。これは、確認を行うためです。

説明: OnDemand は、現行ユーザーのログオン・パスワードを変更します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: Logon

例:

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

rc = pArsCtrl->ChangePassword( "tt1sd",
                               "sfd45r",
                               "sfd45r" );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

  .
  .

rc = ArsOle.ChangePassword ( "tt1sd",   _
                             "sfd45r",  _
                             "sfd45r" )
if rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

  .
  .
```

ClearFolderSearchFields

メソッド:

short ClearFolderSearchFields( )

説明: アクティブ・フォルダーの検索フィールドをクリアします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, SearchFolder

例: 以下の例では、アクティブ・フォルダーの検索フィールドをクリアします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .
```

```
rc = pArsCtrl->ClearFolderSearchFields( );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

 .
 .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .
 .

rc = ArsOle.ClearFolderSearchFields()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
 .
```

CloseAllFolders

メソッド:

short CloseAllFolders( )

説明: 開いているフォルダーすべてをクローズします。この結果、フォルダーに関連したオープン中の文書もすべてクローズします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, CloseFolder

例: 以下の例では、すべてのフォルダーをクローズします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

 rc = pArsCtrl->CloseAllFolders( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .
 .
```

```
rc = ArsOle.CloseAllFolders()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
 .
```

CloseDoc

メソッド:

short CloseDoc( )

説明: 開いている文書をクローズし、コントロール・ウィンドウを白の背景で再描画します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: OpenDoc

例: 以下の例では、文書をクローズします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

   .
   .

  rc = pArsCtrl->CloseDoc( );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

   .
   .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .
 .

rc = ArsOle.CloseDoc()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
 .
```

CloseFolder

メソッド:

short CloseFolder( )

説明: アクティブ・フォルダーをクローズします。この結果、そのフォルダーに関連したオープン中の文書もすべてクローズします。ほかのフォルダーが開いている場合、そのうちの 1 つがアクティブ・フォルダーになります。ほかのフォルダーが複数開いている場合は、コンテナー・アプリケーションは、ActivateFolder メソッドを呼び出してアクティブになるフォルダーを指定する必要があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, CloseAllFolders

例: 以下の例では、アクティブ・フォルダーをクローズします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

 rc = pArsCtrl->CloseFolder( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.CloseFolder()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

CopyBitmap

メソッド:

short CopyBitmap( )

説明: 文書中の選択した領域をクリップボードにビットマップ・フォーマットでコピーします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CopyText, SetSelectionMode

例: 以下の例では、文書中の選択した領域をクリップボードにビットマップ・フォーマットでコピーします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
   .
   .
 rc = pArsCtrl->CopyBitmap( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
   .
   .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.CopyBitmap ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

CopyDocPagesToFile

メソッド:

short CopyDocPagesToFile(

char * **pPath**,

long **page**,

boolean **AsIs**)

パラメーター:

page

コピーするページ番号を指定します。このパラメーターがゼロ以下のときには、文書全体がコピーされます。

pPath

データのコピー先となるファイルの完全修飾名を含むヌル終了文字ストリングを指します。 そのファイルがすでに存在している場合、データは、そのファイルに付加されます。

AsIs (AFP および行データ専用)

値が非ゼロであれば、データを「現状のまま」コピーするよう指示します。ゼロであれば、データを ASCII に変換するよう指示します。

説明: 開いている文書の 1 ページまたは複数のページのデータを、指定のファイルにコピーします。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: SearchFolder, GetNumDocsInList

例: 以下の例では、開いている文書の 5 ページ目を ASCII フォーマットでファイルにコピーします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

rc = pArsCtrl->CopyDocPagesToFile( "C:¥¥FILES¥¥MYDATA.FIL", 5, FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.CopyDocPagesToFile("C:¥FILES¥MYDATA.FIL", 5, False)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

CopyText

メソッド:

short CopyText( )

説明: 文書中の選択した領域をクリップボードにテキスト・フォーマットでコピーします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CopyBitmap, SetSelectionMode

例: 以下の例では、文書中の選択した領域をクリップボードにテキスト・フォーマットでコピーします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
 rc = pArsCtrl->CopyText( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.CopyText ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

DeleteDoc

メソッド:

short **DeleteDoc**(

long **DocIndex**)

パラメーター:

DocIndex

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。

説明: OnDemand は、指定した文書をデータベースから削除します。文書番号が変更されていることがあるので、直前の GetDocDisplayValues メソッドからの情報がすでに無効になっている場合があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocsInList

例: 以下の例では、アクティブ・フォルダーの文書リスト内の最初の文書を削除します。

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

rc = pArsCtrl->DeleteDoc( 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

  .
  .

rc = ArsOle.DeleteDoc (0)
If rc <> ARS_OLE_RC_SUCCESS Then
```

```
    MsgBox "ERROR"
    End
  End If

  .
  .
```

FindStringInDoc

メソッド:

short FindStringInDoc(

char * **pString**,

long **page**,

short **Type**,

boolean **CaseSensitive**,

VARIANT * **pFound**,

VARIANT * **pHorzPosition**,

VARIANT * **pVertPosition**)

パラメーター:

pString

検索するテキストを含むヌル終了文字ストリングを指します。

page

検索を開始するページを指定します。Type で ARS_OLE_FIND_PREV または ARS_OLE_FIND_NEXT を指定する場合、そのページは現在検索されたものが強調表示されているページと同じでなければなりません。

Type

検索操作のタイプを指定します。このタイプは、ARSOLEEX.H の中にある以下のタイプ値の 1 つでなければなりません。

```
ARS_OLE_FIND_FIRST
ARS_OLE_FIND_PREV
ARS_OLE_FIND_NEXT
```

CaseSensitive

値が非ゼロであれば、大/小文字を区別して検索するよう指示します。ゼロであれば、大/小文字を無視するよう指示します。

pFound

見つかった、または見つからなかったという指示を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pHorzPosition

新規水平スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pVertPosition

新規垂直スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定したページで始まっているテキスト・ストリングを検索します。検索が成功した場合は、pFound が指す変数は非ゼロに設定されます。失敗した場合は、ゼロに設定されます。検索が成功した場合、ストリングが見つかったページが現行ページになり、そのストリングは強調表示され、スクロールされて表示されます。また、指定した変数に新規スクロール位置が戻されます。そのスクロール位置は、スクロール範囲が ARS_OLE_SCROLL_RANGE に設定されていることを前提としています。

常に、文書の末尾から先頭または先頭から末尾へと「折り返して」検索が行われます。直前を検索または次を検索の場合、失敗することは決してありません。文書内にストリングが 1 回だけ出現している場合、直前を検索と次を検索の両方で同じストリングが検索されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenDoc, UndoFind

例: 以下の例では、検索を実行します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScrollBar, * pVertScrollBar;
VARIANT found, horz_position, vert_position;
char * pString;
short rc;
  .
  .
  .

 rc = pArsCtrl->FindStringInDoc( pString,
                                 1,
                                 ARS_OLE_FIND_FIRST,
                                 FALSE,
                                 &found,
                                 &horz_position,
                                 &vert_position );

 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

 if ( found.iVal )
 {
   pHorzScrollBar->SetScrollPos( (int)horz_position.iVal );
   pVertScrollBar->SetScrollPos( (int)vert_position.iVal );

     .
     .
 }
 else
 {
     .
     .
 }

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim found, horz_pos, vert_pos As Variant
Dim Temp As String
 .
 >
 .
rc = ArsOle.FindStringInDoc( Temp,
                             1,
                             ARS_OLE_FIND_FIRST,
                             False,
                             found,
                             horz_pos,
                             vert_pos )
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If found <> 0 Then
   hScrollBar.Value = horz_pos
   vScrollBar.Value = vert_pos
End If

 .
 .
```

GetAnnotationForDoc

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetAnnotationForDoc(

short **Index**,

BSTR * **pText**,

BSTR * **pUserId**,

BSTR * **pDateTime**,

VARIANT * **pPage**,

VARIANT * **pPublic**,

VARIANT * **pCanBeCopied**)

パラメーター:

Index

戻される注釈の 0 を基準とした索引を指定します。0 以上かつ GetNumDocAnnotations によって戻される値未満の数値を指定する必要があります。

pText

注釈のテキストを受け取る BSTR を指します。

pUserId

注釈のユーザー ID を受け取る BSTR を指します。

pDateTime

注釈の日時を受け取る BSTR を指します。

pPage

注釈の文書ページ番号を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

pPublic

注釈が共通または専用のどちらであるかを指示するブール・フラグを受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pCanBeCopied

注釈を別のサーバーにコピーできるかどうかを指示するブール・フラグを受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 注釈を取り出します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocAnnotations, GetDocAnnotation

例 以下の例では、文書の注釈を取り出します。

```
Dim rc, j As Integer
Dim num_notes, page, ispublic, canbecopied As Variant
Dim text As String
Dim userid As String
Dim datetime As String

rc = ArsOle.GetNumDocAnnotations( num_notes )
if rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_notes -1
    rc = ArsOle.GetAnnotationForDoc( j,           _
                                     text,        _
                                     userid,      _
                                     datetime,    _
                                     page,        _
                                     ispublic,    _
                                     canbecopied )
    if rc <> ARS_OLE_RC_SUCCESS Then
        MsgBox "ERROR"
        End
    End If
Next j
```

GetAnnotationStatus

メソッド:

short GetAnnotationStatus(

long **Index**,

VARIANT * **pStatus**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。この値がゼロ未満であれば、開いている文書について状況が戻されます。

pStatus

注釈の状況を受け取る変数を指します。これは、ARSOLEEX.H の中にある ARS_OLE_ANNOTATION_YES などの注釈状況値のうちの 1 つです。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に注釈の状況を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: AnnotateDoc, GetNumDocAnnotations, GetDocAnnotation, GetAnnotationForDoc

例 以下の例では、文書の注釈の状況を取得します。

C/C + + の例

```
VARIANT status;
CArsOle * pArsCtrl;
short rc;

 .
 .
 .

rc = pArsCtrl->GetAnnotationStatus( -1, &status );

if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

 .
 .
 .
```

Visual Basic の例

```
Dim rc As Integer
Dim status As Variant

 .
 .
 .

rc = ArsOle.GetAnnotationStatus( -1, status )
if rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

 .
 .
 .
```

GetControlId

メソッド:

short GetControlId(

VARIANT * **pControlId**)

パラメーター:

pControlId

コントロール ID を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

説明: 指定した変数にコントロール ID を戻します。このコントロール ID を使用して、別の OnDemand OLE Control に関連した情報を参照できます。コントロール ID の説明については、3 ページの『単一フォルダーの複数文書の表示』を参照してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenDoc

例: 以下の例では、コントロール ID を取り出します。

C/C + + の例

```
long ControlId;

  .
  .

CArsOle * pArsCtrl;
VARIANT control_id;
short rc;

  .
  .

 rc = pArsCtrl->GetControlId( &control_id );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

 ControlId = control_id.lVal;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim control_id As Variant

 .
 .

rc = ArsOle.GetControlId (control_id)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

GetDocAnnotation

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

```
short  GetDocAnnotation(
    short  Index,
    LPUNKNOWN  pText,
    LPUNKNOWN  pUserId,
    LPUNKNOWN  pDateTime,
    VARIANT  *  pPage,
    VARIANT  *  pPublic,
    VARIANT  *  pCanBeCopied  )
```

パラメーター:

Index

戻される注釈の 0 を基準とした索引を指定します。0 以上かつ GetNumDocAnnotations によって戻される値未満の数値を指定する必要があります。

pText

注釈のテキストを受け取るストリングを指します。

pUserId

注釈のユーザー ID を受け取るストリングを指します。

pDateTime

注釈の日時を受け取るストリングを指します。

pPage

注釈の文書ページ番号を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

pPublic

注釈が共通または専用のどちらであるかを指示するブール・フラグを受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pCanBeCopied

注釈を別のサーバーにコピーできるかどうかを指示するブール・フラグを受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 注釈データを取り出します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocAnnotations, GetAnnotationForDoc

例 以下の例では、文書の注釈を取り出します。

```
VARIANT num_notes, page, ispublic, canbecopied;
CArsOle * pArsCtrl;
short rc, j;
char * pText, userid[100], datetime[200];

rc = pArsCtrl->GetNumDocAnnotations( &num_notes );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pText = new char[35000];

for ( j = 0; j < num_notes.iVal; j++ )
{
  rc = pArsCtrl->GetDocAnnotation( j,
                                   (LPUNKNOWN)pText,
                                   (LPUNKNOWN)userid,
                                   (LPUNKNOWN)datetime,
                                   &page,
                                   &ispublic,
                                   &canbecopied );

  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  // Process annotation

}

delete pText;
```

GetDocBackgroundColor

メソッド:

short GetDocBackgroundColor(

VARIANT * **pColor**,

VARIANT * **pChangeable**)

パラメーター:

pColor

現行の文書の背景色を受け取る変数を指します。これは、ARSOLEEX.H の中にある ARS_OLE_COLOR_WHITE などの色の値のうちの 1 つです。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pChangeable

文書の背景色を変更できるかどうかに関する指示を受け取る変数を指します。色を変更できる場合は、この変数には、リターン時に非ゼロ値が入ります。変更できない場合は、ゼロが入ります。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 現行の文書の背景色、および変更可能かどうかを示す標識を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocBackgroundColor

例: 以下の例では、現行の文書の背景色を取り出し、背景色を変更できない場合にはメニュー項目を使用不可にします。

C/C + + の例

```
CArsOle * pArsCtrl;
CMenu * pSubMenu;
short rc, back_color;
VARIANT current_color, changeable;

  .
  .

rc = pArsCtrl->GetDocBackgroundColor( &current_color, &changeable );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

back_color = current_color.iVal;

pSubMenu->EnableMenuItem(
            ID_VIEW_BKGD_COLOR,
            MF_BYCOMMAND | ( changeable.iVal ? MF_ENABLED : MF_GRAYED ) );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim back_color, changeable As Variant

 .
 .

rc = ArsOle.GetDocBackgroundColor (back_color, changeable)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If changeable <> 0 Then
   menuBackgroundColor.Enabled = True
Else
   menuBackgroundColor.Enabled = False
End If

 .
 .
```

GetDocCurrentPage

メソッド:

short GetDocCurrentPage(

VARIANT * **pPage**)

パラメーター:

pPage

開いている文書の現行ページ番号を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

説明: 指定した変数に、開いている文書の現行ページ番号を戻します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: SetDocCurrentPage, GetDocNumPages

例: 以下の例では、開いている文書の現行ページ番号を取り出します。

C/C + + の例

```
CArsOle * pArsCtrl;
VARIANT vari;
long page_num;
short rc;

  .
  .

 rc = pArsCtrl->GetDocCurrentPage( &vari );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

 page_num = var.lVal;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim page_num As Variant

 .
 .

rc = ArsOle.GetDocCurrentPage (page_num)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

GetDocDisplayValue

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetDocDisplayValue(

long **DocIndex**,

short **ValueIndex**,

BSTR * **pValue**)

パラメーター:

DocIndex

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。

ValueIndex

戻される値の 0 を基準とした索引を指定します。0 以上かつ GetNumFolderDisplayFields によって戻される値未満の数値を指定する必要があります。

pValue

値を受け取る BSTR を指します。

説明: 指定した値を pValue に戻します。

GetDocDisplayValue または GetDocDisplayValues を使用して文書表示値を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocsInList, GetNumFolderDisplayFields, GetFolderDisplayFieldNames, GetDocDisplayValues, OpenDoc

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
 .
 .
 .
rc = ArsOle.GetNumFolderDisplayFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderDisplayFieldName(count, Temp)
    Names(count) = Temp
Next count

rc = ArsOle.GetNumDocsInList(num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_docs -1
    For i = 0 To num_fields -1
        rc = ArsOle.GetDocDisplayValue(j, i, Temp)

        Line = Line + Names(i) + " = " + Temp
        If i < num_fields Then
            Line = Line + ", "
        End If
    Next i

    lbDocs.AddItem Line
Next j
 .
 .
```

```
 .
'During OK button processing:

rc = ArsOle.OpenDoc (lbDocs.List(lbDocs.ListIndex), "", 0)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetDocDisplayValues

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetDocDisplayValues(

long **Index**,

IUnknown * **pValues**,

short **MaxValues**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。

pValues

Index で指定した文書に対するフォルダー表示フィールドの値を受け取る ArsOleValue の配列を指します。表示フィールドと同じ数の値があります。配列には、少なくとも MaxValues のエレメントがなければなりません。

MaxValues

戻される値の最大数を指定します。

説明: MaxValues の最大数までの、文書に対するフォルダー表示フィールドの値を pValues に戻します。それぞれの名前は、ヌル終了文字ストリングです。

その値は、表示フィールド名が GetFolderDisplayFieldNames メソッドによって戻されるのと同じ順序で配列に入ります。

GetDocDisplayValue または GetDocDisplayValues を使用して文書表示値を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocsInList, GetNumFolderDisplayFields, GetFolderDisplayFieldNames, GetDocDisplayValue, OpenDoc

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
ArsOleValue * pValues;
CListBox * pDocList;
char * pLine;
short rc, k, opr, num_fields;
long j, num_docs;
int size;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderDisplayFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderDisplayFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->GetNumDocsInList( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_docs = var.lVal;

pValues = new ArsOleValue[ max( num_fields, 1 ) ];

size = num_fields * ( sizeof(ArsOleName) + sizeof(ArsOleValue) + 5 );
pLine = new char[ size ];
for ( j = 0, pLine[0] = '¥0'; j < num_docs; j++ )
{
  rc = pArsCtrl->GetDocDisplayValues( j, pValues, num_fields );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;
  .
  .

  .
  .
  for ( k = 0; k < num_fields; k++ )
  {
    strcat( pLine, pNames[k] );
    strcat( pLine, " = " );
    strcat( pLine, pValues[k] );
    if ( k < num_fields - 1 )
      strcat( pLine, ", " );
  }
  pDocList->InsertString( -1, pLine );
}
pDocList->SetCurSel( 0 );
  .
  .
// During OK button processing:

rc = pArsCtrl->OpenDoc( (long)pDocList->GetCurSel( ) , NULL, 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
```

GetDocImageColor

メソッド:

short GetDocImageColor(

VARIANT * **pColor**,

VARIANT * **pChangeable**)

パラメーター:

pColor

現行の文書のイメージ・カラーを受け取る変数を指します。これは、ARSOLEEX.H の中にある ARS_OLE_COLOR_BLACK などの色の値のうちの 1 つです。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pChangeable

文書のイメージ・カラーを変更できるかどうかに関する指示を受け取る変数を指します。色を変更できる場合は、この変数には、リターン時に非ゼロ値が入ります。変更できない場合は、ゼロが入ります。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 現行の文書のイメージ・カラー、および変更可能かどうかを示す標識を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocImageColor

例: 以下の例では、現行の文書のイメージ・カラーを取り出し、イメージ・カラーを変更できない場合にはメニュー項目を使用不可にします。

C/C + + の例

```
CArsOle * pArsCtrl;
CMenu * pSubMenu;
short rc, image_color;
VARIANT current_color, changeable;

  .
  .

rc = pArsCtrl->GetDocImageColor( &current_color, &changeable );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

image_color = current_color.iVal;

pSubMenu->EnableMenuItem(
            ID_VIEW_IMAGE_COLOR,
            MF_BYCOMMAND | ( changeable.iVal ? MF_ENABLED : MF_GRAYED ) );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim current_color, changeable As Variant

.
.
```

```
rc = ArsOle.GetDocImageColor (current_color, changeable)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If changeable <> 0 Then
   menuImageColor.Enabled = True
Else
   menuImageColor.Enabled = False
End If

 .
 .
```

GetDocImageIntensity

メソッド:

short GetDocImageIntensity(

VARIANT * **pIntensity**,

VARIANT * **pChangeable**)

パラメーター:

pIntensity

現行の文書のイメージ輝度を受け取る変数を指します。これは、ARSOLEEX.H の中にある ARS_OLE_INTENSITY_NORMAL などの輝度値のうちの 1 つです。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pChangeable

文書のイメージ輝度を変更できるかどうかに関する指示を受け取る変数を指します。輝度を変更できる場合は、この変数には、リターン時に非ゼロ値が入ります。変更できない場合は、ゼロが入ります。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 現行の文書のイメージ輝度、および変更可能かどうかを示す標識を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocImageIntensity

例: 以下の例では、現行の文書のイメージ輝度を取り出し、輝度を変更できない場合にはメニュー項目を使用不可にします。

C/C + + の例

```
CArsOle * pArsCtrl;
CMenu * pSubMenu;
short rc, image_intensity;
VARIANT current_intensity, changeable;

  .
  .

rc = pArsCtrl->GetDocImageIntensity( &current_intensity, &changeable );
```

```
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

image_intensity = current_intensity.iVal;

pSubMenu->EnableMenuItem(
            ID_VIEW_IMAGE_INTENSITY,
            MF_BYCOMMAND | ( changeable.iVal ? MF_ENABLED : MF_GRAYED ) );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim current_intensity, changeable As Variant

 .
 .

rc = ArsOle.GetDocImageIntensity (current_intensity, changeable)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If changeable <> 0 Then
   menuImageIntensity.Enabled = True
Else
   menuImageIntensity.Enabled = False
End If

 .
 .
```

GetDocNumPages

メソッド:

short GetDocNumPages(

VARIANT * **pNumPages**)

パラメーター:

pNumPages

開いている文書のページ数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

説明: 指定した変数に、開いている文書のページ数を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenDoc, GetDocCurrentPage, SetDocCurrentPage

例: 以下の例では、開いている文書のページ数を取り出します。

C/C + + の例

```
CArsOle * pArsCtrl;
VARIANT vari;
long num_pages;
short rc;

  .
```

```
  .

  rc = pArsCtrl->GetDocNumPages( &vari );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  num_pages = var.lVal;

  .
  .
```

Visual Basic Example

```
Dim rc As Integer
Dim num_pages As Variant

 .
 .

rc = ArsOle.GetDocNumPages (num_pages)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

GetDocRotation

メソッド:

short GetDocRotation(

VARIANT * **pRotation**,

VARIANT * **pChangeable**)

パラメーター:

pRotation

現行の文書の回転値を受け取る変数を指します。これは、ARSOLEEX.H の中にある ARS_OLE_ROTATION_0 などの回転値のうちの 1 つです。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pChangeable

文書の回転値を変更できるかどうかに関する指示を受け取る変数を指します。回転値を変更できる場合は、この変数には、リターン時に非ゼロ値が入ります。変更できない場合は、ゼロが入ります。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 現行の文書の回転値、および変更可能かどうかを示す標識を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocRotation

例: 以下の例では、現行の文書の回転値を取り出し、回転値を変更できない場合にはメニュー項目を使用不可にします。

C/C + + の例

```
CArsOle * pArsCtrl;
CMenu * pSubMenu;
short rc, rotation;
VARIANT current_rotation, changeable;

  .
  .

rc = pArsCtrl->GetDocRotation( &current_rotation, &changeable );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rotation = current_rotation.iVal;

pSubMenu->EnableMenuItem(
            ID_VIEW_ROTATION,
            MF_BYCOMMAND | ( changeable.iVal ? MF_ENABLED : MF_GRAYED ) );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim rotation, changeable As Variant

 .
 .

rc = ArsOle.GetDocRotation (rotation, changeable)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If changeable <> 0 Then
   menuRotation.Enabled = True
Else
   menuRotation.Enabled = False
End If

 .
 .
```

GetDocScrollPositions

メソッド:

short GetDocScrollPositions(

VARIANT * **pHorzPosition**,

VARIANT * **pVertPosition**)

パラメーター:

pHorzPosition

新規水平スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pVertPosition

新規垂直スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に現行スクロール位置を戻します。そのスクロール位置は、スクロール範囲が ARS_OLE_SCROLL_RANGE に設定されていることを前提としています。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照:

例: 以下の例では、開いている文書の現行ページ番号を設定し、現行スクロール位置を更新します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScollBar, * pVertScrollBar;
short rc;
VARIANT horz_position, vert_position;
  .
  .
 rc = pArsCtrl->SetDocCurrentPage( 46 );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .

rc = pArsCtrl->GetDocScrollPositions( &horz_position, &vert_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->SetScrollPos( (int)horz_position.iVal );
pVertScrollBar->SetScrollPos( (int)vert_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim horz_pos, vert_post As Variant

 .
 .

rc = ArsOle.SetDocCurrentPage( 46 )
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

rc = ArsOle.GetDocScrollPositions( horz_pos, vert_pos )
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

sbHorz.Value = horz_pos
sbVert.Value = vert_pos

 .
 .
```

GetDocType

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetDocType(

long **Index**,

VARIANT * **pType**,

LPUNKNOWN **pExtension**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。この値がゼロ未満であれば、開いている文書が使用されます。

pType

指定された文書の文書タイプを受け取る変数を指します。この文書タイプは、ARSOLEEX.H 内にある文書タイプの値 (例: ARS_OLE_DOC_TYPE_AFP) のいずれか 1 つになります。

pExtension

文書のファイル拡張子を受け取るストリングを指します。この値は、文書タイプが ARS_OLE_DOC_TYPE_USER_DEF である場合にのみ戻されます。

説明: 文書タイプを取り出します。文書タイプが ARS_OLE_DOC_TYPE_USER_DEF の場合、ファイル拡張子も取り出されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetTypeForDoc

例 以下の例では、文書リスト内の 3 番目の項目の文書タイプを取り出します。

```
VARIANT type;
char ext[ 20 ];
CArsOle * pArsCtrl;
short rc;
  .
  .
  .

rc = pArsCtrl->GetDocType( 2, &type, extension  );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
  .
```

GetDocZoom

メソッド:

short GetDocZoom(

VARIANT * **pCurrentZoomPercent**,

VARIANT * **pMinZoomPercent**,

VARIANT * **pMaxZoomPercent**)

パラメーター:

pCurrentZoomPercent

現行のズームのパーセント値を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pMinZoomPercent

ズームの最小パーセント値を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pMaxZoomPercent

ズームの最大パーセント値を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に、文書に対するズームの現行、最小、および最大パーセント値を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocZoom

例: 以下の例では、ズームの現行、最小、および最大パーセント値を取り出します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc, current_zoom, min_zoom, max_zoom;
VARIANT var1, var2, var3;

  .
  .

rc = pArsCtrl->GetDocZoom( &var1, &var2, &var3 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

current_zoom = var1.iVal;
min_zoom = var2.iVal;
max_zoom = var3.iVal;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim current_zoom, min_zoom, max_zoom As Variant

 .
```

```
.

rc = ArsOle.GetDocZoom (current_zoom, min_zoom, max_zoom)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

.
.
```

GetFolderDisplayFieldName

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetFolderDisplayFieldName(

short **Index**,

BSTR * **pName**)

パラメーター:

Index

戻される名前の 0 を基準とした索引を指定します。0 以上かつ GetNumFolderDisplayFields によって戻される値未満の数値を指定する必要があります。

pName

フィールド名を受け取る BSTR を指します。

説明: 指定したフィールド名を pName に戻します。

GetFolderDisplayFieldName または GetFolderDisplayFieldNames を使用してフォルダー表示フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderDisplayFields, GetFolderDisplayFieldNames

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
 .
 .
 .
rc = ArsOle.GetNumFolderDisplayFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

```
ReDim Names(num_fields)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderDisplayFieldName(count, Temp)
    Names(count) = Temp
Next count

rc = ArsOle.GetNumDocsInList(num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_docs -1
    For i = 0 To num_fields -1
        rc = ArsOle.GetDocDisplayValue(j, i, Temp)

        Line = Line + Names(i) + " = " + Temp
        If i < num_fields Then
            Line = Line + ", "
        End If
    Next i

    lbDocs.AddItem Line
Next j
 .
 .
 .
'During OK button processing:

rc = ArsOle.OpenDoc (lbDocs.List(lbDocs.ListIndex), "", 0)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetFolderDisplayFieldNames

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetFolderDisplayFieldNames(

IUnknown * **pNames**,

short **MaxNames**)

パラメーター:

pNames

アクティブ・フォルダーの表示フィールド名を受け取る ArsOleName の配列を指します。配列には、少なくとも MaxNames のエレメントがなければなりません。

MaxNames

戻される名前の最大数を指定します。

説明: MaxNames の最大数までのアクティブ・フォルダーの表示フィールド名を pNames に戻します。それぞれの名前は、ヌル終了文字ストリングです。

GetFolderDisplayFieldName または GetFolderDisplayFieldNames を使用してフォルダー表示フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderDisplayFields, GetFolderDisplayFieldName

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
ArsOleValue * pValues;
CListBox * pDocList;
char * pLine;
short rc, k, opr, num_fields;
long j, num_docs;
int size;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderDisplayFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;
  .
  .

  .
  .
pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderDisplayFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->GetNumDocsInList( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_docs = var.lVal;

pValues = new ArsOleValue[ max( num_fields, 1 ) ];

size = num_fields * ( sizeof(ArsOleName) + sizeof(ArsOleValue) + 5 );
pLine = new char[ size ];
for ( j = 0, pLine[0] = '¥0'; j < num_docs; j++ )
{
  rc = pArsCtrl->GetDocDisplayValues( j, pValues, num_fields );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  for ( k = 0; k < num_fields; k++ )
  {
    strcat( pLine, pNames[k] );
    strcat( pLine, " = " );
    strcat( pLine, pValues[k] );
    if ( k < num_fields - 1 )
      strcat( pLine, ", " );
  }
  pDocList->InsertString( -1, pLine );
```

```
}
pDocList->SetCurSel( 0 );
  .
  .
// During OK button processing:

rc = pArsCtrl->OpenDoc( (long)pDocList->GetCurSel( ) , NULL, 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

GetFolderFieldName

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short **GetFolderFieldName**(

short **Index**,

BSTR * **pName**)

パラメーター:

Index

戻される値の 0 を基準とした索引を指定します。0 以上かつ GetNumFolderFields で戻される値未満の数値を指定する必要があります。

pName

フィールド名を受け取る BSTR を指します。

説明: 指定した値を pName に戻します。

GetFolderFieldName または GetFolderFieldNames を使用してフォルダー・フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetFolderFieldNames、GetNumFolderFields、StoreDoc

例: **Visual Basic の例**

```
Dim rc, count As Integer
Dim num_fields As Variant
Dim FieldNames() As String
Dim Temp As String

  .
  .

rc = ArsOle.GetNumFolderFields (num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

ReDim FieldNames (num_fields - 1)

For count = 0 To num_fields - 1
```

```
    rc = ArsOle.GetFolderFieldName (count, Temp)
    FieldNames(count) = Temp
Next count

  .
  .
```

GetFolderFieldNames

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short **GetFolderFieldNames**(

IUnknown * **pNames**,

short **MaxNames**)

パラメーター:

pNames

フォルダー・フィールド名を受け取る ArsOleValues の配列を指します。配列には、少なくとも MaxNames のエレメントがなければなりません。

MaxNames

戻される名前の最大数を指定します。

説明: MaxNames の最大数までのフォルダー・フィールド名を pNames に戻します。それぞれの名前は、ヌル終了文字ストリングです。

その名前は、StoreDoc メソッドで使用されるのと同じ順序で配列に入ります。

GetFolderFieldName または GetFolderFieldNames を使用してフォルダー・フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetFolderFieldName、GetNumFolderFields、StoreDoc

例: 以下の例では、StoreDoc メソッドを示しています。まず、ユーザーがフィールド値を入力できるよう、入力フィールドと一緒にフォルダー・フィールドが表示されます。次に、新規文書を OnDemand に保管するためにその値が使用されます。

C/C++ の例

```
VARIANT var;
CArsOle * pArsCtrl;
ArsOleName * pNames;
short rc, j;

  .
  .

rc = pArsCtrl->GetNumFolderFields( &var );
if ( rc != ARS_OLE_RC_SUCCESS )
                ERROR;
```

```
// m_NumFolderFields is a class variable
m_NumFolderFields = var.iVal;

pNames = new ArsOleName[ max( m_NumFolderFields, 1 ) ];
rc = pArsCtrl->GetFolderFieldNames( (IUnknown*)pNames,
                                    m_NumFolderFields );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
for ( j = 0; j < m_NumFolderFields; j++ )
    GetDlgItem( IDC_FIELD1_TEXT + j )->SetWindowText( pNames[j] );

// During OK button processing

CArsOle * pArsCtrl;
short rc, j;
CString fields[16];
SAFEARRAY * pSA;
VARIANT var;
BSTR bstrElement;
long i;

pSA = SafeArrayCreateVector(VT_BSTR, 0, m_NumFolderFields);
if ( pSA == NULL )
   ERROR;

for (j = 0; j < m_NumFolderFields; j++)
GetDlgItem( IDC_FIELD1_EDIT + j )->GetWindowText( fields[j] );

for (i = 0; i < m_NumFolderFields; i++)
{
     bstrElement = fields[i].AllocSysString();
     if (bstrElement == NULL)
        ERROR;
     SafeArrayPutElement (pSA, &i, bstrElement);
}

var.vt = VT_ARRAY ¦ VT_BSTR;
var.parray = pSA;

rc = pArsCtrl->StoreDoc( "G:¥¥download¥¥file.afp",
                           "BKH-CRD",
                           "BKH-CRD",
                           &var );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
```

GetFolderName

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetFolderName(

short **Index**,

BSTR * **pName**)

パラメーター:

Index

戻される名前の 0 を基準とした索引を指定します。0 以上かつ GetNumFolders によって戻される値未満の数値を指定する必要があります。

pName

フォルダー名を受け取る BSTR を指します。

説明: 指定したフォルダー名を pName に戻します。

GetFolderName または GetFolderNames を使用してフォルダー名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolders, GetFolderNames, OpenFolder

例:

Visual Basic の例

```
Dim rc, count As Integer
Dim num_folders As Variant
Dim Temp As String

 .
 .

rc = ArsOle.GetNumFolders (num_folders)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For count = 0 To num_folders -1
    rc = ArsOle.GetFolderName(count, Temp)
    lbFolders.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.OpenFolder (lbFolders.List(lbFolders.ListItem))
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetFolderNames

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetFolderNames(

IUnknown * **pNames**,

short **MaxNames**)

パラメーター:

pNames

現行サーバーで使用可能なフォルダーの名前を受け取る ArsOleName の配列を指します。配列には、少なくとも MaxNames のエレメントがなければなりません。

MaxNames

戻される名前の最大数を指定します。

説明: MaxNames の最大数までの、現行サーバーで使用可能なフォルダーの名前を pNames に戻します。それぞれの名前は、ヌル終了文字ストリングです。

GetFolderName または GetFolderNames を使用してフォルダー名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolders, GetFolderName, OpenFolder

例: 以下の例では、現行サーバーで使用可能なすべてのフォルダーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したフォルダーを取り出したあと、そのフォルダーを開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pFolderNames;
CComboBox * pFoldersList;
char folder[ sizeof( ArsOleName ) ];
short rc, j, num_folders;
int index;
VARIANT vari;
  .
  .

// During dialog initialization:

rc = pArsCtrl->GetNumFolders( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_folders = var.iVal;

pFolderNames = new ArsOleName[ max( num_folders, 1 ) ];
rc = pArsCtrl->GetFolderNames( (IUnknown*)pFolderNames, num_folders );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_folders; j++ )
  index = pFoldersList->AddString( pFolderNames[j] );
pFoldersList->SetCurSel( 0 );
  .
  .
 // During OK button processing:

 pFoldersList->GetWindowText( folder, sizeof(folder) );

 rc = pArsCtrl->OpenFolder( folder );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

GetFolderSearchFieldName

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetFolderSearchFieldName(

short **Index**,

BSTR * **pName**)

パラメーター:

Index

戻される名前の 0 を基準とした索引を指定します。0 以上かつ GetNumFolderSearchFields によって戻される値未満の数値を指定する必要があります。

pName

フィールド名を受け取る BSTR を指します。

説明: 指定したフィールド名を pName に戻します。

GetFolderSearchFieldName または GetFolderSearchFieldNames を使用してフォルダー検索フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderSearchFields, GetFolderSearchFieldNames, SetFolderSearchFieldData, SearchFolder

例:

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
Dim Oprs As Variant
 .
 .
 .
Oprs = Array ("Equal", "Not Equal", ..., "Like", "Not Like")

rc = ArsOle.GetNumFolderSearchFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderSearchFieldName(count, Temp)
    Names(count) = Temp
Next count

for count = 0 To num_fields -1
    lbFieldList.AddItem Names(count)
Next count

for count = 0 To UBound(Oprs)
    lbOprList.AddItem (Oprs(count))
Next count
 .
```

```
 .
 .
' During SET FIELD button processing
rc = ArsOle.SetFolderSearchFieldData (lbFieldList.List(lbFieldList.ListIndex),
                                      lbOprList.ListIndex,
                                      txtValue1.Value,
                                      txtValue2.Value)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

'During OK button processing:

rc = ArsOle.SearchFolder (False)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetFolderSearchFieldNames

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetFolderSearchFieldNames(

IUnknown * **pNames**,

short **MaxNames**)

パラメーター:

pNames

アクティブ・フォルダーの検索フィールド名を受け取る ArsOleName の配列を指します。配列には、少なくとも MaxNames のエレメントがなければなりません。

MaxNames

戻される名前の最大数を指定します。

説明: MaxNames の最大数までのアクティブ・フォルダーの検索フィールド名を pNames に戻します。それぞれの名前は、ヌル終了文字ストリングです。

GetFolderSearchFieldName または GetFolderSearchFieldNames を使用してフォルダー検索フィールド名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderSearchFields, GetFolderSearchFieldName, SetFolderSearchFieldData, SearchFolder

例: 以下の例では、アクティブ・フォルダーの検索フィールドの名前を取り出し、そのフィールドに値を設定する機会をユーザーに提供し、フォルダーの検索を開始します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
CListBox * pFieldList, * pOprList;
CEdit * pValue1, * pValue2;
char name[ sizeof( ArsOleName ) ];
char value1[ sizeof( ArsOleValue ) ];
char value2[ sizeof( ArsOleValue ) ];
short rc, j, opr, num_fields;
VARIANT vari;
  .
  .
struct _OprMap
{
  short  code;
  char * pText;
} OprMap
  .
  .

  .
  .
static OprMap Oprs[] =
  { { ARS_OLE_OPR_EQUAL,                "Equal" },
    { ARS_OLE_OPR_NOT_EQUAL,            "Not Equal" },
        .
        .
    { ARS_OLE_OPR_LIKE,                 "Like" },
    { ARS_OLE_OPR_NOT_LIKE,             "Not Like" } };

#define NUM_OPRS  ( sizeof(Oprs) / sizeof(OprMap) )

// During dialog initialization:

rc = pArsCtrl->GetNumFolderSearchFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderSearchFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_fields; j++ )
  pFieldList->InsertString( -1, pNames[j] );
pFieldList->SetCurSel( 0 );

for ( j = 0; j < NUM_OPRS; j++ )
{
  pOprList->InsertString( -1, Oprs[j].pText );
  pOprList->SetItemData( j, (DWORD)Oprs[j].code );
}
pOprList->SetCurSel( 0 );
  .
  .

  .
  .
// During SET FIELD button processing:

pFieldList->GetText( pFieldList->GetCurSel( ), name );
opr = (short)pOprList->GetItemData( pOprList->GetCurSel( ) );
pValue1->GetWindowText( value1, sizeof(value1) );
pValue2->GetWindowText( value2, sizeof(value2) );

rc = pArsCtrl->SetFolderSearchFieldData( name, opr, value1, value2 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
```

```
  .
  .
// During OK button processing:

rc = pArsCtrl->SearchFolder( FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

GetNumDocAnnotations

メソッド:

short GetNumDocAnnotations(

VARIANT * **pNumAnnotations**)

パラメーター:

pNumAnnotations

文書に付加された注釈の数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

説明: 指定した変数に、文書に付加された注釈の数を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocAnnotation, GetAnnotationForDoc

例: 以下の例では、文書の注釈を取り出します。

C/C + + の例

```
VARIANT num_notes, page, ispublic, canbecopied;
CArsOle * pArsCtrl;
short rc, j;
char * pText, userid[100], datetime[200];

rc = pArsCtrl->GetNumDocAnnotations( &num_notes );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pText = new char[35000];

for ( j = 0; j < num_notes.iVal; j++ )
{
  rc = pArsCtrl->GetDocAnnotation( j,
                                   (LPUNKNOWN)pText,
                                   (LPUNKNOWN)userid,
                                   (LPUNKNOWN)datetime,
                                   &page,
                                   &ispublic,
                                   &canbecopied );

  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  // Process annotation

}

delete pText;
```

Visual Basic の例

```
Dim rc, j As Integer
Dim num_notes, page, ispublic, canbecopied As Variant
Dim text As String
Dim userid As String
Dim datetime As String

rc = ArsOle.GetNumDocAnnotations( num_notes );
if rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_notes -1
    rc = ArsOle.GetAnnotationForDoc( j,            _
                                     text,         _
                                     userid,       _
                                     datetime,     _
                                     page,         _
                                     ispublic,     _
                                     canbecopied )
    if rc <> ARS_OLE_RC_SUCCESS Then
        MsgBox "ERROR"
        End
    End If

    ' Process Annotation

Next j
```

GetNumDocsInList

メソッド:

short GetNumDocsInList(

VARIANT * **pNumDocs**)

パラメーター:

pNumDocs

アクティブ・フォルダーの文書リストに入っている文書の数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I4 に設定されます。

説明: 指定した変数に、アクティブ・フォルダーの文書リストに入っている文書の数を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocDisplayValues, OpenDoc

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
ArsOleValue * pValues;
CListBox * pDocList;
char * pLine;
short rc, k, opr, num_fields;
```

```
long j, num_docs;
int size;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderDisplayFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderDisplayFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->GetNumDocsInList( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_docs = var.lVal;

pValues = new ArsOleValue[ max( num_fields, 1 ) ];
  .
  .

  .
  .
size = num_fields * ( sizeof(ArsOleName) + sizeof(ArsOleValue) + 5 );
pLine = new char[ size ];
for ( j = 0, pLine[0] = '¥0'; j < num_docs; j++ )
{
  rc = pArsCtrl->GetDocDisplayValues( j, pValues, num_fields );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  for ( k = 0; k < num_fields; k++ )
  {
    strcat( pLine, pNames[k] );
    strcat( pLine, " = " );
    strcat( pLine, pValues[k] );
    if ( k < num_fields - 1 )
      strcat( pLine, ", " );
  }
  pDocList->InsertString( -1, pLine );
}
pDocList->SetCurSel( 0 );
  .
  .
// During OK button processing:

rc = pArsCtrl->OpenDoc( (long)pDocList->GetCurSel( ) , NULL, 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
 .
 .
 .
rc = ArsOle.GetNumFolderDisplayFields(num_fields)
```

```
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderDisplayFieldName(count, Temp)
    Names(count) = Temp
Next count

rc = ArsOle.GetNumDocsInList(num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_docs -1
    For i = 0 To num_fields -1
        rc = ArsOle.GetDocDisplayValue(j, i, Temp)

        Line = Line + Names(i) + " = " + Temp
        If i < num_fields Then
            Line = Line + ", "
        End If
    Next i

    lbDocs.AddItem Line
Next j
 .
 .
 .
'During OK button processing:

rc = ArsOle.OpenDoc (lbDocs.List(lbDocs.ListIndex), "", 0)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetNumFolderDisplayFields

メソッド:

short GetNumFolderDisplayFields(

VARIANT * **pNumFields**)

パラメーター:

pNumFields

アクティブ・フォルダーの表示フィールドの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数にアクティブ・フォルダーの表示フィールドの数を戻します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetFolderDisplayFieldNames

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
ArsOleValue * pValues;
CListBox * pDocList;
char * pLine;
short rc, k, opr, num_fields;
long j, num_docs;
int size;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderDisplayFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderDisplayFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->GetNumDocsInList( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_docs = var.lVal;
  .
  .

  .
  .
pValues = new ArsOleValue[ max( num_fields, 1 ) ];
size = num_fields * ( sizeof(ArsOleName) + sizeof(ArsOleValue) + 5 );
pLine = new char[ size ];
for ( j = 0, pLine[0] = '¥0'; j < num_docs; j++ )
{
  rc = pArsCtrl->GetDocDisplayValues( j, pValues, num_fields );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  for ( k = 0; k < num_fields; k++ )
  {
    strcat( pLine, pNames[k] );
    strcat( pLine, " = " );
    strcat( pLine, pValues[k] );
    if ( k < num_fields - 1 )
      strcat( pLine, ", " );
  }
  pDocList->InsertString( -1, pLine );
}
pDocList->SetCurSel( 0 );
  .
  .
// During OK button processing:

rc = pArsCtrl->OpenDoc( (long)pDocList->GetCurSel( ) , NULL, 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
 .
 .
 .
rc = ArsOle.GetNumFolderDisplayFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderDisplayFieldName(count, Temp)
    Names(count) = Temp
Next count

rc = ArsOle.GetNumDocsInList(num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_docs -1
    For i = 0 To num_fields -1
        rc = ArsOle.GetDocDisplayValue(j, i, Temp)

        Line = Line + Names(i) + " = " + Temp
        If i < num_fields Then
            Line = Line + ", "
        End If
    Next i

    lbDocs.AddItem Line
Next j
 .
 .
 .
'During OK button processing:

rc = ArsOle.OpenDoc (lbDocs.List(lbDocs.ListIndex), "", 0)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetNumFolderFields

メソッド:

short **GetNumFolderFields**(

VARIANT * **pNumfields**)

パラメーター:

pNumfields

アクティブ・フォルダーのフォルダー・フィールドの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数にアクティブ・フォルダーのフォルダー・フィールドの数を戻します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: GetFolderFieldName、GetFolderFieldNames

例: 以下の例では、StoreDoc メソッドを示しています。まず、ユーザーがフィールド値を入力できるよう、入力フィールドと一緒にフォルダー・フィールドが表示されます。次に、新規文書を OnDemand に保管するためにその値が使用されます。

C/C++ の例

```
VARIANT var;
CArsOle * pArsCtrl;
ArsOleName * pNames;
short rc, j;
  .
  .
rc = pArsCtrl->GetNumFolderFields( &var );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

// m_NumFolderFields is a class variable
m_NumFolderFields = var.iVal;

pNames = new ArsOleName[ max( m_NumFolderFields, 1 ) ];
rc = pArsCtrl->GetFolderFieldNames( (IUnknown*)pNames,
   m_NumFolderFields );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

for ( j = 0; j < m_NumFolderFields; j++ )
   GetDlgItem( IDC_FIELD1_TEXT + j )->SetWindowText( pNames[j] );

// During OK button processing
CArsOle * pArsCtrl;
short rc, j;
CString fields[16];
SAFEARRAY * pSA;
VARIANT var;
BSTR bstrElement;
long i;

pSA = SafeArrayCreateVector(VT_BSTR, 0, m_NumFolderFields);
if ( pSA == NULL )
   ERROR;

for (j = 0; j < m_NumFolderFields; j++)
   GetDlgItem( IDC_FIELD1_EDIT + j )->GetWindowText( fields[j] );

for (i = 0; i < m_NumFolderFields; i++)
{
   bstrElement = fields[i].AllocSysString();
   if (bstrElement == NULL)
      ERROR;
   SafeArrayPutElement (pSA, &i, bstrElement);
}

var.vt = VT_ARRAY | VT_BSTR;
var.parray = pSA;

rc = pArsCtrl->StoreDoc( "G:¥¥download¥¥file.afp",
                         "BKH-CRD",
```

```
                                "BKH-CRD",
                                &var );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_fields As Variant
Dim FieldNames() As String
Dim Temp As String
  .
  .

rc = ArsOle.GetNumFolderFields (num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

ReDim FieldNames (num_fields - 1)

For count = 0 To num_fields - 1
   rc = ArsOle.GetFolderFieldName (count, Temp)
   FieldNames(count) = Temp
Next count

  .
  .
```

GetNumFolders

メソッド:

short GetNumFolders(

VARIANT * **pNumFolders**)

パラメーター:

pNumFolders

現行サーバーで使用可能なフォルダーの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に、現行サーバーで使用可能なフォルダーの数を戻します。この値を GetFolderNames メソッドで使用してフォルダーを開く準備を行うことができます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetFolderNames, OpenFolder

例: 以下の例では、現行サーバーで使用可能なすべてのフォルダーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したフォルダーを取り出したあと、そのフォルダーを開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pFolderNames;
CComboBox * pFoldersList;
char folder[ sizeof( ArsOleName ) ];
```

```
short rc, j, num_folders;
int index;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolders( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_folders = var.iVal;

pFolderNames = new ArsOleName[ max( num_folders, 1 ) ];
rc = pArsCtrl->GetFolderNames( (IUnknown*)pFolderNames, num_folders );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_folders; j++ )
  index = pFoldersList->AddString( pFolderNames[j] );
pFoldersList->SetCurSel( 0 );
  .
  .

 // During OK button processing:

 pFoldersList->GetWindowText( folder, sizeof(folder) );

 rc = pArsCtrl->OpenFolder( folder );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_folders As Variant
Dim Temp As String

 .
 .

rc = ArsOle.GetNumFolders (num_folders)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim FolderNames(num_folders -1)

For count = 0 To num_folders -1
    rc = ArsOle.GetFolderName(count, Temp)
    lbFolders.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.OpenFolder (lbFolders.List(lbFolders.ListItem))
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetNumFolderSearchFields

メソッド:

short GetNumFolderSearchFields(

VARIANT * **pNumFields**)

パラメーター:

pNumFields

アクティブ・フォルダーの検索フィールドの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数にアクティブ・フォルダーの検索フィールドの数を戻します。この値を GetFolderSearchFieldNames メソッドで使用して、フォルダーの検索フィールド値を設定する準備を行うことができます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetFolderSearchFieldNames, SetFolderSearchFieldData, SearchFolder

例: 以下の例では、アクティブ・フォルダーの検索フィールドの名前を取り出し、そのフィールドに値を設定する機会をユーザーに提供し、フォルダーの検索を開始します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
CListBox * pFieldList, * pOprList;
CEdit * pValue1, * pValue2;
char name[ sizeof( ArsOleName ) ];
char value1[ sizeof( ArsOleValue ) ];
char value2[ sizeof( ArsOleValue ) ];
short rc, j, opr, num_fields;
VARIANT vari;
  .
  .
struct _OprMap
{
  short  code;
  char * pText;
} OprMap

static OprMap Oprs[] =
  { { ARS_OLE_OPR_EQUAL,                "Equal" },
    { ARS_OLE_OPR_NOT_EQUAL,            "Not Equal" },
        .
        .
    { ARS_OLE_OPR_LIKE,                 "Like" },
    { ARS_OLE_OPR_NOT_LIKE,             "Not Like" } };

#define NUM_OPRS  ( sizeof(Oprs) / sizeof(OprMap) )
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderSearchFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;
```

```
pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderSearchFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_fields; j++ )
  pFieldList->InsertString( -1, pNames[j] );
pFieldList->SetCurSel( 0 );

for ( j = 0; j < NUM_OPRS; j++ )
{
  pOprList->InsertString( -1, Oprs[j].pText );
  pOprList->SetItemData( j, (DWORD)Oprs[j].code );
}
pOprList->SetCurSel( 0 );
  .
  .
// During SET FIELD button processing:

pFieldList->GetText( pFieldList->GetCurSel( ), name );
opr = (short)pOprList->GetItemData( pOprList->GetCurSel( ) );
pValue1->GetWindowText( value1, sizeof(value1) );
pValue2->GetWindowText( value2, sizeof(value2) );

rc = pArsCtrl->SetFolderSearchFieldData( name, opr, value1, value2 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
// During OK button processing:

rc = pArsCtrl->SearchFolder( FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
Dim Oprs As Variant
 .
 .
 .
Oprs = Array ("Equal", "Not Equal", ..., "Like", "Not Like")

rc = ArsOle.GetNumFolderSearchFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderSearchFieldName(count, Temp)
    Names(count) = Temp
Next count

for count = 0 To num_fields -1
    lbFieldList.AddItem Names(count)
Next count
```

```
for count = 0 To UBound(Oprs)
    lbOprList.AddItem (Oprs(count))
Next count
 .
 .
 .
' During SET FIELD button processing
rc = ArsOle.SetFolderSearchFieldData (lbFieldList.List(lbFieldList.ListIndex),
                                      lbOprList.ListIndex,
                                      txtValue1.Value,
                                      txtValue2.Value)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
'During OK button processing:

rc = ArsOle.SearchFolder (False)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetNumServerPrinters

メソッド:

short GetNumServerPrinters(

VARIANT * **pNumServerPrinters**)

パラメーター:

pNumServerPrinters

現行サーバーで使用可能なサーバー・プリンターの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に、使用可能なサーバー・プリンターの数を戻します。この値は、GetServerPrinter メソッドと GetServerPrinterInfo メソッドで使用できます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetServerPrinter, GetServerPrinterInfo

例: 以下の例では、使用可能なサーバー・プリンターの名前と属性を取り出します。

C/C + + の例

```
CArsOle * pArsCtrl;

short rc, j, num_prts;
char name[ 200 ];
VARIANT vari, type;
  .
  .
  .

rc = pArsCtrl->GetNumServerPrinters( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_prts = vari.iVal;
```

```
for ( j = 0; j < num_prts; j++ )
{
  rc = pArsCtrl->GetServerPrinter( j, name, type );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  // Process name and type

}
  .
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_prts, type As Variant
Dim name As String

 .
 .

rc = ArsOle.GetNumServers (num_prts)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

For count = 0 To num_prts -1
   rc = ArsOle.GetServerPrinterInfo (count, name, type)
   ' Process name and type
Next count

 .
 .
```

GetNumServers

メソッド:

short GetNumServers(

VARIANT * **pNumServers**)

パラメーター:

pNumServers

ログオンできるサーバーの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 指定した変数に、ログオンできるサーバーの数を戻します。この値を GetServerNames メソッドで使用して、ログオンの準備をすることができます。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: GetServerNames, Logon

例: 以下の例では、ログオンできるすべてのサーバーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したサーバー、ユーザー ID、およびパスワードを取り出したあと、ログオンを実行します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pServerNames;
CComboBox * pServersList;
CEdit * pUserId;
CEdit * pPassword;
char server[ sizeof( ArsOleName ) ];
char user[ sizeof( ArsOleName ) ];
char password[ sizeof( ArsOleName ) ];
short rc, j, num_servers;
int index;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumServers( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_servers = var.iVal;

pServerNames = new ArsOleName[ max( num_servers, 1 ) ];
rc = pArsCtrl->GetServerNames( (IUnknown*)pServerNames, num_servers );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_servers; j++ )
  index = pServersList->AddString( pServerNames[j] );
pServersList->SetCurSel( 0 );
  .
  .
 // During OK button processing:

 pServersList->GetWindowText( server, sizeof(server) );
 pUserId->GetWindowText( user, sizeof(user) );
 pPassword->GetWindowText( password, sizeof(password) );

 rc = pArsCtrl->Logon( server, user, password );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_servers As Variant
Dim Temp As String

 .
 .

rc = ArsOle.GetNumServers (num_servers)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For count = 0 To num_servers -1
    rc = ArsOle.GetServerName(count, Temp)
    lbServers.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.Logon (lbServers.List(lbServers.ListItem), txtUser.Value, txtPasswd.Value)
```

```
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetResourceCacheMode

メソッド:

short **SetResourceCacheMode**(

short **Mode**)

パラメーター:

Mode 新しいリソース・キャッシュ・モードを指定します。このモードは、ARSOLEEX.H の中にある以下のモード値の 1 つにする必要があります。

ARS_OLE_RES_MODE_RETAIN

ARS_OLE_RES_MODE_ERASE_AFTER_RETRIEVE

ARS_OLE_RES_MODE_RETAIN は、RetrieveDoc メソッドの実行中に取り戻されたリソース・グループを、以降の RetrieveDoc または OpenDoc メソッドが使用できるように、保存しておくことを指定します。これらのリソース・グループが入っているファイルは、コントロールのすべてのインスタンスが終了するまで、消去されません。これは、デフォルト・モードです。

ARS_OLE_RES_MODE_ERASE_AFTER_RETRIEVE は、RetrieveDoc メソッドの実行中に取り戻されたリソース・グループのファイルを、呼び出し元に戻る前に消去することを指定します。以降の RetrieveDoc または OpenDoc メソッドが同じリソース・グループにアクセスする必要のある場合は、別に検索が必要となります。

説明: 要求されたモードが設定されます。

リターン値:

戻りコードの説明は、「*Windows クライアント・カスタマイズ・ガイド*」の「戻りコード」を参照してください。

参照: RetrieveDoc

例:

以下の例では、リソース・キャッシュ・モードを保存に設定します。

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;
.
.
rc = pArsCtrl->SetResourceCacheMode( ARS_OLE_RES_MODE_RETAIN );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
.
.
```

Visual Basic の例

```
Dim rc As Integer
.
.
rc = ArsOle.SetResourceCacheMode (ARS_OLE_RES_MODE_RETAIN)
if rc <> ARS_OLE_RC_SUCCESS Then
```

```
   MsgBox "ERROR"
   End
End If
.
.
```

GetServerName

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetServerName(

short **Index**,

BSTR * **pName**)

パラメーター:

Index

戻される名前の 0 を基準とした索引を指定します。0 以上かつ GetNumServers によって戻される値未満の数値を指定する必要があります。

pName

サーバー名を受け取る BSTR を指します。

説明: 指定したサーバー名を pName に戻します。

GetServerName または GetServerNames を使用してサーバー名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumServers, GetServerNames, Logon

例:

Visual Basic の例

```
Dim rc, count As Integer
Dim num_servers As Variant
Dim Temp As String

 .
 .

rc = ArsOle.GetNumServers (num_servers)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For count = 0 To num_servers -1
    rc = ArsOle.GetServerName(count, Temp)
    lbServers.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.Logon (lbServers.List(lbServers.ListItem), txtUser.Value, txtPasswd.Value)
```

```
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

GetServerNames

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetServerNames(

IUnknown * **pNames**,

short **MaxNames**)

パラメーター:

pNames

ログオンできるサーバーの名前を受け取る ArsOleName の配列を指します。配列には、少なくとも MaxNames のエレメントがなければなりません。

MaxNames

戻される名前の最大数を指定します。

説明: MaxNames の最大数までのログオンできるサーバーの名前を pNames に戻します。それぞれの名前は、ヌル終了文字ストリングです。

GetServerName または GetServerNames を使用してサーバー名を取り出すことができます。アプリケーションでは、その環境に好適なほうのメソッドを使用してください。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumServers, GetServerName, Logon

例: 以下の例では、ログオンできるすべてのサーバーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したサーバー、ユーザー ID、およびパスワードを取り出したあと、ログオンを実行します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pServerNames;
CComboBox * pServersList;
CEdit * pUserId;
CEdit * pPassword;
char server[ sizeof( ArsOleName ) ];
char user[ sizeof( ArsOleName ) ];
char password[ sizeof( ArsOleName ) ];
short rc, j, num_servers;
int index;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumServers( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
```

```
  ERROR;
num_servers = var.iVal;

pServerNames = new ArsOleName[ max( num_servers, 1 ) ];
rc = pArsCtrl->GetServerNames( (IUnknown*)pServerNames, num_servers );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_servers; j++ )
  index = pServersList->AddString( pServerNames[j] );
pServersList->SetCurSel( 0 );
  .
  .

