

内容

1. 開発環境の準備	2
2. DB 環境の準備	4
3. メニュー	5
4. 多層ダイアログ	6
5. 動的画面	7
6. カレンダー	7
7. 二重クリック（重複登録）の防げ	9
8. 独自の入力チェック	11
8. 1 入力項目チェック	11
8. 2 アップロードファイルチェック	12
8. 3 画像認証	13
9. 画面遷移	14
10. データマッピング	15
11. Token チェック	17
12. 自作のデータソース	17
12. 1 DB との接続プール	17
12. 2 動的な XML の SQL 文の解析	19
12. 3 Access クラスの DB 操作	21
12. 4 Transact クラスの DB 操作	22
12. 5 五つの DB 種類がサポート	24
13. ファイルアップロード	25
13. 1 単独ファイルアップロード (Post)	25
13. 2 複数ファイルアップロード (Ajax)	26
14. Excel ファイル出力	27
15. PDF ファイル及びバーコード出力	29
16. 検索及び改ページ	29
17. 多国言語	33
18. 寧靜 DB ツールとの連携	35

寧靜フレームワークマニュアル

寧靜フレームワーク(njweb)は最新 J2EE 技術を利用し、各部品より緊密に連携し、ウェブプロジェクトがスムーズに開発できます。

下記には寧靜フレームワーク(njweb)を利用して開発されたサンプル(njsam)に巡って、説明します。実際の開発にもこのサンプルから改造すればと推薦します。

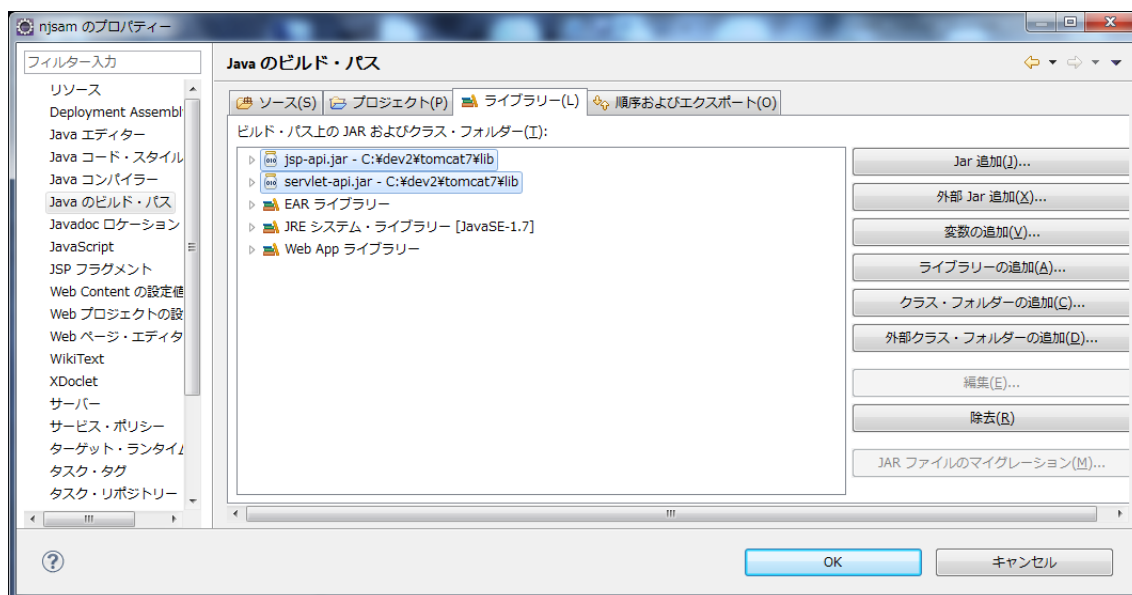
1. 開発環境の準備

1) 寧靜フレームワークには、下記の外部環境のバージョンに確認できました。

- Eclipse Indigo Service Release 2 / Mars Release (4.5.0)
- JDK 1.7 / 1.8
- Tomcat 7 / 8
- PostgreSQL 9 x64
- MySql 5.6
- SqlServer 2008 R2
- Oracle 11g
- DB2 Express-C 10.1

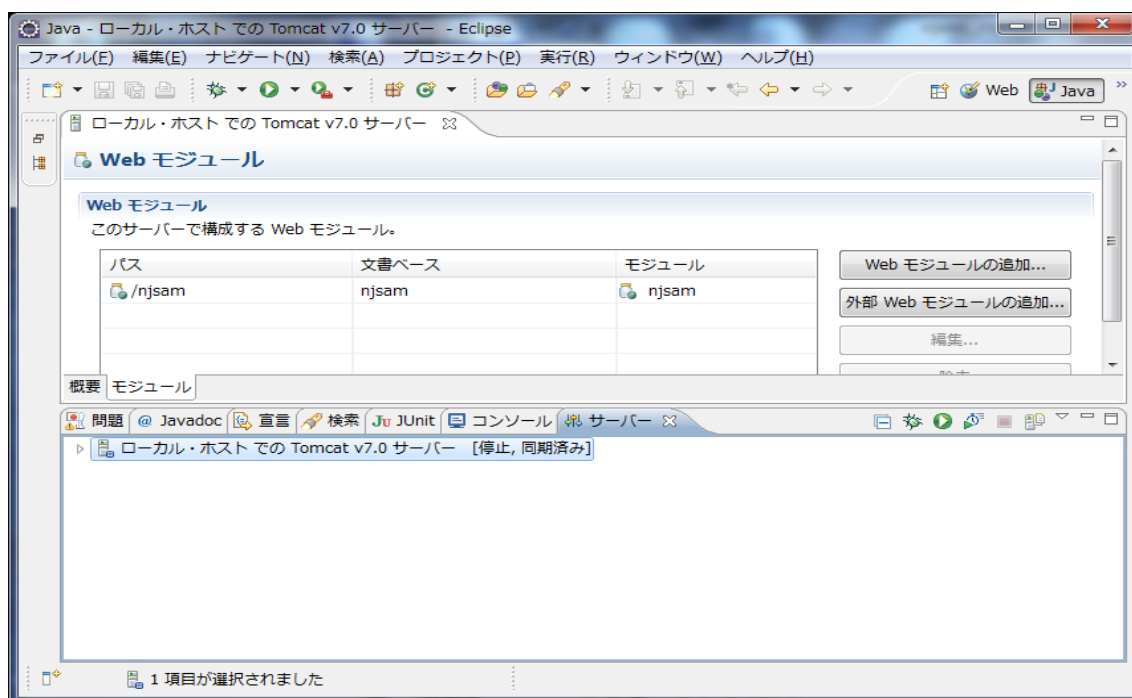
2) 上記のバージョン或いはその以上のバージョンの環境を用意して、njweb_japanese_1.00.zip ファイルを解凍してから、その中の njsam を Eclipse の既存プロジェクトとしてインポートします。

インポートしてから、jsp-api.jar と servlet-api.jar の所在フォルダを実の Tomcat 所在の lib フォルダに変更します。(図 1-1 Jar ファイル所在変更)



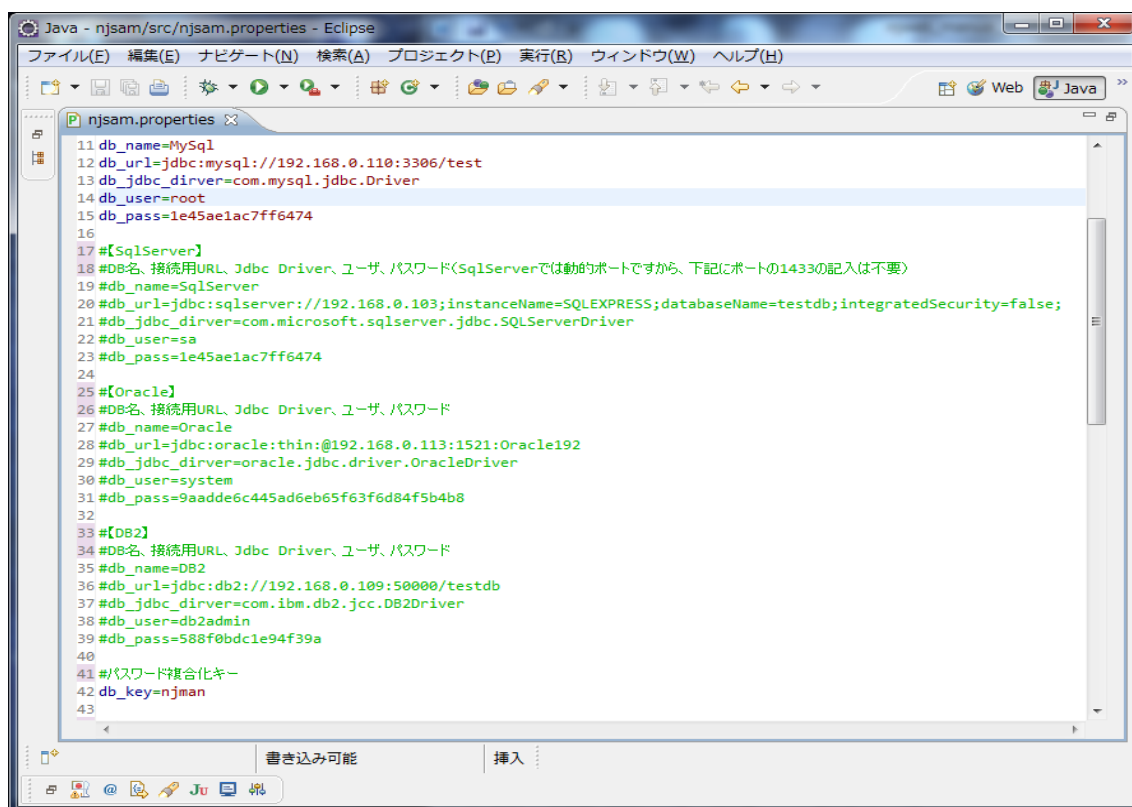
3) プロジェクト njsam をコンパイルしてから、Tomcat の Web モジュールに加入します。

(図 1 - 2 Web モジュール加入)

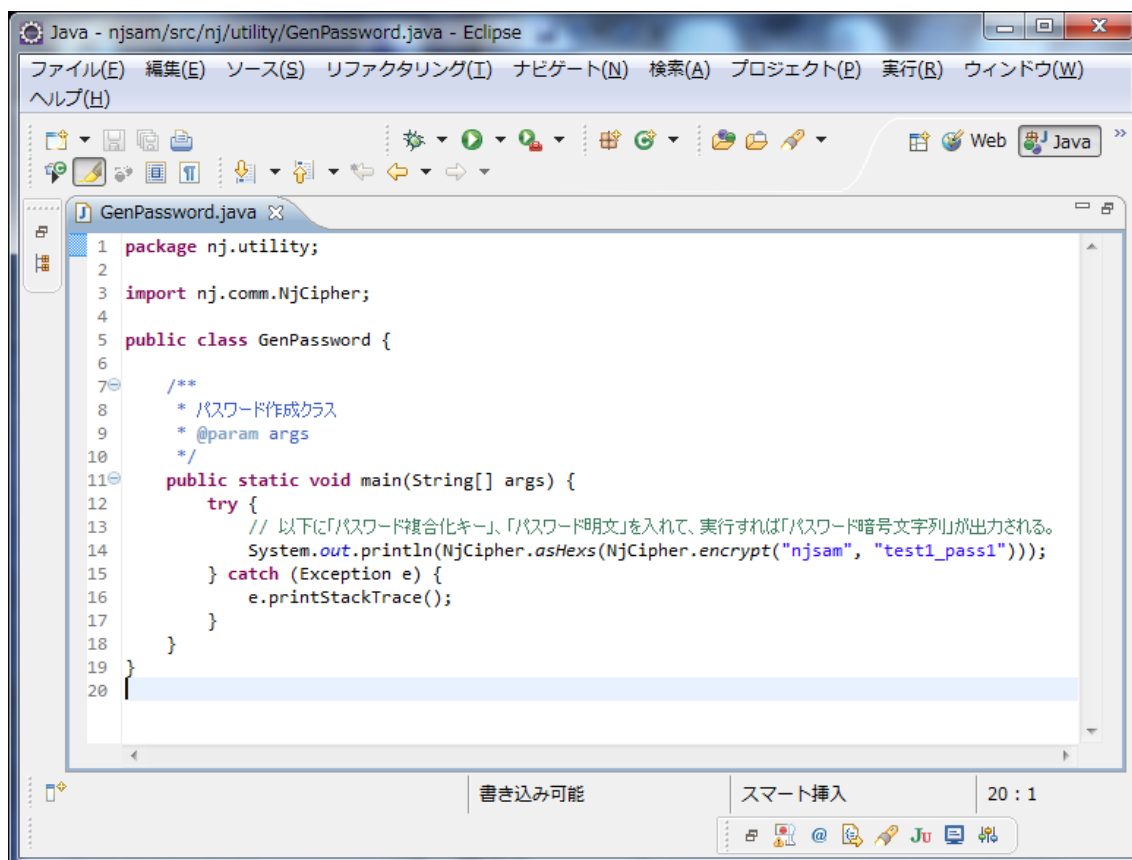


4) 実際利用すると合うように、下記の DB 定義ファイルを修正します。

(図 1 - 3 DB 定義変更)



