iSeries 小売店シナリオ・アプリケーション

Version 5.1 for Windows

iSeries 小売店シナリオ・アプリケーション

Version 5.1 for Windows

目次

第 1 章 iSeries 小売店シナリオ・アプリケーションの概要

このシナリオ・パッケージは、IBM WebSphere Development Studio Client for iSeries (iSeries サーバー用の Eclipse ベース・テクノロジー) で開発されたサンプル・アプリケーションです。このシナリオは、以下の用途で iSeries サーバーを使用したい開発者のために設計されています。

- Java 開発
- Web 開発
- RPG コードの管理および Web アプリケーションへの変換

このアプリケーションは、ユーザーの iSeries サーバー上に復元する 2 個の .savf ファイルと、クライアント製品にインポートする 5 個の .zip ファイルから構成されています。これらは、プロセスの各ポイントで URL のある一連の Web ページを形成します。

重要: すべてのサービス・パック、PTF、および他の情報については、次のサポート・ページをチェックしてください。 **ibm.com**/software/awdtools/wdt400/support/

このシナリオでは、以下の iSeries 固有のコンポーネントに焦点を当ててその製品の各パーツを説明します。

- IBM WebFacing Tool
- Web サービス
- iSeries の Web 開発ツール (「Web 対話」ウィザードおよび「プログラム呼び出し」ウィザードを含む)
- iSeries 用 Java 開発ツール
- IBM Toolbox for Java

このシナリオでは、卸売り業者と衣料品の小売店の 2 つの会社の状態について説明します。この 2 社で取引が行われ、両社ともビジネス・ロジックおよびデータ用に自社の iSeries サーバーを使用します。これまでは、業務連絡に E メール、電話、およびファクシミリを使用して在庫をチェックし、注文を発注し、注文の履行を照会していました。現在両社は、通常の商取引を行なう時に Web を使用したいと考えています。

小売店は、次のことが実行できる Web サイトを持ちたいと考えています。

- お客様が商品の購入に使用できる
- 従業員が卸売り業者へ品目を注文するために使用できる

卸売り業者は、次のような要求を持っています。

- 小売り業者の注文を追跡のめにオンラインで受信する
- 複数の潜在的なお客様にサービスする

このシナリオでは、あなたはこの両社のプログラミング・コンサルタントの役割を果たし、その業務を Web に移行するのを手助けします。

このアプリケーションには、ユーザー・タイプに基づいて 2 つの異なるエントリー・ポイントがあります。お客様としては、その店が提供している商品 (この場合には、カジュアルな洋服) を表示することから開始します。購入したい場合は、リンクをクリックして注文画面にアクセスします。注文後に合計ページが生成され、買い物を続けるか、注文を取り消すか、あるいは注文を発注することが可能です。

管理者としては、そのアプリケーションに対するセキュア・ユーザー ID をもち、デプロイメント中にアプリケーションのセキュリティー・ポリシーを定義する必要があります。エントリー・ポイントはログイン画面であり、ここで、注文を表示し、在庫を表示して、卸売り業者から購入することができます。商品を選択し、最新の卸売り価格をチェックして、必要なサイズおよび数量を注文します。アプリケーションは、卸し売り業者に対して必要なサイズと数量があるかどうかをチェックし、さらにその注文を受けるか、あるいはこの時点ではその注文を受けることができないことを通知します。

このアプリケーションの裏側では、その製品の各種のパーツで多数のアクションが実行されています。以下のテーブルでは、このアプリケーションの各パーツが責任をもつプロセスおよび製品のコンポーネントについて説明します。各タスクの実行方法に関する詳細情報を続けます。

表 1.

カスタマー・アプリケーションのタスク	**管理者アプリケーションのタスク**	**基本タスク**
商品価格の表示		iSeries RPG プログラムを使用して Web サービスを作成し、Web 開発ツールを使用して価格を表示します。
商店からの注文の発行		サーブレットと JSP ファイルを iSeries Java 開発ツール、iSeries Web 開発ツール、および IBM Toolbox for Java とともに使用して、iSeries サーバーの在庫にアクセスして表示し、注文を発行し、購入合計を表示します。
	在庫の表示	既存の RPG プログラムを Web アプリケーションに変換し、Web 開発ツールを使用して Web ページをカスタマイズするには、IBM WebFacing Tool を使用します。
	商品 ID および数量により卸売り業者へ注文します。	**「購入」**ボタンをクリックした時に起動される Web サービスを作成します。
商店の初期 Web ページを表示します。	品目注文用の初期 Web ページを表示します。	両方のホーム・ページを作成するために iSeries の Web 開発ツールを使用します。

プロジェクト基本シナリオ

iSeries シナリオ・アプリケーションは、 SV000501、 SV000514、 SV000618、 SV001585、および SV001586 という名前をもつ 5つのプロジェクトで構成されています。本書には、事前パッケージ・アプリケーションの実行方法の説明があります。また、本書には、ステップバイステップのモジュールおよび拡張モジュールも

含まれていて、各種のプロジェクトをユーザー自身が構築するための方法も示されています。ステップバイステップ・モジュールは、アプリケーション開発と Development Studio Client に詳しくない開発者のためのものです。拡張モジュールは、アプリケーション開発を経験していて、Development Studio Client に詳しい開発者のためのものです。

注: 5 個のプロジェクトがありますが、プロジェクト SV001586 と SV000514 は Web Services モジュールに結合されるので、存在するモジュールは 4 個だけです。

このシナリオは 5 個のプロジェクトの形式になっています。

SV000501 プロジェクト: 未払いの注文、在庫、および商品の詳細を表示するための Web プロジェクトを作成します。– このプロジェクトは iSeries Web 開発ツールおよび IBM WebFacing Tool により作成され、その RPG アプリケーションを Web 上に入れるために IBM WebFacing Tool を使用したいが、Web アプリケーション開発に関する知識は豊富ではない RPG プログラマー用に設計されています。

SV001585 プロジェクト: HTML コード、サーブレット、および顧客の注文を iSeries サーバーに入れる JSP ファイルを作成します。– このプロジェクトは IBM Toolbox for Java の SQL と JDBC クラス、iSeries Java 開発ツールの RecordIOManager Bean、および「iSeries プログラム呼び出し」ウィザードを使用します。これらのエレメントは、iSeries サーバー上に存在するデータおよびプログラムにアクセスして操作する各種の方法を示します。このプロジェクトは、iSeries のデータおよびコードにアクセスする Web ページを開発したい Javaプログラマーと Web アプリケーション開発者のために設計されています。さらに、iSeries サーバー管理および RPG プログラミングの作業知識も必要となります。

SV000514 プロジェクト: 商品価格を提供するための iSeries Web サービスを作成します。– この「Web サービス」ウィザードは、「iSeries プログラム呼び出し」ウィザードで生成された Java Bean を使用して、iSeries サーバー上の 1 つまたは複数のプログラム・プロシージャーを呼び出して、その情報をブラウザーに戻します。このプロジェクトは、ワールド・ワイド・ウェブ上に記述、公開、配置、および起動が可能な内蔵タイプのモジュラー・アプリケーションを作成するために Web サービスを使用したい RPG プログラマー用に設計されています。

SV001586 プロジェクト: iSeries サーバーを介して卸売り業者へ注文を発行する Web サービスを作成します。– この Web サービスは商品 ID とともに、必要な数量を受け入れてから、卸売り業者に注文を発行します。このプロジェクトは SV000514 のコンポーネントであり、Web サービスを作成したい RPG プログラマー用に設計されています。

SV000618 プロジェクト: IBM WebFacing Tool によって生成された、注文フォーム、在庫フォーム、および購入注文のインターフェースを取るための Web プロジェクトを作成します。– このプロジェクトには iSeries Web 開発ツールが必要となり、SV000514 と SV001586 で開発された Web サービスを使用して連携するための HTML および JSP ファイルの作成も含まれます。このプロジェクトは、Web サービスで作業する、RPG と Java のプログラミング知識がある開発者用に設計されています。

第 2 章 テクノロジーの概念

このシナリオ・アプリケーションを介して作業を行うには、Web アプリケーション開発を初めて実行する場合は特に、いくつかのテクノロジーの概念を理解する必要があります。アプリケーションを介して作業する時に出てくる概念の一部の簡単なリストは次のとおりです。

エンタープライズ・アーカイブ・ファイル (EAR)

EAR ファイルは、.ear の拡張子をもつ標準 Java アーカイブ (JAR) ファイルです。これには、複数の Web プロジェクトを含めて、ユーザーはそれを使用して Web アプリケーションにパッケージして、WebSphere Administrative Server (WAS) に展開できます。**注:** J2EE SDK アプリケーション・デプロイメント・ツールの GUI バージョンでは、まず始めに EAR ファイルを作成して、JAR および Web アーカイブ (WAR) ファイルを EAR ファイルに追加します。ただし、コマンド行パッケージャー・ツールを使用する場合は、まず始めに、JAR とWAR ファイルを作成してから、EAR ファイルを作成します。

IBM WebFacing Tool

IBM WebFacing Tool は、既存の 5250 インターフェースをブラウザー・ベースのグラフィカル・ユーザー・インターフェースに変換します。オリジナルの iSeries アプリケーションへの変更が少ないか、またはないので、プログラムの使用をインターネットまたはイントラネットに拡張できます。

Java Archive ファイル (JAR)

JAR ファイルは、.zip ファイルと類似した Java ファイルの圧縮パッケージです。これには、単一ファイルに収集して、ブラウザーへの高速ダウンロードのために圧縮した Java アプレットのクラス、イメージ、およびサウンド・ファイルが含まれています。

Java Server Pages (JSP)

JSP は、動的内容を静的 HTML ページに表示する機能を提供します。 Java で作成される JSP はサーバーやプラットフォームから独立したものです。 Web の内容から Web プレゼンテーションを効率的に分離することによって、設計を迅速に変更して、その Web ページを表示する必要がある開発者は JSP を役立てることができます。

プログラム呼び出し Bean

これは、「プログラム呼び出し」ウィザードによって生成される Java Bean です。この 1 つのタイプは、Java アプリケーションで使用される通常の Java Bean です。もう 1 つのタイプは、Web サービスを作成するために「Web サービス」ウィザードで使用できます。

「プログラム呼び出し」ウィザード

「プログラム呼び出し」ウィザードは、iSeries プログラムまたはプロシージャーの起動に必要な Java Bean と関連の PCML ファイルを作成するのに役立ちます。このウィザードはプログラムまたはサービス・プログラム・オブジェクトと、そのオブジェクトのパラメーターに関する情報のプロンプトを出してから、必要な Java Bean (および PCML ファイル) を作成します。

レポート・プログラム生成プログラム (RPG)

iSeries プログラマーによって使用されるプロシージャー型プログラミング言語です。RPG を使用して、送り状処理プログラムおよび注文入力プログラムなどのビジネス・アプリケーションを作成できます。最新バージョンの ILE RPG IV では、以前のバージョンにおけるプログラマーの経験をサポートすると同時に、RPG 言語の機能も拡張されています。

サーブレット

サーバー・サイドのプログラムで、Java で書かれていて、IBM WebSphere Application Servers などの Java 使用可能サーバーまたはアプリケーション・サーバーで実行されます。サーブレットは、HTML 応答を生成して要求に応答するなどの、サーバーが指定したタスクを実行します。たとえば、オンライン銀行用アプリケーションでサーブレットを使用して、サーバーにデータを送信しながら、ユーザーに応答することができます。

Web コンポーネント

Web コンポーネントを使用して、データ入力フィールドおよびプッシュボタンなどの iSeries オブジェクトを定義できます。これは iSeries サーバー・プログラムと Web ページ間で情報を交換できます。開発者は Web コンポーネントを使用して、入力フィールドの構文検査やボタン・クリックなどのユーザー・イベントを取り込むことができます。

「Web 対話」ウィザード

このウィザードは iSeries Web 開発ツールの一部です。これは、ILE プログラムと Web ページ間の対話を作成して管理します。このウィザードは入力、入出、およびエラー・メッセージの表示位置を制御して、入出力フィールドからのデータを ILE プログラムに送信します。また、この「Web 対話」ウィザードを使用して、ユーザーがエラーの発生元を簡単に識別できるように、エラーが発生したエリアにエラー・メッセージをマップすることもできます。

Web サービス

Web サービスは、インターネットを介して使用するために設計され実装された内蔵タイプのアプリケーションです。これは、SOAP、WSDL および XML などのオープン・スタンダードで作成されます。Web サービスを使用できるビジネス形態は多数あり、クライアントがインターネットを介してその在庫レベルを検査できる在庫管理システムや、供給業者への商品注文を直接追跡したい場合などもこれに含まれます。

Web Services Definition Language (WSDL)

WSDL は Web サービスのインターフェースを定義する XML ベースの言語です。WSDL は Web サービスを理解して、Web サービスとサーバー・プログラム間の情報の流れを管理します。たとえば、開発者は WSDL を使用して、更新された株価情報を示す Web サイトのインターフェースを作成します。

WebSphere Studio Workbench

IBM WebSphere Development Studio Client for iSeries は WebSphere Studio Workbench 上にビルドされていて、Eclipse プラットフォームの IBM のインプリメンテーションです。拡張可能な汎用ワークベンチは、アプリケーションのビルドおよび保守に必要なツールをすべて統合しています。開発者は Development Studio Client を使用し、プラグインの使用により開発環境に新規のオブジェクトを取り込んで、Java ファイル、グラフィックス、ビデオなどをシームレスで追加できます。

第 3 章 シナリオの実行

*卸売り*および*小売店* アプリケーションは Development Studio Client ワークベンチで実行することも、または iSeries プラットフォームを含む任意のプラットフォームの WebSphere Application Server 上でも実行できます。アプリケーションの概説については、1 ページの『第 1 章 iSeries 小売店シナリオ・アプリケーションの概要』を参照してください。

このセクションの終わりまでには、次のことが実行できるようになるはずです。

- サンプル・ファイルおよび Development Studio Client オブジェクトをユーザーの iSeries サーバーに復元する
- Web プロジェクトおよび WebFacing プロジェクトを作成して、そのファイルを保持する
- 復元されたファイルをユーザーの iSeries サーバーから Development Studio Client にインポートする
- WebSphere Application Server (WAS) を構成する
- WebFacing プロジェクトを構成する
- WebSphere テスト環境でお客様としてアプリケーションを実行する
- WebSphere テスト環境で管理者としてアプリケーションを実行する
- ユーザーの WebSphere Application Server を構成する
- 管理者のページを保護する
- 外部 WAS 展開用の EAR ファイルを作成する
- アプリケーションを WAS に展開する
- アプリケーションを WAS で実行する

始める前に

ワークベンチからアプリケーションをテストするには、次のことを確認する必要があります。

- V5R1 以降の iSeries サーバーを使用している
- 最新の WebFacing PTF が適用されている。 PTF については、サポート・ページ (http://www.ibm.com/software/awdtools/wdt400/support/) を参照してください。
- iSeries サーバーに対して NET USE アクセスがある

サンプル・ファイルのインストール

iSeries シナリオ・アプリケーションを使用するには、次のファイルの作業が必要です。

- Wholesale.savf
- Retailstor.savf
- Qdtssfl.savf

- SV000501.zip
- SV000514.zip
- SV000618.zip
- SV001585.zip
- SV001586.zip

.savf ファイルには iSeries データおよび RPG プログラム名が含まれていて、.zip ファイルには iSeries データを操作するために iSeries プログラムと対話する Web アプリケーションが含まれています。まず始めに、.savf ファイルをユーザーの iSeries サーバーに復元する必要があります。これで、.zip ファイルを Development Studio Client にインポートし、ワークベンチでそのアプリケーションを実行することができます。

.savf ファイルの復元

本書のサンプルで扱う仕事をするには、WHOLESALE および RETAILSTOR ライブラリーを ユーザーの iSeries サーバーに復元する必要があります。この製品の前のリリースのライブラリーをすでに復元している場合でも、その内容が異なっているために、それを復元する必要があります。

注: サンプル・ライブラリーのインストールに使用される .savf ファイルは V5R1 またはそれ以降の iSeries サーバーで使用するためのものです。このシナリオの目的のためには、両方のライブラリーを同じ iSeries サーバーに復元しますが、実際のビジネス用にこのアプリケーションを開発している場合は、この 2 つのライブラリーを 2 つの異なる iSeries サーバーに復元します。WHOLESALE ライブラリーは Web サービスを提供する iSeries サーバーに復元し、RETAILSTOR ライブラリーは小売店に属する iSeries サーバーに復元します。

Wholesale.savf ファイルを復元するには:

1. パーソナル・コミュニケーションズ・エミュレーターを介してユーザーの iSeries サーバーにログオンします。
   a. ライブラリーを作成して、その保管ファイルを入れます。エミュレーターで新規ライブラリーを作成するには、CRTLIB を入力します。
   b. そのライブラリーに SCENARIO という名前を指定します。
   c. 次の行にタブし、ライブラリー・タイプとして *TEST を指定し、Enter を押してその変更を保管します。
   d. CRTSAVF コマンドを使用して 2 つの保管ファイルを作成します。次の 2 行をそれぞれ Enter を押して処理します。

      ```
      CRTSAVF FILE(SCENARIO/WHOLESALE)
      CRTSAVF FILE(SCENARIO/RETAILSTOR)
      ```

      これらの行は、ユーザーの Scenario ライブラリーに保管ファイルを作成することを指定します。
2. ワークステーションで、コマンド・プロンプト・ウィンドウをオープンします。
   a. .savf ファイルが存在するディレクトリーに移動する必要があります。デフォルトでは、cd c:¥wdsc¥wdscsampl と入力します。別のドライブに製品をインストールした場合、または Development Studio Client のホーム・ディレクトリーとして ″wdsc″ を選択しなかった場合は、製品をインストールした正しい場所にある wdscsampl ディレクトリーに移動します。

b. コマンド行で、次のように入力します。ftp *hostname* (*hostname* は iSeries サーバーの名前。たとえば PROD400)

c. iSeries サーバーのユーザー ID およびパスワードを入力します。

d. コマンド行で、cd /qsys.lib/scenario.lib と入力して、ユーザーの Scenario ライブラリーに切り替えます。

e. 次の行を入力します。

```
binput WHOLESALE.savf WHOLESALE.savf
put RETAILSTOR.savf RETAILSTOR.savf
quit
```