 // During OK button processing:

 pServersList->GetWindowText( server, sizeof(server) );
 pUserId->GetWindowText( user, sizeof(user) );
 pPassword->GetWindowText( password, sizeof(password) );

 rc = pArsCtrl->Logon( server, user, password );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

GetServerPrinter

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

short GetServerPrinter(

short **Index**,

LPUNKNOWN **pName**,

VARIANT * **pType**)

パラメーター:

Index

戻されるプリンターの 0 を基準とした索引を指定します。0 以上かつ GetNumServerPrinters によって戻される値未満の数値を指定する必要があります。

pName

サーバー・プリンターの名前を受け取るストリングを指します。

pType

サーバー・プリンターのタイプを受け取る変数を指します。このタイプは、ARSOLEEX.H の中にある以下のタイプ値の 1 つです。

ARS_OLE_SERVER_PRINTER_PRINT

ARS_OLE_SERVER_PRINTER_PRINT_WITH_INFO

ARS_OLE_SERVER_PRINTER_FAX

リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: サーバー・プリンター情報を取り出します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: GetNumServerPrinters, GetServerPrinterInfo

例 以下の例では、使用可能なサーバー・プリンターの名前と属性を取り出します。

```
CArsOle * pArsCtrl;

short rc, j, num_prts;
char name[ 200 ];
VARIANT vari, type;
  .
  .
  .

rc = pArsCtrl->GetNumServerPrinters( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_prts = vari.iVal;

for ( j = 0; j < num_prts; j++ )
{
  rc = pArsCtrl->GetServerPrinter( j, name, type );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  // Process name and type

}
  .
  .
  .
```

GetServerPrinterInfo

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetServerPrinterInfo(

short **Index**,

BSTR * **pName**,

VARIANT * **pType**)

パラメーター:

Index

戻されるプリンターの 0 を基準とした索引を指定します。0 以上かつ GetNumServerPrinters によって戻される値未満の数値を指定する必要があります。

pName

サーバー・プリンターの名前を受け取る BSTR を指します。

pType

サーバー・プリンターのタイプを受け取る変数を指します。このタイプは、ARSOLEEX.H の中にある以下のタイプ値の 1 つです。

ARS_OLE_SERVER_PRINTER_PRINT

ARS_OLE_SERVER_PRINTER_PRINT_WITH_INFO

ARS_OLE_SERVER_PRINTER_FAX

説明: サーバー・プリンター情報を取り出します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumServerPrinters, GetServerPrinter

例 以下の例では、使用可能なサーバー・プリンターの名前と属性を取り出します。

```
Dim rc, count As Integer
Dim num_prts, type As Variant
Dim name As String

 .
 .

rc = ArsOle.GetNumServers (num_prts)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

For count = 0 To num_prts -1
   rc = ArsOle.GetServerPrinterInfo (count, name, type)
   ' Process name and type
Next count

 .
 .
```

GetStoreDocInvalidFieldNum

メソッド:

short **GetStoreDocInvalidFieldNum**(

VARIANT * **pFieldNum**)

パラメーター:

pFieldNum

無効フィールドの数を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: StoreDoc メソッドが以前に使用され、戻りコードが次のうちの 1 つだった場合、

ARS_OLE_RC_INVALID_DATE_FIELD
ARS_OLE_RC_INVALID_INTEGER_FIELD
ARS_OLE_RC_INVALID_DECIMAL_FIELD
ARS_OLE_RC_TOO_MANY_VALUE_CHARS
ARS_OLE_RC_INVALID_APPLGRP_FIELD_TYPE,

戻されたフィールド番号は、問題の原因となったフォルダー・フィールドの、0 を基準とした索引です。以前の StoreDoc メソッドがリストされたコードの 1 つを戻していなければ、受け取る値は、予測不能です。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: StoreDoc

例 以下の例は、StoreDoc のあとの、無効フィールドの番号を取り出します。

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;
SAFEARRAY * pSA;
VARIANT var;
BSTR bstrElement;
long i;
.
.
pSA = SafeArrayCreateVector(VT_BSTR, 0, 2);
if ( pSA == NULL )
  ERROR;

bstrElement = SysAllocStringByteLen ("255-546-667", 11);
i = 0;
SafeArrayPutElement (pSA, &i, bstrElement);
bstrElement =SysAllocStringByteLen ("06/07/94", 8);
i = 1;
SafeArrayPutElement (pSA, &i, bstrElement);

var.vt = VT_ARRAY | VT_BSTR;
var.parray = pSA;
rc = pArsCtrl->StoreDoc( "g:¥¥download ¥¥file.afp",
                         "BKH-CRD",
                         "BKH-CRD",
                         &var );
if ( rc != ARS_OLE_RC_SUCCESS )
{
  if ( rc == ARS_OLE_RC_INVALID_DATE_FIELD |
       rc == ARS_OLE_RC_INVALID_INTEGER_FIELD |
       rc == ARS_OLE_RC_INVALID_DECIMAL_FIELD |
       rc == ARS_OLE_RC_TOOMANY_VALUE_CHARS |
       rc == ARS_OLE_RC_INVALID_APPLGRP_FIELD_TYPE )
  {
    VARIANT var1;
    rc = pArsCtrl->GetStoreDocInvalidFieldNum(&var1);
    .
    .
    .
  }
  ERROR;
}
```

Visual Basic の例

```
Dim values(2) As String
Dim rc As Integer
.
.
.
values(0) = "255-546-667"
values(1) = "06/07/94"
var = values

rc = ArsOle.StoreDoc ("g:¥download ¥file.afp",  _
                      "BKH-CRD",                _
                      "BKH-CRD",                _
                      var)
```

```
if rc <> ARS_OLE_RC_SUCCESS Then
  if rc = ARS_OLE_RC_INVALID_DATE_FIELD or
     rc = ARS_OLE_RC_INVALID_INTEGER_FIELD or
     rc = ARS_OLE_RC_INVALID_DECIMAL_FIELD or
     rc = ARS_OLE_RC_TOOMANY_VALUE_CHARS or
     rc = ARS_OLE_RC_INVALID_APPLGRP_FIELD_TYPE then
    rc = ArsOle.GetStoreDocInvalidFieldNum (var1)
  End If
  MsgBox "ERROR"
  End
End If
.
.
.
```

GetTypeForDoc

注: これは、Visual Basic で使用するためのメソッドです。

メソッド:

short GetTypeForDoc(

- long **Index**,
- VARIANT * **pType**,
- BSTR * **pExtension**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。この値がゼロ未満であれば、開いている文書が使用されます。

pType

指定された文書の文書タイプを受け取る変数を指します。この文書タイプは、ARSOLEEX.H 内にある文書タイプの値 (例: ARS_OLE_DOC_TYPE_AFP) のいずれか 1 つになります。

pExtension

文書のファイル拡張子を受け取る BSTR を指します。この値は、文書タイプが ARS_OLE_DOC_TYPE_USER_DEF である場合にのみ戻されます。

説明: 文書タイプを取り出します。文書タイプが ARS_OLE_DOC_TYPE_USER_DEF の場合、ファイル拡張子も取り出されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocType

例 以下の例では、文書リスト内の 3 番目の項目の文書タイプを取り出します。

```
DIM rc As Integer
DIM type As Variant
DIM ext As String
```

```
  .
  .
  .

rc = pArsCtrl->GetTypeForDoc(2, type, ext);
if rc <> ARS_OLE_RC_SUCCESS THEN
  MsgBox "ERROR"
  End
Endif

  .
  .
  .
```

IsDocHorzScrollRequired

メソッド:

short IsDocHorzScrollRequired(

VARIANT * **pRequired**)

パラメーター:

pRequired

結果を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 文書データの幅がコントロール・ウィンドウの幅を超える場合、この結果が生成される変数は非ゼロ値に設定されます。超えない場合は、ゼロに設定されます。文書の表示幅は、そのデータ本来の幅、データのタイプ (つまり、AFP か Linedata か)、およびズーム係数に依存します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: ScrollDocHorz

例: 以下の例では、水平スクロール・バーの範囲を初期設定し、文書のオープン後またはズーム値の変更後にスクロール・バーを表示するか隠して、WM_HSCROLL メッセージを処理します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScrollBar;
short rc, scroll_code;
VARIANT scroll_position, required;
  .
  .
// During initialization:

pHorzScrollBar->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
pHorzScrollBar->ShowScrollBar( FALSE );

// After a document is opened or changing the zoom value:

rc = pArsCtrl->IsDocHorzScrollRequired( &required );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->ShowScrollBar( required.iVal );
```

```
  .
  .
// While processing a WM_HSCROLL message:

scroll_code = (short)LOWORD(wParam);
  .
  .

  .
  .
switch ( scroll_code )
{
  case SB_LINELEFT:
    scroll_code = ARS_OLE_SCROLL_LINELEFT;
    break;
  case SB_LINERIGHT:
    scroll_code = ARS_OLE_SCROLL_LINERIGHT;
    break;
  case SB_PAGELEFT:
    scroll_code = ARS_OLE_SCROLL_PAGELEFT;
    break;
  case SB_PAGERIGHT:
    scroll_code = ARS_OLE_SCROLL_PAGERIGHT;
    break;
  case SB_LEFT:
    scroll_code = ARS_OLE_SCROLL_LEFT;
    break;
  case SB_RIGHT:
    scroll_code = ARS_OLE_SCROLL_RIGHT;
    break;
  case SB_THUMBPOSITION:
    scroll_code = ARS_OLE_SCROLL_THUMBPOSITION;
    break;
  case SB_THUMBTRACK:
    scroll_code = ARS_OLE_SCROLL_THUMBTRACK;
    break;
  default:
    scroll_code = ARS_OLE_SCROLL_ENDSCROLL;
}

if ( scroll_code == (short)ARS_OLE_SCROLL_THUMBPOSITION ||
     scroll_code == (short)ARS_OLE_SCROLL_THUMBTRACK )
{
  scroll_position.vt = VT_I2;
  scroll_position.iVal = (short)HIWORD(wParam);
}

rc = pArsCtrl->ScrollDocHorz( scroll_code, &scroll_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->SetScrollPos( (int)scroll_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim scroll_pos, required As Variant

 .
 .

' During initialization

sbHorz.Min = 0
sbHorz.Max = ARS_OLE_SCROLL_RANGE
sbHorz.Visible = False
```

```
' After a document is opened or changing the zoom value

rc = ArsOle.IsDocHorzScrollRequired (required)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If required <> 0 Then
   sbHorz.Visible = True
End If

 .
 .

' During scroll bar Change method

Private Sub sbHorz_Change()
    Dim Diff As Integer
    Dim rc As Integer
    Dim ScrollCode As Integer
    Dim NewPos As Variant

    NewPos = 0
    Diff = sbHorz.Value - HorzScrollOld
    If Diff = sbHorz.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEDOWN
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = -sbHorz.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEUP
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = sbHorz.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEDOWN
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = -sbHorz.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEUP
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    Else
        ScrollCode = ARS_OLE_SCROLL_THUMBPOSITION
        NewPos = sbHorz.Value
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = sbHorz.Value
    End If
End Sub
```

Logoff

メソッド:

short Logoff( )

説明: 現行サーバーからログオフします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: Logon

例: 以下の例では、ログオフを実行します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
 rc = pArsCtrl->Logoff( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.Logoff ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

Logon

メソッド:

short Logon(

char * **pServerName**,

char * **pUserId**,

char * **pPassword**)

パラメーター:

pServerName

サーバー名を含むヌル終了文字ストリングを指します。

pUserId

ユーザー ID を含むヌル終了文字ストリングを指します。

pPassword

パスワードを含むヌル終了文字ストリングを指します。

説明: pServerName、pUserId、および pPassword のそれぞれが NULL でないときに、空ストリングではないストリングを指す場合は、指定したデータを使用してログオンが実行されます。

pServerName と pUserId が有効なストリングを指し、pPassword がシングル・スペース文字を指す場合は、ユーザーのパスワードなしのままログオンが続行されます。

pServerName、pUserId、および pPassword のどれかが NULL であるかまたは空ストリングを指す場合は、情報が一部だけ提供されている通常の「OnDemand ログオン (OnDemand Logon)」ダイアログ・ボックスが表示されます。ユーザーは、この場合、欠落情報を入力してログオンを完了することができます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumServers, GetServerNames, Logoff, SetLogonReturnOnFailure

例: 以下の例では、ログオンできるすべてのサーバーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したサーバー、ユーザー ID、およびパスワードを取り出したあと、ログオンを実行します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pServerNames;
CComboBox * pServersList;
CEdit * pUserId;
CEdit * pPassword;
char server[ sizeof( ArsOleName ) ];
char user[ sizeof( ArsOleName ) ];
char password[ sizeof( ArsOleName ) ];
short rc, j, num_servers;
int index;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumServers( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_servers = var.iVal;

pServerNames = new ArsOleName[ max( num_servers, 1 ) ];
rc = pArsCtrl->GetServerNames( (IUnknown*)pServerNames, num_servers );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_servers; j++ )
  index = pServersList->AddString( pServerNames[j] );
pServersList->SetCurSel( 0 );
  .
  .
 // During OK button processing:

 pServersList->GetWindowText( server, sizeof(server) );
 pUserId->GetWindowText( user, sizeof(user) );
 pPassword->GetWindowText( password, sizeof(password) );

 rc = pArsCtrl->Logon( server, user, password );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_servers As Variant
Dim Temp As String
```

```
 .
 .

rc = ArsOle.GetNumServers (num_servers)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For count = 0 To num_servers -1
    rc = ArsOle.GetServerName(count, Temp)
    lbServers.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.Logon (lbServers.List(lbServers.ListItem), txtUser.Value, txtPasswd.Value)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

OnSysColorChange

注: これは、C/C++ で使用するためのメソッドです。

メソッド:

void OnSysColorChange( )

説明: OnDemand ダイアログ・ボックスに使用される色を、現行システム・カラーに更新します。文書データに使用される色は影響を受けません。

OLE コンテナー・アプリケーションのメイン・ウィンドウが WM_SYSCOLORCHANGE メッセージを受け取るたびに、このメソッドを呼び出す必要があります。

リターン値:

なし

例: 以下の例では、システム・カラーが変更されたときに OnDemand OLE Control に通知します。

C/C + + の例

```
CArsOle * pArsCtrl;
  .
  .
// During WM_SYSCOLORCHANGE message handling:

pArsCtrl->OnSysColorChange( );
  .
  .
```

OpenDoc

メソッド:

short OpenDoc(

long **Index**,

char * **pPath**,

long **ControlId**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。

pPath

文書データが入っているファイルの完全修飾パスを含むヌル終了文字ストリングを指します。このパラメーターが NULL の場合は、文書データは、OnDemand データベースから取り出されます。NULL でない場合は、文書データは、指定したファイルから取り出されますが、リソース・グループは必要に応じてデータベースから取り出されます。

ControlId

OnDemand OLE Control のコントロール ID を指定します。この値がゼロの場合は、このコントロールのコントロール ID が使用されます。コントロール ID の説明については、3 ページの『単一フォルダーの複数文書の表示』を参照してください。

説明: 指定した OnDemand OLE Control のアクティブ・フォルダーの文書リストに入っている、指定の索引に関連した文書を開いて、このコントロールのウィンドウ内に表示します。

別の OnDemand OLE Control を参照することによって、複数のウィンドウが単一の文書リストの文書を同時に表示できるようになります。このため、ログオン、フォルダー・オープン、フォルダー検索の操作を複数回行うことによるオーバーヘッドがなくなります。1 つの OnDemand OLE Control だけをアプリケーション内で使用する場合は、ControlId を常にゼロに設定する必要があります。

CancelOperation メソッドを使用すると、文書取り出しを取り消すことができます。取り消し機能をユーザーが使用できるようにするときには、OpenDoc メソッドをバックグラウンド・スレッドで呼び出して、ユーザー・インターフェース・スレッドで取り消しシグナルをモニターできるようにしたほうがよい場合があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderDisplayFields, GetFolderDisplayFieldNames, GetNumDocsInList, GetDocDisplayValues, CloseDoc, CancelOperation, WasOperationCancelled, ShowWaitCursorDuringCancelableOperation

例: 以下の例では、フォルダー文書リスト名および関連した値のリスト・ボックスを作成し、ユーザーが選択した文書を開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
ArsOleValue * pValues;
CListBox * pDocList;
char * pLine;
short rc, k, opr, num_fields;
```

```
long j, num_docs;
int size;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderDisplayFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderDisplayFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->GetNumDocsInList( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_docs = var.lVal;
  .
  .

  .
  .
pValues = new ArsOleValue[ max( num_fields, 1 ) ];

size = num_fields * ( sizeof(ArsOleName) + sizeof(ArsOleValue) + 5 );
pLine = new char[ size ];
for ( j = 0, pLine[0] = '¥0'; j < num_docs; j++ )
{
  rc = pArsCtrl->GetDocDisplayValues( j, pValues, num_fields );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;

  for ( k = 0; k < num_fields; k++ )
  {
    strcat( pLine, pNames[k] );
    strcat( pLine, " = " );
    strcat( pLine, pValues[k] );
    if ( k < num_fields - 1 )
      strcat( pLine, ", " );
  }
  pDocList->InsertString( -1, pLine );
}
pDocList->SetCurSel( 0 );
  .
  .
// During OK button processing:

rc = pArsCtrl->OpenDoc( (long)pDocList->GetCurSel( ) , NULL, 0 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String

 .
 .
```

```
rc = ArsOle.GetNumFolderDisplayFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderDisplayFieldName(count, Temp)
    Names(count) = Temp
Next count

rc = ArsOle.GetNumDocsInList(num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For j = 0 To num_docs -1
    For i = 0 To num_fields -1
        rc = ArsOle.GetDocDisplayValue(j, i, Temp)

        Line = Line + Names(i) + " = " + Temp
        If i < num_fields Then
            Line = Line + ", "
        End If
    Next i

    lbDocs.AddItem Line
Next j

 .
 .

'During OK button processing:

rc = ArsOle.OpenDoc (lbDocs.List(lbDocs.ListIndex), "", 0)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

OpenFolder

メソッド:

short OpenFolder(

char * **pFolderName**)

パラメーター:

pFolderName

フォルダー名を含むヌル終了文字ストリングを指します。

説明: pFolderName が NULL でないときに、空ストリングではないストリングを指す場合は、指定したフォルダーを開きます。

pFolderName が NULL の場合、または空ストリングを指す場合は、通常の「OnDemand フォルダーを開く (OnDemand Open Folder)」ダイアログ・ボックスを表示します。ユーザーは、この場合、フォルダーを選択してオープン操作を実行できます。

開いたフォルダーは、アクティブ・フォルダーになります。「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスは最初は隠れています。ShowFolder メソッドを使用すると、このダイアログ・ボックスを表示できます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolders, GetFolderNames, CloseFolder, CloseAllFolders

例: 以下の例では、現行サーバーで使用可能なすべてのフォルダーの名前を取り出し、これらの名前を ComboBox コントロールに入れて、選択したフォルダーを取り出したあと、そのフォルダーを開きます。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pFolderNames;
CComboBox * pFoldersList;
char folder[ sizeof( ArsOleName ) ];
short rc, j, num_folders;
int index;
VARIANT vari;
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolders( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_folders = var.iVal;

pFolderNames = new ArsOleName[ max( num_folders, 1 ) ];
rc = pArsCtrl->GetFolderNames( (IUnknown*)pFolderNames, num_folders );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_folders; j++ )
  index = pFoldersList->AddString( pFolderNames[j] );
pFoldersList->SetCurSel( 0 );
  .
  .
 // During OK button processing:

 pFoldersList->GetWindowText( folder, sizeof(folder) );

 rc = pArsCtrl->OpenFolder( folder );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_folders As Variant
Dim Temp As String

 .
 .

rc = ArsOle.GetNumFolders (num_folders)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
```

```
    End
End If

For count = 0 To num_folders -1
    rc = ArsOle.GetFolderName(count, Temp)
    lbFolders.AddItem Temp
Next count

 .
 .

' During OK button processing

rc = ArsOle.OpenFolder (lbFolders.List(lbFolders.ListItem))
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

PrintDoc

メソッド:

short PrintDoc(

long **Index**,

long **page**,

char * **pPrinterName**,

boolean **LocalPrinter**,

short **Copies**,

short **Orientation**,

float **TopMargin**,

float **BottomMargin**,

float **LeftMargin**,

float **RightMargin**,

boolean **MarginsInMillimeters**)

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。この値がゼロ未満であれば、開いている文書が印刷されます。ローカル・プリンターを指定した場合、開いている文書だけを印刷できます。

page

印刷するページ番号を指定します。このパラメーターがゼロ以下のときには、文書全体が印刷されます。サーバー・プリンターを指定した場合、このパラメーターは無視され、文書全体が印刷されます。

pPrinterName

ローカル・プリンターまたはサーバー・プリンターの名前を含むヌル終了文字ストリングを指します。ローカル・プリンターの場合、その名前は、プリンター選択リストに通常表示される名前 (例えば、LPT1: の IBM Laser

Printer 4019) と同じでなければなりません。 サーバー・プリンターの場合、その名前は、OnDemand データベース内で現行サーバーについて定義されている名前のうちの 1 つでなければなりません。

LocalPrinter

値が非ゼロであれば、pPrinterName に指定した名前はローカル・プリンターであることを意味します。ゼロであれば、サーバー・プリンターを指定することを意味します。ローカル・プリンターの場合は、開いている文書だけを印刷できます。サーバー・プリンターの場合は、用紙の向きとマージンの値は無視され、文書全体が印刷されます。

Copies

印刷部数を指定します。この値は、1 から 100 の値にする必要があります。

Orientation

ページの向きを指定します。これは、ARSOLEEX.H の中にある以下の方向値のいずれか 1 つにする必要があります。

```
ARS_OLE_ORIENTATION_PORTRAIT
ARS_OLE_ORIENTATION_LANDSCAPE
ARS_OLE_ORIENTATION_BEST_FIT
ARS_OLE_ORIENTATION_ASIS
```

サーバー・プリンターを指定した場合、このパラメーターは無視されます。

TopMargin

ページの上部マージンのスペース量を指定します。サーバー・プリンターを指定した場合、このパラメーターは無視されます。

BottomMargin

ページの下部マージンのスペース量を指定します。サーバー・プリンターを指定した場合、このパラメーターは無視されます。

LeftMargin

ページの左マージンのスペース量を指定します。サーバー・プリンターを指定した場合、このパラメーターは無視されます。

RightMargin

ページの右マージンのスペース量を指定します。サーバー・プリンターを指定した場合、このパラメーターは無視されます。

MarginsInMillimeters

値が非ゼロであれば、マージン値をミリメートル単位で指定することを意味します。ゼロであれば、インチ単位で指定することを意味します。サーバー・プリンターを指定した場合、このパラメーターは無視されます。

説明: 指定した文書内の 1 ページまたは複数のページを印刷します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: GetNumDocsInList, OpenDoc

例: 以下の例では、開いている文書の 5 ページ目をローカル・プリンターで印刷します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->PrintDoc( -1,
                         5,
                         "Acrobat PDFWriter on DISK:",
                         TRUE,
                         1,
                         ARS_OLE_ORIENTATION_BEST_FIT,
                         0.5,
                         0.5,
                         0.5,
                         0.5,
                         FALSE );

if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.PrintDoc (-1,                              _
                      5,                               _
                      "Acrobat PDFWriter on DISK:",    _
                      True,                            _
                      1,                               _
                      ARS_OLE_ORIENTATION_BEST_FIT,    _
                      0.5,                             _
                      0.5,                             _
                      0.5,                             _
                      0.5,                             _
                      False)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

RetrieveDoc

メソッド:

```
short  RetrieveDoc(
    long  Index,
    char  *  pDocPath,
    char  *  pResGrpPath,
    char  *  pCombinedPath  )
```

パラメーター:

Index

アクティブ・フォルダーの文書リストに入っている文書の 0 を基準とした索引を指定します。

pDocPath

文書データを入れるファイルの完全修飾パスを含むヌル終了文字ストリングを指します。このパラメーターが NULL の場合、データは何も書き込まれません。

pResGrpPath

リソース・グループ・データを入れるファイルの完全修飾パスを含むヌル終了文字ストリングを指します。このパラメーターが NULL の場合、データは何も書き込まれません。

pCombinedPath

リソース・グループ・データと文書データを結合して入れるファイルの完全修飾パスを含むヌル終了文字ストリングを指します。このパラメーターが NULL の場合、データは何も書き込まれません。

説明: 指定した文書のデータ、およびこの文書に関連付けられているリソース・グループ (もし、あれば) のデータを OnDemand データベースから取り出して、指定したファイルに書き込みます。指定した複数のファイルのうちのどれかがすでに存在していれば、データは、ファイルに付加されます。結合されるファイル内では、文書のデータは、リソース・グループのデータのあとに入ります。pDocPath、pResGrpPath、および pCombinedPath を任意に組み合わせて指定できます。

CancelOperation メソッドを使用すると、文書取り出しを取り消すことができます。取り消し機能をユーザーが使用できるようにするときには、RetrieveDoc メソッドをバックグラウンド・スレッドで呼び出して、ユーザー・インターフェース・スレッドで取り消しシグナルをモニターできるようにしたほうがよい場合があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumDocsInList, SetResourceCacheMode

例: 以下の例では、文書リスト内のすべてのファイルについてデータを取り出し、すべての文書データを、あるファイルにコピーし、リソース・グループ・データを別のファイルにコピーします。

C/C + + の例

```
CArsOle * pArsCtrl;
long num_docs, doc_num;
short rc;
  .
  .
rc = pArsCtrl->GetNumDocsInList( &num_docs );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( doc_num = 0; doc_num < num_docs; doc_num++ )
{
  rc = pArsCtrl->RetrieveDoc( doc_num,
                              "C:¥¥FILES¥¥DATA.DOC",
                              doc_num == 0 ? "C:¥¥FILES¥¥DATA.RG" : NULL,
                              NULL );
  if ( rc != ARS_OLE_RC_SUCCESS )
    ERROR;
}
  .
  .
```

Visual Basic の例

```
Dim rc, count As Integer
Dim num_docs As Variant

 .
 .

rc = ArsOle.GetNumDocsInList (num_docs)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

For count = 0 To num_docs - 1
    If count = 0 Then
        rc = ArsOle.RetrieveDoc (count, _
                                 "C:¥FILES¥DATA.DOC", _
                                 "C:¥FILES¥DATA.RG", _
                                 "")
        If rc <> ARS_OLE_RC_SUCCESS Then
            MsgBox "ERROR"
            End
        End If
    Else
        rc = ArsOle.RetrieveDoc (count, _
                                 "C:¥FILES¥DATA.DOC", _
                                 "", _
                                 "")
        If rc <> ARS_OLE_RC_SUCCESS Then
            MsgBox "ERROR"
            End
        End If
    End If
Next count

 .
 .
```

ScrollDocHorz

メソッド:

short ScrollDocHorz(

short **Type**,

VARIANT * **pPosition**)

パラメーター:

Type

必要なスクロールのタイプを識別するスクロール・バー・コードを指定します。 このコードは、ARSOLEEX.H の中にある以下のスクロール・タイプの 1 つにする必要があります。

```
ARS_OLE_SCROLL_LINELEFT
ARS_OLE_SCROLL_LINERIGHT
ARS_OLE_SCROLL_PAGELEFT
ARS_OLE_SCROLL_PAGERIGHT
ARS_OLE_SCROLL_LEFT
ARS_OLE_SCROLL_RIGHT
ARS_OLE_SCROLL_THUMBPOSITION
ARS_OLE_SCROLL_THUMBTRACK
ARS_OLE_SCROLL_ENDSCROLL
```

pPosition

スクロール位置を現在含んでいるか後に含む変数、または現在含んでいて後にも含む変数を指します。Type が ARS_OLE_SCROLL_THUMBPOSITION または ARS_OLE_SCROLL_THUMBTRACK である場合、この変数には、スクロール先の位置が含まれる必要があります。どのタイプの場合でも、この変数には、リターン時に現在のスクロール位置が含まれます。この変数は、タイプ VT_I2 に設定される必要があります。

説明: Type による指示に従って文書データを横方向にスクロールし、pPosition が指す変数に現在のスクロール位置を戻します。入力時とリターン時に、この位置の値は、水平スクロール範囲が ARS_OLE_SCROLL_RANGE に設定されていることを前提としています。異なる値を使用する場合は、呼び出しの前後で単位を変換する必要があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: IsDocHorzScrollRequired, ScrollDocVert

例: 以下の例では、水平スクロール・バーの範囲を初期設定し、文書のオープン後またはズーム値の変更後にスクロール・バーを表示するか隠して、WM_HSCROLL メッセージを処理します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScrollBar;
short rc, scroll_code;
VARIANT scroll_position, required;
  .
  .
// During initialization:
```

```
pHorzScrollBar->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
pHorzScrollBar->ShowScrollBar( FALSE );
  .
  .
// After a document is opened or changing the zoom value:

rc = pArsCtrl->IsDocHorzScrollRequired( &required );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->ShowScrollBar( required.iVal );
  .
  .
// While processing a WM_HSCROLL message:

scroll_code = (short)LOWORD(wParam);

switch ( scroll_code )
{
  case SB_LINELEFT:
    scroll_code = ARS_OLE_SCROLL_LINELEFT;
    break;
  case SB_LINERIGHT:
    scroll_code = ARS_OLE_SCROLL_LINERIGHT;
    break;
  case SB_PAGELEFT:
    scroll_code = ARS_OLE_SCROLL_PAGELEFT;
    break;
  case SB_PAGERIGHT:
    scroll_code = ARS_OLE_SCROLL_PAGERIGHT;
    break;
  case SB_LEFT:
    scroll_code = ARS_OLE_SCROLL_LEFT;
    break;
  case SB_RIGHT:
    scroll_code = ARS_OLE_SCROLL_RIGHT;
    break;
  case SB_THUMBPOSITION:
    scroll_code = ARS_OLE_SCROLL_THUMBPOSITION;
    break;
  case SB_THUMBTRACK:
    scroll_code = ARS_OLE_SCROLL_THUMBTRACK;
    break;
  default:
    scroll_code = ARS_OLE_SCROLL_ENDSCROLL;
}

if ( scroll_code == (short)ARS_OLE_SCROLL_THUMBPOSITION ||
     scroll_code == (short)ARS_OLE_SCROLL_THUMBTRACK )
{
  scroll_position.vt = VT_I2;
  scroll_position.iVal = (short)HIWORD(wParam);
}

rc = pArsCtrl->ScrollDocHorz( scroll_code, &scroll_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->SetScrollPos( (int)scroll_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim scroll_pos, required As Variant
 .
```

```
 .
' During initialization

sbHorz.Min = 0
sbHorz.Max = ARS_OLE_SCROLL_RANGE
sbHorz.Visible = False

' After a document is opened or changing the zoom value

rc = ArsOle.IsDocHorzScrollRequired (required)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

If required <> 0 Then
   sbHorz.Visible = True
End If

 .
 .

' During scroll bar Change method

Private Sub sbHorz_Change()
    Dim Diff As Integer
    Dim rc As Integer
    Dim ScrollCode As Integer
    Dim NewPos As Variant

    NewPos = 0
    Diff = sbHorz.Value - HorzScrollOld
    If Diff = sbHorz.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEDOWN
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = -sbHorz.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEUP
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = sbHorz.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEDOWN
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    ElseIf Diff = -sbHorz.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEUP
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = NewPos
        sbHorz.Value = NewPos
    Else
        ScrollCode = ARS_OLE_SCROLL_THUMBPOSITION
        NewPos = sbHorz.Value
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = sbHorz.Value
    End If
End Sub
```

ScrollDocVert

メソッド:

short ScrollDocVert(

short **Type**,

VARIANT * **pPosition**)

パラメーター:

Type

必要なスクロールのタイプを識別するスクロール・バー・コードを指定します。 このコードは、ARSOLEEX.H の中にある以下のスクロール・タイプの 1 つにする必要があります。

```
ARS_OLE_SCROLL_LINEUP
ARS_OLE_SCROLL_LINEDOWN
ARS_OLE_SCROLL_PAGEUP
ARS_OLE_SCROLL_PAGEDOWN
ARS_OLE_SCROLL_TOP
ARS_OLE_SCROLL_BOTTOM
ARS_OLE_SCROLL_THUMBPOSITION
ARS_OLE_SCROLL_THUMBTRACK
ARS_OLE_SCROLL_ENDSCROLL
```

pPosition

スクロール位置を現在含んでいるか後に含む変数、または現在含んでいて後にも含む変数を指します。Type が ARS_OLE_SCROLL_THUMBPOSITION または ARS_OLE_SCROLL_THUMBTRACK である場合、この変数には、スクロール先の位置が含まれる必要があります。どのタイプの場合でも、この変数には、リターン時に現在のスクロール位置が含まれます。この変数は、タイプ VT_I2 に設定される必要があります。

説明: Type による指示に従って文書データを縦方向にスクロールし、pPosition が指す変数に現在のスクロール位置を戻します。入力時とリターン時に、この位置の値は、垂直スクロール範囲が ARS_OLE_SCROLL_RANGE に設定されていることを前提としています。異なる値を使用する場合は、呼び出しの前後で単位を変換する必要があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: ScrollDocHorz

例: 以下の例では、垂直スクロール・バー範囲を初期設定し、WM_VSCROLL メッセージを処理します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pVertScrollBar;
short rc, scroll_code;
VARIANT scroll_position;
  .
  .
// During initialization:

pVertScrollBar->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
```

```
pVertScrollBar->ShowScrollBar( TRUE );

// While processing a WM_VSCROLL message:

scroll_code = (short)LOWORD(wParam);

switch ( scroll_code )
{
  case SB_LINEUP:
    scroll_code = ARS_OLE_SCROLL_LINEUP;
    break;
  case SB_LINEDOWN:
    scroll_code = ARS_OLE_SCROLL_LINEDOWN;
    break;
  case SB_PAGEUP:
    scroll_code = ARS_OLE_SCROLL_PAGEUP;
    break;
  case SB_PAGEDOWN:
    scroll_code = ARS_OLE_SCROLL_PAGEDOWN;
    break;
  case SB_TOP:
    scroll_code = ARS_OLE_SCROLL_TOP;
    break;
  case SB_BOTTOM:
    scroll_code = ARS_OLE_SCROLL_BOTTOM;
    break;
  case SB_THUMBPOSITION:
    scroll_code = ARS_OLE_SCROLL_THUMBPOSITION;
    break;
  case SB_THUMBTRACK:
    scroll_code = ARS_OLE_SCROLL_THUMBTRACK;
    break;
  default:
    scroll_code = ARS_OLE_SCROLL_ENDSCROLL;
}
  .
  .

  .
  .
if ( scroll_code == (short)ARS_OLE_SCROLL_THUMBPOSITION ||
     scroll_code == (short)ARS_OLE_SCROLL_THUMBTRACK )
{
  scroll_position.vt = VT_I2;
  scroll_position.iVal = (short)HIWORD(wParam);
}

rc = pArsCtrl->ScrollDocVert( scroll_code, &scroll_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pVertScrollBar->SetScrollPos( (int)scroll_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim scroll_pos, required As Variant

 .
 .

' During initialization

sbVert.Min = 0
sbVert.Max = ARS_OLE_SCROLL_RANGE
sbVert.Visible = True
```

```
.
.

' During scroll bar Change method

Private Sub sbVert_Change()
    Dim Diff As Integer
    Dim rc As Integer
    Dim ScrollCode As Integer
    Dim NewPos As Variant

    NewPos = 0
    Diff = sbVert.Value - VertScrollOld
    If Diff = sbVert.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEDOWN
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        sbVert.Value = NewPos
    ElseIf Diff = -sbVert.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEUP
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        sbVert.Value = NewPos
    ElseIf Diff = sbVert.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEDOWN
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        sbVert.Value = NewPos
    ElseIf Diff = -sbVert.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEUP
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        sbVert.Value = NewPos
    Else
        ScrollCode = ARS_OLE_SCROLL_THUMBPOSITION
        NewPos = sbVert.Value
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = sbVert.Value
    End If
End Sub
```

SearchFolder

メソッド:

short SearchFolder(

boolean **Append**)

パラメーター:

Append

値が非ゼロの場合は、既存の文書リストに検索結果を付加するよう指示します。ゼロの場合は、既存の文書リストを検索結果で置換するよう指示します。

説明: 現行フィールド値によってアクティブ・フォルダーを検索します。そのフィールド値は、デフォルト、ユーザー、または SetFolderSearchFieldData メソッドにより設定された値です。

CancelOperation メソッドを使用すると、検索操作を取り消すことができます。取り消し機能をユーザーが使用できるようにするときには、

SearchFolder メソッドをバックグラウンド・スレッドで呼び出して、ユーザー・インターフェース・スレッドで取り消しシグナルをモニターできるようにしたほうがよい場合があります。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, GetNumFolderSearchFields, GetFolderSearchFieldNames, SetFolderSearchFieldData, CancelOperation, WasOperationCancelled, ShowWaitCursorDuringCancelableOperation

例: 以下の例では、アクティブ・フォルダーの検索フィールドの名前を取り出し、そのフィールドに値を設定する機会をユーザーに提供し、フォルダーの検索を開始します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
CListBox * pFieldList, * pOprList;
CEdit * pValue1, * pValue2;
char name[ sizeof( ArsOleName ) ];
char value1[ sizeof( ArsOleValue ) ];
char value2[ sizeof( ArsOleValue ) ];
short rc, j, opr, num_fields;
VARIANT vari;
  .
  .
struct _OprMap
{
  short  code;
  char * pText;
} OprMap

static OprMap Oprs[] =
  { { ARS_OLE_OPR_EQUAL,                "Equal" },
    { ARS_OLE_OPR_NOT_EQUAL,            "Not Equal" },
        .
        .
    { ARS_OLE_OPR_LIKE,                 "Like" },
    { ARS_OLE_OPR_NOT_LIKE,             "Not Like" } };

#define NUM_OPRS  ( sizeof(Oprs) / sizeof(OprMap) )
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderSearchFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderSearchFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_fields; j++ )
  pFieldList->InsertString( -1, pNames[j] );
pFieldList->SetCurSel( 0 );

for ( j = 0; j < NUM_OPRS; j++ )
{
```

```
  pOprList->InsertString( -1, Oprs[j].pText );
  pOprList->SetItemData( j, (DWORD)Oprs[j].code );
}
pOprList->SetCurSel( 0 );
  .
  .

// During SET FIELD button processing:

pFieldList->GetText( pFieldList->GetCurSel( ), name );
opr = (short)pOprList->GetItemData( pOprList->GetCurSel( ) );
pValue1->GetWindowText( value1, sizeof(value1) );
pValue2->GetWindowText( value2, sizeof(value2) );

rc = pArsCtrl->SetFolderSearchFieldData( name, opr, value1, value2 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .

// During OK button processing:

rc = pArsCtrl->SearchFolder( FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
Dim Oprs As Variant

 .
 .

Oprs = Array ("Equal", "Not Equal", ..., "Like", "Not Like")

rc = ArsOle.GetNumFolderSearchFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields -1)

For count = 0 To num_fields -1
    rc = ArsOle.GetFolderSearchFieldName(count, Temp)
    Names(count) = Temp
Next count

for count = 0 To num_fields -1
    lbFieldList.AddItem Names(count)
Next count

for count = 0 To UBound(Oprs) -1
    lbOprList.AddItem (Oprs(count))
Next count
' During SET FIELD button processing
rc = ArsOle.SetFolderSearchFieldData (lbFieldList.List(lbFieldList.ListIndex),
                                      lbOprList.ListIndex,
                                      txtValue1.Value,
                                      txtValue2.Value)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
```

```
    End
End If
 .
 .
 .
'During OK button processing:

rc = ArsOle.SearchFolder (False)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

SetDefaultFolderSearchFields

メソッド:

short SetDefaultFolderSearchFields( )

説明: アクティブ・フォルダーの検索フィールドをデフォルト値に設定します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, SearchFolder

例: 以下の例では、アクティブ・フォルダーの検索フィールドをデフォルト値に設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
 rc = pArsCtrl->SetDefaultFolderSearchFields( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetDefaultFolderSearchFields ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetDocBackgroundColor

メソッド:

short SetDocBackgroundColor(

short **Color**)

パラメーター:

Color

文書の新しい背景色を指定します。この色は、ARSOLEEX.H の中にある以下の色の値の 1 つにする必要があります。

```
ARS_OLE_COLOR_WHITE
ARS_OLE_COLOR_BLACK
ARS_OLE_COLOR_RED
ARS_OLE_COLOR_BLUE
ARS_OLE_COLOR_GREEN
ARS_OLE_COLOR_YELLOW
ARS_OLE_COLOR_GREY
```

説明: 文書を新しい背景色で表示します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: SetDocBackgroundColor

例: 以下の例では、文書の背景色をグレーに設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetDocBackgroundColor( ARS_OLE_COLOR_GREY );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetDocBackgroundColor (ARS_OLE_COLOR_GREY)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetDocCurrentPage

メソッド:

short SetDocCurrentPage(

long **page**)

パラメーター:

page

開いている文書の現行ページ番号をページ番号とするよう指定します。

説明: 開いている文書の現行ページ番号を、指定したページに設定し、コントロール・ウィンドウをそのページのデータで再描画します。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: GetDocCurrentPage, GetDocNumPages

例: 以下の例では、開いている文書の現行ページ番号を設定し、現行スクロール位置を更新します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScollBar, * pVertScrollBar;
short rc;
VARIANT horz_position, vert_position;
  .
  .
 rc = pArsCtrl->SetDocCurrentPage( 46 );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .

rc = pArsCtrl->GetDocScrollPositions( &horz_position, &vert_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->SetScrollPos( (int)horz_position.iVal );
pVertScrollBar->SetScrollPos( (int)vert_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim horz_pos, vert_post As Variant

 .
 .

rc = ArsOle.SetDocCurrentPage( 46 )
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

rc = ArsOle.GetDocScrollPositions( horz_pos, vert_pos )
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

sbHorz.Value = horz_pos
sbVert.Value = vert_pos

 .
 .
```

SetDocImageColor

メソッド:

short SetDocImageColor(

short **Color**)

パラメーター:

Color

文書の新しいイメージ・カラーを指定します。この色は、ARSOLEEX.H の中にある以下の色の値の 1 つにする必要があります。

```
ARS_OLE_COLOR_BLACK
ARS_OLE_COLOR_RED
ARS_OLE_COLOR_BLUE
ARS_OLE_COLOR_GREEN
ARS_OLE_COLOR_YELLOW
ARS_OLE_COLOR_GREY
ARS_OLE_COLOR_MAGENTA
ARS_OLE_COLOR_CYAN
```

説明: 文書を新しいイメージ・カラーで表示します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocImageColor

例: 以下の例では、文書のイメージ・カラーを赤に設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetDocImageColor( ARS_OLE_COLOR_RED );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetDocImageColor (ARS_OLE_COLOR_RED)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetDocImageIntensity

メソッド:

short SetDocImageIntensity(

short **Intensity**)

パラメーター:

Intensity

文書の新しいイメージ輝度を指定します。この輝度は、ARSOLEEX.H の中にある以下の輝度値の 1 つにする必要があります。

```
ARS_OLE_INTENSITY_NORMAL
ARS_OLE_INTENSITY_LIGHT
ARS_OLE_INTENSITY_NONE
```

説明: 文書を新しいイメージ輝度で表示します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocImageIntensity

例: 以下の例では、文書のイメージ輝度を light に設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetDocImageIntensity( ARS_OLE_INTENSITY_LIGHT );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetDocImageIntensity (ARS_OLE_INTENSITY_LIGHT)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetDocRotation

メソッド:

short SetDocRotation(

short **Rotation**)

パラメーター:

Rotation

文書の新しい回転値を指定します。この回転値は、ARSOLEEX.H の中にある以下の回転値の 1 つにする必要があります。

```
ARS_OLE_ROTATION_0
ARS_OLE_ROTATION_90
ARS_OLE_ROTATION_180
ARS_OLE_ROTATION_270
```

説明: 文書を新しい回転値で表示します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocRotation

例: 以下の例では、文書を 90 度回転させます。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetDocRotation( ARS_OLE_ROTATION_90 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetDocRotation (ARS_OLE_ROTATION_90)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetDocZoom

メソッド:

short SetDocZoom(

short **ZoomPercent**,

VARIANT * **pHorzPosition**,

VARIANT * **pVertPosition**)

パラメーター:

ZoomPercent

新たに設定するズームのパーセント値を指定します。

pHorzPosition

新規水平スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

pVertPosition

新規垂直スクロール位置を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 文書を新規ズーム・パーセント値で表示し、指定した変数に新規スクロール位置を戻します。そのスクロール位置は、スクロール範囲が ARS_OLE_SCROLL_RANGE に設定されていることを前提としています。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetDocZoom

例: 以下の例では、新規ズーム値を設定し、スクロール・バーの位置を変更します。

C/C + + の例

```
CArsOle * pArsCtrl;
CScrollBar * pHorzScrollBar, * pVertScrollBar;
short rc;
VARIANT horz_position, vert_position;
  .
  .
// During initialization:

pHorzScrollBar->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
pVertScrollBar->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
  .
  .
// When required to double the zoom factor:

rc = pArsCtrl->SetDocZoom( 200, &horz_position, &vert_position );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

pHorzScrollBar->SetScrollPos( (int)horz_position.iVal );
pVertScrollBar->SetScrollPos( (int)vert_position.iVal );
  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim horz_pos, vert_pos As Variant

 .
 .

' During initialization

sbHorz.Min = 0
sbHorz.Max = ARS_OLE_SCROLL_RANGE
sbVert.Min = 0
sbVert.Max = ARS_OLE_SCROLL_RANGE

 .
 .

' When required to double the zoom factor

rc = ArsOle.SetDocZoom (200, horz_pos, vert_pos)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
```

```
   End
End If

sbHorz.Value = horz_pos
sbVert.Value = vert_pos
```

SetFolderCloseMemoryRelease

メソッド:

short SetFolderCloseMemoryRelease(

boolean **Release**)

パラメーター:

Release

値が非ゼロの場合は、フォルダーがクローズしたとき、このフォルダーに関連したすべてのメモリーを解放するよう指示します。ゼロの場合は、メモリーを保持するよう指示します。

説明: フォルダーがクローズしたときにメモリーを解放するかどうかを決定します。デフォルトの設定値は偽です。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CloseFolder

例: 以下の例では、フォルダーがクローズしたときにメモリーを解放します。

C/C + + の例

```
CArsOle * pArsCtrl;
  .
  .
  .
pArsCtrl->SetFolderCloseMemoryRelease( TRUE );
  .
  .
  .
```

Visual Basic の例

```
  .
  .
  .
ArsOle.SetFolderCloseMemoryRelease (True)
  .
  .
  .
```

SetFolderSearchFieldData

メソッド:

short SetFolderSearchFieldData(

char * **pFieldName**,

short **operator**,

char * **pValue1**,

char * **pValue2**)

パラメーター:

pFieldName

アクティブ・フォルダーの検索フィールドの名前を含むヌル終了文字ストリングを指します。

operator

使用する検索演算子を指定します。この演算子は、ARSOLEEX.H の中にある以下の演算子の値の 1 つにする必要があります。

```
ARS_OLE_OPR_EQUAL
ARS_OLE_OPR_NOT_EQUAL
ARS_OLE_OPR_LESS_THAN
ARS_OLE_OPR_LESS_THAN_OR_EQUAL
ARS_OLE_OPR_GREATER_THAN
ARS_OLE_OPR_GREATER_THAN_OR_EQUAL
ARS_OLE_OPR_BETWEEN
ARS_OLE_OPR_NOT_BETWEEN
ARS_OLE_OPR_IN
ARS_OLE_OPR_NOT_IN
ARS_OLE_OPR_LIKE
ARS_OLE_OPR_NOT_LIKE
```

pValue1

フィールドに設定する最初の値または唯一の値を含むヌル終了文字ストリングを指します。

pValue2

フィールドに 2 番目に設定する値を含むヌル終了文字ストリングを指します。演算子が ARS_OLE_OPR_BETWEEN または ARS_OLE_OPR_NOT_BETWEEN でない限り、このパラメーターは無視されます。

説明: アクティブ・フォルダーについて指定したフィールドに検索演算子と値 (1 つまたは複数の) を設定します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderSearchFields, GetFolderSearchFieldNames, SearchFolder

例: 以下の例では、アクティブ・フォルダーの検索フィールドの名前を取り出し、そのフィールドに値を設定する機会をユーザーに提供し、フォルダーの検索を開始します。

C/C + + の例

```
CArsOle * pArsCtrl;
ArsOleName * pNames;
CListBox * pFieldList, * pOprList;
CEdit * pValue1, * pValue2;
char name[ sizeof( ArsOleName ) ];
char value1[ sizeof( ArsOleValue ) ];
char value2[ sizeof( ArsOleValue ) ];
short rc, j, opr, num_fields;
VARIANT vari;
```

```
  .
  .
struct _OprMap
{
  short  code;
  char * pText;
} OprMap

static OprMap Oprs[] =
  { { ARS_OLE_OPR_EQUAL,                "Equal" },
    { ARS_OLE_OPR_NOT_EQUAL,            "Not Equal" },
        .
        .
    { ARS_OLE_OPR_LIKE,                 "Like" },
    { ARS_OLE_OPR_NOT_LIKE,             "Not Like" } };

#define NUM_OPRS  ( sizeof(Oprs) / sizeof(OprMap) )
  .
  .
// During dialog initialization:

rc = pArsCtrl->GetNumFolderSearchFields( &vari );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
num_fields = var.iVal;

pNames = new ArsOleName[ max( num_fields, 1 ) ];
rc = pArsCtrl->GetFolderSearchFieldNames( (IUnknown*)pNames, num_fields );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

for ( j = 0; j < num_fields; j++ )
  pFieldList->InsertString( -1, pNames[j] );
pFieldList->SetCurSel( 0 );

for ( j = 0; j < NUM_OPRS; j++ )
{
  pOprList->InsertString( -1, Oprs[j].pText );
  pOprList->SetItemData( j, (DWORD)Oprs[j].code );
}
pOprList->SetCurSel( 0 );
  .
  .
// During SET FIELD button processing:

pFieldList->GetText( pFieldList->GetCurSel( ), name );
opr = (short)pOprList->GetItemData( pOprList->GetCurSel( ) );
pValue1->GetWindowText( value1, sizeof(value1) );
pValue2->GetWindowText( value2, sizeof(value2) );

rc = pArsCtrl->SetFolderSearchFieldData( name, opr, value1, value2 );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
// During OK button processing:

rc = pArsCtrl->SearchFolder( FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc, count, i, j As Integer
Dim num_fields, num_docs As Variant
Dim Names() As String
Dim Line As String
Dim Temp As String
Dim Oprs As Variant

 .
 .