5) 但し、上記の db_pass では暗号化後の文字列であるため、明文のパスワードから暗号化文字列の取得には、上記ファイルの db_key が利用されています。下記のソースに「パスワード複合化キー」(db_key の値)、「パスワード明文」を入れて、実行すれば「パスワード暗号文字列」が出力される。(図 1-4 パスワード暗号文字列の取得)



2. DB 環境の準備

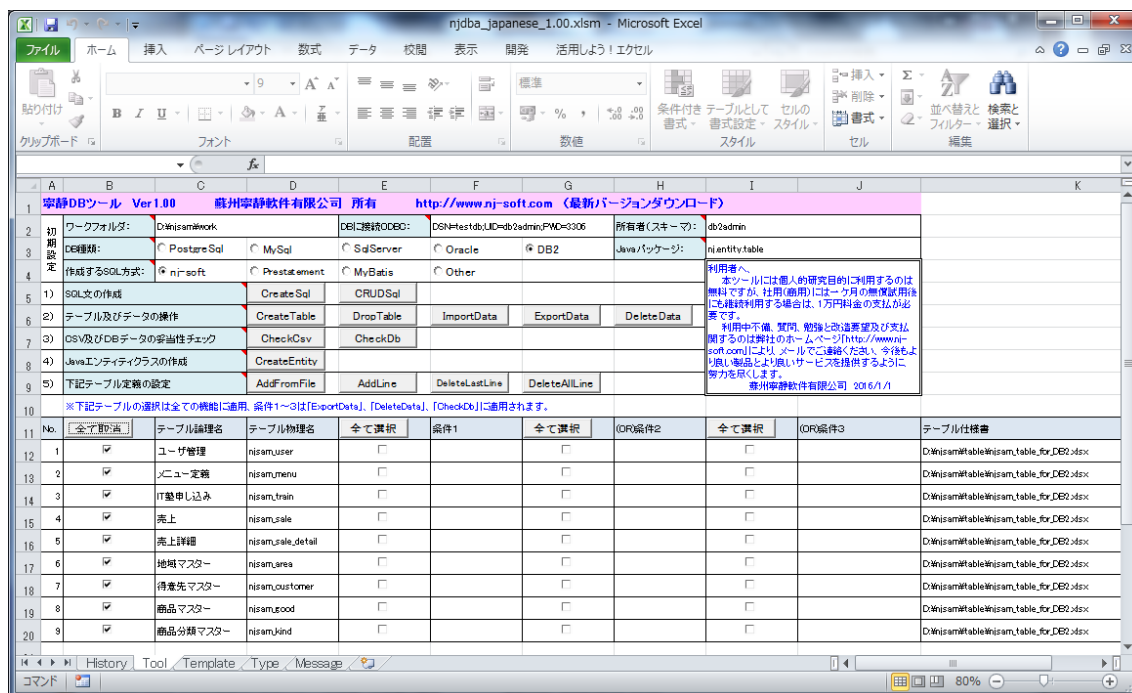
寧靜フレームワークには、PostgreSQL、MySQL、SqlServer、Oracle、DB2 など五つのデータベースがサポートされているが、実際に使われている DB のみテーブルやデータを準備すれば、OK です。

テーブルやデータの導入には寧靜 DB ツールが使われています、解凍した njsam.1.00_release.zip の中に、table の下にある DB 定義書と work¥Data の下にある csv データを、寧靜 DB ツールで物理テーブルの作成や、データのインポートを行います。

※寧靜 DB ツールの詳細には、弊社の HP からダウンロードできる寧靜 DB ツールを参照。

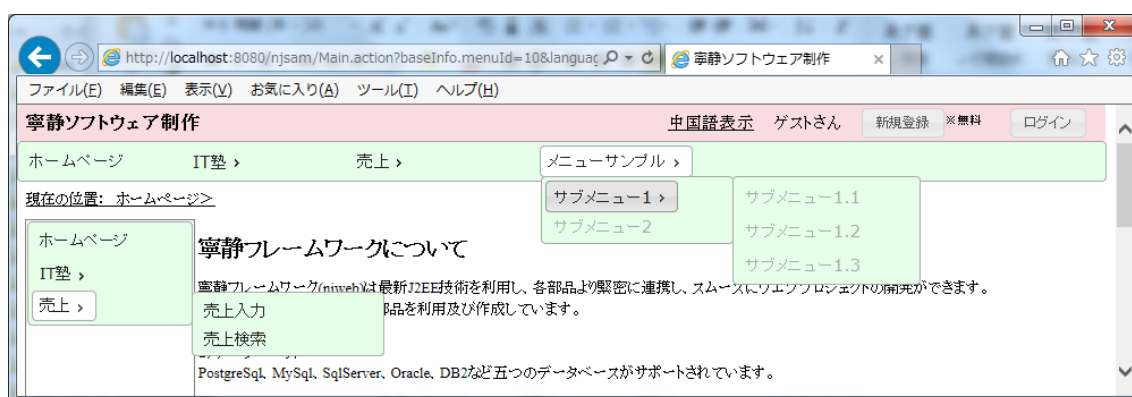
<http://www.nj-soft.com/njweb/Download.action?baseInfo.menuId=3030&baseInfo.curPage=0>

(図 2-1 寧静 DB ツールイメージ)



3. メニュー

寧静フレームワークには、jQuery を利用して、横と縦の多層メニューができます、パンくずリスト（現在の位置）も表示されます。権限などを設定すれば、メニューの活性・非活性の制御もできます。(図 3-1 メニューサンプル)



※メニューの作成には、下記のソースを参照。

/njsam/src/nj/action/ext/Help.java

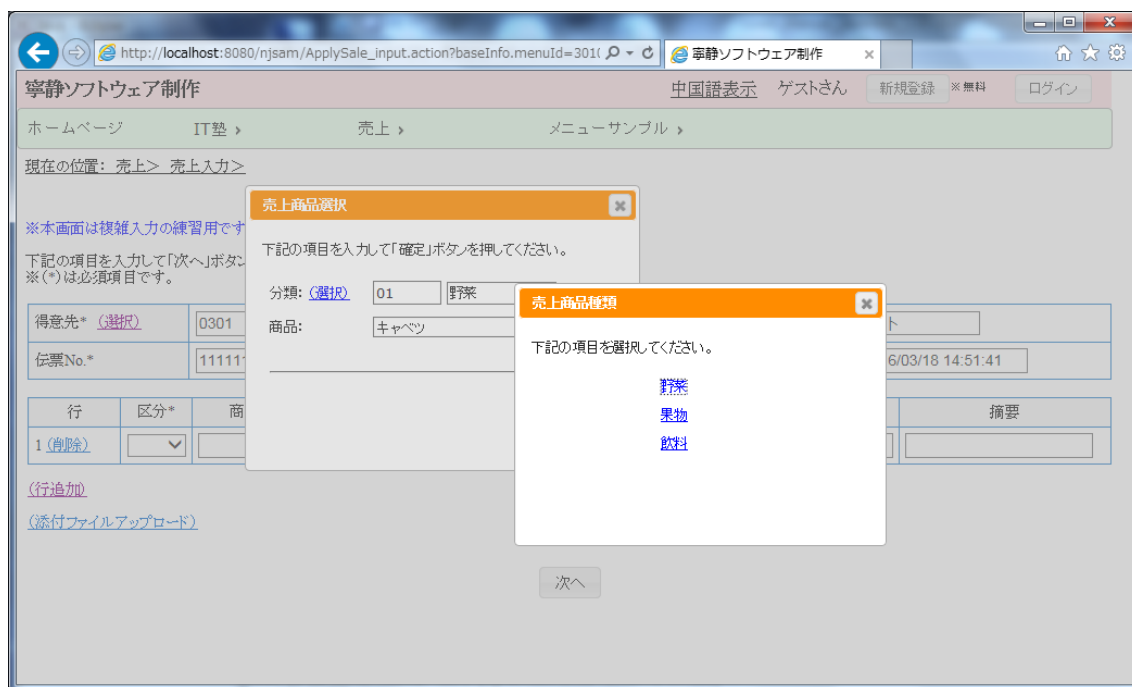
/njsam/WebContent/jsp/include/header.jsp

/njsam/WebContent/jsp2/include/header.jsp

4. 多層ダイアログ

寧靜フレームワークには、jQuery を利用して、多層ダイアログの制御ができます。下記の例は売上入力画面の商品コードの「(選択)」で、売上商品選択ダイアログを開き、更に売上商品選択ダイアログの分類の「(選択)」で、売上商品種類ダイアログを開きます。

(図 4-1 多層ダイアログ画面)



※多層ダイアログの作成には、下記のソースを参照。

```
/njsam/WebContent/jsp/sale/sale-input.jsp  
/njsam/WebContent/jsp2/sale/sale-input.jsp  
/njsam/src/nj/action/ApplySale.java
```

5. 動的画面

寧靜フレームワークには、jQuery を利用して、動的画面ができます。下記の例は「(行追加)」で、4 行まで追加しまして、更に 2 行目を削除した結果です。(図 5 - 1 動的画面)

寧靜ソフトウェア制作 中国語表示 ゲストさん 新規登録 ※無料 ログイン

ホームページ IT塾 > 売上 > メニューサンプル >

現在の位置: 売上> 売上入力>

売上入力

※本画面は複雑入力の練習用です、誰でもご自由に使えます。
 下記の項目を入力して「次へ」ボタンを押してください。
 ※(*)は必須項目です。

得意先* (選択)	0101	東京A社	担当者	ゲスト
伝票No.*	111111		注文日時	2016/03/23 10:54:04

行	区分*	商品コード*	商品名	売上数量	単位	売上単価	売上金額	摘要
1 (削除)	売上 ▼	0101 (選択)	人参	1	パック	200	200	
3 (削除)	売掛 ▼	0102 (選択)	キャベツ	3	個	300	600	
4 (削除)	売掛 ▼	0103 (選択)	白菜	4	個	400	1600	一つ箱

(行追加)
 (添付ファイルアップロード)

次へ

※動的画面の作成には、下記のソースを参照。

```
/njsam/WebContent/jsp/sale/sale-input.jsp
/njsam/WebContent/jsp2/sale/sale-input.jsp
/njsam/src/nj/action/ApplySale.java
```

6. カレンダー

画面の jsp に、単純にカレンダー部品をインクルードすれば、カレンダーで日付の選択入力ができます。下記の例は「希望開始日*」をクリックした後、日本語と中国語のカレンダー表示です。

(図 6 - 1 カレンダー日本語)

(図 6 - 2 カレンダー中国語)

※カレンダーの利用には、下記のソースを参照。

/njsam/WebContent/jsp/train/train-input.jsp

/njsam/WebContent/jsp2/train/train-input.jsp

7. 二重クリック（重複登録）の防げ

寧靜フレームワークには、ボタンやリンクをクリックしてコミットや検索する際、画面がロックされているため、二重クリック（重複登録）の防ぐことができます。実装にも簡単にできます。（図7-1 画面ロック）

