WebWorkBench DeveloperCafe

導入ガイド

サイオステクノロジー株式会社

WebWorkBench

DeveloperCafe

第1章はじめに 1
概要 2
コンセプト 3
製品構成 4
特徴 5
機能 6
DeveloperCafeによるEJB利用モデル 8
導入のメリット 9
動作環境 10
第2章はじめてのDeveloperCafe 11
Webアプリケーションの構築 12
DeveloperCafeの仕組み 13
DeveloperCafeによる開発スタイル 14
DeveloperCafeのコンポーネント 15
セッション管理 15
データの取得、更新 17
画面の表示と遷移 21

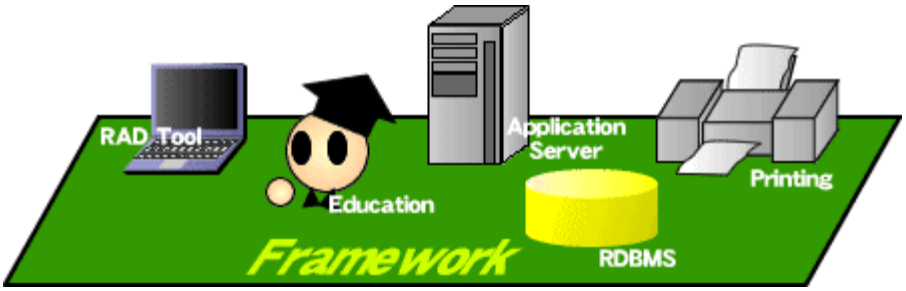
Copyright(C) 1997-2007 by SIOS Technology, Inc. All Rights Reserved.

第１章はじめに

概要

Web システム開発の現状と課題

C/S システムの衰退に伴い、企業情報システムでは、Web システムへの移行が急速に進んでいます。この背景には、インターネット／イントラネットの活用が一般化し、インフラが整備されたことが挙げられます。Web システムの構築においては、従来の開発手法とは異なるオブジェクト指向を取り入れており、メンテンナンス性、再利用性を高めるために、その特性を理解し、また、留意した開発が必要不可欠です。



フレームワークとは・・・

アプリケーションサーバさえあれば、簡単に Web システム構築を構築できるという見方が圧倒的ですが、これは誤りです。Web システムを構築するための最も重要なポイントが全く考慮されておりません。Web システム構築のポイントは、開発を行うプロジェクトメンバーすべてが、共通に作業できるソフトウェア環境、つまり、「Web システム構築の作業台＝フレームワーク」が、極めて重要なのです。

DeveloperCafe とは・・・

DeveloperCafe は、C/S システムから Web システムへのマイグレーションを短期間で、しかも、効率良く実現できるフレームワークです。また、従来の手法（ActiveServerPage, CGI）からの Web システムへの移行も容易に行えます。フレームワークの導入により、プログラミング手法が標準化され、コンポーネントの再利用も促進されることから、短期間で効率の良い開発が行うことができるのです。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

コンセプト

DeveloperCafe とは...

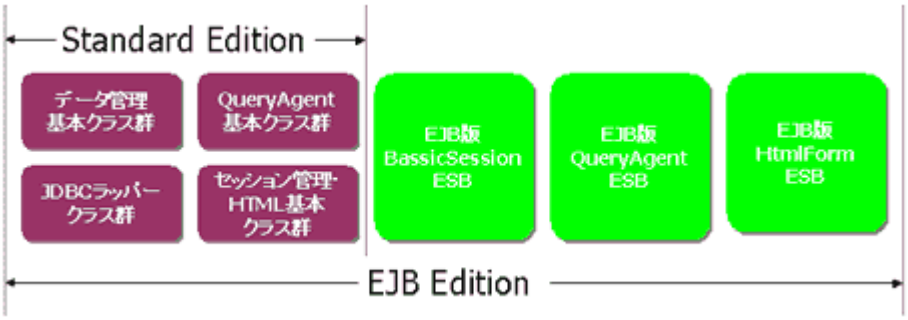
J2EE 標準アプリケーションモデル、および、MVC モデル準拠の Web アプリケーション構築フレームワーク「DeveloperCafe」は、Web アプリケーション開発の生産性向上と品質の均質化を目的としたフレームワークです。

DeveloperCafe で変わる開発スタイル

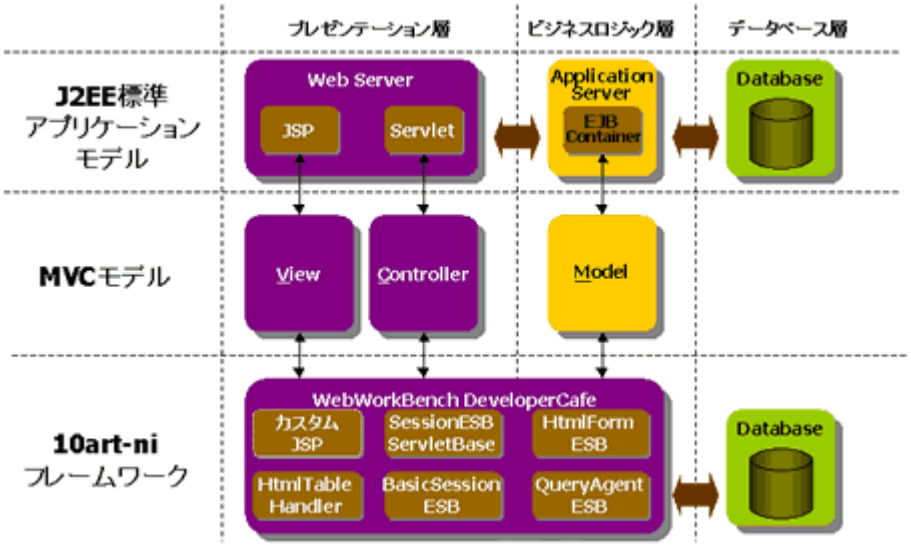
DeveloperCafe が提供するコンポーネントは、アプリケーションに依存しない完全な汎用サービスオブジェクトです。このサービスを利用することで MVC モデルの基本構造は既に実現されていることになります。また、アプリケーション開発者は、EJB に関する設計・実装に関する知識がなくても、その機能を利用することが可能となります。詳細は、「White Paper」にて解説しておりますので、こちらを参照ください。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

製品構成



DeveloperCafe は、Web システム構築に必須のコンポーネントをすべて網羅した J2EE アプリケーションモデル・MVC モデル準拠のアプリーション構築フレームワークです。



Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

特徴

EJB を意識せずに開発できる DeveloperCafe

Web システム構築にあたって必須となるデータベース・アクセスやセッション管理などの機能をコディンング不要とし、開発生産性を向上する、J2EE・MVC モデル準拠の Web システム構築フレームワークです。JDBC、Servlet などの低水準 API を隠蔽し、高水準 API として提供することで、開発工数を削減できると同時に、実装方式を統一できるため、品質のばらつきを解消し、ソースコードの可読性向上も実現できます。また、EJB Edition では、EJB 実装を意識することなくアプリケーション層のプログラミングで EJB の特長を生かしたシステム構築がおこなえます。

EJB 対応による短期間で高品質なシステム構築

データベース・アクセス、セッション管理といった Web システム開発に必須の機能をすべて網羅したプリミティブなフレームワークなので、ビジネスロジックの開発のみに注力でき、開発期間の短縮とシステムの品質を向上することができます。また、業務に特化していないので、ベースとなるフレームワークとして汎用に活用できると同時に、専用の EJB コンポーネントを開発することで業務に特化したフレームワークとしても利用可能です。

開発したソフトウェアの再利用性を向上

エンタープライズ環境で要求される大規模な Web システムの開発にも最適なフレームワークとして利用できます。また、複数のプロジェクトによる開発を行った場合でも、フレームワークに沿った開発を行うことで、プログラミング手法を標準化できることから、コンポーネントの再利用が可能となり、長期的な開発効率の向上を実現できます。

主要アプリケーション・サーバをサポート

EJB の実行環境として主要なアプリケーション・サーバ上での動作を検証しており、柔軟なプラットフォーム選択を行えます。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

機能

DB トランザクションサポート機能

✓ **JDBC コーディングレスの SQL コンポーネント**
WDC の SQL コンポーネント群は JDBCAPI をカプセル化し、大幅な生産性向上、初歩的なミス防止、そして安定した品質を実現します。また、DB コネクションは J2EE 標準データソース、独自のコネクションプールを選択可能です。DB コネクションの利用と解放が完全に自動化されています。

✓ **項目名による SQL パラメータアクセス**
各 SQL コンポーネントでは SQL パラメータ記述とそのアクセスに項目名を使用することができます。"?"文字を使用する JDBCAPI より、はるかに安全で可読性の高いプログラミングを実現します。

✓ **キャッシュと更新を自動化する *SQLRowSet***
SQLRowSet は JDBC 結果セットを自動的にキャッシュします。DB 更新もわずかなプロパティ設定で自動実行可能です。更新 SQL 文記述の必要がありません。

✓ **DB トランザクション層をカプセル化する *QueryAgent***
QueryAgent はアプリケーションフロント層と DB トランザクション層を分離する DB トランザクション実行アーキテクチャです。これにより大規模システムの SQL 実装拡散を防止し、柔軟な拡張性と優れた保守性を提供します。

✓ **コミットとロールバックのカプセル化**
QueryAgent はコミット／ロールバックのディフォルト動作を提供します。この上に更新処理を実装する為、堅牢なトランザクション制御を簡単に実現することができます。

データキャッシュ／入出力サポート機能

✓ **洗練されたデータキャッシュコンポーネント**
RowSet、*ParamSet* は統一／洗練された API を持つ汎用性の高いデータキャッシュコンポーネントです。これらはデータ項目を名前とデータ型で管理し、複雑なビジネスロジックを可読性の高いシンプルなコードで表現します。

✓ **RDB データキャッシュに適合する *RowSet***
RowSet データには行位置移動、位置指定など多彩なメソッドでアクセス可能です。行の追加、削除、変更も自由に行うことができます。また RDB との入出力だけでなく CSV ファイルとの入出力も可能です。

✓ ***RowSet* のソート／フィルター機能**
RowSet のデータは任意項目値でソートすることが可能です。ソート項目数は無制限、昇順、降順の指定も可能です。また、項目値でのフィルタリングも可能です。キャッシュ内容を破壊することなく必要な情報だけを取り出す操作が可能です。

✓ **JDBC メタ情報による入力値検証自動化**
RowSet はデータ項目値の単体チェックを自動化します。JDBC メタ情報をもとに文字/数値/日付などの基本的なデータ型、文字列長、数値桁数/精度を自動的に検証します。

✓ **入力値チェックオプション**
RowSet、*ParamSet* の文字列型データには”英数シングルバイトのみ”、”ダブルバイトのみ”、”カタカナを含むシングルバイトのみ”等の入力値検証オプションが設定可能です。また、ユーザー定義検証オプションを追加設定することも可能です。

✓ **数値/日付のフォーマット**
RowSet、*ParamSet* の数値型/日付型データ項目には文字列化する際の書式設定することが可能です。また和暦編集にも対応しています。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

J2EE アプリケーションサポート機能

✓ シンプルで柔軟な画面遷移メカニズム
MVC モデルをベースとしたセッション管理と画面遷移メカニズムを提供します。セッション管理に必要な制御は完全に自動化されている為、画面遷移に必要なわずかなコードを記述するだけで Web アプリケーション基本動作が実現します。また、これらは VisualBasic 等のプログラミングに慣れた開発者に理解しやすいものとなっています。

✓ HTML フォーム送信の直列性チェック
ブラウザバックボタンによる HTML フォーム再送信はセッションの状態を破壊し、予想しない障害を引き起こします。WDC はこのエラーを検出すると同時に、本来送信すべき HTML フォームを再表示させることができます。

✓ HTTP リクエストによるデータ自動更新
HTML フォームから送信されたリクエスト内容を解析し、キャッシュデータを更新する手続きが自動化されています。これにより面倒なコーディングをすることなく画面入力処理を実装することが可能です。

✓ 統一されたインターフェースによる JSP コーディング
JSP からの JavaBean アクセス方法が統一されています。JSP は常に共有インターフェースを介して必要なデータを受け取る仕組みになっている為、JSP 毎に必要な Bean やアクセスメソッドを設計実装する工数を削減することが可能です。

✓ DB データをもとにした HTML フィールド／テーブル出力
HtmlTableHandler はデータキャッシュコンポーネントの内容をもとに、HTML フォームの INPUT TEXT や INPUT SELECT に必要なタグを自動生成します。これにより一覧形式テーブルや見出し/データが整然と配置されたテーブルなど一般的なレイアウトを簡単に作成することが可能な為、JSP コーディングもシンプルなものとなります。