Oprs = Array ("Equal", "Not Equal", ..., "Like", "Not Like")

rc = ArsOle.GetNumFolderSearchFields(num_fields)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

ReDim Names(num_fields)

For count = 1 To num_fields
    rc = ArsOle.GetFolderSearchFieldName(count, Temp)
    Names(count) = Temp
Next count

for count = 1 To num_fields
    lbFieldList.AddItem Names(count)
Next count

for count = 1 To UBound(Oprs)
    lbOprList.AddItem (Oprs(count))
Next count
 .
 .
 .
' During SET FIELD button processing
rc = ArsOle.SetFolderSearchFieldData (lbFieldList.List(lbFieldList.ListIndex),
                                      lbOprList.ListIndex,
                                      txtValue1.Value,
                                      txtValue2.Value)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If

'During OK button processing:

rc = ArsOle.SearchFolder (False)
If rc <> ARS_OLE_RC_SUCCESS Then
    MsgBox "ERROR"
    End
End If
```

SetLogonReturnOnFailure

メソッド:

short SetLogonReturnOnFailure(

boolean **Return**)

パラメーター:

Return

値が非ゼロの場合は、ログオンに失敗したときに制御を戻すよう指示します。ゼロの場合は、ログオンに失敗したときに「OnDemand ログオン (OnDemand Logon)」ダイアログ・ボックスを表示するよう指示します。

説明: ログオンに失敗したときに取られるアクションを決定します。デフォルトの設定値は偽です。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: Logon

例: 以下の例では、ログオンに失敗したときに取られるアクションを設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
  .
  .
  .
pArsCtrl->SetLogonReturnOnFailure( TRUE );
  .
  .
  .
```

Visual Basic の例

```
  .
  .
  .
ArsOle.SetLogonReturnOnFailure (True)
  .
  .
  .
```

SetResourceCacheMode

メソッド:

short **SetResourceCacheMode**(

short **Mode**)

パラメーター:

Mode

新しいリソース・キャッシュ・モードを指定します。このモードは、ARSOLEEX.H の中にある以下のモード値の 1 つにする必要があります。

```
ARS_OLE_RES_MODE_RETAIN
ARS_OLE_RES_MODE_ERASE_AFTER_RETRIEVE
```

ARS_OLE_RES_MODE_RETAIN は、RetrieveDoc メソッドの実行中に取り戻されたリソース・グループを、以降の RetrieveDoc または OpenDoc メソッドが使用できるように、保存しておくことを指定します。これらのリソース・グループが入っているファイルは、コントロールのすべてのインスタンスが終了するまで、消去されません。これは、デフォルト・モードです。

ARS_OLE_RES_MODE_ERASE_AFTER_RETRIEVE は、RetrieveDoc メソッドの実行中に取り戻されたリソース・グループのファイルを、呼び出し元に戻る前に消去することを指定します。以降の RetrieveDoc または OpenDoc メソッドが同じリソース・グループにアクセスする必要のある場合は、別に検索が必要となります。

説明: 要求されたモードが設定されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: RetrieveDoc

例: 以下の例では、リソース・キャッシュ・モードを保存に設定します。

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;
.
.
rc = pArsCtrl->SetResourceCacheMode( ARS_OLE_RES_MODE_RETAIN );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
.
.
```

Visual Basic の例

```
Dim rc As Integer
.
.
rc = ArsOle.SetResourceCacheMode (ARS_OLE_RES_MODE_RETAIN)
if rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If
.
.
```

SetRightButtonMenu

メソッド:

short SetRightButtonMenu(

char * **pMenuData**,

VARIANT * **pErrorPosition**)

パラメーター:

pMenuData

メニューの作成に使用されるデータを指定します。このパラメーターがヌルの場合、既存のメニューは削除されます。

pErrorPosition

メニューを作成できない場合にエラー位置を受け取る変数を指します。その場合には、この変数は、タイプ VT_I4 に設定されます。

説明: ユーザーが右マウス・ボタンを文書ウィンドウ内でクリックすると表示されるメニューが、OnDemand によって作成されます。このメニューで既存のメニューが置換されます。このメニューの形式は、pMenuData パラメーターで指定したデータによって決まります。コマンド、区切り文字、およびサブメニューを含む複雑なメニューを作成できます。

メニュー・データは、改行文字 (X'0A') で区切られた一連のメニュー項目から成ります。最大で ARS_OLE_MAX_MENU_ITEMS の個数の項目を含める

ことができ、各項目には、ARS_OLE_MAX_MENU_ITEM_LEN までの数の文字を含めることができます。それらの項目は、メニュー内に項目が表示される順序で記述する必要があります。

各項目は、キーワードと値によって記述します。キーワードとこれに関連付けられる値とは等号で区切ることが必要です。キーワードと値の各対は、1 個以上のスペースで区切る必要があります。認識されるキーワードは、以下のとおりです。

level 項目のネスト・レベルを指示する数。最初の項目は、レベル 0 でなければなりません。それ以降、ネスト・レベルが上がるごとに 1 を加算する必要があります。ネスト・レベルを増加できるのは、ポップアップ・サブメニューを指定するときだけです。

このキーワードは必須指定です。

id 項目に関連付けられるユーザー・コマンド番号。この id は、ユーザーがメニュー項目を選択すると UserCommand イベントについて報告される番号です。

2 つの特別な id が以下のように定義されています。

- id に ARS_OLE_MENU_ID_SEPARATOR を指定すると、区切り文字の項目が作成されます。
- id に ARS_OLE_MENU_ID_POPUP を指定すると、ポップアップ・サブメニューが作成されます。この場合、次の項目のレベルは、この項目に指定したレベルよりも 1 つ大きいレベルにする必要があります。

その他の id は、最小値が ARS_OLE_MIN_MENU_ID、最大値が ARS_OLE_MAX_MENU_ID にする必要があります。

このキーワードは必須指定です。

enabled

キーワード enabled の値は、1 または 0 にすることが必要です。1 の場合は、項目は使用可能になり、0 の場合は使用不可になります。

このキーワードはオプションであり、区切り文字の項目については無視されます。デフォルトでは、項目が使用可能になります。

checked

キーワード checked の値は、1 または 0 にすることが必要です。1 の場合は項目が検査され、0 の場合は検査されません。

このキーワードはオプションであり、区切り文字の項目については無視されます。デフォルトでは、項目は検査されません。

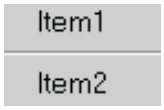
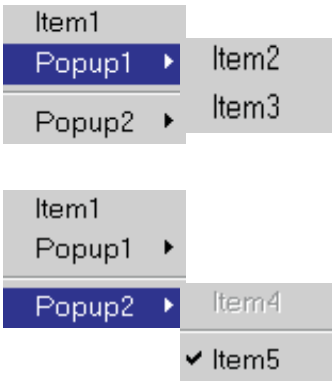
text キーワード text は、項目のテキストを指定します。このテキストには組み込みブランクを含めることができ、単一引用符でこのテキストを囲む必要があります。単一引用符がテキストの一部である場合は、連続した 2 個の単一引用符を指定する必要があります。

このテキストには、Windows の通常の特殊文字、例えば、次の文字に下線を引くよう指示する & (アンパーサンド) などを含めることができます。

このキーワードはオプションであり、区切り文字の項目については無視されます。

メニュー・データが無効な場合、エラー位置が pErrorPosition パラメーターを介して戻されます。この位置は 0 を基準としたものであり、メニュー・データの最初の文字に対して相対的な位置です。通常、この位置によって、エラーがある項目の最初の文字が識別されます。

キーワードの使用例を以下に示します。

データ	メニュー
level=0 id=368 text='Copy Text'	
level=0 id=44 text='Item1'¥n level=0 id=1¥n level=0 id=45 text='Item2'	
level=0 id=32457 text='Item1'¥n level=0 id=0 text='Popup1'¥n level=1 id=32458 text='Item2'¥n level=1 id=32459 text='Item3'¥n level=0 id=1¥n level=0 id=0 text='Popup2'¥n level=1 id=32460 enabled=0 text='Item4'¥n level=1 id=1¥n level=1 id=32461 checked=1 text='Item5'	

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: UserCommand イベント

例: 以下の例では、コマンド Copy Text を含む右ボタン・メニューを作成する方法を示します。

C/C ++ の例

```
CArsOle * pArsCtrl;
short rc;
VARIANT var;
  .
  .
  .

rc = pArsCtrl->SetRightButtonMenu( "level=0 id=368 text='Copy Text'", var );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim var As Variant

 .
 .

rc = ArsOle.SetRightButtonMenu ("level=0 id=368 text='Copy Text'", var)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If
```

SetSelectionMode

メソッド:

short SetSelectionMode(

short **Mode**)

パラメーター:

Mode

新しい選択モードを指定します。このモードは、ARSOLEEX.H の中にある以下の選択モード値の 1 つにする必要があります。

```
ARS_OLE_SELECTION_MODE_NONE
ARS_OLE_SELECTION_MODE_AREA
ARS_OLE_SELECTION_MODE_TEXT
```

ARS_OLE_SELECTION_MODE_NONE は、ユーザーが文書の一部をマウスで選択できないことを指定します。これは、文書のオープン時のデフォルト・モードです。このモードを設定すると、既存の選択モードは除去されます。

ARS_OLE_SELECTION_MODE_AREA は、OnDemand クライアント GUI で「オプション」->「選択モード」->「領域」を選択した場合と同じ方式でユーザーの選択が実行されることを指定します。

ARS_OLE_SELECTION_MODE_TEXT は、OnDemand クライアント GUI で「オプション」->「選択モード」->「テキスト」を選択した場合と同じ方式でユーザーの選択が実行されることを指定します。

説明: ユーザーによるマウス選択が、これ以後、指定したモードで実行されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CopyBitmap, CopyText

例: 以下の例では、選択モードをエリア選択に設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetSelectionMode( ARS_OLE_SELECTION_MODE_AREA );
```

```
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
   .
   .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetSelectionMode (ARS_OLE_SELECTION_MODE_AREA)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetServerPrinterData

メソッド:

short SetServerPrinterData(

short **Index**,

char * **pData**)

パラメーター:

Index

サーバー・プリンターのデータ索引を指定します。これは、ARSLOEEX.H の中にある以下の値の 1 つにする必要があります。

```
ARS_OLE_SERVER_PRINTER_DATA_FAX_RECEIVER_NAME
ARS_OLE_SERVER_PRINTER_DATA_FAX_RECEIVER_COMPANY
ARS_OLE_SERVER_PRINTER_DATA_FAX_RECEIVER_FAX_NUMBER
ARS_OLE_SERVER_PRINTER_DATA_FAX_SENDER_NAME
ARS_OLE_SERVER_PRINTER_DATA_FAX_SENDER_COMPANY
ARS_OLE_SERVER_PRINTER_DATA_FAX_SENDER_TEL_NUMBER
ARS_OLE_SERVER_PRINTER_DATA_FAX_SENDER_FAX_NUMBER
ARS_OLE_SERVER_PRINTER_DATA_FAX_SENDER_COVER_PAGE
ARS_OLE_SERVER_PRINTER_DATA_FAX_SUBJECT
ARS_OLE_SERVER_PRINTER_DATA_FAX_NOTES
ARS_OLE_SERVER_PRINTER_DATA_INFO_FROM
ARS_OLE_SERVER_PRINTER_DATA_INFO_TO
```

pData

索引に関連付けられるデータを含むヌル終了文字ストリングを指します。このパラメーターはヌルでも構いません。

説明: 各索引に関連付けられるデータは、デフォルトでは空ストリングになります。このメソッドを使用するデータ・セットは、変更されるまでは保持されます。pData がヌルの場合、データは、空ストリングにリセットされます。PrintDoc メソッドを使用し、サーバー・プリンターを指定した場合、プリンターのタイプ (FAX など) が判別され、該当するデータが文書と一緒に送信されます。

リターン値:

戻りコードの説明については、 4 ページの『戻りコード』を参照してください。

参照: PrintDoc

例 以下の例では、サーバー・プリンターに FAX 受信者名を設定します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
  .

rc = pArsCtrl->SetServerPrinterData(
                  ARS_OLE_SERVER_PRINTER_DATA_FAX_RECEIVER_NAME,
                  "John Doe" );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

  .
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetServerPrinterData (
               ARS_OLE_SERVER_PRINTER_DATA_FAX_RECEIVER_NAME,
               "John Doe");
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

SetUserMessageMode

メソッド:

short SetUserMessageMode(

short **Mode**)

パラメーター:

Mode

新しいユーザー・メッセージ・モードを指定します。このモードは、ARSOLEEX.H の中にある以下のメッセージ・モード値の 1 つにする必要があります。

```
ARS_OLE_USER_MSG_MODE_SHOW
ARS_OLE_USER_MSG_MODE_SUPPRESS
```

ARS_OLE_USER_MSG_MODE_SHOW は、OLE Control 操作で生成されるすべての OnDemand 例外メッセージをユーザーに表示するよう指示しま

す。ARS_OLE_USER_MSG_MODE_SUPPRESS は、OLE Control 操作で生成されるすべての OnDemand 例外メッセージを抑止するよう指示します。

説明: OLE Control 操作で生成される以降の OnDemand 例外メッセージを、必要に応じて表示するかまたは抑止します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

例: 以下の例では、OLE Control 操作で生成される OnDemand 例外メッセージを抑止します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
rc = pArsCtrl->SetUserMessageMode( ARS_OLE_USER_MSG_MODE_SUPPRESS );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.SetUserMessageMode (ARS_OLE_USER_MSG_MODE_SUPPRESS)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

ShowFolder

メソッド:

short ShowFolder(

boolean **Show**,

short **Left**,

short **Top**,

short **Right**,

short **Bottom**)

パラメーター:

Show

値が非ゼロの場合は、「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示するよう指示します。ゼロの場合は隠すように指示します。

Left

ダイアログ・ボックスが表示される左上の地点の X 座標を指定します。Show がゼロの場合は、このパラメーターは無視されます。Show が非ゼロでこのパラメーターがゼロ未満の場合、ダイアログ・ボックスは、現行サイズで現行位置に表示されます。

Top

ダイアログ・ボックスが表示される左上の地点の Y 座標を指定します。

Right

ダイアログ・ボックスが表示される右下の地点の X 座標を指定します。

Bottom

ダイアログ・ボックスが表示される右下の地点の Y 座標を指定します。

説明: 「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示するかまたは隠します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: OpenFolder, ActivateFolder

例: 以下の例では、画面サイズの 10% に相当する枠で画面中央に「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc, left, top, right, bottom;
int screen_width, screen_height;
  .
  .
 screen_width = GetSystemMetrics( SM_CXSCREEN );
 screen_height = GetSystemMetrics( SM_CYSCREEN );
 left = screen_width / 10;
 top = screen_height / 10;
 right = screen_width * 8 / 10;
 bottom = screen_height * 8 / 10;

 rc = pArsCtrl->ShowFolder( TRUE, left, top, right, bottom );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.ShowFolder (True, _
                        Screen.Width / 10, _
                        Screen.Height / 10, _
                        Screen.Width * 8 / 10, _
                        Screen.Height * 8 / 10)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
```

```
End If

 .
 .
```

ShowWaitCursorDuringCancelableOperation

メソッド:

short ShowWaitCursorDuringCancelableOperation(

boolean **Show**)

パラメーター:

Show

値がゼロの場合は、SearchFolder 操作または OpenDoc 操作中に OnDemand が待機カーソルを表示しないよう指示します。非ゼロの場合は、待機カーソルを表示するよう指示します。これがデフォルトの動作です。

説明: OnDemand は、通常、取り消し可能な操作中に待機カーソルを表示します。アプリケーションで取り消しボタンまたは類似機能を提供し、ユーザーが取り消し操作を開始できるようにする場合、待機カーソルが表示されないようにしたほうがよいことがあります。そのようにすると、砂時計や矢印などのアプリケーション独自のカーソルをアプリケーションで表示し、取り消しを要求できることをユーザーに知らせることができます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CancelOperation

例: 以下の例では、取り消し可能な操作中に待機カーソルが表示されないようにします。

C/C + + の例

```
CArsOle * pArsCtrl;
  .
  .
  .
pArsCtrl->ShowWaitCursorDuringCancelableOperation( FALSE );
  .
  .
  .
```

Visual Basic の例

```
  .
  .
  .
ArsOle.ShowWaitCursorDuringCancelableOperation (False)
  .
  .
  .
```

StoreDoc

メソッド:

short **StoreDoc**(
char * **DocPath**,
char * **ApplGrpName**,
char * **ApplName**,
VARIANT * **Values**)

パラメーター:

DocPath

OnDemand データベースに保管される文書データが入っているファイルの完全修飾パスを指定します。このパラメーターは必須です。

ApplGrpName

フォルダー内の「アプリケーション・グループ (Application Group)」の名前を指定します。アクティブ・フォルダーに関連した「アプリケーション・グループ (Application Group)」名は、呼び出し元の責任で知る必要があります。このパラメーターは必須です。StoreDoc() が正常に実行されるためには、次の要件を満たしている必要があります。

- 「アプリケーション・グループ (Application Group)」のデータベース編成は、`Multiple Loads per Table` 属性がなければなりません。この属性は、アプリケーション・グループの「プロパティー」ウィンドウの「一般/拡張 (General/Advanced)」タブで設定します。
- 「ストレージ管理」内の「アプリケーション・グループの満了タイプ (Application Group's Expiration Type)」は、「満了タイプ: セグメント (`Expiration Type: Segment`)」に設定されていなければなりません。この属性は、アプリケーション・グループの「プロパティー」ウィンドウの「ストレージ管理」タブで設定します。

ApplName

指定した「アプリケーション・グループ (Application Group)」内の「アプリケーション」の名前を指定します。指定した「アプリケーション・グループ (Application Group)」に関連した「アプリケーション」名は、呼び出し元の責任で知る必要があります。このパラメーターは必須です。

Values

フォルダー値の SafeArray を指します。各値は、フィールド・タイプ (整数や日付など) のデータに変換される文字ストリングです。

フォルダー・フィールドの数と順序は、GetFolderFieldName メソッドまたは GetFolderFieldNames メソッドを使用して判別できます。これらのメソッドの詳細は、「*Windows クライアント・カスタマイズ・ガイド*」の「GetFolderFieldName」と「GetFolderFieldNames」を参照してください。

値を指定しないフォルダー・フィールドには、ストリング・フィールドについては空ストリング、数値フィールドについてはゼロが入ります。無関係なフィールドを指定すると、そのフィールドは無視されます。

日付フィールドには、このフィールドに必須の形式で値を指定する必要があります (例えば、February 3, 1996 とする必要があるときに 02/03/96 と指定すると無効になります)。

説明: OnDemand は、フォルダー・フィールド値をアプリケーション・グループ・フィールドに変換し、指定した「アプリケーション・グループ (Application Group)」および「アプリケーション」に関連した文書として、指定したファイルのデータをデータベースに保管します。

戻りコードが以下のいずれかである場合、

```
ARS_OLE_RC_INVALID_DATE_FIELD
ARS_OLE_RC_INVALID_INTEGER_FIELD
ARS_OLE_RC_INVALID_DECIMAL_FIELD
ARS_OLE_RC_TOO_MANY_VALUE_CHARS
ARS_OLE_RC_INVALID_APPLGRP_FIELD_TYPE
```

GetStoreDocInvalidFieldNum メソッドは、問題の原因となったフォルダー・フィールドを突き止めるために使用できます。

リターン値:

戻りコードの説明は、「*Windows クライアント・カスタマイズ・ガイド*」の「戻りコード」を参照してください。

参照: GetNumFolderFields、GetFolderFieldName、GetFolderFieldNames、GetStoreDocInvalidFieldNum

例:

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;
SAFEARRAY * pSA;
VARIANT var;
BSTR bstrElement;
long i;
.
.
pSA = SafeArrayCreateVector(VT_BSTR,0,2);
if ( pSA == NULL )
ERROR;

bstrElement = SysAllocStringByteLen ("255-546-667",11);
i = 0;
SafeArrayPutElement (pSA, &i,bstrElement);
bstrElement = SysAllocStringByteLen ("06/07/94",8);
i = 1;
SafeArrayPutElement (pSA, &i,bstrElement);

var.vt = VT_ARRAY | VT_BSTR;
var.parray = pSA;

rc = pArsCtrl->StoreDoc("g:¥¥download ¥¥file.afp",
                        "BKH-CRD",
                        "BKH-CRD",
                        &var);
if (rc != ARS_OLE_RC_SUCCESS)
```

```
  ERROR;
.
.
.
```

Visual Basic の例

```
Dim values(2) As String
Dim rc As Integer
.
.
.
values(0) = "255-546-667"
values(1) = "06/07/94"
var = values
rc = ArsOle.StoreDoc ("g:¥download ¥file.afp",  _
                      "BKH-CRD",                _
                      "BKH-CRD",                _
                      var)
if rc <> ARS_OLE_RC_SUCCESS Then
  MsgBox "ERROR"
  End
End If
.
.
.
```

UndoFind

メソッド:

short UndoFind( )

説明: 現在検索されたストリングが強調表示されないようにします。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: FindStringInDoc

例: 以下の例では、検索されたストリングが強調表示されないようにします。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
  .
  .
 rc = pArsCtrl->UndoFind( );
 if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;
  .
  .
```

Visual Basic の例

```
Dim rc As Integer

 .
 .

rc = ArsOle.UndoFind ()
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

 .
 .
```

UpdateDoc

メソッド:

short **UpdateDoc**(

long * **DocIndex**,

char * **pFieldName**,

char * **pValue**)

パラメーター:

DocIndex

アクティブ・フォルダーの文書リスト内の、0 を基準とした相対文書番号を指定します。このパラメーターは必須です。

pFieldName

フォルダー・フィールドの名前を指定します。このパラメーターは必須です。

pValue

指定したフォルダー・フィールドに保管される値を指定します。この値は、フィールド・タイプ (整数や日付など) のデータに変換される文字ストリングです。

日付フィールドには、このフィールドに必須の形式で値を指定する必要があります (例えば、February 3, 1996 とする必要があるときに 02/03/96 と指定すると無効になります)。

このパラメーターは必須です。

説明: OnDemand は、フォルダー・フィールド値をアプリケーション・グループ・フィールドに変換し、データベース内の指定した値からデータを更新します。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: GetNumFolderFields、GetFolderFieldName、GetFolderFieldNames

例: 以下の例では、アクティブ・フォルダーの文書リスト内で最初の文書の「『バランス®』」フィールドを更新します。

C/C++ の例

```
CArsOle * pArsCtrl;
short rc;

  .
  .

rc = pArsCtrl->UpdateDoc( 0,
                          "Balance",
                          "123.45" );
if ( rc != ARS_OLE_RC_SUCCESS )
   ERROR;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer

  .
  .

rc = ArsOle.UpdateDoc ( 0,          _
                        "Balance", _
                        "123.45" )
if rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

  .
  .
```

WasOperationCancelled

メソッド:

short WasOperationCancelled(

VARIANT * **pCancelled**)

パラメーター:

pCancelled

結果を受け取る変数を指します。リターン時に、この変数は、タイプ VT_I2 に設定されます。

説明: 最新の SearchFolder 操作、OpenDoc 操作または RetrieveDoc 操作が取り消された場合、この結果の変数は、非ゼロ値に設定されます。それ以外の場合は、ゼロに設定されます。

リターン値:

戻りコードの説明については、4 ページの『戻りコード』を参照してください。

参照: CancelOperation

例: 以下の例ではフォルダーを検索し、検索が取り消されたかどうかを判別します。

C/C + + の例

```
CArsOle * pArsCtrl;
short rc;
VARIANT cancelled;

  .
  .

rc = pArsCtrl->SearchFolder( FALSE );
if ( rc != ARS_OLE_RC_SUCCESS )
  ERROR;

rc = pArsCtrl->WasOperationCancelled( &cancelled );
if ( cancelled.iVAL )
  CANCELLATION LOGIC;

  .
  .
```

Visual Basic の例

```
Dim rc As Integer
Dim cancelled As Variant

 .
 .


rc = ArsOle.SearchFolder (False)
If rc <> ARS_OLE_RC_SUCCESS Then
   MsgBox "ERROR"
   End
End If

rc = ArsOle.WasOperationCancelled (cancelled)
If cancelled <> 0 Then
   CANCELLATION LOGIC
End If

 .
 .
```

第 3 章 OLE イベント

以下のイベントは、OnDemand OLE Control によって引き起こされます。

FolderSearchCompleted

このイベントは、フォルダー検索が完了したときに発生します。この検索は、以下のいずれかによって開始された可能性があります。

- SearchFolder メソッドを呼び出すコンテナー・アプリケーション
- ユーザー。ShowFolder メソッドを使用して「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示した場合。

FolderClosed

このイベントは、フォルダーがクローズするときに発生します。このクローズは、以下のいずれかによって開始された可能性があります。

- CloseFolder メソッドを呼び出すコンテナー・アプリケーション
- ユーザー。ShowFolder メソッドを使用して「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示した場合。
- ログオフ
- コントロールの破損

DocOpened

このイベントは、文書が開くときに発生します。このオープンは、以下のいずれかによって開始された可能性があります。

- OpenDoc メソッドを呼び出すコンテナー・アプリケーション
- ユーザー。ShowFolder メソッドを使用して「OnDemand フォルダー (OnDemand Folder)」ダイアログ・ボックスを表示した場合。

DocClosed

このイベントは、文書がクローズするときに発生します。このクローズは、以下のいずれかによって開始された可能性があります。

- CloseDoc メソッドを呼び出すコンテナー・アプリケーション
- クローズされる文書に関連したフォルダー
- ログオフ
- コントロールの破損

AreaSelected

このイベントは、ユーザーが画面のエリアを選択するときに発生します。これが発生する可能性があるのは、SetSelectionMode メソッドを使用して選択モードを ARS_OLE_SELECTION_MODE_AREA または ARS_OLE_SELECTION_MODE_TEXT に設定した場合だけです。

AreaDeselected

このイベントは、選択が解除されるときに発生します。この解除は、以下のいずれかによって発生した可能性があります。

- ユーザーが選択対象の外側でマウスをダブルクリックする。
- SetSelectionMode メソッドで ARS_OLE_SELECTION_MODE_NONE を指定する。
- 文書がクローズされる。
- 特定の文書に関して、ページ、スクロール位置、ズーム、またはその他の表示属性を変更する。

UserCommand(long CommandID)

このイベントは、ユーザーが右マウス・ボタンをクリックしてメニュー項目を選択すると発生します。そのメニュー項目に関連したコマンド ID は、パラメーターとして引き渡されます。

メニューの作成については、SetRightButtonMenu メソッドを参照してください。

第 2 部 Windows 32 ビット GUI カスタマイズ・ガイド

第 4 章 OnDemand のカスタマイズの概要

OnDemand アプリケーションのカスタマイズは、コマンド行パラメーターを指定する、このアプリケーションを動的データ交換インターフェースで別のアプリケーションから呼び出して操作する、別のアプリケーションと DLL ファイルをクライアントから呼び出す、関連文書を取り出す、Program Information File を作成する、文書を監査する、という各方法によって行われます。

第 5 章 コマンド行

このセクションでは、以下について説明します。
- OnDemand の開始
- コマンド行パラメーターに適用されるパラメーター構文規則
- OnDemand 32 ビット Windows クライアントが認識するパラメーター

OnDemand 32 ビット・クライアントの開始

デスクトップ上でプログラム・アイコンをダブルクリックすると、OnDemand を開始できます。

パラメーター構文

コマンド行パラメーターには、以下の構文規則が適用されます。
- 実行可能名のあとにスペース文字を少なくとも 1 つ入れる必要があります。
- パラメーターは、スラッシュおよびこのすぐあとに続く 1 つの文字によって識別されます。
- パラメーター文字は、大/小文字の区別がなされません。
- 関連したパラメーター値がある場合には、その値は、パラメーター文字に続く最初の非スペース文字で始まり、次のパラメーターの前にある最後の非スペース文字で終了するか、またはコマンド行の終わりによって終了します。
- スラッシュ文字をパラメーター値の中に含めるときには、2 個のスラッシュ文字を連続して指定する必要があります。
- キーワード・パラメーター値は、大/小文字の区別がなされません。
- 認識されないパラメーターは無視されます。
- 同じパラメーターが複数回出現した場合、最後に出現したパラメーターが使用されます。

パラメーター

以下のパラメーターは、OnDemand Windows クライアントで認識されます。

製品名称 (Product Title) - /T 名称

このパラメーターを指定した場合、OnDemand は、OnDemand のメイン・ウィンドウの最上部および多数のメニュー項目上にあるデフォルトの製品名称を置換します。Program Information File の中にある名称よりもこの名称のほうが優先されます(195 ページの『第 12 章 Program Information File』を参照してください)。

例

```
C:¥ARS32¥ARSGUI32 /T Joe's Windows Application
```

ログオン・サーバー名 (Logon Server Name) - /S 名前

このパラメーターを指定した場合、OnDemand は、初期ユーザー・ログオン用のサーバー名としてこの値を使用します。この名前は、レジストリーに定義されているサーバー名のうちの 1 つにする必要があります。(サーバー名をレジストリーに追加するには、Update Servers コマンドを使用してください。)

このパラメーターは、初期設定時にユーザーをログオンさせるために、「ログオン・ユーザー ID (Logon User ID)」および「ログオン・パスワード (Logon Password)」パラメーターと組み合わせて使用できます。3 つのパラメーターすべてを指定したときに、すべてが有効であれば、初期「ログオン」ダイアログ・ボックスは表示されません。

例

```
C:¥ARS32¥ARSGUI32 /S pikes  /U user1  /P pass1
```

ログオン・ユーザー ID (Logon User ID) - /U ID

このパラメーターを指定した場合、OnDemand は、初期ユーザー・ログオン用のユーザー ID としてこの値を使用します。

このパラメーターは、初期設定時にユーザーをログオンさせるために、「ログオン・サーバー名 (Logon Server Name)」および「ログオン・パスワード (Logon Password)」パラメーターと組み合わせて使用できます。3 つのパラメーターすべてを指定したときに、すべてが有効であれば、初期「ログオン」ダイアログ・ボックスは表示されません。

例

```
C:¥ARS32¥ARSGUI32 /S pikes  /U user1  /P pass1
```

ログオン・パスワード (Logon Password) - /P パスワード

このパラメーターを指定した場合、OnDemand は、初期ユーザー・ログオン用のパスワードとしてこの値を使用します。

このパラメーターは、初期設定時にユーザーをログオンさせるために、「ログオン・サーバー名 (Logon Server Name)」および「ログオン・ユーザー ID (Logon User ID)」パラメーターと組み合わせて使用できます。3 つのパラメーターすべてを指定したときに、すべてが有効であれば、初期「ログオン」ダイアログ・ボックスは表示されません。

例

```
C:¥ARS32¥ARSGUI32 /S pikes  /U user1  /P pass1
```

パスワード変更 (Change Password) - /C 新規パスワード

このパラメーターを指定した場合、OnDemand は、正常なログオン後にユーザーのパスワードを変更します。このパラメーターを指定できるのは、/S、/U、および /P パラメーターも指定した場合だけです。

例

```
C:¥ARS32¥ARSGUI32 /S pikes  /U user1  /P pass1 /C newpass
```

フォルダー名 (Folder Name) - /F 名前

このパラメーターを指定した場合、OnDemand は、開く対象の初期フォルダーとしてこの値を使用します。この値が有効なフォルダー名であれば、初期「フォルダーを開く」ダイアログ・ボックスは表示されません。

このパラメーターは、「ログオン・サーバー名 (Logon Server Name)」、「ログオン・ユーザー ID (Logon User ID)」、「ログオン・パスワード (Logon Password)」、「ログオフの使用不可化 (Disable Logoff)」、「パスワード変更 (Password Change)」、「最大オープン・フォルダー数 (Maximum Open Folders)」パラメーターと組み合わせて使用して、ユーザーによる操作を単一サーバー上の単一のフォルダーに制限できます。

例

```
C:¥ARS32¥ARSGUI32 /F Credit Card Statements
```

最大オープン・フォルダー数 (Maximum Open Folders) - /O 数

このパラメーターを指定した場合、OnDemand は、指定した値までにオープン・フォルダーの数を制限します。このパラメーターは、フォルダーが開くたびにフォルダー情報がサーバーから最新表示されるよう、「フォルダーのクローズ時でのメモリーの解放 (Free Memory When Folder Closed)」パラメーターと組み合わせて使用できます。

例

```
C:¥ARS32¥ARSGUI32 /O 1  /Q
```

ウィンドウの配置 (Window Placement) - /W 配置

このパラメーターを指定した場合、OnDemand は、この配置による指示に従ってメイン・ウィンドウを表示します。この配置は、以下の値のいずれかでなければなりません。

- **Z** によって、メイン・ウィンドウをズーム (最大化) するよう指示します。
- **I** によって、メイン・ウィンドウをアイコン化 (最小化) するよう指示します。
- **N** によって、メイン・ウィンドウを表示しない (非可視にする) よう指示します。このオプションを指定するのは、OnDemand を DDE インターフェースによって操作する場合だけにしてください。
- ***x***、***y***、***w***、***h*** によって、メイン・ウィンドウの配置を指示します。この場合、
  - ***x*** は、左端の起点を画面の幅に対するパーセンテージで示す値です。
  - ***y*** は、上端の起点を画面の高さに対するパーセンテージで示す値です。
  - ***w*** は、幅を画面の幅に対するパーセンテージで示す値です。
  - ***h*** は、高さを画面の高さに対するパーセンテージで示す値です。

このパラメーターを指定しないか、または配置の値を何も指定しないと、前回 OnDemand を終了するときに表示されていたようにメイン・ウィンドウが表示されます。

例

```
C:¥ARS32¥ARSGUI32 /W 5,10,90,80
```

DDE インターフェースの使用可能化 (Enable DDE Interface) - /I 数、パス、resid

このパラメーターを指定した場合、OnDemand は、動的データ交換インターフェースを使用可能にし、別のアプリケーションからのコマンドを受け入れる準備をします。

数、**パス**、および **resid** は、以下のように解釈されます。

- **数**は、1 から 5 の間の整数でなければなりません。この数は、提供される DDE アプリケーション切り替えメニュー項目の数を指示します (詳しくは、 149 ページの『ENABLE_SWITCH』を参照)。 デフォルト値は 1 です。
- **パス**は、DDE アプリケーション切り替えメニュー項目に関連付けられるツールバー・ボタンのビットマップが入っているリソース DLL の完全修飾名を指定します。このビットマップのサイズは、メニュー項目の数によって決まります。それぞれのボタンには、16 X 15 ピクセルのイメージが必要です。メニュー項目が 1 つの場合は、ビットマップは 16 X 15、メニュー項目が 2 つの場合は 16 X 30 という具合になります。
- **resid** は、DLL 内のビットマップ・リソース ID を指定します。この ID は整数値でなければなりません。

例

```
C:¥ARS¥ARSGUI32 /I 3,C:¥MYAPP¥BITMAP.DLL,81
```

終了の使用不可化 (Disable Exit) - /K

このパラメーターを指定した場合、OnDemand は、「終了」メニュー項目を使用不可にします。

例

```
C:¥ARS¥ARSGUI32 /K
```

ログオフとパスワード変更の使用不可化 (Disable Logoff or Password Change) - /X

X パラメーターは、以下に説明するように、OnDemand がログオフまたはログオン・パスワードの変更を使用不可にするかどうかを指示するために使用します。

- オプションを使用しないで **X** を指定すると、OnDemand は、「ログオフ」メニュー項目と「ログオン・パスワードの変更」メニュー項目を使用不可にします。
- **L** オプションを使用して **X** を指定すると、OnDemand は、「ログオフ」メニュー項目を使用不可にします。
- **P** オプションを使用して **X** を指定すると、OnDemand は、「ログオン・パスワードの変更」メニュー項目を使用不可にします。

例

```
C:¥ARS32¥ARSGUI32 /X
```

サーバーの更新の使用不可化 (Disable Update Servers) - /Y

このパラメーターを指定した場合、OnDemand は、「ログオン」ダイアログ・ボックスの「サーバーの更新」ボタンを使用不可にします。

例

```
C:¥ARS32¥ARSGUI32 /Y
```

フォルダーのクローズの使用不可化 (Disable Close Folder) - /Z

このパラメーターを指定した場合、OnDemand は、「フォルダーを閉じる」メニュー項目、「検索基準と文書リスト」ダイアログ・ボックスの「フォルダーを閉じる」ボタン、および「検索基準と文書リスト」ダイアログ・ボックスのシステム・メニューの「閉じる」メニュー項目を使用不可にします。

例

```
C:¥ARS32¥ARSGUI32 /Z
```

予期の使用不可化 (Disable Anticipation) - /V

このパラメーターを指定した場合、OnDemand は、次のユーザー要求、例えば、初期設定後の「ログオン」ダイアログ・ボックスの表示とか現行フォルダーのクローズ後の「フォルダーを開く」ダイアログ・ボックスの表示などを予期しようとしなくなります。

例

```
C:¥ARS32¥ARSGUI32 /V
```

ユーザーの確認の使用不可化 (Disable User Confirmation) - /B

このパラメーターを指定した場合、OnDemand は、フォルダーのクローズ、ログオフ、または終了の際に開いている文書があってもユーザーのアクションの確認を求めません。

例

```
C:¥ARS32¥ARSGUI32 /B
```

フォルダーのクローズ時でのメモリーの解放 (Free Memory When Folder Closed) - /Q

このパラメーターを指定した場合、OnDemand は、フォルダーのクローズ時にそのフォルダーに関連したメモリーすべてを解放し、フォルダーの再オープン時にはサーバーからフォルダー情報を最新表示します。

例

```
C:¥ARS32¥ARSGUI32 /Q
```

言語パス (Language Path) - /1

このパラメーターは、クライアントの各国語環境をサポートするファイルの絶対パスを識別します。

例

```
C:¥ARS32¥ARSGUI32 /1 C:¥ARS32¥ARSGUI32¥LOCALE¥ENU
```

第 6 章 動的データ交換 (DDE) および DDE Management Library

動的データ交換 (DDE) は、Windows でサポートされているプロセス間通信メカニズムです。 2 つの Windows アプリケーションが、DDE プロトコルで「会話」を行うことができます。 OnDemand は、このような会話でサーバーとして機能することができ、クライアント・アプリケーションの指示でデータへのアクセスを提供し、アクションを実行します。クライアント・アプリケーションは、「リモート制御」によって OnDemand を操作できます。

DDE は、Windows に組み込まれたメッセージ・システムをベースにしています。メッセージを互いに通知することによって、アプリケーション間で通信します。このオリジナルの DDE 通信プロトコルは、理解してインプリメントするのが困難なことがあります。Windows では、DDE Management Library (DDEML) が提供されて DDE に関する複雑な面の多くが隠れるようになっています。この DDEML は、関数呼び出しインターフェースです。

オリジナルの DDE プロトコルと DDEML のどちらも、OnDemand と通信するときにクライアント・アプリケーションで使用できます。これ以降の項の説明と例では DDEML を使用しているので、DDEML の操作を全般的に理解していることが前提となっています。DDE および DDEML の詳細説明は、「*Microsoft Windows Software Development Kit*」、または適切な資料を参照してください。

別の Windows アプリケーションからの OnDemand 32 ビット の呼び出し

DDE インターフェースが必要なときに、通常、この呼び出し方式が用いられます。

OnDemand 32 ビット は、*CreateProcess* Windows API を使用して別の Windows のアプリケーションから呼び出すことができます。 この関数のプロトタイプは、次のとおりです。

```
BOOL CreateProcess(
              lpszImageName,
              lpszCmdLine,
              lpsaProcess,
              lpsaThread,
              fInheritHandles,
              fdwCreate,
              lpvEnvironment,
              lpszCurDir,
              lpsiStartInfo,
              lppiProcInfo )
```

lpszCmdLine は、OnDemand 実行可能ファイルの名前を含むヌル終了文字ストリングであり、このあとには、下に記述されている一連のオプション・パラメーターが続きます。

その他のパラメーターの説明については、お手持ちの資料を参照してください。

動的データ交換インターフェースが必要なときに、通常、この呼び出し方式が用いられます。

CreateProcess を使用した呼び出しの例を以下に示します。

```
PROCESS_INFORMATION pi;
STARTUPINFO sui;
char * pCmd;

memset( &sui, 0, sizeof(STARTUPINFO) );
sui.cb = sizeof(STARTUPINFO);

pCmd = "C:¥¥ARS¥¥ARSGUI32 "
       "/T Special OnDemand "
       "/I 3,C:¥¥MYAPP¥¥BITMAP.DLL,81 "
       "/W 5,10,90,80";

CreateProcess( NULL,
               pCmd,
               NULL,
               NULL,
               FALSE,
               CREATE_NEW_CONSOLE,
               NULL,
               NULL,
               &sui,
               &pi );
```

OnDemand の呼び出しと DDEML の初期設定

OnDemand とクライアント・アプリケーション間の DDE 会話が確立されるためには、あらかじめ、OnDemand がアクティブな Windows アプリケーションになっていることが必要です。通常、クライアント・アプリケーションは、(少なくとも) DDE インターフェースの使用可能化 (Enable DDE Interface) パラメーターを指定して、133 ページの『別の Windows アプリケーションからの OnDemand 32 ビットの呼び出し』で説明したように OnDemand を呼び出します。このパラメーターは、OnDemand にその DDE インターフェースを使用可能にさせます。このパラメーターを指定しないと、OnDemand は、すべての DDE 通信を無視します。

その他のパラメーターを使用すると、ウィンドウの配置、カスタム名称の提供、ユーザーのログオン、フォルダーのオープンなどを行うことができます。これらのアクションの多くは、DDE 関数によって実行することもできます。OnDemand ウィンドウの初期の外観を設定するには、コマンド行パラメーターを使用する必要があります。コマンド行パラメーターの詳細については、127 ページの『パラメーター』を参照してください。

OnDemand がアクティブになったあと、クライアント・アプリケーションはそれ自身を DDEML に知らせる必要があります。これは、DdeInitialize DDEML 関数によって実行されます。

最後に、クライアント・アプリケーションは、OnDemand との DDE 会話を確立する必要があります。これは、DdeConnect DDEML 関数によって実行されます。OnDemand サーバー・アプリケーションのサービス名は ARS です。会話のトピック名も ARS です。

Windows 版 OnDemand との DDE 会話を確立するための典型的な例を以下に示します。

```
 /* Global Variables */
 DWORD DdeInstance;
 HCONV hDdeConv;
      .
      .
      .
 /* Local Variables */
 FARPROC pfnDdeCallBack;
 HSZ hDdeString1, hDdeString2;
 UINT rc;
 char cmdline[500], buffer[500];
      .
      .
      .
 /* Invoke OnDemand */
PROCESS_INFORMATION pi;
STARTUPINFO sui;

memset( &sui, 0, sizeof(STARTUPINFO) );
sui.cb = sizeof(STARTUPINFO);

rc = CreateProcess( NULL,
                    cmdline,
                    NULL,
                    NULL,
                    FALSE,
                    CREATE_NEW_CONSOLE,
                    NULL,
                    NULL,
                    &sui,
                    &pi );

if ( !rc )
{
  sprintf( buffer,
           "CreateProcess of '%s' failed with error %ld",
           cmdline,
           (long)GetLastError( ) );
  MESSAGE( buffer );
  return;
}
 /* Initialize DDEML */
 pfnDdeCallBack = MakeProcInstance( (FARPROC)DdeCallBack, hInst );
 DdeInstance = 0;
 DdeInitialize( DdeInstance,
                (PFNCALLBACK)pfnDdeCallBack,
                APPCLASS_STANDARD | APPCMD_CLIENTONLY,
                0L );
 if ( DdeInstance == 0 )
 {
   MESSAGE( "DdeInitialize failed" );
   return;
 }
 /* Connect to OnDemand DDE Interface */
 hDdeString1 = DdeCreateStringHandle( DdeInstance, "ARS", 0 );
 hDdeString2 = DdeCreateStringHandle( DdeInstance, "ARS", 0 );
 for ( j = 0; j < 100; j++ )
 {
   hDdeConv = DdeConnect( DdeInstance, hDdeString1, hDdeString2, NULL );
   if ( hDdeConv != NULL )
     break;
 }
 DdeFreeStringHandle( DdeInstance, hDdeString1 );
 DdeFreeStringHandle( DdeInstance, hDdeString2 );
```

```
if ( hDdeConv == NULL )
{
  MESSAGE( "Unable to connect to OnDemand" );
  return;
}
    .
    .
    .
```

DDEML の終了

クライアント・アプリケーションが DDE 会話を終了させるときは、DdeUninitialize DDEML 関数を使用するだけで済みます。この関数は、DDE クライアントが会話を終了したことを OnDemand に通知しますが、OnDemand を終了させることはありません。クライアントは、OnDemand を終了させる場合、会話を終了する前に、OnDemand に対して 終了 (EXIT) する (150 ページの『EXIT』 を参照) よう指示する必要があります。

また、クライアントは、OnDemand によって DDE 会話が終了されるように準備をしておく必要があります。OnDemand が終了すると、DDE EXIT とは無関係に、またはこの結果として、クライアントは、DDEML から XTYP_DISCONNECT トランザクションを受け取ります。

DDEML トランザクション

どの OnDemand コマンドも DDEML XTYP_REQUEST トランザクションです。コマンドによっては、操作が実行されたり、追加データが戻されたりします。また、どのコマンドでも、戻りコードが提供されます。

トランザクションを開始するには、DDEML DdeClientTransaction 関数を使用します。DDEML 項目名ストリングには、コマンドと連結パラメーターが含まれます。その構文は、実行可能名を置換する DDE コマンドのコマンド行 (127 ページの『パラメーター構文』を参照) と同じです。

DDEML DdeClientTransaction 関数は、戻りコードを含むデータ・ハンドルを戻し、オプションにより追加データも戻します。その戻りコードは、個々のコマンドについて記述された値 (その要約が 177 ページの『第 8 章 戻りコード』に記載されています。) の 1 つに対応する一連の ASCII 数字です。これらの ASCII 数字のあとには、必ず、1 つのスペース文字が続いています。追加データがある場合、このデータは、ヌル終了文字ストリングであり、スペース文字の直後の文字で始まります。

OnDemand DDE コマンドを実行してリターン情報を受け取る C 関数について、以下の例で説明します。個々の OnDemand DDE コマンドについての例はすべて、この C 関数の例をベースにしています。

```
/* Global Variables */
DWORD DdeInstance;
HCONV hDdeConv;

#define ERROR_MAP struct _ErrorMap
ERROR_MAP
{
  int    code;
  char * pMsg;
```

```
};

static ERROR_MAP Errors[] =
  { { ARS_DDE_RC_UNKNOWN_COMMAND,      "Unknown command." },
    { ARS_DDE_RC_PARM_NOT_SPECIFIED,   "Parameter not specified." },
    { ARS_DDE_RC_INVALID_PARM_VALUE,   "Invalid parameter value." },
    { ARS_DDE_RC_SERVER_ERROR,         "Server error." },
    { ARS_DDE_RC_FILE_ERROR,           "File error." },
    { ARS_DDE_RC_NOT_LOGGED_ON,        "Not logged on." },
    { ARS_DDE_RC_MAX_FOLDERS_OPEN,     "Maximum folders open." },
    { ARS_DDE_RC_FOLDER_NOT_OPEN,      "Folder not open." },
    { ARS_DDE_RC_NO_DOC,               "No document exists." },
    { ARS_DDE_RC_OPEN_DOCS,            "Documents are open." } };
#define NUM_ERRORS  ( sizeof(Errors) / sizeof(ERROR_MAP) )

    .
    .
    .
BOOL DoDdeCommand( char * pCmd,    /* -> Command string           */
                   char * pParms,  /* -> Command parameters       */
                   char * pData )  /* -> Buffer for returned data */
{
  HSZ hDdeString;
  HDDEDATA hDdeResult;
  DWORD data_len;
  char * pString;
  int j, rc;

  /* Add parameters to command line. */
  if ( pParms == NULL )
    pParms = "";
  pString = malloc( strlen( pCmd ) + strlen( pParms ) + 2 );
  strcpy( pString, pCmd ):
  strcat( pString, " " );
  strcat( pString, pParms );

  /* Perform OnDemand DDE command. */
  hDdeString = DdeCreateStringHandle( DdeInstance, pCmd, 0 );
  hDdeResult = DdeClientTransaction( NULL,
                                     0,
                                     hDdeConv,
                                     hDdeString1,
                                     CF_TEXT,
                                     type,
                                     10000L,
                                     NULL );
  DdeFreeStringHandle( DdeInstance, hDdeString );
  free( pString);

  /* Process command result. */
  if ( hDdeResult == NULL )
  {
    /* This should never happen. */
    MESSAGE( "DDE Timeout." );
    return FALSE;
  }
  else
    {
    pString = (char*)DdeAccessData( hDdeResult, &data_len );
    rc = atoi( pString );
    if ( rc == ARS_DDE_RC_NO_ERROR )
    {
      if ( pData != NULL )
      strcpy( pData, strchr( pString, ' ' ) + 1 );
    }
    else
    {
```

```
    if ( pData != NULL )
      pData[0] = '¥0';
    for ( j = 0; j < NUM_ERRORS; j++ )
      if ( Errors[j].code == rc )
        break;
    MESSAGE( j < NUM_ERRORS ? Errors[j].pMsg : "Error - invalid return code." );
   }
   DdeUnaccessData( hDdeResult );
   return rc == ARS_DDE_RC_NO_ERROR;
  }
}
```

第 7 章 OnDemand DDE コマンド

ヘッダー・ファイル ARSDDEEX.H は、OnDemand に付随しています。このヘッダー・ファイルには、DDE コマンドで使用される多くの値についての記号定義が含まれています。このヘッダー・ファイルは、C または C++ インプリメンテーションに組み込んだり、その他の言語で参照用に使用することができます。

このヘッダー・ファイルは、OnDemand インストール・ディレクトリーの INC サブディレクトリーにインストールされます。このサブディレクトリーを組み込みファイル・パスに追加することも、ヘッダー・ファイルを別のディレクトリーにコピーすることもできます。

ACTIVATE_DOC

コマンド	パラメーター
ACTIVATE_DOC	/D *doc id*

パラメーター:

D

アクティブにする文書の文書 ID (OPEN_DOC コマンドによって戻される) を指定します。このパラメーターは必須です。

アクション:

OnDemand は、その文書ウィンドウを一番上に表示し、このウィンドウに入力フォーカスを与えることによって、文書をアクティブ文書にします。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
6 ARS_DDE_RC_NOT_LOGGED_ON
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "ACTIVATE_DOC", "/D 1234", NULL );
```

ACTIVATE_FOLDER

コマンド	パラメーター
ACTIVATE_FOLDER	/F *folder name*

パラメーター:

F

現在開いているフォルダーのフォルダー名を指定します。このパラメーターは必須です。

アクション:

OnDemand は、指定したフォルダーをアクティブ・フォルダーにします。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
6 ARS_DDE_RC_NOT_LOGGED_ON
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "ACTIVATE_FOLDER", "Mary's Folder", NULL );
```

ANNOTATE_DOC

コマンド	パラメーター
ANNOTATE_DOC	/N *doc number* /P *annotation path* /G *page number* /U /C

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。

P

注釈のテキストが入っているファイルの完全修飾パスを指定します。このファイルに 32,700 バイトを超えるテキストが入っている場合、テキストは切り捨てられます。

このパラメーターは必須です。

G

注釈に関連した文書ページ番号を指定します。

このパラメーターはオプションです。

U

注釈は専用ではなく共通であることを示します。

このパラメーターはオプションです。

C

注釈を他のサーバーにコピーできることを示します。

このパラメーターはオプションです。

アクション:

OnDemand は、指定した文書に注釈を追加します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
4 ARS_DDE_RC_SERVER_ERROR
5 ARS_DDE_RC_FILE_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
9 ARS_DDE_RC_NO_DOC
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/N %s /L %s /C %s /R /K",
         "22",
         "HP LaserJet on LPT1.AES:",
         "2" );

DoDdeCommand( "PRINT_DOC", parms, NULL );
```

ARRANGE_DOCS

コマンド	パラメーター
ARRANGE_DOCS	/C /H /V /Z /R

パラメーター:

C

文書ウィンドウをカスケード表示するよう指示します。

H

文書ウィンドウを横方向にタイル表示するよう指示します。

V

文書ウィンドウを縦方向にタイル表示するよう指示します。

Z

文書ウィンドウをズーム (最大化) するよう指示します。

R

文書ウィンドウを復元するよう指示します。これがデフォルト・パラメーターです。

アクション:

OnDemand は、パラメーターによる指示に従って文書ウィンドウを配置します。パラメーターをまったく指定しない場合は、デフォルトで **R** になります。複数のパラメーターを指定した場合は、予測できない結果となります。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
6 ARS_DDE_RC_NOT_LOGGED_ON
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "ARRANGE_DOCS", "/V", NULL );
```

CHANGE_PASSWORD

コマンド	パラメーター
CHANGE_PASSWORD	/C *Current Password* /N *New Password* /V *New Password*

パラメーター:

C

ユーザーの現行パスワードを指定します。

N

ユーザーの新規パスワードを指定します。

V

ユーザーの新規パスワードをもう一度指定します。これは、確認を行うためです。

アクション:

OnDemand は、アクティブ・ユーザーのログオン・パスワードを変更します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
4 ARS_DDE_RC_SERVER_ERROR
6 ARS_DDE_RC_NOT_LOGGED_ON
22 ARS_DDE_RC_INCORRECT_CURRENT_PASSWORD
23 ARS_DDE_RC_PASSWORD_TOO_SHORT
24 ARS_DDE_RC_NEW_PASSWORD_MISMATCH

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CHANGE_PASSWORD", "/C tt1sd /N sfd45r /V sfd45r", NULL );
```

CLEAR_FIELDS

コマンド	パラメーター
CLEAR_FIELDS	なし

パラメーター:

なし

アクション:

OnDemand は、アクティブ・フォルダーの検索基準入力ウィンドウを消去します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

8 ARS_DDE_RC_FOLDER_NOT_OPEN

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CLEAR_FIELDS", "", NULL );
```

CLOSE_ALL_DOCS

コマンド	パラメーター
CLOSE_ALL_DOCS	なし

パラメーター:

なし

アクション:

OnDemand は、開いているすべての文書をクローズしてすべての文書ウィンドウを破棄し、アクティブ・フォルダー・ウィンドウを表示します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CLOSE_ALL_DOCS", "", NULL );
```

CLOSE_ALL_FOLDERS

コマンド	パラメーター
CLOSE_ALL_FOLDERS	なし

パラメーター:
なし

アクション:
OnDemand は、開いているすべてのフォルダーをクローズします。

戻りコード:
0 ARS_DDE_RC_NO_ERROR
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:
なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CLOSE_ALL_FOLDERS", "", NULL );
```

CLOSE_DOC

コマンド	パラメーター
CLOSE_DOC	/D *doc id*

パラメーター:
D

クローズする文書の文書 ID (OPEN_DOC コマンドによって戻される) を指定します。このパラメーターは必須です。

アクション:
OnDemand は、文書ウィンドウを破棄して文書をクローズします。

戻りコード:
0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
6 ARS_DDE_RC_NOT_LOGGED_ON
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:
なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CLOSE_DOC", "/D 1234", NULL );
```

CLOSE_FOLDER

コマンド	パラメーター
CLOSE_FOLDER	なし

パラメーター:
なし

アクション:
OnDemand は、アクティブ・フォルダーをクローズします。開いているフォルダーがほかにある場合、そのうちの 1 つがアクティブ・フォルダーになります。

戻りコード:
0 ARS_DDE_RC_NO_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:
なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "CLOSE_FOLDER", "", NULL );
```

COPY_DOC_PAGES

コマンド	パラメーター
COPY_DOC_PAGES	/P *path* /G *page number* | C /A

パラメーター:

P

ページのコピー先となるファイルの完全修飾パスを指定します。コピー先ファイルが存在していない場合は、そのファイルが作成されます。存在している場合は、その既存のファイルにページが付加されます。このパラメーターは必須です。

G

コピーするページを指定します。
コピーするページの *page number* または *page numbers* を指定します。

単一のページを指定することもページの範囲を指定することもできます。また、単一のページとページの範囲をコンマで区切りながら組み合わせて指定することもできます。例えば、次のように指定すると、

```
/G 3, 5-7, 9, 110-120
```

3 ページ、5 ページから 7 ページ、9 ページ、110 ページから 120 ページが印刷されます。

G パラメーターを指定しないと、すべてのページがコピーされます。

G パラメーターを指定してページまたはページの範囲を指定することも、**C** を指定して現行ページを指定することもできますが、ページと **C** を同時に指定することはできません。

G パラメーターはオプションです。

C

現行ページをコピーするには、**C** を指定します。

C パラメーターはオプションです。

A

このパラメーターを指定すると、ページは、データ・ストリーム・フォーマットで「現状のまま」コピーされます。指定しないと、ページは、ASCII テキストとしてコピーされます (AFP および行データについてのみ)。

このパラメーターはオプションです。

アクション:

OnDemand は、指定したページ (1 ページまたは複数の) をアクティブ文書から指定のファイルにコピーします。ページのリストを指定すると、指定した順序でページがコピーされます。

戻りコード:

2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
4 ARS_DDE_RC_SERVER_ERROR
5 ARS_DDE_RC_FILE_ERROR
6 ARS_DDE_RC_NOT_LOGGED_ON
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION
13 ARS_DDE_RC_USER_CANCELLED_OPERATION
14 ARS_DDE_RC_NO_ACTIVE_DOC

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/P %s /G 6, 3-4, 8, 112-118",
         "C:¥¥ASCII.TXT")

DoDdeCommand( "COPY_DOC_PAGES", parms, NULL );
```

DELETE_DOC

コマンド	パラメーター
DELETE_DOC	/N *doc number*

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。

doc number に -1 を指定して、選択したすべての文書を削除するよう指示することができます。

アクション:

OnDemand は、指定した文書または選択したすべての文書をデータベースから削除します。削除された文書 (1 つまたは複数の) は、「Document List」から削除されます。場合によっては文書番号が変更されているために、直前の GET_DOC_VALUES コマンドからの情報がすでに無効になっている場合があります。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM
4 ARS_DDE_RC_SERVER_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION

リターン・データ:

OnDemand は、正常に削除された文書の番号を戻します。戻されたヌル終了ストリングは、長整数に変換できます。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "DELETE_DOC", "23", NULL );
```