これらの行 (quit の前まで) は、保管ファイルをローカル・システムから取り出して、iSeries サーバーに入れることを指定します。

3. iSeries コンソールに戻り、Wholesale ライブラリーを復元します。

a. RSTLIB と入力し F4 を押して、ライブラリーの復元方法を定義します。

b. **「保管されたライブラリー」**フィールドに、WHOLESALE と入力して、Tab キーを押します。

c. **「装置」**フィールドに、*savf と入力して、Tab キーを押します。

d. 次のフィールドで Enter を押し、追加の値を表示して、**「保管ファイル」**フィールドまでタブ移動します。

e. **「保管ファイル」**フィールドに WHOLESALE と入力して、Tab キーを押します。

f. **「ライブラリー」**フィールドは、既存の値を削除して scenario と入力します。

g. Enter キーを押して、WHOLESALE ライブラリーを iSeries サーバーに復元します。

4. Retailstor.savf ファイルに対し、この手順を繰り返します。

a. RSTLIB と入力し F4 を押して、ライブラリーの復元方法を定義します。

b. **「保管されたライブラリー」**フィールドに、RETAILSTOR と入力して、Tab キーを押します。

c. **「装置」**フィールドに、*savf と入力して、Tab キーを押します。

d. 次のフィールドで Enter を押し、追加の値を表示して、**「保管ファイル」**フィールドまでタブ移動します。

e. **「保管ファイル」**フィールドに RETAILSTOR と入力し、Tab キーを押します。

f. **「ライブラリー」**フィールドは、既存の値を削除して scenario と入力します。

g. Enter キーを押してアクションを保管し、RETAILSTOR ライブラリーを iSeries サーバーに復元します。

ライブラリー・リストの変更

RETAILSTOR および WHOLESALE ライブラリーを iSeries サーバーに復元しましたが、これらのライブラリーが、サインオン時に使用するライブラリー・リストにあることを確認しなければなりません。リストにない場合、Web アプリケーションがこれらのライブラリーを検出できず、アプリケーションの実行または作成時に問

題が生じる可能性があります。また、QGPL ライブラリーがライブラリー・リストにあることも確認しなければなりません。

これを行うには、ジョブ記述とユーザー・プロファイルを更新し、ライブラリー・リストへの変更が永続的になるようにする必要があります。

それぞれのユーザーは、異なった方法でライブラリー・リストを変更しますが、その方法が分からない場合は、その 1 例は次のとおりです。

1. iSeries サーバーで、`dsplibl` と入力し、Enter キーを押してライブラリー・リストを表示します。
2. RETAILSTOR、WHOLESALE、および QGPL のライブラリーがライブラリー・リスト中にあるかどうかを検査します。リストにあれば、エミュレーター・ウィンドウを終了してシャットダウンすることができます。これがない場合:
   a. F12 を押して、現行の画面を取り消します。
   b. ジョブ記述の名前を見つけます。これを行うには、`dspusrprf` *`username`* (*username* は、iSeries サーバーへのサインオンに使用する ID) と入力して Enter を押し、ユーザー・プロファイルを表示します。ジョブ記述は、ユーザー・プロファイルのいずれかのページにあります (「Page Down」を使用)。ジョブ記述を見つけたら、それをメモして F12 キーを押し、画面を終了します。
   c. コマンド行で `chgjobd` と入力し、F4 キーを押します。
   d. **「ジョブ記述」**フィールドにジョブ記述名を入力し、F10 キーを押して追加パラメーターを確認します。
   e. **「初期ライブラリー・リスト」**にページ送りします。
   f. **「初期ライブラリー・リスト」**の下のフィールドに「値の続きは +」と表示されます。正符号 (+) を入力して、Enter キーを押します。
   g. **「値の追加指定」**ページで、存在するエントリーをすべて削除し、`RETAILSTOR`、`WHOLESALE`、および `QGPL` と入力して、値それぞれの間をタブ移動します。アプリケーションはまず RETAILSTOR ライブラリーを検出する必要があるため、ライブラリーはこの順序で入力する必要があります。Enter を押します。
   h. Enter を押して、変更を再保管します。
   i. コマンド行で `chgusrprf` と入力し、Enter キーを押します。
   j. ユーザー名を入力して Enter を押します。
   k. F10 を押して追加パラメーターを表示します。
   l. **「ジョブ記述」**にページ送りします。
   m. ジョブ記述の名前が `dspusrprf` コマンドで検出したジョブ記述に一致しない場合、このコマンドで検出したジョブ記述の名前を入力し、Enter キーを押して変更内容を保管します。

Development Studio Client でのアプリケーションのインポートおよび実行

5 つの zip ファイルが、Development Studio Client 用にインストールしたプログラム・ファイル (デフォルトでは、c:¥wdsc¥wdscsampl ディレクトリー) にあります。以下のとおりです。

- SV000514.zip
- SV000618.zip
- SV001585.zip
- SV001586.zip
- SV000501.zip

それぞれを個別に Development Studio Client ワークベンチにインポートし、対応するエンタープライズ・アーカイブ (EAR) ファイルをそれぞれ指定します。 EAR ファイルは、.ear の拡張子をもつ標準 Java アーカイブ (JAR) ファイルです。これには、複数の Web プロジェクトを含めて、ユーザーはそれを使用して Web アプリケーションにパッケージして、WebSphere Administrative Server (WAS) に展開できます。

初めの 4 個のファイルは Web プロジェクトと対応し、5 番目のファイルは WebFacing プロジェクトと対応しています。まず始めに、Web プロジェクトを作成してから、WebFacing プロジェクトを作成します。その後で、ZIP ファイルの内容を 5 個のプロジェクトにインポートします。

ローカル・システムで .zip ファイルを解凍

シナリオ .zip ファイルの内容を Development Studio Client ワークベンチにインポートするには、これらを別々のディレクトリーに解凍して、.zip ファイルとしてではなく、ファイル・システムとしてインポートできるようにする必要があります。

注: WinZip 以外のプログラムを使用している場合は、そのプログラムでのタスクを実行してください。

1. ローカル・システムで、c:¥wdsc¥wdscsampl にナビゲートします。別のドライブまたは別の基本フォルダーにインストールするよう選択した場合、c:¥wdsc ディレクトリーではない可能性があります。この場合、そのディレクトリーにナビゲートして ¥wdscsampl を見つけてください。
2. .zip ファイル SV000501.zip をダブルクリックします。
3. 「Extract」ボタンをクリックして、.zip ファイルが存在するフォルダー (デフォルトでは c:¥wdsc¥wdscsampl) にナビゲートします。
4. **「All Files」**ラジオ・ボタンを選択し、**「Extract」**をクリックします。
5. wdscsampl ディレクトリーを調べると、必要なファイルがすべて入っている SV000501 フォルダーが作成されていることが分かります。
6. これで、すべてのファイルが c:¥scenario¥SV000501 に解凍されました。ファイル・システムをインポートするときに間違って選択しないよう、SV000501.zip ファイルを削除してください。
7. このセクションのステップを他の 4 つの .zip ファイル (SV000618.zip、SV001585.zip、 SV001586.zip、および SV000514.zip.) に対して繰り返します。

Web プロジェクトの作成

このセクションでは、SV000514、SV000618、SV001585、および SV001586 の Web プロジェクトを作成します。各プロジェクトを作成するためのステップは個別に要約されています。このステップは反復の多いもののように思われますが、どの EAR ファイルがどのプロジェクトに関連するかについて、ささいではあっても重要な違いがいくつかあり、 4 個のプロジェクトを作成するときに注意する必要があります。

Web プロジェクトを作成するには:

1. 「スタート」メニューから IBM WebSphere Development Studio Client for iSeries をオープンします。
2. Development Studio Client で、**「ウィンドウ」>「パースペクティブのオープン」>「その他」>「Web」**をクリックしてメニュー・バーで Web パースペクティブをオープンし、**「OK」**をクリックします。
3. **「ファイル」>「新規」>「動的 Web プロジェクト」**をクリックします。
   a. **「プロジェクト名」**フィールドに SV000514 と入力します。
   b. **「拡張オプションの構成」**チェック・ボックスを選択して、**「次へ」**をクリックします。
   c. **「EAR プロジェクト」**フィールドの横にある**「新規」**ボタンをクリックします。
   d. **「プロジェクト名」**フィールドに SVWholeSaleEAR と入力し、両方のダイアログ・ボックスで**「終了」**をクリックします。
4. **「ファイル」>「新規」>「動的 Web プロジェクト」**をクリックします。
   a. **「プロジェクト名」**フィールドに SV001586 と入力し、**「次へ」**をクリックします (**「拡張オプションの構成」**チェック・ボックスが自動的にチェックされています)。
   b. SWholeSaleEAR が**「EAR プロジェクト」**ドロップダウン・フィールドで選択されていることを確認します。
   c. **「終了」**をクリックします。
5. **「ファイル」>「新規」>「動的 Web プロジェクト」**をクリックします。
   a. **「プロジェクト名」**フィールドに、SV000618 と入力して**「次へ」**をクリックします。
   b. **「EAR プロジェクト」**フィールドの横にある**「新規」**ボタンをクリックします。
   c. **「新規プロジェクト名」**フィールドに SVStoreEAR と入力し、両方のダイアログ・ボックスで**「終了」**をクリックします。
6. **「ファイル」>「新規」>「動的 Web プロジェクト」**をクリックします。
   a. **「プロジェクト名」**フィールドに、SV001585 と入力して**「次へ」**をクリックします。
   b. SVStoreEAR が**「EAR プロジェクト」**ドロップダウン・フィールドで選択されていることを確認します。
   c. **「終了」**をクリックします。

Web プロジェクト・ファイルのインポート

ファイル・システムをワークスペースにインポートするには、以下の手順で SV000514 のファイルをインポートします。この手順は、他の 3 つの Web プロジェクト・ファイル・システム、SV001586、SV001585、および SV000618 と同じです。これ以降のセクションでは、WebFacing プロジェクトを作成してインポートする方法について説明します。

ファイルをインポートするには:

1. 「参照」をクリックしてナビゲートします。
2. **SV000514** フォルダーを右クリックして**「インポート」**を選択します。
3. **「インポート」**ウィザードで、**「ファイル・システム」**をクリックしてから**「次へ」**をクリックします。
4. シナリオ・ファイルを解凍したディレクトリーを調べます。デフォルトでは C:¥wdsc¥wdscsampl です。
5. **SV000514** をクリックし、さらに **「OK」**をクリックします。
6. **「すべて選択」**をクリックして、SV000501 ファイル・システムのすべてのコンポーネントが選択されていることを確認します。
7. **「フォルダー」**フィールドに、デフォルトで **SV000514** と入力されていない場合は、これを入力します。
8. **「警告なしに既存のリソースを上書き」**チェック・ボックスを選択します。
9. コンテキスト・ルートの変更に関するダイアログを受け取った場合には、**「終了」** をクリックして、 **「はい」**をクリックします。
10. ステップ 2 から 9 を SV001586、SV001585、および SV000618 に対して繰り返しますが、ステップの完了時には、必ず異なるプロジェクト名を使用します。

WebFacing プロジェクト・ファイルのインポートと WebFacing プロジェクトの作成

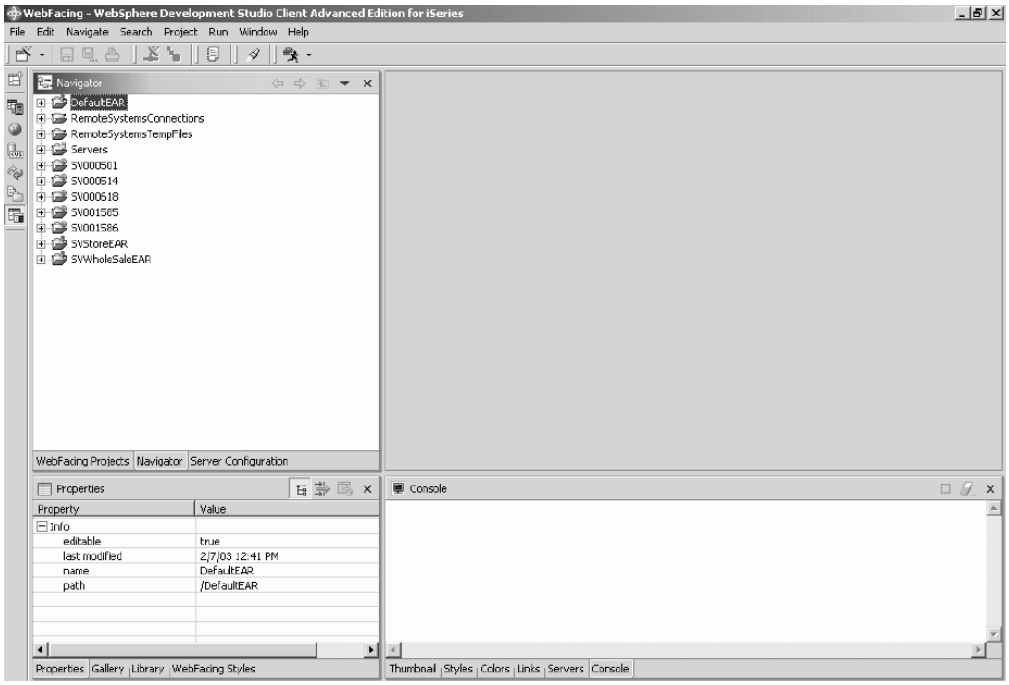
SV000501 は WebFacing プロジェクトであるため、ここで作成した 4 個の Web プロジェクトとは異なるプロジェクトです。異なるファイルを含んでいて、iSeries サーバーから DDS ファイルを処理するための固有の構造を持つために、WebFacing プロジェクトは異なります。

ファイルをインポートし、WebFacing プロジェクトを作成するには:

1. 「プロジェクト・ナビゲーター」ビューが表示されていない場合、**「プロジェクト・ナビゲーター」**タブをクリックします。
2. ビューを右クリックして**「インポート」**を選択します。
3. 「インポート」ウィンドウで、**「WebFacing プロジェクト」**を選択して**「次へ」**をクリックします。
4. シナリオ・ファイルを解凍したディレクトリーを調べます。デフォルトでは C:¥wdsc¥wdscsampl です。
5. **SV000501** を選択して**「OK」**をクリックします。**「次へ」**をクリックします。
6. ウィンドウの**「見つかった WebFacing プロジェクト」**エリアで**「SV000501」**チェック・ボックスを選択します。

7. **「エンタープライズ・アプリケーション・プロジェクト (EAR)」**フィールドに SVStoreEAR と入力します。
8. **「終了」**をクリックします。

これで、すべてのファイルがインポートされ、ワークスペースは次のように表示されます。



インポートしたプロジェクトの横にエラーが表示 (赤い x マーク) されている場合は、そのプロジェクトを右クリックして**「プロジェクトの再ビルド」**を選択します。ワークスペースの内容によりますが、これで多くのエラーがクリアされ、アプリケーションが正しく実行されるようになります。コードのいくつかが Development Studio Client リリース 5.1 に完全に移行されていないことが原因で、まだエラーが残る場合もあります。このセクションの演習を続けて製品について学習するか、または次のセクションに進み、最初からアプリケーションのビルドを試すことができます。

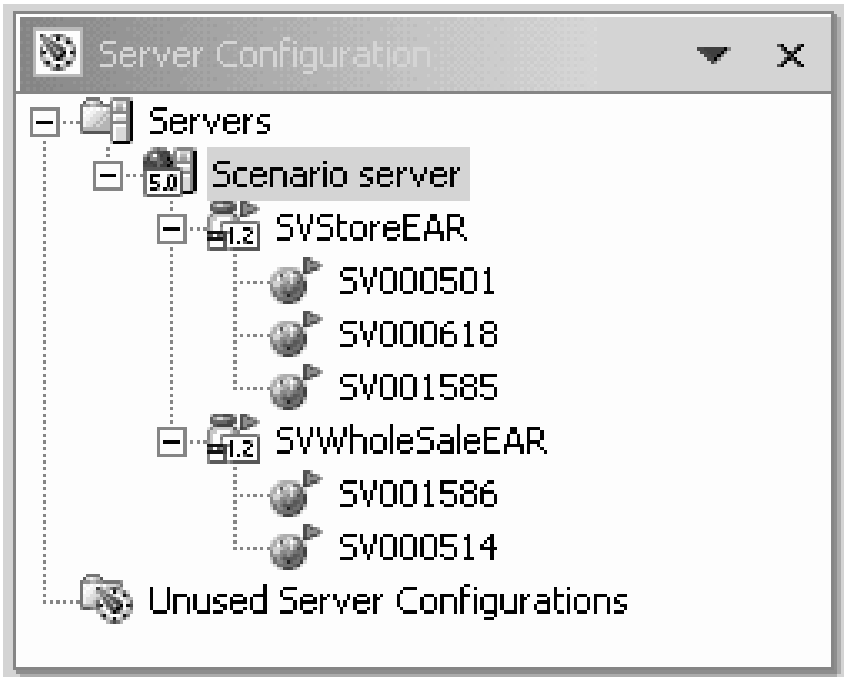
サーバー構成

これで、すべてのソース・ファイルをインポートしましたが、WebSphere テスト環境サーバーが SVWholesaleEAR および SVStoreEAR アプリケーションを認識するように、そのサーバーを構成する必要があります。

WebSphere テスト環境サーバーを構成するには:

1. サーバー・パースペクティブに切り替えます。ワークベンチのメニュー・バーから、**「ウィンドウ」>「パースペクティブのオープン」>「その他」> 「サーバー」**をクリックして、**「OK」**をクリックします。
2. 「サーバー構成」ビューでは、**「サーバー」**を右クリックして、**「新規」>「サーバーおよびサーバー構成」**を選択します。
3. **「サーバー名」**フィールドに、Scenario server と入力します。

4. **「フォルダー」**フィールドに、Scenario folder と入力します。
5. **「サーバー・タイプ」**ボックスでは、**WebSphere バージョン 5.0** を展開し、**テスト環境**がデフォルトによってまだ選択されていない場合には、それをクリックします。
6. 何らかのメッセージを受け取った場合は、**「完了」** をクリックし、さらに**「はい」**をクリックします。
7. サーバー構成に EAR ファイルを追加するには:
   a. **サーバー**を展開します。
   b. **Scenario サーバー**を右クリックして、**「追加」>「SVWholeSaleEAR」**を選択します。
   c. **サーバー**を展開します。
   d. **Scenario サーバー**をもう一度右クリックして、**「プロジェクトの追加と削除」**を選択します。
   e. **「すべてを追加」**ボタンをクリックして、SVStoreEAR および SVWholeSaleEAR の両方をウィンドウの右側に移動します。
   f. **「終了」**をクリックします。
8. 「サーバー構成」ビューでは、**Scenario サーバー**を展開し、次に **SVStoreEAR** および **SVWholeSaleEAR** を展開して、すべてのアプリケーションがリストされていることを調べます。



サーバーがプロジェクトを選出したことを確認するには:

1. 「サーバー構成」ビューの右にある「サーバー」ビュー (「サーバー」ビューを表示できない場合は**「サーバー」**タブをクリック) で **Scenario サーバー**をクリックして、**「公開」**を選択します。**「公開」**ダイアログ・ボックスが表示されて、公開の進行状況が示されます。
2. アプリケーションを WebSphere Application Server テスト環境に公開するには、「OK」をクリックします。
3. 公開が終了したら、そのダイアログ・ボックスで**「OK」**をクリックします。
4. 同じビューで、**Scenario サーバー**をもう一度クリックして、**「開始」**をクリックします。「コンソール」ビューがオープンし、そのサーバーのアクティビティーが示されます。「サーバー *servername* は e-business 用にオープンされます」のメッセージ行がビューの下部に示された時点でサーバーが開始されます。スクロールダウンして、すべてのメッセージを表示します。

WebFacing プロジェクトの構成

アプリケーションを実行するか、あるいは SV000501 モジュールを操作する前に、実行するアプリケーションの WebFacing サーバーを開始して、プロジェクトが正しい iSeries サーバーを使用するよう、そのプロジェクトを構成する必要があります。

WebFacing サーバーを開始するには:

1. パーソナル・コミュニケーションズ 5250 エミュレーターをオープンし、ユーザーのユーザー ID およびパスワードでサインオンします。
2. コマンド行で、`strtcpsvr *webfacing` と入力します。
3. 画面の下部には、メッセージ「WEBFACING サーバーを開始中」が表示されるはずです。

ここで、Development Studio Client ワークベンチ内の後方で、WebFacing プロパティーを変更する必要があります。

1. Web パースペクティブに切り替えます (画面の左側の一群のアイコンをクリックして、パースペクティブ間の切り替えができます)。
2. **「プロジェクト・ナビゲーター」**タブをクリックして、プロジェクト構成を表示できるようにします。
3. **SV000501** を展開して、**Web デプロイメント記述子**をダブルクリックします。
4. **「パラメーター」**タブをクリックします。
5. **WFDefaultHost** をクリックします。
6. **「値」**フィールド既存の値を消去し、iSeries サーバーの名前を入力します。
7. ワークベンチのメニュー・バーから、「保管」アイコンをクリックするか、あるいは**「ファイル」>「Web デプロイメント記述子の保管」**をクリックします。メッセージを受け取った場合は、**「OK」**をクリックします。ファイルをクローズします。

サーバー情報の定義

ここで、ユーザーのユーザー ID およびパスワードで iSeries サーバーを実行するために 5 個のすべてのプロジェクトが構成されるように、これらのすべてのプロジェクトの iSeries サーバー情報を定義する必要があります。サーバー情報を定義するには:

1. Web パースペクティブ内にいることを確認します。
2. 「プロジェクト・ナビゲーター」ビューで **SV000501** を右クリックし、**「iSeries Web Tools 実行時構成の指定」**を選択します。
3. **「iSeries サーバー名」**フィールド既存の値を削除し、iSeries サーバーの名前を入力します。
4. その他の既存値を削除して、ユーザーのユーザー ID およびパスワードを入力します。
5. **「終了」**をクリックします (必要な場合は、もう一度**「終了」**をクリックします)。
6. 他の 4 個のプロジェクト、すなわち、SV000514、SV000618、SV001585 および SV001586 に対しても、ステップ 2 から 5 を繰り返します。

ワークベンチでのアプリケーションの実行

この時点で、ワークベンチの WebSphere テスト環境でアプリケーションを実行できます。正常に実行していることが判明した場合は、EAR ファイルを iSeries サーバーにエクスポートして、それを iSeries WebSphere Application Server に展開できます。

ワークベンチで、お客様としてアプリケーションを実行

お客様として、小売店のショッピング Web ページから開始し、品目内を調べ、数量およびサイズを注文します。

注: ユーザーの web.xml ファイルはファイル http://java.sun.com/j2ee/dtds/web-app_2_2.dtd を検索するために、ファイアウォールの背後でアプリケーションを実行しようとすると、問題が起こることがあります。この問題を解決するには、アプリケーションを実行する前に、すべての web.xml ファイルの DOCTYPE ステートメントを次のように変更してください。

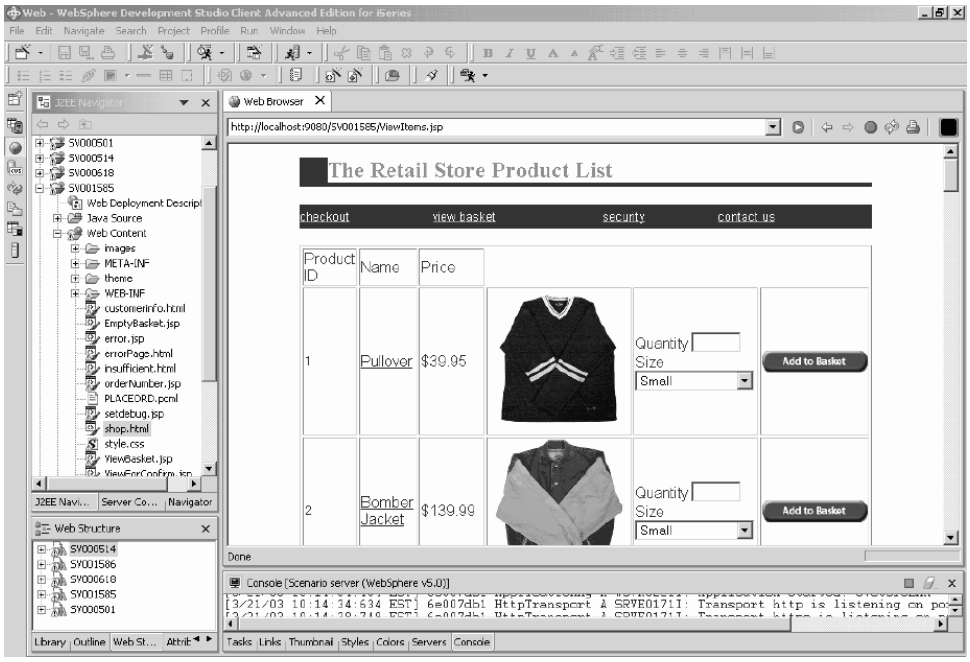
```
!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
"x:/WDSC/eclipse/plugins/com.ibm.etools.j2ee/dtds/web-app_2_2.dtd"
```

ここで、x:/wdsc は、その製品がインストールされたディレクトリーです。通常、ユーザーの web.xml ファイルは *「プロジェクト名」* **>「Web コンテンツ」>「WEB-INF」** にあります。ファイルをオープンした後、**「ソース」**タブをクリックして、そのファイルを直接編集します。

ワークベンチでお客様としてアプリケーションを実行するには:

1. Web パースペクティブでは、**「SV001585」>「Web コンテンツ」**を展開します。
2. **shop.html** を右クリックして、**「サーバーで実行」**を選択します。これで、ワークベンチ・ブラウザーでアプリケーションが起動します。
3. **「終了」**をクリックします。

4. T シャツのイメージをクリックすることによって、アプリケーションを入力します。
5. 次のページに値を入力してみましょう。ユーザーはお客様としてプルオーバーまたはボンバー・ジャケットを注文し、サイズと数量を選択し、ユーザーの買い物カゴに品目を追加します。

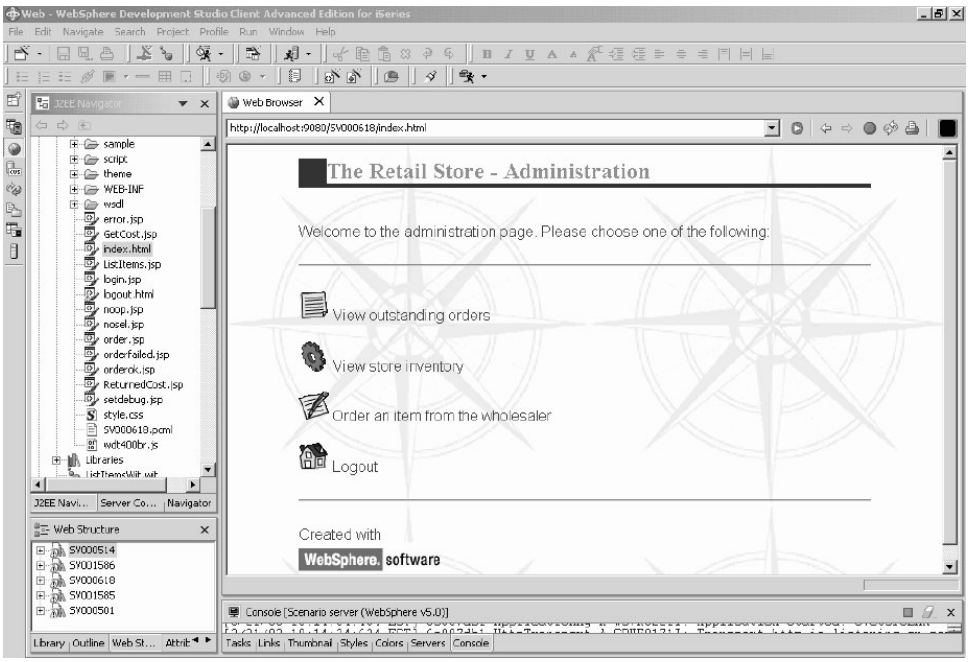


「注文は処理されました」のテキストが表示されるページは、このサンプルの最後のページです。このサンプルで満足すれば、そのブラウザーをクローズできます。

ワークベンチで、管理者としてアプリケーションを実行

管理者として、その小売店の管理 Web ページで開始し、注文または商店の在庫を検査します。ワークベンチで管理者としてアプリケーションを実行するには:

1. Web パースペクティブでは、**「SV000618」>「Web コンテンツ」**を展開します。
2. **index.html** を右クリックして、**「サーバーで実行」**を選択します。これで、ワークベンチ・ブラウザーでアプリケーションが起動します。
3. **「商店在庫の表示」**の左にあるアイコンをクリックし、その商店で購入するものを決定する管理者であるかのように、次のページを表示します。



そのアイコンをクリックして、管理ページを表示してみてください。

WebSphere Application Server へのシナリオ・アプリケーションの配置

この iSeries シナリオ・アプリケーションをワークベンチ・テスト環境で実行したら、そのアプリケーションを実際に実行するものとして、WebSphere Application Server に配置できます。ただし、アプリケーションを配置する前に、管理者ページを保護したり、次の項で説明するように、アプリケーションが正しい場所を指示するように Web サービスの URL を変更するなどの、いくつかの調整が必要となります。

注: WebSphere Application Server への配置はオプションです。さらに、次のセクションに進み、WebSphere Application Server でアプリケーションをテストしないでモジュールを完了することもできます。

WebSphere Application Server の構成

このセクションの前の方で、 19 ページの『ワークベンチでのアプリケーションの実行』を説明しました。iSeries サーバーに対してファイルは配置されているため、WebSphere Application Server を使用して iSeries でアプリケーションを実行できます。

WebSphere Application Server に対してアプリケーションを配置する (オプション) には、次を確認する必要があります。

- WebSphere Application Server バージョン 4.0、バージョン 5.0、または WebSphere Application Server Express インスタンスを iSeries サーバー上に作成しているか、あるいはこれに対するアクセスがあり、そのインスタンスが実行中である (WebSphere Application Server への配置をテストしたい場合のみ)。

- iSeries サーバーの HTTP および WebSphere Application Server インスタンスのポート番号が判明している。
- WebSphere Administrative Console バージョン 4.0 または 5.0 がワークステーションにインストールされている。

注: WAS のバージョン 5.0 では、コンソールはブラウザー・ベースであるため、ワークステーションにコンソールは不要です。 WAS の各バージョンの詳細については、オンライン文書を参照してください。

Web 使用可能 iSeries アプリケーションは WebSphere Application Server を使用して、 Web ユーザーのブラウザーと iSeries プログラムまたはデータ間を通信する Java™ サーブレットおよび JavaServer ページ™ (JSP) を実行します。また、同じ iSeries システムから HTML ページと JSP ファイルにサービスするには、そのシステム上に HTTPサーバーもなければなりません。Apache で稼働する IBM HTTP Server の使用をお勧めします。このサーバーの使用法に関する資料は、以下で調べることができます。 IBM HTTP Server for iSeries Documentation Center については http://publib.boulder.ibm.com/pubs/html/iseries_http/v5r1/index.htm

WebSphere Application Server は、各種のプロセスで生成された JavaServer ページ、Java Bean™、および Java サーブレットを実行します。 IBM WebSphere Application Server for iSeries の基本文書リソースおよび iSeries の IBM WebSphere Administrative Console は、以下の Web サイトで使用可能です。

- IBM WebSphere Application Server バージョン 4.0 Advanced Edition for iSeries については http://publib.boulder.ibm.com/was400/40/AE/english/docs/
- IBM WebSphere Application Server バージョン 4.0 Advanced Single Server Edition for iSeries については、 http://publib.boulder.ibm.com/was400/40/AEs/english/docs/

IBM WebSphere® Application Server の文書では、特に、 *J2EE* モジュール、*WebSphere Application Server* のインストール、および *WebSphere* 管理サーバーの複数インスタンスのセットアップ について十分理解してください。また、少なくとも インストール ・リンクにあるステップを実行することも必要となります。

WebSphere Administrative Console に必要となる PTF のインストール、構成、および入手の方法を調べるには、そのサイト・マップを使用してください。

管理者のページの保護

管理者のページ index.html は許可されたユーザーしかアクセスできませんし、適切に保護する必要があります。これは、Web アプリケーション・ロジック中で方針に基づいて実行するか、あるいは WebSphere のセキュリティー機能を使用して実行できます。このシナリオでは、WebSphere セキュリティーを使用して、ページを保護しています。 WebSphere Application Server V4.0 Advanced Edition を使用していることに注意してください。 WebSphere Application Server に関する情報は、以下の Web サイトで調べることができます。

- IBM WebSphere Application Server バージョン 4.0 Advanced Edition for iSeries については http://publib.boulder.ibm.com/was400/40/AE/english/docs/

- IBM WebSphere Application Server バージョン 4.0 Advanced Single Server Edition for iSeries については、
  http://publib.boulder.ibm.com/was400/40/AEs/english/docs/
- WebSphere のレッドブック・ページ: WebSphere Application Server バージョン 5.0 および WebSphere Application Server Express Edition バージョン 5.0 のレッドブックを検索するには、
  http://publib-b.boulder.ibm.com/redbooks.nsf/portals/Websphere

この他のバージョンの WebSphere Application Server を使用している場合は、Web リソースの保護に関するそのバージョンの文書を参照してください。

Web ページまたはサーブレットなどの Web リソースのセキュリティーは、Development Studio Client 内に、またはアプリケーション・アセンブリー・ツール中に構成できます。このシナリオでは、Development Studio Client を使用します。

この Web アプリケーションのセキュリティー構成およびプロパティーを検討するには:

1. Development Studio Client で、Web パースペクティブに切り替えます。
2. 「ナビゲーター」ビューでは、**「SV000618」>「Web コンテンツ」>「WEB-INF」**を展開します。
3. **web.xml** をダブルクリックして、web.xml ビューをオープンします。
4. **「セキュリティー」**タブをクリックします。
5. index.html の管理ページを保護するために、「セキュリティー」制約も定義されています。ビューの上部で、**「セキュリティー制約」**をクリックします。
6. リスト中の **SecurityContraint** の最初のインスタンスをクリックします。
7. 右にある **AdminPage** をクリックします。
8. **「編集」**をクリックして、**「Web リソース・コレクション」**ダイアログ・ボックスを呼び出します。 index.html の GET および POST メソッドが事前選択されていることに注意してください。
9. **「OK」**をクリックします。

「セキュリティーの役割」セクションでは、定義されたセキュリティーの役割の名前が「Administrator」であることが分かります。配置の時に、この役割に個人が割り当てられるので、index.html ページへのアクセスが付与されます。**「許可される役割」**セクションでは、このセキュリティー制約への管理者アクセスをその役割に付与していることが分かります。適切なこのセキュリティーにより、ユーザー ID およびパスワードなどの適切な証明書を提供した後で、管理者の役割に割り当てられたユーザーだけが index.html ページへのアクセスが許可されます。リソースが保護されている時には、WebSphere Application Server はまず始めにそのユーザーの認証を試みます。認証は、証明書を使用するか、あるいはユーザーに対してユーザー ID とパスワードのプロンプトを出して実行されます。このプロンプトは、基本認証ダイアログによるか、あるいはカスタム・フォームを使用して実行できます。

このシナリオでは、login.jsp の名前の独自のログオン・ページを設計しました。その認証プロンプトを構成するには、web.xml ビューで**「ページ」**タブを選択します。**「ログイン」**セクションでは、**「フォーム」**は認証メソッドとして事前選択されることに注意してください。また、「ログイン」ページの名前は login.jsp である

ことに注意してください。ログオンが正常に行われない時は、「エラー」ページが表示されます。この場合、アプリケーションは login.jsp ページを再表示します。

iSeries WebSphere Application Server デプロイメントのための EAR ファイルの作成

アプリケーションを iSeries WebSphere Application Server に配置するには、EAR ファイルを作成する必要があります。 EAR ファイルは、.ear の拡張子をもつ標準 Java アーカイブ (JAR) ファイルです。これには、複数の Web プロジェクトを含めて、ユーザーはそれを使用して Web アプリケーションにパッケージして、WebSphere Administrative Server (WAS) に展開できます。

EAR ファイルを作成するには:

1. Web パースペクティブに切り替えます。
2. 「ナビゲーター」ビューでは、**「SVStoreEAR」** を右クリックして、**「エクスポート」**を選択します。
3. 「エクスポート」ウィンドウでは、**「EAR ファイル」**をクリックして、**「次へ」**をクリックします。
4. **「リソースのエクスポート先」**の下では、**「参照」**をクリックし、その EAR ファイルを保管できる iSeries 統合ファイル・システム上のディレクトリーにナビゲートします。 (ユーザーのネットワーク・ドライブを iSeries IFS にマップしておく必要があります。)
5. **「ファイル名」**フィールドに `SVRetailStorEAR.ear` と入力して、**「オープン」**をクリックします。
6. **「終了」**をクリックします。
7. SVWholeSaleEAR に対してステップ 2 から 6 を繰り返します。

iSeries WebSphere Application Server への EAR ファイルの配置

これで、EAR ファイルが作成されたので、これを WebSphere Application Server に配置できます。

1. WebSphere Administrative Console をオープンします。
2. **Enterprise アプリケーション** を右クリックして、**「Enterprise アプリケーションのインストール」** を選択します。
3. **「アプリケーションのインストール」(*ear)** ラジオ・ボタンを選択します。
4. 上部の**「参照」**ボタン (下部のものは使用不可) をクリックします。
5. EAR ファイルをエクスポートした IFS ディレクトリーにナビゲートします。
6. **SVRetailStorEAR.ear** を選択します。
7. **「アプリケーション名」**フィールドに「RETAILSTOR」と入力します。
8. **「次へ」**をクリックして、管理者の役割の iSeries サーバー・ユーザー ID を入力します。
9. **「Web モジュールの仮想ホストの選択」**のタイトル・ページになるまで、**「次へ」**をクリックします。