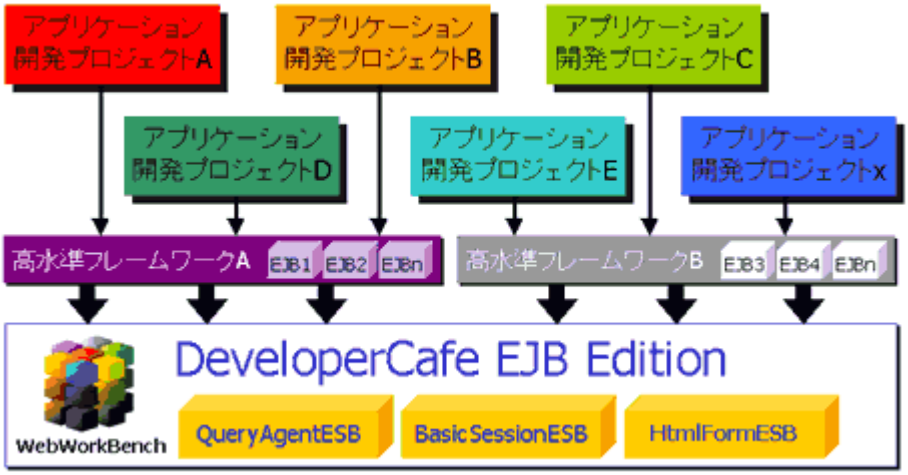
✓ 一覧表示データのスクロール機能
HtmlTableHandler が出力する一覧形式テーブルはキャッシュの一部を表示することが可能です。この表示内容のスクロール制御もメソッドひとつで簡単に実現できます。

✓ DB データをもとにした HTML 選択リスト／ラジオボタンの構築
選択フィールド(INPUT SELECT,INPUT RADIO)をデータベース問合せ結果から構築することができます。問合せ結果をロードする RowSet を準備し、これを選択フィールドへ関連付けるという簡単な手順で、選択フィールドとその選択枝に必要な HTML タグが自動出力されます。

✓ EJB による WDC サービスの実現
WDC のセッション管理サービスと *QueryAgent* サービスをセッション Bean として EJB コンテナ内で走行させることが可能です。この際、ユーザー定義クラスは 100%修正することなく EJB コンテナ内へ移行することが可能です。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

DeveloperCafe による EJB 利用モデル



DeveloperCafe EJB コンポーネントを利用すると・・・

アプリケーション層の開発のみに専念できます
各プロジェクトで開発ノウハウが共有され、生産性向上を実現できます
EJB のメリットのみを享受できます

DeveloperCafe EJB コンポーネントを利用し、業務や顧客に特化した高水準フレーワークを開発すると・・・

アプリケーション開発のスピードは飛躍的にアップします

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

導入のメリット

- プログラマのスキル及び創意工夫による品質のばらつきが解消され、全体の品質が向上
- プロジェクト開発メンバー全員が共通となる基盤の上で開発する為、ノウハウ共有が促進
- 作成したアプリケーション（Java Source Code）の可読性が向上
- プロジェクトの生産性向上及び開発工数（QA コスト）の削減
- 開発メンバーのスタディー期間を短縮し、試行錯誤に費やす時間を削減

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

動作環境

OS	以下 JDK が動作する OS
JDK	JDK1.3.1, JDK 1.4.x, JDK 1.5.0
Servlet/JSP	Servlet API 2.2/2.3/2.4, Java Server Pages 1.1/1.2/2.0
DBMS	http://www.sios.com/product/WebWorkBench/index.html を参照して下さい
EJB Edition 動作検証済み アプリケーションサーバー	弊社までお問い合わせください。 お問い合わせ先: wwbench@sios.com

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

第２章はじめての DeveloperCafe

この章は、はじめて DeveloperCafe を利用して Web アプリケーションを構築する
開発者のための手引書です。
前提知識として、【HTML】、【SQL】、【Java（JDBC,Servlet,JSP）】の基礎が必要です。
この章では、DeveloperCafe を利用する上で頻繁に使われるクラス及びメソッドのみを紹介します。
他のクラス、メソッドに関する詳細は、[開発者ガイド: コンポーネント解説]及び JavaDoc をご覧ください。

Web アプリケーションの構築

Web アプリケーションを構築するには、どんな技術、知識が必要でしょうか？

例としてオンラインショッピングサイトの構築を考えてみましょう。
オンラインショッピングの利用者は、ブラウザを用いて以下の操作を行います。

1. 店に入る
2. 品物を見る
3. 気に入った商品を買い物カゴに入れる。
4. 購入する。
5. 店を出る。

これらを実現するには、以下の処理をサーバ側で行う必要があります。

1. ログイン（ユーザー認証）する。
2. データベースから商品情報を取得し、表示する。
3. 購入予定の商品情報を登録する。
4. データベースを更新する。
5. ログアウトする。