DESELECT_DOC

コマンド	パラメーター
DESELECT_DOC	/N *doc number*

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参

照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。

doc number に -1 を指定して、すべての文書の選択を解除するよう指示することができます。

アクション:

OnDemand は、特定の文書番号に対応する「Document List」行の選択を解除します (強調表示をやめます)。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "DESELECT_DOC", "15", NULL );
```

DISABLE_SWITCH

コマンド	パラメーター
DISABLE_SWITCH	**/ N** *number*

パラメーター:

N

使用不可にするメニュー項目またはツールバー・ボタンの番号を指定します。この番号は、1 から 130 ページの『DDE インターフェースの使用可能化 (Enable DDE Interface) - /I 数、パス、resid』で指定した番号までの間の整数でなければなりません。1 を指定すると、最初のメニュー項目またはツールバー・ボタンが使用不可になり、2 を指定すると 2 番目が使用不可になるという具合です。

このパラメーターはオプションです。デフォルトは、1 です。

アクション:

OnDemand は、ユーザーがウィンドウ・フォーカスをクライアント・アプリケーションに移すために使用できるツールバー・アイコンとメニュー項目を使用不可にします。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
3 ARS_DDE_RC_INVALID_PARM_VALUE
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "DISABLE_SWITCH", "", NULL );
```

ENABLE_SWITCH

コマンド	パラメーター
ENABLE_SWITCH	/H *client hWnd* /C *client name* /N *number*

パラメーター:

H

クライアント・アプリケーションのウィンドウ・ハンドルを一連の ASCII 数字で指定します。このパラメーターは必須です。このパラメーターの準備のしかたを、下記の例で示しています。

C

クライアント・アプリケーションの名前を指定します。このパラメーターはオプションです。このパラメーターを指定しないと、その名前は変更されません。

N

使用可能にするメニュー項目またはツールバー・ボタンの番号を指定します。この番号は、1 から 130 ページの『DDE インターフェースの使用可能化 (Enable DDE Interface) - /I 数、パス、resid』で指定した番号までの間の整数でなければなりません。1 を指定すると、最初のメニュー項目またはツールバー・ボタンが使用可能になり、2 を指定すると 2 番目が使用可能になるという具合です。

このパラメーターはオプションです。デフォルトは、1 です。

アクション:

OnDemand は、ユーザーがウィンドウ・フォーカスをクライアント・アプリケーションに移すために使用できるツールバー・アイコンとメニュー項目を使用可能にします。そのクライアント名は、メニュー項目のテキストとして設定します。

クライアント・アプリケーションが Advise Loop (詳しくは、 179 ページの『第 9 章 DDEML Advise Loop』を参照) を設定している場合、OnDemand は、ユーザーがフォーカスを移すとクライアント・アプリケーションに通知します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/H %ld /C %s",
         (long)hWnd,
         "Fred's Windows Application" );
DoDdeCommand( "ENABLE_SWITCH", parms, NULL );
```

EXIT

コマンド	パラメーター
EXIT	なし

パラメーター:

なし

アクション:

OnDemand は、このコマンドを受け取ると終了しようとします。文書が少なくとも 1 つは開いているときに、「ユーザー確認の使用不可化 (Disable User Confirmation)」コマンド行パラメーターが指定されていない場合 (131 ページの『ユーザーの確認の使用不可化 (Disable User Confirmation) - /B』を参照)、プログラムの終了についてユーザーに確認を求めるメッセージ・ボックスが表示されます。

OnDemand が終了する場合、クライアントは、DDEML から XTYP_DISCONNECT トランザクションを受け取ります。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "EXIT", "", NULL );
```

GET_DISPLAY_FIELDS

コマンド	パラメーター
GET_DISPLAY_FIELDS	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、アクティブ・フォルダーの表示フィールドのリストを戻します。このリストはヌル終了文字ストリングであり、各フィールド名がタブ文字で区切られています。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], fields[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_DISPLAY_FIELDS", "", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( fields[j], pToken );
  }
}
```

GET_DOC_VALUES

コマンド	パラメーター
GET_DOC_VALUES	/N *doc number* /S

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。

このパラメーターは必須です。

S

指定したリスト項目を選択した場合にだけ値を戻すよう指示します。

このパラメーターはオプションです。

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、アクティブ・フォルダーの文書リスト内の指定した文書について文書値のリストを戻します。このリストはヌル終了文字ストリングであり、各フィールド値がタブ文字で区切られています。その値は、GET_DISPLAY_FIELDS コマンド (150 ページの『GET_DISPLAY_FIELDS』を参照) で戻される表示フィールド名と同じ順序になっています。

文書番号がリストに入っていない場合、または **S** パラメーターを指定したときに文書番号を選択していない場合には、値は何も戻りません。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], values[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_DOC_VALUES", "8", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( values[j], pToken );
  }
}
```

GET_FOLDER_FIELDS

コマンド	パラメーター
GET_FOLDER_FIELDS	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、アクティブ・フォルダーのフォルダー・フィールドのリストを戻します。このリストはヌル終了文字ストリングであり、各フィールド名がタブ文字で区切られています。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], fields[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_FOLDER_FIELDS", "", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
```

```
      pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( fields[j], pToken );
  }
}
```

GET_FOLDERS

コマンド	パラメーター
GET_FOLDERS	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR

6 ARS_DDE_RC_NOT_LOGGED_ON

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、使用可能フォルダーのリストを戻します。このリストはヌル終了文字ストリングであり、各フォルダー名がタブ文字で区切られています。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], folders[10][31], * pToken;
int j;

if ( DoDdeCommand( "GET_FOLDERS", "", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( folders[j], pToken );
  }
}
```

GET_NUM_DOCS_IN_LIST

コマンド	パラメーター
GET_NUM_DOCS_IN_LIST	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR

8 ARS_DDE_RC_FOLDER_NOT_OPEN

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、アクティブ・フォルダーの文書リストに現在入っている文書の数を戻します。戻されたヌル終了ストリングは、長整数に変換できます。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char data[100];
long num_docs;

if ( DoDdeCommand( "GET_NUM_DOCS_IN_LIST", "", data ) )
  num_docs = atol( data );
```

GET_NUM_DOC_PAGES

コマンド	パラメーター
GET_NUM_DOC_PAGES	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR

6 ARS_DDE_RC_NOT_LOGGED_ON

14 ARS_DDE_RC_NO_ACTIVE_DOC

リターン・データ:

OnDemand は、アクティブ文書のページ数を戻します。

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char data[50];
long num_pages;

if ( DoDdeCommand( "GET_NUM_DOC_PAGES.", "", data ) )
  num_pages = atol( data );
```

GET_PRINTERS

コマンド	パラメーター
GET_PRINTERS	/L /S

パラメーター:

L

ローカル・プリンターのリストを戻すように指示します。このパラメーターはオプションですが、このパラメーターか **S** パラメーターのどちらかを指定する必要があります。

S

サーバー・プリンターのリストを戻すように指示します。このパラメーターはオプションですが、このパラメーターか **L** パラメーターのどちらかを指定する必要があります。

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
6 ARS_DDE_RC_NOT_LOGGED_ON
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、パラメーターによる指示に従ってローカル・プリンターかサーバー・プリンターのどちらかの使用可能プリンターのリストを戻します。両方のパラメーターを指定すると、予測できない結果となります。このリストはヌル終了文字ストリングであり、各プリンター名がタブ文字で区切られています。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], local_printers[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_PRINTERS", "/L", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( local_printers[j], pToken );
  }
}
```

GET_QUERY_FIELDS

コマンド	パラメーター
GET_QUERY_FIELDS	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、アクティブ・フォルダーの照会フィールドのリストを戻します。このリストはヌル終了文字ストリングであり、各フィールド名がタブ文字で区切られています。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], fields[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_QUERY_FIELDS", "", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( fields[j], pToken );
  }
}
```

GET_SERVERS

コマンド	パラメーター
GET_SERVERS	なし

パラメーター:

なし

アクション:

なし

戻りコード:

0 ARS_DDE_RC_NO_ERROR

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

OnDemand は、使用可能サーバーのリストを戻します。このリストはヌル終了文字ストリングであり、各サーバー名がタブ文字で区切られています。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char data[1000], servers[10][100], * pToken;
int j;

if ( DoDdeCommand( "GET_SERVERS", "", data ) )
{
  for ( pToken = strtok( data, "¥t" ), j = 0;
        pToken != NULL &&;amp; j < 10;
        pToken = strtok( NULL, "¥t" ), j++ )
  {
    strcpy( servers[j], pToken );
  }
}
```

LOGOFF

コマンド	パラメーター
LOGOFF	なし

パラメーター:

なし

アクション:

OnDemand は、ログオフを実行します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "LOGOFF", "", NULL );
```

LOGON

コマンド	パラメーター
LOGON	/S *server name* /U *user id* /P *password* /R

パラメーター:

S

ログオン・サーバー名を指定します。このパラメーターは必須です。

U

ログオン・ユーザー ID を指定します。このパラメーターは必須です。

P

ログオン・パスワードを指定します。このパラメーターは必須です。

R

ログオンに失敗したときに制御を戻すかどうかを指示します。このパラメーターはオプションです。このパラメーターを指定しないときにログオンに失敗すると、「OnDemand ログオン (OnDemand Logon)」ダイアログ・ボックスが表示されます。

アクション:

OnDemand は、指定したサーバー名、ユーザー ID、およびパスワードによってログオンを試行します。GET_SERVERS コマンド (156 ページの『GET_SERVERS』を参照) を使用すると、使用可能サーバー名のリストを取り出すことができます。

ユーザーがすでにログオンしている場合は、ログオンの前にログオフが実行されます。ログオンが失敗したときのアクションは、/R パラメーターによって決まります。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
13 ARS_DDE_RC_USER_CANCELLED_OPERATION
25 ARS_DDE_RC_INVALID_USER_PASS_SERVER
26 ARS_DDE_RC_PASSWORD_EXPIRED

リターン・データ:

ログオンに成功した場合、OnDemand は、ログオンに使用されたユーザー ID とパスワードを戻します。ヌル終了文字ストリングにはユーザー ID が含まれていて、そのあとにタブ文字、さらにそのあとにパスワードが続いています。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/S %s /U %s /P %s",
         "server26",
         "horatio",
         "vrxotwc" );

DoDdeCommand( "LOGON", parms, NULL );
```

OPEN_DOC

コマンド	パラメーター
OPEN_DOC	**/N** *doc number* **/P** *path*

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。 **P**

文書データが入っているファイルの完全修飾パスを指定します。

このパラメーターを指定すると、OnDemand は、データベースにアクセスして文書を取り出す代わりに、リソース・グループが要求された場合にこれを取り出します。

このパラメーターはオプションです。

アクション:

OnDemand は、文書を開いて 1 ページ目を文書ウィンドウに表示します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION
13 ARS_DDE_RC_USER_CANCELLED_OPERATION

リターン・データ:

文書が正常にオープンした場合、OnDemand は、*doc id* を戻します。このストリングには、最高 20 文字まで入っています。

この *doc id* は、ACTIVATE_DOC や CLOSE_DOC などのその他のコマンドのパラメーターとして必要になります。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char doc_id[21];

DoDdeCommand( "OPEN_DOC", "/N 23", doc_id );
```

OPEN_FOLDER

コマンド	パラメーター
OPEN_FOLDER	/F *folder name* /S /C /R /D /V /P /T /A /N /L /0 *button text* /1 *button text* /2 *button text* /3 *button text* /4 *button text* /5 *button text* /6 *button text* /7 *button text* /8 *button text* /9 *button text*

パラメーター:

F

サーバー上のユーザーが使用できるフォルダーのうちの 1 つのフォルダーの名前を指定します。このパラメーターは必須です。

S

「検索」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

C

「全フィールドの消去」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

R

「デフォルトに戻す」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

D

「AND/OR」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

V

「選択したすべてを表示」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示**しない**ように指示します。このパラメーターはオプションです。

P

「選択したすべてを印刷」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示**しない**ように指示します。このパラメーターはオプションです。

T

「リストのソート... (Sort List ...)」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

A

「監査 (Audit)」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示**しない**ように指示します。このパラメーターはオプションです。

N

「付加」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示**しない**ように指示します。このパラメーターはオプションです。

L

「自動スクロール」ボタンを「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示**しない** ように指示します。このパラメーターはオプションです。

0 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

1 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

2 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

3 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

4 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの検索基準に表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

5 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

6 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのア

プリケーションに通知します。Advise Loop の詳細については、 179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

7 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、 179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

8 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、 179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

9 *button text*

「検索基準と文書リスト」ダイアログ・ボックスの文書リストに表示するボタンのテキストを指定します。このパラメーターはオプションです。

テキストに & が含まれている場合、このあとの文字はアクセラレーター・キーになります。ユーザーがボタンをクリックすると、OnDemand は、クライアント・アプリケーション用に設定された Advise Loop を介してそのアプリケーションに通知します。Advise Loop の詳細については、 179 ページの『第 9 章 DDEML Advise Loop』を参照してください。

アクション:

OnDemand は、指定したフォルダーのオープンを試行します。
GET_FOLDERS コマンド (153 ページの『GET_FOLDERS』を参照) を使用すると、使用可能フォルダー名のリストを取り出すことができます。オープンに失敗した場合以外は、「フォルダーを開く」ダイアログ・ボックスは表示されません。フォルダーが正常に開いた場合、そのフォルダーがアクティブ・フォルダーになります。

同時に複数のフォルダーを開いたままにすることができます。それらのフォルダーのうちの 1 つは、アクティブ・フォルダーになっています。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
6 ARS_DDE_RC_NOT_LOGGED_ON
7 ARS_DDE_RC_MAX_FOLDERS_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "OPEN_FOLDER", "/F Mary's Folder", NULL );
```

PRINT_DOC

コマンド	パラメーター
PRINT_DOC	/N *doc number* /L *printer name* /S *printer name* /C *copies* /R *orientation* /K *margins* /M

パラメーター:

N *doc number*

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターはオプションです。このパラメーターを指定しない場合は、アクティブ文書が印刷されます。

L *printer name*

ローカル・プリンター名を指定します。GET_PRINTERS コマンド (154 ページの『GET_PRINTERS』を参照) を使用すると、使用可能ローカル・プリンターの名前を判別できます。

このパラメーターはオプションですが、このパラメーターか **S** パラメーターのどちらかを指定する必要があります。

S *printer*

サーバー・プリンター名を指定します。GET_PRINTERS コマンド (154 ページの『GET_PRINTERS』を参照) を使用すると、使用可能サーバー・プリンターの名前を判別できます。

このパラメーターはオプションですが、このパラメーターか **L** パラメーターのどちらかを指定する必要があります。

C *copies*

印刷する文書の部数を指定します。この値は、1 から 100 の数値にする必要があります。

このパラメーターはオプションです。このパラメーターを指定しない場合は、1 部印刷されます。

R *orientation*

印刷の際の文書の方向を指定します。*orientation* には、以下の値の 1 つを指定してください。

B は、用紙に最も合うように文書を回転させます。
P は、縦長方向で文書を印刷します。
L は、横長方向で文書を印刷します。
A は、プリンターでの指定に従って文書を印刷します。

orientation を指定しない場合、デフォルトで **B** になります。

このパラメーターはオプションです。このパラメーターを指定しない場合、デフォルトで **/R A** になります。

サーバー・プリンターを指定した場合、**R** パラメーターは無視されます。

K *margins*

使用するページ・マージンを指定します。*margins* には、***t***、***b***、***l***、***r*** を指定してください。ここで、
t は、上部マージンです。
b は、下部マージンです。
l は、左マージンです。
r は、右マージンです。

各マージン値は、非負の 10 進数にする必要があります。マージン値を何も指定しないと、現行マージン値が使用されます。

このパラメーターはオプションです。このパラメーターを指定しない場合、マージン 0 が使用されます。(マージンを 0 にすると、プリンターによってはデータが切り捨てられることがあります。)

サーバー・プリンターを指定した場合、**K** パラメーターは無視されます。

M

K パラメーターで指定するマージンをインチ単位ではなくミリメートル単位で指定することを指示します。

このパラメーターはオプションです。**M** パラメーターを指定しない場合、マージンは、デフォルトでインチ単位になります。

アクション:

OnDemand は、指定したプリンターで文書の 1 ページまたは複数のページを印刷します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE
6 ARS_DDE_RC_NOT_LOGGED_ON
8 ARS_DDE_RC_FOLDER_NOT_OPEN
9 ARS_DDE_RC_NO_DOC
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/N %s /L %s /C %s /R B /K 0.5,1.2,1,1",
         "17",
         "HP LaserJet on LPT1.AES:",
         "2" );

DoDdeCommand( "PRINT_DOC", parms, NULL );
```

RESTORE_DEFAULTS

コマンド	パラメーター
RESTORE_DEFAULTS	なし

パラメーター:

なし

アクション:

OnDemand は、アクティブ・フォルダーの検索基準入力ウィンドウをデフォルト値に設定します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

8 ARS_DDE_RC_FOLDER_NOT_OPEN

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、 136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "RESTORE_DEFAULTS", "", NULL );
```

RETRIEVE_DOC

コマンド	パラメーター
RETRIEVE_DOC	/N *doc number* /P *doc path* /R *resgrp path* /C *combined path* /A

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参

照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。

P

文書データを入れるファイルの完全修飾パスを指定します。このパラメーターはオプションです。

R

リソース・グループ・データを入れるファイルの完全修飾パスを指定します。このパラメーターはオプションです。

C

リソース・グループとデータを組み合わせて入れるファイルの完全修飾パスを指定します。このパラメーターはオプションです。

A

既存ファイルを置換するのではなく、データを既存ファイルに付加するよう指示します。このパラメーターはオプションです。

アクション:

OnDemand は、リソース・グループまたは文書データあるいはこの両方を取り出して、指定したファイルにそれをコピーするか付加します。パラメーターを任意に組み合わせて指定できますが、ただし、**N** は必須です。

戻りコード:

0	ARS_DDE_RC_NO_ERROR
2	ARS_DDE_RC_PARM_NOT_SPECIFIED
3	ARS_DDE_RC_INVALID_PARM_VALUE
4	ARS_DDE_RC_SERVER_ERROR
5	ARS_DDE_RC_FILE_ERROR
8	ARS_DDE_RC_FOLDER_NOT_OPEN
9	ARS_DDE_RC_NO_DOC
11	ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/N %d /C %s",
         26,
         "C:¥¥DATA¥¥COMBINED.FIL");

DoDdeCommand( "RETRIEVE_DOC", parms, NULL );
```

SEARCH_FOLDER

コマンド	パラメーター
SEARCH_FOLDER	/A /O

パラメーター:

A

検索の結果取り出された文書を既存のリストに付加するかどうかを指示します。このパラメーターはオプションです。このパラメーターを指定しないと、その文書で以前のリストが置換されます。

O

検索基準を OR にするよう指示します。このパラメーターはオプションです。このパラメーターを指定しない場合、検索基準は AND になります。

アクション:

OnDemand は、現行検索基準を使用してアクティブ・フォルダーを検索します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SEARCH_FOLDER", "", NULL );
```

SELECT_DOC

コマンド	パラメーター
SELECT_DOC	/N *doc number*

パラメーター:

N

アクティブ・フォルダーの文書リスト内の 0 を基準とした相対文書番号を指定します。リスト内の文書の数は、GET_NUM_DOCS_IN_LIST コマンドを使用して判別できます (153 ページの『GET_NUM_DOCS_IN_LIST』を参照)。GET_DOC_VALUES コマンド (151 ページの『GET_DOC_VALUES』を参照) を使用すると、特定の文書番号に関連した値を取り出すことができます。

このパラメーターは必須です。

doc number に -1 を指定すると、すべての文書が選択されるよう指示することができます。

アクション:

OnDemand は、特定の文書番号に対応する「文書リスト」行を選択 (強調表示) します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SELECT_DOC", "-1", NULL );
```

SET_FIELD_DATA

コマンド	パラメーター
SET_FIELD_DATA	/F *field name* /O *operator* /1 *value1* /2 *value2*

パラメーター:

F

アクティブ・フォルダー内で検索フィールドの名前を指定します。このパラメーターは必須です。

GET_QUERY_FIELDS コマンド (155 ページの『GET_QUERY_FIELDS』を参照) を使用すると、フィールド名のリストを取り出すことができます。

O

フィールドに使用する検索演算子を指定します。この演算子は、以下の値の 1 つにする必要があります。

EQ 「Equal」を表します。
NE 「Not Equal」を表します。
LT 「Less Than」を表します。
LE 「Less Than or Equal」を表します。
GT 「Greater Than」を表します。
GE 「Greater Than or Equal」を表します。
BW 「Between」を表します。
NB 「Not Between」を表します。
IN 「In」を表します。
NI 「Not In」を表します。
LK 「Like」を表します。

NL　「Not Like」を表します。

演算子には、フィールドに許可される演算子のうちの 1 つを使用する必要があります。

このパラメーターはオプションです。このパラメーターを指定しない場合、演算子は変更されないままになります。

1

フィールドに対する最初の、そして多くの場合唯一の入力ウィンドウに使用される値を指定します。このパラメーターはオプションです。このパラメーターを指定しない場合、その値は変更されないままになります。

2

フィールドの 2 番目の入力ウィンドウに使用される値を指定します。フィールドの検索演算子が「Between」または「Not Between」以外であれば、この値は無視されます。このパラメーターはオプションです。このパラメーターを指定しない場合、その値は変更されないままになります。

アクション:

OnDemand は、アクティブ・フォルダー内で、指定した検索フィールドの検索演算子、最初の入力ウィンドウ、2 番目の入力ウィンドウのいずれかまたはこれらを合わせて更新できます。

戻りコード:

0　ARS_DDE_RC_NO_ERROR
2　ARS_DDE_RC_PARM_NOT_SPECIFIED
3　ARS_DDE_RC_INVALID_PARM_VALUE
8　ARS_DDE_RC_FOLDER_NOT_OPEN
11　ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例:　DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
char parms[200];

sprintf( parms,
         "/F %s /0 %s /1 %s /2 %s",
         "Account",
         "BW",
         "123456",
         "987654" );

DoDdeCommand( "SET_FIELD_DATA", parms, NULL );
```

SET_FOCUS

コマンド	パラメーター
SET_FOCUS	なし

パラメーター:

なし

アクション:

OnDemand がアクティブ・ウィンドウになります。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SET_FOCUS", "", NULL );
```

SET_HELP_PATH

コマンド	パラメーター
SET_HELP_PATH	/P *path*

パラメーター:

P

Windows ヘルプ・ファイルの完全修飾パスを指定します。

このパラメーターは必須です。

アクション:

クライアント・アプリケーションに関連したボタンまたはメニュー項目の 1 つについてユーザーがヘルプを要求したとき、OnDemand は、指定したヘルプ・ファイルを呼び出します。

ヘルプ・ファイルは、以下のコンテキスト ID の 1 つによって呼び出されます。

「検索基準と文書リスト」ダイアログ・ボックス:

0x50800

ARS_DDE_HELP_ID_CRITERIA_BUTTON_1

0x50801

ARS_DDE_HELP_ID_CRITERIA_BUTTON_2

0x50802

ARS_DDE_HELP_ID_CRITERIA_BUTTON_3

0x50803

ARS_DDE_HELP_ID_CRITERIA_BUTTON_4

0x50804

ARS_DDE_HELP_ID_CRITERIA_BUTTON_5

0x50805

ARS_DDE_HELP_ID_DOCLIST_BUTTON_1

0x50806
ARS_DDE_HELP_ID_DOCLIST_BUTTON_2

0x50807
ARS_DDE_HELP_ID_DOCLIST_BUTTON_3

0x50808
ARS_DDE_HELP_ID_DOCLIST_BUTTON_4

0x50809
ARS_DDE_HELP_ID_DOCLIST_BUTTON_5

ツールバーおよびメニュー項目:

0x5080A
ARS_DDE_HELP_ID_SWITCH_FOCUS

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:
なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SET_HELP_PARM", "C:¥DDEAPPL¥DDEAPPL.HLP", NULL);
```

SET_USER_MSG_MODE

コマンド	パラメーター
SET_USER_MSG_MODE	/M *mode*

パラメーター:

M

ユーザー・メッセージ・モードを指定します。

mode には、以下の値の 1 つを指定してください。

1 OnDemand は、DDE コマンドに起因するユーザー・メッセージを常に表示します。

2 OnDemand は、OnDemand ウィンドウが表示されていて最小化されてはいない場合にだけ、DDE コマンドに起因するユーザー・メッセージを表示します。

3 OnDemand は、DDE コマンドに起因するユーザー・メッセージをすべて抑止します。

このパラメーターは必須です。

アクション:

OnDemand は、パラメーターによる指定に従って、これ以降のユーザー・メッセージの表示または抑止を行います。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
3 ARS_DDE_RC_INVALID_PARM_VALUE

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SET_USER_MSG_MODE", "/M 2", NULL);
```

SHOW_WINDOW

コマンド	パラメーター
SHOW_WINDOW	/W *placement*

パラメーター:

W

OnDemand ウィンドウの表示状況および位置を指定します。このパラメーターは、ウィンドウの配置 (Window Placement)コマンド行パラメーターと同じ値を取ります。その値については、129 ページの『ウィンドウの配置 (Window Placement) - /W 配置』を参照してください。

値を指定しないでこのパラメーターを指定すると、ウィンドウは、それが直前に表示されていた位置に直前の寸法で表示されます。

このパラメーターは必須です。

アクション:

OnDemand は、パラメーター値による指定に従って、そのメイン・ウィンドウの表示または配置あるいはこの両方を実行します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS

リターン・データ:

なし

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "SHOW_WINDOW", "/W 25,0,75,100", NULL );
```

STORE_DOC

コマンド	パラメーター
STORE_DOC	/P *doc path* /G *appl group name* /A *appl name* /n *value* /n *value* ...

パラメーター:

P

OnDemand データベースに保管される文書データが入っているファイルの完全修飾パスを指定します。このパラメーターは必須です。

G

アクティブ・フォルダー内のアプリケーション・グループの名前を指定します。アクティブ・フォルダーに関連したアプリケーション・グループ名は、呼び出し元の責任で知る必要があります。このパラメーターは必須です。

A

指定したアプリケーション・グループ内の「アプリケーション」の名前を指定します。指定したアプリケーション・グループに関連したアプリケーション名は、呼び出し元の責任で知る必要があります。このパラメーターは必須です。

n

フォルダー・フィールドに関連付けられる値を指定します。n は、フォルダーの最初のフィールドについては X'01'、2 番目のフィールドについては X'02' などという具合です。関連付けられる *value* は、フィールド・タイプのデータ (整数や日付など) に変換できる文字ストリングです。

フォルダー・フィールドの数と順序は、GET_FOLDER_FIELDS コマンドを使用して判別できます。このコマンドの詳細については、152 ページの『GET_FOLDER_FIELDS』を参照してください。

値を指定しないフォルダー・フィールドには、ストリング・フィールドについては空ストリング、数値フィールドについてはゼロが入ります。無関係なフィールドを指定すると、そのフィールドは無視されます。

日付フィールドには、このフィールドに必須の形式で値を指定する必要があります (例えば、February 3, 1996 とする必要があるときに 02/03/96 と指定すると無効になります)。

スラッシュ文字を日付などの値の中に含めるときには、2 個のスラッシュを連続して指定する必要があります。詳しくは、127 ページの『パラメーター構文』を参照してください。

アクション:

OnDemand は、フォルダー・フィールド値をアプリケーション・グループ・フィールドに変換し、指定したアプリケーション・グループおよびアプリケーションに関連した文書として、指定したファイルのデータをデータベースに保管します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR
2 ARS_DDE_RC_PARM_NOT_SPECIFIED
4 ARS_DDE_RC_SERVER_ERROR
5 ARS_DDE_RC_FILE_ERROR
8 ARS_DDE_RC_FOLDER_NOT_OPEN
11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS
12 ARS_DDE_RC_UNAUTHORIZED_OPERATION

15 ARS_DDE_RC_INVALID_APPL_GROUP_NAME
16 ARS_DDE_RC_INVALID_APPL_NAME
17 ARS_DDE_RC_INVALID_INTEGER_FIELD
18 ARS_DDE_RC_INVALID_DECIMAL_FIELD
19 ARS_DDE_RC_INVALID_DATE_FIELD
20 ARS_DDE_RC_INVALID_APPLGRP_FIELD_TYPE
27 ARS_DDE_RC_TOO_MANY_VALUE_CHARS

リターン・データ:

戻りコードが以下のいずれかである場合、

ARS_DDE_RC_INVALID_INTEGER_FIELD
ARS_DDE_RC_INVALID_DECIMAL_FIELD
ARS_DDE_RC_INVALID_DATE_FIELD
ARS_DDE_RC_INVALID_APPLGRP_FIELD_TYPE
ARS_DDE_RC_TOO_MANY_VALUE_CHARS

OnDemand は、無効フィールドの相対フォルダー・フィールド番号を戻します。例えば、最初のフォルダー・フィールドが無効のときには、OnDemand は 0 を戻し、2 番目のフォルダー・フィールドが無効のときには、OnDemand は 1 を戻すという具合です。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

C/C + + の例

```
char parms[200];

sprintf( parms,
         "/D %s /G %s /A %s /¥x'01' %s /¥x'02' %s /¥x'03' %s",
         "D:¥DATA¥DOCDATA.AFP",
         "Student Data",
         "Grades",
         "Coed, Mary",
         "05//23//95",
         "3.15" );

DoDdeCommand( "STORE_DOC", parms, NULL );
```

Visual Basic の例

```
Dim cmdline As String
cmdline = "STORE_DOC /P D:¥Data¥DocData.AFP "
cmdline = cmdline + "/G Student Data "
cmdline = cmdline + "/A Grades "
cmdline = cmdline + "/" Chr(1) + " Coed.Mary "
cmdline = cmdline + "/" Chr(2) + " 05//23//95 "
cmdline = cmdline + "/" Chr(3) + " 3.15"

Call fncDDElink ( arstopic, cmdline, linktype, 3000 )
```

注: fncDDElink の定義については、247 ページの『付録 A. Microsoft Visual Basic 5.0 DDE サンプル・プログラム』を参照してください。

UPDATE_DOC

コマンド	パラメーター
UPDATE_DOC	/N *doc number* /F *field name* /V *field value*

パラメーター:

N

アクティブ・フォルダーの文書リスト内の、0 を基準とした相対文書番号を指定します。このパラメーターは必須です。

doc number に -1 を指定すると、選択したすべての文書を更新するよう指示することができます。

F

フォルダー・フィールドの名前を指定します。このパラメーターは必須です。

V

指定したフォルダー・フィールドに保管される値を指定します。この値は、フィールド・タイプのデータ (整数や日付など) に変換されます。

日付フィールドには、このフィールドに必須の形式で値を指定する必要があります (例えば、February 3, 1996 とする必要があるときに 02/03/96 と指定すると無効になります)。

スラッシュ (/) 文字を値 (例えば、日付) の中に含めるときには、2 個のスラッシュを連続して指定する必要があります。詳しくは、127 ページの『パラメーター構文』を参照してください。

このパラメーターは必須です。

アクション:

OnDemand は、フォルダー・フィールド値をアプリケーション・グループ・フィールドに変換し、データベース内の指定した値からデータを更新します。

戻りコード:

0 ARS_DDE_RC_NO_ERROR

3 ARS_DDE_RC_INVALID_PARM_VALUE

4 ARS_DDE_RC_SERVER_ERROR

8 ARS_DDE_RC_FOLDER_NOT_OPEN

12 ARS_DDE_RC_UNAUTHORIZED_OPERATION

20 ARS_DDE_RC_INVALID_APPLGRP_FIELD_TYPE

リターン・データ:

OnDemand は、正常に更新された文書の番号を戻します。戻されたヌル終了ストリングは、長整数に変換できます。

例: DoDdeCommand 関数については、136 ページの『DDEML トランザクション』を参照してください。

```
DoDdeCommand( "UPDATE_DOC", "/N 1 /F Balance /V 123.45", NULL );
```

第 8 章 戻りコード

DDE コマンドから戻る可能性のある戻りコードを以下に示します。

0 ARS_DDE_RC_NO_ERROR - エラーを発生することなくコマンドが実行されたことを示します。

1 ARS_DDE_RC_UNKNOWN_COMMAND - コマンドが無効な OnDemand DDE コマンドであったことを示します。

2 ARS_DDE_RC_PARM_NOT_SPECIFIED - 必須パラメーターを指定しなかったことを示します。

3 ARS_DDE_RC_INVALID_PARM_VALUE - パラメーターに無効値を指定したことを示します。

4 ARS_DDE_RC_SERVER_ERROR - データベースへのアクセス中にサーバー・エラーが発生したことを示します。

5 ARS_DDE_RC_FILE_ERROR - 入出力操作中にエラーが発生したことを示します。

6 ARS_DDE_RC_NOT_LOGGED_ON - コマンドを実行するためにはユーザーがログオンする必要があることを示します。

7 ARS_DDE_RC_MAX_FOLDERS_OPEN - 最大数のフォルダーがすでに開いていることを示します。

8 ARS_DDE_RC_FOLDER_NOT_OPEN - コマンドを実行するためにはフォルダーが開いてアクティブになっていなければならないことを示します。

9 ARS_DDE_RC_NO_DOC - 文書リスト内の文書に関連したデータベース・データがないことを示します。

11 ARS_DDE_RC_USER_ACTION_IN_PROGRESS - ユーザーが開始したアクションの実行のために OnDemand は使用中であることを示します。モーダル・ダイアログ・ボックスまたはメッセージ・ボックスが表示されている可能性があります。

12 ARS_DDE_RC_UNAUTHORIZED_OPERATION - 現在ログオンしているユーザーは要求した操作の実行を許可されていないことを示します。

13 ARS_DDE_RC_USER_CANCELLED_OPERATION - DDE によって要求した操作をユーザーが取り消したことを示します。

14 ARS_DDE_RC_NO_ACTIVE_DOC - 現在アクティブな文書はないことを示します。

15 ARS_DDE_RC_INVALID_APPL_GROUP_NAME - 指定したアプリケーション・グループ名がフォルダーにとって無効であることを示します。

16 ARS_DDE_RC_INVALID_APPL_NAME - 指定したアプリケーション名がアプリケーション・グループにとって無効であることを示します。

17 ARS_DDE_RC_INVALID_INTEGER_FIELD - フォルダー・フィールドに指定した値が無効な整数であることを示します。

18 ARS_DDE_RC_INVALID_DECIMAL_FIELD - フォルダー・フィールドに指定した値が無効な 10 進数であることを示します。

19 ARS_DDE_RC_INVALID_DATE_FIELD - フォルダー・フィールドに指定した値が無効な日付であることを示します。

20 ARS_DDE_RC_INVALID_APPLGRP_FIELD_TYPE - アプリケーション・グループ内のフィールド・タイプが無効であることを示します。

21 ARS_DDE_RC_DOC_NOT_VIEWABLE_OR_PRINTABLE - 指定した文書の表示または印刷が DDE インターフェースによって行えないことを示します。

22 ARS_DDE_RC_INCORRECT_CURRENT_PASSWORD - ユーザーについて指定した現行パスワードが正しくないことを示します。

23 ARS_DDE_RC_PASSWORD_TOO_SHORT - 指定した新規パスワードが短すぎることを示します。

24 ARS_DDE_RC_NEW_PASSWORD_MISMATCH - 新規パスワードに指定した 2 つのパスワードが一致していないことを示します。

25 ARS_DDE_RC_INVALID_USER_PASS_SERVER - ユーザー ID、パスワード、またはサーバーが無効であったことを示します。

26 ARS_DDE_RC_PASSWORD_EXPIRED - パスワードの有効期限が切れていることを示します。

第 9 章 DDEML Advise Loop

クライアント・アプリケーションは、以下のイベントの 1 つが発生したときに通知を受けるために、DDEML Advise Loop を作成できます。

コード イベント

S ユーザーは、最初のメニュー項目またはツールバー・ボタンを使用してフォーカスをクライアント・アプリケーションに切り替えました。

S2 ユーザーは、2 番目のメニュー項目またはツールバー・ボタンを使用してフォーカスをクライアント・アプリケーションに切り替えました。

S3 ユーザーは、3 番目のメニュー項目またはツールバー・ボタンを使用してフォーカスをクライアント・アプリケーションに切り替えました。

S4 ユーザーは、4 番目のメニュー項目またはツールバー・ボタンを使用してフォーカスをクライアント・アプリケーションに切り替えました。

S5 ユーザーは、5 番目のメニュー項目またはツールバー・ボタンを使用してフォーカスをクライアント・アプリケーションに切り替えました。

0 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「検索基準」ボタン 1 をクリックしました。

1 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「検索基準」ボタン 2 をクリックしました。

2 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「検索基準」ボタン 3 をクリックしました。

3 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「検索基準」ボタン 4 をクリックしました。

4 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「検索基準」ボタン 5 をクリックしました。

5 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「文書リスト」ボタン 5 をクリックしました。

6 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「文書リスト」ボタン 1 をクリックしました。

7 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「文書リスト」ボタン 2 をクリックしました。

8 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「文書リスト」ボタン 3 をクリックしました。

9 ユーザーは、「検索基準と文書リスト」ダイアログ・ボックスの「文書リスト」ボタン 4 をクリックしました。

DDEML DdeClientTransaction 関数を ADV_START トランザクションと ADV_STOP トランザクションで使用して Advise Loop を開始したり停止したりします。DDEML 項目名ストリングには「1」が入っていなければなりません。クライアント・アプリケーションは、OnDemand との接続を確立したあとで Advise Loop

を開始し、会話の継続中このループを保持することができ、また、このループの開始および停止を自由に行うことができます。イベントの通知は、ループがアクティブなときにだけ行われます。

クライアント・アプリケーションは、その DDEML コールバック関数で XTYP_ADVDATA トランザクションを介してイベントの通知を受け取ります。戻されるデータは、**イベント**の**コード**が入っているヌル終了文字ストリングです。

第 10 章 外部アプリケーションとダイナミック・リンク・ライブラリー

Windows クライアントの場合、OnDemand は、エンド・ユーザーが別の Windows アプリケーションを呼び出したり、ダイナミック・リンク・ライブラリー (DLL) の関数を実行することができるメニューおよびツールバー拡張機能を提供します。Windows のシステム・レジストリーに情報を入れることによって、この機能をアクティブにすることができます。 OnDemand は、初期設定時にこの情報を検出すると、メニュー項目とツールバー・ボタンを追加します。ユーザーがそのメニュー項目の 1 つを選択するか、またはそのツールバー・ボタンの 1 つをクリックすると、OnDemand は、関連したアプリケーションを呼び出すかまたは DLL 内の関連したエントリー・ポイントを呼び出します。

OnDemand は、Windows のシステム・レジストリーの中の HKEY_CURRENT_USER¥Software¥IBM¥OnDemand32¥Client (または HKEY_LOCAL_MACHINE¥Software¥IBM¥OnDemand32¥Client) キーを調べて、以下のキーとストリング値を求めます。

表 4. レジストリー内の外部アプリケーションとダイナミック・リンク・ライブラリーのキー

キー	値の名前	値データ
ExternalApps	Apps	*a1*[,*a2*][,*a3*][,*a4*][,*a5*]
a1	Path	アプリケーション・パス
	MenuText	メニュー項目テキスト
	BitmapDLL	ツールバー・ボタンのビットマップ DLL パス
	BitmapResid	ツールバー・ボタンのビットマップのリソース ID
	Folders	フォルダー名[¥folder name]...
	ExcludeFolders	フォルダー名[¥folder name]...
	Doc	**0** | **1** | **2** | **3**
	CopyDoc	**ASIS** | **ASCII**
	Parameter	パラメーター・データ
ExternalDlls	Dlls	*d1*[,*d2*][,*d3*][,*d4*][,*d5*]

表 4. レジストリー内の外部アプリケーションとダイナミック・リンク・ライブラリーのキー (続き)

キー	値の名前	値データ
d1	Path	DLL パス
	Function	機能名
	MenuText	メニュー項目テキスト
	BitmapDll	ツールバー・ボタンのビットマップ DLL パス
	BitmapResid	ツールバー・ボタンのビットマップのリソース ID
	Folders	フォルダー名[¥folder name]...
	ExcludeFolders	フォルダー名[¥folder name]...
	Doc	**0** | **1** | **2** | **3**
	CopyDoc	**ASIS** | **ASCII**
	Parameter	パラメーター・データ

最高で 5 つまでのアプリケーションと 5 つまでの DLL を指定できます。これらは、HKEY_CURRENT_USER キーと HKEY_LOCAL_MACHINE キーに分かれます。キー名は、任意のストリングにすることができます。その値 (ストリング値でなければならない) の解釈を以下に示します。

- Path は、アプリケーションまたは DLL のパスを指定します。この値は必須指定であり、完全修飾パスにするか実行パスに入っている必要があります。
- Function は、Path で指定した DLL 内のエントリー・ポイントの名前を指定します。この値は、DLL については必須指定です。アプリケーションについては、この値は関係ありません。
- MenuText は、「OnDemand ウィンドウ (OnDemand Window)」メニューの関連メニュー項目に表示されるテキストを指定します。この値はオプションです。この値を指定しない場合、メニュー項目はブランクになります。
- BitmapDLL は、ツールバー・ボタンをアプリケーションまたは DLL に関連付けるために使用されるビットマップ・リソースを含む DLL のパスを指定します。この DLL は、Path または別の BitmapDLL に指定された DLL と同じである場合も異なっている場合もあります。ビットマップは、幅 16 ピクセル、高さ 16 ピクセルでなければなりません。

  この値はオプションです。この値を指定しない場合、ツールバー・ボタンは作成されません。
- BitmapResid は、BitmapDLL に指定した DLL に入っているビットマップのリソース ID を指定します。BitmapDLL を指定しない場合には、この値は無視され、指定した場合には、この値はオプションになります。この値を指定しないと、デフォルトで値 0 になります。
- Folders は、OnDemand フォルダーの名前を 1 つまたは複数指定します。複数の名前を指定する場合、円記号 (「¥」) 文字でこれらの名前を区切る必要があります。名前の末尾にアスタリスク (「*」) をワイルドカード文字として使用できます。こうすると、アスタリスクより前の部分の文字が一致するすべてのフォルダー名がリストされることになります。

関連したメニュー項目、およびこれに対応するツールバー・ボタンは、以下の場合に常に使用可能になります。1) 文書が表示されているときに、アクティブ文書に関連したフォルダーが指定したフォルダーの 1 つである場合。または 2) 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つである場合。

この値はオプションです。ExcludeFolders 値を指定した場合、この値は無視されます。両方とも指定しないと、メニュー項目とツールバー・ボタンを使用可能にする前にフォルダー名のテストが実行されません。

- ExcludeFolders は、OnDemand フォルダーの名前を 1 つまたは複数指定します。その構文は、Folders 値と同じです。

  関連したメニュー項目、およびこれに対応するツールバー・ボタンは、以下の場合に常に使用可能になります。1) 文書が表示されているときに、アクティブ文書に関連したフォルダーが指定したフォルダーの 1 つ**ではない**場合。または 2) 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つ**ではない**場合。

  この値はオプションです。この値を指定しないと、Folders 値によって使用可能化が制御されます。

- Doc には、以下の値の 1 つを指定できます。

  0 は、関連したメニュー項目、およびこれに対応するツールバー・ボタンの使用可能化が Folders 値および ExcludeFolders 値によってだけ制限されるよう指示します。

  1 は、文書が表示されているときにだけ、関連したメニュー項目、およびこれに対応するツールバー・ボタンが使用可能になるよう指示します。

  2 は、文書リストが表示されていて文書が少なくとも 1 つは選択されているときにだけ、関連したメニュー項目、およびこれに対応するツールバー・ボタンが使用可能になるよう指示します。

  3 は、文書が表示されているとき、または文書リストが表示されていて文書が少なくとも 1 つは選択されているときにだけ、関連したメニュー項目、およびこれに対応するツールバー・ボタンが使用可能になるよう指示します。

  この値はオプションです。この値を指定しないと、デフォルトで値 3 になります。

- CopyDoc は、1 つまたは複数の文書のコピーを外部アプリケーションあるいは DLL に提供するよう指示し、提供するデータのタイプを指定します。その値が ASIS の場合は、文書データは元のフォーマットのままです。その値が ASCII の場合は、文書データは ASCII ファイルに変換されます。

  文書が表示されているときに、エンド・ユーザーが関連したメニュー項目またはこれに対応するツールバー・ボタンを選択した場合、提供されるデータは、アクティブ文書のデータです。フォルダー文書リストが表示されている場合、そのデータは、文書リストで選択されたすべての文書を連結したものになります。

  この値はオプションです。この値を指定しないと、外部アプリケーションまたは DLL に文書は提供されません。

- Parameter は、パラメーター・データとして外部アプリケーションまたは DLL に渡される最高 255 までの文字を指定します。この値はオプションです。この値を指定しないと、外部アプリケーションまたは DLL にパラメーター・データは提供されません。

以下の 3 つの画面は、外部アプリケーションの必要なレジストリー項目を示しています。

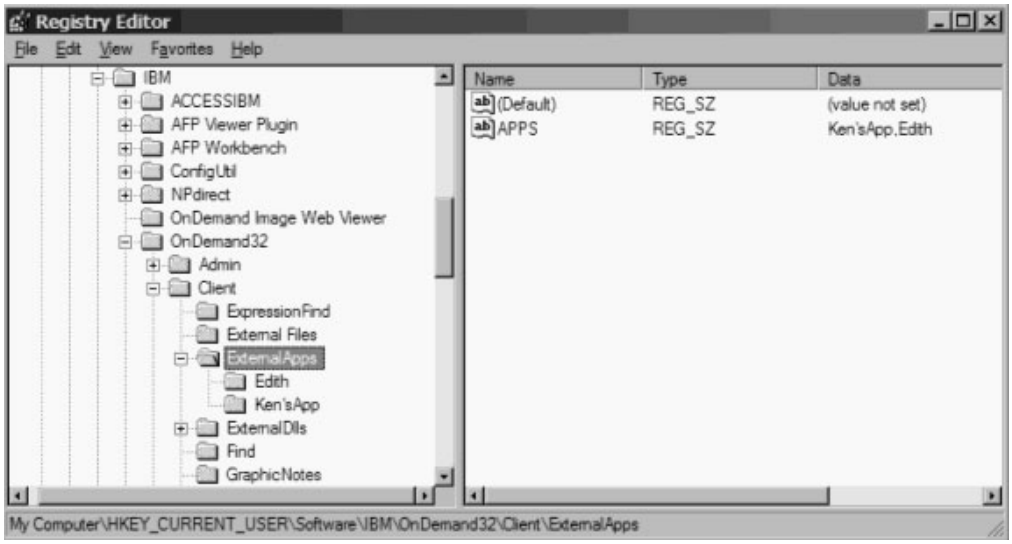


図 1. 「レジストリー (Registry)」部分の外部アプリケーションの例 1/3

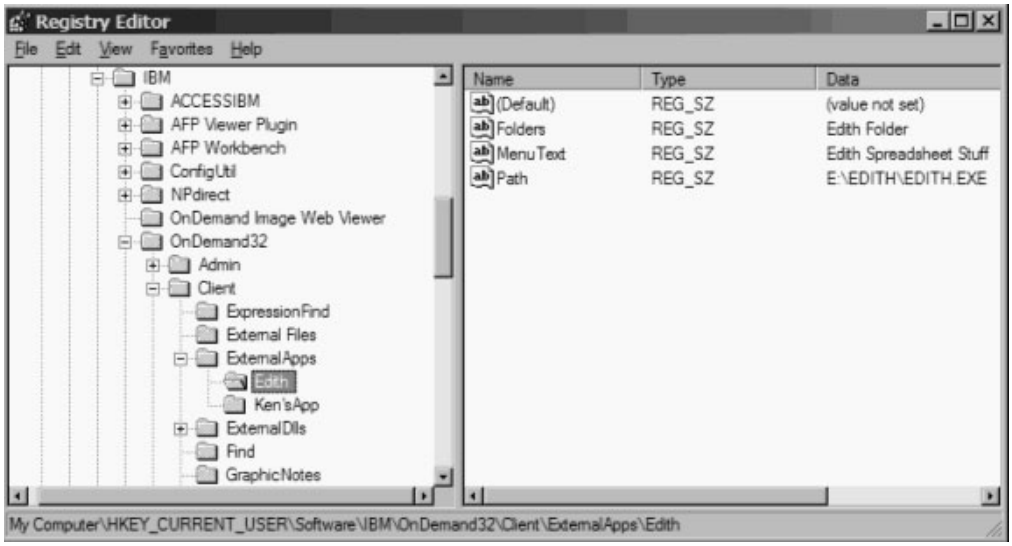


図 2. 「レジストリー (Registry)」部分の外部アプリケーションの例 2/3

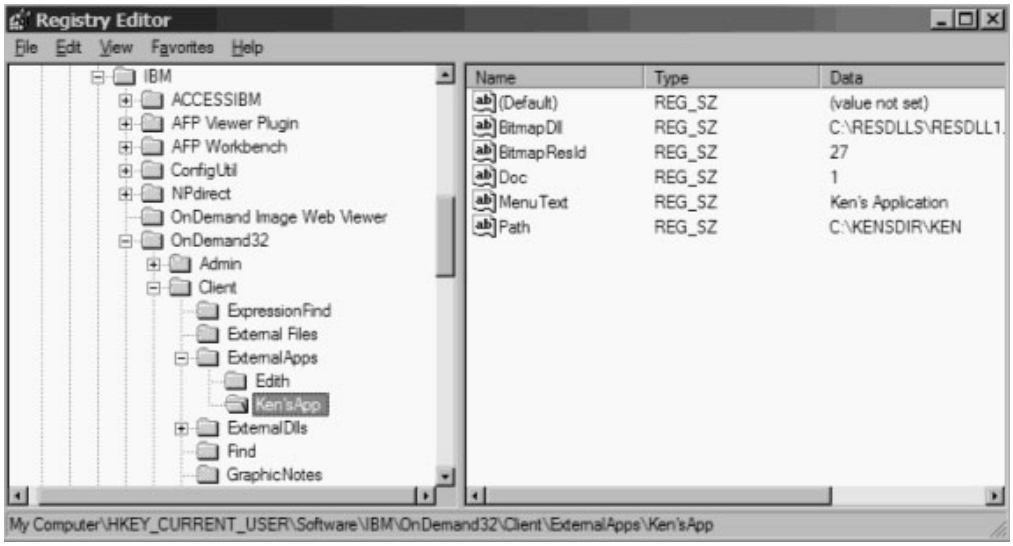


図 3. 「レジストリー (Registry)」部分の外部アプリケーションの例 3/3

以下の 2 つの画面は、外部 DLL の必要なレジストリー項目を示しています。

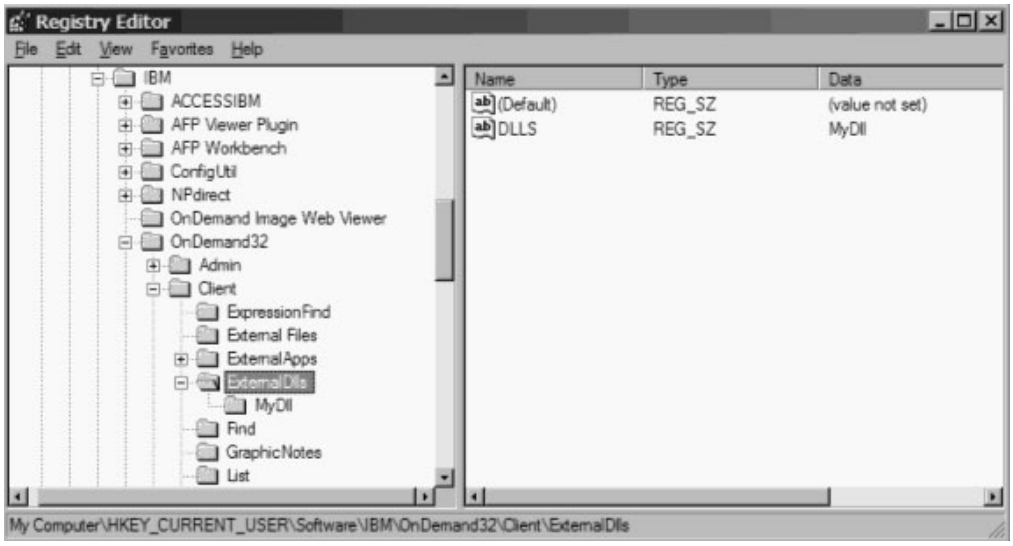


図 4. 「レジストリー (Registry)」部分の外部 DLL の例 1/2

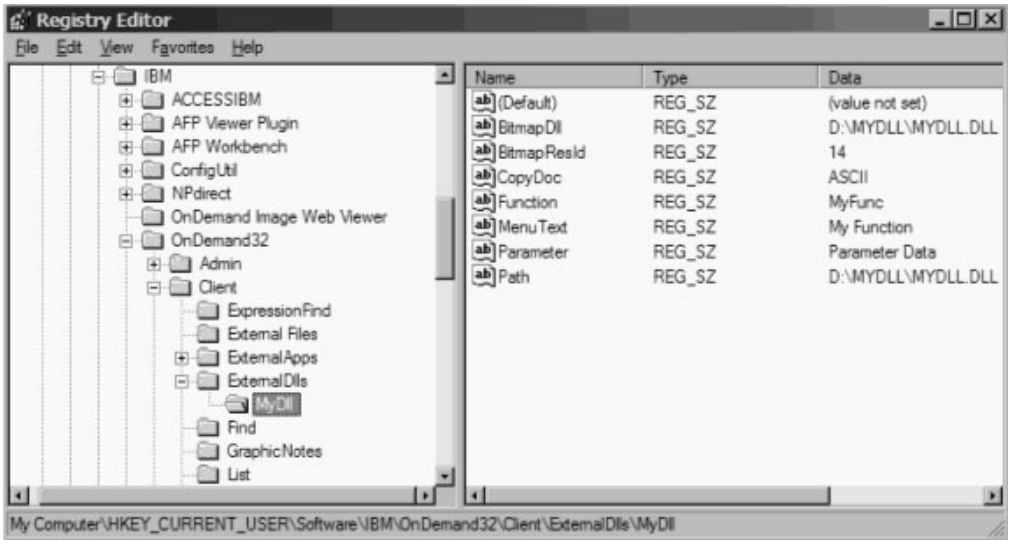


図 5. 「レジストリー (Registry)」部分の外部 DLL の例 2/2

ユーザーがメニュー項目を選択するか、またはアプリケーションに関連したツールバー・ボタンをクリックすると、OnDemand は、次のコマンド行でアプリケーションを呼び出します。

```
application path /F folder /P page /T text /A attrvalue /D fields /N filename /R data
```

この場合:

- application path は、レジストリー内の Path 値で指定したパスです。
- folder は、アクティブ・フォルダーの名前です。または、文書が表示されている場合は、その文書に関連したフォルダーです。フォルダーが開いていない場合、/F パラメーターは提供されません。
- page は、表示中の文書の現行ページです。文書が表示されていないか、アクティブ・フォルダー・ウィンドウが表示されている場合、/P パラメーターは提供されません。
- text は、表示中の文書内で現在選択されているテキスト、またはアクティブ・フォルダーの文書リストです。テキストのフォーマットは、それをクリップボードにコピーすることで得られるフォーマットと同じです。テキストには、場合によってタブおよび改行文字が含まれます。そのようなテキストが存在しない場合、/T パラメーターは提供されません。
- attrvalue は、表示中の文書の現行ページに関連した属性と値の対です。その属性と値は、タブ文字で区切られます。そのような属性と値の対が存在しない場合、/A パラメーターは提供されません。
- フィールドは、表示中の文書に関連したフォルダー・フィールド名と値の一連の対です。各対には、フォルダー・フィールドの名前、および文書のそのフィールドの値がタブ文字で区切られて含まれます。対同士は改行文字で区切られます。そのようなテキストが存在しない場合、/T パラメーターは提供されません。
- ファイル名は、CopyDoc レジストリー項目に関して記述された文書データの完全修飾パスおよびファイル名です。CopyDoc 項目を指定しないかデータが使用可能でない場合、/N パラメーターは提供されません。
- データは、Parameter レジストリー項目に指定されたパラメーター・データです。Parameter 項目を指定しない場合、/R パラメーターは提供されません。

ユーザーがメニュー項目を選択するか、または DLL に関連したツールバー・ボタンをクリックすると、OnDemand は、レジストリー内の Function 値で指定されたエントリー・ポイント (関数) を呼び出します。この関数の型は、以下のいずれかでなければなりません。