10. 3 つのすべての Web モジュールについて、**「仮想ホストの選択」**をクリックし、ドロップダウン・リストから目的の仮想サーバーを選択します。 (どれを選択するかが分からない場合には、**デフォルト**または**デフォルト・ホスト**を使用します。)
11. **「次へ」**をクリックします。
12. 3 つのすべての Web モジュールについて、**「サーバーの選択」**をクリックして、使用したいサーバーを選択します。 (どれを選択するかが分からない場合には、**デフォルト・サーバー**を使用します。)
13. **「次へ」**をクリックします。
14. **「終了」**をクリックして、ダイアログ・ボックスで **「OK」**をクリックします。
15. **Enterprise アプリケーション**をもう一度右クリックして、**「Enterprise アプリケーションのインストール」** を選択します。
16. 下部の**「参照」**ボタン (上部のものは使用不可) をクリックします。
17. EAR ファイルを入れた IFS ディレクトリーにナビゲートします。
18. **SVWholesaleEAR.ear** を選択します。
19. **「アプリケーション名」**フィールドに「WHOLESALE」と入力します。
20. **「終了」**が使用可能になるまで、**「次へ」**のクリックを繰り返します。
21. **「終了」**をクリックして、ダイアログ・ボックスで **「OK」**をクリックします。

    注: 確認メッセージが表示されるのに、数分かかることがあります。

WebSphere Application Server でのアプリケーションの実行

お客様の役割になって、小売店エントリー・ポイントを実行するには、Web ブラウザーに次の URL を入力します。

```
http://your iSeriesHostName:yourHTTPPortNumber/SV001585/shop.html
```

管理者の役割になって、WholeSale エントリー・ポイントを実行するには、Web ブラウザーに次の URL を入力します。

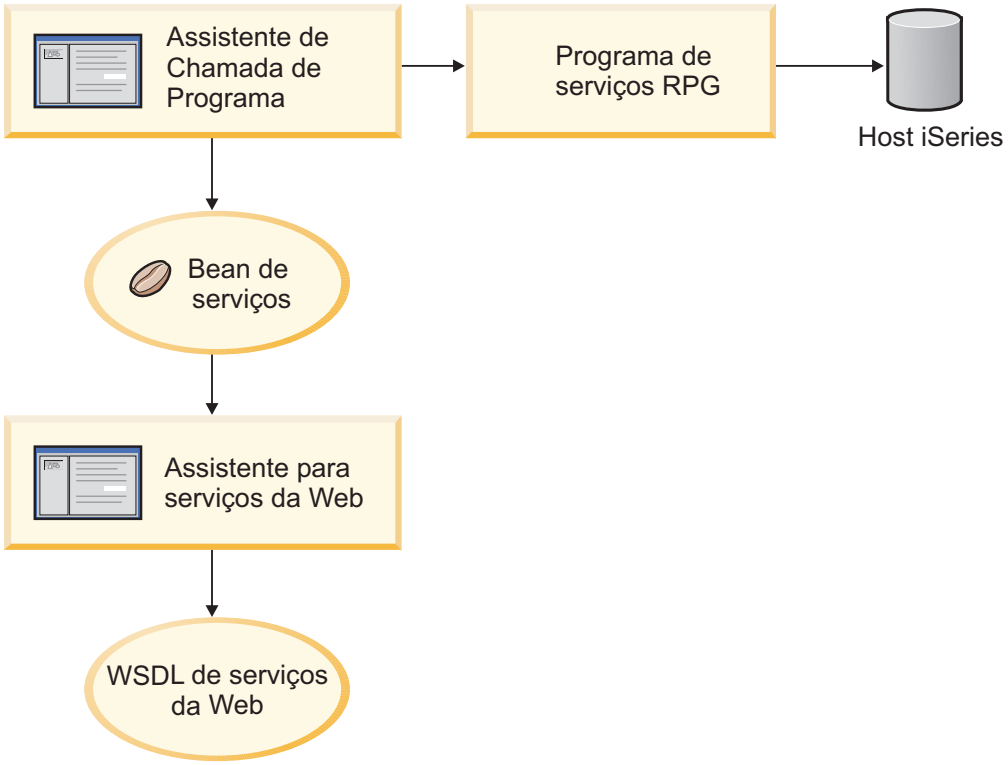
```
http://your iSeriesHostName:yourHTTPPortNumber/SV000618/index.html
```

HTTP ポート番号が分からない場合には、WebSphere Application Server の管理者に連絡してください。

第 4 章 段階的モジュール 1: 商品価格を返す Web サービスの作成 (SV000514)

概要

このモジュールでは、在庫の商品価格を表示するために、iSeries サーバー上に存在する RPG プログラムから Web サービスを作成します。まず始めに、品目番号を指定して、iSeries データベースから品目の原価を検索できるプロシージャーにより RPG サービス・プログラムを作成します。 iSeries Java 開発ツールから「プログラム呼び出し」ウィザードを使用して RPG プログラムを起動し、サービス Bean を作成します。次に、「Web サービス」ウィザードを使用して Web サービスを作成し、生成されたサンプルを使用してその Web サービスを検査します。



始める前に

この演習を完了できるのは、次の前提条件に適合する場合だけです。前提条件については、9 ページの『第 3 章 シナリオの実行』で詳しく説明されています。

- iSeries サーバーに TCP/IP でアクセスしている。
- 最新の iSeries サーバー PTF が適用されている。 PTF およびサービス・パック情報については、サポート・ページ (http://www.ibm.com/software/awdtools/wdt400/support/) を参照してください。
- iSeries サーバーをコマンド `STRTCPSVR *ALL` で開始した。

- WebFacing サーバーをコマンド STRTCPSVR *WEBFACING で開始した。
- 9 ページの『第 3 章 シナリオの実行』ですべてのタスク (ワークベンチでのアプリケーションのテストには不要なオプションの WebSphere Application Server タスクは除く) を完了した。

新規の Web プロジェクトの作成

この Web サービスを作成する最初のステップは、ユーザー情報を収容する新規の Web プロジェクトを作成することです。

1. ワークベンチ IDE から、Web パースペクティブに切り替えるか、あるいは**「ウィンドウ」>「パースペクティブのオープン」>「その他」 >「Web」>「OK」**をクリックして、Web パースペクティブをオープンします。
2. **「ファイル」>「新規」>「動的 Web プロジェクト」**をクリックします。
3. **「プロジェクト名」**フィールドに Project514 と入力します。
4. **「拡張オプションの構成」**チェック・ボックスを選択して、**「次へ」**をクリックします。
5. **「EAR プロジェクト」**フィールドの横にある**「新規」**ボタンをクリックします。
6. **「プロジェクト名」**フィールドに Project514EAR と入力し、両方のダイアログ・ボックスで**「終了」**をクリックします。

これで、Project514 および Project514EAR プロジェクトが「プロジェクト・ナビゲーター」ビューのワークスペースに追加されたことが分かります。

iSeries サーバー情報の定義

Web プロジェクトを作成した後、そのプロジェクトが情報の入手のためにどの iSeries サーバーを使用するかを定義する必要があります。

1. **Project514** プロジェクトを右クリックし、**「iSeries Web Tools 実行時構成の指定」**を選択します。
2. 復元された WHOLESALE ライブラリーが常駐する iSeries サーバーの名前 (たとえば PROD400) と入力します。
3. この iSeries サーバーのユーザー ID およびパスワードを入力します。
4. **「ライブラリー」**フィールドに Wholesale と入力し、**「追加」**をクリックします。
5. **「終了」**をクリックします。

RPG サービス・プログラムの作成

品目番号を指定して、アプリケーションによって品目の価格を検索したいことがあります。これは、QryProdCost と呼ばれるプロシージャーを含む RPG サービス・プログラムによって処理されます。 WHOLESALE ライブラリーには、CWWSSRV の名前の RPG サービス・プログラムが含まれています。このプログラムには、品目番号を入力値とする QryProdCost プロシージャーが含まれており、WHOLESALE ライブラリー内の在庫ファイルをオープンし、在庫データベースから価格を検索して、その価格を返すことができます。このタスクのサービスを行うには、インター

フェース時に 2 つのパラメーターがあります。その 1 つは品目番号で、もう 1 つは価格です。品目番号または価格が検出されないと、RPG プログラムはそのインターフェースにメッセージを戻します。

この Web サービスを作成するには、まず始めに、「プログラム呼び出し」ウィザードで Java Bean を作成して、QryProdCost RPG プロシージャーを呼び出します。次に、Web サービスを使用し、その Java Bean を介した Web サービスとして RPG プロシージャーを使用可能にします。

Java Bean を作成するには:

1. Web パースペクティブで、**Project514** を右クリックして、**「新規」>「その他」**を選択します。
2. ウィンドウの左側で**「iSeries」>「Java」**と選択し、ウィンドウの右側で**「プログラム呼び出し Bean」**を選択します。
3. **「次へ」**をクリックして、「プログラム呼び出し」ウィザードを呼び出します。
4. タイトル**「プログラムの追加」**の下にある**「Java Bean 名」**フィールドに Inventory と入力します。
5. **「プログラム・オブジェクト」**フィールドでは、RPG サービス・プログラムの名前の CWWSSRV と入力します。
6. **「ライブラリー」**フィールドでは、WHOLESALE と入力します。
7. **「プログラム・タイプ」**ドロップダウン・リストから、***SRVPGM** を選択します。
8. **「エントリー・ポイント」**フィールドに QryProdCost と入力します。
9. **「OK」**をクリックして、そのプログラム定義を追加します。

パラメーターの作成および Java Bean の生成

これで、プログラムを指定しました。パラメーターを追加することができます。CWWSSRV RPG プログラムには、次の 2 つのパラメーターが含まれています。

- 「品目番号」パラメーター - プログラムはこの番号を使用して、データベースで品目を検索します。
- 「品目原価」パラメーター - プログラムはこの番号を使用して、データベースでその品目の原価を検索します。

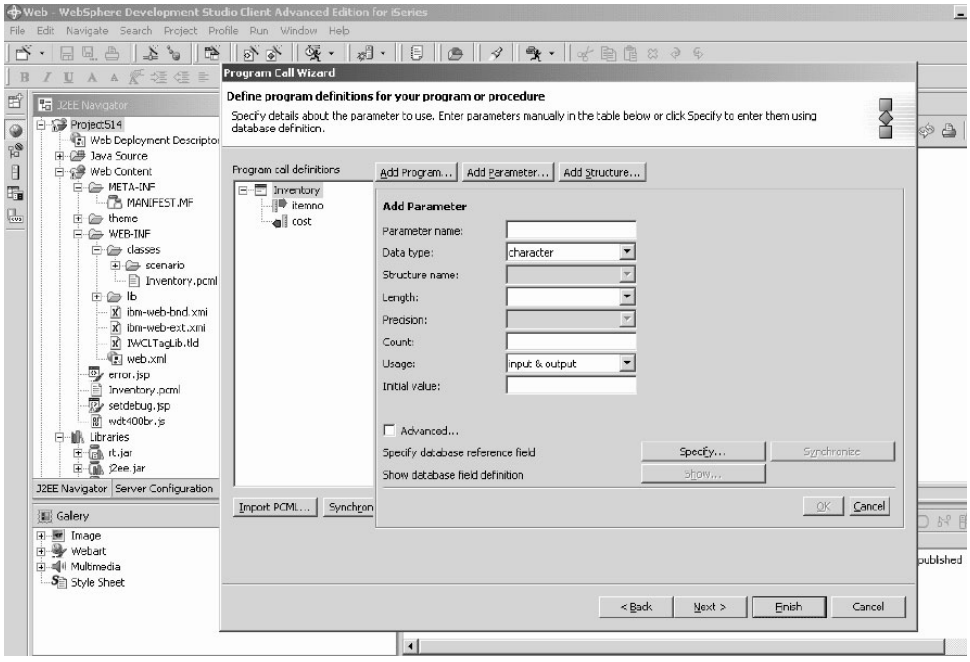
「品目番号」パラメーターを追加するには:

1. 「プログラム呼び出し」ウィザードの左のパネルで、 **Inventory** プログラム呼び出し定義をクリックして、それを選択します。このアクションによって、ページの右側にそのフィールドが再び移植されます。
2. **「パラメーターの追加」**をクリックします。
3. **「パラメーター名」**フィールドに、itemno と入力します。
4. **「データ・タイプ」**ドロップダウン・リストから、**パック 10 進数**を選択します。
5. **「長さ」**フィールドに、5 と入力します。
6. **「精度」**フィールドに、0 と入力します。
7. **「使用法」**ドロップダウン・リストから、**「入力」**を選択します。

8. **「OK」**をクリックして、このパラメーターを追加します。左側では、**itemno** が **「在庫」** の下に示されていることが分かります。これで、2 番目のパラメーターを追加する準備ができました。

「品目原価」パラメーターを入力するには:

1. 「プログラム呼び出し」ウィザードの左のパネルで、**Inventory** プログラム呼び出し定義を再びクリックして、それを選択します。このアクションによって、ページの右側にそのフィールドが再び移植されます。
2. **「パラメーターの追加」**をクリックします。
3. **「パラメーター名」**フィールドに、`cost` と入力します。
4. **「データ・タイプ」**ドロップダウン・リストから、**パック 10 進数**を選択します。
5. **「長さ」**フィールドに、7 と入力します。
6. **「精度」**フィールドに、2 と入力します。
7. **「使用法」**ドロップダウン・リストから、**「出力」**を選択します。
8. **「OK」**をクリックして、このパラメーターを追加します。左側では、**cost** が **「在庫」** の下に示されていることが分かります。この時点では、ウィザードはこのようになるはずです。パラメーターの左のアイコンは、これが入力、入出力、または出力のタイプであるかを示すことが分かります。



「プログラム呼び出し」ウィザードで、これらのパラメーターのパッケージ名を定義する場合

1. **「次へ」**をクリックします。
2. **「パッケージ」**フィールドに、パッケージ名として `scenario` と入力します。その他のフィールドはデフォルトのままにします。
3. **「Java アプリケーション」**チェック・ボックスをクリアします。

注: 「これらのファイルはウィザードによって生成されます」の下のファイルのリストを検討して、生成された Java Bean の名前が InventoryServices.java であることに注意してください。

4. **「終了」**をクリックして、ファイルを生成します。

Java Bean からの Web サービスの作成

RPG プログラムを呼び出す Java Bean を作成した後、次のステップは、他のプログラムがインターネットを介して同じ RPG プログラムにアクセスできるように、その Bean を Web サービスに変換することです。

Web サービスを作成する時に、「Web サービス」ウィザードは Web サービスを使用する必要があるユーザーに配布される WSDL ファイルを生成します。サービスを作成するには:

1. Web パースペクティブの「ナビゲーター」ビューでは、**「Project514」>「Java ソース」>「シナリオ」**を展開します。
2. **InventoryServices.java** を右クリックして、**「新規」>「その他」**を選択します。
3. 新しいウィンドウでは、**「Web サービス」**をクリックし、ウィンドウの右側で**「Web サービス」**をクリックしてから、**「次へ」**をクリックします。
4. **「プロキシーの生成」**および**「生成されたプロキシーのテスト」**チェック・ボックスを選択します。
5. **「次へ」**を 3 回クリックし、必要なファイルの生成を確認します。「Web サービス Java Bean 識別」ウィンドウが表示されたら、**「終了」**をクリックします。エラーがあった場合は、もう一度**「完了」**をクリックする必要があります。ウィザードを終了できない場合は、**「キャンセル」**をクリックすると、とりあえず必要なファイルの多くが作成されます。

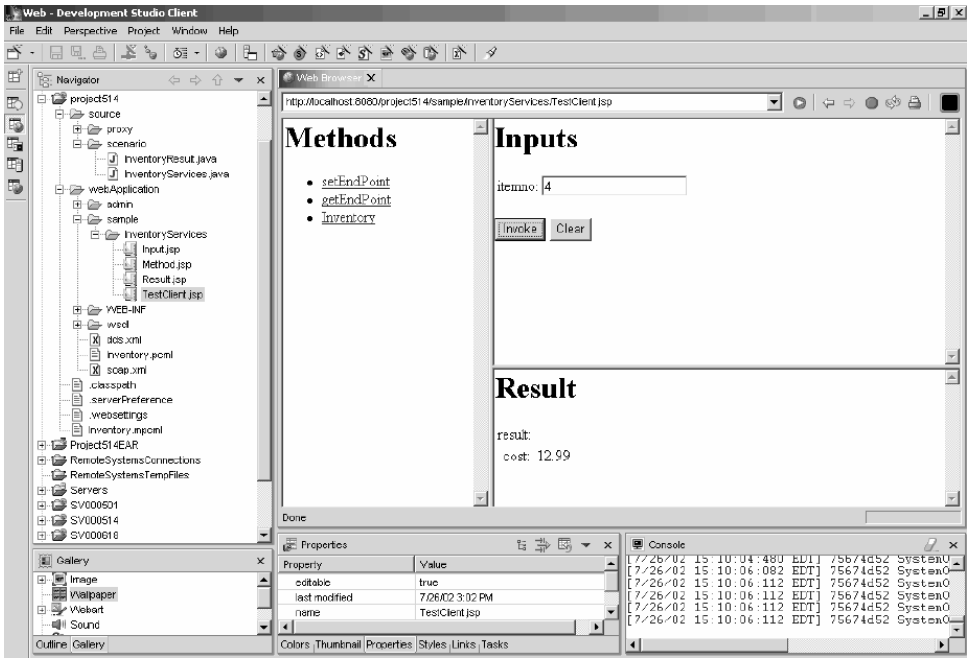
サンプルのテスト

Web サービスを作成した時の指示の 1 つは、サンプルの生成を要求することでした。オプションが選択されたので、「Web サービス」ウィザードは Web サービスのテストに使用できるテスト・ページで新規のプロジェクトを作成しました。**「ナビゲーター」**ビューでは、新規のこのプロジェクト名が **Project514Client** であることに注意してください。また、そのツールがワークベンチの右側で TestClient.jsp を自動的にオープンしたことにも注意してください。このファイルは **「Project514Client」>「Web コンテンツ」>「サンプル」>「InventoryServices」**の下にあります。

サンプルをテストするには:

1. ロードされたページでは、メソッド・ペインの最下部までスクロールダウンします。**「在庫 (java.math.BigDecimal)」**メソッドをクリックします。
2. **「itemno」**フィールドに、4 と入力して、 **「呼び出し」**をクリックします。
3. 戻された出力が次のとおりであることを検査します。

```
result:
cost: 12.99
```

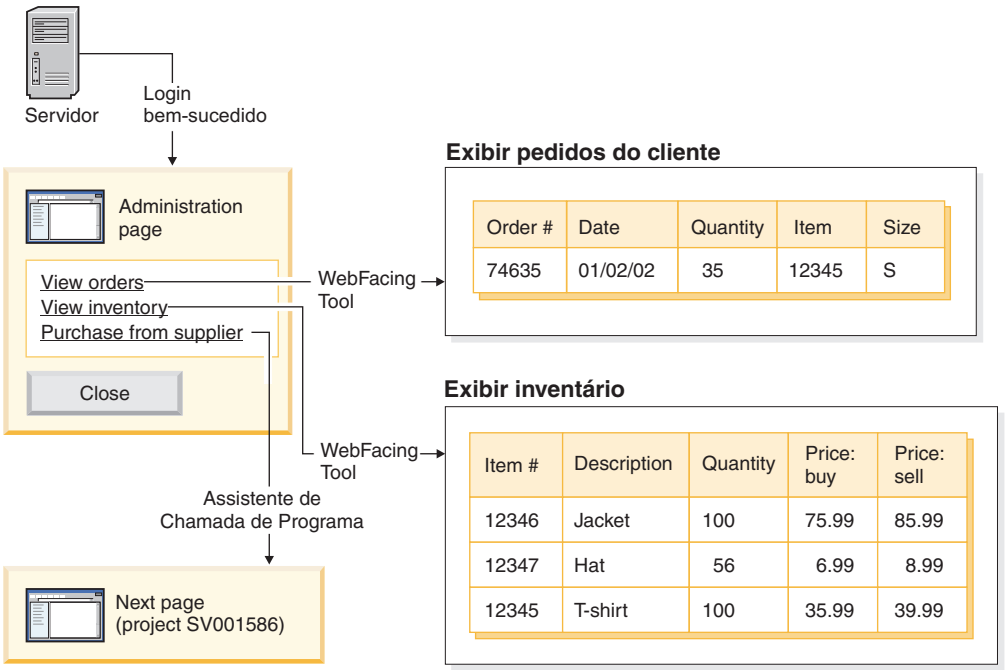


この結果を確認したら、そのモジュールを終了します。これで、アプリケーション・インターフェースで商品価格を返す Web サービスが作成されました。

第 5 章 段階的モジュール 2: 在庫および注文品目を表示するインターフェースの作成 (SV000501)

概要

このモジュールでは、IBM WebFacing Tool を使用して管理者の操作オプションを表示するインターフェースを作成します。 iSeries サーバーに正常にログインした後、インターフェースには使用可能な在庫が表示され、既存の注文を検査できます。管理者として、卸売り業者から商品を購入できます。



このプロジェクトでは、iSeries サーバーに保管した 2 つのプログラムおよび 2 つのディスプレイ・ファイルで作業を行います。そのプログラム名は ViewInventory と ViewOrder です。このプログラムは 2 つのディスプレイ・ファイル ORDERDSP と QUERY を使用します。この 2 つのファイルには、WebFacing Tool で生成され、イメージおよびハイパーリンクで使用される、 JSP ファイル用にカスタマイズされた Web 設定が含まれています。このイメージ Web 設定によって、フィールドのコンテンツを使用してイメージ・ファイルの名前を生成し、そのイメージを JSP ファイルに表示できます。ハイパーリンク Web 設定によって、JSP ファイルのイメージをクリックして別の Web アプリケーションの呼び出しを行うことができます。 CODE エディターまたは CODE 設計機能のいずれかを使用し、ディスプレイ・ファイルのソース・コードを検査して、Web 設定の指定方法を決定できます。

始める前に

この演習を完了できるのは、次の前提条件に適合する場合だけです。前提条件については、9 ページの『第 3 章 シナリオの実行』で詳しく説明されています。

- iSeries サーバーに TCP/IP でアクセスしている。
- iSeries サーバーをコマンド `STRTCPSVR *ALL` で開始した。
- WebFacing サーバーをコマンド `STRTCPSVR *WEBFACING` で開始した。
- 9 ページの『第 3 章 シナリオの実行』ですべてのタスク (ワークベンチでのアプリケーションのテストには不要なオプションの WebSphere Application Server タスクは除く) を完了した。

WebFacing プロジェクトの作成

最初に実行する必要があることは、WebFacing プロジェクトを作成して、関係のある CL コマンドを指定することです。WebFacing プロジェクトを作成するには:

1. ワークベンチでは、ワークスペースの左下にあるパースペクティブのアイコンの 1 つをクリックするか、あるいは**「ウィンドウ」>「パースペクティブのオープン」>「その他」>「WebFacing」**をクリックしてから、**「OK」**をクリックして、WebFacing パースペクティブに切り替えます。
2. **「ファイル」>「新規」>「WebFacing プロジェクト」**をクリックして、新規の WebFacing プロジェクトを作成します。
3. そのプロジェクトに `Project501` という名前を指定します。
4. **Enterprise アプリケーション・プロジェクト**の**「既存」**ラジオ・ボタンをクリックします。
5. **「既存のプロジェクト名」**フィールドの値に、`SVStoreEAR` と入力します。この値では大文字小文字が区別されません。**「次へ」**をクリックします。
6. **「J2EE レベル」**ドロップダウン・リストで、**1.3** を選択します。**「次へ」**をクリックします。
7. **「接続」**フィールドに、iSeries サーバーの名前が自動的に入っています。入っていない場合、「新規」をクリックして iSeries サーバーの名前をダイアログに入力し、**「終了」**をクリックします。
8. **「DDS リストのリフレッシュ」**をクリックし、ポップアップ・ダイアログにパスワードおよびユーザー ID を入力して、そのリストを最新表示します。
9. ライブラリーの生成リストでは、**RETAILSTOR** を展開します。
10. **QDDSSRC** をクリックしてから右矢印 (>>) をクリックし、そのファイルに移動します。
11. **「CL コマンドの指定」**ページに至るまで、**「次へ」**を 2 回クリックします。ここで、管理者のコマンドおよびコマンド・ラベルを追加する必要があります。
12. **「CL コマンド」**フィールドに、`call call viewinvent` と入力します。
13. **「コマンド・ラベル」**フィールドでは、既存のすべての値を削除して、`View inventory` と入力します。
14. **「指定した値でサインオンする」**ラジオ・ボタンを選択します。

15. **「追加」**をクリックし、ウィンドウの下部のリストに追加されることに注意してください。

ここで、2 番目のコマンドおよびお客様用のコマンド・ラベルを追加する必要があります。

16. **「CL コマンド」**フィールドは、既存の値を削除して `call vieworder` と入力します。
17. **「コマンド・ラベル」**フィールドは、既存のすべての値を削除して、`View orders` と入力します。
18. **「サインオンのプロンプトを出す」**ラジオ・ボタンを選択します。
19. **「追加」**をクリックし、ウィンドウの下部のリストに追加されることに注意してください。**「次へ」**をクリックします。
20. **「Web スタイルの選択」**ウィンドウでは、そのスタイル内をスクロールして、使用可能なものを調べます。後でこのスタイルから情報を検索する必要があるために、このモジュールでは、**avenue** を選択します。**「次へ」**をクリックします。
21. **「いいえ、今はプロジェクトを作成したいだけです」**チェック・ボックスを選択していることを確認して、**「終了」**をクリックします。
22. 「サーバー構成の修復」メッセージが表示された場合は、**「OK」**をクリックします。このメッセージは、SVStoreEAR ファイルにプロジェクトを追加したいことを確認するだけです。

完全な SV000501 プロジェクトのミラーとして Project501 を構成して、その Project501 が正しく表示するように、イメージ・ファイルの一部を SV000501 から Project501 にコピーする必要があります。(ただし、必要であれば、インターフェースのユーザー固有のイメージを追加できます。) イメージをコピーするには:

1. Web パースペクティブに切り替えます。
2. 「ナビゲーター」ビューでは、**「SV000501」>「Web コンテンツ」 >「イメージ」**を展開します。
3. (オプション) プロジェクトにインポートしているすべてのイメージのメモを取るために、**「生成済み」**を展開します。
4. **「生成済み」**を右クリックして、**「コピー」**を選択します。
5. **Project501** が再表示されるまで、「ナビゲーター」ビューをスクロールアップします。
6. **「Project501」>「Web コンテンツ」**を展開します。
7. **「イメージ」**フォルダーを右クリックし、ポップアップ・メニューから**「貼り付け」**を選択します。
8. この時点では、Project501 の**生成済み**フォルダーが存在しています。**生成済み**を展開して、追加したイメージのメモを取ります。

DDS ソースの変換

これで、プロジェクトが作成されたので、DDS ディスプレイ・ファイルを Web ページの JSP ファイルに変換できます。DDS ディスプレイ・ファイルを変換する時には、WebFacing Tool は、 DDS コードを置き換えて Web アクセスを可能にする JSP および XML ファイルを生成します。生成されたファイルにはレコード・フォーマットのデータが入れられるか、あるいはその外観をコントロールし、画面の

Web バージョンを表示し、データのプロンプトを出して、入力エラーを処理します。また、このウィザードはアプリケーションのホーム・ページを生成し、プログラムの Web 使用可能バージョンを起動します。

ただし、まず始めに、正しい iSeries サーバーを参照するようソースを変更する必要があります。さらに、アプリケーションのリンクが機能するように、DDS ディスプレイ・ファイルの 1 つを変更する必要もあります。デフォルト名 SV000501 をプロジェクト名 Project501 に変更する必要があります。

1. 「WebFacing プロジェクト」ビューがデフォルトでオープンされない場合は、WebFacing パースペクティブに戻ってこのビューに切り替えます。
2. **「Project501」>「DDS」**を展開します。
3. 2 番目のエントリー *<iSeriesserver>* **RETAILSTOR/QDDSSRC(QUERY)** をダブルクリックします。
4. 16 行目付近にスクロールダウンすると、次の行が表示されるはずです。

   ```
   A*%%WB 12 FLD 1 next ('/SV000501/DetailPage.do?PRODNO=&{PRODNO}')
   ```

5. `SV000501` を削除して、`Project501` と入力します。
6. ファイルを保管してクローズします。

注: 特定の時点で SV000501 プロジェクトを再び実行したい場合、このファイルは iSeries サーバー上にあり、両方のプロジェクトからアクセスされるため、Project501 値を SV000501 に戻す必要があります。

DDS ソースを変換するには:

1. WebFacing パースペクティブで、「WebFacing プロジェクト」ビューがデフォルトでオープンされていない場合、**「WebFacing プロジェクト」**タブをクリックしてこのビューに切り替えます。
2. **「Project501」>「DDS」**を展開します。
3. 最初の 1 つをクリックし、シフト・キーを押したまま 2 番目のものをクリックして、 *iSeriesserver* **> RETAILSTOR/QDDSSRC(ORDERDSP)** および *iSeriesserver* **> RETAILSTOR/QDDSSRC(QUERY)** を選択します。
4. **「変換」**を右クリックして選択し、変換を開始します。特定の時点で iSeries サーバーから切断した場合は、ユーザー ID およびパスワードの入力が必要となることがあります。

ワークベンチでの UTF-8 サポートの構成 — WAS バージョン 4.0 のユーザーのみ

IBM WebFacing Tool アプリケーションは、画面で複数言語の表示をサポートします。それぞれの言語は異なる文字セットを使用するので、ブラウザーと WebSphere Application Server 間のデータ・ストリームは UTF-8 によりエンコードされます。IBM WebFacing Tool を適切に機能するようにするには、ワークベンチのアプリケーションのプロパティー・ファイルに UTF-8 サポートを構成する必要があります。

注: このセクションが適用されるのは WAS 4.0 ユーザーに対してだけです。 WAS バージョン 5.0 はこのタスクを自動的に実行します。

UTF-8 サポートを構成するには:

1. Web パースペクティブに切り替えます (画面の左側の一群のアイコンをクリックして、パースペクティブ間の切り替えができます)。
2. **「プロジェクト・ナビゲーター」**タブをクリックして、プロジェクト構成を表示できるようにします。
3. **Project501** を展開して、**Web デプロイメント記述子**をダブルクリックします。
4. **「環境」**タブをクリックします。
5. **「追加」**ボタンをクリックします。
6. 左上のタスクバーで「サーバー・パースペクティブ」のアイコンをクリックしてそれに切り替えるか、あるいはメニュー・バーから**「パースペクティブ」>「オープン」>「その他」>「サーバー」**をクリックしてから、**「OK」**をクリックします。
7. 「ナビゲーター」ビューでは、**「サーバー」**フォルダーを展開します。
8. **defaultInstance.wsi** をダブルクリックして、デフォルト・エディターでそれをオープンします。
9. **「環境」**タブをクリックして、**「追加」**ボタンをクリックします。
10. 編集可能な値の「(新規の変数)」が表示されます。このデフォルト・ストリングを削除して、次を入力します。`client.encoding.override`。
11. **「値」**フィールドに、`UTF-8` と入力します。
12. 「保管」アイコンをクリックするか、あるいは**「ファイル」>「Web デプロイメント記述子の保管」**をクリックします。

WebSphere Application Server の UTF-8 サポートの構成

(オプション) iSeries アプリケーションを WebSphere Application Server に配置したい場合は、ワークベンチとともに、WebSphere Application Server にも UTF-8 サポートを構成する必要があります。

WebSphere Application Server 4.0 Advanced Edition に UTF-8 サポートを構成する場合

1. WebSphere Administrative Console を始動します。
2. **「ノード」**アイコンを展開して、**「ノード名」>「アプリケーション・サーバー」>「デフォルト・サーバー」**を展開します。
3. **「JVM 設定」**タブを選択し、**「拡張 JVM 設定」**ボタンをクリックして、**「拡張 JVM 設定」**ダイアログをオープンします。
4. **「コマンド行引き数」**フィールドに、次を入力します。

   ```
   -Dclient.encoding.override=UTF-8
   ```

5. **「OK」**をクリックし、**「JVM 設定」**タブの下で**「適用」**をクリックします。
6. この変更を WebSphere アプリケーションで有効にするには、そのデフォルト・サーバーを停止してから、それを再始動してください。サーバーを停止するには、**「デフォルト・サーバー」**を右クリックして、**「停止」**を選択します。この処理が完了した後、**「デフォルト・サーバー」**を右クリックして、**「開始」**を選択します。

WebSphere Application Server 4.0 Advanced Single Server Edition で UTF-8 サポートを構成する場合

1. WebSphere Administrative Console を始動します。
2. ブラウザー・ベース Administrative Console では、**「ノード」**アイコンを展開して、**「ノード名」>「アプリケーション・サーバー」>「デフォルト・サーバー」>「処理定義」>「JVM 設定」**を展開します。
3. 「JVM 設定」ページの「拡張設定」セクションにスクロールして、「システム・プロパティー」リンクをクリックします。「システム・プロパティー」ページが表示されます。
4. **「新規」**をクリックして、新規のシステム・プロパティーを追加します。
5. **「名前」**フィールドに、`client.encoding.override` と入力します。
6. **「値」**フィールドに、`UTF-8` と入力します。
7. **「OK」**をクリックします。**「JVM 設定」**ページの上のリンクで**「構成を保管する必要がある」**のメッセージを受け取った場合には、そのリンクをクリックして、**「構成の保管」**ページに進みます。**「保管」**を選択してから、**「OK」**をクリックします。
8. この変更を WebSphere アプリケーションで有効にするには、そのアプリケーション・サーバーを停止してから、それを再始動してください。アプリケーション・サーバーを停止して開始する方法は、WebSphere Application Server をインストールしたプラットフォームによって異なることがあります。アプリケーション・サーバーの停止および開始の情報については、ユーザーのプラットフォームの WebSphere Application Server 文書を参照してください。

スタイル・シートの作成

追加ページを Cascading Style Sheet (CSS) に統合したい場合は、WebFacing プロジェクトのスタイルか、その Cascading Style Sheet のいずれかをカスタマイズして、以下のように作成する必要があります。スタイル・シートのカスタマイズを終了した後、「Web 対話」ウィザードを使用し、そのスタイル・シートを使用して詳細な Web ページを作成し、価格や色などの保管品目に関する情報を表示できます。このモジュールの目的のために、SV000501 プロジェクトからの `DetailPageResults.jsp` スタイル・シートを結合します。ただし、将来の参照用に、CSS ファイルを手動でカスタマイズできます (次のセクションで概略を説明します)。

Cascading Style Sheet の手動のカスタマイズ (オプション)

前にも記述したように、Project501 の SV000501 プロジェクトからの `DetailPageResults.jsp` スタイル・シートを結合するのではなく、ユーザー独自のスタイル・シートをカスタマイズすることもできます。

1. 「プロジェクト・ナビゲーター」ビューに切り替えて、**「Project501」>「Web コンテンツ」>「スタイル」>「apparea」** を展開します。
2. **apparea.css** をダブルクリックして、CSS Designer でそれをオープンします。このスタイル・シートはアプリケーション域の外観を制御します。
3. そのファイルを変更して保管します。
4. 「プロジェクト・ナビゲーター」ビューに戻り、**「Project501」>「Web コンテンツ」>「スタイル」>「chrome」** とナビゲートします。

5. **avenue.css** をダブルクリックして、CSS Designer でそれをオープンします。このスタイル・シートはそのページ全体の外観を制御します。
6. そのファイルを変更して保管します。

Web 対話での WebFacing プロジェクトの展開および拡張

このセクションでは、WebFacing プロジェクトを拡張します。品目のリストが表示された時に、その品目の詳細を調べるためにその品目イメージをクリックします。これを行なうには、「Web 対話」ウィザードを使用して RPG プログラムを呼び出し、その品目の詳細を検索してそれを別の Web ページに表示します。次のステップを実行します。

- サーバー情報の定義
- 正しいスタイル・シートにコピー
- 対話の作成
- プログラムおよびパラメーターの対話への追加
- パラメーターの使用法の変更

まず始めに、サーバー情報を定義する必要があります。

1. Web パースペクティブに切り替えます。
2. 「ナビゲーター」ビューで、**Project501** を右クリックして**「iSeries Web Tools 実行時構成の指定」**を選択します。
3. iSeries サーバー名、ユーザー ID、およびパスワードを入力します。
4. **「ライブラリー」**フィールドに `Retailstor` と入力し、**「追加」**をクリックします。
5. **「終了」**をクリックします。必要な場合は、**「終了」** をもう一度クリックします。