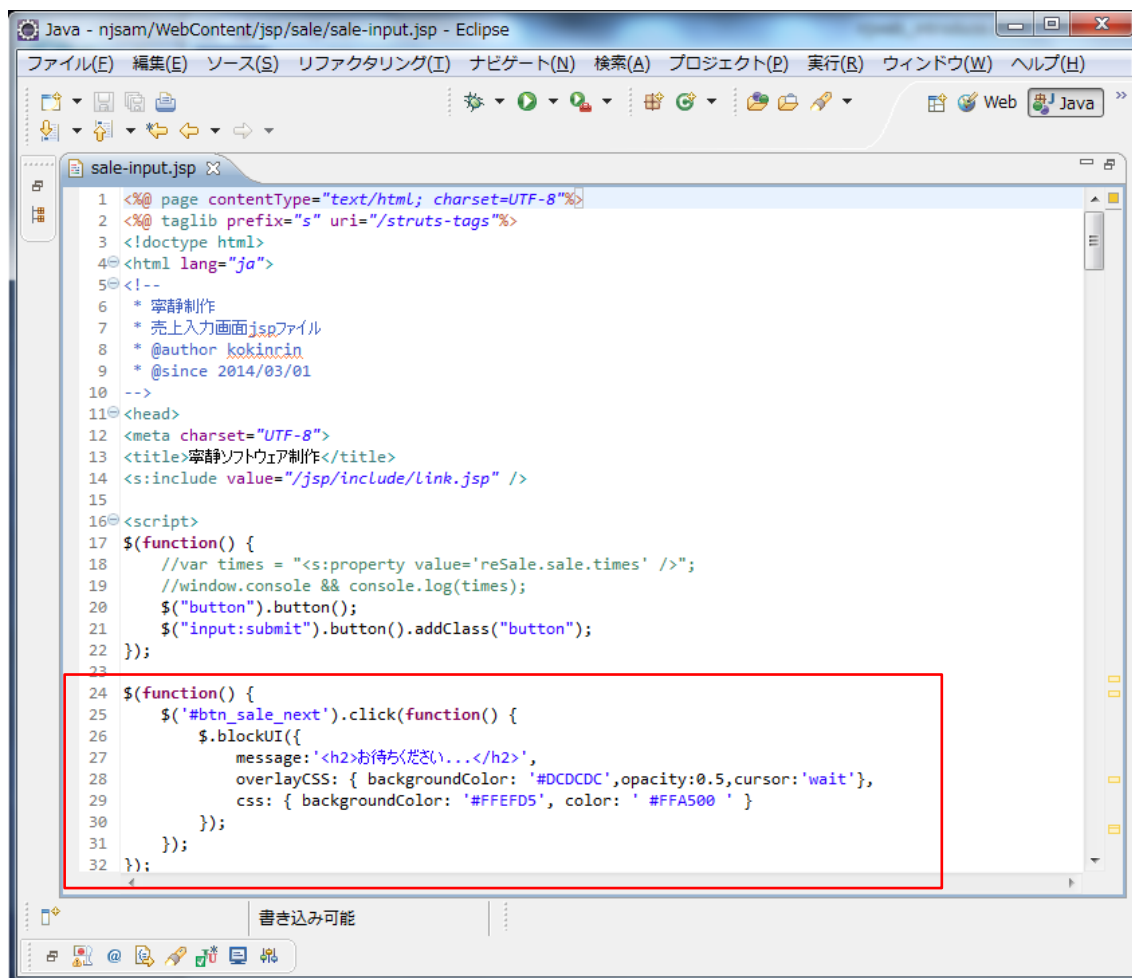
The screenshot shows a web browser window with the URL `http://localhost:8080/njsam/ApplySale_input.action?baseInfo.menuId=:`. The page title is 'localhost の待機中'. The application header includes '寧靜ソフトウェア制作' and navigation links like '中国語表示', 'ゲストさん', '新規登録', '※無料', and 'ログイン'. The main content area is titled '売上入力' (Sales Entry) and contains a form for entering sales data. A yellow confirmation message 'お待ちください...' (Please wait...) is displayed over the form, indicating a lock state. The form includes fields for '得意先*' (Customer), '担当者' (Staff), '伝票No.*' (Invoice No.), and a table for '商品' (Items) with columns for '行' (Line), '区分*' (Category), '商品コード*' (Item Code), '商品' (Item), '数量' (Quantity), '単価' (Unit Price), '売上金額' (Sales Amount), and '摘要' (Remarks). The table shows one item: 'リンゴ' (Apple) with a quantity of 1 and a sales amount of 100. A '次へ' (Next) button is at the bottom.

行	区分*	商品コード*	商品	数量	単価	売上金額	摘要
1 (削除)	売上 ▼	0201 (選択)	リンゴ	1	100	100	

[\(行追加\)](#)
以下は省略...

次へ

(図 7-2 画面ロック実装)



※画面ロック実装には、下記のソースを参照。

/njsam/WebContent/jsp/sale/sale-input.jsp

/njsam/WebContent/jsp2/sale/sale-input.jsp

8. 独自の入力チェック

寧靜フレームワークには、独自の入力チェックが組んでいます、エラーメッセージの表示や、エラー項目の背景色など、分かり易くて作っています。下記は入力項目チェック、アップロードファイルチェックと画像認証を分けて説明します。

8. 1 入力項目チェック

入力間違いや、必須項目の未入力など、下記のようにエラーメッセージとエラー項目が表示されます、しかもチェック関数は全て用意されているため、実装にはこれらの関数を呼び出すだけです。(図8-1 入力項目チェック)

ユーザ名(かな)は必須入力項目です。
性別は必須入力項目です。
入力したメールアドレスが正しくありません。
週に受講回数に半角数字を入力してください。
希望時間帯は必須入力項目です。

ユーザ名(漢字)	<input type="text"/>
ユーザ名(かな)*	<input type="text"/>
性別*	<input type="radio"/> 男 <input type="radio"/> 女
電話番号	<input type="text"/>
メールアドレス*	<input type="text" value="aaa"/>
写真	<input type="button" value="参照..."/>
希望開始日*	<input type="text" value="2016/02/15"/> ※YYYY/MM/DDで入力してください。
週に受講回数*	<input type="text" value="e"/> ※半角数字を入力してください。
希望時間帯*	※受講回数より多数を選択してください。
	日曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
	月曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
	火曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
	水曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
	木曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
	金曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
土曜日> <input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30	
希望受講場所	※受講場所はビビット南船橋2階であり、別場所の希望があれば記入してください。

※入力項目チェックには、下記のソースを参照。

```
/njsam/src/nj/action/ApplyTrain.java
/njsam/src/nj/action/chk/Check.java
/njsam/WebContent/jsp/train/train-input.jsp
/njsam/WebContent/jsp2/train/train-input.jsp
```

8. 2 アップロードファイルチェック

アップロードファイルチェックには、アップロードしたファイルのタイプや、サイズなどのチェックができます。下記の例は入力したファイルが画像ファイルではない場合のエラーです。

(図 8-2 アップロードファイルチェック)

寧靜ソフトウェア制作

中国語表示 ゲストさん 新規登録 ※無料 ログイン

ホームページ IT塾 > 売上 > メニューサンプル >

現在の位置: IT塾 > IT塾申込 >

IT塾申し込み入力

※本画面は複雑入力の練習用です、誰でも自由に使えます。
 ※塾は1対1で、基本知識を含んで、寧靜フレームワークを細かい講習します。
 ※講習に全ての最新ソースを無償提供します。
 ※講習は10回毎2時間、料金は合計10万円です、詳細内容は本サイトの「寧靜IT塾内容」を参照。
 ※プロジェクトやホームページの作成は10画面まで作成は無料です、
 10画面超えた場合は1画面毎に講習1回増えと5千円の追加料金になります。
 ※さくらのVPSの申請、DB、メール、Webサイトの構築は無償代行します。
 ※メールのWeb管理機能(アドレスの追加と削除、パスワード変更)も無償提供します。

入力したファイルのタイプが許可ではありません。

ユーザ名(漢字)	<input type="text"/>
ユーザ名(かな)*	<input type="text" value="a"/>
性別*	<input checked="" type="radio"/> 男 <input type="radio"/> 女
電話番号	<input type="text"/>
メールアドレス*	<input type="text" value="a@b.c"/>
写真	<input type="text"/> 参照...
希望開始日*	<input type="text" value="2016/02/15"/> ※YYYY/MM/DDで入力してください。
週に受講回数*	<input type="text" value="1"/> ※半角数字を入力してください。
	※受講回数より多数を選択してください。
日曜日>	<input checked="" type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30
曜日>	<input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30

※アップロードファイルチェックには、下記のソースを参照。

/njsam/src/nj/action/inter/InterUpload.java

8. 3 画像認証

画像認証にはロボットによる登録でないことを証明することです。下記の例は間違い入力です。(図8-3 画像認証チェック)

入力した画像認証文字が正しくありません。

ユーザ名(漢字)	
ユーザ名(かな)*	a
性別*	男
電話番号	
メールアドレス*	a@b.c
写真	
希望開始日*	2016/02/15
週に受講回数*	1
希望時間帯*	日曜日>指定なし
希望受講場所	※希望受講場所が入力されていないため、「ビビット南船橋2階」とします。
ご要望	
画像認証* ※ロボットによる登録でないことを証明	<div>lyyy</div> <div>※下記の文字を入力してください、読めない場合は変更します。</div> <div>65cb</div>

戻る 実行

※アップロードファイルチェックには、下記のソースを参照。

```

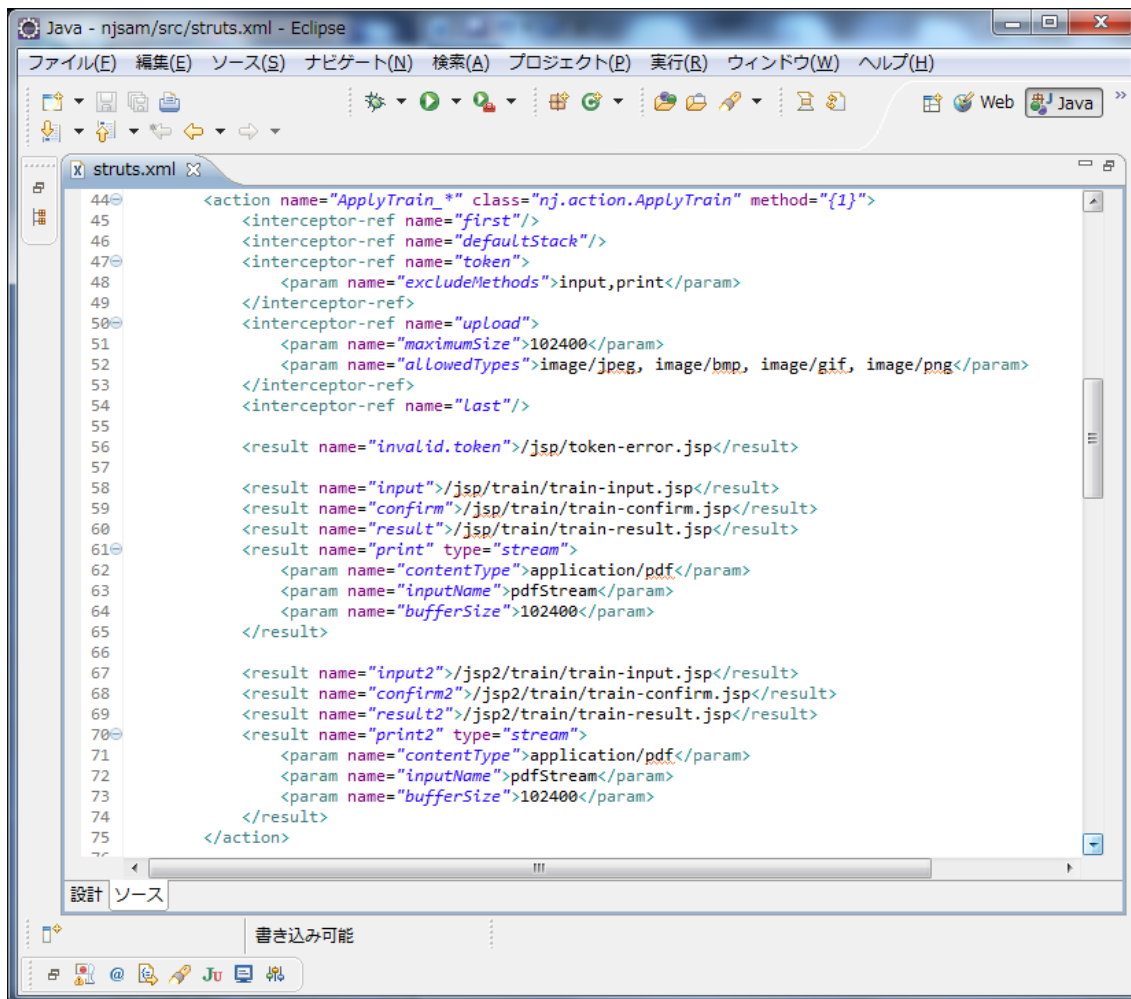
/njsam/src/nj/action/ajax/GetCaptcha. java
/njsam/src/nj/action/ApplyTrain. java
/njsam/src/nj/action/chk/Check. java
/njsam/WebContent/jsp/train/train-confirm. jsp
/njsam/WebContent/jsp2/train/train-confirm. jsp