```
typedef void ( WINAPI * ArsExternalDllFunction )
                          ( long   page_number,
                            char * pFolderName,
                            char * pSelectedText,
                            char * pAttributeAndValue,
                            char * pFieldsAndValues,
                            char * pFilename,
                            char * pParameterData );
```

この場合:

- page_number は、表示中の文書の現行ページです。文書が表示されていないか、アクティブ・フォルダー・ウィンドウが表示されている場合、この値は 0 です。
- pFolderName は、アクティブ・フォルダーの名前、または文書が表示中の場合はその文書に関連したフォルダーの名前を含むヌル終了文字ストリングを指すポインターです。フォルダーが開いていない場合、この値は NULL です。
- pSelectedText は、表示中の文書内で現在選択されているテキストまたはアクティブ・フォルダーの文書リストを含むヌル終了文字ストリングを指すポインターです。テキストのフォーマットは、それをクリップボードにコピーすることで得られるフォーマットと同じです。テキストには、場合によってタブおよび改行文字が含まれます。そのようなテキストが存在しない場合、この値は NULL です。
- pAttributeAndValue は、表示中の文書の現行ページに関連した属性と値の対を含むヌル終了文字ストリングを指すポインターです。その属性と値は、タブ文字で区切られます。そのような属性と値の対が存在しない場合、この値は NULL です。
- pFieldsAndValues は、表示中の文書に関連したフォルダー・フィールド名と値の一連の対を含むヌル終了文字ストリングを指すポインターです。各対には、フォルダー・フィールドの名前、および文書のそのフィールドの値がタブ文字で区切られて含まれます。対同士は改行文字で区切られます。そのようなテキストが存在しない場合、この値は NULL です。
- pFilename は、CopyDoc レジストリー項目に関して記述された文書データの完全修飾パスおよびファイル名を含むヌル終了文字ストリングを指すポインターです。CopyDoc 項目を指定しないかデータが使用可能でない場合、この値は NULL です。
- pParameterData は、Parameter レジストリー項目に指定されたパラメーター・データを含むヌル終了文字ストリングを指すポインターです。Parameter 項目を指定しないと、この値は、NULL になる場合も空ストリングを指す場合もあります。

第 11 章 関連文書

Windows クライアントの場合、OnDemand は、エンド・ユーザーが現在表示中の文書に関連した文書の取り出しおよび表示を行うことができるメニューおよびツールバー拡張機能を提供します。 Windows のシステム・レジストリーに情報を入れることによって、この機能をアクティブにすることができます。 OnDemand は、初期設定時にこの情報を検出すると、メニュー項目とツールバー・ボタンを追加します。ユーザーがそのメニュー項目の 1 つを選択するか、そのツールバー・ボタンの 1 つをクリックすると、OnDemand は、関連した文書を取り出して元の文書と一緒に表示します。

関連文書の代表的なアプリケーションとしては、月次明細書を含むクレジット・カード・フォルダーがあります。エンド・ユーザーは、クレジット・カード・トランザクションごとに 1 行ずつあるカスタマーの明細書を表示し、これらの行のうちのいずれかのトランザクション番号を選択してからツールバー・ボタンをクリックします。OnDemand は、関連したトランザクション・フォルダーを検索してカスタマーのアカウント番号および選択されたトランザクション番号を求めたあと、トランザクション文書を明細書と並べて表示します。

OnDemand は、Windows のシステム・レジストリーの中の HKEY_CURRENT_USER¥Software¥IBM¥OnDemand32¥Client (または HKEY_LOCAL_MACHINE¥Software¥IBM¥OnDemand32¥Client) キーを調べて、以下のキーとストリング値を求めます。

表 5. レジストリー内の関連文書のキー

キー	値の名前	値データ
RelatedDocs	Related	*r1*[, *r2*][, *r3*][, *r4*][, *r5*][, *r6*][, *r7*][, *r8*][, *r9*][, *r10*]
r1	MenuText	メニュー項目テキスト
	BitmapDLL	ツールバー・ボタンのビットマップ DLL パス
	BitmapResid	ツールバー・ボタンのビットマップのリソース ID
	Folders	フォルダー名[¥folder name]...
	ExcludeFolders	フォルダー名[¥folder name]...
	RelatedFolder	関連するフォルダー名
	Fields	フィールド情報
	Arrange	**V** | **H**| **M**| **C**| **U**

RelatedDocs キーには、1 つのストリング値が含まれます。ストリング名は、「Related」です。 Related ストリング値の値データは、関連文書を表す追加の Client キーをコンマで区切ったリストです。Related ストリング値に指定された各関連文書は、キーとして Client キーに追加する必要があります。最高 10 までの関連

文書を指定できます。これらは、HKEY_CURRENT_USER キーと HKEY_LOCAL_MACHINE キーに分かれます。キー名は、任意のストリングにすることができます。

その値 (ストリング値でなければならない) の解釈を以下に示します。

- MenuText は、OnDemand の「ウィンドウ」メニューの関連するメニュー項目に表示されるテキストを指定します。この値はオプションです。この値を指定しない場合、メニュー項目はブランクになります。
- BitmapDLL は、ツールバー・ボタンを関連文書に関連付けるために使用されるビットマップ・リソースを含む DLL のパスを指定します。ビットマップは、幅 16 ピクセル、高さ 16 ピクセルでなければなりません。

  この値はオプションです。この値を指定しない場合、ツールバー・ボタンは作成されません。
- BitmapResid は、BitmapDLL に指定した DLL に入っているビットマップのリソース ID を指定します。BitmapDLL を指定しない場合には、この値は無視され、指定した場合には、この値はオプションになります。この値を指定しないと、デフォルトで値 0 になります。
- Folders は、OnDemand フォルダーの名前を 1 つまたは複数指定します。複数の名前を指定する場合、円記号 (「¥」) 文字でこれらの名前を区切る必要があります。名前の末尾にアスタリスク (「*」) をワイルドカード文字として使用できます。こうすると、アスタリスクより前の部分の文字が一致するすべてのフォルダー名がリストされることになります。

  関連したメニュー項目、およびこれに対応するツールバー・ボタンは、以下の場合に常に使用可能になります。1) 文書が表示されているときに、アクティブ文書に関連したフォルダーが指定したフォルダーの 1 つである場合。または 2) 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つである場合。

  この値はオプションです。ExcludeFolders 値を指定した場合、この値は無視されます。両方とも指定しないと、メニュー項目とツールバー・ボタンを使用可能にする前にフォルダー名のテストが実行されません。
- ExcludeFolders は、OnDemand フォルダーの名前を 1 つまたは複数指定します。その構文は、Folders 値と同じです。

  関連したメニュー項目、およびこれに対応するツールバー・ボタンは、以下の場合に常に使用可能になります。1) 文書が表示されているときに、アクティブ文書に関連したフォルダーが指定したフォルダーの 1 つ**ではない**場合。または 2) 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つ**ではない**場合。

  この値はオプションです。この値を指定しないと、Folders 値によって使用可能化が制御されます。
- RelatedFolder は、表示中の文書に関連した文書を含む OnDemand フォルダーの名前を指定します。このフォルダーは、Folders によって指定したフォルダーと同じである場合も異なっている場合もあります。

- Fields は、関連フォルダーから取り出される文書を記述する情報を指定します。この情報は、フォルダーの検索を実行するために使用されます。フォルダー・フィールドは、まず、そのデフォルトの演算子と値に初期設定されます。ここに指定したフィールドは、その後、要求された演算子と値に設定されます。検索の結果取り出された文書リスト内の最初の文書が関連文書であると見なされます。

  その情報のフォーマットは、次のとおりです。

  ```
  fieldname=operator¥value1[¥value2]; ... ;fieldname=operator¥value1[¥value2]
  ```

  この場合:

  – fieldname は、関連フォルダーについて定義するフィールドの名前です。
  – operator は、フィールドに使用する検索演算子です。この演算子は、以下の値の 1 つにする必要があります。

    EQ 「Equal」を表します。
    NE 「Not Equal」を表します。
    GT 「Greater Than」を表します。
    GE 「Greater Than Or Equal」を表します。
    LT 「Less Than」を表します。
    LE 「Less Than Or Equal」を表します。
    BW 「Between」を表します。
    NB 「Not Between」を表します。
    IN 「In」を表します。
    NI 「Not In」を表します。
    LK 「Like」を表します。
    NL 「Not Like」を表します。

  – value1 は、フィールドに使用する最初の、または唯一の値です。これは、空ストリングも含めて任意のストリングにすることができ、以下の置換文字を含めることができます。

    %v 元のフォルダーに同じ名前のフィールドがある場合は、これらの文字は、元の文書のそのフィールドの値で置換されます。ない場合は、空ストリングで置換されます。

    %s これらの文字は、ユーザーが元の文書内で現在選択しているテキストで置換されます。

    %% これらの文字は、単一の % で置換されます。

  – value2 は、フィールドに使用する 2 番目の値です。演算子が「Between」または「Not Between」でない限り、この値は無視されます。そのフォーマットは、value1 と同じです。

- Arrange は、関連文書が表示されたあとの文書ウィンドウの配置について指定します。この値は、以下のいずれかを指定する必要があります。

  V 縦方向にタイル表示します。
  H 横方向にタイル表示します。
  M 関連文書を元の文書にオーバーレイして最大化します。
  C カスケード表示します。
  U 「文書の配置を維持する (Maintain Document Arrangement)」メニュー項目を介したユーザーの指定に任されます。

以下の 3 つの画面は、2 つの関連文書のサポートに必要なレジストリー項目の例を示します。

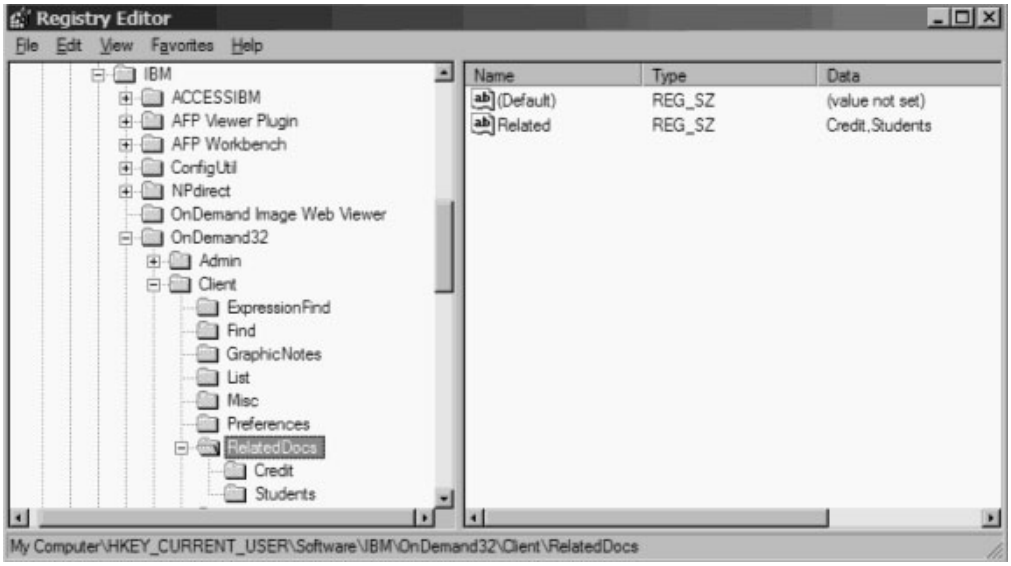


図 6. RelatedDocs キー、2 つの関連文書の値を示す

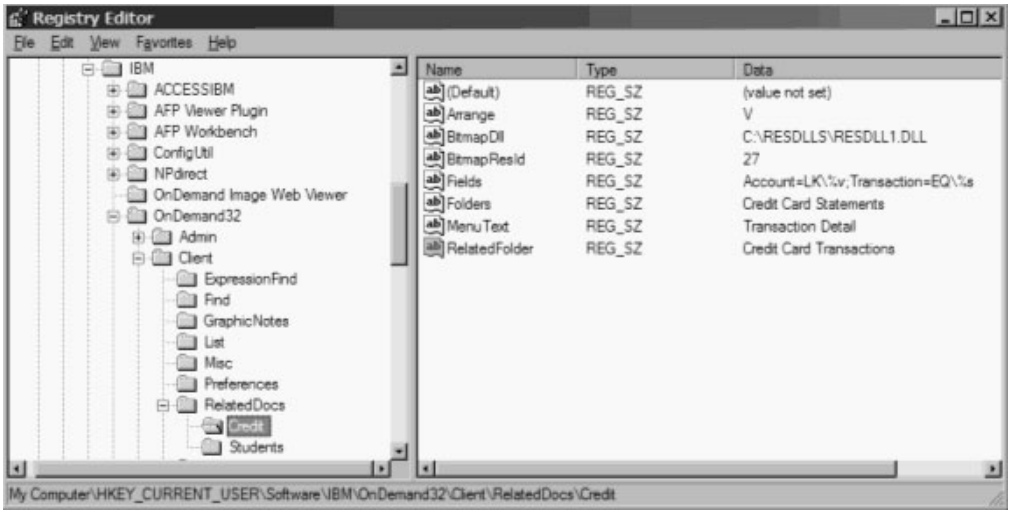


図 7. Credit キー、最初の関連文書の値を示す

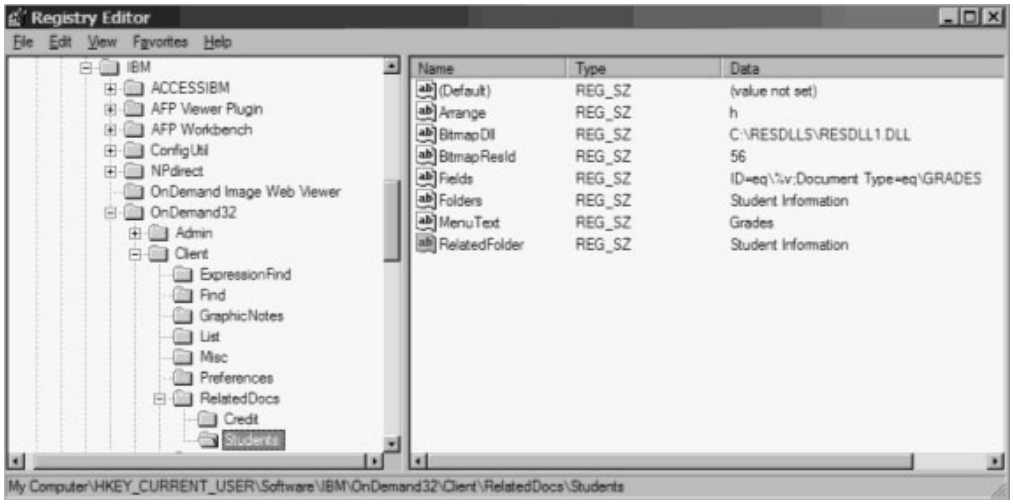


図 8. Students キー、2 番目の関連文書の値を示す

第 12 章 Program Information File

初期設定時、およびユーザーが「ヘルプ」メニューの「製品情報」項目を選択したときに表示される OnDemand アプリケーションの表題と「製品情報」ダイアログ・ボックスの外観をカスタマイズするために、Program Information File (PIF) を作成できます。

PIF は PRODUCT.INF という名前で、OnDemand が実行されるディレクトリーと同じディレクトリーに入っています。PIF のフォーマットと構文は、標準 Windows INI ファイルと同じです。 PIF には、PRODUCT というセクションが 1 つあります。このセクションには、以下の項目があります。

- NAME は、OnDemand のメイン・ウィンドウの最上部および多数のメニュー項目で使用される表題を指定します。
- LOGO_FILE は、「製品情報」ダイアログ・ボックスでロゴとして使用される Device Independent Bitmap (DIB) ファイルの完全修飾パスを指定します。
- ABOUT_TITLE は、「製品情報」ダイアログ・ボックスに使用される表題を指定します。
- ABOUT_LINEn は、「製品情報」ダイアログ・ボックスのテキストの各行を指定します。ABOUT_LINE1 は 1 行目、ABOUT_LINE2 は 2 行目を指定するという具合に ABOUT_LINE8 まで指定します。

第 13 章 Document Audit Facility

概要

OnDemand に保管された文書を監査するために、Document Audit Facility (DAF) を使用できます。DAF を使用するには、まず、制御ファイルを作成し、監査されるレポートを OnDemand に定義する必要があります。そのレポートをシステムにロードしたあと、Windows クライアントを使用して文書の監査を行うことができます。DAF 制御ファイルに定義したフォルダーから文書を取り出すとき、OnDemand は、2 つの追加コマンド・ボタンをクライアントの表示ウィンドウに表示します。これらのボタンの 1 つは、監査にパスする文書にマークを付けるために使用します。もう 1 つのボタンは、監査にパスしない文書にマークを付けるために使用します。

注: 文書を監査する必要のあるユーザーは、文書の更新許可を得ることが必要です。詳しくは、200 ページの『DAF へのアクセスの制御』を参照してください。

以下のトピックには、追加情報が記載されています。

- DAF 制御ファイルの作成
- レポートの定義
- DAF へのアクセスの制御
- DAF の使用

DAF 制御ファイルの作成

DAF は、ARSGUI.CFG というファイルによって制御されます。このファイルを作成し、Windows クライアント・プログラム・ディレクトリー (デフォルトでは ¥Program Files¥IBM¥OnDemand32) に保管する必要があります。 DAF ファイルのフォーマットと構文は、標準 Windows INI ファイルと同じです。 DAF ファイルには AUDIT というセクションがあり、これによって 1 つまたは複数のフォルダー・セクションが識別されます。各フォルダー・セクションでは、監査できるフォルダーが識別されます。 198 ページの図 9 にサンプル DAF ファイルを示します。

```
[AUDIT]
FOLDERS=LDR,Student Information

[LDR]
FOLDER=Loan Delinquency Report
AUDIT_FIELD=Document Audit
PASS_TEXT=Pass
FAIL_TEXT=Fail
PASS_VALUE=P
FAIL_VALUE=F

[Student Information]
FOLDER=Student Information
AUDIT_FIELD=Audit Status
PASS_VALUE=P
FAIL_VALUE=F
```

図 9. サンプル DAF (ARSGUI.CFG) ファイル

AUDIT セクション

AUDIT セクションには、FOLDERS レコードというレコードが 1 つあります。この FOLDERS レコードには、フォルダー・セクション名のコンマで区切られたリストが含まれています。FOLDERS レコードに指定するフォルダー・セクションごとに、追加セクションを DAF ファイルの中に作成する必要があります。FOLDERS レコード内の合計文字数は、255 文字以下でなければなりません。

フォルダー・セクション

各フォルダー・セクションには、以下のレコードが含まれます。

- FOLDER には、OnDemand で表示されるのとまったく同じフォルダー名を指定します。FOLDER レコードは必須です。
- AUDIT_FIELD には、文書の監査に使用されるフォルダー・フィールドの名前を、OnDemand で表示されるのとまったく同じ名前にして指定します。詳しくは、 200 ページの『フォルダーの定義』を参照してください。AUDIT_FIELD レコードは必須です。
- PASS_TEXT は、監査にパスする文書にマークを付けるために使用されるコマンド・ボタンに表示される表題です。PASS_TEXT レコード内の合計文字数は、50 文字以下でなければなりません。 PASS_TEXT レコードはオプションです。デフォルトの表題は `Pass` です。
- FAIL_TEXT は、監査にパスしない文書にマークを付けるために使用されるコマンド・ボタンに表示される表題です。FAIL_TEXT レコード内の合計文字数は、50 文字以下でなければなりません。 FAIL_TEXT レコードはオプションです。デフォルトの表題は `Fail` です。
- PASS_VALUE は、監査にパスする文書についてデータベース内に保管される値です。この値は、アプリケーション・グループ・フィールドに保管されます。詳しくは、 199 ページの『アプリケーション・グループの定義』を参照してください。PASS_VALUE レコード内の合計文字数は、254 文字以下でなければなりません。 PASS_VALUE レコードは必須です。
- FAIL_VALUE は、監査にパスしない文書についてデータベース内に保管される値です。この値は、アプリケーション・グループ・フィールドに保管されます。詳

しくは、『アプリケーション・グループの定義』を参照してください。
FAIL_VALUE レコード内の合計文字数は、254 文字以下でなければなりません。
FAIL_VALUE レコードは必須です。

レポートの定義

以下のトピックでは、DAF を使用するレポートを定義するときに指定しなければならない情報を提供します。提供される情報は、OnDemand に対してレポートを定義するときに指定しなければならないその他の属性すべてに対する追加情報です。

- アプリケーション・グループの定義
- アプリケーションの定義
- フォルダーの定義

アプリケーション・グループの定義

「フィールド定義」ページのアプリケーション・グループに監査フィールドを追加します。フィールド・タイプは STRING でなければなりません。

「フィールド情報」ページの監査フィールドの属性を定義します。

- 「ストリング」エリアで、「大/小文字」を「大文字」、「タイプ」を「固定」、「長さ」を 1 に設定します。
- 「マッピング」エリアで、監査フィールドの「データベース値と表示値」を追加します。データベース値はデータベースに保管され、表示値は、検索フィールドと文書リストに表示されます。提供されたスペースにデータベース値およびこれに対応する表示値を入力します。「追加」をクリックし、値の各対をアプリケーション・グループに追加します。

  以下に示すように、監査にパスする (PASS) 文書について一組の値、監査にパスしない (FAIL) 文書について一組の値をそれぞれ追加し、さらに一組のデフォルト値を追加する必要があります。

  – PASS の「データベース値」は、DAF ファイルのフォルダー・セクション内の PASS_VALUE レコード値と一致しなければなりません。また、PASS の「表示値」が DAF ファイルのフォルダー・セクション内の PASS_TEXT レコード値と一致することが推奨されます。
  – FAIL の「データベース値」は、DAF ファイルのフォルダー・セクション内の FAIL_VALUE レコード値と一致しなければなりません。また、FAIL の「表示値」が DAF ファイルのフォルダー・セクション内の FAIL_TEXT レコード値と一致することが推奨されます。
  – レポートがアプリケーション・グループにロードされるとき、すべての文書の状況を設定するために、デフォルト値が使用されます。通常、デフォルト値は、None または Not Audited を表す N に設定します。

  200 ページの表 6 に例を示します。

表 6. 「データベース値」と「表示値」

データベース値	DAF ファイル	表示値	DAF ファイル
F	FAIL_VALUE=F	Fail	FAIL_TEXT=Fail
P	PASS_VALUE=P	Pass	PASS_TEXT=Pass
N		Not Audited	

注: N の「データベース値」と Not Audited の「表示値」は、DAF ファイルに保管されません。

アプリケーションの定義

「ロード情報」ページで監査フィールドのデフォルト値を追加します。このデフォルト値は、レポートがアプリケーション・グループにロードされるとき、すべての文書について OnDemand がデータベースに保管する値です。このデフォルト値は、N (None または Not Audited を表す) に設定するようお勧めします。

デフォルト値は、監査フィールドのデフォルトの「データベース値」と一致しなければなりません。この監査フィールドのデフォルトの「データベース値」は、アプリケーション・グループの「フィールド情報」ページで定義した値です。詳しくは、199 ページの『アプリケーション・グループの定義』を参照してください。

フォルダーの定義

「フィールド定義」ページのフォルダーに監査フィールドを追加します。フィールド・タイプは STRING でなければなりません。このフィールドの名前は、DAF ファイルのフォルダー・セクション内の AUDIT_FIELD レコード値と一致しなければなりません。詳しくは、198 ページの『フォルダー・セクション』を参照してください。

「フィールド・マッピング」ページのアプリケーション・グループ監査フィールドに、フォルダー監査フィールドをマップしてください。

DAF へのアクセスの制御

監査する必要のある文書へのアクセスは、数通りの方法で管理できます。しかし、システムの管理を単純化するために、グループを利用することをぜひお勧めします。例えば、カスタマーで、場合により、以下の 2 つのグループを定義します。

- Viewers (文書を表示するユーザー)。Viewers グループのユーザーは、少なくとも文書を監査することは許可されません。さらに、Viewers グループのユーザーが文書を開いたときに監査フィールドが表示されないよう、カスタマーで、場合により、デフォルトの論理ビューを作成します。また、ある場合には、監査にパスした文書だけが Viewers グループのユーザーに表示されるよう、カスタマーで照会の制限を定義します。
- Auditors (監査担当者)。Auditors グループのユーザーは、文書の監査を許可されます。Auditors グループのユーザーは、監査対象文書が含まれているアプリケーション・グループ内の文書の更新許可を得る必要があります。

システムの要件がこれら 2 つのグループによって満たされないときには、追加グループを定義したりシステムの構成を変更することが必要な場合があります。ユーザ

ー、グループ、またはシステム管理に関するその他の点について質問がある場合は、IBM サポート・センターに問い合わせてください。

DAF の使用

レポートが OnDemand にロードされたあと、許可ユーザーは、DAF を使用して文書の監査を行うことができます。文書を監査するには、DAF 制御ファイルに定義されているフォルダーの 1 つを開きます。そのフォルダーを検索し、監査する必要のある文書を求めます。例えば、監査フィールドに値 Not Audited が入っている文書を検索します。文書リストから 1 つまたは複数の文書を選択して表示します。文書表示ウィンドウで、監査にパスする文書にマークを付けるには、「パス (Pass)」ボタンをクリックし、監査にパスしない文書にマークを付けるには、「失敗 (Fail)」ボタンをクリックします。OnDemand は、DAF 制御ファイルに指定したパスまたは非パスの値でデータベースを更新します。

第 14 章 レジストリー (Registry) によるクライアントの動作の変更

注: 以下の手順では、コンピューターのレジストリーの編集が必要です。本当に必要な場合以外は、レジストリーを編集しないでください。レジストリーにエラーがあると、コンピューターが正しく動作しないことがあります。始める前に、レジストリーのバックアップ・コピーを取っておいてください。また、最後に正常にコンピューターを始動できたときに使用していたものと同じバージョンのレジストリーに復元する方法をよく知っている必要があります。説明については、Windows の情報を参照してください。

次のレジストリー値によって、Windows クライアントの動作を変更できます。 値は、このレジストリー・キーに設定する必要があります。

HKEY_CURRENT_USER¥Software¥IBM¥OnDemand32¥Client¥Preferences

FORCE_DEF_CHARSET

1 に設定すると、文書リストのフォントを選択するときに、デフォルトの文字セットの使用が強制されます。

デフォルト値: `0`

GREEN_BAR_COLOR

緑色バー背景色の「緑」の色調を変更できます。RGB 値で指定する必要があります。例えば、緑ではなくグレー・バーを使用するには、`192,192,192` と指定します。

デフォルト値: `128,255,128`

LINEDATA_FONT_FACE_1
LINEDATA_FONT_FACE_2
LINEDATA_FONT_FACE_3

行データ文書に使用する初期フォントを変更できます。 OnDemand がインストールされたあと、行データ文書を初めて開くときに、クライアントはフォントを選択します。これらのキーのどれかに値が指定されていれば、OnDemand は、他のフォント書体を検索する前に、まず LINEDATA_FONT_FACE_1 に指定されたフォント書体を検索し、次に LINEDATA_FONT_FACE_2、最後に LINEDATA_FONT_FACE_3 に指定されたフォント書体を検索します。

例えば、`LINEDATA_FONT_FACE_1` に、`FixedSys` を指定します。

MAXIMIZE_WINDOW

1 に設定すると、クライアント・ウィンドウが最大化されます。

デフォルト値: `0`

MULTIFIND

1 に設定した場合、AFP 文書でテキスト検索操作をすると、検出されたストリングがすべて、ページ上で強調表示されます。

デフォルト値: 0

SCS_CODEPAGE

AFP ビューアーで、SCS データに使用されるコード・ページを変更できます。 値が 0 であれば、使用されるコード・ページは、言語によって決まります。

デフォルト値: 0

SUPPRESS_FILE_CLEANUP

1 に設定すると、クライアントの初期化の間に、データ・ディレクトリーおよびリソース・ディレクトリー内の使用されなくなったファイルの削除は行いません。

デフォルト値: 0

SUPPRESS_WIN_POS_SAVE

1 に設定すると、クライアントの終了処理の間に、ウィンドウの位置とサイズの情報の保管は行いません。

デフォルト値: 0

TRACE

1 に設定すると、AFP ビューアーの操作の間に、トレース・ファイルが作成されます。 そのファイルは VIEWER.LOG という名前で、OnDemand のプロジェクト・ディレクトリーに入れられます。

デフォルト値: 0

TTONLY

1 に設定すると、AFP ビューアーが True Type フォントのみを使用するようにします。

デフォルト値: 0

UNDEFINED_CP

AFP ビューアーで、未定義のコード・ポイントの代わりに表示される文字を変更できます。 16 進文字で指定する必要があります。例えば、ピリオドならば、2E と指定します。

デフォルト値: 20

第 15 章 Monarch バージョン 5 との統合

このセクションでは、Monarch バージョン 5 を OnDemand Windows クライアントと統合する方法について説明します。 この機能を使用すると、ユーザーは、OnDemand クライアントから Monarch に文書を自動的にロードすることができます。ユーザーは、その場合、派生列の作成および図表やレポートの生成など、複雑なデータ操作を Monarch から行うことができます。

注: Monarch は、Datawatch Corporation から入手できるソフトウェア・プログラムです。

このセクションは、ソフトウェア製品のインストール、構成、および配布を担当する管理者にとって最重要なセクションです。このセクションでは、Monarch を OnDemand と統合するために通常実行する必要のある手順を示します。また、このセクションでは、一部の作業の実行方法について説明します。その他の作業を実行するときには、OnDemand のほかの情報や Monarch の情報が必要になります。

管理者は、OnDemand Windows 32 ビット・クライアント・インストール・プログラムを使用して、OnDemand クライアントとともに Monarch ファイルを配布することができ、OnDemand クライアントから Monarch を実行するようにワークステーションを構成することができます。 管理者が OnDemand クライアント・ソフトウェアのコピーをリファレンス・ワークステーション [1] 上に構成し、インストール・プログラムを配分サーバー [2] 上に構成することをお勧めします。OnDemand から Monarch を実行する予定の各ユーザーは、OnDemand クライアントのインストールまたはアップグレードをこの機能によって行う前に、各自のワークステーション上に Monarch ソフトウェアをインストールしておく必要があります。

管理者は、配分サーバー上の OnDemand Windows クライアントのインストール・ディレクトリー・ツリーに Monarch ファイルを保管できます。 ユーザーがサーバーからワークステーションに OnDemand クライアントをインストールするとき、インストール・プログラムによって Monarch ファイルが標準 OnDemand クライアント・ファイルと一緒にコピーされます。 以下のファイルも含めて数種類のファイルを配布できます。

- レジストリー・ファイル。ユーザーのワークステーション上のレジストリーに、インストール・プログラムによってレジストリー・ファイルがインポートされます。 インストール・プログラムにより OnDemand と Monarch がユーザーのワークステーション上で統合されるためには、OnDemand クライアント用の Monarch DLL に関する情報が入っている ODMonarch.Reg というレジストリー・ファイルがなければなりません。

1. リファレンス・ワークステーションには、Monarch DLL が OnDemand に対して定義されている OnDemand クライアントのインストール・コピーが入っています。管理者は、リファレンス・ワークステーションの構成情報を使用して、ソフトウェアを他のユーザーに配布します。

2. 配分サーバーとは、共用の場所に OnDemand クライアント・インストール・ソフトウェアのコピーがあるネットワーク・ファイル・サーバーです。ネットワーク上の他のユーザーは、このコピーを使用して OnDemand インストール・プログラムを実行し、配分サーバーから各自のワークステーションに OnDemand クライアントをインストールします。

重要: レジストリー・ファイルのインポートによって重要なデータを誤って破棄し、システムがまったく使用できなくなることが簡単に起こり得ます。Setup プログラムが実行される前にユーザーのワークステーション上のレジストリーをバックアップしておく計画を立ててください。

- Monarch モデル・ファイル。インストール・プログラムは、これらのファイルをユーザーのワークステーション上の Monarch¥Models ディレクトリーにコピーします。

以下のトピックで追加情報を提供します。

- はじめに
- OnDemand クライアント・ソフトウェアの構成
- OnDemand Setup プログラムの構成
- OnDemand Setup プログラムの実行
- OnDemand からの Monarch の実行
- クライアントのアップグレード

はじめに

重要: OnDemand バージョン 2.2.1.2 またはこれ以前のバージョンと Monarch が統合されている場合は、216 ページの『OnDemand クライアントのアップグレード』を参照してください。

先へ進む前に、あらかじめ以下の作業を完了しておく必要があります。

- Monarch バージョン 5 をリファレンス・ワークステーションにインストールしておきます。詳しくは、Monarch の情報を参照してください。
- OnDemand Windows クライアント用の最新の PTF を WWW 上の IBM サービスから入手しておきます。 このためには、Web ブラウザーで下記にアクセスします。

  ```
  ftp://www.service.software.ibm.com/software/ondemand/fixes
  ```

  次に、そのクライアント用の最新の PTF までリンクをたどります。odwin32.zip ファイルをクリックし、ディスクにこのファイルを保管します。

クライアントの構成

この章では、Monarch を OnDemand と統合するのに必要な情報を使用してクライアントをリファレンス・ワークステーション上に構成する方法について説明します。Monarch バージョン 5 のコピーをまだインストールしていない場合は、これをインストールしてください (詳しくは、Monarch の情報を参照してください)。次に、OnDemand クライアント・ソフトウェアのコピーをインストールします (詳しくは、OnDemand のほかの情報を参照してください)。次に、Monarch DLL のプロパティーを OnDemand クライアントに対して定義するレジストリーのキー、値、および値データを追加します。このあと、Monarch を OnDemand と正しく統合したことを検証します。そのためには、OnDemand クライアントを開始して文書を開き、関連メニュー・コマンドまたはツールバー・ボタンを使用して Monarch を呼び出しま

す。すべて正しく機能していると納得した場合、レジストリー・キーをファイルにエクスポートします。詳細は、以下の各トピックで説明します。

- レジストリー・キーの追加
- レジストリー・キーのエクスポート
- 複数の Monarch モデル・ファイルの使用

レジストリー・キーの追加

注: レジストリー・キーを追加する前に、OnDemand Windows クライアントをリファレンス・ワークステーションにインストールする必要があります。

レジストリーに情報を入れると、Monarch と OnDemand クライアントとの統合がアクティブになります。OnDemand は、初期化時にこの情報を検出すると、メニュー項目とツールバー・ボタンをクライアントのワークスペースに追加します。ユーザーがそのメニュー項目の 1 つを選択するか、またはそのツールバー・ボタンの 1 つをクリックすると、OnDemand は、Monarch DLL 内の関連したエントリー・ポイントを呼び出します。OnDemand は、**ExternalDlls** サブキーの下のレジストリーを調べ、実行するアクションおよびその他の情報を判別します。ExternalDlls サブキーの例を図 10 に示します。

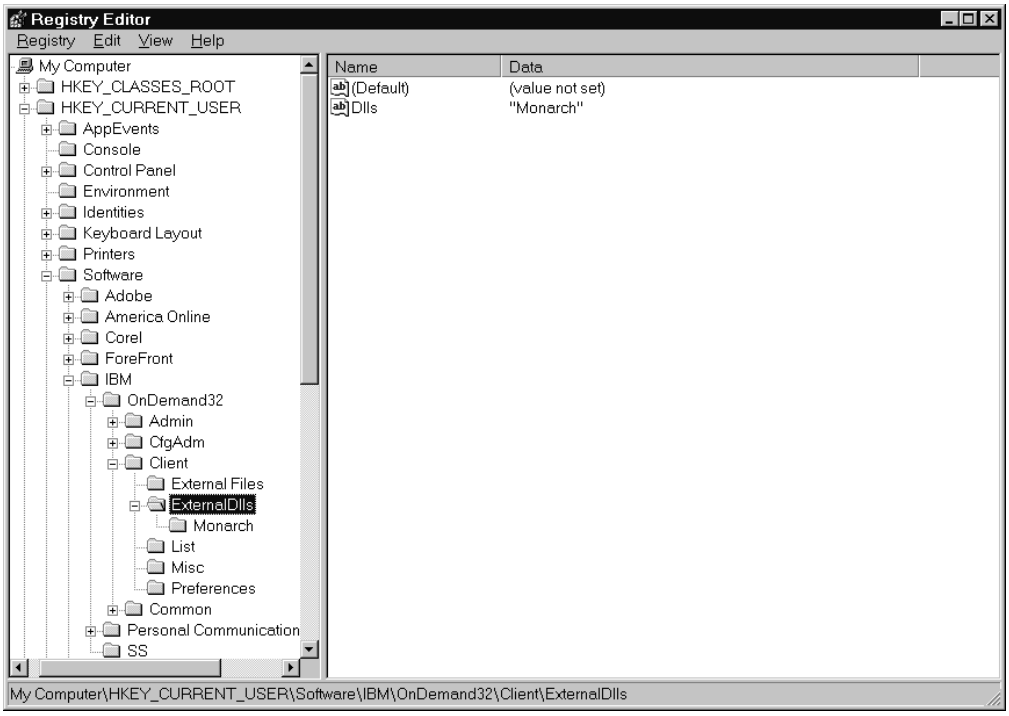


図 10. Dlls 内に値が 1 つある ExternalDlls サブキー

ExternalDlls の下のサブキーには、クライアントに統合された DLL を呼び出すために OnDemand が使用する情報が入っています。ユーザーが新規モデルの Monarch DLL をクライアントから呼び出すことができるサブキーの例を 208 ページの図 11 に示します。実行するモデル・ファイルの名前はサブキーでは指定されないので、OnDemand は、現行の文書で Monarch を開始します。

注: ユーザーが Monarch をクライアントから呼び出すときにモデル・ファイルをレジストリー・キー内に指定しないと、OnDemand は、現行の文書で Monarch を開始します。ユーザーは、この場合、Monarch の機能を使用してデータを分析することができ、任意で、モデルを作成して保管することができます。クライアントから作成して保管したモデルをその後実行するには、このモデルが DLL 内で識別される必要があり、このモデルのサブキーを ExternalDlls の下に追加する必要があります。モデルをクライアントから実行するために必要な値を、モデルのサブキーで指定する必要があります。例えば、Parameter 値には、モデル・ファイルの絶対パス名、およびモデルの実行に必要な Monarch パラメーターを入れる必要があります。

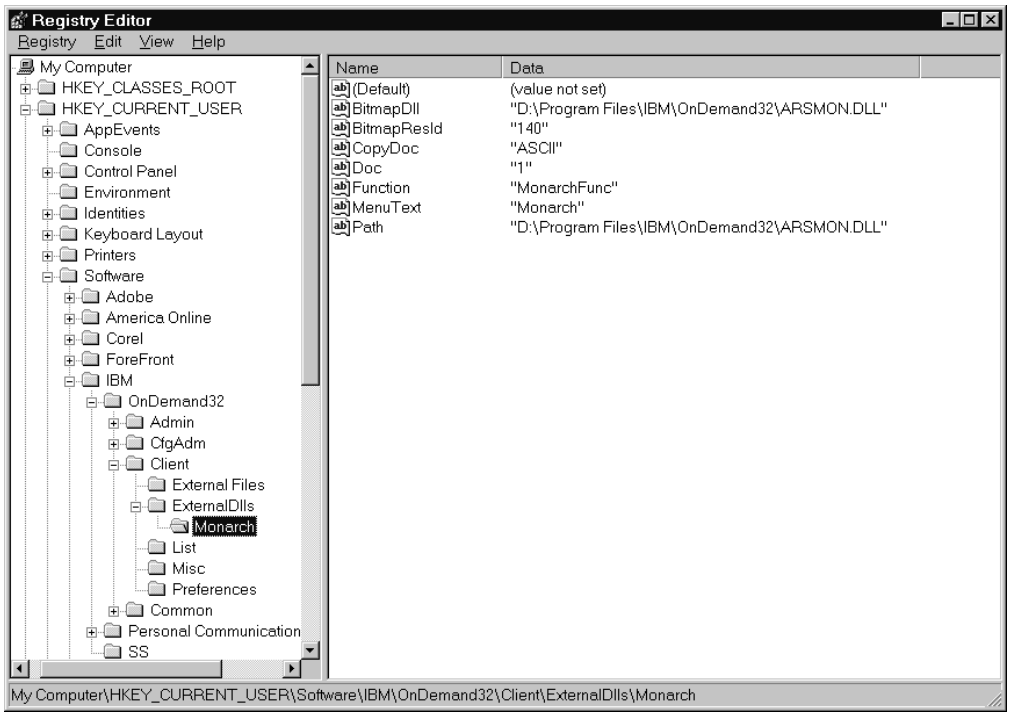


図 11. Monarch DLL のサブキーと値

図 11 に示されているサブキー値と値データについて、表 7 で説明します。場合により指定するその他の値 (図 11 には示されていない) については、210 ページの表 8 で説明します。

表 7. Monarch DLL のサブキー値

値とデータ	説明
BitmapDll C:¥Program Files¥IBM¥ OnDemand32¥ ARSMON.DLL	Monarch を OnDemand クライアントから呼び出すために使用するツールバー・ボタンのビットマップ・リソースが入っている DLL ファイルの絶対パス名。例には、IBM 提供のデフォルトを示してあります。 ただし、この DLL ファイルは、Path に指定される DLL ファイルとは異なる場合があり、また、異なる BitmapDll の DLL ファイルを使用することもできます。ビットマップは、幅 16 ピクセル、高さ 16 ピクセルでなければなりません。この値はオプションです。この値を指定しない場合、ツールバー・ビットマップは作成されません。

表 7. Monarch DLL のサブキー値 (続き)

値とデータ	説明
BitmapResId 140	BitmapDll に指定した DLL ファイルに入っているビットマップのリソース ID。BitmapDll を指定しない場合には、この値は無視され、指定した場合には、この値はオプションになります。この値を指定しないと、デフォルトで値 0 になります。
CopyDoc ASCII	1 つまたは複数の文書のコピーを Monarch に提供するよう指示し、提供するデータのタイプを指定します。その値が ASCII の場合は、文書データは ASCII ファイルに変換されます。その値が ASIS の場合は、文書データは元のフォーマットのままです。 文書が表示されているときに、ユーザーがメニュー項目またはツールバー・ボタンを選択した場合、提供されるデータは、現行の文書のデータです。文書リストが表示されている場合、そのデータは、文書リストで選択されたすべての文書を連結したものになります。 この値はオプションです。この値を指定しないと、Monarch に文書はまったく提供されません。
Doc 1	これには、以下の値の 1 つを指定できます。 0 は、関連したメニュー項目およびこれに対応するツールバー・ボタンの使用可能化が Folders 値と ExcludeFolders 値によってだけ制限されるよう指示します。Folders 値と ExcludeFolders 値に関する情報を参照してください。 1 は、文書が表示されているときにだけ、メニュー項目とツールバー・ボタンが使用可能になるよう指示します。 2 は、文書リストが表示されていて文書が少なくとも 1 つは選択されているときにだけ、メニュー項目とツールバー・ボタンが使用可能になるよう指示します。 3 は、文書が表示されているとき、または文書リストが表示されていて文書が少なくとも 1 つは選択されているときにだけ、メニュー項目とツールバー・ボタンが使用可能になるよう指示します。 この値はオプションです。この値を指定しないと、デフォルトで値 3 になります。
Function MonarchFunc	Monarch DLL ファイルへのエントリー・ポイントの名前。この値は必須指定です。

表 7. Monarch DLL のサブキー値 (続き)

値とデータ	説明
MenuText Monarch	Monarch を OnDemand クライアントから呼び出すときに使用するメニュー項目に表示されるテキスト。このメニュー項目は、「ウィンドウ」メニューで表示されます。関連したツールバー・ボタン上にユーザーがマウス・ポインターを置いたときにも、このテキストが表示されます。注: 例えば、Loan Analysis などのように、指定するテキストで Monarch 内の特定のアプリケーションについて記述できます。この値はオプションです。この値を指定しない場合、メニュー項目はブランクになります。
Path C:¥Program Files¥IBM¥ OnDemand32¥ ARSMON.DLL	ユーザーのワークステーション上の Monarch DLL ファイルの絶対パス名。 例には、デフォルトのインストール・ドライブおよびディレクトリーを示してあります。この値は必須指定です。

表 8. その他の DLL のサブキー値

値	説明
ExcludeFolders	1 つまたは複数の OnDemand フォルダーの名前を指定します。その構文は、Folders 値と同じです。 以下の場合には、常に、メニュー項目とツールバー・ボタンが使用可能になります。 • 文書が表示されているときに、この文書に関連したフォルダーが指定したフォルダーの 1 つではない場合。 • 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つではない場合。 この値はオプションです。この値を指定しないと、Folders 値によって使用可能化が制御されます。
Folders	1 つまたは複数の OnDemand フォルダーの名前を指定します。複数の名前を指定する場合は、名前を円記号 (¥) 文字で区切る必要があります。名前の末尾にアスタリスク (*) 文字をワイルドカード文字として使用できます。(こうすると、アスタリスクより前の部分の文字が一致するすべてのフォルダー名がリストされることになります。) 以下の場合には、常に、メニュー項目とツールバー・ボタンが使用可能になります。 • 文書が表示されているときに、この文書に関連したフォルダーが指定したフォルダーの 1 つである場合。 • 文書が表示されていないときに、現行フォルダーが指定したフォルダーの 1 つである場合。 この値はオプションです。ExcludeFolders 値を指定した場合、この値は無視されます。両方とも指定しないと、メニュー項目とツールバー・ボタンを使用可能にする前にフォルダー名のテストが実行されません。

表 8. その他の DLL のサブキー値 (続き)

値	説明
Parameter	パラメーター・データとして Monarch に渡される最高 255 までの文字を指定します。この値はオプションです。この値を指定しないと、Monarch にパラメーター・データは何も渡されません。指定できるパラメーターのリストについては、Monarch の情報を参照してください。 **注:** 特定の Monarch モデルを実行するには、モデル・ファイルの絶対パス名、およびそのモデルの実行に必要な Monarch パラメーターを、Parameter 値に入れる必要があります。Parameter 値を指定しないと、OnDemand は、現行の文書で Monarch を開始します。ユーザーは、この場合、Monarch の機能を使用してデータを分析することができ、任意で、モデルを作成して保管することができます。

レジストリー・キーを追加するには、以下の手順を実行してください。

- ExternalDlls サブキーには、OnDemand クライアントと統合される DLL のリストが含まれます。このサブキーには、各 DLL のプロパティーを定義するサブキーも含まれます。ExternalDlls サブキーを次のサブキーに追加してください。

  HKEY_CURRENT_USER¥Software¥IBM¥OnDemand32¥Client

- 値 Dlls を ExternalDlls サブキーに追加します。Dlls のタイプは String (REG_SZ) でなければなりません。Dlls のデータ値は、OnDemand クライアントと統合される DLL のコンマで区切った ID のリストです。このリスト内の各 ID には、ExternalDlls の下に関連したサブキーがなければなりません。例では、この ID は Monarch です。
- ExternalDlls サブキーの下にサブキーを追加することによって、Dlls にリストされている各 DLL を定義します。(追加する各サブキーは、Dlls 内にリストされている必要があります。)例では Monarch サブキーを追加しました。
- 追加したサブキーにストリング値を追加することによって、DLL のプロパティーを定義します。例 (213 ページの図 12 を参照) では、7 つのストリング値を追加しました。特定の Monarch モデルを実行するには、モデル・ファイルの絶対パス名、およびモデルの実行に必要な Monarch パラメーターを Parameter サブキー値に入れる必要があります。

レジストリー・キーのエクスポート

Monarch DLL のプロパティーを OnDemand Windows 32 ビット・クライアントに対して定義するレジストリー・キーを追加したあと (そして Monarch をこのクライアントから呼び出すことができるかどうかテストしたあと)、そのレジストリー・キーをファイルにエクスポートできます。 そのあと、配分サーバー上の OnDemand Windows 32 ビット・クライアント・インストール・ディレクトリー・ツリーの下にある CUSTOM サブディレクトリー・ツリーに、このファイルを保管します。そのサブディレクトリー・ツリーにこのファイルを配置したあとでは、OnDemand クライアントをインストールするユーザーが Setup を配分サーバーから実行すると、そのユーザーのワークステーションのレジストリーに自動的にレジストリー・キーがインポートされるようになります。 レジストリー・キーについて分からないことがあれば、 207 ページの『レジストリー・キーの追加』 を参照してください。

重要: 他のアプリケーションが OnDemand に統合されている (つまり、Monarch 以外のアプリケーションの DLL が ExternalDlls サブキーの下に定義されている) 場合は、この先へ進む前に IBM サポート・センターに相談してください。

レジストリー・キーをエクスポートするには、以下の手順を実行してください。

1. REGEDIT プログラムを開始します。
2. HKEY_CURRENT_USER¥IBM¥OnDemand32¥Client¥ExternalDLLs キーに移動します。
3. 「レジストリー (Registry)」メニューから「レジストリー・ファイルのエクスポート (Export Registry File)」を選択します。
4. このファイルを保持するディレクトリーを選択します。
5. 「ファイル名」フィールドに ODMonarch.Reg と入力します。
6. 「エクスポート範囲 (Export Range)」で「選択したブランチ (Selected Branch)」を選択します。
7. 「保管」をクリックします。

複数の Monarch モデル・ファイルの使用

207 ページの『レジストリー・キーの追加』では、Monarch の単一インスタンスを実行するためにクライアントをリファレンス・ワークステーション上に構成する方法について説明しています。異なるモデル・ファイルを Monarch のインスタンスごとに使用して、Monarch の複数インスタンスを実行するようにクライアントを構成することもできます。そのクライアントを構成するには、Monarch のインスタンスごとの ID を Dlls ストリング値に追加し、ID ごとにサブキーを ExternalDlls サブキーの下に追加します。異なるモデル・ファイルをユーザーがクライアントから実行できるよう、それぞれのサブキーを異なるモデル・ファイルに対応させる必要があります。このセクションでは、複数のモデル・ファイルとともにクライアントを構成するのに役立つ情報を提供します。

クライアントから実行されるそれぞれのモデル・ファイルごとに、以下の手順を実行してください。

- ExternalDlls サブキーの下の Dlls ストリング値に ID を追加します。モデル・ファイルごとに固有 ID を追加します。 213 ページの図 12 には、複数の ID がある ExternalDlls サブキーの例が示されています。(Model1、Model2、および Model3 という ID が追加されています。)

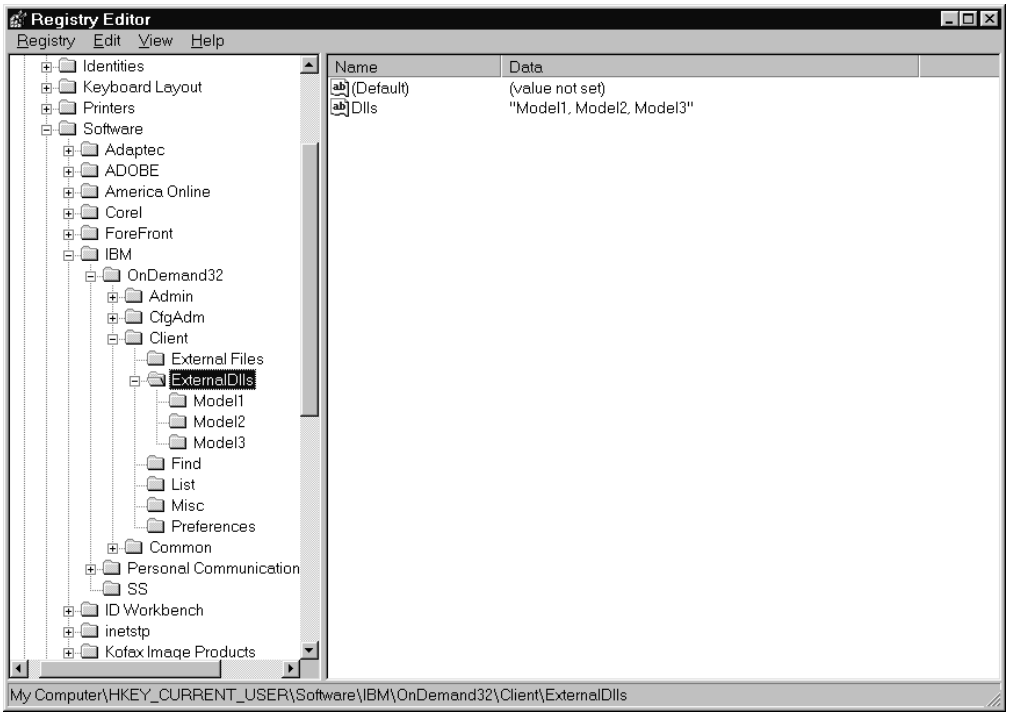


図 12. Dlls 内に複数の値がある ExternalDlls サブキー

- ExternalDlls サブキーの下にサブキーを追加することによって、クライアントに対して DLL を定義します。追加する各サブキーは、Dlls 内にリストされている必要があります。追加したサブキーにストリング値を追加することによって、DLL のプロパティーを定義します。
  - Parameter 値を使用して、モデル・ファイルの絶対パス名を指定します。特定の Monarch モデルを実行するには、モデル・ファイルの絶対パス名、およびそのモデルの実行に必要な Monarch パラメーターを、Parameter 値に入れる必要があります。指定できるパラメーターのリストについては、Monarch の情報を参照してください。
  - MenuText 値を使用すると、それぞれのモデル・ファイルごとに特定のメニュー・テキストと吹き出しヘルプを提供できます。
  - BitMapDll 値を使用すると、それぞれのモデル・ファイルごとに異なるツールバー・ボタン・ビットマップを指定できます。

  指定できるプロパティーに関するヘルプについては、208 ページの表 7 と 210 ページの表 8 を参照してください。

モデル・ファイルをクライアントに対して定義した後、レジストリー・キーをエクスポートしてください。211 ページの『レジストリー・キーのエクスポート』で説明している手順に従ってください。

Setup の構成

このトピックでは、Monarch モデル・ファイルをユーザーのワークステーションにコピーし、レジストリー・キーをユーザーのワークステーション上のレジストリーにインポートするためにクライアント・インストール・プログラムを構成する方法について説明します。 最初に、ZIP ファイルから配分サーバー上のディレクトリー

に最新版の OnDemand Windows 32 ビット・クライアント・ソフトウェアを抜き出します。 次に、配分サーバー上の OnDemand Windows 32 ビット・クライアントのインストール・ディレクトリー・ツリーにカスタム・ディレクトリーを追加します。 次に、Monarch ファイルをカスタム・ディレクトリーにコピーします。このあと、そのインストール・ディレクトリー・ツリーをユーザーが使用できるようにそのツリーを共用します。詳細は、以下の各トピックで説明します。

- 配分サーバーへのクライアント・ソフトウェアのコピー
- カスタム・ディレクトリーの追加
- 配分サーバーへの Monarch ファイルのコピー
- インストール・ディレクトリーの共用

クライアント・ソフトウェアのコピー

注: 最新版の OnDemand Windows クライアント・ソフトウェアを入手する方法については、 206 ページの『はじめに』を参照してください。

配分サーバーにクライアント・ソフトウェアをコピーするには、以下の手順を実行してください。

1. 管理者権限のあるユーザー ID を使用してサーバーにログオンします。
2. OnDemand Windows 32 ビット・クライアント ZIP ファイル (odwin32.zip) の内容をサーバー上のディレクトリーに抜き出します。 例えば、ファイルを ¥OD2217 ディレクトリーに抽出します。

   重要: ZIP 内容ディレクトリーおよびファイル構造がサーバー上に保持されるファイル抽出方式を利用してください。

完了すると、ターゲット・ディレクトリー (例では ¥OD2217) に、Setup プログラムとファイル、および ARS ディレクトリー・ツリーが格納されています。

サブディレクトリーの追加

配分サーバー上の OnDemand Windows クライアントのディレクトリー・ツリーの下にある CUSTOM サブディレクトリー・ツリーに Monarch ファイルを保管する必要があります。デフォルトでは、このディレクトリー・ツリーは ARS32 です。 (『クライアント・ソフトウェアのコピー』にある例に従った場合は、配分サーバー上のクライアント・ディレクトリー・ツリーが ¥OD2217¥ARS32 になります。) Monarch ファイルを保持するサブディレクトリーを追加するには、以下の手順を実行してください。

1. Windows クライアントのディレクトリーの下に CUSTOM ディレクトリーを作成します。 その例を以下に示します。

   ```
   mkdir ¥od2217¥ars32¥custom
   ```

2. ¥od2217¥ars32¥custom¥registry ディレクトリーを追加します。

   211 ページの『レジストリー・キーのエクスポート』で作成したレジストリー・ファイル (ODMonarch.Reg) がこのディレクトリーに保持されます。ユーザーが Setup プログラムを配分サーバーから実行すると、このファイルが Setup プログラムによってワークステーション上のレジストリーにインポートされます。

3. ¥od2217¥ars32¥custom¥monarch ディレクトリーを追加します。

ユーザーに配布する必要のある Monarch モデル・ファイルがこのディレクトリーに保持されます。Setup プログラムは、Monarch モデル・ファイルをユーザーのワークステーション上の Monarch¥Models ディレクトリーにコピーします。

Monarch ファイルのコピー

OnDemand クライアント・ソフトウェアを配分サーバーにコピーし、CUSTOM ディレクトリーを作成したあと (214 ページの『サブディレクトリーの追加』を参照)、Monarch ファイルをサブディレクトリーにコピーします。その例を以下に示します。

- 211 ページの『レジストリー・キーのエクスポート』で作成した ODMonarch.Reg ファイルを、¥OD2217¥ARS32¥CUSTOM¥REGISTRY ディレクトリーにコピーします。
- Monarch モデル・ファイルを ¥OD2217¥ARS32¥CUSTOM¥MONARCH ディレクトリーにコピーします。

インストール・ディレクトリーの共用

配分サーバーへの OnDemand クライアント・ソフトウェアのコピー、CUSTOM ディレクトリーの作成、およびこのサーバーへの Monarch ファイルのコピーを行ったあと、サーバー上のインストール・ディレクトリーをユーザーが使用できるようにする必要があります。その手順は、それぞれのネットワークおよびオペレーティング・システムごとに異なります。ただし、このディレクトリーへの読み取り専用アクセスをユーザーに提供することが一般的に必要です。使用ネットワークに適する場合は、配分サーバー上のフォルダーの場所にネットワーク名 (共用名) を付けることによって、フォルダーを共用してください。例えば、ユーザーがサーバー上の ¥OD2217 ディレクトリーから Setup プログラムを実行し、¥OD2217¥ARS32 ディレクトリー・ツリー内のすべてのファイルにアクセスできるようにするには、共用名 ODMONAR を ¥OD2217 ディレクトリーに割り当てます。

セットアップの実行

注: OnDemand クライアントをインストールして Monarch をこのクライアントと統合するためにユーザーが Setup プログラムを配分サーバーから実行する前に、Monarch バージョン 5 のコピーがユーザーのワークステーションにインストールされていることが必要です。

レジストリー・キーと Monarch モデル・ファイルとともに Setup プログラムを配分サーバーに構成したあと、Setup プログラムを配分サーバーからユーザーに実行してもらうことによって、インストールのテストを行うことができます。

注: レジストリー・ファイルのインポートによって重要なデータを誤って破棄し、システムがまったく使用できなくなることが簡単に起こり得ます。Setup プログラムが実行される前にユーザーのワークステーション上のレジストリーをバックアップしておく計画を立ててください。

インストール時にユーザーが「標準」オプションまたは「コンパクト」オプションを選択すると、Setup プログラムは、自動的に Monarch ファイルをワークステーションにコピーします。

インストール時にユーザーが「カスタム」オプションを選択したとき、クライアントの 1 つをインストールすることをユーザーが選択した場合は、Setup プログラムは、Monarch ファイルをワークステーションにコピーします。 ユーザーがクライアントの 1 つを選択しない場合は、Setup プログラムは、Monarch ファイルをワークステーションにコピーしません。 例えば、ユーザーが「カスタム」オプションを選択し、「Sonoran Fonts」または管理クライアントだけをインストールすることを選択した場合、Setup プログラムは、Monarch ファイルをワークステーションにコピーしません。ユーザーが「カスタム」オプションを選択し、「US English」クライアントをインストールすることを選択した場合、Setup プログラムは、Monarch ファイルをワークステーションにコピーします。

OnDemand からの Monarch の実行

ユーザーがクライアントを配分サーバーからインストールしたあと、すべて正しく機能していることをユーザーがテストする必要があります。ユーザーに OnDemand クライアントを開始して文書を開いてもらい、そのあと、関連メニュー・コマンドまたはツールバー・ボタンを使用して Monarch を開始してもらってください。これにより、Monarch が開始し、文書が別のウィンドウにロードされます。

OnDemand クライアントのアップグレード

OnDemand バージョン 2.2.1.2 またはこれ以前のバージョンと Monarch が統合されている場合は、OnDemand バージョン 2.2.1.3 またはこれ以降のバージョンへのクライアントのアップグレードがユーザーによって行われる前に、以下の重要な指示に従ってください。

- 古いレジストリー・ファイル (例: ODMonarch.Reg) を、配分サーバー上の CUSTOM¥REGISTRY ディレクトリーにコピーします。

  古いレジストリー・ファイルが存在しない場合は、手順に従ってこのファイルを作成し、作成したレジストリー・ファイルを、配分サーバー上の CUSTOM¥REGISTRY ディレクトリーにコピーします。
- Monarch モデル・ファイルを配分サーバー上の CUSTOM¥MONARCH ディレクトリーにコピーします。

第 16 章 ネットワーク上のクライアント・ソフトウェアのインストール

このセクションでは、複数のユーザーがネットワークを介して共用するクライアント・ソフトウェアのインストール方法について説明します。このセクションは、ソフトウェア製品のインストールと構成を担当する管理者を主な対象読者としています。

複数のユーザー間の OnDemand クライアントの共用

以下に示すように、OnDemand クライアント・ソフトウェアをインストールするための方式は数通りあります。

標準またはカスタム・インストール

この方式は、ユーザーのワークステーションのハード・ディスクに OnDemand クライアントをインストールするときに使用します。この場合、ユーザーは、このソフトウェアを実行するためにネットワーク・ファイル・サーバーに依存する必要がありません。標準インストールを実行するには、Setup プログラムを開始して「標準」オプションを選択します。ワークステーションに管理クライアントをインストールする必要がある場合、あるいはワークステーションに設定されているデフォルトの言語以外の言語で OnDemand クライアントをインストールする必要がある場合は、カスタム・インストールが必要になります。 カスタム・インストールを実行するには、Setup プログラムを開始して「カスタム」オプションを選択します。次に、インストールするコンポーネントを選択します。標準およびカスタム・インストールの手順は、「*ユーザーズ・ガイド*」で説明しています。

配布インストール

この方式は、ネットワーク・ファイル・サーバーのハード・ディスクに OnDemand クライアント・ソフトウェアをコピーするときに使用します。この場合、ユーザーは、そのサーバー上のコピーを使用して、標準、カスタム、配布、複数ユーザー、およびノード・インストールを実行できます。配布インストールは、ユーザー定義ファイルをクライアントのインストールに追加する必要がある場合にも役立ちます。219 ページの『配布インストール』で配布インストール手順について説明します。

複数ユーザー・インストール

この方式は、ネットワーク・ファイル・サーバーのハード・ディスクに OnDemand クライアントのコピーをインストールするときに使用します。この場合、ユーザーは、各自の PC でノード・インストールを実行すると、このクライアントをサーバーから実行するようにワークステーションを構成することができます。 220 ページの『複数ユーザー・インストール』で複数ユーザー・インストール手順について説明します。

ノード・インストール

この方式は、OnDemand クライアントのコピーをネットワーク・ファイル・サーバー上の共用の場所から実行するようにユーザーのワークステーション

を構成するときに使用します。 ノード・インストールを実行するには、「コンパクト」オプションを Setup プログラムから選択します。ノード・インストールでは、OnDemand クライアント制御ファイルがユーザーのワークステーションにコピーされます。プログラム・ファイルはユーザーのワークステーションにコピーされません。 ユーザーは、OnDemand クライアント・プログラムのコピーをネットワーク・サーバーから実行します。インストールの完了後、ユーザーが OnDemand を開始すると、オペレーティング・システムは、サーバーからワークステーション上のメモリーに OnDemand プログラムをロードします。 OnDemand は、ユーザーのワークステーション上の一時ワークスペースをデータ、リソース、および印刷のために割り振ります。 ユーザーが文書を表示したとき、OnDemand はユーザーがそのワークステーション上で変更したすべてのクライアント・オプションを保管します。 ノード・インストールでは、ユーザーのワークステーション上に約 2.5 MB のディスク・スペースが必要です。 221 ページの『ノード・インストール』でノード・インストール手順について説明します。

インストール・ディレクトリー

OnDemand クライアント CD-ROM が、それぞれのクライアントごとにディレクトリーに編成されます。Windows クライアントには、標準、カスタム、複数ユーザー、およびノード・インストール用の次のプログラムが 1 つ付随しています。