Web 対話を作成する前に、Project501 が適切な JSP ファイル・フォーマットを表示するように、正しいスタイルおよびスタイル・シートにコピーする必要があります。 (演習 38 ページの『Cascading Style Sheet の手動のカスタマイズ (オプション)』を終了している場合は、このタスクを実行する必要はありません。) まず始めに、スタイル・フォルダーにコピーしてから、DetailPageResults.jsp にコピーします。

1. 「ナビゲーター」ビューで、**「SV000501」>「Web コンテンツ」**を展開します。
2. **「スタイル」**を右クリックして、**「コピー」**を選択します。
3. **Project501** が再表示されるまで、「ナビゲーター」ビューをスクロールアップします。
4. そのプロジェクトを展開し、**「Web コンテンツ」**を右クリックして、ポップアップ・メニューから**「貼り付け」**を選択します。**「はい」**をクリックして、既存のスタイルを上書きします。
5. **「SV000501」>「Web コンテンツ」**にスクロールして戻ります。
6. **DetailPageResults.jsp** を右クリックして、**「コピー」**を選択します。
7. **Project501** が再表示されるまでスクロールアップして戻ります。

8. そのプロジェクトを展開し、**「Web コンテンツ」**を右クリックして、ポップアップ・メニューから**「貼り付け」**を選択します。

Project501 の Web Content フォルダーの下に新規のエレメントが追加されていることが分かります。

これで、Web 対話を作成できます。

1. **「ファイル」>「新規」>「その他」**をクリックします。
2. **「新規」**ダイアログでは、左側の**「Web」**をクリックしてから、右側の**「Web 対話」**をクリックします。**「次へ」**をクリックします。
3. **「Web 対話名」**フィールドに、`DetailPage` と入力して、**「次へ」**をクリックします。
4. **「入力 JSP の生成」**ラジオ・ボタンを選択します。
5. **「出力ページの使用」**ラジオ・ボタンを選択し (これがまだ選択されていない場合) 、**「追加」**をクリックします。
6. **「出力 JSP」**ダイアログから、**「Web コンテンツ」**を展開し、**DetailPageResults.jsp** の事前フォーマット済み出力ページを選択して、**「OK」**をクリックします。
7. **「次へ」**をクリックします。

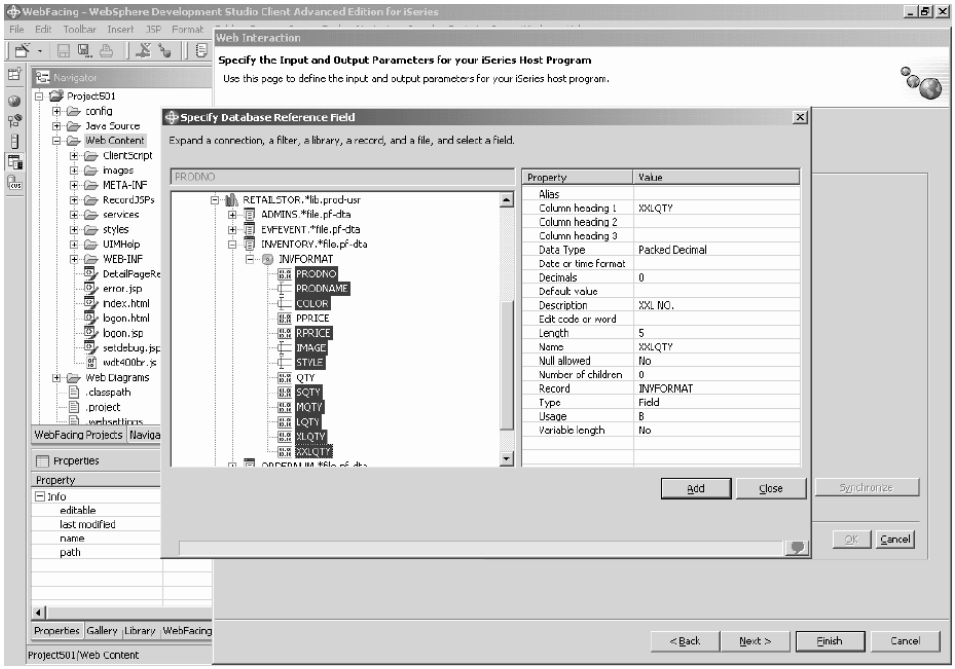
これで、ユーザーの対話にプログラムおよびパラメーターを追加できます。同じプログラムに 11 のパラメーターを追加する必要があります。個々の値をもつ各パラメーターを手動で追加する代わりに、もっと速い方法でこれらを追加できます。

1. まだ選択していない場合は、**「iSeries ILE プログラムの使用」**を選択します。
2. **「プログラムの追加」**をクリックします。
3. **「プログラム別名」**フィールドに、`DetailPage` と入力します。
4. **「プログラム・オブジェクト」**フィールドで、**「参照」**をクリックします。
   a. *「iSeriesserver」* **>「*LIBL」>「RETAILSTOR」** を展開します。
   b. **DETAILPAGE.*pgm.rpgle** (RETAILSTOR の下の最初のファイル) をクリックして、**「OK」**をクリックします。
5. 「Web 対話」ウィザードに戻り、**「OK」** (右下の方にある) をクリックします。

「プログラム呼び出し」定義の下のウィザードの左側に DetailPage が追加されたのが分かります。

6. ウィザードの左側の「プログラム呼び出し」定義のセクションでは、**DetailPage** をクリックしてそれを選択します。
7. **「パラメーターの追加」**をクリックします。
8. ウィザードの下部にある **「データベース参照フィールドの指定」** の横で、**「指定」**をクリックします。
9. *「iSeriesserver」* **>「*LIBL」>「RETAILSTOR」>「INVENTORY.*file.pf-dta」>「INVFORMAT」**を展開して、13 個のパラメーターのリストを表示します。以下の 11 個の追加では、それぞれを 1 回クリックして**「追加」**をクリックするか、あるいは CTRLキーを押したままそれぞれをクリックしてから、**「追加」**をクリックすることによっ

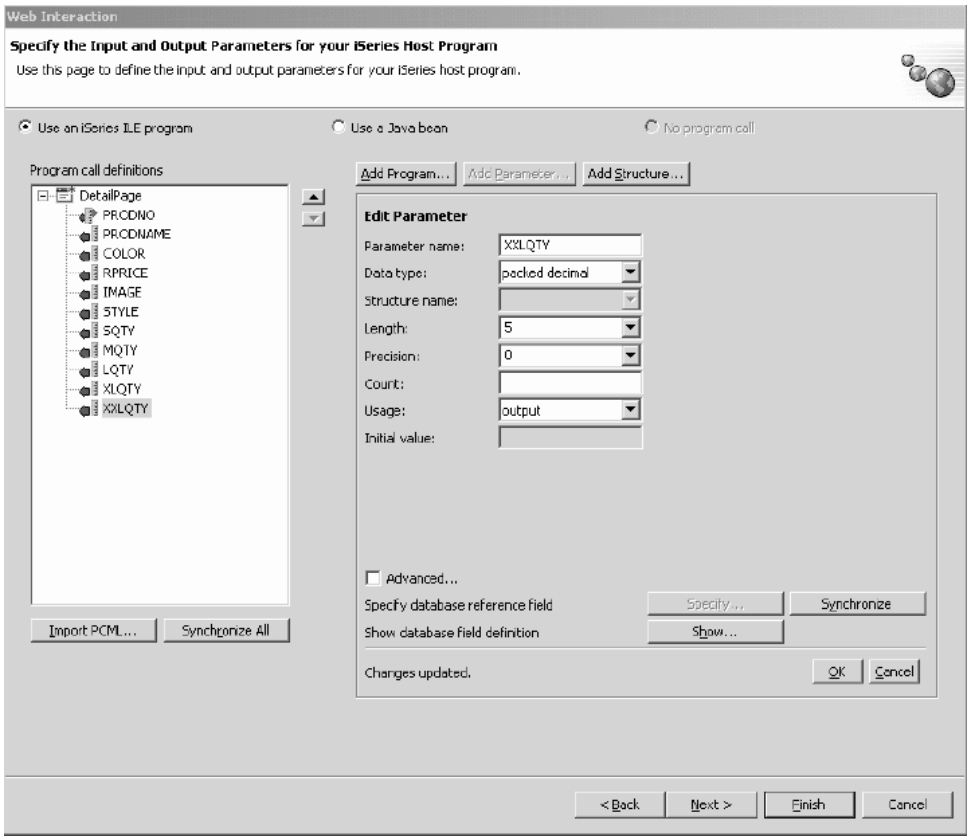
て、これらを追加する必要があります。すなわち、 PRODNO、 PRODNAME、 COLOR、 RPRICE、 IMAGE、 STYLE、 SQTY、 MQTY、 LQTY、 XLQTY、 XXLQTY です。基本的には、PPRICE と QTY を除いてすべてのフィールドを選択します。



10. **「クローズ」**をクリックします。

ここで、11 個のパラメーターの 10 個 (PRODNO を除くすべて) の使用法を、「出力」に変更する必要があります。

1. 「Web 対話」ウィザードでは、さらに、**PRODNAME** をクリックしてそれを選択します。
2. **「使用法」**コンボ・ボックスでは、選択した値を **「出力」**に切り替えて、**「OK」**をクリックします。最初のパラメーター (PRODNO) を除いて、すべてのパラメーターに対して、このステップ (前のステップも) を繰り返します。これで、ユーザーのワークスペースは、すべてのパラメーターを調整し終えた時のものとなるはずです。パラメーターの左のアイコンは、これが入力、入出力、または出力のタイプであるかを示していることが分かります。



3. **「次へ」**をクリックして、**「入力フォーム」**プレビューを表示します。
4. **「終了」**をクリックして、Web 対話を作成します。
5. **「すべてはい」**または**「OK」**を選択し、メッセージを受け取った場合は**「終了」**を再選択します。

プロジェクトの Web 対話へのリンク

これで、入出力パラメーターを使用する Web 対話 JSP ファイルが作成されたので、JSP ファイルが WebFacing コンポーネントでも機能するように、このファイルをカスタマイズする必要があります。コードを入力し、WebFacing アプリケーションからのリンクを作成して、この Web 対話を呼び出す必要があります。これを行なうには、新規のウィンドウで PRODNO パラメーターで DetailPageServlet サーブレットを呼び出すことができるように、`webface.js` ファイルに JavaScript 機能を追加しなければなりません。

リンクを作成するには:

1. 「ナビゲーター」ビューでは、**「Project501」>「Web コンテンツ」>「ClientScript」**を展開します。
2. **webface.js** をダブルクリックして、エディターでそれをオープンします。
3. ファイルの下部にスクロールして、次の行を入力します。

```
var mywindow
 function next(app)
{
mywindow = window.open(app,"Details","RESIZABLE=YES, HEIGHT=700, WIDTH=800");
}
```

4. 「保管」アイコンをクリックするか、あるいは **「ファイル」>「webface.js の保管」**をクリックします。

SV000501 と類似したアプリケーションを将来作成したい場合は、追加したイメージを使用可能にし、さらに、JavaScript 機能のウィンドウ・クローズ・リンクも使用可能にするために、DDS ソースの Web 設定を変更する必要もあります。

このアプリケーションに組み込まれた RPG コードはその変更を示すために変更されますが、将来のアプリケーションでの変更は手動で行う必要があります。さらに、Web 設定を変更した後で、DDS ソースを再変換する必要もあります。

そのコードをチェックして、その結果を複写できます。

DDS ソースを表示するには:

1. WebFacing パースペクティブに切り替えます。
2. 「WebFacing プロジェクト」ビューで、**「Project501」>「DDS」**を展開します。
3. **<iSeriesserver > RETAILSTOR/QDDSSRC(QUERY)**を右クリックして、**「オープン」>「CODE 設計機能」**を選択します。
4. CODE 設計機能をオープンした後、**「SCREEN1」> 「ITEMSUB」**を展開します。
5. **「IMAGESRC」**をクリックして、それを選択します。
6. **「ソース」**タブをクリックします。
7. ウィンドウの右下の**「Web 設定」**タブをクリックします。

   注: 幅（ピクセル数）およびファイル名などの Web 設定プロパティーをチェックします。将来、DDS ソースに対して同じ変更を行なってから、そのソースを再変換する必要があります。
8. ソースの以下の行に注意してください。

```
A     PRODNO  R   0 5 6
A     PRODNAME  R   0 5 16
A     IMAGESRC   19A 0 5 33
A*%%WB 13 FLD 100|100|&{IMAGESRC}
A*%%WB 12 FLD 1 javascript:next
   ('/Project501/DetailPageServlet?PRODNO=&{PRODNO}')
```

   特に、Project501 が最後の行に指定されることに注意してください。前にも説明したように、SV000501 アプリケーションをもう一度実行したい場合は、Project501 値を SV000501 に戻す必要があります。
9. 「保管」アイコンをクリックするか、あるいはメニュー・バーから **「ファイル」>「保管」**をクリックして、ファイルを保管します。
10. ファイルをクローズして、CODE 設計機能をクローズします。

DDS ソースでの操作の詳細については、ワークベンチのヘルプ・パースペクティブに切り替えて IBM WebFacing Tool の文書を参照してください。

ファイルの公開およびサーバーの再始動

このセクションでは、Project501 アプリケーションがすべての変更を選出できるように、ユーザー・ファイルを公開して再始動します。

ただし、サーバーを再始動する前に、Project501.war が SVStoreEAR ファイルに追加されていることを確認しなければなりません。

構成を確認するには:

1. 「ナビゲーター」ビューで、**「シナリオ・フォルダー」**を展開します。
2. **Scenario server.wsi** タブをダブルクリックします。
エディターによってプロジェクト入力を自動的に修正したいかどうかを尋ねるダイアログが表示されます。**「はい」**をクリックします。ダイアログが表示されない場合は、アクションは不要です。いずれの場合も、ファイルのオープン時にそれが自動的に変更されます。
3. ファイルを保管してクローズします。

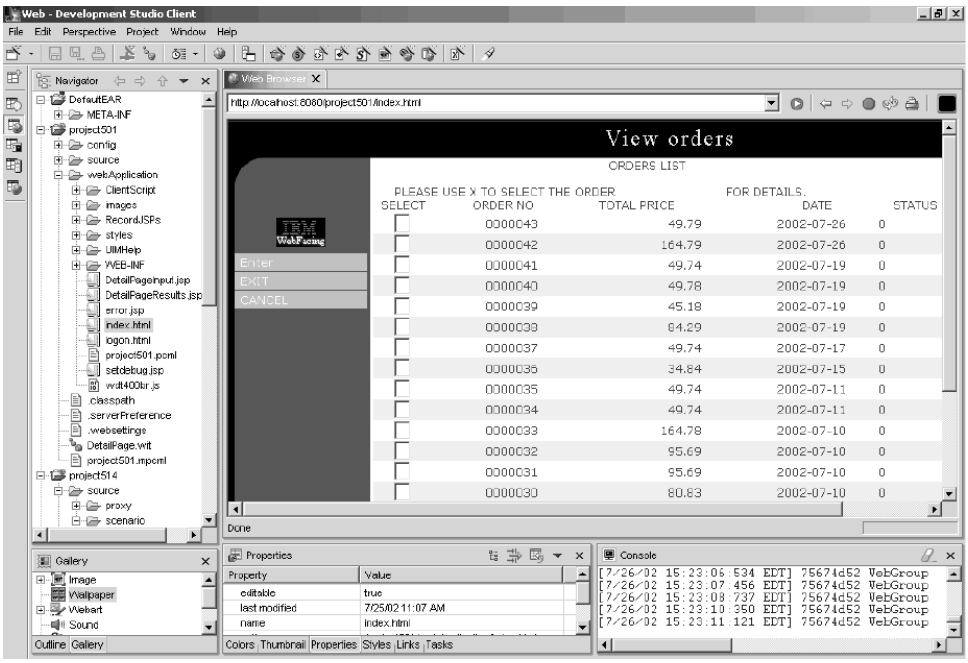
サーバーを再始動するには:

1. サーバー・パースペクティブに切り替えます。
2. 左下の「サーバー構成」ビューでは、**「サーバー」**を展開して、**「Scenario サーバー」**をダブルクリックします。
3. 画面の右下の**「サーバー」**タブをクリックして、そのサーバーの状況を「サーバー」ビューに表示します。
4. **Scenario サーバー**を右クリックして、**「公開」**を選択します。公開が完了した時に**「OK」**をクリックします。
5. 「サーバー」ビューでユーザーのサーバーを右クリックして、**「開始」**または**「再始動」** (使用可能などちらか) を選択します。未保管ファイルのプロンプトが出された場合には、そのダイアログを取り消し、すべてのオープン・ファイルを保管して続行します。
6. コンソール・ログ情報を検査します (自動的にオープンされる)。「サーバー *server_name* は e-business 用にオープンされます」の行がログの下部に示された時点でサーバーが開始されることが分かります。

インターフェースのテスト

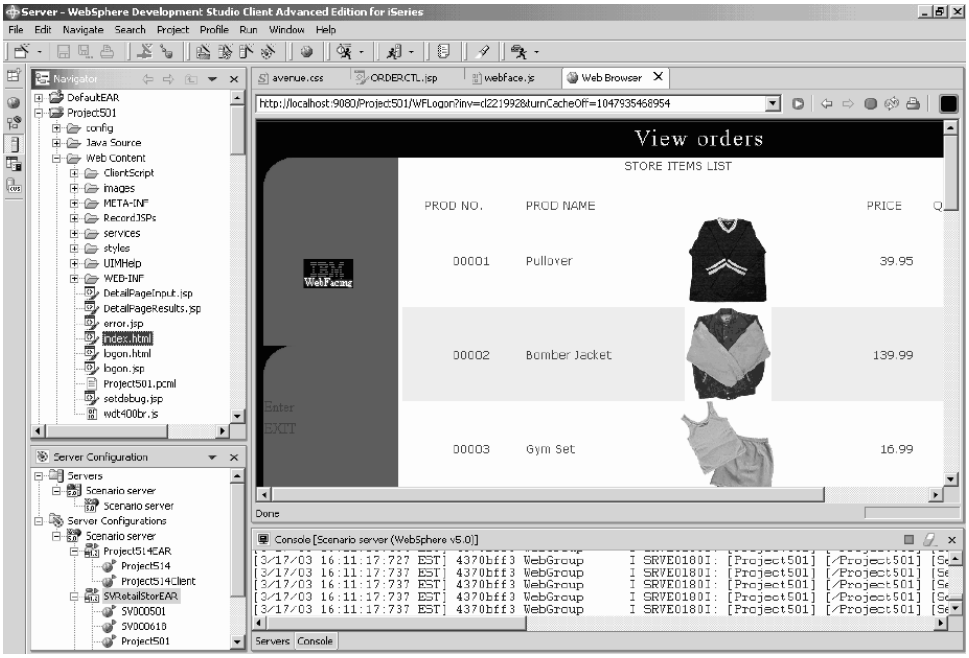
これで、注文を表示し在庫を表示するためのインターフェースの作成に必要なステップが完了しました。インターフェースをテストするには:

1. Web パースペクティブに切り替えます。
2. **「Project501」>「Web コンテンツ」**を展開します。
3. **index.html** を右クリックして、**「サーバーで実行」**を選択します。
4. **「サーバー選択」**ダイアログがオープンします。**Scenario サーバー**がデフォルトによって選択されていることを確認して、**「終了」**をクリックします。
5. **「注文の表示 – メイン・ブラウザー・ウィンドウで起動」**をクリックして、アドミニストレーター・アプリケーションを起動します。 iSeries ユーザー ID およびパスワード (iSeries サーバーで、このプロジェクトの開発中に使用したもの) でログインした後、次のページがオープンします。



どれかの商品番号の横に X と入力して、その商品の詳細を表示してみてください。

6. 戻り矢印をクリックして index.html ページを表示し、**「在庫の表示 – メイン・ブラウザーで起動」** ウィンドウをクリックしてカスタマー・アプリケーションを起動すると、次のページがオープンします (ユーザー ID およびパスワードで iSeries サーバーにログオンした後)。



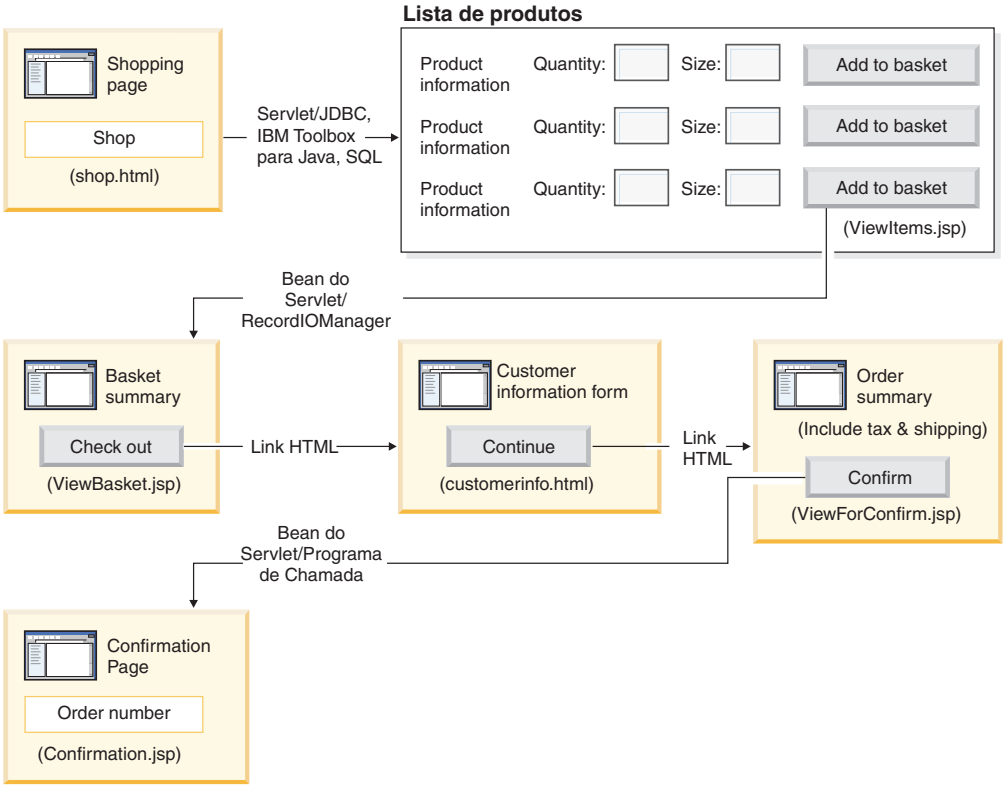
1 つのイメージをクリックして、特定品目に関する詳細を表示してみてください。

これで、このモジュールは終了しました。在庫および注文品目を表示するためのユーザー・インターフェースが作成されました。

第 6 章 拡張モジュール 1: iSeries サーバー上にお客様の注文を入れる HTML、サーブレット、および JSP ファイルの作成 (SV001585)

概要

このプロジェクトは、Java プログラミングに詳しい知識があり、iSeries データ管理および RPG にある程度の知識があるユーザーを対象にしています。プロジェクトは、iSeries サーバー上にお客様の注文を入れる HTML コード、サーブレット、および JSP を作成するための、 IBM Toolbox for iSeries データ・アクセス・クラス、RecordIOManager Bean、およびプログラム呼び出し Bean の処理方法を示します。ユーザーとして、ショッピング・ページから選択可能な製品を表示して、項目を買い物カゴに入れます。選択したすべての項目 (買い物カゴの中身) に満足したときには、「勘定」ボタンをクリックしてお客様情報フォームに進みます。フォームを完了すると、プロジェクトは確認ボタンと共に注文の要約を戻し、注文および注文番号を表示した確認ページが示されます。



高水準ステップの要約

これは拡張モジュールであるため、指示はプロジェクト作成の各ステップには進みませんが、そのようなプロジェクトを作成するために実行される iSeries 特定の展開ステップを概説します。以下は高水準ステップです。

1. HTML ショッピング Web ページを作成します。
2. お客様が購入するために選択可能な項目と共に Java Bean を取り込む (JDBC および SQL を使用) サーブレットを作成します。
3. 販売用項目を表示して、お客様が必要な項目の数量とサイズを入力して**「買い物カゴに入れる」**ボタンをクリックすることにより項目を選択できる、JSP ファイルを作成します。
4. RecordIOManager Bean を使用して、**「買い物カゴに入れる」**ボタンをクリックして呼び出すサーブレットを作成します。これは、項目に必要な数量とサイズを減算して、この選択を「買い物カゴ」と呼ばれる Java Bean に追加することにより、iSeries INVENTORY データベースを更新します。操作が正常に実行されると、次にサーブレットは応答を `ViewBasket.jsp` に転送します。操作が正常に実行されないと、サーブレットはエラー・ページを表示します。
5. 個人情報を入力するお客様用 HTML フォームを作成します。
6. 買い物カゴの内容、税、および配送と処理の代金を表示する JSP ファイル購入確認ページを作成します。このページには、お客様用に確認ボタンがなければなりません。
7. お客様が確認ボタンをクリックすると呼び出されるサーブレットを作成します。これは、iSeries プログラム呼び出しウィザードで作成される Java Bean を使用します。 Java Bean のメソッドの 1 つが RPG プログラムを呼び出して、お客様の買い物カゴの内容に対応して iSeries サーバー上の ORDERS データベースに新しい注文を作成します。次に、サーブレットは注文番号を戻し、Java Bean を Web アプリケーションのセッションに入れ、注文番号が入っている注文確認 JSP ファイルをロードします。

始める前に

この演習を完了できるのは、次の前提条件に適合する場合だけです。前提条件については、9 ページの『第 3 章 シナリオの実行』で詳しく説明されています。

- iSeries サーバーに TCP/IP でアクセスしている。
- iSeries サーバーをコマンド `STRTCPSVR *ALL` で開始した。
- WebFacing サーバーをコマンド `STRTCPSVR *WEBFACING` で開始した。
- WHOLESALE および RETAILSTOR ライブラリーを iSeries サーバーに復元した。
- WHOLESALE, RETAILSTOR および QGPL ライブラリーがユーザーのライブラリー・リストに入っている。
- 9 ページの『第 3 章 シナリオの実行』ですべてのタスク (ワークベンチでのアプリケーションのテストには不要なオプションの WebSphere Application Server タスクは除く) を完了した。

Web ページ、サーブレット、および JSP ファイルの作成

プロジェクト SV001585 のコンポーネントを構成するには、以下のようにしてください。

1. Web プロジェクトを作成します。
2. Page Designer を使用して、GetItems サーブレットを起動するリンクが入っている `shop.html` ページを作成します。
3. iSeries Toolbox for Java クラス用の jt400.jar ファイルを Web プロジェクト *lib* フォルダーにインポートします。この jar ファイルは *x*:¥wdsc¥wssd¥plugins¥com.ibm.etools.iseries.toolbox¥runtime に入っています。ここで、*x* は Development Studio Client をインストールしたディレクトリーです。

   注: SV001585 プロジェクトの `GetItems.java` および `ViewItems.jsp` を表示すると、サーブレットと JSP に対して JDBC および SQL 関連パーツ用のメイン iSeries Toolbox for Java が表示されます。 Web パースペクティブのナビゲーター・ビューでは、`GetItems.java` は**「SV001585」>「ソース」**を展開して表示し、`ViewItems.jsp` は**「SV001585」>「webApplication」**を展開して表示してください。

   .
4. JDBC および SQL 用 iSeries Toolbox for Java を使用して GetItems サーブレットを作成して、以下を実行するために iSeries INVENTORY データベースから衣類の項目を検索します。
   a. SQL 照会結果が入っている ResultSet Bean をセッションに入れる
   b. `ViewItems.jsp` に対する要求をリダイレクトする

   `GetItems.jsp` のコード・サンプルは以下のとおりです。

```
public void init() {
   .
   .
   .
   // Load the IBM Toolbox for Java JDBC driver.
   DriverManager.registerDriver(new com.ibm.as400.access.AS400JDBCDriver());
   // Note that we have retrieved the as400 name, userid, and password from
   // web.xml file using and xml parser.
   as400conn =
    DriverManager.getConnection(
     "jdbc:as400://" + as400 + ";naming=sql;errors=full",
     userid,
     password);

   dmd = as400conn.getMetaData();
   .
   .
   .
}

public void service(HttpServletRequest request, HttpServletResponse response){
   .
   .
   .
   Statement select =
    as400conn.createStatement(
     ResultSet.TYPE_SCROLL_INSENSITIVE,
     ResultSet.CONCUR_READ_ONLY);
   ResultSet rs =
```

```
    select.executeQuery(
     "SELECT PRODNO, PRODNAME, RPRICE, IMAGE FROM "
      + retailLibrary
      + dmd.getCatalogSeparator()
      + inventoryFile);

   HttpSession session = request.getSession(true);
   session.setAttribute("resultset", rs);

   response.sendRedirect("/ViewItems.jsp");

   .
   .
   .
}
```

5. 前のステップで入手した ResultSet Bean から衣類項目を検索する ViewItems JSP ファイルを作成して、衣類項目を表形式で表示します。また、JSP ファイルには、サイズと数量を選択して項目を買い物カゴに入れるために使用できる、各項目のフォームも入っている必要があります。 iSeries Web 開発ツールの Page Designer を使用して、JSP を作成できます。つまり、デザイン・ビューでページを配列し、ソース・ビューで適切なコードを追加できます。 ViewItems.jsp のコード・サンプルは以下のとおりです。

```
<!--Getting the ResultSet Object from the session--><%
   int columnCount = 0;
   ResultSet rs = (ResultSet)session.getAttribute("resultset");
   if(rs !=null)
%>
<%
  {
  rs.beforeFirst();
            ResultSetMetaData rsmd = rs.getMetaData ();
            columnCount = rsmd.getColumnCount ();

%>
<TABLE border="1">
<TBODY>
<TR>
  <TD>Product ID</TD>
  <TD>Name</TD>
  <TD width="551">Price</TD>
  <TD colspan="2"></TD>
</TR>
<%while (rs.next ()){
<TR>
<!--Creating a form for this row (or this item)-->
<FORM name="myform" action="/SV001585/AddtoBasket" onsubmit="return errorChecking(this);">
<!--Getting each column data from this row of ResultSet object-->
<!--Process data is a user defined method to modify the data for
display if needed-->
          <%
                for (int i = 1; i <= columnCount; ++i){
                    String value = rs.getString(i);
                    if (rs.wasNull ())
                        value = "<null>";
                    else{
                     if(i==1)
                      prodID=value;
                     value = processData(i,value);

                   }
     %>
    <TD><%=value%></TD>
      <%
                }
   %>

<!--Creating quantity input field and size drop down menue-->
<!--Note that we are using product id as the name of the field-->
         <TD width="290">Quantity
```

```
   <INPUT size="5" type="text" name='<%=prodID+"Q"%>' ><BR>
     Size <SELECT name='<%=prodID+"S"%>'>
       <OPTION value="s" selected>Small</OPTION>
       <OPTION value="m" selected>Medium</OPTION>
       <OPTION value="l" selected>Large</OPTION>
       <OPTION value="XL" selected>Extra Large</OPTION>
       <OPTION value="XXL" selected>Extra Extra Large</OPTION>
       </SELECT>
</TD>
<TD><INPUT type="image"   name="submit" src="images/Add_to_basket.gif"></TD>
</FORM>
</TR>
<%
}
%>
</TBODY>
</TABLE>
<%
}
```

6. Web プロジェクト用に iSeries Java 開発ツール iseriesut.jar ファイルを *lib* フォルダーにインポートします。この JAR ファイルは *x*:¥wdsc¥wssd¥plugins¥com.ibm.etools.iseries.toolbox¥runtime に入っています。ここで、*x* は製品をインストールしたディレクトリーです。インプリメンテーションを表示するには、SV001585 プロジェクトの AddtoBasket.java および ViewBasket.jsp を表示してください。Web パースペクティブのナビゲーター・ビューで、AddtoBasket.java は**「SV001585」>「ソース」**を展開して表示し、ViewBasket.jsp は**「SV001585」>「webApplication」**を展開して表示できます。

7. iSeries Java 開発ツールから RecordIOManager Bean を使用して、**「買い物カゴに入れる」**ボタンによって呼び出される AddtoBasket サーブレットを作成します。これは、お客様が要求した数量を減算し、項目をセッションの買い物カゴ Java Bean に追加することによって、iSeries INVENTORY データベースを更新します。AddtoBasket.jsp コード・サンプルは以下のとおりです。

```
public class AddtoBasket extends HttpServlet {

 //Inner class of AddtoBasket
 public class MyRecordIOManager extends RecordIOManager {
  .
  .
  .
  public MyRecordIOManager(
     String hostInfo1,
     String hostInfo2,
     String hostInfo3,
     String file,
     String lib)throws Exception{
   super(hostInfo1, hostInfo2,hostInfo3,file,lib);
   setFileAccessType(RecordIOManager.FILEACCESS_KEYED);
   setCommitLockLevel(RecordIOManager.COMMITLOCKLEVEL_ALL);
   //journal has the same name as the database file
   setJournal(file);
   //journal is in the same library as the database file
   setJournalLibrary(lib);
  }
  .
  .
  .
  public synchronized String updateDBFile(
   String id,
   String size,
   String quantity
   ) {
```

```
  .
  .
  .
  //opening the file
  try {
   if (openFile()) {
    record = readRecord(key);
    quantityAvailable = ((BigDecimal)
     record.getValueAt(0,sizeColumn)).intValue();
    totalQuantityAvailable = ((BigDecimal)
     record.getValueAt(0, 8)).intValue();
    if (quantityRequested <= quantityAvailable) {
     newQuantity =
        new BigDecimal(quantityAvailable - quantityRequested);
     totalNewQuantity =
        new BigDecimal
      (totalQuantityAvailable - quantityRequested);
     record.setValueAt(newQuantity, 0, sizeColumn);
     record.setValueAt(totalNewQuantity, 0, 8);
     // Note that we update the record but we don't commit
     // in case the customer decides to
     // empty the basket in which
     // case we call the rollBack method
     updateRecord(record);
     status = success;
    } else {
     status = notEnough;
    }
   } else
    status = accessError;
  } catch (Exception e) {
   e.printStackTrace();
   status = accessError;
  }

  //closing the file and adding
  try {
   closeFile();
  } catch (Exception e) {
   //in case of error rollback
   try {
    rollback();
   } catch (Exception e1) {
    e1.printStackTrace();
   }
   status = accessError;
  }

  return status;
 }
}

//init method of AddtoBasket servlet
public void init() {
 hostInfo = GetItems.getHostInfo();

}

public void doGet(HttpServletRequest req, HttpServletResponse res) {
 .
 .
 .
 Basket basket = (Basket) session.getAttribute("basket");
 MyRecordIOManager recIO =
 (MyRecordIOManager) session.getAttribute("recIO");
 if (basket == null) {
  basket = new Basket();
```

```
 session.setAttribute("basket", basket);
}

if(recIO == null){
 if (recIO == null) {
 try {
  recIO =
   new MyRecordIOManager(
    hostInfo[0],
    hostInfo[1],
    hostInfo[2],
    GetItems.getInventoryFile(),
    GetItems.getRetailLibrary());
 } catch (Exception e) {
  try {
   res.sendRedirect("errorPage.html");
   return;
  } catch (Exception e1) {
   e1.printStackTrace();
  }
 }
}
id = req.getParameter("id");
size = req.getParameter(id + "S");
quantity = req.getParameter(id + "Q");

status = recIO.updateDBFile(id, size, quantity);
session.setAttribute("recIO", recIO);

if (status.equals("SUCCESS")) {
 basket.addItem(id, quantity, size);
 try {
  res.sendRedirect("ViewBasket.jsp");
  return;
 } catch (Exception e) {
  e.printStackTrace();
 }
} else {
 if (status.equals("NOT_ENOUGH")) {
  try {
   res.sendRedirect("insufficient.html");
   return;
  } catch (Exception e) {
   e.printStackTrace();
  }
 } else
  if (status.equals("ACCESS_ERROR")) {
   try {
    res.sendRedirect("errorPage.html");
    return;
   } catch (Exception e) {
    e.printStackTrace();
   }
  }

 }

}
```

8. 買い物カゴの内容を表示する ViewBasket JSP ファイルを作成します。
9. 支払情報を入力するための customerinfo.html フォームを作成します。これには、ViewForConfirm.jsp を呼び出す **「次へ」**ボタンがあります。

10. 買い物カゴの全体の内容と合計差引金額を表示する ViewForConfirm.jsp を作成します。 ViewForConfirm.jsp は、計算された配送料と**「確認」**ボタンを iSeries ORDERS データベースでの注文に加えると、ViewBasket.jsp と同じ方法で作成できます。
11. iSeries Java 開発ツールの iSeries プログラム呼び出しウィザードを使用して、PLACEORD.java Bean を作成します。これは、RETAILSTOR ライブラリーの PLACEODR サービス・プログラムにアクセスします。ウィザードは、Java アプリケーションまたは Web サービス・ウィザードによって使用される Bean を作成して、iSeries ILE プログラムにアクセスします。
    a. ウィザードを開くには、ナビゲーター・ビューで SV001585 を右クリックして、**「新規作成」>「その他」**を選択します。
    b. **「新規作成」**ウィンドウで、**「iSeries」>「Java」>「プログラム呼び出し Bean」**をクリックします。
    c. プログラム呼び出しウィザードに、iSeries ILE プログラム名、ライブラリー、プログラムのタイプ、および入出力パラメーターについての情報を入力します。
    d. ウィザードの最後のウィンドウには、Java アプリケーションまたは Web サービス、あるいはその両方の Bean を作成するためのオプションが示されます。このプロジェクトで作成する必要があるのは、Java アプリケーション用のものだけです。