```

9. 画面遷移

寧靜フレームワークの画面遷移には、Struts2 の遷移機能を使って、画面遷移を行います。
下記の設定例は日本語・中国語二つ言語の IT 塾申込の各画面の遷移です。

IT 塾申し込み入力（ファイルアップロードチェック、Token チェックも含み）←→IT 塾申し込み確認→IT 塾申し込み結果→IT 塾申し込み結果 PDF 出力。（図 9－1 画面遷移設定）



※画面遷移には、下記のソースを参照。

/njsam/src/struts.xml

/njsam/src/ajax.xml

10. データマッピング

寧靜フレームワークには Struts2 の OGNL (Object-Graph Navigation Language) によって、画面データと Java のエンティティクラス (リスト型やマップ型なども含み) の間に、直接的なセットと取得ができます。下記の例は売上入力画面の各項目と対応する Java エンティティクラスです。(図 10-1 データマッピング画面)

寧靜ソフトウェア制作 中国語表示 systemさん 新規登録 ※無料 ログイン

ホームページ IT塾 売上 メニューサンプル

現在の位置: 売上> 売上入力

売上入力

※本画面は複雑入力の練習用です、誰でも自由に使えます。

下記の項目を入力して「次へ」ボタンを押してください。
※(*)は必須項目です。

得意先* (選択)	0301	神奈川IE社	担当者	ゲスト
伝票No.*	111111		注文日時	2016/03/18 16:00:12

行	区分*	商品コード*	商品名	売上数量	単位	売上単価	売上金額	摘要
1 (削除)	売上 ▼	0102 (選択)	キャベツ	1	個	200	200	
2 (削除)	売掛 ▼	0103 (選択)	白菜	2	個	200	400	
3 (削除)	売上 ▼	0201 (選択)	リンゴ	2	パック	300	600	

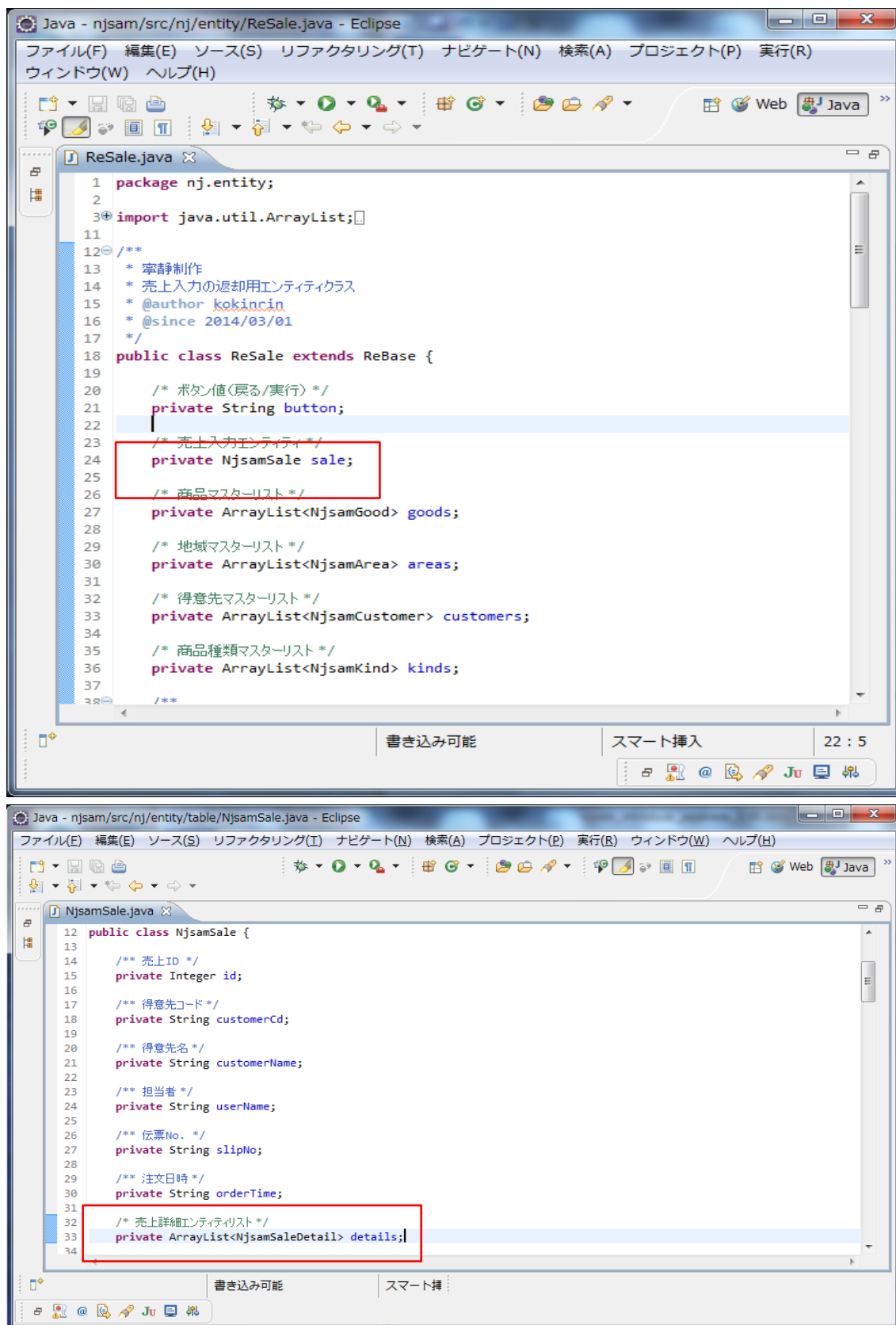
(行追加)

(添付ファイルアップロード)

1 注文書1.pdf (削除)
2 注文書2.pdf (削除)

次へ

(図 10-2 データマッピングエンティティクラス)

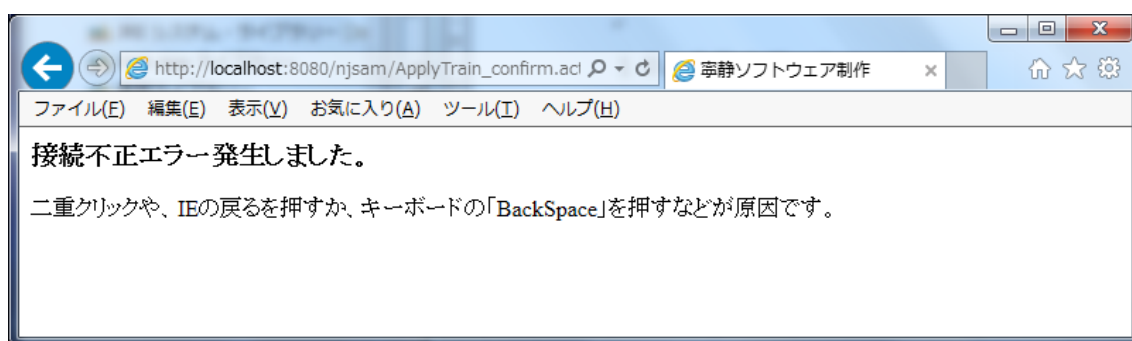


※データマッピングには、下記のソースを参照。

```
/njsam/src/nj/entity/ReSale. java  
/njsam/src/nj/entity/table/NjsamSale. java  
/njsam/src/nj/entity/table/NjsamSaleDetail. java  
/njsam/src/nj/action/ApplySale. java  
/njsam/WebContent/jsp/sale/sale-input. jsp  
/njsam/WebContent/jsp2/sale/sale-input. jsp
```

1 1. Token チェック

Token チェックには、二重登録や不正操作などを防ぐために、設けられています。寧靜フレームワークには Struts2 の Token チェックを利用し、単純に設定すればチェックできます。下記の例は IT 塾申し込み完了画面でブラウザの戻り (←) で、前の IT 塾申し込み確認画面に戻って、再度実行 (二重登録) するとエラーです。(図 1 1 - 1 Token チェック)



※Token チェックには、下記のソースを参照。

```
/njsam/src/nj/action/ApplyTrain. java  
/njsam/src/struts.xml
```

1 2. 自作のデータソース

寧靜フレームワークは自作のデータソースにより、DB との接続プール管理や、動的 SQL 文の解析や、SQL 文の改ページ実行など簡単に実現できます。

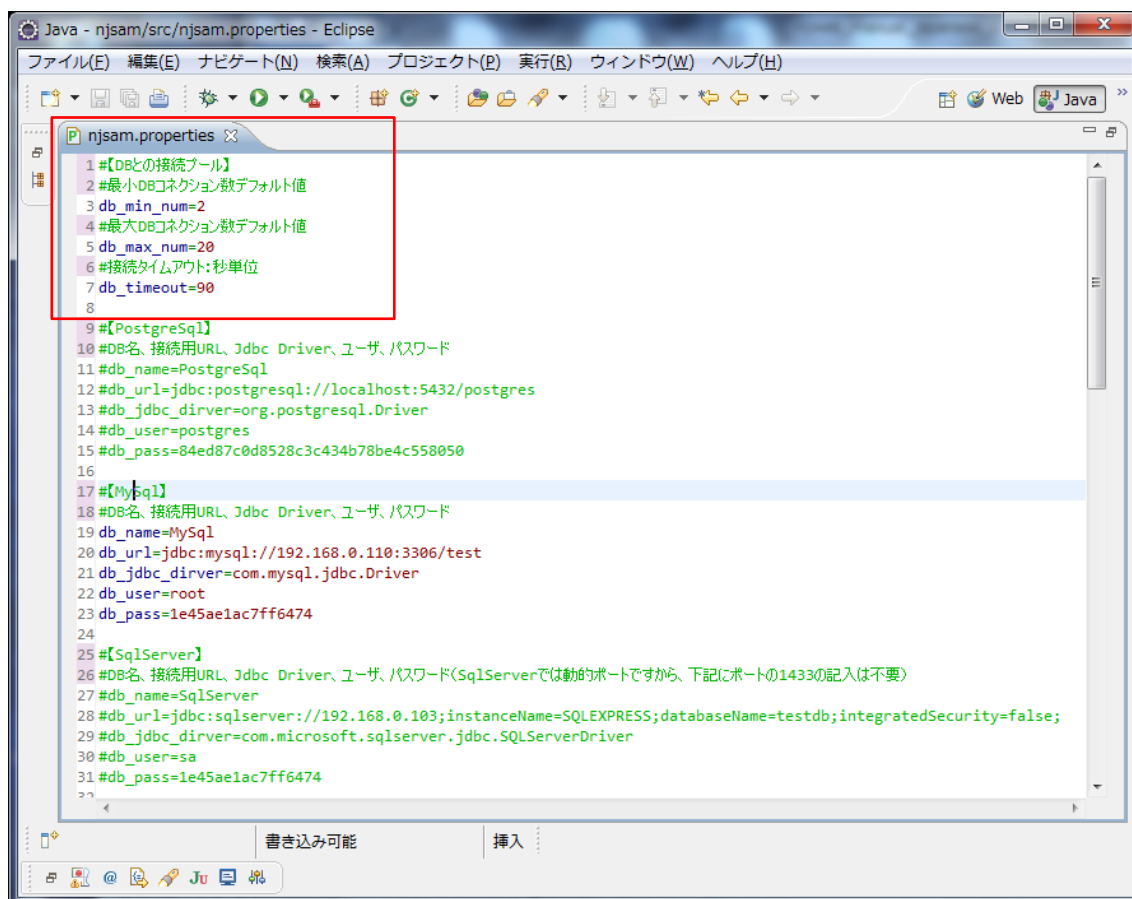
1 2. 1 DB との接続プール

下記のように DB との接続プールのパラメータを設定すれば、DB との接続プールが自動的に管理されます。ウェブサーバ起動する際、最小数の DB 接続を作成し、利用の申請を待ちます。