```
¥client¥windows¥setup.exe
```

クライアントをインストールするとき、表 9 にリストされたディレクトリーに、このインストール・プログラムによってクライアント・ソフトウェアがコピーされます。

表 9. Windows クライアントのインストール・ディレクトリー

ディレクトリー	目的
¥Program Files¥IBM¥OnDemand32	プログラム・ディレクトリー。標準、カスタム、および複数ユーザー・インストールの場合は、ワークステーション上のハード・ディスクが識別されます。 ノード・インストールの場合は、ネットワーク・ドライブが識別されます。
¥Program Files¥IBM¥OnDemand32	ローカル・ディレクトリー。ワークステーション上のハード・ディスクが識別されます。
¥Program Files¥IBM¥OnDemand32¥DATA	一時データ・ディレクトリー。ワークステーション上のハード・ディスクが識別されます。
¥Program Files¥IBM¥OnDemand32¥FONTS	AFP フォント・ファイル・ディレクトリー。標準、カスタム、および複数ユーザー・インストールの場合は、ワークステーション上のハード・ディスクが識別されます。 ノード・インストールの場合は、ネットワーク・ドライブが識別されます。
¥Program Files¥IBM¥OnDemand32¥PRINT	一時印刷ディレクトリー。ワークステーション上のハード・ディスクが識別されます。

表 9. *Windows* クライアントのインストール・ディレクトリー *(続き)*

ディレクトリー	目的
¥Program Files¥IBM¥OnDemand32¥RES	一時リソース・ディレクトリー。ワークステーション上のハード・ディスクが識別されます。

配布インストール

概要

配布インストールでは、OnDemand クライアント CD-ROM からネットワーク・ファイル・サーバー上の共用の場所に OnDemand Windows クライアントのディレクトリー・ツリーの内容がコピーされます。 この場合、ネットワーク・ファイル・サーバー上のコピーを使用して、標準、カスタム、配布、複数ユーザー、およびノード・インストールを実行できます。配布インストールは、ユーザー定義ファイルをクライアントのインストールに追加する必要がある場合にも役立ちます。

注: ネットワーク・ファイル・サーバー上の CD-ROM ドライブから CD-ROM を共用することによって、OnDemand クライアント・ソフトウェアを配布することもできます。

Adobe ソフトウェアの配布

ATM または Adobe Acrobat 表示ソフトウェアを OnDemand ユーザーに配布する必要がある場合、ネットワーク・ファイル・サーバー上の共用の場所に Adobe ソフトウェアをコピーしてください。 この場合、他のユーザーがそのコピーを使用して、Adobe ソフトウェアを各自のワークステーションにインストールできます。 Adobe ソフトウェアの入手およびインストール方法については、「*ユーザーズ・ガイド*」を参照してください。

サーバーへの OnDemand ソフトウェアのコピー

ネットワーク・ファイル・サーバーに OnDemand クライアント・ソフトウェアをコピーするには、以下の手順に従ってください。

1. 管理者権限のあるユーザー ID を使用してサーバーにログオンします。
2. OnDemand クライアント CD-ROM をドライブに挿入します。
3. `¥client¥windows` クライアント・ディレクトリー・ツリーの内容全体をネットワーク・ファイル・サーバー上のドライブにコピーします。

   注: CD-ROM のディレクトリーおよびファイル構造がネットワーク・ファイル・サーバー上に保持されるコピー方式を使用してください。

OnDemand クライアント・ソフトウェアをネットワーク・ファイル・サーバーにコピーしたあと、そのサーバーのディレクトリー (またはフォルダー) をネットワーク・ユーザーが使用できるようにしてください。その手順は、それぞれのネットワークおよびオペレーティング・システムごとに異なります。ただし、ディレクトリーへの読み取り専用アクセスをユーザーに提供することが一般的に必要です。ネッ

トワーク・ファイル・サーバー上のフォルダーの場所にネットワーク名 (共用名) を付けてフォルダーを共用することが使用ネットワークにとって適切であれば、そのようにしてください。

管理者がこのソフトウェアをサーバーにコピーしてディレクトリーを共用すると、その後、ユーザーは、標準インストールまたはカスタム・インストールをサーバーから実行してクライアントを各自のワークステーションにインストールできます。また、管理者は、複数ユーザー・インストールをサーバーから実行し、クライアントをネットワーク上の別の場所にインストールすることもできます。その場合、ユーザーは、ノード・インストールを実行すると、このクライアントをその場所から実行するように各自のワークステーションを構成することができます。

ユーザー定義ファイルの配布

管理者は、配分サーバー上の OnDemand クライアントのインストール・ディレクトリー・ツリーにユーザー定義ファイルを保管できます。そこに保管されているユーザー定義ファイルはすべて、ユーザーがサーバーから Setup プログラムを実行したときに、ワークステーションにコピーされます。 ユーザー定義ファイルは、Windows クライアントに分配できます。 ユーザー定義ファイルに対するサポートの詳細については、223 ページの『第 17 章 ユーザー定義ファイルの配布』を参照してください。

複数ユーザー・インストール

概要

複数ユーザー・インストールの第 1 段階として、OnDemand クライアントの標準インストールを実行します。標準インストールを使用して、ネットワーク・ファイル・サーバー上の共用の場所に OnDemand クライアントをインストールしてください。クライアントをそのサーバーにインストールすると、その後、ネットワーク上の他のユーザーがノード・インストールを実行し、クライアントをネットワーク・サーバーから実行するように各自のワークステーションを構成できるようになります。 ネットワーク・ユーザーのワークステーション上でノード・インストールを実行する方法については、221 ページの『ノード・インストール』で説明しています。

Adobe ソフトウェアのインストール

OnDemand に保管された PDF 文書を表示するために、Adobe Acrobat 表示ソフトウェアがユーザーに必要となります。 場合により、ネットワーク・ファイル・サーバー上の共用の場所に Acrobat 表示ソフトウェアをインストールする必要があります。 そのインストールによって、ネットワーク・ユーザーは、Acrobat 表示ソフトウェアをサーバーから実行できるようになります。 Adobe Acrobat ソフトウェアの入手およびインストール方法については、「*ユーザーズ・ ガイド*」を参照してください。

サーバーへの OnDemand クライアントのインストール

ネットワーク・ファイル・サーバーに OnDemand クライアントをインストールするには、以下の手順に従ってください。

1. 管理者権限のあるユーザー ID を使用してサーバーにログオンします。
2. OnDemand クライアント CD-ROM をドライブに挿入します。
3. 次の OnDemand クライアント・インストール・プログラムを開始します。

   ```
   ¥client¥windows¥setup.exe
   ```

4. 画面上のインストールに関する指示に従います。

   注: ユーザーが管理クライアントを実行する必要がある場合、管理者は、サーバーに設定されているデフォルトの言語以外の言語でクライアントを実行するか、「標準」オプションではなく「カスタム」オプションを選択する必要があります。 「カスタム」オプションを選択すると、サーバーにインストールする必要のある追加コンポーネントを選択できます。

5. ドライブとディレクトリーを確認します。ネットワーク・ファイル・サーバーにクライアントをインストールした場合、通常、すべてのディレクトリーがサーバーのディスクに存在します。表 10 に例を示します。

表 10. ネットワーク・ファイル・サーバー上のドライブとディレクトリー

ドライブとディレクトリー	目的
C:¥APPS¥ARS32	宛先フォルダー

OnDemand クライアントをネットワーク・ファイル・サーバーにインストールしたあと、そのサーバーのディレクトリー (またはフォルダー) をネットワーク・ユーザーが使用できるようにしてください。その手順は、それぞれのネットワークおよびオペレーティング・システムごとに異なります。ただし、ディレクトリーへの読み取り専用アクセスをユーザーに提供することが一般的に必要です。ネットワーク・ファイル・サーバー上のフォルダーの場所にネットワーク名 (共用名) を付けてフォルダーを共用することが使用ネットワークにとって適切であれば、そのようにしてください。クライアントをそのサーバーにインストールし、そのディレクトリーを共用すると、その後、ユーザーは、ノード・インストールを実行し、クライアントをそのサーバーから実行するように各自のワークステーションを構成できるようになります。 ノード・インストールの実行方法については、『ノード・インストール』で説明します。

ユーザー定義ファイルの共用

管理者は、ネットワーク・ファイル・サーバー上の OnDemand クライアントのディレクトリー・ツリーにユーザー定義ファイルを保管できます。このディレクトリー・ツリーに保管されたすべてのユーザー定義ファイルは、クライアントをその場所から実行するユーザーが共用できます。Windows クライアントは、ユーザー定義ファイルを共用できます。 ユーザー定義ファイルに対するサポートの詳細については、223 ページの『第 17 章 ユーザー定義ファイルの配布』を参照してください。

ノード・インストール

概要

OnDemand クライアントをネットワーク・ファイル・サーバー上の共用の場所から実行するようにユーザーのワークステーションを構成するには、ノード・インストールを実行します。 ノード・インストールを OnDemand クライアント CD-ROM

から実行する方法について、以下の手順で説明します。この手順では、ネットワーク・ファイル・サーバー上の共用の場所に OnDemand クライアントをあらかじめインストールしてあることを前提としています (詳しくは、 219 ページの『配布インストール』 を参照してください)。ノード・インストールを配分サーバーから実行することもできます。OnDemand ソフトウェアを配分サーバーにコピーする方法については、 219 ページの『配布インストール』を参照してください。

Adobe ソフトウェアのインストール

OnDemand に保管された PDF 文書を表示するために、Adobe Acrobat 表示ソフトウェアがユーザーに必要となります。 管理者が、ネットワーク・ファイル・サーバー上の共用の場所に Acrobat 表示ソフトウェアをインストールすると、ユーザーは、このソフトウェアをその場所から実行できるようになります。 そのようにインストールしない場合には、Adobe Acrobat ソフトウェアを入手してワークステーション上にインストールする方法について、「ユーザーズ・ガイド」を参照してください。

クライアントのインストール

OnDemand クライアントをネットワーク・ファイル・サーバー上の共用の場所から実行するようにユーザーのワークステーションを構成するには、以下の手順に従ってください。

1. OnDemand クライアント CD-ROM をドライブに挿入します。
2. 次の OnDemand クライアント・インストール・プログラムを実行します。

   ```
   ¥client¥windows¥setup.exe
   ```

3. 画面上の指示に従います。

   注: ほとんどのユーザーは、「コンパクト」オプションを選択する必要があります。

4. ドライブとディレクトリーを確認します。OnDemand クライアント・ソフトウェアをインストールしたネットワーク・ファイル・サーバー上のドライブとディレクトリーがネットワーク・プログラム・フォルダーおよびネットワーク・フォント・フォルダーの場所によって識別されることを確認してください (220 ページの『複数ユーザー・インストール』 を参照してください)。ユーザーのワークステーション上のドライブとディレクトリーが宛先フォルダーによって識別される必要があります。 表 11 に例を示します。

表 11. ネットワーク・ユーザーのワークステーション上のドライブとディレクトリー

ドライブとディレクトリー	目的
C:¥Program Files¥IBM¥OnDemand32	宛先フォルダー。通常、ユーザーのワークステーション上のハード・ディスクが識別されます。
N:¥APPS¥ARS32	ネットワーク・プログラム・フォルダー。ネットワーク・ファイル・サーバー上の OnDemand Windows クライアント・プログラム・ファイルの場所が識別されます。

第 17 章 ユーザー定義ファイルの配布

このセクションでは、ユーザー定義ファイルをユーザーに配布するための OnDemand Windows クライアント・インストール・プログラムの構成方法について説明します。 このセクションは、ソフトウェア製品のインストールと構成を担当する管理者を主な対象読者としています。

概要

ユーザー定義ファイルは、配分サーバー[3] の OnDemand Windows クライアントのディレクトリー・ツリーに保管できます。このディレクトリー・ツリーに保管したすべてのファイル (およびファイルのサブディレクトリー) は、ユーザーがクライアントをサーバーから PC にインストールすると、標準 OnDemand クライアント・ファイルと一緒にコピーされます。 以下に示すタイプのユーザー定義ファイルを配布できます。

- レジストリー・ファイル。このファイルは、ユーザーのワークステーション上のレジストリーにインポートされます。

  注: レジストリー・ファイルのインポートによって重要なデータを誤って破棄し、システムがまったく使用できなくなることが簡単に起こり得ます。このインポート機能を使用するときには注意が必要です。

- Windows ファイル。 このファイルは、ユーザーのワークステーション上の基本 Windows ディレクトリー・ツリーにコピーされます。
- 各種クライアント・ファイル。このファイルは、ユーザーのワークステーション上の ¥Program Files¥IBM¥OnDemand32 ディレクトリー・ツリーにコピーされます。 このコピー機能によって IBM 提供のファイルを置換 (上書き) できます。ユーザーが Setup プログラムをサーバーから実行すると、Setup プログラムは、まず IBM 提供ファイルをワークステーションにコピーします。 Setup プログラムは、次にすべてのユーザー定義ファイルをワークステーションにコピーします。 ユーザー定義ファイルの名前が IBM 提供ファイルと同じであれば、ユーザー定義ファイルによって IBM 提供ファイルは上書きされます。 例えば、文書の作成に使用したフォントを PC 上で表示可能なフォントにマップするために、AFP 文字セット定義ファイル (CSDEF.FNT) を変更する必要がある場合、 変更された CSDEF.FNT ファイルを配分サーバー上の CUSTOM¥FILES¥FONT ディレクトリーに保管すると、更新されたバージョンのファイルを自動的にユーザーに配布できます。ユーザーがクライアントをサーバーからインストールすると、Setup プログラムは、まず IBM 提供の CSDEF.FNT ファイルをワークステーションにコピーします。 Setup プログラムは、次に変更した CSDEF.FNT ファイルをワークステーションにコピーします。

3. 配分サーバーとは、共用の場所に OnDemand クライアント・ソフトウェアのコピーがあるネットワーク・ファイル・サーバーです。ネットワーク上の他のユーザーは、このコピーを使用して標準、カスタム、配布、複数ユーザー、およびノード・インストールを実行します。ネットワーク上の OnDemand ソフトウェアを共用するように配分サーバーを設定する方法の詳細については、 217 ページの『第 16 章 ネットワーク上のクライアント・ソフトウェアのインストール』を参照してください。

以下のトピックには、ユーザー定義ファイルの配布に関するより詳しい情報が記載されています。

- 配分サーバーへの OnDemand クライアント・ソフトウェアのコピー
- ユーザー定義ファイルを保持するサブディレクトリーの追加
- サブディレクトリーへのユーザー定義ファイルの保管
- ユーザーのワークステーションへの OnDemand クライアントのインストール

サーバーへの OnDemand クライアント・ソフトウェアのコピー

配分サーバーへの OnDemand クライアント・ソフトウェアのコピーに関する説明については、217 ページの『第 16 章 ネットワーク上のクライアント・ソフトウェアのインストール』を参照してください。

サブディレクトリーの追加

ユーザー定義ファイルはすべて、配分サーバー上の OnDemand Windows クライアントのディレクトリー・ツリーの下にある CUSTOM サブディレクトリー・ツリーに保管する必要があります。 デフォルトでは、Windows クライアントのディレクトリー・ツリーは ARS32 です。 (217 ページの『第 16 章 ネットワーク上のクライアント・ソフトウェアのインストール』にある指示に従った場合は、配分サーバー上の OnDemand Windows クライアント・ディレクトリー・ツリーは ¥client¥windows¥ars32 です。)

ユーザー定義ファイルをインストールするためにサーバーを構成するには、以下の手順を実行してください。

1. Windows 32 ビット・クライアントのディレクトリーの下に CUSTOM ディレクトリーを作成します。 その例を以下に示します。

   ```
   mkdir ¥client¥windows¥ars32¥custom
   ```

2. 以下のサブディレクトリーのうち 1 つまたは複数を CUSTOM ディレクトリーに追加します。どのサブディレクトリーを追加するかは、ユーザーに配布するユーザー定義ファイルのタイプによって決まります。その例を以下に示します。

   ¥client¥windows¥ars32¥custom¥registry

   レジストリー・ファイル (ファイル・タイプ REG) を保持します。このファイルは、ユーザーのワークステーション上のレジストリーにインポートされます。 レジストリー・エディターを使用してエクスポートされたレジストリーの選択対象の分岐は、通常、レジストリー・ファイルによって構成されます。

   注: レジストリー・ファイルのインポートによって重要なデータを誤って破棄し、システムがまったく使用できなくなることが簡単に起こり得ます。このインポート機能を使用するときには注意が必要です。

   ¥client¥windows¥ars32¥custom¥windows

   Windows ファイルを保持します。 Setup プログラムは、これらのファイルとサブディレクトリーをユーザーのワークステーション上の基本 Windows ディレクトリーにコピーします。 (Setup プログラムは、ユーザーのワークステーション上の基本 Windows ディレクトリーの名前を

自動的に判別します。) 必要に応じて、サブディレクトリーを追加してください。例えば、Windows 32 ビット・システム・ファイルを配布する予定の場合は、 SYSTEM32 サブディレクトリー (¥CLIENT¥WINDOWS¥ARS32¥CUSTOM¥WINDOWS¥SYSTEM32) を追加します。 Setup プログラムは、このディレクトリー内のファイルをユーザーのワークステーション上の ¥WINDOWS¥SYSTEM32 ディレクトリーにコピーします。

¥client¥windows¥ars32¥custom¥files

各種 OnDemand クライアント・ファイルを保持します。Setup プログラムは、これらのファイルとサブディレクトリーをユーザーのワークステーション上の ¥Program Files¥IBM¥OnDemand32 ディレクトリー・ツリーにコピーします。 必要に応じて、サブディレクトリーを追加してください。例えば、AFP フォント・ファイルを配布する予定の場合は、FONT サブディレクトリー (¥CLIENT¥WINDOWS¥ARS32¥CUSTOM¥FILES¥FONT) を追加してください。 Setup プログラムは、ファイルをユーザーのワークステーション上の ¥Program Files¥IBM¥OnDemand32¥FONT ディレクトリーにコピーします。

サーバー上のユーザー定義ファイルの保管

OnDemand クライアント・ソフトウェアを配分サーバーにコピーし、`¥custom` ディレクトリーを作成したあと、ファイルをそれぞれのサブディレクトリーに保管することができます。例えば、 OnDemand が実行する DLL ファイルを ¥CLIENT¥WINDOWS¥ARS32¥CUSTOM¥FILES ディレクトリーに保管してください。

OnDemand クライアントのインストール

管理者が CUSTOM ディレクトリー・ツリーを配分サーバー上に設定すると、その後、ユーザーは、クライアントとユーザー定義ファイルのインストールを開始できるようになります。次回、ユーザーが Setup プログラムをサーバーから実行すると、Setup プログラムは、OnDemand クライアントをワークステーションにインストールし、サーバーに保管されたユーザー定義ファイルすべてをユーザーのワークステーションにコピーします。

インストール時にユーザーが「標準」オプションか「コンパクト」オプションのいずれかを選択すると、Setup プログラムは、自動的にユーザー定義ファイルをワークステーションにコピーします。

インストール時にユーザーが「カスタム」オプションを選択したときには、Setup プログラムは、ユーザー定義ファイルをワークステーションにコピーする場合もしない場合もあります。 ユーザーがクライアントの 1 つをインストールすることを選択した場合は、Setup プログラムは、ユーザー定義ファイルをワークステーションにコピーします。 そうしない場合には、Setup プログラムは、ユーザー定義ファイルをワークステーションにコピーしません。 例えば、ユーザーが「カスタム」オプションを選択し、Sonoran Fonts だけをインストールすることを選択した場合、Setup プログラムは、ユーザー定義ファイルをワークステーションにコピーしません。 し

かし、ユーザーが「カスタム」オプションを選択し、管理クライアントをインストールすることを選択した場合、Setup プログラムは、ユーザー定義ファイルをワークステーションにコピーします。

第 18 章 応答ファイルの使用

このセクションでは、応答ファイルを使用してクライアント・ソフトウェアのインストールを自動化する方法について説明します。このセクションは、ソフトウェア製品のインストールと構成を担当する管理者を主な対象読者としています。

概要

このセクションでは、ネットワークに接続しているワークステーションに Windows クライアント・ソフトウェアをインストールするための応答ファイルを作成および使用するときに利用できる情報を提供します。 通常、Setup プログラムを使用して応答ファイルを作成します。そのあと、Setup プログラムを実行し、応答ファイルの名前を指定して、このソフトウェアを他のワークステーションにインストールします。

応答ファイルとは、ワークステーション上での製品のリダイレクトされたインストール中に必要となるクライアント特有の構成情報を提供する ASCII ファイルです。例えば、インストール・ドライブおよびディレクトリー、インストールするコンポーネントなど、ユーザーが製品のインストール時に通常尋ねられる構成上の質問について事前定義された応答が、この応答ファイルには入っています。システム管理者は、応答ファイルを使用すると、ワークステーションのネットワークを介した Windows 32 ビット・クライアント・ソフトウェアのインストールと構成を自動化できます。 応答ファイルを使用すると、システム管理者 (またはその他のユーザー) がそれぞれのワークステーションの前に座って、インストール時に表示されるそれぞれの質問への応答を手動で入力する必要がなくなります。

応答ファイルのフォーマット

応答ファイルのフォーマットは、.INI ファイルのフォーマットに似ています。応答ファイルには、セクションに編成されたキーワードと値の対が含まれています。このキーワードと値は、ソフトウェアのインストール時に解釈されます。

応答ファイルの作成

応答ファイルには、一般に拡張子 .ISS が付いています。Windows ディレクトリーにこのファイルは入っています。

-r コマンド行オプションを指定して Setup プログラムを実行すると、応答ファイルを作成できます。例えば、次のコマンドを実行します。

```
¥client¥windows¥setup -r
```

このコマンドを実行すると、Setup プログラムは、製品のインストール上の質問に対するすべての応答を SETUP.ISS 応答ファイルに記録します。**-f1** コマンド行オプションを指定すると、別のディレクトリーに応答ファイルを入れるように Setup プログラムに指示することができ、応答ファイル名を指定することができます。例えば、次のコマンドを実行します。

```
¥client¥windows¥setup -r -f1n:¥client¥windows¥ars32in.iss
```

これにより Setup プログラムは、N ドライブ上の ¥CLIENT¥WINDOWS ディレクトリーに ARS32IN.ISS ファイルを作成します。

応答ファイルを使用したソフトウェアのインストール

応答ファイルは、直接的には呼び出されません。代わりに、インストール・プログラムに対するパラメーター値として応答ファイルを指定します。**-s** コマンド行オプションを指定して Setup プログラムを実行し、応答ファイルを指定できます。例えば、次のコマンドを実行します。

```
¥client¥windows¥setup -s
```

これにより Setup プログラムは、現行ドライブ上の ¥CLIENT¥WINDOWS ディレクトリーの SETUP.ISS 応答ファイルにある命令を使用して、ソフトウェアをインストールします。デフォルトでは、Setup プログラムを実行するディレクトリーに、応答ファイルは入っていなければなりません。**-f1** オプションを使用して、応答ファイルの場所と名前を示してください。例えば、次のコマンドを実行します。

```
¥client¥windows¥setup -s -f1n:¥client¥windows¥ars32in.iss
```

これにより Setup プログラムは、N ドライブ上の ¥CLIENT¥WINDOWS ディレクトリーにある ARS32IN.ISS 応答ファイルを使用して、ソフトウェアをインストールします。

応答ファイルは、Windows 32 ビット・クライアント・ソフトウェアのインストールの処理を指示します。 **-s** オプションを指定して Setup プログラムを実行すると、メッセージやダイアログ・ボックスはまったく表示されません。代わりに、ログ・ファイルにメッセージが書き込まれます。デフォルトでは、ログ・ファイル (SETUP.LOG) は、Setup プログラムが入っているディレクトリーに書き込まれます。**-f2** コマンド行オプションを指定すると、別のディレクトリーにログ・ファイルを入れるように Setup プログラムに指示することができ、ログ・ファイル名を指定することができます。例えば、次のコマンドを実行します。

```
¥client¥windows¥setup -s -f1n:¥client¥windows¥ars32in.iss -f2c:¥temp
¥ars32in.log
```

このコマンドを実行すると、Setup プログラムは、C ドライブの TEMP ディレクトリーにログ・ファイル ARS32IN.LOG を書き込みます。

ソフトウェアのインストールの検証

応答ファイルを使用してインストールした製品のインストールについて検証するには、ログ・ファイルを開いて ResponseResult セクションを見つけます。ResultCode パラメーターの値を調べてください。戻りコードが 0 になっている必要があります。

OnDemand ソフトウェアをインストールするための応答ファイルの使用

応答ファイルを使用して OnDemand Windows クライアント・ソフトウェアをインストールする準備を行ったあと、ネットワークに接続している他のワークステーションにこのソフトウェアをインストールするには、以下の手順を完了するのが一般的です。

1. ソフトウェアをワークステーションにインストールします。 応答ファイルを作成するための **-r** オプション、および応答ファイルの場所と名前を識別するための **-f1** オプションを指定して、Setup プログラムを実行します。Setup プログラムからプロンプトが出たとき、「標準」オプションを選択します。
2. ユーザーのワークステーションにソフトウェアをインストールし、インストール・プロセスと応答ファイルのテストを行います。 手順 1 で作成した応答ファイルの名前を指定します。
3. 応答ファイルのテストと検証を行ったあと、ソフトウェアを他のワークステーションにインストールします。 手順 1 で作成した応答ファイルを読み取るための **-s** オプション、応答ファイルを識別するための **-f1** オプション、および Setup プログラムがログ・ファイルを書き込むディレクトリーを識別するための **-f2** オプションを指定して、Setup プログラムを実行します。
4. ログ・ファイルを調べて、ソフトウェアのインストールについて検証します。

第 19 章 AFP フォントのマッピング

文書を作成するときに使用された AFP フォントを、ワークステーションで表示可能なフォントに、OnDemand クライアントがマップする必要があります。 AFP 文書を表示するのに最適なアウトライン・フォントへのマッピングをこのクライアントが行うためには、文書の作成時に使用されたフォントに関するある種の特性を、このクライアントは認識する必要があります。 AFP フォントとアウトライン・フォントのマッピングは、クライアントの一部としてインストールされる IBM 提供のフォント定義ファイルを使用して実行されます。 そのファイルは、クライアントをインストールしたディレクトリーの下の FONT ディレクトリーにインストールされます。そのファイルは、任意のワークステーション・エディターを使用して編集できます。インストールされるバージョンのフォント定義ファイルでは、IBM Core Interchange (Latin だけ) フォント、互換フォント、整合フォント、Sonoran フォント、および Data1 フォントがマップされます。

フォント・ファミリー (ファミリー名) がワークステーションにインストールされていない AFP フォントを文書で使用する場合は、ALIAS.FNT ファイル (クライアントとともにインストールされるフォント定義ファイルの 1 つ) を使用して、そのフォント・ファミリー名を別のものに置換してください。 ALIAS.FNT ファイルによって、AFP フォントのうちのいくつかは IBM Core Interchange フォントに再マップされます。 アウトライン・フォントがワークステーションにインストールされている場合、ALIAS.FNT ファイル内のフォント・ファミリー名の置換を除去したりコメント化することが必要な場合があります。ALIAS.FNT ファイルの使用方法の詳細については、241 ページの『別名ファイル』を参照してください。

フォントのマッピングが必要なとき

OnDemand に対して定義されていないフォントを使用する場合、IBM AFP フォントを変更した場合、または独自の AFP フォントを作成した (例えば、PSF/2 Type Transformer によって) 場合は、これらのフォントを使用している文書がクライアントによって正しく表示されるために、フォント定義ファイル内にこれらのフォントを定義する必要があります。

- 新しいコード化フォント (またはリネームされたフォント) を作成した場合、コード化フォント・ファイル (ICODED.FNT または CODED.FNT) 内にそのコード化フォントを定義する必要があります。
- 文字セットを新たに作成した場合、それを文字セット定義ファイル (CSDEF.FNT) 内に定義する必要があります。
- コード・ページを新たに作成した場合、それをコード・ページ定義ファイル (CPDEF.FNT) 内に定義する必要があります。
- コード・ページを新たに作成したか、または文字を移動してコード・ページを変更した場合、新しいコード・ページ・ マップ・ファイル (cp_id.CP) を作成する必要があります。

既存の IBM フォント・コンポーネントを変更しただけであれば、上記の手順を何も実行する必要がない場合もあります。 例えば、IBM コード・ページ内のコード・ポイントを削除しただけであれば、クライアントによって提供されるフォント・ファイルを使用できます。

フォントのマッピング用に提供されるファイル

デフォルトでは、クライアントがインストールされたディレクトリーの下のサブディレクトリーに、フォント・サポート用の以下に示したタイプのファイルがインストールされます。

表 12. フォント・ファイルおよびディレクトリー

ファイル	ファイル名	サブディレクトリー	説明
コード化フォント・ファイル	CODED.FNT[1] ICODED.FNT ICODED.CHS[2] ICODED.CHT[3] ICODED.JPN[4] ICODED.KOR[5]	..¥FONT	コード化フォントを作成する AFP コード・ページと AFP フォント文字セットを指定します。
文字セット定義ファイル	CSDEF.FNT CSDEF.CHS[2] CSDEF.CHT[3] CSDEF.JPN[4] CSDEF.KOR[5]	..¥FONT	AFP 文字セット属性、例えば、ポイント・サイズなどを定義します。 また、このファイルは、そのフォント・グローバル ID にフォント文字セットをマップします。
コード・ページ定義ファイル	CPDEF.FNT CPDEF.CHS[2] CPDEF.CHT[3] CPDEF.JPN[4] CPDEF.KOR[5]	..¥FONT	各 AFP コード・ページを Windows 文字セット[6] にマップし、クライアントが使用するコード・ページ・マップ・ファイルを指示します。
コード・ページ・マップ・ファイル	*cpgid*.CP	..¥FONT¥MAPS	文字 ID マッピングを定義します。このファイルは、IBM コード・ページの文字 ID およびその 16 進コード・ポイントを、これらに対応する文字 ID、および Windows の ANSI または SYMBOL 文字セット[6] を表す ASCII コード・ポイントと突き合わせます。
別名ファイル	ALIAS.FNT	..¥FONT	AFP フォント・タイプ・ファミリーを TrueType フォント・ファミリー名にマップします。

注::

1. CODED.FNT は、オプション・ファイルです。FONT サブディレクトリーの SAMPLES サブディレクトリーにサンプルがあります。CODED.FNT ファイルは、作成されたコード化フォントを入れるためのファイルです。
2. 中国語 (簡体字) 版クライアント用のコード・ページおよび文字セット・ファイル。
3. 中国語 (繁体字) 版クライアント用のコード・ページおよび文字セット・ファイル。
4. 日本語版クライアント用のコード・ページおよび文字セット・ファイル。
5. 韓国語版クライアント用のコード・ページおよび文字セット・ファイル。
6. Windows 用語の「文字セット」は AFP 用語の「コード・ページ」と大体同じです。

フォントのマッピングの手順

この章のこれ以降を読んで、変更する必要のあるフォント・ファイルを判別したあと、以下の手順に従ってください。

1. フォント定義ファイル内にフォントを定義するために必要な情報を収集します。この情報については、この付録の後続のセクションで説明します。
2. 変更する予定の以下のフォント定義ファイルのバックアップ・コピーを作成します。
   - CSDEF.FNT
   - CPDEF.FNT
   - ICODED.FNT
   - ALIAS.FNT

   注: これらのファイルを変更したあとのコピーが操作できなくなるような事態になっても、未変更のコピーを使用できるよう、これらのファイルのバックアップ・コピーを作成しておくことが必要です。
3. クライアントで使用する予定の他のアウトライン・フォントをインストールします。(ATM によるフォントのインストール方法は、「*Adobe Type Manager User Guide*」を参照してください。)
4. コード・ページを作成または変更した場合は、次に示すように、BLDCPMAP REXX™ プログラムを使用してコード・ページ・マップ・ファイルを作成します。
   a. どちらの Windows 文字セット (ANSI または SYMBOL) が AFP コード・ページに適合するかを判別します。
   b. 必要に応じて、コード・ページ・マップ・ファイルまたは ALIAS.FNT ファイルの中に、適合しない文字があれば、それを置換します。(コード・ページ・マップ・ファイルおよびコード・ページ・マップ・ファイル・プログラムについては、それぞれ、 238 ページの『コード・ページ・マップ・ファイル』および 239 ページの『コード・ページ・マップ・ファイル作成用のコード・ページ・マップ・ファイル REXX プログラム』を参照してください。)
   c. 使用する予定のフォントについて、CPDEF.FNT ファイルを編集し、コード・ページ名、コード・ページ ID、および最適な Windows 文字セット名を追加します。

      注: Windows 文字セット SYMBOL を指定する場合、コード・ページで使用されるフォント・ファミリー名は、記号フォントでなければなりません。
5. 文字セットを新たに作成した場合は、CSDEF.FNT ファイルを編集し、文字セット名を [CHARSET] セクションに追加します。フォントについて正しい属性を CSDEF.FNT の中に指定してください。新規フォント・グローバル ID を指定しようとする場合は、適切な情報をこのファイルの [FGID] セクションに追加します。
6. コード化フォントを作成した場合は、CODED.FNT ファイルを作成または編集し、コード化フォントを追加します。

フォント定義ファイルの構文規則

フォント定義ファイルの構文規則は、以下のとおりです。

- フォント定義ファイルの最初の欄にセミコロン (;) があると、その行は、コメント・ステートメントとして扱われて無視されます。
- ファイル内のセクション・ヘッダーは、ブラケット [ ] で囲まれます。このセクション・ヘッダーを除去したり変更することはできません。
- すべての値について大/小文字を区別しません。
- パラメーター値が無効なときにデフォルト値が存在していると、そのパラメーター値は置換されます。
- すべてのパラメーターが定位置パラメーターです。
- パラメーター値間にブランクを入れることができます。

コード化フォント・ファイル

IBM コード化フォント・ファイル (ICODED.FNT) によって、AFP コード化フォントがその AFP 文字セットおよび AFP コード・ページにマップされます。 以下の 2 種類のコード化フォント・ファイルをクライアントで使用できます。

ICODED.FNT
このファイルには、約 2500 の IBM 提供コード化フォントについての定義が含まれています。

CODED.FNT
作成したコード化フォントのリストを定義するために、このオプション・ファイルを作成できます。CODED.FNT ファイルを作成した場合、これを FONT サブディレクトリーに入れることが必要です。FONT サブディレクトリーの SAMPLES サブディレクトリーに、このファイルのサンプルがあります。

CODED.FNT ファイルが FONT サブディレクトリーの中に存在していると、このファイルが最初に検索され、AFP ファイルで使用されるコード化フォントが求められます。 コード化フォント名が CODED.FNT 内にないか、または CODED.FNT が存在していない場合は、クライアントによって提供された ICODED.FNT ファイルだけが検索されます。

```
X?A155N2 = C?A155N1, T1DCDCFS
X?AE10   = C?S0AE10, T1S0AE10
X?GT10   = C?D0GT10, T1D0BASE
X?ST15   = C?D0ST15, T1D0BASE
X?A0770C = C?A07700, T1DCDCFS
X?A0770I = C?A07700, T1GI0361
X0T0550C = C0T05500, T1DCDCFS
```

図 13. CODED.FNT ファイルの内容の一部の例

コード化フォント・ファイルの規則

- 疑問符 (?)。コード化フォント名と文字セット名の 2 番目の文字についてだけ、ワイルドカード文字として疑問符 (?) を使用できます。これによって、コード化フォントのすべての文字回転が検索の際に 1 つのエントリーで処理されるようになります。

注: コード化フォントについて順次検索が実行され、最初に一致したもの (ワイルドカード文字も含めて) が使用されます。

- コード化フォント名のあとに、まず、文字セット名をリストし、そのあとにコード・ページ名を続ける必要があります。
- 文字セットとコード・ページは、コンマで区切ることが必要 です。

文字セット定義ファイル

文字セット定義ファイルでは、フォントの文字セット属性とフォント・グローバル ID を指定します。このファイルは、2 つのセクションに分かれ、その 1 つは文字セット [CHARSET] のセクション、もう 1 つはフォント・グローバル ID [FGID] のセクションです。

```
[CHARSET]
;charset = fgid, height, width, strikeover, underline
C?H200A0=2304,110,73,0,0
C?H200D0=2304,140,93,0,0
C?N200B0=2308,120,80,0,0
C?4200B0=416,120,144,0,0
C?D0GT15=230,80,96,0,0
C?A155A0=33207,110,73,0,0
C?A175A0=33227,110,73,0,0
C?T055D0=4407,140,93,0,0
C?T17500=4555,100,67,0,0
C?T17560=4555,60,40,0,0
DEFAULT =2308,80,0
```

図 14. [CHARSET] セクション : 文字セット定義ファイル (CSDEF.FNT) 内の文字セット [CHARSET] セクションの例

セクション・ヘッダー [CHARSET] によって識別される最初のセクションには、各 AFP フォント文字セットおよびこれに対応する以下の属性がリストされています。

- フォント・グローバル ID (fgid)
- フォントの高さ
- フォントの幅
- 重ね打ち
- 下線

表 13. [CHARSET] の文字セット定義ファイル属性値

属性	指定可能な値	出荷時のデフォルト	説明
Fgid	IBM 定義 FGID または次の範囲内のユーザー定義 FGID。3840 から 4095 あるいは 65260 から 65534	2308	タイプ・ファミリー、書体、および場合により文字セットのポイント・サイズを識別する固有の値。
高さ	1 から 990	80	10 分の 1 ポイント単位で表される文字セットの垂直方向のサイズ (最小限ではベースライン相互間の値)。例えば、9 ポイントのフォントの高さは 90 となります。

表 13. [CHARSET] の文字セット定義ファイル属性値 (続き)

属性	指定可能な値	出荷時のデフォルト	説明
幅	0 から 99 (現在では無視されます。)	0	1440 分の 1 インチ単位の文字の水平方向の平均サイズ。Windows では、フォントの高さに基づいて適切なフォントの幅が決定されるので、現在では常に 0 が使用されます。
重ね打ち	1 (「はい」の意味)、 0 (「いいえ」の意味)	0	フォントの文字すべての中央に、文字のベースラインに平行な線が引かれるフォント。
下線	1 (「はい」の意味)、 0 (「いいえ」の意味)	0	フォントの文字すべての下に、文字のベースラインに平行な線が引かれるフォント。

セクション・ヘッダー [FGID] によって識別される 2 番目のセクションには、各フォント・グローバル ID およびこれに対応する以下の属性がリストされています。

- フォント・タイプ・ファミリー
- スタイル
- ウェイト
- イタリック

```
[FGID]
;fgid = familyname, style, weight, italic
230=Gothic,MODERN,MED,0
416=Courier,MODERN,MED,0
2304=Helvetica,SWISS,MED,0
2308=TimesNewRoman,ROMAN,MED,0
4407=SonoranSerif,ROMAN,MED,0
4555=SonoranSerif,ROMAN,BOLD,1
33207=SonoranSansSerif,SWISS,MED,1
33227=SonoranSansSerif,SWISS,BOLD,1
```

図 15. [FGID] セクション： 文字セット定義ファイル (CSDEF.FNT) 内のフォント・グローバル ID [FGID] セクションの例

表 14. [FGID] の文字セット定義ファイル属性値

属性	説明	指定可能な値	出荷時のデフォルト
ファミリー名[1]	AFP タイプ・ファミリー名。	任意の AFP タイプ・ファミリー名	TimesNewRoman
スタイル[2]	Windows の「ファミリー」と同じ。 これは、AFP フォントのタイプ・ファミリーに書体スタイルを加えたものとほとんど同じです。	SWISS、[3] ROMAN、[4] SCRIPT、[5] MODERN、[6] DISPLAY[7]	ROMAN
ウェイト	図形文字を形成するストロークのさまざまな太さによって生じる書体の太さの程度。	LIGHT、MED、BOLD	MED
イタリック	文字が右側へ傾斜するフォント。	1 (「はい」の意味)、 0 (「いいえ」の意味)	0

表 14. [FGID] の文字セット定義ファイル属性値 (続き)

属性	説明	指定可能な値	出荷時のデフォルト
注: 1. 「ファミリー名」は、AFP フォントの「タイプ・ファミリー」、Windows の「書体名」と同じです。 2. 「スタイル」は、Windows の「ファミリー」と同じであり、AFP フォントの「書体スタイル」および「タイプ・ファミリー」と大体同じです。 3. SWISS は、プロポーショナル・スペース・フォントでセリフがありません。 4. ROMAN は、プロポーショナル・スペース・フォントでセリフがあります。 5. SCRIPT は、手書きに似た感じに見えるように設計された固定ピッチのフォントです。 6. MODERN は、固定ピッチのフォントで、セリフがある場合もない場合もあります。 7. DISPLAY は装飾的フォントです。			

文字セット定義ファイルの規則

- パラメーターは、コンマで区切ることが必要です。各パラメーターの指定可能な値および出荷時のデフォルト値が 235 ページの表 13 と 236 ページの表 14 にリストされています。
- ファイルの [CHARSET] セクションでは、`fgid` と `height` (ポイント・サイズ) だけが必須です。
- ファイルの [FGID] セクションでは、タイプの `familyname` と `style` だけが必須です。
- 疑問符 (?)。文字セット名の 2 番目の文字についてだけ、ワイルドカード文字として疑問符 (?) を使用できます。これによって、コード化フォントのすべての文字回転が検索の際に 1 つのエントリーで処理されるようになります。

  注: 文字セットについて順次検索が実行され、最初に一致したもの (ワイルドカード文字も含めて) が使用されます。
- [CHARSET] セクションは、[FGID] セクションよりも前でなければなりません。
- デフォルトの文字セットを設定できます。ファイル内に定義するデフォルトの文字セットは、[CHARSET] セクションの最後のエントリーにする必要があります。
- 独自の AFP フォント文字セットを [CHARSET] セクションに追加する場合、これにフォント・グローバル ID を割り当てる必要があります。 作成するフォント・グローバル ID は、3840 から 4095 または 65260 から 65534 の範囲内にする必要があります。新しい文字セットのファミリー名、スタイル、ウェイト、およびイタリック属性が既存の文字セットと同じである場合は、同じフォント・グローバル ID を使用できます。同じでない場合は、固有のフォント・グローバル ID を [FGID] セクションに追加することが必要です。

コード・ページ定義ファイル

コード・ページ定義ファイルによって、IBM AFP コード・ページ名がそのコード・ページ・グローバル ID (CPGID) および Windows 文字セットにマップされます。セクション・ヘッダー [CODEPG] のあとに、AFP コード・ページおよびそのパラメーターのリストが続いています。 各行の最初のパラメーターは、コード・ページ・マップ・ファイルにマップする AFP コード・ページ・グローバル ID です。(コード・ページのマッピング方法の詳細については、 238 ページの『コード・ペー

ジ・マップ・ファイル』を参照してください。) 2 番目のパラメーターは、使用 AFP コード・ページに最適なものとして決定する Windows 文字セットです。 最後の行では、デフォルトが必要なときに使用されるデフォルトのパラメーター値が提供されます。

```
[CODEPG]
;codepage = cpgid,wincp
T1DCDCFS=1003,ANSI
T1DEBASE=2058,ANSI
T1D0BASE=2063,ANSI
T1D0GP12=2085,ANSI
T1GI0395=2079,ANSI
T1GPI363=2066,SYMBOL
T1V10037=37,ANSI
T1V10273=273,ANSI
T1000290=290,ANSI
T1000310=310,ANSI
T1000423=423,ANSI
T1000905=905,ANSI
DEFAULT =361,ANSI
```

図 16. コード・ページ定義ファイル (CPDEF.FNT) の内容の例

表 15. コード・ページ定義ファイル属性値

属性	指定可能な値	出荷時のデフォルト
コード・ページ・グローバル ID	IBM 定義 CPGID、または 65280 から 65534 の範囲内の ユーザー定義 CPGID。	361
Windows 文字セット	ANSI または SYMBOL	ANSI

コード・ページ定義ファイルの規則

- パラメーターは、コンマで区切ることが必要です。各パラメーターの指定可能な値および出荷時のデフォルト値が表 15 にリストされています。
- 最初のパラメーター (コード・ページ ID) だけが必須です。
- 独自のコード・ページを作成する場合、これに固有のコード・ページ ID を割り当てる必要があります。先行ゼロは無効になります。(文字と 16 進コードとのマッピングが、作成するコード・ページと同じ場合にだけ、IBM コード・ページ・グローバル ID を使用できます。)
- デフォルトのコード・ページを設定できます。ファイル内に設定するデフォルトのコード・ページは、ファイル内の最後のエントリーにする必要があります。

コード・ページ・マップ・ファイル

Infoprint®、Data1 ライセンス・プログラム、および Sonoran ライセンス・プログラムで提供されるそれぞれの AFP コード・ページごとに、IBM は、1 つのコード・ページ・マップ・ファイルを提供します。 そのファイルは、FONT サブディレクトリーの下にある、各ファイルごとのサブディレクトリー (MAPS) にインストールされます。そのファイルのコード・ページ・グローバル ID (CPGID) を表す名前が、そのファイルには付けられ、ファイル拡張子は .cp になります (例えば、2063.cp は、T1D0BASE コード・ページ・マップのファイル名であり、その CPGID は 2063 です)。各ファイルには、IBM コード・ページの文字 ID (およびこれに関連し

た EBCDIC 16 進コード・ポイント) が含まれていて、各ファイルによって、その文字 ID が Windows の ANSI または SYMBOL 文字セットの文字 ID (およびこれに関連した ASCII コード・ポイント) にマップされます。