上記で使われる技術をまとめると、以下の3つに集約されます。

- セッション管理
- データベースからのデータ取得と、更新
- 画面の表示と遷移

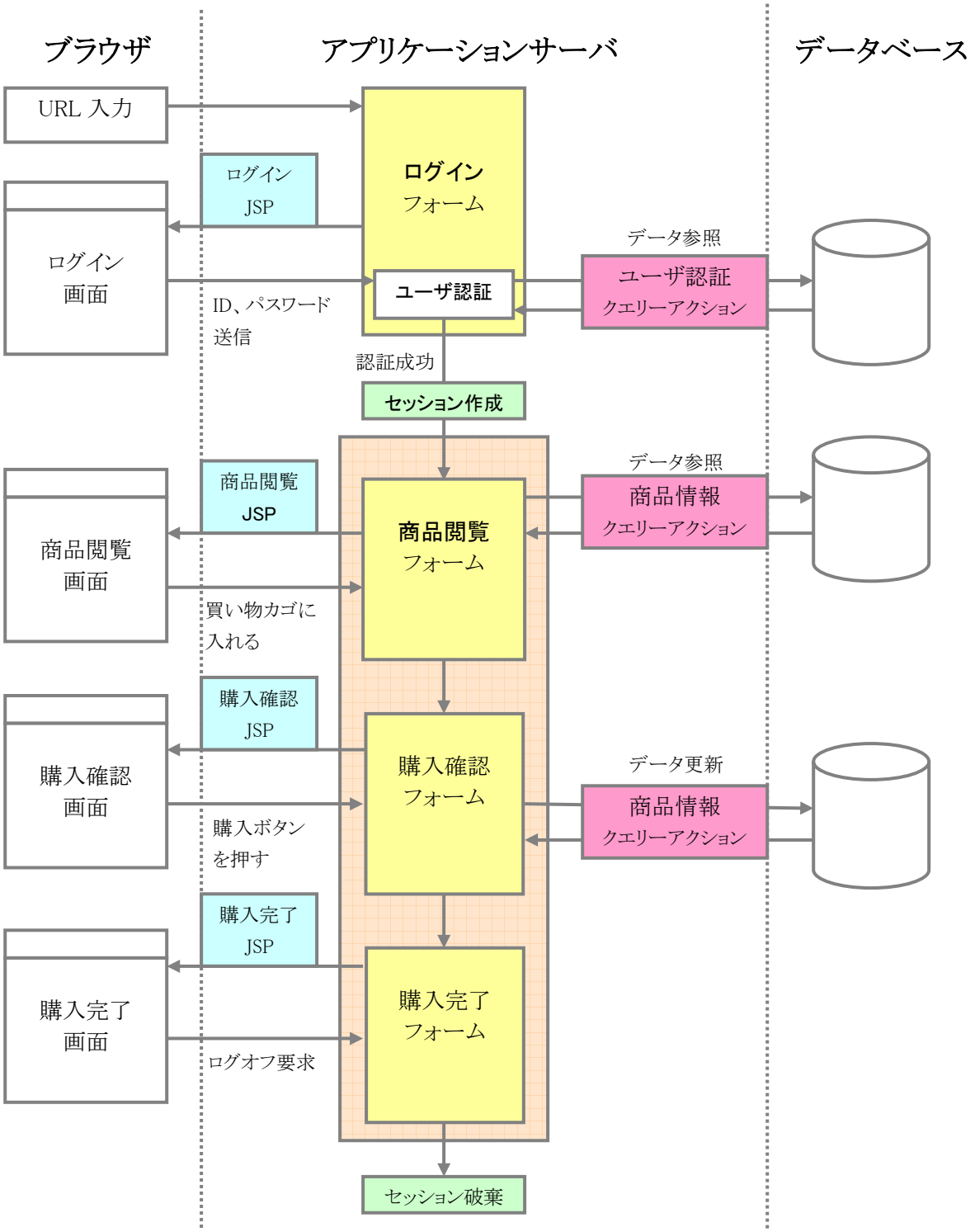
これらの技術は、オンラインショッピングに限らず、Web アプリケーションの構築に毎回必要な技術です。Java でこれらを実現するには、Servlet,JDBC の API を利用します。しかし、Servlet,JDBC は、低レベルな API であるため、上記のようなオンラインショッピングサイトを構築するには非常に多くのコーディングが必要になります。

DeveloperCafe は、これらの低レベル API を隠蔽し、容易に利用できる高レベル API のコンポーネント群を提供しています。これらコンポーネント群を利用することで、少ないコーディング量で、短期間に、なおかつ高品質な Web アプリケーションを構築することが可能になります。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

DeveloperCafe の仕組み

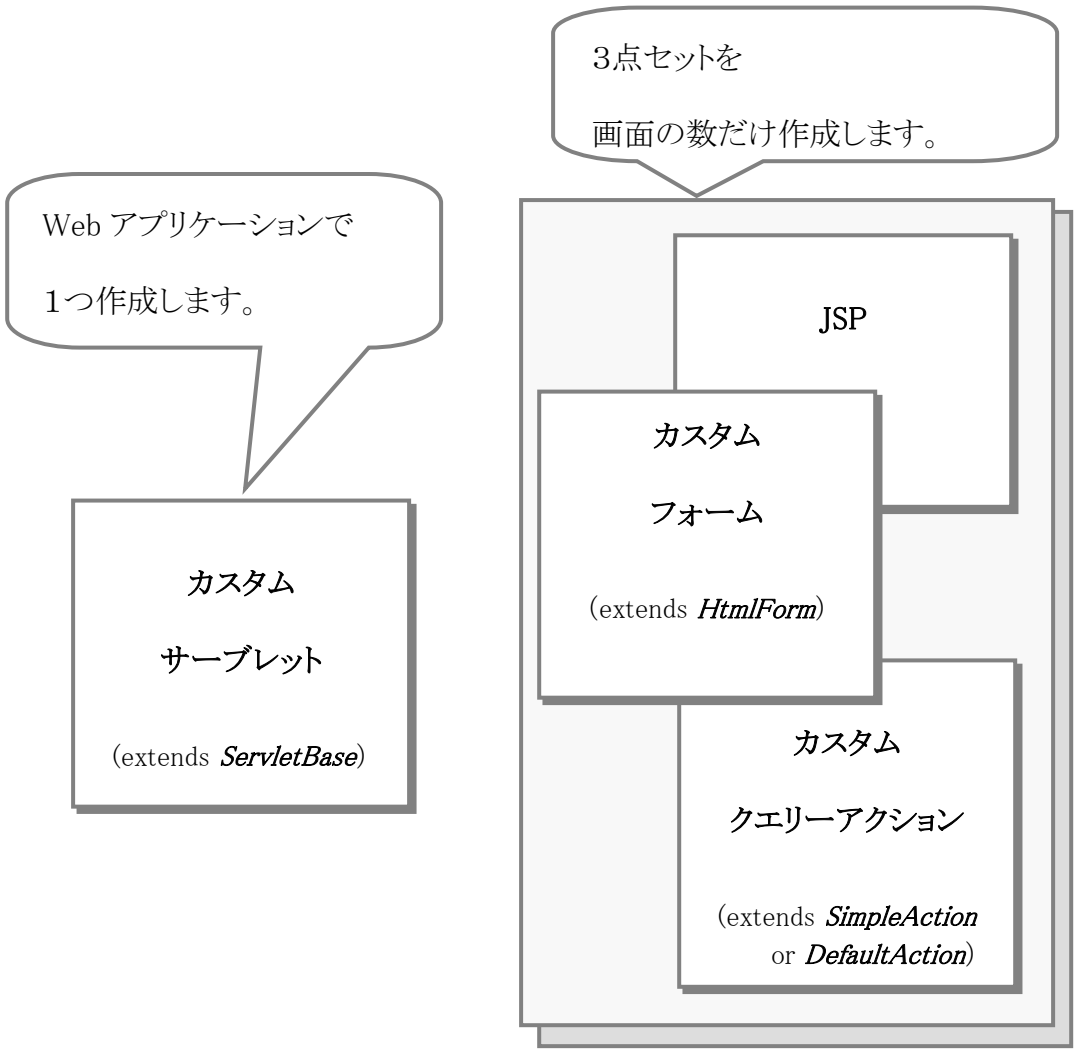
前述のオンラインショッピングサイトを DeveloperCafe で構築した場合、以下の構造になります。
DeveloperCafe では、フォームクラスが、画面遷移や JSP を用いた画面表示のコントロールを行い、クエリーアクションクラスが、データベースへの問合せや更新処理を行います。



Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

DeveloperCafe による開発スタイル

開発者が作成すべきクラスは以下の4種類です。



ほとんどの開発者は、「カスタムフォーム」、「カスタムクエリーアクション」、「JSP」の3点セットのみを画面数に応じて作成します。開発者は、これらのクラスの各メソッドにそこで行うべきロジックを記述するだけです。

カスタムサーブレットは、Web アプリケーションで1つ必要です。よって、開発者全員がカスタムサーブレットの作成に詳しい必要はありません。

なお、上記のように、基本的には3点セットを画面の数だけ作成しますが、場合によっては、JSP、カスタムクエリーアクションを、別のカスタムフォームで使いまわすこともあります。

以下でこれらのクラスを構成する DeveloperCafe のコンポーネントについて解説します。

※各コンポーネントの詳細については、[開発者ガイド: コンポーネント解説]を参照して下さい。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

DeveloperCafe のコンポーネント

セッション管理

パッケージ名　*jp.co.tenartni.xsv.**

■ セッション管理とは…

セッション管理とは、サーバがクライアントを一意に識別して、一連のやりとりが行えるように管理する仕組みをいいます。では、Web アプリケーション構築に、なぜこの仕組みが必要なのでしょうか？

先ほどのオンラインショッピングに話を戻します。もしセッション管理の仕組みがなければ、以下の事態が発生します。

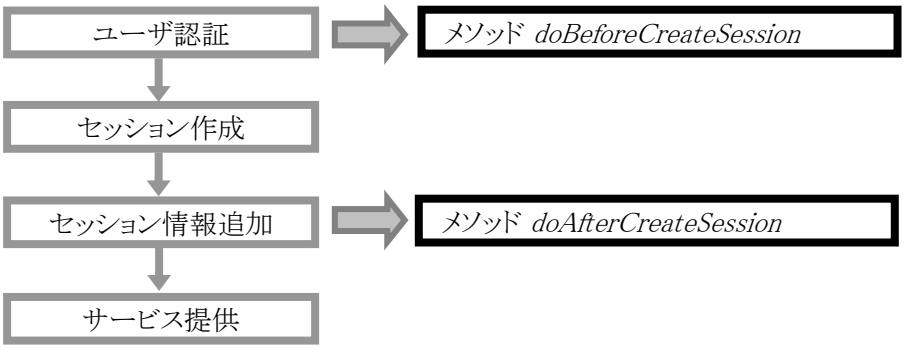
- 前の画面でログインしたユーザが誰かわからない。
- 前の画面でどの商品を選択したかわからない。
- ログインしていないユーザでもサービスの画面にアクセスできてしまう。

HTTP は、1回目にアクセスした利用者と、2回目にアクセスした利用者を識別できないため、上記の現象が発生してしまいます。そのため、Web アプリケーションで利用者を識別する仕組み、つまりセッション管理を行う必要があるのです。

■ DeveloperCafe のセッション管理

オンラインショッピングをする際、利用者はまずログイン画面でユーザ ID とパスワードを入力します。サーバはその入力情報からユーザ認証を行い、成功すればセッションを作成します。作成後、セッションに情報（利用者の ID、名前）を追加し、サービスが利用できるようになります。セッション情報は、利用者がサービスを利用している間、または、ある一定時間の間、サーバ側で保持します。

DeveloperCafe は、上記におけるサービス提供までの処理を2つのメソッド（*doBeforeCreateSession*、*doAfterCreateSession*）に記述することで実現します。
セッションは、メソッド *doBeforeCreateSession* でユーザ認証が成功し、戻り値 true を返した際、自動的に作成されます。なお、これらのメソッドは、*ServletBase* のサブクラスであるカスタムサーブレットクラスに記述します。



Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

よって、メソッド *doBeforeCreateSession* では、ユーザ認証の処理を記述し、戻り値を true にします。メソッド *doAfterCreateSession* では、セッション情報追加の処理を記述します。

なお、セッション情報の追加は、メソッド *doAfterCreateSession* 以外でも行うことが出来ます。必要に応じてセッション情報の追加、取得を繰り返します。

DeveloperCafe のセッション管理は、以下の機能を提供します。

1. セッション情報の追加
2. セッション情報の取得
3. セッションの有効時間の設定
4. セッションの有効時間を調べる間隔の設定
5. 有効時間が過ぎたセッションの切断処理
6. セッション数の取得（※ HTTPSession 利用時は不可）

・ 3～6 については、サーブレット初期化パラメータに情報を記述します。詳しくは、開発者ガイド、JavaDoc をご覧ください。
・ ※ DeveloperCafe では、HTTPSession を利用することも可能です。利用するには、サーブレット初期化パラメータに情報を記述する必要があります。

■ サンプルコード

セッションの維持、破棄などの管理は DeveloperCafe が行うため、開発者はセッション管理を意識する必要はありません。よって、開発者はセッション情報の追加と取得の方法を理解すればよいことになります。

◆ セッション情報の追加（利用者の ID,名前を追加）

```
session.setParam("USER_ID"  ,Type.STRING ,"1234567890");
session.setParam("USER_NAME",Type.STRING ,"テンアートニ");
```

◆ セッション情報の取得（利用者の ID,名前を取得）

```
getSessionParam().getString("USER_ID");
getSessionParam().getString("USER_NAME");
```

◆ JSP 内でのセッション情報の取得（利用者の ID、名前を取得）

```
<jsp:useBean id="Session" class="jp.co.tenartni.xsv.BasicSessionClient" scope="request"/>

<%=Session.getParam("USER_ID")%>
<%=Session.getParam("USER_NAME")%>
```


Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

データの取得、更新

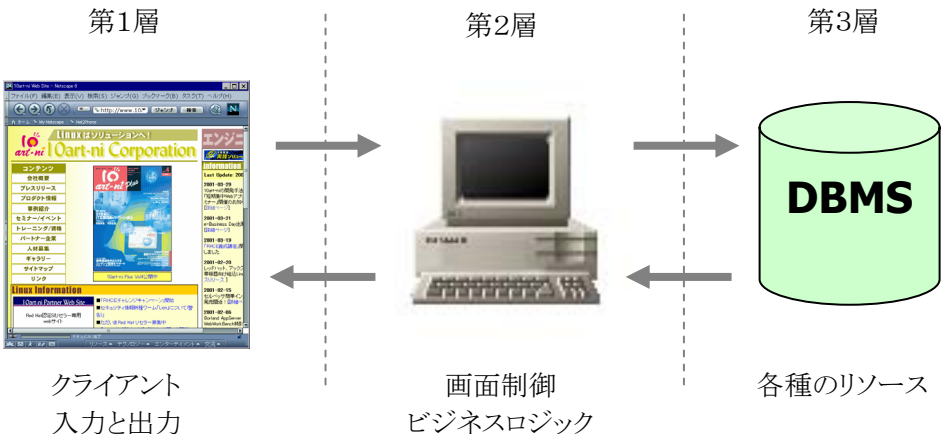
パッケージ名　*jp.co.tenartni.xqa.**

■ データの取得、更新

前述のオンラインショッピングでログインに成功すると、利用者は商品を見ることが出来るようになります。では、この商品のデータはどこにあり、どのように取得し、管理しているのでしょうか？

一般的にこれらのデータはデータベースに格納されています。よって、利用者が要求するデータをデータベースから取得する仕組みが必要になります。

Web アプリケーションでは、利用者が直接データベースからデータを取得することはありません。下図のように、利用者の要求は一旦サーバが受け付け、サーバがデータベースから必要なデータを取得し、利用者に結果を返します。



つまり、利用者からの要求をサーバが受け付け、サーバがデータベースに接続し、データの取得、更新を行い、さらに取得したデータを管理する仕組みが必要になります。

Java には、JDBC という API があり、JDBC ドライバを経由してデータベースに接続し、データの取得、更新を行うことが出来ます。
しかし、低レベルな API であるため、開発者が以下の処理を行いたい場合、複雑なコーディングが必要になります。

- 取得したデータの管理、加工
- 更新 SQL（Insert,Update,Delete）の生成
- コネクションプーリング（データベース接続の使い回し）

データベースからの単純なデータ取得を記述する際にも、20～30 行ほどのコーディングが必要になります。このようなコーディングを毎回していたのでは、開発効率は上がりません。そこで DeveloperCafe は上記の要求をスマートに満たすコンポーネントを提供しています。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

■ DeveloperCafe のデータの取得、更新

DeveloperCafe は、データベースの接続、データ取得、更新の機能を強力にサポートしています。主に以下の3つのクラス(*ClientRowSet*、*QueryAgentManager*、*QueryAction*)を利用します。

★ *QueryAgentManager*(データベース管理)

- データベースからのデータの取得、更新を代行します。データベース接続を保持しており(コネクションプーリング)、クライアントからの要求に対し、それらを自動で割り当てて処理を行います。

- *QueryAgentManager* のインスタンス生成以外は、自動で上記の処理を行います。よって、開発者はこれらの仕組みを詳しく理解している必要はありません。

★ *QueryAction*(SELECT 文記述)

- *QueryAgentManager* に依頼するデータベースからのデータの取得、更新処理を実装する為の基底クラスです。

- 開発者は、*DefaultAction* (*QueryAction* を継承)または、*SimpleAction*(*DefaultAction* を継承)を継承したカスタムクエリーアクションクラスを作成し、そこに SELECT 文を記述して利用します。更新系 SQL(INSERT・UPDATE・DELETE 文)は、自動的に生成されるため、記述する必要はありません。

- 単純な問い合わせや更新処理には *SimpleAction* を利用します。また、複数のテーブルに対する更新や、高度なデータ操作が必要な場合は、*DefaultAction* を利用します。

★ *ClientRowSet*(データ管理)

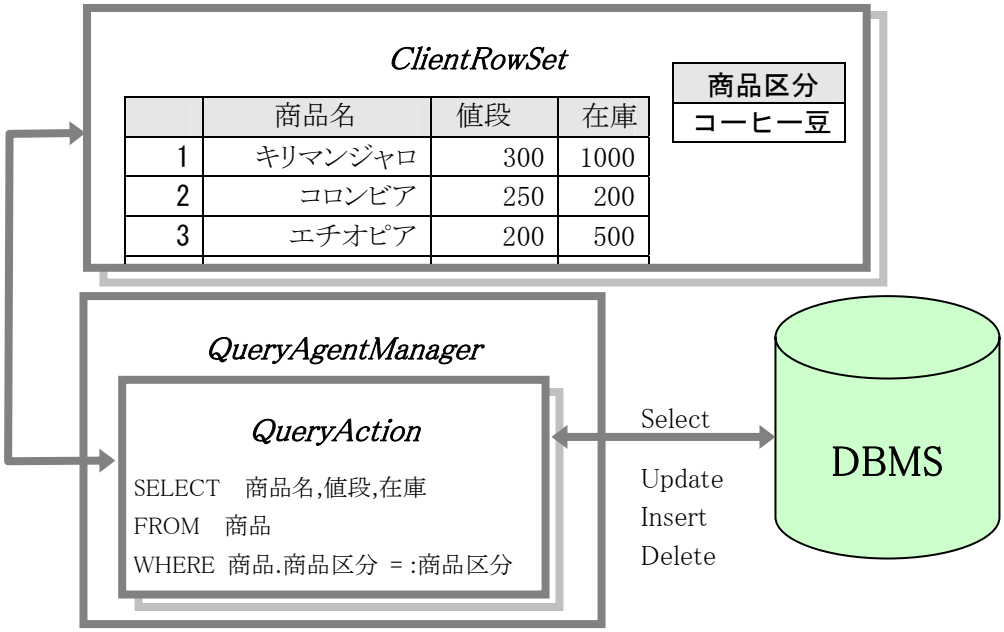
- データベースから取得したデータを行列データ(下図参照)として保持することが出来ます。取り扱うデータ型は文字、数値、日付等で、データベースのデータ型は自動的に Java のデータ型に変換されます。(*XRowSet* クラスの機能です。)

- 行列データ以外にも、データ項目:商品区分、値:コーヒー豆(下図参照)のように、行の概念が無い任意のデータ項目を定義し、値を保持することも出来ます。少量データの管理や、SQL パラメータとして利用することが出来ます。(*ParamSet* クラスの機能です。)

- データの追加、変更、削除ができます。*QueryAgent* は、更新データと SELECT 文を実行したときのデータを比較し、更新系 SQL を自動的に生成します。これにより、*ClientRowSet* に対する更新内容を、簡単にデータベースへ反映させることができます。

- *jp.co.tenartni.xsv.**HTMLTableHandler*** を用いることで、*ClientRowSet* のデータから HTML テーブルを作成し、ブラウザに表示することが簡単にできます。(*HTMLTableHandler* については、「画面の表示と遷移」で説明します。)

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.