利用した接続のクローズが漏れた場合 (接続がプールに返還することが漏れた場合)、設定した接続タイムアウトであれば、監視スレッドでクローズされます。もちろん本フレームワークに提供した DB アクセス機能にはクローズ機能も整えています、それにウェブ系であれば、リクエスト単位で最後に利用中の接続は必ずクローズされます。

利用する接続が不足の場合は動的追加されます、但し、DB 接続数が設定した最大値に達成する場合は、新たな接続の作成ではなく、利用中の接続の返還を待ち続けます。最後にウェブサーバ停止するとき、プールに未利用と利用中の接続を全てロールバックし、クローズします。

(図 1 2 - 1 接続プール)



※接続プールの設定には、下記のファイルを参照。

/njsam/src/njsam.properties

(njsam はコンテキストのルートであり、プロジェクトのコンテキストのルートが変わると、このファイル名も変更してください。)

※詳細には下記の njcomm-1.00 API 仕様を参照。

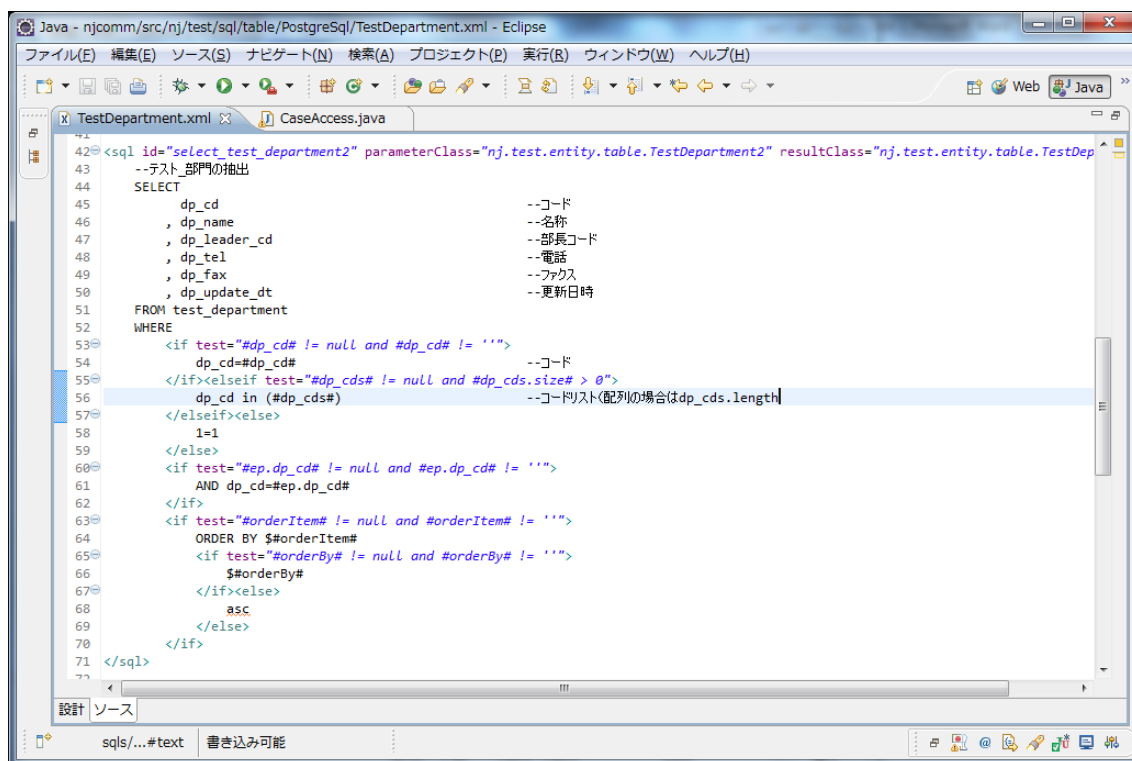
解凍したファイル njweb_japanese_1.00.zip の中の njcomm-api を参照 (index.html で起動)。

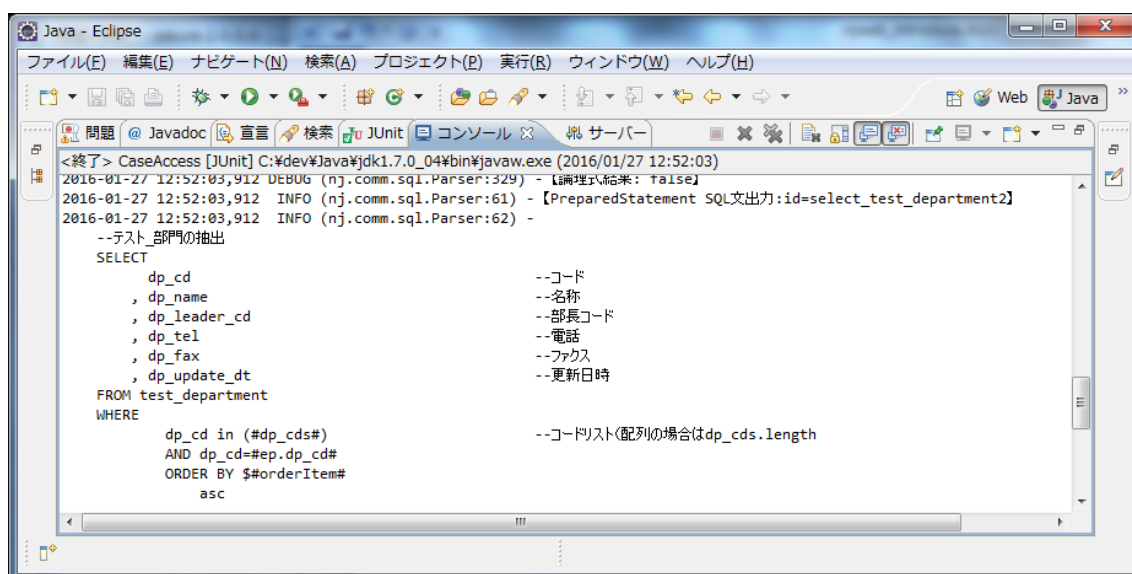
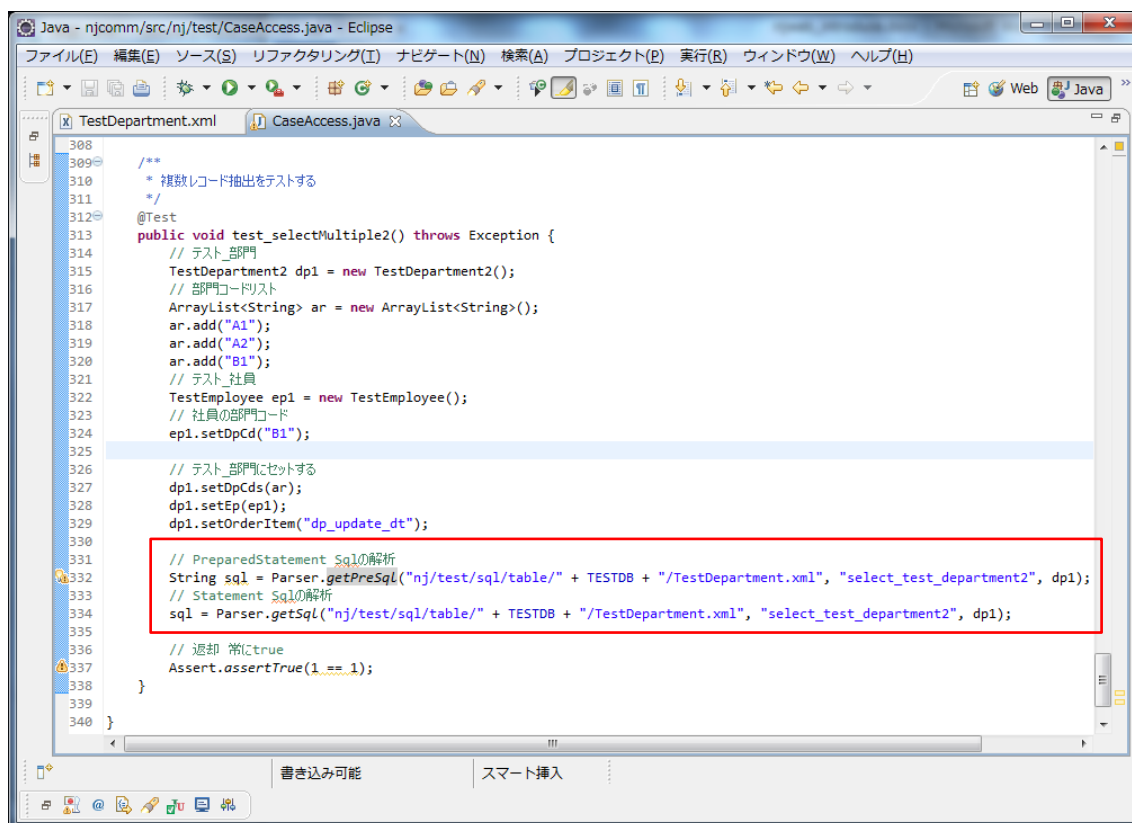
1 2. 2 動的な XML の SQL 文の解析

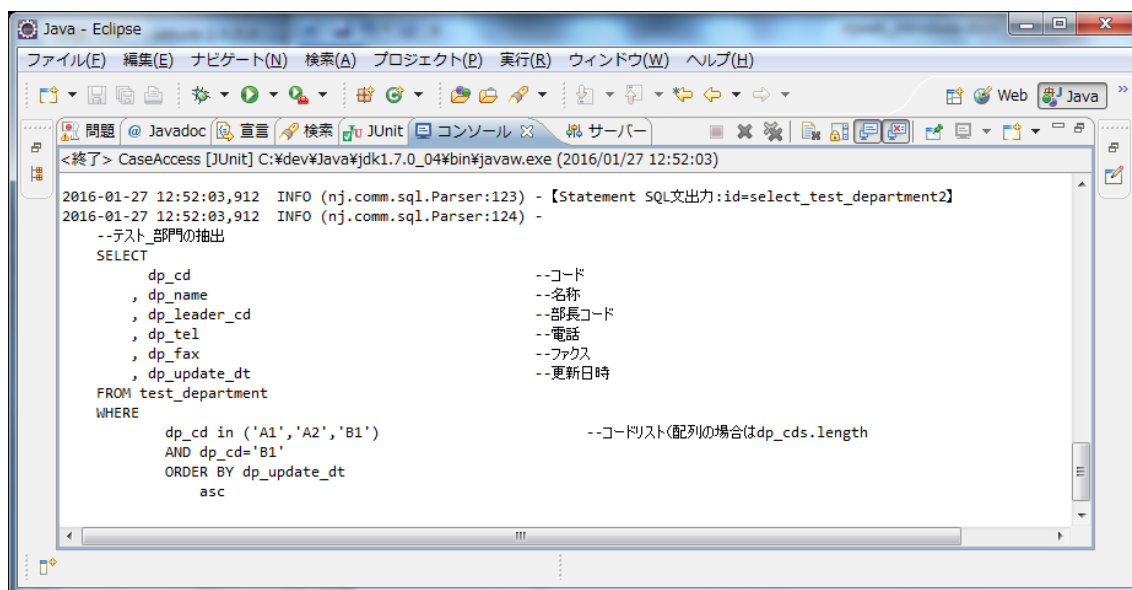
動的な XML の SQL 文の test 条件により、PreparedStatement 及び Statement の SQL の出力ができます。下記の例には定義した「テスト_部門」の XML を基づいて、「TestDepartment」エンティティクラスの属性の値によって、test 条件を判定し、PreparedStatement 及び Statement の SQL を出力します。

PreparedStatement の SQL の場合は、パラメータをバインドする際、一度「#FIELD#」を「?」に変換して、再度「?」に「#FIELD#」に対応する値をバインドします。