    注: PLACEODR サービス・プログラムは構造の配列を使用して、各エレメントを ORDERS データベースのレコードに入れ、それぞれの配列ごとに出力として注文番号を生成します。
12. ViewForConfirm.jsp **「確認」**ボタンによって呼び出される PlaceOrder サーブレットを作成します。
    - サーブレットは、iSeries プログラム呼び出しウィザードによって生成される Bean を使用し、iSeries サーバーにアクセスして RETAILSTOR ライブラリーの ORDERS データベースで注文を発行します。
    - 注文は買い物カゴ中の項目であり、ILE プログラムに配列の構成として送信されます。
    - この配列の各構成は、買い物カゴ中の項目です。
    - Bean によって呼び出される PLACEORD RPG サービス・プログラムは、注文番号を出力パラメーターとして戻し、セッションに入れます。

    以下のコード・セグメントは、PlaceOrder サーブレットによる PLACEORD Bean の使用方法を示します。

```
.
 .
 .
public void init() throws ServletException {
 hostInfo = GetItems.getHostInfo();
 super.init();

 try {
  /* creating an instance of the PLACEORD bean created
     by iSeries Program Call Bean wizard */
  orderBean = new PLACEORD();
  orderBean.setConnectionData(hostInfo[0],
  hostInfo[1], hostInfo[2]);
```

```
 } catch (Exception e) {
  e.printStackTrace();
 }
}

.
.
.

public void doPost(HttpServletRequest request,
HttpServletResponse response) {

 ...

 PLACEORD.Orditems_Struct inputStruct = null;

 // retrieving the order items from the basket
 Basket basket =
 (Basket) request.getSession().getAttribute("basket");
 AddtoBasket.MyRecordIOManager recIO =
 (AddtoBasket.MyRecordIOManager)
   request.getSession().getAttribute("recIO");
 if (basket == null || basket.size() == 0
 || recIO == null) {
  try {
   response.sendRedirect("errorPage.html");
  } catch (IOException e) {
   e.printStackTrace();
  }
 } else {
  items = basket.elements();
  // setting array of structure elements
  while (items.hasMoreElements()) {
   item = (String[]) items.nextElement();
   inputStruct = orderBean.getOrdItemAr(j);
   inputStruct.setItemNo(new BigDecimal(item[0]));
   inputStruct.setQuantity(new BigDecimal(item[1]));
   inputStruct.setSizeOrd(item[2]);
   j = j + 1;
  }
  // setting the rest of the array elements to dummy values
  for (int i = j - 1; i < 100; i++) {
   inputStruct = orderBean.getOrdItemAr(i);
   inputStruct.setItemNo(new BigDecimal(0));
   inputStruct.setQuantity(new BigDecimal(0));
   inputStruct.setSizeOrd("s");
  }

  // setting the other two input parameters of the bean
  orderBean.setNumOfItems(new BigDecimal(j));
  orderBean.setBalance((BigDecimal)
  request.getSession().getAttribute("balance"));
  try {
   // invoking the iSeries program
   orderBean.invoke();

   // retrieving the order number from PLACEORD bean
   orderNumber = (orderBean.getRetCode()).toString();
   request.getSession().setAttribute("orderNumber",
   orderNumber);
   basket.empty();
   // commit this order now
   recIO.commit();
   response.sendRedirect("orderNumber.jsp");
   return;
  } catch (Exception e) {
   response.sendRedirect("errorPage.html");
```

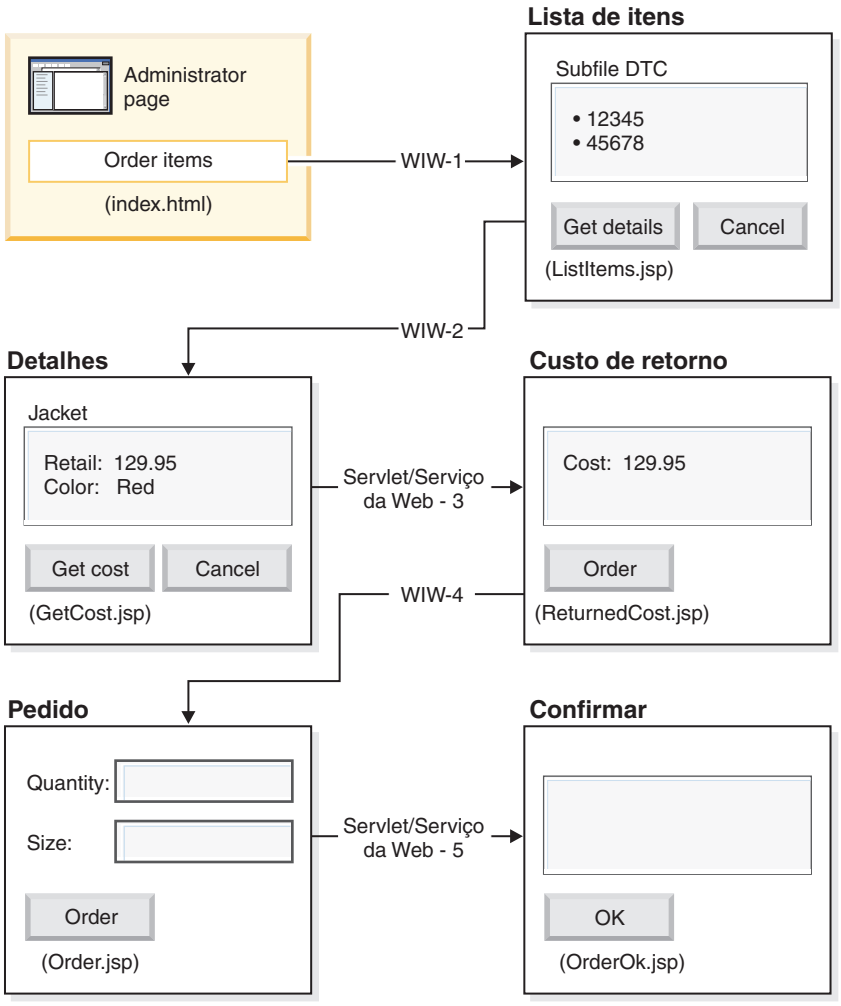
```
            e.printStackTrace();
          }

        }
      }
```

13. 番号を検索して、それを確認メッセージと共にお客様に表示する OrderNumber サーブレットを作成します。お客様が買い物カゴに項目を入れていない場合は、代わりにエラー・ページが戻されるようにしてください。

第 7 章 拡張モジュール 2: SV000514 および SV001586 Web サービスを使用する Web プロジェクトの作成 (SV000618)

概要

このプロジェクトは、RPG プログラミング知識を使用して iSeries Web サービスおよび RPG プログラムの Web クライアントを作成する方法について説明しています。このプロジェクトでユーザーは、管理者の役割を演じて Web ページのシリーズを段階的に進み、在庫数量を判別して、卸売業者から小売店用に追加の在庫を注文します。項目番号を入力し、項目の詳細を表示し、数量とサイズを注文して、確認を受け入れます。



(WIW = Web Interaction Wizard)

これは拡張モジュールであるため、指示はプロジェクト作成の各ステップには進みませんが、そのようなプロジェクトを作成するために実行される展開ステップについて概説します。このプロジェクトは Development Studio Client の以下のコンポーネントを使用します。

- Web 対話ウィザードおよび各種 Web コンポーネントからの出力を組み込む Page Designer を使用して Web ページを作成する iSeries Web 開発ツール
- 項目情報を戻す TNLSTITM RPG サービス・プログラムを作成する Remote System Explorer
- 項目の価格を検索して項目を注文するサーブレット・プロキシー・コードを生成する Web サービス・ウィザード
- 必要なサーブレットを作成する iSeries Java 開発ツール
- WebSphere Application Server を介して iSeries サーバーに対して配置する以前にアプリケーションを検査する WebSphere テスト環境

始める前に

この演習を完了できるのは、次の前提条件に適合する場合だけです。前提条件については、9 ページの『第 3 章 シナリオの実行』で詳しく説明されています。

- iSeries サーバーに TCP/IP でアクセスしている。
- iSeries サーバーをコマンド `STRTCPSVR *ALL` で開始した。
- WebFacing サーバーをコマンド `STRTCPSVR *WEBFACING` で開始した。
- WHOLESALE および RETAILSTOR ライブラリーを iSeries サーバーに復元した。
- WHOLESALE, RETAILSTOR および QGPL ライブラリーがユーザーのライブラリー・リストに入っている。
- 9 ページの『第 3 章 シナリオの実行』ですべてのタスク (ワークベンチでのアプリケーションのテストには不要なオプションの WebSphere Application Server タスクは除く) を完了した。

Web ページ、サーブレット、JSP、および RPG コードの作成

SV000618 のコンポーネントを構成するには、以下のようにしてください。

1. Web プロジェクトを作成して、作成するすべてのファイルを保持します。
2. iSeries 在庫データベースに項目をリストする ListItems JSP ファイルを書き込みます。iSeries Web 開発ツールの Page Designer を使用して、サーブレットを作成できます。つまり、デザイン・ビューでページを配列し、ソース・ビューで適切なコードを追加できます。また、サブファイルをデータベース・レコードで埋め込んで、TNLSTITM RPG サービス・プログラムと対話するためにサブファイルの Design Time Control (DTC) を挿入する必要もあります。DTC 制御設定にはサービス・プログラムを指定できます。

Web 対話ウィザードを使用して入力ページを作成する必要があります。

- `ListItems.jsp` を出力ページとして指定し、在庫項目をリストします。これは、Web 対話ウィザードによる `ListItems.wit` ファイルの作成を確実にします。

- サブファイル DTC は自動的に TNLSTITM RPG サービス・プログラムを起動するので、Web 対話ウィザードにプログラム呼び出しが指定されていないことを確認します。また、ウィザードは ListItemsWitServlet を生成します。これは ListItems.jsp ページを呼び出すリンクとして機能します。
- Web 対話ウィザードによって生成される ListItemsWit.wit ファイルを検討するには、次のようにしてください。
  1. **SV001618** を展開して **ListItems.wit** をダブルクリックし、ファイルの対話ウィザードを表示します。
  2. ウィザードで**「次へ」**をクリックして、対話に指定された値を検討します。

Page Designer を使用して GetCost JSP 出力ページを作成する必要があります。これは ListItems.jsp 入力ページから入力を実行します。ユーザーが ListItems.jsp ページの項目をクリックすると、GetCost.jsp ページには項目の詳細が表示されます。

GetCost.jsp ページを作成した後で、Web 対話ウィザードを使用して ListItems.jsp (入力ページとして選択) および GetCost.jsp (出力ページとして選択) 間の WitOrder 対話を作成します。

- ウィザードのプログラム呼び出しページで、TNLSTITM RPG サービス・プログラムから GetDetail プロシージャーおよびパラメーターの呼び出しを指定します。
- プロシージャーにおいて、選択されたサブファイル・レコードを判別するためにサブファイル DTC API が組み込まれます。プロシージャーは、この情報を使用して INVENTORY データベースから選択済みレコードを検索し、GetCost.jsp 上に選択した項目の詳細 (画像を含む) を表示します。
- Web 対話ウィザードによって生成される WitOrderWit.wit ファイルを検討するには、次のようにしてください。
  1. **SV001618** を展開して **WitOrder.wit** をダブルクリックし、ファイルの対話ウィザードを表示します。
  2. ウィザードで**「次へ」**をクリックして、対話に指定された値を検討します。

  flow パラメーターは出力ページでフロー・コントローラーとして指定されることに注意してください。これにより、パラメーターの値に適切な JSP ファイルが表示されるようになります。

「原価」ボタンを押すことで卸売業者からの項目の現在の原価を管理者が検索できるように、Web サービス定義言語 (WSDL) ファイルをプロジェクト SV000514 からインポートする必要があります。

- **「原価」**ボタンにより、QryProdCostServlet.jsp および対応する Web サービスがプロジェクト SV000514 から呼び出されます。
- Web サービス・ウィザードおよびインポートした WSDL ファイルを使用して、Web サービスを呼び出すために必要な Java プロキシー・コードを生成します。
- QryProdCostServlet.jsp は、GetCost.jsp ページから入力を取り出し、選択済み項目の原価を検出する SV000514 Web サービスを呼び出すために Java プロキシー・コードを使用して、ReturnedCost.jsp と呼ばれるページに原価を表示します。
- QryProdCostServicesProxy.java コードおよび QryProdCostServlet.java を表示するには、次のようにしてください。

1. **「SV001618」>「ソース」>「プロキシー」>「SOAP」** を展開します。
2. **QryProdCostServicesProxy.java** をダブルクリックします。
3. QryProdCostServlet.java の場合は、**「SV001618」>「ソース」** の下で **QryProdCostServlet.java** をダブルクリックして、Java プロキシー・コードのインスタンス化方法を注記します。

管理者が**「注文」**ボタンをクリックして選択項目を卸売業者に注文できるよう、「Web 対話」ウィザードで ReturnCost.jsp を入力として、また Order.jsp を出力としてリンクさせる必要があります。

- この対話では、2 ページのリンクが正しい情報の表示に十分であるときには、プログラム呼び出しを使用する必要がありません。
- WitPlaceOrder.wit を表示するには、次のようにしてください。
  1. **SV001618** を展開します。
  2. **WitPlaceOrder.wit** をダブルクリックして、対話を開きます。
  3. ウィザードで**「次へ」**をクリックして、指定値を検討します。

管理者が注文した項目のサイズと数量を指定できるように、SV001586 Web サービスを使用します。

- SV001586 WSDL ファイルをこのプロジェクトにインポートし、Java プロキシー・コードを生成して Web サービスを呼び出し、ユーザーが Order.jsp から**「注文」**ボタンを押すと呼び出される OrderSupplyServlet を作成します。
- サーブレットは、Order.jsp から情報を収集して Web サービス Java プロキシー・コードを呼び出します。これは SV001586 Web サービスを呼び出して項目を注文します。
- サーブレットは、注文が正常に実行されると OrderOK.jsp を表示し、正常に実行されないとエラー・ページを表示します。
- 生成された Web サービス・プロキシーがインスタンス化されて、SV001586 Web サービスを使用するために呼び出される方法を表示するには、次のようにしてください。
  1. **「SV000618」>「ソース」** を展開します。
  2. **OrderSupplyServlet.java** をダブルクリックして、内容を調べます。

WebSphere Application Server に展開する前に

アプリケーションを WebSphere Application Server に展開する前に、アプリケーションが適切に機能するように SV000514 および SV001586 Java クラス・ファイル内部の特定 URL を変更する必要があります。

SV000514 の場合:

- Web サービス・プロキシー・クラス QryProdCostServicesProxy において、呼び出す Web サービスの URL が入っている変数が定義されます。
- 最初にプロキシーを作成するときには、この URL は http://localhost:9080/SV000514/servlet/rpcrouter に設定されています。
- この値に設定されている変数を使用して、IDE 中のプロジェクト SV000514 に入っている Web サービスが呼び出されます。これは、19 ページの『ワークベンチでのアプリケーションの実行』で説明されています。

- このアプリケーションを展開する前に、EAR ファイル SVWholeSale.ear を iSeries IFS ディレクトリー上で展開した場所を示すように、この URL 値を変更する必要があります。

プロジェクト SV001586 の場合:

- Web サービス・プロキシー・クラス `OrderSupplyServicesProxy` において、呼び出す Web サービスの URL が入っている変数が定義されます。
- 最初にプロキシーを作成するときには、この URL は http://localhost:9080/SV001586/servlet/rpcrouter に設定されています。
- この値に設定されている変数を使用して、IDE 中のプロジェクト SV001586 に入っている Web サービスが呼び出されます。これは、19 ページの『ワークベンチでのアプリケーションの実行』で説明されています。
- このアプリケーションを展開する前に、SVWholeSaleEAR.ear ファイルを iSeries IFS ディレクトリー上で展開した場所を示すように、この URL 値を変更する必要があります。

第 8 章 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あり、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。

使用許諾については、下記の宛先に書面にてご照会ください。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
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または保証条件は適用されないものとします。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。

Lab Director IBM Canada Ltd. Laboratory 8200 Warden Avenue Markham, Ontario, Canada L6G 1C7

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。

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。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。お客様は、IBM のアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、 IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

(C) (お客様の会社名) (年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 (C) Copyright IBM Corp. 1992, 2002. All rights reserved.

プログラミング・インターフェース情報

プログラミング・インターフェース情報は、プログラムを使用してアプリケーション・ソフトウェアを作成する際に役立ちます。

一般使用プログラミング・インターフェースにより、お客様はこのプログラム・ ツール・サービスを含むアプリケーション・ソフトウェアを書くことができます。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

警告: 診断、修正、調整情報は、変更される場合がありますので、プログラミング・インターフェースとしては使用しないでください。

商標

- 400
- AFP
- AIX
- AIX windows
- APPN
- Application System/400
- AS/400
- AS/400e
- BookManager
- C Set++
- C/400
- CICS
- CICS/400
- CICS/ESA
- COBOL/2
- COBOL/400
- Common User Access
- CUA
- DB2
- DB2 Extenders
- DB2 Universal Database
- Domino
- 
- GDDM
- IBM
- IBMLink
- Integrated Language Environment
- iSeries
- Language Environment
- Lotus
- Lotus Notes
- MQSeries
- Network Station
- Open Class
- Operating System/2
- Operating System/400
- OS/2
- OS/390
- OS/400
- POWER2
- PowerPC
- PROFS
- RPG/400
- RS/6000
- S/390
- SQL/400
- System/36
- System/38
- VisualAge
- VTAM
- WebSphere

InstallShield は InstallShield Corporation の商標です。

Intel および Pentium は、Intel Corporation の米国およびその他の国における商標です。

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または登録商標です。

Rational は、IBM Corporation および Rational Software Corporation の米国およびその他の国における商標です。

Lotus、Lotus Notes、および Domino は Lotus Development Corporation の米国およびその他の国における商標です。

ActiveX、 Microsoft、 SourceSafe、 Visual C++、 Visual SourceSafe、 Windows、 Windows NT、 Win32、 Win32s、および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

Netscape Navigator は Netscape Communications Corporation の商標です。

UNIX は、The Open Group の登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

プログラム番号: 5724-A81

Printed in Japan

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