```
;T1000395   to    ANSI
SP010000 40    SP010000 20
LA150000 42    LA150000 E2
LA170000 43    LA170000 E4
LA130000 44    LA130000 E0
SP180000 8B    SP180000 BB
SM560000 8C    SM560000 89
SA000000 8D   SP100000 2D
LI510000 8E   NOMATCH 00
LF570000 8F   NOMATCH 00
SM190000 90    SM190000 B0
LJ010000 91    LJ010000 6A
LF510000 A0   NOMATCH 00
;;;;;;;; ;     SD150000 5E
;;;;;;;; ;     SD130000 60
;;;;;;;; ;     LT630000 FE
/*
```

図 17. コード・ページ・マップ・ファイル: Windows の ANSI 文字セットにマップされる T1000395 コード・ページ用のコード・ページ・マップ・ファイルの内容の一部を示した例

コード・ページ・マップ・ファイルの規則

- パラメーターは、ブランクで区切ることが必要です。
- 「NOMATCH」は、Windows 文字セットの中に一致する文字がないことを意味します。
- 「NOMATCH」の 16 進コード 00 は、未定義のコード・ポイントにマップされます。Windows 文字セットの中に存在しない文字が文書に含まれている場合、その文字は、画面上に表示できません。その文字がコード・ページ・マップ・ファイルまたは別名ファイル[4] の中に再マップされていない場合、未定義のコード・ポイントの文字は、置換文字で表示されます。未定義のコード・ポイントを表すために表示される文字は、「**設定 (Preferences)**」ダイアログ・ボックスで指定できます。
- セミコロンのストリング (;;;;;;;;) は、この行がコメントとして無視されることを意味します。また、このストリングは、IBM コード・ページの中に存在しない文字が Windows コード・ページに含まれていることも意味します。 IBM コード・ページの中にない Windows 文字のコード・ポイントは、NOMATCH の文字に置き換えるために使用できます。

コード・ページ・マップ・ファイル作成用のコード・ページ・マップ・ファイル REXX プログラム

コード・ページ・マップ・ファイルを作成するために使用できる再構造化拡張実行プログラム言語 (REXX 言語) のサンプル・プログラム (BLDCPMAP.REX) を、IBM は提供します。 このプログラムは、MVS™、OS/2®、OS/390、および Windows

4. コード・ポイントの再マップの詳細については、 241 ページの『別名ファイル』を参照してください。

REXX 環境で実行されます。 この REXX プログラムは、FONT サブディレクトリー[5] の SAMPLES サブディレクトリーに入っています。

BLDCPMAP.REX プログラムには、ホスト AFP コード・ページ、および Windows 文字セット・ファイル ANSI.WCP または SYMBOL.WCP[6] のうちの 1 つが必要です。 このプログラムの出力は、コード・ページ・マップ・ファイルです。このマップ・ファイルによって、ホストのコード・ページ内の文字は、Windows 文字セット内の一致する文字にマップされるので、そのコード・ページがクライアントで使用できるようになります。 また、このマップ・ファイルによって、コード・ページ内のアンマッチの文字数が確認され、その結果、どちらの Windows 文字セット (ANSI または SYMBOL) に一致する文字がより多く含まれているかを判断できます。 マッチングは、図形文字 ID を使用して行われます。

BLDCPMAP.REX EXEC を MVS システムまたは OS/390 システムで使用しようとする場合は、必ず、これを ASCII から EBCDIC に変換してください。また、復帰改行 (CR/LF) によって改行が指示される必要があります (詳しくは、BLDCPMAP.REX ファイルのプロローグを参照してください)。

コード・ページ・マップ・ファイルを作成するための設定

BLDCPMAP REXX プログラムと Windows 文字セット・ファイルをホスト・システムに転送し、そこでこのプログラムを実行できます。または、AFP コード・ページをワークステーションに転送し、Windows のもとでこのプログラムを実行することもできます (REXX がワークステーションにインストールされている場合)。ASCII から EBCDIC への変換を行って、または行わないで、標準ホスト・レコード・フォーマット・ファイルおよび ASCII CR/LF 行終了を扱う任意のファイル転送プログラムを使用できます。(IBM eNetwork Personal Communications プログラムを使用することをお勧めします。)

REXX プログラムと Windows 文字セット・ファイル (このファイルの拡張子は .WCP です。) を MVS または OS/390 ホスト・システムに転送する場合、これらを ASCII から EBCDIC に変換する必要があり、CR/LF によって改行が指示される必要があります。 ホスト・システムに転送されるすべてのファイルが、人間が理解できるものでなければなりません。AFP コード・ページを Windows ワークステーションに転送する場合、**バイナリー**・フォーマットを指定することが必要 です。ファイル転送を正しく行わないと、BLDCPMAP プログラムを実行したときに REXX エラーが発生します。

Windows ワークステーションで BLDCPMAP.REX ファイルを BLDCPMAP.CMD に名前変更し、REXX がインストールされていることを確認してください。 exec (通常は SYSEXEC または SYSPROC) を含むシステム・ファイルに、BLDCPMAP プログラムを含む区分データ・セットがすでに割り振られている場合、このプログラムは、MVS システムまたは OS/390 システム上で EXEC コマンドによって明示的に実行されることも、メンバー名によって暗黙的に実行されることもあります。REXX プログラムに正しい名前を付ければ、コマンドの正しい構文を取得するため

5. FONT サブディレクトリーは、クライアントをインストールしたディレクトリーにあります。

6. これらの Windows 文字セット・ファイルは、クライアントに付随していて、FONT サブディレクトリーの SAMPLES サブディレクトリーにあります。

のパラメーターを指定しないでこのプログラムを実行できます。 EXEC のプロローグを参照して構文について調べることもできます。

BLDCPMAP プログラムを実行するとき、クライアントで使用したいコード・ページ・マップ・ファイルをすでに選択している場合は、クライアントをインストールしたディレクトリーにある FONT サブディレクトリーの MAPS サブディレクトリーに、そのコード・ページ・マップ・ファイルを入れてください。FONT サブディレクトリーの CPDEF.FNT ファイルを更新してください。クライアントがコード・ページ・マップ・ファイルを見つけるためには、このファイルに次の名前を付ける必要があります。

```
code-page-global-identifier.CP
```

コード・ページ・マップ・ファイルの使用方法の詳細については、 238 ページの『コード・ページ・マップ・ファイル』を参照してください。BLDCPMAP プログラムの詳細 (例えば、このプログラムを実行するための構文など) については、BLDCPMAP.REX ファイルのプロローグを参照してください。

別名ファイル

別名ファイルには 2 つのセクション、つまり、フォント・ファミリー名の別名のセクション [FONT] および文字 ID の別名のセクション [CHARID] があります。

セクション・ヘッダー [FONT] によって識別される最初のセクションには、フォント・ファミリー名の別名がリストされています。フォント・ファミリー名の別名を使用すると、フォント・ファミリー名 (文字セット定義ファイルに定義されている) の要求されたインスタンスすべてを別のフォント・ファミリー名に変更できます。例えば、 242 ページの図 18 に示すように、このファイルを使用して、SonoranSerif フォント (ワークステーションに存在していない場合がある) に対するすべての要求を、TimesNewRoman フォント (クライアントに付随している中心的なフォント) に対する要求に変更します。

TrueType フォントをクライアントで使用できます。しかし、精度と文字のマッピングが不正確になる可能性があります。

注: AFP 文書の作成に使用したフォントと表示フォントが同じではないために、特に TrueType フォントの場合、フォント・ファミリー名の再マップが原因でテキスト文字が正しく位置合わせされなくなることがあるので注意してください。特性 (STYLE など) が非常に異なっている別のフォント・ファミリー名にフォント・ ファミリー名を再マップすると、一致するフォントが見つからないことになる場合があります。 両方のフォントの置換文字が見つからない場合は、エラー・メッセージが出ます。

```
[FONT]
; ***** Requested font = TrueType font  *****
Book=TimesNewRoman,Times New Roman
CourierOverstrike=Courier,Courier New
SonoranSerif=TimesNewRoman,Times New Roman
SonoranSansSerif=Helvetica,Arial
Text=Courier,Courier New
```

図 18. [FONT] セクション： [FONT] セクションのこの例は、別名ファイル (ALIAS.FNT) のセクションです。

セクション・ヘッダー [CHARID] によって識別される 2 番目のセクションには、文字 ID の別名がリストされています。文字 ID の別名 (絵文字 ID とも呼ばれる) を使用すると、文字の要求されたインスタンスすべてを別の文字に変更できます。例えば、Windows ANSI 文字セットには合字の ff (LF510000) は含まれていないので、この合字は、コード・ページ・マップ・ファイル (239 ページの図 17 を参照) 内の文字にマップされません。 代わりに、この合字は、NOMATCH 00 にマップされます。LF510000 — NOMATCH の対のオカレンスすべてを小文字の f にマップしたい場合、ALIAS.FNT ファイルの [CHARID] セクションに次のエントリーを入れると、そのようにマップできます。

```
LF510000=LF010000
```

ある特定のコード・ページの特定の 1 文字を変更したい場合は、 239 ページの図 17 に示すように、そのコード・ページのその文字を別の文字に再マップできます。

NOMATCH 00 が文字マッピングの中に見つかったときにだけ、別名ファイルが調べられます。

注: プログラムのパフォーマンスに影響が出るので、多数の文字を置換するために別名ファイルを使用することはお勧めしません。多数の文字を置換する必要がある場合は、使用しているコード・ページ・マップ・ファイル内のマッピングを直接変更するほうが適切です。

```
[CHARID]
LF510000=LF010000
SA000000=SP320000,SP100000
```

図 19. [CHARID] セクション： [CHARID] セクションのこの例は、別名ファイル (ALIAS.FNT) のセクションです。

別名ファイルの規則

- ファミリー名の別名の場合、文字セット定義ファイル内の最初のファミリー名に対するすべての要求には、それに置換される第 2 のファミリー名があります。第 2 のファミリー名が見つからない場合は、TrueType font (第 3 のファミリー名) が要求されます。
- 1 行につきファミリー名の置換が 2 つだけ許可され (等号の右側に)、これらは、コンマで区切る必要があります。
- 同じファミリー名について複数のマッピングがファイル内にリストされている場合、最初に一致したものだけが使用されます。

- 別名ファイルは逐次処理され、チェーンされません (例えば、「Century Schoolbook」は「Times」に等しいと設定され、「Times」は「TimesNewRoman」に等しいと設定されても、「Century Schoolbook」が「TimesNewRoman」に設定されることはありません)。
- ファミリー名の中のブランクは、文字として扱われます (例えば、「Times New Roman」は「TimesNewRoman」と同じフォントではありません)。
- 2 番目の文字 ID が NOMATCH 00 である場合にだけ、別名ファイルの [CHARID] セクションが使用されます。
- [CHARID] セクションで変更されることを望む文字 ID のあとに、等号、およびその文字 ID の変更後の文字 ID が続くことが必要です。変更される文字 ID ([CHARID] セクションの等号の左側にある文字) が、コード・ページ・マップ・ファイル内で一致しない対の最初の文字 ID と一致すると、文字の再マップが行われます。
- いくつかの文字 ID (置換文字 ID) を、等号の右側にコンマで区切ってリストできます。変更される文字 ID が Windows フォントの中に存在しない場合、この文字 ID は、最初の置換文字 ID で置換されます。 その置換文字 ID が存在しない場合は、次の置換文字 ID が使用されます。置換文字 ID がまったく存在しない場合は、未定義のコード・ポイントが使用されます。Windows 文字セットの内容を見るには、FONT ディレクトリーの SAMPLES サブディレクトリーにある .WCP ファイルを参照してください。
- 置換文字 ID は最高 4 つまで認められます。

TrueType フォントのサポート

TrueType フォント

TrueType フォントを使用して文書を表示できます。特定の TrueType フォントを要求するには、 241 ページの『別名ファイル』の説明に従って、ALIAS.FNT ファイル内の 2 番目のフォント置換ファミリー名を使用してください。

TrueType フォント置換に関する問題

要求した TrueType フォントがワークステーションにインストールされていることを確認してください。フォントが使用可能でないときに行われるフォント置換が原因で、ファイルを表示するときに予期しない結果が生じることもあります。例えば、Courier New が ALIAS.FNT ファイル内で要求されて、Windows NT® では使用可能でも Windows 2000 では使用可能でない場合、同じ文書がこれら 2 種類のシステム間で異なった見えかた (表示のされかた) となることがあります (ただし、Windows 2000 システムにこのフォントをインストールすることはできます)。

第 20 章 トラブルシューティング

StoreDoc() API はエラー・コード 2 を戻す

症状

StoreDoc() API はエラー・コード 2 を戻します。

考えられる原因

StoreDoc() API が文書を正しくロードするには、次のように、2 つの属性が設定されていなければなりません。

- StoreDoc() が正常に実行されるためには、「アプリケーション・グループ (Application Group)」のデータベース編成が、属性 Multiple Loads per Table を持っていなければなりません。この属性は、アプリケーション・グループの「プロパティー」ウィンドウの「一般/拡張 (General/Advanced)」タブで設定します。
- StoreDoc() が正常に実行されるためには、「ストレージ管理」内の「アプリケーション・グループの満了タイプ (Application Group's Expiration Type)」が、「満了タイプ: セグメント (Expiration Type: Segment)」に設定されていなければなりません。この属性は、アプリケーション・グループの「プロパティー」ウィンドウの「ストレージ管理」タブで設定します。

アクション

アプリケーション・グループの 2 つの属性が正しく設定されているか確認してください。

ヒント

ユーザーが、ワークベンチまたはナビゲーション・プラグインで AFP ファイルをオープンせずに印刷するには

ユーザーが、ワークベンチまたはナビゲーション・プラグインで AFP ファイルをオープンせずに 印刷するには、npoafp32.dll を使用しないで、 Content Manager OnDemand OLE コントロールを使用する必要があります。

付録 A. Microsoft Visual Basic 5.0 DDE サンプル・プログラム

このサンプル・プログラムは、現状のままで提供します。OnDemand 製品のライセンス所有者は、適宜、このサンプル・プログラムのコピー、改訂、変更、および関連作業を自由に行うことができます。

このプログラムは、Microsoft® Visual Basic 5.0 を使用して作成され、コンパイルされています。このプログラムによって、これらの OnDemand DDE コマンドを例証します。

ACTIVATE_DOC
CLOSE_ALL_DOCS
ENABLE_SWITCH
EXIT
GET_DOC_VALUES
GET_NUM_DOCS_IN_LIST
LOGON
OPEN_DOC
OPEN_FOLDER
SEARCH_FOLDER
SET_FIELD_DATA
SET_FOCUS
SHOW_WINDOW

サンプル・プログラムによって使用されるグローバル変数

このセクションでは、サンプル・プログラムによって使用されるグローバル変数を定義します。

```
Option Explicit
Global Const apptitle = "OnDemand VB Demo"
Global Const yes = 1
Global Const no = 0
Global Const leave = 100000#

Global Const arstopic = "ARS|ARS"     'Default DDE application at topic

Global Const defini = "arsvblan.ini"  'Default ini file name
Global Const defstanza = "VBDEMO"     'Default stanza
Global Const arsgui = "ARSGUI32.EXE"  'Default Client exe

'Default Client parms
' Default Client startup parms
' /I   = enable DDE interface
' /B   = disable user confirmation
' /W N = invisible window (don't use during debugging)
' /K   = disable Exit (don't use during debugging)
' /V   = disable anticipation
Global Const arsguiopts = " /I /B /V /W N /K"

Global ininame As String              'Ini file name
Global server As String               'Server name
```

```
Global userid As String               'userid
Global pass As String                 'password
Global folder As String               'folder

Global guipath As String              'Client exe path
Global Const linktype = 2             'DDE linkage = manual
Global Const linktime = 3000          'DDE wait time

Global doc_ids(0 To 2) As String      'Doc ids returned on OPEN_DOC

Global txtStack(1 To 10) As String

'Define the Windows APIs used by the program
Declare Function GetModuleHandle Lib "Kernel" (ByVal lpModuleName As String) As Integer
Declare Function GetPrivateProfileString Lib "Kernel" (ByVal sname$, ByVal Kname$,
 ByVal Def$, ByVal ret$,ByVal Size%, ByVal Fname$) As Integer
Declare Function SetFocusAPI Lib "User" Alias "SetFocus" (ByVal hWnd As Integer) As Integer
```

サンプル・プログラム用のエントリー・ポイント

このサンプル・プログラムでは、DDE インターフェースを使用して OnDemand Windows クライアントを駆動します。 注意すべき重要事項があります。

- OnDemand クライアントに送信されるコマンドは DDE EXECUTE ではなく、すべて DDE REQUEST です。 OnDemand クライアントに DDE EXECUTE を送信すると結果的にエラーが起きるので、このことは重要です。
- また、少なくとも /I オプションを指定して OnDemand クライアントを開始することが重要です。このオプションによって、OnDemand クライアントの DDE インターフェースが使用可能になります。(どのようなオプションが VBDEMO に使用されたかを知るには、globals セクションを参照してください。)

このサンプル・プログラムは、Visual Basic 5.0 (32 ビット) を使用して作成されています。OnDemand クライアント・ウィンドウとの DDE 会話の中で、DDE クライアントとして txtDemo コントロール (これは隠れている) が使用されます。DDE 要求を実行するためにこのコントロールを設定した場所、およびデータがこのコントロールに戻ることが、fncDDElink() の中を見ると分かります。したがって、OnDemand クライアント・ウィンドウで戻されるデータは、このコントロールの外部から解析されることが必要になります。

```
Sub Main()
    Dim rc As Integer

    'Initialize globals and read in data from ini file(s).
    Call fncInit

    frmStatusDlg.lblStatus.Caption = "Starting client..."
    frmStatusDlg.Show 0

    'Start OnDemand client.
    Shell (guipath + arsgui + arsguiopts)

    'Logon to the server (logon information was gathered from ini
    'file during fncInit. User cannot do anything else while this
    'is going on. Try and use SetFocusAPI() to restore focus to our
    'status message while this is going on.
    frmStatusDlg.lblStatus.Caption = "Logging on to Server..."
    Call frmCreditV1.fncLogon
    rc = SetFocusAPI(frmStatusDlg.hWnd)

    'Open the "Baxter Bay Credit" folder
```

```
    frmStatusDlg.lblStatus.Caption = "Opening folder..."
    Call frmCreditV1.fncOpenFolder

    'Don't need the status message box any more.
    frmStatusDlg.Hide

    'Only after we have logged on and opened the folder do we
    'display the VBDEMO form.
    frmCreditV1.Show 1

End Sub

'Send DDE REQUEST of CLOSE_ALL_DOCS to the client window:

Private Sub fncCloseDoc()
    Dim cmdline, qrc As String

    Call fncDispStatus("Close all open docs...")
    cmdline = "CLOSE_ALL_DOCS"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        Call quit(cmdline, qrc)
    End If

    Call fncDispStatus("All open docs closed.")
End Sub

'This procedure handles the link to the OnDemand client.
' Topic should come is as ARS|ARS, this is the app name and topic name.

Private Sub fncDDElink(ByVal topic As String, ByVal cmnd As String,
   ByVal mode As Integer, ByVal waittime As Integer)
'Setup local variables
    Dim sync, lntxtDemo, i, rc As Integer
    Dim workchar, workline, msg As String
'Set up error handler to show contact errors
    On Error GoTo HandleError
'Set up DDE link and pass required data:
    txtDemo = "-"
    txtDemo.LinkTimeout = waittime
    txtDemo.LinkTopic = topic
    txtDemo.LinkItem = cmnd
    txtDemo.LinkMode = mode
    'Calling LinkRequest performs the request.
    txtDemo.LinkRequest
Exit Sub

HandleError:
    'Handle DDE errors
    rc = Err
    Select Case rc
        Case 280 To 297
        Select Case rc
          Case 280
            msg = "DDE channel not closed; awaiting response from foreign application"
          Case 281
            msg = "No more DDE channels"
          Case 282
            msg = "DDE requests are being refused"
          Case 283
            msg = "Too many apps responded"
          Case 284
            msg = "DDE channel locked"
          Case 285
            msg = "App is not accepting DDE requests...' "
        End Select
```

```
        Case Else
          msg = "Non-DDE error occurred " + Str(rc)
    End Select
    MsgBox msg
    Resume Next
End Sub

'Used to send DDE REQUEST command of ACTIVATE_DOC or OPEN_DOC to
' the OnDemand client.

Private Sub fncDispDoc(ByVal docnum As Integer)
    Dim cmdline, qrc As String

    'If the document the user is requesting to be displayed has
    'previously been opened, then use ACTIVATE_DOC to redisplay
    'it, otherwise we will need to OPEN_DOC and store away the
    'document id.
    'If the user closes the document view from the client interface
    'we need to be notified of this event so that we can update
    'our doc_id array. Currently the client does not support this.

    If doc_ids(docnum) <> "0" Then
        Call fncDispStatus("Activating the document...")
        cmdline = "ACTIVATE_DOC /D " + doc_ids(docnum)
        Call fncDDElink(arstopic, cmdline, linktype, 3000)
        qrc = fncGetrc(txtDemo)
        If qrc <> "0" Then
            'The user possibly closed the view from the client,
            'reset the document id to 0 and tell the user to try again.
            doc_ids(docnum) = "0"
            Call fncDispStatus("Activating the document...ERROR")
            MsgBox "Could not activate, try to view again!"
            Exit Sub
        End If
        Call fncDispStatus("Activating the document...done")
    Else
        'Open the document
        Call fncDispStatus("Open the document...")
        cmdline = "OPEN_DOC /N " + Str(docnum)
        Call fncDDElink(arstopic, cmdline, linktype, 3000)
        qrc = fncGetrc(txtDemo)
        If qrc <> "0" Then
            Call quit(cmdline, qrc)
        End If
        doc_ids(docnum) = fncGetdochandle(txtDemo)
        Call fncDispStatus("Open the document...done.")
    End If

'Make the display visible
    Call fncDispStatus("Opening the display...")
    cmdline = "SHOW_WINDOW /W"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        Call quit(cmdline, qrc)
    End If
    Call fncDispStatus("Opening the display...done.")

    Call fncDispStatus("Document retrieval complete.")
End Sub

'Obtains the hitlist of documents from the OnDemand client.
Private Sub fncGetHitlist()
    Dim cmdline, qrc As String
    Dim num_docs As Integer

    'Get the number of documents which matched the search
```

```
    'criteria.
    cmdline = "GET_NUM_DOCS_IN_LIST"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    num_docs = CInt(Mid(txtDemo, 3, 10))
    If num_docs = 0 Then
        MsgBox "No documents found matching search criteria!"
        Exit Sub
    End If

    Call fncDispStatus("Getting account information...")

    'Get the first document and parse its data to display
    cmdline = "GET_DOC_VALUES /N 0"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    Call fncExtract(txtDemo.Text)
    'Display its data
    pnlPayData1.Caption = txtStack(1)
    Panel3D1.Caption = txtStack(4)
    'Add about 20 days or so to the statement date'
    Panel3D2.Caption = fncParseDate(txtStack(1))
    cmdViewStmt1.Enabled = True

    'If there are at lease two documents then get number 2
    If num_docs > 1 Then
        cmdline = "GET_DOC_VALUES /N 1"
        Call fncDDElink(arstopic, cmdline, linktype, 3000)
        Call fncExtract(txtDemo.Text)
        'Display its data
        pnlPayData2.Caption = txtStack(1)
        Panel3D3.Caption = txtStack(4)
        'Add about 20 days or so to the statement date'
        Panel3D4.Caption = fncParseDate(txtStack(1))
        cmdViewStmt2.Enabled = True
    Else
        'There was only 1 document so disable 2nd "View" button.
        cmdViewStmt2.Enabled = False
    End If

'If there are at lease three documents then get number 3
    If num_docs > 2 Then
        cmdline = "GET_DOC_VALUES /N 2"
        Call fncDDElink(arstopic, cmdline, linktype, 3000)
        Call fncExtract(txtDemo.Text)
        'Display its data
        pnlPayData3.Caption = txtStack(1)
        Panel3D5.Caption = txtStack(4)
        'Add about 20 days or so to the statement date'
        Panel3D6.Caption = fncParseDate(txtStack(1))
        cmdViewStmt3.Enabled = True

Else
        'There were only 2 documents so disable 3rd "View" button.
        cmdViewStmt3.Enabled = False
    End If
    Call fncDispStatus("Getting account information...done.")
End Sub

'Procedure used to hide the OnDemand client window.
'Sends a DDE REQUEST message of SHOW_WINDOW to the client.
Private Sub fncHideWindow()
    Dim cmdline, qrc As String

    cmdline = "SHOW_WINDOW /W N"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        Call quit(cmdline, qrc)
```

```
    End If
End Sub

'Logon to the OnDemand client.
Public Sub fncLogon()
    Dim cmdline, qrc As String

    Call fncDispStatus("Logon to Client...")
    cmdline = "LOGON /S " + server + " /U " + userid + " /P " + pass
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        'If we fail the logon the client will display his logon dialog.
        'We will not return from this DDE call until the user either
        'successfully log's onto a server or cancel's the process, in
        'which case we end up with an error code and inside of this If
        'statement. Close the client and then ourselves.
        'I am not sure if the above statement is valid if you started up
        ' the OnDemand client with the Disable anticipation (/V) parameter.
        Call fncDDElink(arstopic, "EXIT", linktype, 3000)
        Call fncDispStatus("Logon to client...failed.")
        End
    End If
    Call fncDispStatus("Logon to Client...done.")
End Sub

'Open up an OnDemand folder.
Public Sub fncOpenFolder()
  Dim cmdline, qrc As String

  Call fncDispStatus("Open the folder...")
  cmdline = "OPEN_FOLDER /F " + folder
  Call fncDDElink(arstopic, cmdline, linktype, 3000)
  qrc = fncGetrc(txtDemo)
  If qrc <> "0" Then Call quit(cmdline, qrc)
  Call fncDispStatus("Open the folder...done.")

End Sub

'Search the OnDemand folder for documents.
Private Sub fncSearchDoc(ByVal AcctNum As String)
    Dim cmdline, qrc As String

    'Setup our search fields with the client.
    Call fncDispStatus("Setting up Search...")
    cmdline = "SET_FIELD_DATA /F Account /1 " + AcctNum
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        Call quit(cmdline, qrc)
    End If
    Call fncDispStatus("Setting up Search...done.")

    'Have the client perform the search.
    Call fncDispStatus("Performing the Search...")
    cmdline = "SEARCH_FOLDER"
    Call fncDDElink(arstopic, cmdline, linktype, 3000)
    qrc = fncGetrc(txtDemo)
    If qrc <> "0" Then
        Call quit(cmdline, qrc)
    End If
    Call fncDispStatus("Performing the Search...done.")
End Sub

'Performs three DDE steps for us:
'    - Inform client to retrieve selected document.
'    - Enable the switch back toolbar button on the clients toolbar so that the
```

```
'      user can get back easily to the VBDEMO.
'    - Switch focus to the client.
Private Sub fncViewDoc(ByVal docnum As Integer)
    'Setup local variables
    Dim MyHandle As Integer

    'Display the document
    Call fncDispDoc(docnum)
    'Activate DDE and transfer Focus to OnDemand
    MyHandle = frmCreditV1.hWnd
    Call fncDDElink(arstopic, "ENABLE_SWITCH /H " + Str(MyHandle) + " /C " + apptitle,
 linktype, 3000)
    Call fncDDElink(arstopic, "SET_FOCUS", linktype, 3000)
End Sub

'Displays error code.
Private Sub quit(ByVal qinfo As String, ByVal qrc As String)
    Dim quitstring As String

    quitstring = "Error encountered: " + qinfo + " rc=" + qrc
    MsgBox quitstring
    End
End Sub

'GUI control used to display customer information.
'We do not obtain the customer information from out of OnDemand, it is
'  not stored there. The normal way to obtain this information would be
'  to get it out of your business database. After which you would look up
'  the customer statements in OnDemand. We simply get this information from
'  out of an ini file.
Private Sub cmdCustInfo_Click()
    Dim acct_num, ini_str As String
    Dim cmdline, qrc As String
    Dim rc As Integer
    Dim first_num, second_num, third_num As Integer

    'Zero out the Payment record fields before retrieving new customer
    pnlPayData1.Caption = ""
    Panel3D1.Caption = ""
    Panel3D2.Caption = ""
    pnlPayData2.Caption = ""
    Panel3D3.Caption = ""
    Panel3D4.Caption = ""
    pnlPayData3.Caption = ""
    Panel3D5.Caption = ""
    Panel3D6.Caption = ""
    'Zero out the Customer Information fields.
    pnlNameData.Caption = ""
    pnlSSNData.Caption = ""
    pnlDOBData.Caption = ""
    pnlMNameData.Caption = ""
    pnlAddrData1.Caption = ""
    pnlAddrData2.Caption = ""
    pnlPhoneData.Caption = ""

'Disable "View" buttons
    cmdViewStmt1.Enabled = False
    cmdViewStmt2.Enabled = False
    cmdViewStmt3.Enabled = False

    'Hide client window
    Call fncHideWindow

    'Look up the account number, contained in the pnlAcctnumData text field
    'in the arsvblan.ini file. If found, read the respective
    'fields. If not found display error message.
    acct_num = txtAcctnumData.Text
```

```
    'Do at least a little validation.
    If Len(acct_num) <> 11 Then
        MsgBox "Correct format for account # is 000-000-000"
        Exit Sub
    End If

'If we have gotten to here we know that we have an account
    'number of the format 000-000-000. If either of the first
    'two sections of the number are non-zero or if the third
    'section is not between 001-046 then default to the account
    'number 000-000-001.
    first_num = Int(Mid(acct_num, 1, 3))
    second_num = Int(Mid(acct_num, 5, 3))
    third_num = Int(Mid(acct_num, 9, 3))
    If first_num <> 0 Or second_num <> 0 Or third_num > 46 Then
        acct_num = "000-000-001"
    ElseIf third_num = 0 Then
        MsgBox "Invalid account number!"
        Exit Sub
    End If

    ini_str = fncParmGet(acct_num, "Name", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'Name' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlNameData.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "SSN", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'SSN' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlSSNData.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "DOB", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'DOB' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlDOBData.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "MaidenName", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'MaidenName' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub

End If
    pnlMNameData.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "StreetAddress", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'StreetAddress' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlAddrData1.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "CityStateZip", ininame)
    If Len(ini_str) = 0 Then
        MsgBox "'CityStateZip' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlAddrData2.Caption = "  " + ini_str

    ini_str = fncParmGet(acct_num, "PhoneNum", ininame)
    If Len(ini_str) = 0 Then
```

```
        MsgBox "'PhoneNum' field not found for acct#" + acct_num + "in " + ininame
        Exit Sub
    End If
    pnlPhoneData.Caption = "  " + ini_str

'We are changing customer accounts so before we get new customer
    'information, close old customers open documents.
    If doc_ids(0) <> "0" Or doc_ids(1) <> "0" Or doc_ids(2) <> "0" Then
        doc_ids(0) = "0"
        doc_ids(1) = "0"
        doc_ids(2) = "0"
        cmdline = "CLOSE_ALL_DOCS"
        Call fncDDElink(arstopic, cmdline, linktype, 3000)
        qrc = fncGetrc(txtDemo)
        If qrc <> "0" Then
            Call quit(cmdline, qrc)
        End If
    End If

    'Set up the search fields and perform search.
    Call fncSearchDoc(acct_num)

    'Get the 3 most recent statements.
    Call fncGetHitlist

    'Give ourselves back the focus
    rc = SetFocusAPI(frmCreditV1.hWnd)
End Sub

'User has chosen to exit the VBDEMO. Before exiting close down the client.
Private Sub cmdExit_Click()
    Dim OnDemandHandle As Integer

    Call fncDispStatus("Program ending...")

    'Determine if OnDemand is loaded
    OnDemandHandle = GetModuleHandle(arsgui)
    'If not loaded, then quit, else shutdown
    If OnDemandHandle > 0 Then
        Call fncDispStatus("Shutting Client Down...")
        Call fncDDElink(arstopic, "EXIT", linktype, linktime)
    End If

    'Terminate the VBDEMO
    End
End Sub

'View button number 1. Have the client retrieve the first document in
'  the hitlist and display it.
Private Sub cmdViewStmt1_Click()

  Call fncViewDoc(0)

End Sub

'View button number 2. Have the client retrieve the second document in
'  the hitlist and display it.
Private Sub cmdViewStmt2_Click()

  Call fncViewDoc(1)

End Sub

'View button number 3. Have the client retrieve the third document in
'  the hitlist and display it.
Private Sub cmdViewStmt3_Click()
```

```
  Call fncViewDoc(2)

End Sub

'If the user is not using the Exit button to close down the demo
'  this function will be called as a result of the form being unloaded
'  so go ahead and shut down the client then exit.
Private Sub Form_Unload(Cancel As Integer)
    Dim OnDemandHandle As Integer

    Call fncDispStatus("Program ending...")

    'Determine if OnDemand is loaded
    OnDemandHandle = GetModuleHandle(arsgui)
    'If not loaded, then quit, else shutdown
    If OnDemandHandle > 0 Then
        Call fncDispStatus("Shutting Client Down...")
        Call fncDDElink(arstopic, "EXIT", linktype, linktime)
    End If

    'Terminate the VBDEMO
    End
End Sub

'Make sure that the data they are entering for the account number
'  is valid.
Private Sub txtAcctnumData_KeyPress(KeyAscii As Integer)

    Dim pos As Integer

    pos = txtAcctnumData.SelStart

    Select Case KeyAscii
        Case 48 To 59
            'pos must be 0-2, 4-6, 8-10
            Select Case pos
                Case 0 To 2, 4 To 6, 8 To 10
                    'OK
                Case Else
                    Beep
                    KeyAscii = 0
            End Select
        Case 45              ' the - character
            'pos must be 3 or 7
            If pos <> 3 And pos <> 7 Then
                Beep
                KeyAscii = 0
            End If
        Case 8, 127
            'Just let these through.
        Case Else
            Beep
            KeyAscii = 0
    End Select

End Sub

'This procedure fills in the status line on the form and left adjusts.
Public Sub fncDispStatus(ByVal status As String)
    frmCreditV1.pnlStatus.Caption = status + Space$(255)
End Sub

'This procedure breaks out the words of the input
'string. Words must be delimited by a tab character.
'The words are stored in the global string array txtStack.
Sub fncExtract(ByVal workstring As String)
    Dim txtptr, lenstring, i As Integer
```

```
    Dim tabchar, workline, workchar As String

    txtptr = 0
    tabchar = Chr(9)
    workline = ""
    lenstring = Len(workstring)
    workstring = Mid$(workstring, 3, lenstring)

'Extract chars to the first blank
    For i = 1 To lenstring
        workchar = Mid$(workstring, i, 1)
        'When a tab is found, store result, reset
        If workchar = tabchar Then
            txtptr = txtptr + 1
            txtStack(txtptr) = workline
            workline = ""
        'Otherwise, keep building the work string
        Else
            workline = workline + workchar
        End If
    Next

    If Len(workline) > 0 Then
        txtptr = txtptr + 1
        txtStack(txtptr) = workline
    End If
End Sub

'This function extracts out the document handle from the
'return string.
Function fncGetdochandle(ByVal workstring As String)
    Dim lenstring, first, i As Integer
    Dim rc, workline, workchar As String

'Set the return code for invalid function call
    rc = "999"
    first = yes
    workstring = Trim$(workstring)
    lenstring = Len(workstring)

'Extract chars to the first blank
    If lenstring > 0 Then
        workline = ""
        For i = 1 To lenstring
            workchar = Mid$(workstring, i, 1)
            'When a second blank is found, stop
            If workchar = " " Then
                If first = yes Then
                    first = no
                    workline = ""
                Else
                    rc = workline
                    i = leave
                End If
            'Otherwise build up return code
            Else
                workline = workline + workchar
            End If
        Next
        'If the doc handle has been built, assign it
        If workline <> "" Then
            rc = workline
        End If
    End If

    'Set the function return value
    fncGetdochandle = rc
```

```
End Function

'This function extracts out the return code from the
'return string.
Function fncGetrc(ByVal workstring As String)
    Dim lenstring, i As Long
    Dim rc, workline, workchar As String

    'Set the return code for invalid function call
    rc = "999"
    workstring = Trim$(workstring)
    lenstring = Len(workstring)

    'Extract chars to the first blank
    If lenstring > 0 Then
        workline = ""
        For i = 1 To lenstring
            workchar = Mid$(workstring, i, 1)
            'When a blank is found, stop
            If workchar = " " Then
                rc = workline
                i = leave
            'Otherwise build up return code
            Else
                workline = workline + workchar
            End If
        Next
        'If a return code has been built, assign it
        If workline <> "" Then
            rc = workline
        End If
    End If

    'Set the function return value
    fncGetrc = rc

End Function

'Perform global initialization
Sub fncInit()
    Dim ini_str As String

    'Set document ids for all three to 0
    doc_ids(0) = "0"
    doc_ids(1) = "0"
    doc_ids(2) = "0"

    'Disable "View" buttons
    frmCreditV1.cmdViewStmt1.Enabled = False
    frmCreditV1.cmdViewStmt2.Enabled = False
    frmCreditV1.cmdViewStmt3.Enabled = False

    'The VBDEMO keyword in the PATHS stanza of the ars.ini file
    'points to the .ini file where the other
    'demo settings can be picked up. If the
    'VBDEMO keyword cannot be found, the ini
    'file is set to arsvblan.ini.

    'Try to find vbdemo inifile name
    ininame = defini
    ini_str = fncParmGet("PATHS", "VBDEMO", "ars.ini")
    'If the ini name is found, then set
    If Len(ini_str) > 0 Then
        ininame = ini_str
    End If
```

```
    'Try to find arsgui execution path
    ini_str = fncParmGet(defstanza, "GUIPath", ininame)
    'If it can't be found, check for an env var
    If Len(ini_str) = 0 Then
        MsgBox "Cannot find GUIPath in " + ininame
    End If

    'If the path is found, then set
    If Len(ini_str) > 0 Then
        guipath = ini_str + "¥"
    End If

    'Try to find the server in the ars ini file
    ini_str = fncParmGet(defstanza, "Server", ininame)
    'If it can't be found, check for an env var
    If Len(ini_str) = 0 Then
        MsgBox "Cannot find Server in " + ininame
    End If
    If Len(ini_str) > 0 Then
        server = ini_str
    End If

'Try to find the userid in the ars ini file
    ini_str = fncParmGet(defstanza, "Userid", ininame)
    'If it can't be found, check for an env var
    If Len(ini_str) = 0 Then
        MsgBox "Cannot find Userid in " + ininame
    End If
    If Len(ini_str) > 0 Then
        userid = ini_str
    End If

    'Try to find the password in the ars ini file
    ini_str = fncParmGet(defstanza, "Password", ininame)
    'If it can't be found, check for an env var
    If Len(ini_str) = 0 Then
        MsgBox "Cannot find Password in " + ininame
    End If
    If Len(ini_str) > 0 Then
        pass = ini_str
    End If
    If pass = "<NULL>" Then
        pass = ""
    End If

    'Try to find the folder in the ars ini file
    ini_str = fncParmGet(defstanza, "Folder", ininame)
    folder = ini_str

End Sub
'This function returns information from the ini file.
Function fncParmGet(ByVal stanza As String, ByVal keyname As String, ByVal inifile As String)
    Dim Default, result As String
    Dim rc As Integer

    Default = ""
    result = Space$(255)

    rc = GetPrivateProfileString(stanza, keyname, Default, result, Len(result), inifile)
    If rc Then
        fncParmGet = Trim$(result)
        If Len(fncParmGet) > 1 Then
            fncParmGet = Left$(fncParmGet, Len(fncParmGet) - 1)
        End If
    Else
        fncParmGet = ""
    End If
```

```
End Function

'This function is only used to dummy up the date paid
'field of the form. The reason being is that for the
'demo, which uses the 'Baxter Bay Credit' folder,
'we cannot get this information from the database.
'This function adds approximately 20 days to the statement
'date field (which is passed in).
Public Function fncParseDate(ByVal stmtdate As String)
    Dim date_array(1 To 3) As String
    Dim searchch, workline, workchar As String
    Dim txtptr, lenstring, i As Integer
    Dim pay_day, pay_month, pay_year As Integer

    txtptr = 0
    searchch = Chr(47)
    workline = ""
    lenstring = Len(stmtdate)

'Extract chars to the first '/'
    For i = 1 To lenstring
        workchar = Mid$(stmtdate, i, 1)
        'When a '/' is found, store result, reset
        If workchar = searchch Then
            txtptr = txtptr + 1
            date_array(txtptr) = workline
            workline = ""
        'Otherwise, keep building the work string
        Else
            workline = workline + workchar
        End If
    Next

    If Len(workline) > 0 Then
        txtptr = txtptr + 1
        date_array(txtptr) = workline
    End If

'date_array contains three elements, the first is the month
    'number, the second is the day of the month and third is
    'the year. Simply check if the day of the month plus 20
    'is greater than 28, if so the difference becomes the new
    'day of the month and we increment the month number.
    pay_day = Int(date_array(2)) + 20
    pay_month = Int(date_array(1))
    pay_year = Int(date_array(3))
    If pay_day > 28 Then
        pay_day = pay_day - 28
        pay_month = pay_month + 1
        If pay_month > 12 Then
            pay_month = 1
            pay_year = pay_year + 1
        End If
    End If

    fncParseDate = LTrim(Str(pay_month)) + "/" + LTrim(Str(pay_day)) + "/" + LTrim(Str(pay_year))
End Function
```

付録 B. Microsoft Visual C++ 5.0 DDE サンプル・プログラム

このサンプル・プログラムは、現状のままで提供します。OnDemand 製品のライセンス所有者は、適宜、このサンプル・プログラムのコピー、改訂、変更、および関連作業を自由に行うことができます。

このプログラムは、Microsoft VC++ 5.0 を使用して作成され、コンパイルされています。このプログラムによって、これらの OnDemand DDE コマンドを例証します。

- CLOSE_DOC
- CLOSE_FOLDER
- ENABLE_SWITCH
- EXIT
- GET_DOC_VALUES
- GET_NUM_DOCS_IN_LIST
- GET_PRINTERS
- LOGOFF
- LOGON
- OPEN_DOC
- OPEN_FOLDER
- PRINT_DOC
- SEARCH_FOLDER
- SHOW_WINDOW

```
#include "stdafx.h"
#include <ddeml.h>
#include <winspool.h>

#include "vcdde32.h"
#include "MainDlg.h"
#include "arsddeex.h"                       // Shipped with OnDemand

static CMainDlg * pMainDlg;
static char       RequestedData[10000];          // Returned data from DDE command
static HSZ        hsz1, hsz2;
       DWORD      DdeInstance;
       HCONV      hDdeConv;

extern CDdeTestApp * pApp;                  // Pointer to application instance

#define ERROR_MAP struct _ErrorMap
ERROR_MAP
{
  int    code;
  char * pMsg;
};

static ERROR_MAP Errors[] =
  { { ARS_DDE_RC_UNKNOWN_COMMAND,           "Unknown command." },
    { ARS_DDE_RC_PARM_NOT_SPECIFIED,        "Parameter not specified." },
    { ARS_DDE_RC_INVALID_PARM_VALUE,        "Invalid parameter value." },
    { ARS_DDE_RC_SERVER_ERROR,              "Server error." },
    { ARS_DDE_RC_FILE_ERROR,                "File error." },
    { ARS_DDE_RC_NOT_LOGGED_ON,             "Not logged on." },
    { ARS_DDE_RC_MAX_FOLDERS_OPEN,          "Maximum folders open." },
```

```
    { ARS_DDE_RC_FOLDER_NOT_OPEN,            "Folder not open." },
    { ARS_DDE_RC_NO_DOC,                     "No document exists." },
    { ARS_DDE_RC_NO_ACTIVE_DOC,              "No document is active." },
    { ARS_DDE_RC_USER_ACTION_IN_PROGRESS,    "User action in process." },
    { ARS_DDE_RC_UNAUTHORIZED_OPERATION ,    "Unauthorized operation." },
    { ARS_DDE_RC_USER_CANCELLED_OPERATION,   "User cancelled operation." },
    { ARS_DDE_RC_INVALID_APPL_GROUP_NAME,    "Invalid Appl Group Name." },
    { ARS_DDE_RC_INVALID_APPL_NAME,          "Invalid Appl Name." },
    { ARS_DDE_RC_INVALID_INTEGER_FIELD,      "Invalid integer field." },
    { ARS_DDE_RC_INVALID_DECIMAL_FIELD,      "Invalid decimal field." },
    { ARS_DDE_RC_INVALID_DATE_FIELD,         "Invalid date field." },
    { ARS_DDE_RC_INVALID_APPLGRP_FIELD_TYPE, "Invalid Appl Group field type." } };

#define NUM_ERRORS  ( sizeof(Errors) / sizeof(ERROR_MAP) )

#define ADV_MAP struct _AdvMap
ADV_MAP
{
  char * pAdvData;
  char * pMsg;
};

static ADV_MAP Advises[] =
  { { ARS_DDE_EVENT_CRITERIA_BUTTON_1, "DDE Criteria 1 Button Clicked." },
    { ARS_DDE_EVENT_CRITERIA_BUTTON_2, "DDE Criteria 2 Button Clicked." },
    { ARS_DDE_EVENT_CRITERIA_BUTTON_3, "DDE Criteria 3 Button Clicked." },
    { ARS_DDE_EVENT_CRITERIA_BUTTON_4, "DDE Criteria 4 Button Clicked." },
    { ARS_DDE_EVENT_CRITERIA_BUTTON_5, "DDE Criteria 5 Button Clicked." },
    { ARS_DDE_EVENT_DOCLIST_BUTTON_1,  "DDE Doclist 1 Button Clicked." },
    { ARS_DDE_EVENT_DOCLIST_BUTTON_2,  "DDE Doclist 2 Button Clicked." },
    { ARS_DDE_EVENT_DOCLIST_BUTTON_3,  "DDE Doclist 3 Button Clicked." },
    { ARS_DDE_EVENT_DOCLIST_BUTTON_4,  "DDE Doclist 4 Button Clicked." },
    { ARS_DDE_EVENT_DOCLIST_BUTTON_5,  "DDE Doclist 5 Button Clicked." },
    { ARS_DDE_EVENT_SWITCH_FOCUS,      "Switch focus requested." },
    { ARS_DDE_EVENT_SWITCH_FOCUS_2,    "Switch focus *** 2 *** requested." },
    { ARS_DDE_EVENT_SWITCH_FOCUS_3,    "Switch focus *** 3 *** requested." },
    { ARS_DDE_EVENT_SWITCH_FOCUS_4,    "Switch focus *** 4 *** requested." },
    { ARS_DDE_EVENT_SWITCH_FOCUS_5,    "Switch focus *** 5 *** requested." } };

#define NUM_ADVISES  ( sizeof(Advises) / sizeof(ADV_MAP) )


// DDE variables and functions

static HDDEDATA hDdeData, hDdeResult;

HDDEDATA FAR PASCAL DdeCallBack ( UINT     iType,
                                  UINT     iFmt,
                                  HCONV    hConv,
                                  HSZ      hsz1,
                                  HSZ      hsz2,
                                  HDDEDATA hData,
                                  DWORD    dwData1,
                                  DWORD    dwData2 )
{
    int     j;
    char  * pData;
    DWORD   data_len;

    switch ( iType )
    {
        case XTYP_DISCONNECT:
            hDdeConv = NULL;
            break;
        case XTYP_ADVDATA:
            if ( hData == NULL )
                AfxMessageBox( "hData is NULL in XTYP_ADVDATA" );
```

```
            else
            {
                pData = (char*)DdeAccessData( hData, &data_len );
                for ( j = 0; j < NUM_ADVISES; j++ )
                    if ( strcmp( Advises[j].pAdvData, pData ) == 0 )
                        break;
                AfxMessageBox( j < NUM_ADVISES
                                    ? Advises[j].pMsg
                                    : "Logic Error - invalid ADVDATA." );
                DdeUnaccessData( hData );
            }
            break;
    }
    return NULL;
}

static BOOL DoDdeCommand( char * pCommand, char * pParms )
{
    DWORD   data_len;
    char  * pString1, * pData, * pFirstChar;
    int     j, rc;

    if ( pParms == NULL )
        pParms = "";
    pString1 = new char[ strlen( pCommand ) + strlen( pParms ) + 2 ];
    strcpy( pString1, pCommand );
    strcat( pString1, " " );
    strcat( pString1, pParms );

    hsz1 = DdeCreateStringHandle( DdeInstance, pString1, 0 );
    hDdeResult = DdeClientTransaction( NULL,
                                       0,
                                       hDdeConv,
                                       hsz1,
                                       CF_TEXT,
                                       XTYP_REQUEST,
                                       120000L,
                                       NULL );
    DdeFreeStringHandle( DdeInstance, hsz1 );

    delete pString1;

    RequestedData[0] = '¥0';
    if ( hDdeResult == NULL )
    {
        int     error;
        char  * pErr;

        error = DdeGetLastError( DdeInstance );
        switch ( error )
        {
            case DMLERR_ADVACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_ADVACKTIMEOUT";
                break;
            case DMLERR_BUSY:
                pErr = "DdeClientTransaction failed with DMLERR_BUSY";
                break;
            case DMLERR_DATAACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_DATAACKTIMEOUT";
                break;
            case DMLERR_DLL_NOT_INITIALIZED:
                pErr = "DdeClientTransaction failed with DMLERR_DLL_NOT_INITIALIZED";
                break;
            case DMLERR_DLL_USAGE:
                pErr = "DdeClientTransaction failed with DMLERR_DLL_USAGE";
                break;
            case DMLERR_EXECACKTIMEOUT:
```

```
                pErr = "DdeClientTransaction failed with DMLERR_EXECACKTIMEOUT";
                break;
            case DMLERR_INVALIDPARAMETER:
                pErr = "DdeClientTransaction failed with DMLERR_INVALIDPARAMETER";
                break;
            case DMLERR_LOW_MEMORY:
                pErr = "DdeClientTransaction failed with DMLERR_LOW_MEMORY";
                break;
            case DMLERR_MEMORY_ERROR:
                pErr = "DdeClientTransaction failed with DMLERR_MEMORY_ERROR";
                break;
            case DMLERR_NO_CONV_ESTABLISHED:
                pErr = "DdeClientTransaction failed with DMLERR_NO_CONV_ESTABLISHED";
                break;
            case DMLERR_NOTPROCESSED:
                pErr = "DdeClientTransaction failed with DMLERR_NOTPROCESSED";
                break;
            case DMLERR_POKEACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_POKEACKTIMEOUT";
                break;

            case DMLERR_POSTMSG_FAILED:
                pErr = "DdeClientTransaction failed with DMLERR_POSTMSG_FAILED";
                break;
            case DMLERR_REENTRANCY:
                pErr = "DdeClientTransaction failed with DMLERR_REENTRANCY";
                break;
            case DMLERR_SERVER_DIED:
                pErr = "DdeClientTransaction failed with DMLERR_SERVER_DIED";
                break;
            case DMLERR_SYS_ERROR:
                pErr = "DdeClientTransaction failed with DMLERR_SYS_ERROR";
                break;
            case DMLERR_UNADVACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_UNADVACKTIMEOUT";
                break;
            case DMLERR_UNFOUND_QUEUE_ID:
                pErr = "DdeClientTransaction failed with DMLERR_UNFOUND_QUEUE_ID";
                break;
        }
        AfxMessageBox( pErr );
        return FALSE;
    }
    else
    {
        pData = (char*)DdeAccessData( hDdeResult, &data_len );
        rc = atoi( pData );
        if ( rc == ARS_DDE_RC_NO_ERROR )
        {
            pFirstChar = strchr( pData, ' ' );
            strcpy( RequestedData, &pFirstChar[1] );
        }
        else
        {
            for ( j = 0; j < NUM_ERRORS; j++ )
                if ( Errors[j].code == rc )
                    break;
            AfxMessageBox( j < NUM_ERRORS
                           ? Errors[j].pMsg
                           : "Logic Error - invalid return code." );
        }
        DdeUnaccessData( hDdeResult );
        return rc == ARS_DDE_RC_NO_ERROR;
    }
}

static BOOL DoAdviseLoop( char * pName, BOOL stop )
```

```
{
    hsz1 = DdeCreateStringHandle( DdeInstance, pName, 0 );
    hDdeResult = DdeClientTransaction( NULL,
                                       0,
                                       hDdeConv,
                                       hsz1,
                                       CF_TEXT,
                                       stop ? XTYP_ADVSTOP : XTYP_ADVSTART,
                                       120000L,
                                       NULL );
    DdeFreeStringHandle( DdeInstance, hsz1 );

    if ( hDdeResult == NULL )
    {
        int     error;
        char  * pErr;

        error = DdeGetLastError( DdeInstance );
        switch ( error )
        {
            case DMLERR_ADVACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_ADVACKTIMEOUT";
                break;
            case DMLERR_BUSY:
                pErr = "DdeClientTransaction failed with DMLERR_BUSY";
                break;
            case DMLERR_DATAACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_DATAACKTIMEOUT";
                break;
            case DMLERR_DLL_NOT_INITIALIZED:
                pErr = "DdeClientTransaction failed with DMLERR_DLL_NOT_INITIALIZED";
                break;
            case DMLERR_DLL_USAGE:
                pErr = "DdeClientTransaction failed with DMLERR_DLL_USAGE";
                break;
            case DMLERR_EXECACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_EXECACKTIMEOUT";
                break;
            case DMLERR_INVALIDPARAMETER:
                pErr = "DdeClientTransaction failed with DMLERR_INVALIDPARAMETER";
                break;
            case DMLERR_LOW_MEMORY:
                pErr = "DdeClientTransaction failed with DMLERR_LOW_MEMORY";
                break;
            case DMLERR_MEMORY_ERROR:
                pErr = "DdeClientTransaction failed with DMLERR_MEMORY_ERROR";
                break;
            case DMLERR_NO_CONV_ESTABLISHED:
                pErr = "DdeClientTransaction failed with DMLERR_NO_CONV_ESTABLISHED";
                break;
            case DMLERR_NOTPROCESSED:
                pErr = "DdeClientTransaction failed with DMLERR_NOTPROCESSED";
                break;
            case DMLERR_POKEACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_POKEACKTIMEOUT";
                break;
            case DMLERR_POSTMSG_FAILED:
                pErr = "DdeClientTransaction failed with DMLERR_POSTMSG_FAILED";
                break;

           case DMLERR_REENTRANCY:
                pErr = "DdeClientTransaction failed with DMLERR_REENTRANCY";
                break;
            case DMLERR_SERVER_DIED:
                pErr = "DdeClientTransaction failed with DMLERR_SERVER_DIED";
                break;
            case DMLERR_SYS_ERROR:
```

```
                pErr = "DdeClientTransaction failed with DMLERR_SYS_ERROR";
                break;
            case DMLERR_UNADVACKTIMEOUT:
                pErr = "DdeClientTransaction failed with DMLERR_UNADVACKTIMEOUT";
                break;
            case DMLERR_UNFOUND_QUEUE_ID:
                pErr = "DdeClientTransaction failed with DMLERR_UNFOUND_QUEUE_ID";
                break;
        }
        AfxMessageBox( pErr );
        return FALSE;
    }
    else
        return TRUE;
}

/////////////////////////////////////////////////////////////////////////////
// CMainDlg dialog

CMainDlg::CMainDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CMainDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CMainDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
        // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CMainDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CMainDlg)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CMainDlg, CDialog)
        //{{AFX_MSG_MAP(CMainDlg)
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_LBN_DBLCLK(IDC_DOCLIST, OnDblclkDoclist)
        ON_BN_CLICKED(IDC_PRINT, OnPrint)
        ON_BN_CLICKED(IDC_CLOSE, OnCloseDlg)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMainDlg message handlers

BOOL CMainDlg::OnInitDialog()
{
    CListBox    * pList = (CListBox*)GetDlgItem( IDC_DOCLIST );
    CComboBox   * pPrinterList = (CComboBox*)GetDlgItem( IDC_PRINTERS );
    long          l, num_hits;
    char        * pToken;
    PROCESS_INFORMATION     pi;
    STARTUPINFO   &sui;
    char          cmd[300], Misc[100];
    BOOL          rc;
    DWORD         id;

    CDialog::OnInitDialog();

    pMainDlg    = this;
    DdeInstance = 0;
```

```
m_DocOpened = FALSE;
m_DocID     = 0;

( (CButton*)GetDlgItem( IDC_PRINT ) )->EnableWindow( FALSE );

    SetIcon(m_hIcon, FALSE);

// Start up the OnDemand client

// /I - Enable DDE Interface
// /W - Window placement (N = hidden)
// /V - Disable anticipation
// /B - Disable User Confirmation
strcpy( cmd, "g:¥¥ars32¥¥arsgui32.exe /I /W N /V /B /1 g:¥¥ars32¥¥locale¥¥enu" );

memset( &sui, 0, sizeof(STARTUPINFO) );
sui.cb = sizeof(STARTUPINFO);
rc = CreateProcess( NULL, cmd, NULL, NULL, FALSE, CREATE_NEW_CONSOLE, NULL, NULL, &sui, &pi );
if ( !rc )
{
    id = GetLastError( );
    FormatMessage( FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_IGNORE_INSERTS,
                   NULL, id, 0, cmd, sizeof(cmd), NULL );
    sprintf( Misc, "CreateProcessFailed - %s", cmd );
    AfxMessageBox( Misc );
    Misc[0] = '¥0';
}
else
{
    // Start a dde conversation with the client.

    if ( DdeInstance == 0 )
    {
        FARPROC pfnDdeCallBack;
        pfnDdeCallBack = MakeProcInstance( (FARPROC)DdeCallBack, pApp->m_hInstance );
        DdeInitialize( &DdeInstance,
                       (PFNCALLBACK)pfnDdeCallBack,
                       APPCLASS_STANDARD | APPCMD_CLIENTONLY,
                       0L );
    }
    hsz1 = DdeCreateStringHandle( DdeInstance, ARS_DDE_SERVICE, 0 );
    hsz2 = DdeCreateStringHandle( DdeInstance, ARS_DDE_TOPIC, 0 );
    for ( int j = 0; j < 1000; j++ )
    {
        hDdeConv = DdeConnect( DdeInstance, hsz1, hsz2, NULL );
        if ( hDdeConv != NULL )
            break;
    }
    DdeFreeStringHandle( DdeInstance, hsz1 );
    DdeFreeStringHandle( DdeInstance, hsz2 );
    if ( hDdeConv == NULL )
        AfxMessageBox( "Unable to connect to ARSGUI32." );

    else
    {
        int k;

        // Begin sending dde commands to the client.