■ サンプルコード

上図のように、商品区分（パラメータ）によって、取得データが異なる場合、これらのデータを取得するには、以下の処理を行います。

◆ *QueryAgentManager* に関する処理

・ *QueryAgentManager* のインスタンスを生成する。

実際には、*LocalConnection* を用いて以下のよう記述することで自分の VM 内にインスタンスを生成することが出来ます。

```
public LocalConnection conn = new LocalConnection();
clientRowSet.setClientConnection(conn);
```

◆ *QueryAction* に関する処理

・ *SimpleAction* を継承したクラスを作成し、上図の Select 文を記述する。

「where 商品.商品区分 = :商品区分」と記述することで、*ClientRowSet* のパラメータである「商品区分」をバインド変数として利用することが出来ます。

```
setSQL(
        " SELECT  商品名，値段，  在庫" +
        "    FROM  商品 " +
        "    WHERE 商品.商品区分 = :商品区分  ");
```

データを更新する場合は、更新対象のテーブル、更新時 SQL の WHERE 句に使用するカラム名（基本的には主キーのカラム名）を指定します。

```
setUpdateTableName("商品");
setRowIdColumnNames(new String[]{"商品区分","更新日","更新"});
```


Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

◆ ClientRowSet に関する処理

・ *ClientRowSet* のインスタンスを生成する。

```
ClientRowSet clientRowSet = new ClientRowSet();
```

・ パラメータ「商品区分」を追加する。

```
clientRowSet.getParamSet().addParam( “商品区分” , Type.STRING ,ParamType.IN );
```

・ パラメータ値「コーヒー豆」をセットする。

```
clientRowSet.getParamSet().setString( “商品区分”, “コーヒー豆”);
```

・ *QueryAction*(SimpleAction を継承したクラス)を設定する。

```
clientRowSet.setQueryActionName(クラス名);
```

上記処理を行った後、*ClientRowSet* の *setActive*()メソッドを呼び出し、引数に”true”をセットすることで、上図の行列データが *ClientRowSet* のインスタンスに格納されます。

```
clientRowSet.setActive(true);
```

なお、バインド変数を利用しない Select 文の結果を取得する場合、パラメータに関する処理は必要ありません。

データを更新する場合、*secureRow*()で更新結果を確定し、*resolveChanges*(false)でデータベースへ変更結果を適用します。引数は、コミットを保留するかどうかです。

```
clientRowSet.secureRow();
clientRowSet.resolveChanges(false);
```

clientRowSet に格納されたデータの操作は,以下のメソッドを利用します。

データの取得

```
clientRowSet.getString(“商品名”);
```

データの追加

```
clientRowSet.setString(“商品名”,”ブルーマウンテン”);
```

パラメータデータの取得

```
clientRowSet.getParamSet().getString( “商品区分” );
```

パラメータデータの変更

```
clientRowSet.getParamSet().setString( “商品区分” ,” コーヒーメーカー” );
```

行カーソルを移動するには、ClientRowSet にある以下のメソッドを利用します。

```
moveAt(int index) moveFirst() moveLast() moveNext() movePrior()
```


Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

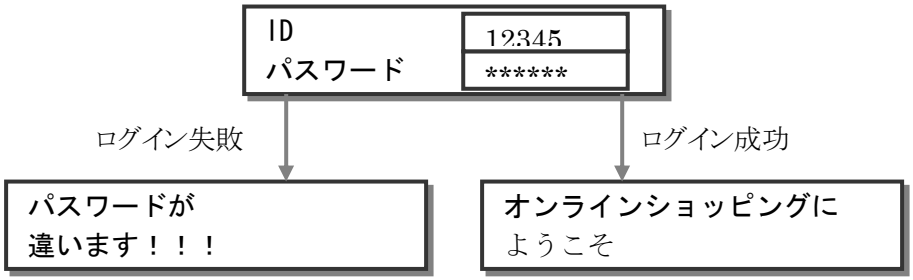
画面の表示と遷移

パッケージ名　*jp.co.tenartni.xsv.**

■ 画面の表示と遷移

利用者の要求に応じた画面の表示と遷移は、Webアプリケーションの構築における骨格であり、必ず必要な仕組みです。では、具体的にどのような仕組みが必要なのでしょうか？

例えばオンラインショッピングにおいて、利用者がログイン画面で ID とパスワードを入力したとします。ここで、ユーザ認証に成功すれば、オンラインショッピングのメイン画面が表示されます。しかし、利用者がパスワードを間違えて入力した場合、次に表示される画面には、「パスワードが違います！！！」などのメッセージが表示されます。



また、利用者がある商品を3つ購入したいとします。個数を入れるテキストフィールドに、誤って数字以外の文字を入れてしまい、気づかずに購入ボタンを押したとします。この場合、次に表示される画面には、「数字を入力してください。」などのエラーメッセージが表示されます。つまり、入力された値が数字かどうかをチェックしているのです。

よって上記から、画面の表示と遷移において、以下の仕組みが必要になります。

- 単に画面を移動するだけでなく、利用者からの入力データをチェックし、その内容に応じて次の画面やメッセージを表示する仕組み
- 利用者からの入力データを簡単にデータベースへ反映させることができる仕組み
- データベースから取得したデータから HTML（テーブルなど）を簡単に生成し、ブラウザに返す仕組み

上記の項目は、特定の画面に限った処理ではありません。これらを1画面ごとにコーディングするのは大変です。そこで DeveloperCafe は、入力データのチェック、データベースへの反映、HTML の自動生成などの処理を簡単に行えるコンポーネント（*HtmlForm*、*HtmlTableHandler*）を用意しています。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

■ DeveloperCafe の画面の表示と遷移

★ *HtmlForm*

・ 開発者は *HtmlForm* クラスを継承したカスタムフォームクラスを1画面ごとに作成し、これらにデータ定義、画面遷移ロジック、ビジネスロジックなど、画面の構築に必要なコードを記述します。インスタンスは、必要なフォーム名をセッションに指定することで取得できます。

以下はカスタムフォームクラスの主なメソッドと、メソッド内に記述する処理内容です。

カスタムフォームクラス(extends *HtmlForm*)

インスタンス生成 ：*HtmlTableHandler*、*ClientRowSet*

コンストラクタ ：フォーム全体で共通の処理を記述

- HTML のタイトル設定
- JSPファイルの設定
- *ClientRowSet* に *QueryAction* を設定
- テーブル属性の設定
- Submit ボタンの設定
- パラメータの設定 など