(図 1 2 - 2 XML の SQL 文の解析)







※詳細には下記の njcomm-1.00 API 仕様を参照。

解凍したファイル njweb_japanese_1.00.zip の中の njcomm-api を参照 (index.html で起動)。

1 2. 3 Access クラスの DB 操作

Access クラスには Statement の SQL をパラメータとして、SQL を実行して、接続をクローズします。バイナリフィールドを除いて、パラメータの値は全て事前に SQL 文に組んでいるため、SQL 文は分かり易くて出力できます。SQL 文の性能を極端に追及した場合は、PreparedStatement の SQL の Transact クラスを利用してください。

バイナリフィールドがある場合、本フレームワークで解析した Statement の SQL にも「?」が残っているため、バイナリフィールドの値を選択可能なパラメータとして渡せば、同じく SQL の実行ができます。

複数の更新 SQL が同一なトランザクションに更新したい場合は、下記の「updates」を利用するか、Transact クラスを利用してください。

Access クラスには以下の DB 操作が用意されています。

- 1) DB に新規登録 (重複キー判定)

```
public static int insert(String sql, byte[]... bss) throws Exception
```

- 2) DB に更新 (重複キー判定しない)

```
public static int update(String sql, byte[]... bss) throws Exception
```

- 3) DB に複数新規登録及び更新 (重複キー判定)

```
public static int updates(String[] sqls, Integer[] effects, byte[][]... bss)
throws Exception
```

- 4) DB に 1 レコード検索

```
public static Object selectSingle(String sql, String entityName) throws Exception
```

5) DB に複数レコード検索

```
public static Object selectMultiple(String sql, String entityName) throws  
Exception
```

6) DB にページ単位レコード検索

```
public static void selectPage(String sql, IntSelPage selPage) throws Exception
```

7) 検索結果を CSV ファイルに出力する。

```
public static Integer writeCsvFile(String sql, String path, String charset,  
String head) throws Exception
```

※詳細には下記の njcomm-1.00 API 仕様を参照。

解凍したファイル njweb_japanese_1.00.zip の中の njcomm-api を参照 (index.html で起動)。

1 2. 4 Transact クラスの DB 操作

Transact クラスには事前に DB との接続の取得 (Transact.getInstance(this)) が必要です、PreparedStatement の SQL とパラメータ値を含んでエンティティクラスをパラメータとして実行します、実行の後にも、接続のクローズはしません、最後に Application 系の場合は手動でクローズが必要です、Web 系の場合はリクエストの最後に自動的にクローズされます。

Transact クラスには以下の DB 操作が用意されています。

1) インスタンス取得

it Transact インターフェース (同じ Transact インターフェースの場合は、同一インスタンスが取得される。)

```
public static synchronized Transact getInstance(IntTransact it) throws Exception
```

2) コミットする

```
public void commit() throws Exception
```

3) ロールバックする

```
public void rollback() throws Exception
```

4) コミットしてから、DB との接続をクローズ (プールへ返還)

```
public void close() throws Exception
```

5) DB に新規登録 (重複キー判定)

```
public int insert(String sql, Object entity) throws Exception
```

6) DB に更新 (重複キー判定しない)

```
public int update(String sql, Object entity) throws Exception
```

7) DB に複数新規登録及び更新 (重複キー判定)

```
public int updates(String[] sqls, Integer[] effects, Object[] entities) throws  
Exception
```

8) DB に 1 レコード検索

```
public Object selectSingle(String sql, String entityName, Object entity) throws  
Exception
```

9) DB に複数レコード検索

```
public Object selectMultiple(String sql, String entityName, Object entity) throws  
Exception
```

10) DB にページ単位レコード検索

```
public void selectPage(String sql, IntSelPage selPage, Object entity) throws  
Exception
```

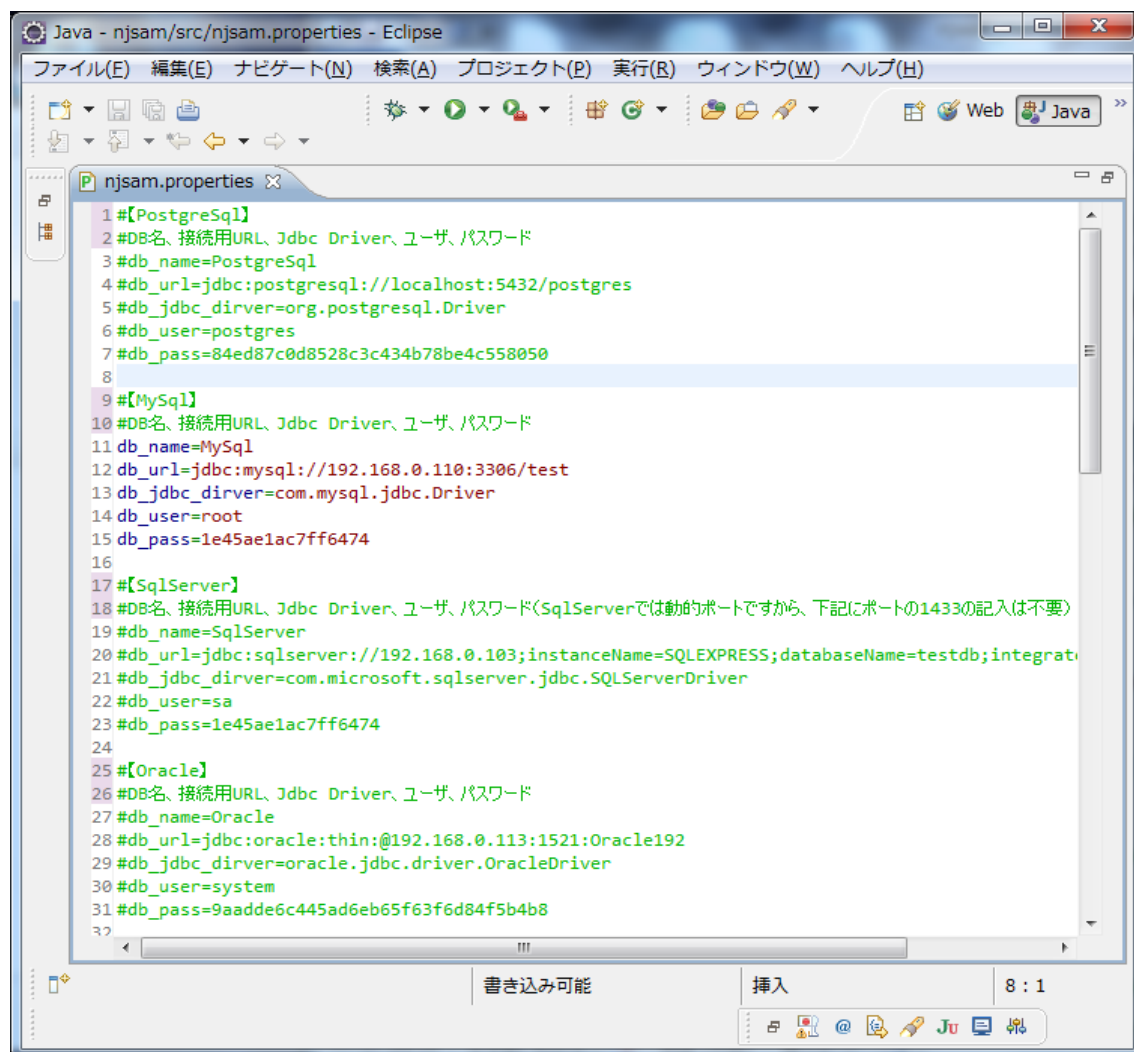
※詳細には下記の njcomm-1.00 API 仕様を参照。

解凍したファイル njweb_japanese_1.00.zip の中の njcomm-api を参照 (index.html で起動)。

1 2. 5 五つの DB 種類がサポート

寧靜フレームワークにはPostgreSQL、MySQL、SqlServer、Oracle、DB2 など五つのデータベースがサポートされています。下記の設定のみ変更すれば、全ての機能が適用されます。

(図 1 2 - 3 五つの DB 種類がサポート)



1 3. ファイルアップロード

寧靜フレームワークは Post 方式と Ajax 方式のファイルアップロードがサポートされています。

1 3. 1 単独ファイルアップロード (Post)

Post 方式には他の入力項目と同時に、サブミットする際、アップロードしたファイルを Java エンティティクラスの byte[] 属性にセットされています。アップロードしたファイルのチェックにも同じく用意されています。以下の例は「写真」項目のファイルアップロードです。

(図 1 3 - 1 Post 方式ファイルアップロード)

寧靜ソフトウェア制作 中国語表示 ゲストさん 新規登録 ※無料 ログイン

ホームページ IT塾 売上 メニューサンプル

現在の位置: IT塾 > IT塾申込

IT塾申し込み入力

※本画面は複雑入力の練習用です、誰でもご自由に使えます。
 ※塾は1対1で、基本知識を含んで、寧靜フレームワークを細かい講習します。
 ※講習に全ての最新ソースを無償提供します。
 ※講習は10回毎2時間、料金は合計10万円です、詳細内容は本サイトの「寧靜IT塾内容」を参照。
 ※プロジェクトやホームページの作成は10画面まで作成は無料です、
 10画面を超えた場合は1画面毎に講習1回増えと5千円の追加料金になります。
 ※さくらのVPSの申請、DB、メール、Webサイトの構築は無償代行します。
 ※メールのWeb管理機能(アドレスの追加と削除、パスワード変更)も無償提供します。

下記の項目を入力して「次へ」ボタンを押してください。
 ※(*)は必須項目です。

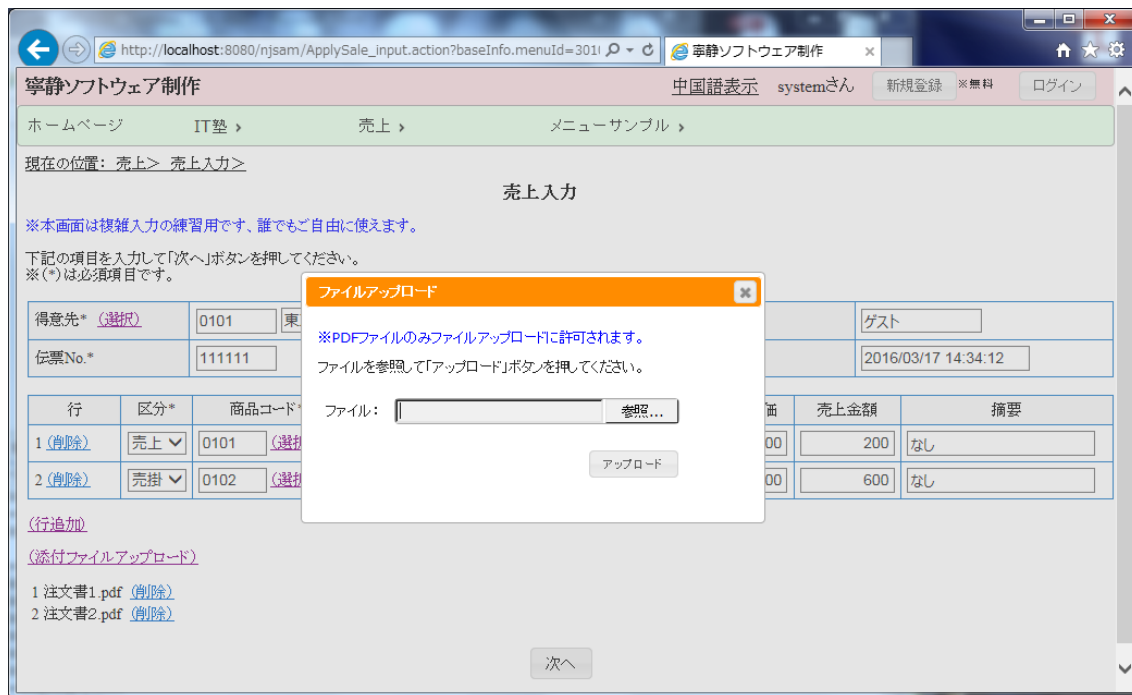
ユーザ名(漢字)	<input type="text"/>
ユーザ名(かな)*	<input type="text" value="寧靜"/>
性別*	<input checked="" type="radio"/> 男 <input type="radio"/> 女
電話番号	<input type="text"/>
メールアドレス*	<input type="text"/>
写真	<input type="text" value="C:\temp\1.jpg"/> 参照...
希望開始日*	<input type="text"/> ※YYYY/MM/DDで入力してください。
週に受講回数*	<input type="text"/> ※半角数字を入力してください。
	※受講回数より多数を選択してください。
日曜日	<input type="checkbox"/> 指定なし <input type="checkbox"/> 10:15-12:15 <input type="checkbox"/> 14:30-16:30 <input type="checkbox"/> 18:30-20:30