        Misc[0] = '/';
        Misc[1] = ARS_DDE_SWITCH_HANDLE;
        sprintf( &Misc[2], "%ld", (long)(char far *)pApp->m_pMainWnd->m_hWnd );
        strcat( Misc, " " );
        strcat( Misc, " /" );
        k = strlen( Misc );
        Misc[k++] = ARS_DDE_SWITCH_CLIENT_NAME;
        strcpy( &Misc[k], "DDE Partner 1" );
```

```
            DoDdeCommand( ARS_DDE_CMD_ENABLE_SWITCH_FOCUS, Misc );

            DoDdeCommand( ARS_DDE_CMD_LOGON, "/S gunnar /U demo /P" );

            DoDdeCommand( ARS_DDE_CMD_OPEN_FOLDER, "/F Credit Card Statementss" );

            DoDdeCommand( ARS_DDE_CMD_SEARCH_FOLDER, "" );

            if ( DoDdeCommand( ARS_DDE_CMD_GET_NUM_DOCS_IN_LIST, "" ) )
            {
                num_hits = atol( RequestedData );
                for ( l = 0; l < num_hits; l++ )
                {
                    Misc[0] = '/';
                    Misc[1] = ARS_DDE_DOC_NUMBER;
                    sprintf( &Misc[2], "%ld", l );

                    if ( DoDdeCommand( ARS_DDE_CMD_GET_DOC_VALUES, Misc ) )
                    {
                        for ( pToken = strtok( RequestedData, ARS_DDE_DATA_SEPARATOR ),
                                 Misc[0] = '¥0';
                              pToken != NULL;
                              pToken = strtok( NULL, ARS_DDE_DATA_SEPARATOR ) )
                        {
                            strcat( Misc, pToken );
                            strcat( Misc, " - " );
                        }
                        if ( Misc[0] != '¥0' )
                        {
                            j = pList->InsertString( -1, Misc );
                            pList->SetItemData( j, (DWORD)l );
                        }
                    }
                    else
                        break;
                }
            }

            DoAdviseLoop( ARS_DDE_ADVISE_LOOP_1, FALSE );
        }
    }

    if ( DoDdeCommand( ARS_DDE_CMD_GET_PRINTERS, "/L" ) )
    {
        for ( pToken = strtok( RequestedData, ARS_DDE_DATA_SEPARATOR );
              pToken != NULL;
              pToken = strtok( NULL, ARS_DDE_DATA_SEPARATOR ) )
            pPrinterList->InsertString( -1, pToken );

        pPrinterList->SetCurSel( 0 );
    }

    return TRUE;
}

void CMainDlg::OnPaint()
{
         if (IsIconic())
         {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
```

```
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
        CDialog::OnPaint();
        }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CMainDlg::OnQueryDragIcon()
{
        return (HCURSOR) m_hIcon;
}

void CMainDlg::OnDblclkDoclist()
{
        CListBox * pDocsList;
    char       printer[100];
    char       Misc[100];

    pDocsList = (CListBox*)GetDlgItem( IDC_DOCLIST );

    if( m_DocOpened )
    {
        sprintf( Misc, "/D %d", m_DocID );
        DoDdeCommand( ARS_DDE_CMD_CLOSE_DOC, Misc );
    }

    Misc[0] = '/';
    Misc[1] = ARS_DDE_DOC_NUMBER;
    sprintf( &Misc[2], "%d", (int)pDocsList->GetCurSel() );
    if ( DoDdeCommand( ARS_DDE_CMD_OPEN_DOC, Misc ) )
    {
       m_DocID = atol( RequestedData );
       m_DocOpened = TRUE;

       DoDdeCommand( ARS_DDE_CMD_SHOW_WINDOW, "/W" );
    }
    else
    {
        m_DocID = 0;
        m_DocOpened = TRUE;
    }

    GetDlgItem( IDC_PRINTERS )->GetWindowText( printer, sizeof(printer) );
    if( printer != NULL && printer[0] != '¥0' )
        ( (CButton*)GetDlgItem( IDC_PRINT ) )->EnableWindow( TRUE );
}

void CMainDlg::OnPrint()
{
    char printer[100];
    char Misc[100];

    GetDlgItem( IDC_PRINTERS )->GetWindowText( printer, sizeof(printer) );

    Misc[0] = '¥0';
    sprintf( Misc, "/L %s", printer );
    DoDdeCommand( ARS_DDE_CMD_PRINT_DOC, Misc );
}
```

```
void CMainDlg::OnCloseDlg()
{
    char Misc[100];

    if( m_DocOpened )
    {
        sprintf( Misc, "/D %d", m_DocID );
        DoDdeCommand( ARS_DDE_CMD_CLOSE_DOC, Misc );
    }

    DoDdeCommand( ARS_DDE_CMD_CLOSE_FOLDER, "" );

    DoDdeCommand( ARS_DDE_CMD_LOGOFF, "" );

    DoDdeCommand( ARS_DDE_CMD_EXIT, "" );

    EndDialog(0);
}
```

付録 C. Microsoft Visual Basic 5.0 OLE サンプル・プログラム

このサンプル・プログラムは、現状のままで提供します。OnDemand 製品のライセンス所有者は、適宜、このサンプル・プログラムのコピー、改訂、変更、および関連作業を自由に行うことができます。

このプログラムは、Microsoft Visual Basic 5.0 を使用して作成され、コンパイルされています。このプログラムによって、これらの OnDemand OLE 制御メソッドを例証します。

CloseDoc
CloseFolder
GetDocDisplayValue
GetNumDocsInList
Logoff
Logon
OpenDoc
OpenFolder
ScrollDocHorz
ScrollDocVert
SearchFolder
SetDocZoom
SetFolderSearchFieldData
SetUserMessageMode

サンプル・プログラムによって使用されるグローバル変数

```
Option Explicit
Global Const defini = "vbarsole.ini"  'Default ini file name
Global Const defstanza = "VBARSOLE"   'Default stanza

' The following constants were obtained from arsoleex.h
Global Const ARS_OLE_USER_MSG_MODE_SHOW = 1
Global Const ARS_OLE_USER_MSG_MODE_SUPPRESS = 2

Global Const ARS_OLE_FIND_FIRST = 1
Global Const ARS_OLE_FIND_PREV = 2
Global Const ARS_OLE_FIND_NEXT = 3

Global Const ARS_OLE_OPR_EQUAL = 1
Global Const ARS_OLE_OPR_NOT_EQUAL = 2
Global Const ARS_OLE_OPR_LESS_THAN = 3
Global Const ARS_OLE_OPR_LESS_THAN_OR_EQUAL = 4
Global Const ARS_OLE_OPR_GREATER_THAN = 5
Global Const ARS_OLE_OPR_GREATER_THAN_OR_EQUAL = 6
Global Const ARS_OLE_OPR_BETWEEN = 7
Global Const ARS_OLE_OPR_NOT_BETWEEN = 8
Global Const ARS_OLE_OPR_IN = 9
Global Const ARS_OLE_OPR_NOT_IN = 10
Global Const ARS_OLE_OPR_LIKE = 11
Global Const ARS_OLE_OPR_NOT_LIKE = 12
Global Const ARS_OLE_RC_SUCCESS = 0
```

```
Global Const ARS_OLE_RC_NO_MEMORY = 1
Global Const ARS_OLE_RC_SERVER_ERROR = 2
Global Const ARS_OLE_RC_USER_CANCELLED = 3
Global Const ARS_OLE_RC_INVALID_DIRECTORY = 4
Global Const ARS_OLE_RC_UNAUTHORIZED_OPERATION = 5
Global Const ARS_OLE_RC_NOT_SUPPORTED = 6
Global Const ARS_OLE_RC_FILE_ERROR = 7
Global Const ARS_OLE_RC_ALREADY_LOGGED_ON = 8
Global Const ARS_OLE_RC_NOT_LOGGED_ON = 9
Global Const ARS_OLE_RC_FOLDER_ALREADY_OPEN = 10
Global Const ARS_OLE_RC_FOLDER_NOT_OPEN = 11
Global Const ARS_OLE_RC_UNKNOWN_FOLDER = 12
Global Const ARS_OLE_RC_NO_FOLDERS_AVAILABLE = 13
Global Const ARS_OLE_RC_DOC_NOT_OPEN = 14
Global Const ARS_OLE_RC_DOC_ALREADY_OPEN = 15
Global Const ARS_OLE_RC_NO_DOC_AVAILABLE = 16
Global Const ARS_OLE_RC_OPEN_DOC_FAILED = 17
Global Const ARS_OLE_RC_DOC_CANNOT_HORZ_SCROLL = 18
Global Const ARS_OLE_RC_INVALID_DOC_INDEX = 19
Global Const ARS_OLE_RC_INVALID_CONTROL_ID = 20
Global Const ARS_OLE_RC_INVALID_FIELD = 21
Global Const ARS_OLE_RC_INVALID_OPERATOR = 22
Global Const ARS_OLE_RC_INVALID_MESSAGE_MODE = 23
Global Const ARS_OLE_RC_INVALID_ZOOM_PERCENT = 24
Global Const ARS_OLE_RC_INVALID_PAGE_NUMBER = 25
Global Const ARS_OLE_RC_INVALID_ROTATION = 26
Global Const ARS_OLE_RC_INVALID_COLOR = 27
Global Const ARS_OLE_RC_INVALID_COPIES = 28
Global Const ARS_OLE_RC_INVALID_ORIENTATION = 29
Global Const ARS_OLE_RC_INVALID_PRINTER = 30
Global Const ARS_OLE_RC_INVALID_FIND_TYPE = 31
Global Const ARS_OLE_RC_ERROR_DURING_PRINT = 32
Global Const ARS_OLE_SCROLL_LINEUP = 0
Global Const ARS_OLE_SCROLL_LINELEFT = 0
Global Const ARS_OLE_SCROLL_LINEDOWN = 1
Global Const ARS_OLE_SCROLL_LINERIGHT = 1
Global Const ARS_OLE_SCROLL_PAGEUP = 2
Global Const ARS_OLE_SCROLL_PAGELEFT = 2
Global Const ARS_OLE_SCROLL_PAGEDOWN = 3
Global Const ARS_OLE_SCROLL_PAGERIGHT = 3
Global Const ARS_OLE_SCROLL_THUMBPOSITION = 4
Global Const ARS_OLE_SCROLL_THUMBTRACK = 5
Global Const ARS_OLE_SCROLL_TOP = 6
Global Const ARS_OLE_SCROLL_LEFT = 6
Global Const ARS_OLE_SCROLL_BOTTOM = 7
Global Const ARS_OLE_SCROLL_RIGHT = 7
Global Const ARS_OLE_SCROLL_ENDSCROLL = 8

Global Const DocZoom = 110

Global server   As String               'Server name
Global userid   As String               'userid
Global password As String               'password
Global folder   As String               'folder

Global doc_id As Integer
Global doc_values(0 To 8) As String

Global OpenDoc As Boolean
Global VertScrollOld As Integer
Global HorzScrollOld As Integer

'Define the Windows APIs used by the program
Declare Function GetPrivateProfileInt Lib "kernel32" Alias "GetPrivateProfileIntA"
 (ByVal  lpApplicationName As String, ByVal lpKeyName As String, ByVal nDefault As Long,
 ByVal lpFileName As String) As Long
Declare Function GetPrivateProfileString Lib "kernel32" Alias "GetPrivateProfileStringA"
```

```
 (ByVal lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpDefault As String,
 ByVal lpReturnedString As String, ByVal nSize As Long, ByVal lpFileName As String) As Long
Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)
'Declare Function GetPrivateProfileString Lib "kernel32" (ByVal sname$, ByVal Kname$, ByVal Def$,
 ByVal ret$, ByVal Size%, ByVal Fname$) As Integer

Public Sub Main()
    Dim rc As Integer

    Load frmMain
    Load frmInit

    doc_id = 0
    OpenDoc = False
    VertScrollOld = 0
    HorzScrollOld = 0

    'Disable "View" buttons
    frmMain.cmdViewStmt1.Enabled = False
    frmMain.cmdViewStmt2.Enabled = False
    frmMain.cmdViewStmt3.Enabled = False

    'Because we need the ocx file and the arssck32.dll
    'which reside in the ars directory I will require
    'that this exe and its ini file also reside in the
    'ars install directory.

    'I should check for ini file existance first.

    'Try to find the "Server" name in the ini file
    server = fncParmGet(defstanza, "Server", defini)
    If Len(server) = 0 Then
        MsgBox "Cannot find Server in " + defini
        End
    End If

'Try to find the "Userid" name in the ini file
    userid = fncParmGet(defstanza, "Userid", defini)
    If Len(userid) = 0 Then
        MsgBox "Cannot find Userid in " + defini
        End
    End If

    'Try to find the "Server" name in the ini file
    password = fncParmGet(defstanza, "Password", defini)
    If Len(password) = 0 Then
        password = " "
    End If

    'Try to find the "Folder" name in the ini file
    folder = fncParmGet(defstanza, "Folder", defini)
    If Len(folder) = 0 Then
        MsgBox "Cannot find Folder in " + defini
        End
    End If

    'The following call is for debug.
    rc = frmMain.ArsOle.SetUserMessageMode(ARS_OLE_USER_MSG_MODE_SHOW)

    frmInit.Show
    frmInit.pnlStatus.Caption = "Logging on to Server..."

    'Attempt to logon to the specified server.
    rc = frmMain.ArsOle.Logon(server, userid, password)
    If rc <> ARS_OLE_RC_SUCCESS Then
        frmInit.pnlStatus.Caption = ""
        MsgBox "Cannot Logon to server " + server + "; rc = " + Str(rc)
```

```
        End
    End If

    frmInit.SetFocus

    'Attempt to open the folder specified in the ini file.
    frmInit.pnlStatus.Caption = "Opening folder..."
    rc = frmMain.ArsOle.OpenFolder(folder)
    If rc <> ARS_OLE_RC_SUCCESS Then
        frmMain.pnlStatus.Caption = ""
        MsgBox "Cannot open folder " + folder + "; rc = " + Str(rc)
        frmMain.ArsOle.Logoff
        End
    End If

    frmInit.SetFocus
    frmInit.pnlStatus.Caption = ""
    frmInit.Hide

    frmMain.Show
End Sub

'This function returns information from the ini file.
Function fncParmGet(ByVal stanza As String, ByVal keyname As String, ByVal inifile As String)
    Dim Default, result As String
    Dim rc As Integer

    Default = ""
    result = Space$(255)

    rc = GetPrivateProfileString(stanza, keyname, Default, result, Len(result), inifile)
    If rc Then
        fncParmGet = Trim$(result)
        If Len(fncParmGet) > 1 Then
            fncParmGet = Left$(fncParmGet, Len(fncParmGet) - 1)
        End If
    Else
        fncParmGet = ""
    End If

End Function

'This function is only used to dummy up the date paid
'field of the form because for the
'demo, which uses the 'Baxter Bay Credit' folder,
'we cannot get this information from the database.
'This function adds approximately 20 days to the statement
'date field (which is passed in).
Public Function fncParseDate(ByVal stmtdate As String)
    Dim date_array(1 To 3) As String
    Dim searchch, workline, workchar As String
    Dim txtptr, lenstring, i As Integer
    Dim pay_day, pay_month, pay_year As Integer

    txtptr = 0
    searchch = Chr(47)
    workline = ""
    lenstring = Len(stmtdate)

    'Extract chars to the first '/'
    For i = 1 To lenstring
        workchar = Mid$(stmtdate, i, 1)
        'When a '/' is found, store result, reset
        If workchar = searchch Then
            txtptr = txtptr + 1
            date_array(txtptr) = workline
            workline = ""
```

```
        'Otherwise, keep building the work string
        Else
            workline = workline + workchar
        End If
    Next

    If Len(workline) > 0 Then
        txtptr = txtptr + 1
        date_array(txtptr) = workline
    End If

    'date_array contains three elements, the first is the month
    'number, the second is the day of the month, and the third is
    'the year. Simply check if the day of the month plus 20
    'is greater than 28, if so the difference becomes the new
    'day of the month, and we increment the month number.
    pay_day = Int(date_array(2)) + 20
    pay_month = Int(date_array(1))
    pay_year = Int(date_array(3))
    If pay_day > 28 Then
        pay_day = pay_day - 28
        pay_month = pay_month + 1
        If pay_month > 12 Then
            pay_month = 1
            pay_year = pay_year + 1
        End If
    End If

    fncParseDate = LTrim(Str(pay_month)) + "/" + LTrim(Str(pay_day)) + "/" + LTrim(Str(pay_year))
End Function

Private Sub cmdCustInfo_Click()
    Dim rc As Integer
    Dim acct_num, ini_str As String
    Dim first_num, second_num, third_num As Integer
    Dim temp As String
    Dim numdocs As Variant

    If OpenDoc Then
       pnlStatus.Caption = "Closing document..."
       rc = ArsOle.CloseDoc()
       pnlStatus.Caption = ""
    End If

    'Clear the payment record fields
    pnlStmtDate1.Caption = ""
    pnlStmtDate2.Caption = ""
    pnlStmtDate3.Caption = ""
    pnlBalance1.Caption = ""
    pnlBalance2.Caption = ""
    pnlBalance3.Caption = ""
    pnlDatePaid1.Caption = ""
    pnlDatePaid2.Caption = ""
    pnlDatePaid3.Caption = ""

    'Clear the customer information fields
    pnlNameData = ""
    pnlSOSData = ""
    pnlDOBData = ""
    pnlMNameData = ""
    pnlAddrData1 = ""
    pnlAddrData2 = ""
    pnlPhoneData = ""

    'Disable "View" buttons
    cmdViewStmt1.Enabled = False
    cmdViewStmt2.Enabled = False
```

```
cmdViewStmt3.Enabled = False

'Look up the account number, contained in the pnlAcctnumData text field
'in the arsvblan.ini file. If found, read the respective
'fields. If not found display error message.
acct_num = txtAcctnumData.Text

'Do at least a little validation.
If Len(acct_num) <> 11 Then
    MsgBox "Correct format for account # is 000-000-000"
    Exit Sub
End If

'If we have gotten to here we know that we have an account
'number of the format 000-000-000. If either of the first
'two sections of the number are non-zero or if the third
'section is not between 001-046 then default to the account
'number 000-000-001.
first_num = Int(Mid(acct_num, 1, 3))
second_num = Int(Mid(acct_num, 5, 3))
third_num = Int(Mid(acct_num, 9, 3))
If first_num <> 0 Or second_num <> 0 Or third_num > 46 Then
    acct_num = "000-000-001"
ElseIf third_num = 0 Then
    MsgBox "Invalid account number!"
    Exit Sub
End If

ini_str = fncParmGet(acct_num, "Name", defini)
If Len(ini_str) = 0 Then
    MsgBox "'Name' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlNameData.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "SSN", defini)
If Len(ini_str) = 0 Then
    MsgBox "'SSN' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlSSNData.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "DOB", defini)
If Len(ini_str) = 0 Then
    MsgBox "'DOB' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlDOBData.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "MaidenName", defini)
If Len(ini_str) = 0 Then
    MsgBox "'MaidenName' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlMNameData.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "StreetAddress", defini)
If Len(ini_str) = 0 Then
    MsgBox "'StreetAddress' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlAddrData1.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "CityStateZip", defini)
If Len(ini_str) = 0 Then
    MsgBox "'CityStateZip' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
```

```
End If
pnlAddrData2.Caption = "  " + ini_str

ini_str = fncParmGet(acct_num, "PhoneNum", defini)
If Len(ini_str) = 0 Then
    MsgBox "'PhoneNum' field not found for acct#" + acct_num + "in " + ininame
    Exit Sub
End If
pnlPhoneData.Caption = "  " + ini_str

'We are changing customer accounts so before we get new customer
'information, close old customers open documents.
If doc_id <> 0 Then
    rc = ArsOle.CloseDoc
    If rc <> ARS_OLE_RC_SUCCESS Then
        pnlStatus.Caption = ""
        MsgBox "Cannot set folder search criteria; rc = " + Str(rc)
        ArsOle.CloseFolder
        ArsOle.Logoff
        End
    End If
End If

pnlStatus.Caption = "Searching folder..."
rc = ArsOle.SetFolderSearchFieldData("Account", ARS_OLE_OPR_EQUAL, acct_num, "")

If rc <> ARS_OLE_RC_SUCCESS Then
    pnlStatus.Caption = ""
    MsgBox "Cannot set folder search criteria; rc = " + Str(rc)
    ArsOle.CloseFolder
    ArsOle.Logoff
    End
End If

rc = ArsOle.SearchFolder(0)
If rc <> ARS_OLE_RC_SUCCESS Then
    pnlStatus.Caption = ""
    MsgBox "Search folder failed; rc = " + Str(rc)
    ArsOle.CloseFolder
    ArsOle.Logoff
    End
End If

rc = ArsOle.GetNumDocsInList(numdocs)

rc = ArsOle.GetDocDisplayValue(numdocs - 1, 0, temp)
pnlStmtDate1.Caption = temp
pnlDatePaid1.Caption = fncParseDate(temp)
rc = ArsOle.GetDocDisplayValue(numdocs - 2, 0, temp)
pnlStmtDate2.Caption = temp
pnlDatePaid2.Caption = fncParseDate(temp)
rc = ArsOle.GetDocDisplayValue(numdocs - 3, 0, temp)
pnlStmtDate3.Caption = temp
pnlDatePaid3.Caption = fncParseDate(temp)

rc = ArsOle.GetDocDisplayValue(numdocs - 1, 3, temp)
pnlBalance1.Caption = temp
rc = ArsOle.GetDocDisplayValue(numdocs - 2, 3, temp)
pnlBalance2.Caption = temp
rc = ArsOle.GetDocDisplayValue(numdocs - 3, 3, temp)
pnlBalance3.Caption = temp

'Enable "View" buttons
cmdViewStmt1.Enabled = True
cmdViewStmt2.Enabled = True
cmdViewStmt3.Enabled = True
```

```
    pnlStatus.Caption = ""
End Sub

Private Sub cmdExit_Click()
    'If OpenDoc Then
    '    ArsOle.CloseDoc
    'End If
    'ArsOle.CloseFolder
    'ArsOle.Logoff
    End
End Sub

Private Sub cmdViewStmt1_Click()
    Dim numdocs As Variant

    rc = ArsOle.GetNumDocsInList(numdocs)

    If OpenDoc Then
       pnlStatus.Caption = "Closing document..."
       rc = ArsOle.CloseDoc()
       pnlStatus.Caption = ""
       vscrollDoc.Value = 0
       hscrollDoc.Value = 0
    End If

    pnlStatus.Caption = "Retrieving document..."
    rc = ArsOle.OpenDoc(numdocs - 1, "", 0)
    If rc <> ARS_OLE_RC_SUCCESS Then
        pnlStatus.Caption = ""
        MsgBox "Open document failed; rc = " + Str(rc)
        ArsOle.CloseFolder
        ArsOle.Logoff
        End
    End If
    pnlStatus.Caption = ""

    OpenDoc = True

    rc = ArsOle.SetDocZoom(DocZoom, horzPos, vertPos)
    vscrollDoc.Value = vertPos
    hscrollDoc.Value = horzPos
End Sub

Private Sub cmdViewStmt2_Click()
    Dim numdocs As Variant

    rc = ArsOle.GetNumDocsInList(numdocs)

    If OpenDoc Then
       pnlStatus.Caption = "Closing document..."
       rc = ArsOle.CloseDoc()
       pnlStatus.Caption = ""
       vscrollDoc.Value = 0
       hscrollDoc.Value = 0
    End If

    pnlStatus.Caption = "Retrieving document..."
    rc = ArsOle.OpenDoc(numdocs - 2, "", 0)
    If rc <> ARS_OLE_RC_SUCCESS Then
        pnlStatus.Caption = ""
        MsgBox "Open document failed; rc = " + Str(rc)
        ArsOle.CloseFolder
        ArsOle.Logoff
        End
    End If
    pnlStatus.Caption = ""
```

```
    OpenDoc = True

    rc = ArsOle.SetDocZoom(DocZoom, horzPos, vertPos)
End Sub

Private Sub cmdViewStmt3_Click()
    Dim numdocs As Variant

    rc = ArsOle.GetNumDocsInList(numdocs)

    If OpenDoc Then
       pnlStatus.Caption = "Closing document..."
       rc = ArsOle.CloseDoc()
       pnlStatus.Caption = ""
       vscrollDoc.Value = 0
       hscrollDoc.Value = 0
    End If

    pnlStatus.Caption = "Retrieving document..."
    rc = ArsOle.OpenDoc(numdocs - 3, "", 0)
    If rc <> ARS_OLE_RC_SUCCESS Then
        pnlStatus.Caption = ""
        MsgBox "Open document failed; rc = " + Str(rc)
        ArsOle.CloseFolder
        ArsOle.Logoff
        End
    End If
    pnlStatus.Caption = ""

    OpenDoc = True

    rc = ArsOle.SetDocZoom(DocZoom, horzPos, vertPos)
End Sub

Private Sub Form_Unload(Cancel As Integer)
    If OpenDoc Then
        ArsOle.CloseDoc
    End If
    ArsOle.CloseFolder
    ArsOle.Logoff
    End
End Sub

Private Sub hscrollDoc_Change()
    Dim Diff As Integer
    Dim rc As Integer
    Dim ScrollCode As Integer
    Dim NewPos As Variant

    NewPos = 0
    Diff = hscrollDoc.Value - HorzScrollOld
    If Diff = hscrollDoc.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGERIGHT
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        hscrollDoc.Value = NewPos
    ElseIf Diff = -hscrollDoc.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGELEFT
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        hscrollDoc.Value = NewPos
    ElseIf Diff = hscrollDoc.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINERIGHT
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        hscrollDoc.Value = NewPos
    ElseIf Diff = -hscrollDoc.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINELEFT
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        hscrollDoc.Value = NewPos
```

```
    Else
        ScrollCode = ARS_OLE_SCROLL_THUMBPOSITION
        NewPos = hscrollDoc.Value
        rc = ArsOle.ScrollDocHorz(ScrollCode, NewPos)
        HorzScrollOld = hscrollDoc.Value
    End If

    HorzScrollOld = hscrollDoc.Value

End Sub

Private Sub vscrollDoc_Change()
    Dim Diff As Integer
    Dim rc As Integer
    Dim ScrollCode As Integer
    Dim NewPos As Variant

    NewPos = 0
    Diff = vscrollDoc.Value - VertScrollOld
    If Diff = vscrollDoc.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEDOWN
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        vscrollDoc.Value = NewPos
    ElseIf Diff = -vscrollDoc.LargeChange Then
        ScrollCode = ARS_OLE_SCROLL_PAGEUP
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        vscrollDoc.Value = NewPos
    ElseIf Diff = vscrollDoc.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEDOWN
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        vscrollDoc.Value = NewPos
    ElseIf Diff = -vscrollDoc.SmallChange Then
        ScrollCode = ARS_OLE_SCROLL_LINEUP
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = NewPos
        vscrollDoc.Value = NewPos
    Else
        ScrollCode = ARS_OLE_SCROLL_THUMBPOSITION
        NewPos = vscrollDoc.Value
        rc = ArsOle.ScrollDocVert(ScrollCode, NewPos)
        VertScrollOld = vscrollDoc.Value
    End If
End Sub
```

付録 D. Microsoft Visual C++ 5.0 OLE サンプル・プログラム

このサンプル・プログラムは、現状のままで提供します。OnDemand 製品のライセンス所有者は、適宜、このサンプル・プログラムのコピー、改訂、変更、および関連作業を自由に行うことができます。

このプログラムは、Microsoft VC++ 5.0 を使用して作成され、コンパイルされています。このプログラムによって、これらの OnDemand OLE 制御メソッドを例証します。

- CloseDoc
- CloseFolder
- GetDocBackgroundColor
- GetDocDisplayValues
- GetDocImageColor
- GetDocNumPages
- GetDocRotation
- GetDocZoom
- GetNumDocsInList
- GetNumFolderDisplayFields
- IsDocHorzScrollRequired
- Logoff
- Logon
- OpenDoc
- OpenFolder
- PrintDoc
- SearchFolder
- ScrollDocHorz
- ScrollDocVert
- SetDocBackgroundColor
- SetDocImageColor
- SetDocRotation
- SetDocZoom
- SetUserMessageMode

```
#include "stdafx.h"
#include [winspool.h]

#include "vcole32.h"
#include "MainDlg.h"
#include "AttrsDlg.h"

static CMainDlg * pMainDlg;

#define COLOR_MAP struct _ColorMap
COLOR_MAP
{
  short  color;
  char * pText;
};

static COLOR_MAP Colors[;]; =
```

```
  { { ARS_OLE_COLOR_BLACK,   "Black" },
    { ARS_OLE_COLOR_WHITE,   "White" },
    { ARS_OLE_COLOR_RED,     "Red" },
    { ARS_OLE_COLOR_BLUE,    "Blue" },
    { ARS_OLE_COLOR_GREEN,   "Green" },
    { ARS_OLE_COLOR_YELLOW,  "Yellow" },
    { ARS_OLE_COLOR_GREY,    "Grey" },
    { ARS_OLE_COLOR_CYAN,    "Cyan" },
    { ARS_OLE_COLOR_MAGENTA, "Magenta" } };

#define NUM_COLORS  ( sizeof(Colors) / sizeof(COLOR_MAP) )

#define ERROR_MAP struct _ErrorMap
ERROR_MAP
{
  short  code;
  char * pMsg;
};

static ERROR_MAP Errors[;]; =
  { { ARS_OLE_RC_NO_MEMORY,              "insufficient memory" },
    { ARS_OLE_RC_UNKNOWN_FOLDER,         "unknown folder" },
    { ARS_OLE_RC_NO_FOLDERS_AVAILABLE,   "no folders availble" },
    { ARS_OLE_RC_SERVER_ERROR,           "server error" },
    { ARS_OLE_RC_FOLDER_ALREADY_OPEN,    "folder already open" },
    { ARS_OLE_RC_NOT_LOGGED_ON,          "not logged on" },
    { ARS_OLE_RC_ALREADY_LOGGED_ON,      "already logged on" },
    { ARS_OLE_RC_INVALID_DIRECTORY,      "invalid directory" },
    { ARS_OLE_RC_FOLDER_NOT_OPEN,        "folder not open" },
    { ARS_OLE_RC_DOC_ALREADY_OPEN,       "document already open" },
    { ARS_OLE_RC_DOC_NOT_OPEN,           "no document is open" },
    { ARS_OLE_RC_OPEN_DOC_FAILED,        "open doc failed" },
    { ARS_OLE_RC_UNAUTHORIZED_OPERATION, "unauthorized operation" },
    { ARS_OLE_RC_USER_CANCELLED,         "user cancelled operation" },
    { ARS_OLE_RC_INVALID_INDEX,          "invalid index" },
    { ARS_OLE_RC_INVALID_FIELD,          "invalid field" },
    { ARS_OLE_RC_INVALID_OPERATOR,       "invalid operator" },
    { ARS_OLE_RC_INVALID_MESSAGE_MODE,   "invalid message mode" },
    { ARS_OLE_RC_INVALID_ZOOM_PERCENT,   "invalid zoom percent" },
    { ARS_OLE_RC_DOC_CANNOT_HORZ_SCROLL, "cannot horz scroll" },
    { ARS_OLE_RC_INVALID_PAGE_NUMBER,    "invalid page number" },
    { ARS_OLE_RC_INVALID_CONTROL_ID,     "invalid other control" },
    { ARS_OLE_RC_INVALID_ROTATION,       "invalid rotation" },
    { ARS_OLE_RC_NO_DOC_AVAILABLE,       "no document for hit" },
    { ARS_OLE_RC_NOT_SUPPORTED,          "not supported" },
    { ARS_OLE_RC_FILE_ERROR,             "file error" },
    { ARS_OLE_RC_INVALID_COPIES,         "invalid copies" },
    { ARS_OLE_RC_INVALID_ORIENTATION,    "invalid orientation" },
    { ARS_OLE_RC_INVALID_PRINTER,        "invalid printer" },
    { ARS_OLE_RC_INVALID_FIND_TYPE,      "invalid find type" },
    { ARS_OLE_RC_ERROR_DURING_PRINT,     "error during print" },
    { ARS_OLE_RC_INVALID_COLOR,          "invalid color" } };

#define NUM_ERRORS  ( sizeof(Errors) / sizeof(ERROR_MAP) )

BEGIN_MESSAGE_MAP(CMainDlg, CDialog)
        //{{AFX_MSG_MAP(CMainDlg)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_BN_CLICKED(IDC_PRINT, OnPrint)
        ON_LBN_DBLCLK(IDC_DOCLIST, OnDblclkDoclist)
        ON_BN_CLICKED(IDC_CLOSE, OnCloseDlg)
        ON_WM_HSCROLL()
        ON_WM_VSCROLL()
        ON_WM_CLOSE()
        ON_BN_CLICKED(IDC_ATTRIBUTES, OnSetDocAttrs)
```

```
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

BEGIN_EVENTSINK_MAP(CMainDlg, CDialog)
    //{{AFX_EVENTSINK_MAP(CMainDlg)
        ON_EVENT(CMainDlg, IDC_ARSCTRL, 4, OnFolderSearchCompletedArsctrl, VTS_NONE)
        ON_EVENT(CMainDlg, IDC_ARSCTRL, 3, OnDocOpenedArsctrl, VTS_NONE)
        ON_EVENT(CMainDlg, IDC_ARSCTRL, 1, OnDocClosedArsctrl, VTS_NONE)
        //}}AFX_EVENTSINK_MAP
END_EVENTSINK_MAP()

CMainDlg::CMainDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CMainDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CMainDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
        // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
        m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

CMainDlg::CMainDlg()
{
}

void CMainDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CMainDlg)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}

/////////////////////////////////////////////////////////////////////////////
// CMainDlg message handlers

BOOL CMainDlg::OnInitDialog()
{
    VARIANT   var;
    short     rc;
    char      Misc[1024;];
    CArsOle * pArsCtrl;
    int       index;

    pMainDlg = this;

    m_DocOpened = FALSE;

    ( (CButton*)GetDlgItem( IDC_PRINT ) )->EnableWindow( FALSE );
    ( (CButton*)GetDlgItem( IDC_ATTRIBUTES ) )->EnableWindow( FALSE );

    SetIcon(m_hIcon, FALSE);

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );

    ( (CScrollBar*)GetDlgItem( IDC_HORZ_SCROLLBAR ) )->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
    ( (CScrollBar*)GetDlgItem( IDC_VERT_SCROLLBAR ) )->SetScrollRange( 0, ARS_OLE_SCROLL_RANGE );
    ( (CScrollBar*)GetDlgItem( IDC_HORZ_SCROLLBAR ) )->ShowScrollBar( FALSE );
    ( (CScrollBar*)GetDlgItem( IDC_VERT_SCROLLBAR ) )->ShowScrollBar( FALSE );

    // Begin calling functions in the OnDemand OLE control

    rc = pArsCtrl->SetUserMessageMode( ARS_OLE_USER_MSG_MODE_SHOW );
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "SetUserMessageMode" );

    rc = pArsCtrl->Logon( "gunnar", "demo", " " );
```

```
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "Logon" );

    rc = pArsCtrl->OpenFolder( "Credit Card Statements" );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "OpenFolder" );
        return FALSE;
    }

    rc = pArsCtrl->GetNumFolderDisplayFields( &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "GetNumFolderDisplayFields" );
        return FALSE;
    }
    m_NumDisplayFields = var.iVal;

    rc = pArsCtrl->SearchFolder( FALSE );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "SearchFolder" );
        return FALSE;
    }

    // Get the list of local printers
    CComboBox * pPrintersList;
    PRINTER_INFO_2 * pPrinterInfoArray;
    DWORD size, num_printer_infos, printer_info_index, port_index;
    char * pPortNames, * pIndividualPortName;
    pPrintersList = (CComboBox*)GetDlgItem( IDC_PRINTERS );
    EnumPrinters( PRINTER_ENUM_LOCAL |; PRINTER_ENUM_CONNECTIONS,
                  NULL, 2, NULL, 0, &size;, &num_printer_infos; );
    pPrinterInfoArray = (PRINTER_INFO_2*)new char[; size ];
    EnumPrinters( PRINTER_ENUM_LOCAL |; PRINTER_ENUM_CONNECTIONS,
                  NULL, 2, (BYTE*)pPrinterInfoArray, size, &size;, &num_printer_infos; );
    if ( num_printer_infos > 0 )
    {
        for ( printer_info_index = 0;
          printer_info_index < num_printer_infos; printer_info_index++ )
        {
            pPortNames =
              new char[; strlen( pPrinterInfoArray[printer_info_index;].pPortName; ) + 1 ];
            strcpy( pPortNames, pPrinterInfoArray[printer_info_index;].pPortName; );
            for ( pIndividualPortName = strtok( pPortNames, "," ), port_index = 0;
                  pIndividualPortName != NULL;
                  pIndividualPortName = strtok( NULL, "," ), port_index++ )
            {
                strcpy( Misc, pPrinterInfoArray[printer_info_index;].pPrinterName; );
                strcat( Misc, " on " );
                strcat( Misc, pIndividualPortName );
                index = pPrintersList->AddString( Misc );
            }
            delete pPortNames;
        }
        pPrintersList->SetCurSel( 0 );
    }
    delete [;]; pPrinterInfoArray;

    return TRUE;
}

void CMainDlg::DisplayMsg( short rc, char * pMsg )
{
    int   j;
    char  Misc[1024;];
```

```
    if ( rc == ARS_OLE_RC_SUCCESS )
        AfxMessageBox( pMsg );
    else
    {
        for ( j = 0; j < NUM_ERRORS; j++ )
            if ( Errors[j;].code; == rc )
                break;
        sprintf( Misc, "%s returned '%s'.", pMsg, j < NUM_ERRORS
                            ? Errors[j;].pMsg;                              : "***INVALID RETURN CODE***" );
        AfxMessageBox( Misc );
    }
}

void CMainDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
       CDialog::OnSysCommand(nID, lParam);
}

void CMainDlg::OnPaint()
{
       if (IsIconic())
       {
       CPaintDC dc(this); // device context for painting

       SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

       // Center icon in client rectangle
       int cxIcon = GetSystemMetrics(SM_CXICON);
       int cyIcon = GetSystemMetrics(SM_CYICON);
       CRect rect;
       GetClientRect(&rect;);
       int x = (rect.Width() - cxIcon + 1) / 2;
       int y = (rect.Height() - cyIcon + 1) / 2;

       // Draw the icon
       dc.DrawIcon(x, y, m_hIcon);
       }
       else
       {
       CDialog::OnPaint();
       }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CMainDlg::OnQueryDragIcon()
{
    return (HCURSOR) m_hIcon;
}

void CMainDlg::OnPrint()
{
    CArsOle  * pArsCtrl;
    CListBox * pDocsList;
    short      rc;
    char       printer[100;];

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );
    pDocsList = (CListBox*)GetDlgItem( IDC_DOCLIST );

    GetDlgItem( IDC_PRINTERS )->GetWindowText( printer, sizeof(printer) );

    rc = pArsCtrl->PrintDoc (-1,     // the open doc
                             0,      // entire document
                             printer,
                             1,      // local printer (not server)
                             1,      // # of copies
```

```
                            ARS_OLE_ORIENTATION_PORTRAIT,
                            .5, .5, .5, .5, // margins
                            0);     // margins in inches
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "PrintDoc" );
        return;
    }
}

void CMainDlg::OnDblclkDoclist()
{
    CArsOle  * pArsCtrl;
    CListBox * pDocsList;
    short      index;
    short      rc;
    char       printer[100;];

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );
    pDocsList = (CListBox*)GetDlgItem( IDC_DOCLIST );

    if( m_DocOpened )
        pArsCtrl->CloseDoc();

    index = pDocsList->GetCurSel();

    rc = pArsCtrl->OpenDoc( index, NULL, 0 );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "OpenDoc" );
        return;
    }

    m_DocOpened = TRUE;

    GetDlgItem( IDC_PRINTERS )->GetWindowText( printer, sizeof(printer) );
    if( printer != NULL && printer[0;]; != '¥0' )
        ( (CButton*)GetDlgItem( IDC_PRINT ) )->EnableWindow( TRUE );

    ( (CButton*)GetDlgItem( IDC_ATTRIBUTES ) )->EnableWindow( TRUE );
}

void CMainDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
    VARIANT     var;
    CArsOle   * pArsCtrl;
    short       rc;

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );

    var.vt = VT_I2;
    var.iVal = nPos;
    rc = pArsCtrl->ScrollDocHorz( (short)nSBCode, &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "ScrollDocHorz" );
    else
        pScrollBar->SetScrollPos( var.iVal );
}

void CMainDlg::OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
    VARIANT     var;
    CArsOle   * pArsCtrl;
    short       rc;

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );
```

```
    var.vt = VT_I2;
    var.iVal = nPos;
    rc = pArsCtrl->ScrollDocVert( (short)nSBCode, &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "ScrollDocVert" );
    else
        pScrollBar->SetScrollPos( var.iVal );
}

void CMainDlg::RefreshDocList( )
{
    VARIANT     var;
    CArsOle   * pArsCtrl;
    ArsOleValue * pValues;
    CListBox  * pDocsList;
    char        temp[21;];
    short       rc;
    long        num_docs = 0, j;

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );
    pDocsList = (CListBox*)GetDlgItem( IDC_DOCLIST );

    if ( pArsCtrl == NULL |;|; pDocsList == NULL )
        return;

    rc = pArsCtrl->GetNumDocsInList( &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "GetNumDocsInList" );
        return;

    }
    num_docs = var.lVal;

    pValues = new ArsOleValue[; max( m_NumDisplayFields, 1 ) ];
    pDocsList->ResetContent( );

    for ( j = 0; j < num_docs; j++ )
    {
        rc = pArsCtrl->GetDocDisplayValues( j, (IUnknown*)pValues, m_NumDisplayFields );
        if ( rc != ARS_OLE_RC_SUCCESS )
        {
            DisplayMsg( rc, "GetDocDisplayValues" );
            break;
        }
        sprintf( temp, "%s¥t%s¥t%s", pValues[0;];, pValues[3;];, pValues[2;]; );
        pDocsList->InsertString( -1, temp );
    }
    pDocsList->SetCurSel( 0 );

    delete [;]; pValues;
}

void CMainDlg::OnFolderSearchCompletedArsctrl()
{
    RefreshDocList( );
}

void CMainDlg::OnDocOpenedArsctrl()
{
    VARIANT     var;
    BOOL        required;
    CScrollBar * pHorz, * pVert;
    CArsOle    * pArsCtrl;
    short       rc;

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );
```

```
    rc = pArsCtrl->GetDocNumPages( &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "GetDocNumPages" );
        return;
    }
    m_NumPages = var.lVal;

    rc = pArsCtrl->IsDocHorzScrollRequired( &var; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        DisplayMsg( rc, "IsDocHorzScrollRequired" );
        return;
    }
    required = var.iVal;

    m_CurrentPage = 1;

    pHorz = (CScrollBar*)GetDlgItem( IDC_HORZ_SCROLLBAR );
    pVert = (CScrollBar*)GetDlgItem( IDC_VERT_SCROLLBAR );

    pHorz->ShowScrollBar( required );
    pVert->ShowScrollBar( TRUE );
    pHorz->SetScrollPos( 0 );
    pVert->SetScrollPos( 0 );
}

void CMainDlg::OnDocClosedArsctrl()
{
    CScrollBar * pBar;

    pBar = (CScrollBar*)GetDlgItem( IDC_HORZ_SCROLLBAR );
    if ( pBar != NULL )
        pBar->ShowScrollBar( FALSE );

    pBar = (CScrollBar*)GetDlgItem( IDC_VERT_SCROLLBAR );
    if ( pBar != NULL )
        pBar->ShowScrollBar( FALSE );
}

void CMainDlg::OnCloseDlg()
{
    short      rc;
    CArsOle  * pArsCtrl;

    pArsCtrl = (CArsOle*)GetDlgItem( IDC_ARSCTRL );

    if( m_DocOpened )
        pArsCtrl->CloseDoc();

    rc = pArsCtrl->CloseFolder( );
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "CloseFolder" );

    rc = pArsCtrl->Logoff( );
    if ( rc != ARS_OLE_RC_SUCCESS )
        DisplayMsg( rc, "Logoff" );

    EndDialog(0);
}

void CMainDlg::OnSetDocAttrs()
{
        CAttrsDlg dlg;
        dlg.DoModal( );
}
```

```
/////////////////////////////////////////////////////////////////////////////
// CAttrsDlg dialog


CAttrsDlg::CAttrsDlg(CWnd* pParent /*=NULL*/)
        : CDialog(CAttrsDlg::IDD, pParent)
{
        //{{AFX_DATA_INIT(CAttrsDlg)
                // NOTE: the ClassWizard will add member initialization here
        //}}AFX_DATA_INIT
}


void CAttrsDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAttrsDlg)
                // NOTE: the ClassWizard will add DDX and DDV calls here
        //}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CAttrsDlg, CDialog)
        //{{AFX_MSG_MAP(CAttrsDlg)
        ON_BN_CLICKED(IDC_BACK_COLOR, OnBackColor)
        ON_BN_CLICKED(IDC_IMAGE_COLOR, OnImageColor)
        ON_BN_CLICKED(IDC_ROTATION, OnRotation)
        ON_BN_CLICKED(IDC_ZOOM, OnZoom)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CAttrsDlg message handlers

BOOL CAttrsDlg::OnInitDialog()
{
    CArsOle  * pArsCtrl;
    CListBox * pBackList, * pImageList;
    CEdit    * pZoom, * pRotation;
    VARIANT    var1, var2, var3;
    BOOL       chg;
    short      rc, j, back_color, image_color, zoom, rotation, min, max;
    int        index;
    char       data[10;];

    pArsCtrl = (CArsOle*)pMainDlg->GetDlgItem( IDC_ARSCTRL );

    pBackList = (CListBox*)GetDlgItem( IDC_BACK_COLORS );
    pImageList = (CListBox*)GetDlgItem( IDC_IMAGE_COLORS );
    pZoom = (CEdit*)GetDlgItem( IDC_ZOOMPCT );
    pRotation = (CEdit*)GetDlgItem( IDC_ROT );

    rc = pArsCtrl->GetDocBackgroundColor( &var1;, &var2; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "GetDocBackgroundColor" );
        EndDialog( IDABORT );
        return TRUE;
    }
    back_color = var1.iVal;
    chg = var2.iVal;

    rc = pArsCtrl->GetDocImageColor( &var1;, &var2; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
```

```
        pMainDlg->DisplayMsg( rc, "GetDocImageColor" );
        EndDialog( IDABORT );
        return TRUE;
    }
    image_color = var1.iVal;
    chg = var2.iVal;

    for ( j = 0; j < NUM_COLORS; j++ )
    {
        index = pBackList->AddString( Colors[j;].pText; );
        pBackList->SetItemData( index, Colors[j;].color; );
        if ( Colors[j;].color; == back_color )
            pBackList->SetCurSel( index );
        index = pImageList->AddString( Colors[j;].pText; );
        pImageList->SetItemData( index, Colors[j;].color; );
        if ( Colors[j;].color; == image_color )
            pImageList->SetCurSel( index );
    }

    rc = pArsCtrl->GetDocZoom( &var1;, &var2;, &var3; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "GetDocZoom" );
        EndDialog( IDABORT );
        return TRUE;
    }
    zoom = var1.iVal;
    min = var2.iVal;
    max = var3.iVal;

    sprintf( data, "%d", (int)zoom );
    pZoom->SetWindowText( data );

    rc = pArsCtrl->GetDocRotation( &var1;, &var2; );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "GetDocRotation" );
        EndDialog( IDABORT );
        return TRUE;
    }
    rotation = var1.iVal;
    chg = var2.iVal;

    sprintf( data, "%d", (int)rotation );
    pRotation->SetWindowText( data );

    return TRUE;
}

void CAttrsDlg::OnBackColor()
{
    CArsOle  * pArsCtrl;
    CListBox * pList;
    VARIANT    var1, var2;
    BOOL       chg;
    short      rc, j, color;

    pArsCtrl = (CArsOle*)pMainDlg->GetDlgItem( IDC_ARSCTRL );
    pList = (CListBox*)GetDlgItem( IDC_BACK_COLORS );

    color = (short)pList->GetItemData( pList->GetCurSel( ) );

    rc = pArsCtrl->SetDocBackgroundColor( color );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "SetDocBackgroundColor" );
        rc = pArsCtrl->GetDocBackgroundColor( &var1;, &var2; );
```

```
        if ( rc != ARS_OLE_RC_SUCCESS )
            pMainDlg->DisplayMsg( rc, "GetDocBackgroundColor" );
        else
        {
            color = var1.iVal;
            chg = var2.iVal;
            for ( j = 0; j < NUM_COLORS; j++ )
            {
                if ( (short)pList->GetItemData(j) == color )
                {
                    pList->SetCurSel(j);
                    break;
                }
            }
        }
    }
}

void CAttrsDlg::OnImageColor()
{
    CArsOle  * pArsCtrl;
    CListBox * pList;
    VARIANT    var1, var2;
    BOOL       chg;
    short      rc, j, color;

    pArsCtrl = (CArsOle*)pMainDlg->GetDlgItem( IDC_ARSCTRL );
    pList = (CListBox*)GetDlgItem( IDC_IMAGE_COLORS );

    color = (short)pList->GetItemData( pList->GetCurSel( ) );

    rc = pArsCtrl->SetDocImageColor( color );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "SetDocImageColor" );
        rc = pArsCtrl->GetDocImageColor( &var1;, &var2; );
        if ( rc != ARS_OLE_RC_SUCCESS )
            pMainDlg->DisplayMsg( rc, "GetDocImageColor" );
        else
        {
            color = var1.iVal;
            chg = var2.iVal;
            for ( j = 0; j < NUM_COLORS; j++ )
            {
                if ( (short)pList->GetItemData(j) == color )
                {
                    pList->SetCurSel(j);
                    break;
                }
            }
        }
    }
}

void CAttrsDlg::OnRotation()
{
    CArsOle * pArsCtrl;
    CEdit   * pRotation;
    VARIANT   var1, var2;
    BOOL      chg;
    short     rc, value;
    char      data[10;];

    pArsCtrl = (CArsOle*)pMainDlg->GetDlgItem( IDC_ARSCTRL );
    pRotation = (CEdit*)GetDlgItem( IDC_ROT );

    pRotation->GetWindowText( data, sizeof(data) );
```

```
    rc = pArsCtrl->SetDocRotation( (short)atoi( data ) );
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "SetDocRotation" );
        rc = pArsCtrl->GetDocRotation( &var1;, &var2; );
        if ( rc != ARS_OLE_RC_SUCCESS )
            pMainDlg->DisplayMsg( rc, "GetDocRotation" );
        else
        {
            value = var1.iVal;
            chg = var2.iVal;
            sprintf( data, "%d", (int)value );
            pRotation->SetWindowText( data );
        }
    }
    else
        OnZoom();
}

void CAttrsDlg::OnZoom()
{
    CArsOle    * pArsCtrl;
    CEdit      * pZoom;
    CScrollBar * pHorz, * pVert;
    VARIANT      var1, var2, var3;
    BOOL         required;
    short        rc, value, min, max, horz_pos, vert_pos;
    char         data[10;];

    pArsCtrl = (CArsOle*)pMainDlg->GetDlgItem( IDC_ARSCTRL );
    pZoom = (CEdit*)GetDlgItem( IDC_ZOOMPCT );

    pZoom->GetWindowText( data, sizeof(data) );

    rc = pArsCtrl->SetDocZoom( (short)atoi( data ), &var1;, &var2; );
    horz_pos = var1.iVal;
    vert_pos = var2.iVal;
    if ( rc != ARS_OLE_RC_SUCCESS )
    {
        pMainDlg->DisplayMsg( rc, "SetDocZoom" );
        rc = pArsCtrl->GetDocZoom( &var1;, &var2;, &var3; );
        if ( rc != ARS_OLE_RC_SUCCESS )
            pMainDlg->DisplayMsg( rc, "GetDocZoom" );
        else
        {
            value = var1.iVal;
            min = var2.iVal;
            max = var3.iVal;
            sprintf( data, "%d", (int)value );
            pZoom->SetWindowText( data );
        }
    }

    else
    {
        rc = pArsCtrl->IsDocHorzScrollRequired( &var1; );
        if ( rc != ARS_OLE_RC_SUCCESS )
        {
            pMainDlg->DisplayMsg( rc, "IsDocHorzScrollRequired" );
            return;
        }
        required = var1.iVal;

        pHorz = (CScrollBar*)pMainDlg->GetDlgItem( IDC_HORZ_SCROLLBAR );
        pVert = (CScrollBar*)pMainDlg->GetDlgItem( IDC_VERT_SCROLLBAR );
```

```
        pHorz->ShowScrollBar( required );
        pHorz->SetScrollPos( horz_pos );
        pVert->SetScrollPos( vert_pos );
    }
}
```

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

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
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

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で提供し、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すが、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にお願いします。

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、単に目標を示しているものです。

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。

商標

以下は、IBM Corporation の商標です。

IBM
Advanced Function Presentation
Advanced Function Printing
AFP
AIX
AIX/6000
AS/400
Bar Code Object Content Architecture
BCOCA
BookManager
CICS
Cryptolope
DataJoiner
DB2
DB2 Connect
DB2 Universal Database

DFSMSdfp
DFSMSdss
DFSMShsm
DFSMS/MVS
Domino
Domino.Doc
DRDA
EDMSuite
Enterprise Storage Server
ES/3090
eServer
FlowMark
ImagePlus
Infoprint
Intelligent Printer Data Stream
IPDS
i5/OS
iSeries
Language Environment
Lotus
Lotus Notes
Mixed Object Document Content Architecture
MO:DCA
MQSeries
MVS
MVS/DFP
Notes
OpenEdition
OS/2
OS/390
Parallel Sysplex
OS/400
Presentation Manager
Print Services Facility
pSeries
RACF
RS/6000
S/390
SecureWay
SET
SP
System/370
Tivoli
Ultrastar

VideoCharger
VisualInfo
WebSphere
z/OS

Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Intel Centrino、Intel Centrino ロゴ、Celeron、Intel Xeon、Intel SpeedStep、Itanium、Pentium は Intel Corporation または子会社の米国およびその他の国における商標または登録商標です。

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。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft、Windows、Windows NT 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。

OnDemand Windows クライアント・プログラムの一部には、Pixel Translations Incorporated のライセンス・ソフトウェアが含まれています。

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]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

[マ行]

［ヤ行］

［ラ行］

A

B

C

D

E

F

G

L

M

N

O

P

R

S

T

U

W

[特殊文字]

プログラム番号: 5697-N93
5724-J33
5722-RD1

SC88-8840-04

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