***initialize* ：フォームの初期化に伴う処理を記述**

- *DataHandler* をアクティブ化する、あるいは非アクティブ化する。
- メッセージテーブルをクリアする など

***doExecute* ：リクエストチェック、画面遷移等の実行を記述**

1. 送信されたボタンの値により、次のアクションを決定する。
2. リクエストチェックを行う。
3. データの取得、更新や、ビジネスロジックを処理する。
4. 他の画面に遷移する場合、セッションより必要なフォームのインスタンスを取得する。
5. 返り値として上記フォームを返す。 など

***onColmnSetup* ：カラム設定を記述**

- 見出し名の設定
- フォーマットパターンの設定 など

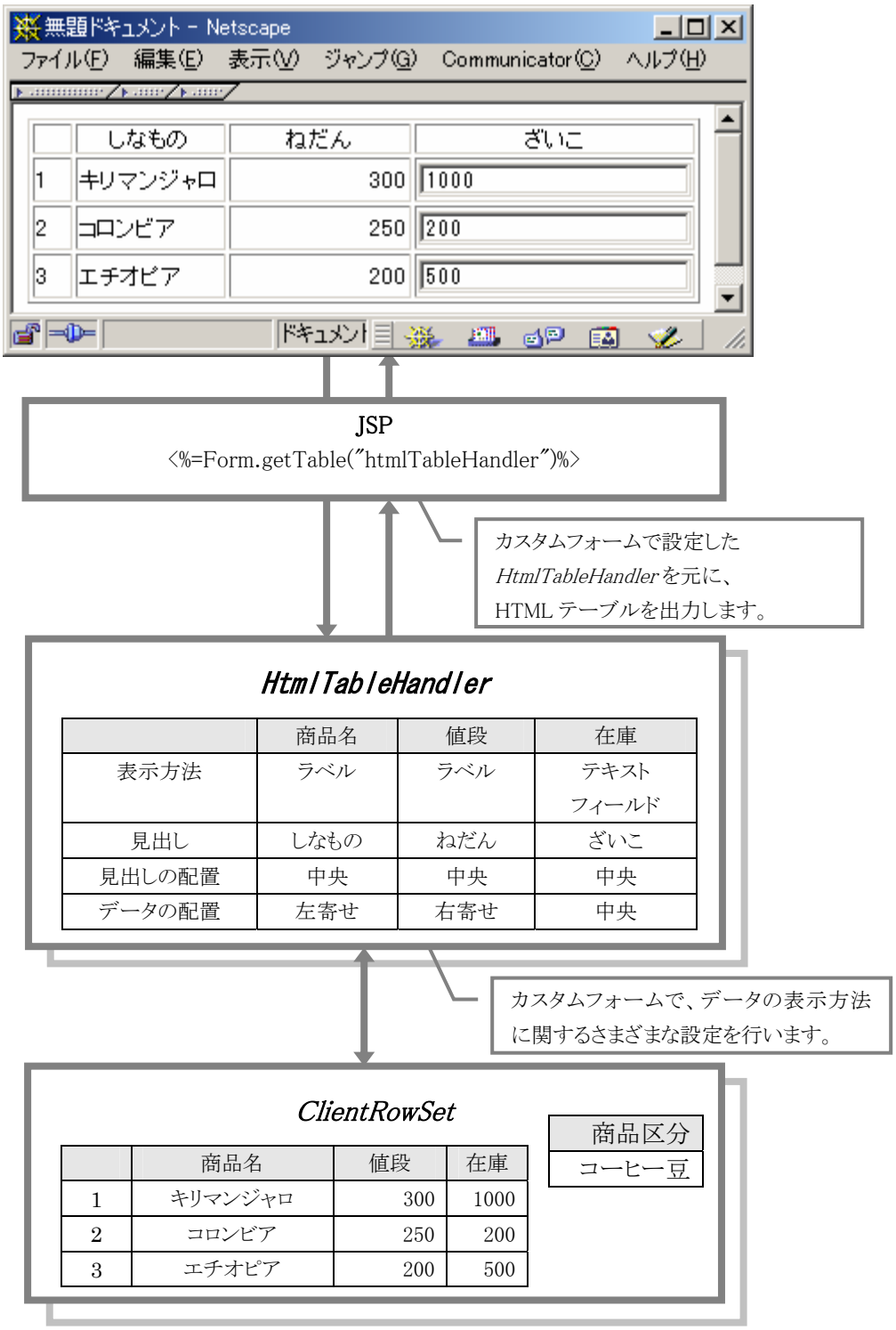
***onFieldSetup* ：フィールド設定を記述**

- 読み取り専用フィールドに設定
- 可視フィールドに設定
- フィールドタイプ（ラベル、テキストフィールドなど）の設定
- フィールド出力行位置の設定 など

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

★ *HtmlTableHandler*

・ *HtmlTableHandler* は *ClientRowSet* が保持しているデータをもとに HTML を自動生成します。又、自動生成した HTML からのリクエストを解釈し、*ClientRowSet* のデータを書き換えることができます。



Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.

■ サンプルコード

◆ *HtmlTableHandler* に関する処理

上図のように、*HtmlTableHandler* に各種表示に関する情報を設定するには、以下のメソッドを利用します。また、下記以外にも多くのメソッドがあります。詳細は、JavaDoc をご覧ください。

・ 表示方法

```
htmlTableHandler.getField("商品名").setFieldType(HtmlFieldType.INPUT_TEXT);
```

・ 見出し

```
htmlTableHandler.getField("商品名").setCaption("しなもの")
```

・ 見出しの配置

```
htmlTableHandler.setDefaultCaptionTDAttribute(″align=¥″center¥″″);
```

・ データの配置

```
htmlTableHandler.setDefaultDataTDAttribute(″align=¥″left¥″″);
```

◆ JSP に関する処理。

JSP ファイルから、カスタムフォームにアクセスするには、JSP ファイルに以下のタグを記述します。

```
<jsp:useBean id=″Form″ class=″jp.co.tenartni.xsv.HtmlFormClient″ scope=″request″/>
```

これで、"Form"を利用してカスタムフォームにアクセスすることができます。

さらに、JSP ファイルに上図のタグを記述することで、*HtmlTableHandler* の情報から HTML テーブルを生成し、ブラウザに出力することができます。

Copyright(C) 1997-2006 by SIOS Technology, Inc. All Rights Reserved.