※単独ファイルアップロード (Post) には、下記のファイルを参照。

```
/njsam/src/nj/action/ApplyTrain.java
/njsam/src/nj/action/inter/InterUpload.java
/njsam/WebContent/jsp/train/train-input.jsp
/njsam/WebContent/jsp2/train/train-input.jsp
```

1 3. 2 複数ファイルアップロード (Ajax)

Ajax 方式にはダイアログ画面で一つずつファイルをアップロードして、サーバ側に一時的保存し、最後主画面サブミットする際処理を行います。アップロードしたファイルのチェックにはダイアログ画面にチェックされています。(図 1 3-2 Ajax 方式ファイルアップロード)



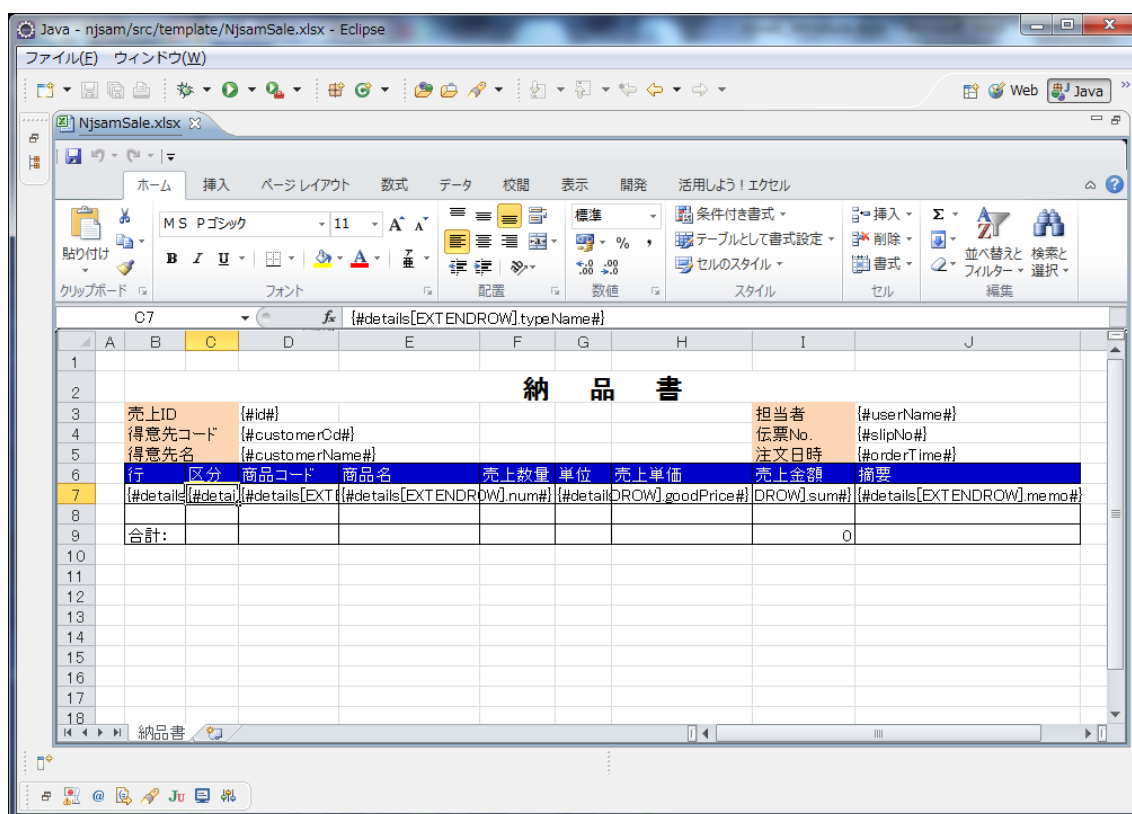
※複数ファイルアップロード (Ajax) には、下記のソースを参照。

```
/njsam/src/nj/action/ajax/Upload.java
/njsam/src/nj/action/inter/InterUpload.java
/njsam/WebContent/jsp/include/upload.jsp
/njsam/WebContent/jsp2/include/upload.jsp
```

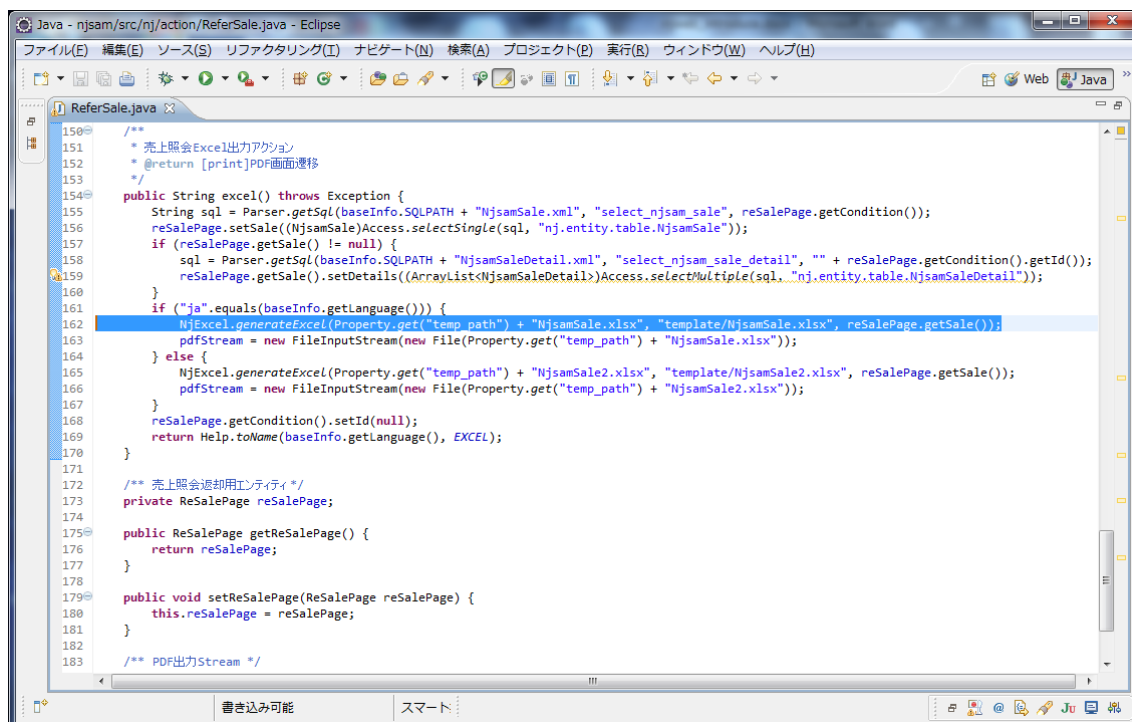
1 4. Excel ファイル出力

寧靜フレームワークはPoiを利用して、Excel ファイルの出力が簡単にできます。Excel のテンプレートとエンティティクラスを直接結びつけ、サブエンティティや、配列、リストなど複雑のデータ型にもサポートされています。それと配列とリストには、自動的 Excel に縦及び横に展開することもできます、二重配列とリストなら、同時に横・縦展開もできます。下記の例にはテンプレートにより、商品明細が行に展開されています。

(図 1 4 - 1 Excel テンプレート)



(図 1 4 - 2 Excel 作成実装)



(図 1 4 - 3 Excel ファイル出力)

行	区分	商品コード	商品名	売上数量	単位	売上単価	売上金額	摘要
1	売上	102	キャベツ	2	個	200	400	
2	売上	101	人参	1	パック	125	125	
3	売掛	201	リンゴ	5	個	80	400	一つ箱
合計:							925	

※Excel ファイル出力には、下記のソースを参照。

/njsam/src/nj/action/ReferSale.java

/njsam/src/template/NjsamSale.xlsx

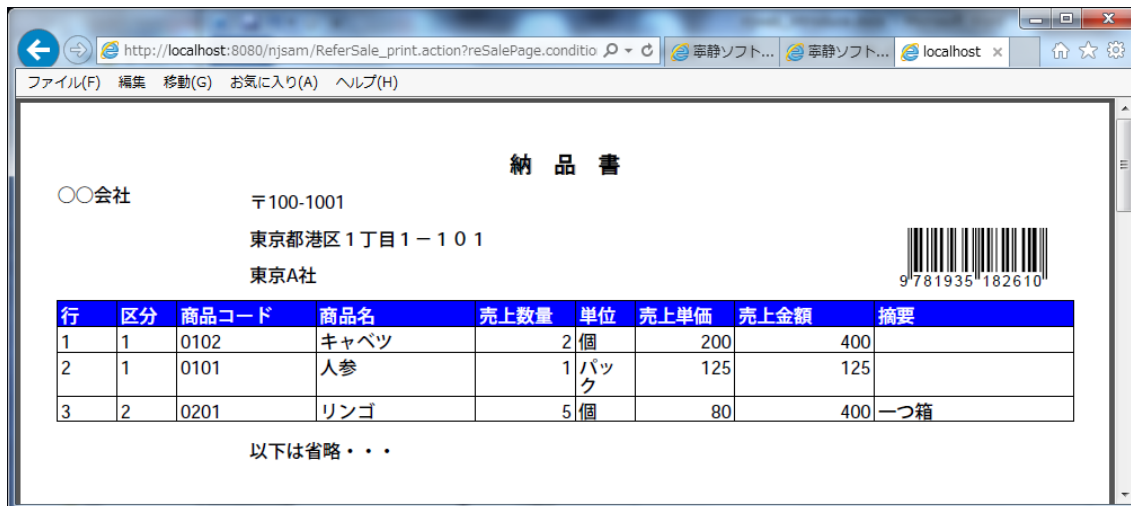
/njsam/src/template/NjsamSale2.xlsx

/njsam/WebContent/jsp/sale/sale-refer-result.jsp

/njsam/WebContent/jsp2/sale/sale-refer-result.jsp

1 5. PDF ファイル及びバーコード出力

寧静フレームワークは iText を利用して、PDF ファイルにテーブルや、バーコードなど、より細かいことが出力できます。(図 1 5 - 1 PDF ファイル出力)



※PDF ファイル及びバーコード出力には、下記のソースを参照。

/njsam/src/nj/action/ext/SalePdf. java

/njsam/src/nj/action/ext/SalePdf2. java

/njsam/WebContent/jsp/sale/sale-refer-result. jsp

/njsam/WebContent/jsp2/sale/sale-refer-result. jsp

1 6. 検索及び改ページ

寧静フレームワークはページ単位で検索機能が用意されています、普通の検索 SQL のみで、検索するページとページ毎件数を指定すれば、該当ページのレコードの検出ができます、しかも指定ページが範囲外の場合は最終ページが検出されます。

実装には業務ロジックに注力し、普通の検索 SQL のみで OK ですから、改ページの SQL や、Java ロジックなどには力が必要ありません、画面の jsp にも改ページの部品が用意されているため、より簡単にページ単位の検索が実現できます。

改ページにも五つのデータベースがサポートされているため、データベース名を指定すれば、普段の SQL 文から、全件数検索 SQL 文と該当ページの検索 SQL 文が自動に作成できます。

(図 16-1 ページ単位で検索)

[←](#)
[→](#)
http://localhost:8080/njsam/ReferSale_refer.action?reSalePage.type=2
[🔍](#)
[🔗](#)

[🏠](#)
[★](#)
[⚙️](#)

[ファイル\(E\)](#)
[編集\(E\)](#)
[表示\(V\)](#)
[お気に入り\(A\)](#)
[ツール\(T\)](#)
[ヘルプ\(H\)](#)

[寧静ソフトウェア制作](#)

[中国語表示](#)
[ゲストさん](#)
[新規登録](#)
[※無料](#)
[ログイン](#)

[ホームページ](#)
[IT塾 >](#)
[売上 >](#)
[メニューサンプル >](#)

[現在の位置: 売上 > 売上検索 >](#)

売上検索

※本画面は複雑検索の練習用です、誰でも自由に使えます。

検索条件を入力してから「検索」ボタンを押してください。

得意先 (選択)	<input type="text"/>	<input type="text"/>	担当者	<input type="text"/>
伝票No.	<input type="text"/>		注文日	<input type="text"/>
ソート項目	<input checked="" type="radio"/> 得意先 <input type="radio"/> 担当者 <input type="radio"/> 注文日		ソート順	<input type="radio"/> 昇順 <input type="radio"/> 降順

検索

※詳細内容には「詳細」リンクを押してください。

現在の表示: 11 - 20 / 21

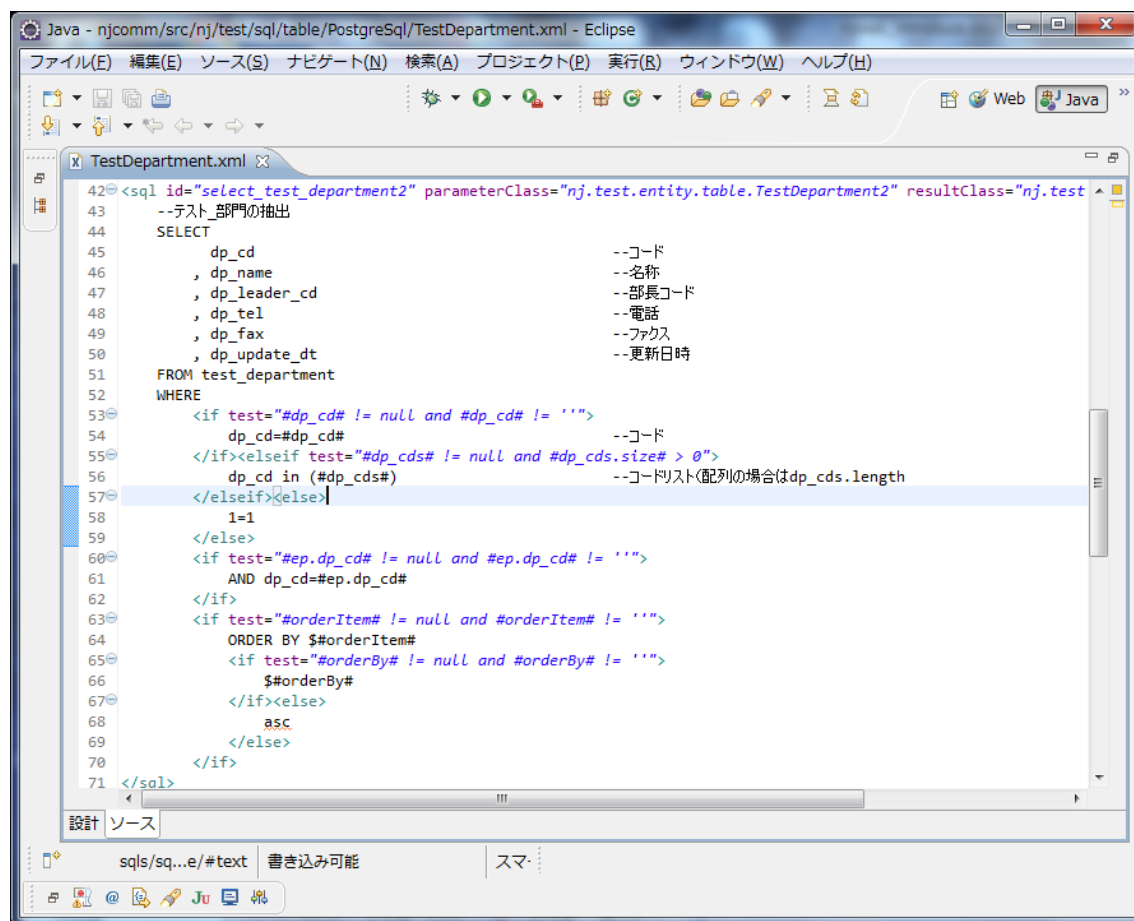
[最初ページ](#)
[前ページ](#)
[次ページ](#)
[最後ページ](#)

得意先	担当者	伝票No.	注文日時	売上商品	売上金額	摘要	
0201 千葉C社	ゲスト	111111	2016/01/20 12:35:43	0202 ミカン 0203 スイカ	1000 1000	なし なし	詳細
0201 千葉C社	过客	333	2016/01/20 16:13:49	0202 ミカン	3	3	詳細
0201 千葉C社	过客	333	2016/01/20 16:13:49	0202 ミカン	3	3	詳細
0201 千葉C社	ゲスト	111111	2016/01/27 22:20:54	0101 人参	1	1	詳細
0202 千葉D社	a	10	2016/01/16 19:32:18	0201 リンゴ	10		詳細
0301 神奈川E社	a	777777	2016/01/16 19:30:02	0201 リンゴ	7	7	詳細
0301 神奈川E社	a	444444	2016/01/16 19:27:19	0202 ミカン	4	4	詳細
0302 神奈川E社	a	222222	2016/01/16 19:27:19	0203 スイカ	2	2	詳細

(図 16-2 改ページの jsp 部品)

[illegible]

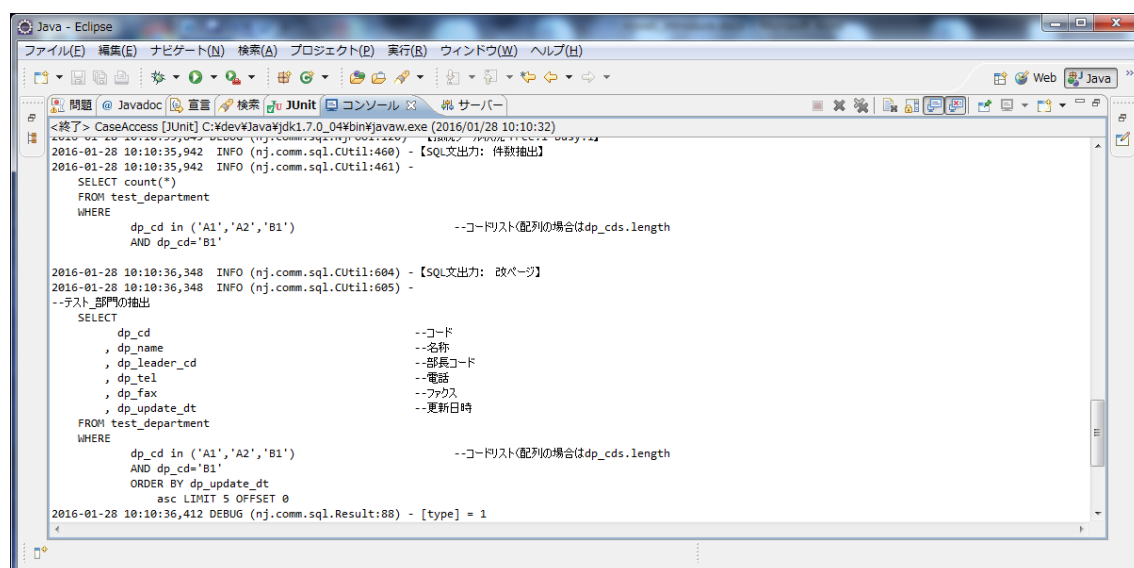
1) 下記の例は PostgreSQL の普通 SQL 文から、自動的に件数検索 SQL 文とページ検索 SQL 文の作成です。(図 16-3 PostgreSQL のページ SQL)



```

42 <sql id="select_test_department2" parameterClass="nj.test.entity.table.TestDepartment2" resultClass="nj.test
43 --テスト_部門の抽出
44 SELECT
45     dp_cd                                --コード
46     , dp_name                            --名称
47     , dp_leader_cd                       --部長コード
48     , dp_tel                             --電話
49     , dp_fax                             --ファクス
50     , dp_update_dt                       --更新日時
51 FROM test_department
52 WHERE
53     <if test="#dp_cd# != null and #dp_cd# != ''">
54         dp_cd=#dp_cd#                    --コード
55     </if><elseif test="#dp_cds# != null and #dp_cds.size# > 0">
56         dp_cd in (#dp_cds#)              --コードリスト(配列の場合はdp_cds.length
57     </elseif><else>
58         1=1
59     </else>
60     <if test="#ep.dp_cd# != null and #ep.dp_cd# != ''">
61         AND dp_cd=#ep.dp_cd#
62     </if>
63     <if test="#orderItem# != null and #orderItem# != ''">
64         ORDER BY $#orderItem#
65     <elseif test="#orderBy# != null and #orderBy# != ''">
66         $#orderBy#
67     </elseif>
68     asc
69     </else>
70     </if>
71 </sql>

```



```

<終了> CaseAccess [JUnit] C:\dev\Java\jdk1.7.0_04\bin\javaw.exe (2016/01/28 10:10:32)
2016-01-28 10:10:35,942 INFO (nj.comm.sql.CUtil:460) - 【SQL文出力: 件数抽出】
2016-01-28 10:10:35,942 INFO (nj.comm.sql.CUtil:461) -
SELECT count(*)
FROM test_department
WHERE
    dp_cd in ('A1','A2','B1')              --コードリスト(配列の場合はdp_cds.length
    AND dp_cd='B1'
2016-01-28 10:10:36,348 INFO (nj.comm.sql.CUtil:604) - 【SQL文出力: 改ページ】
2016-01-28 10:10:36,348 INFO (nj.comm.sql.CUtil:605) -
--テスト_部門の抽出
SELECT
    dp_cd                                --コード
    , dp_name                            --名称
    , dp_leader_cd                       --部長コード
    , dp_tel                             --電話
    , dp_fax                             --ファクス
    , dp_update_dt                       --更新日時
FROM test_department
WHERE
    dp_cd in ('A1','A2','B1')              --コードリスト(配列の場合はdp_cds.length
    AND dp_cd='B1'
    ORDER BY dp_update_dt
    asc LIMIT 5 OFFSET 0
2016-01-28 10:10:36,412 DEBUG (nj.comm.sql.Result:88) - [type] = 1

```

2) 下記の例は DB2 の普通 SQL 文から、自動的に件数検索 SQL 文とページ検索 SQL 文の作成です。(図 16-4 DB2 のページ SQL)

```

41
42 <sql id="select_test_department2" parameterClass="nj.test.entity.table.TestDepartment2" resultClass="nj.test.entity.table.TestDepartment">
43   --テスト_部門の抽出
44   SELECT
45     dp_cd          AS "dp_cd"          --コード
46     , dp_name      AS "dp_name"        --名称
47     , dp_leader_cd AS "dp_leader_cd"    --部長コード
48     , dp_tel       AS "dp_tel"         --電話
49     , dp_fax       AS "dp_fax"         --ファクス
50     , dp_update_dt AS "dp_update_dt"    --更新日時
51   FROM test_department
52   WHERE
53     <if test="#dp_cd# != null and #dp_cd# != ''">
54       dp_cd=#dp_cd# --コード
55     </if><elseif test="#dp_cds# != null and #dp_cds.size# > 0">
56       dp_cd in (#dp_cds#) --コードリスト(配列の場合はdp_cds.length
57     </elseif><else>
58       1=1
59     </else>
60     <if test="#ep.dp_cd# != null and #ep.dp_cd# != ''">
61       AND dp_cd=#ep.dp_cd#
62     </if>
63     <if test="#orderItem# != null and #orderItem# != ''">
64       ORDER BY "#orderItem#"
65     </if><elseif test="#orderBy# != null and #orderBy# != ''">
66       $orderBy#
67     </if><else>
68       asc
69     </else>
70   </if>
71 </sql>

```

```

<終了> CaseAccess [JUnit] C:\dev\Java\jdk1.7.0_04\bin\javaw.exe (2016/01/28 10:53:18)
2016-01-28 10:53:21,009 DEBUG (nj.commm.sql.NjPool:120) - 【接続プール状況 tree:1 busy:1】
2016-01-28 10:53:21,040 INFO (nj.commm.sql.CUtil:460) - 【SQL文出力: 件数抽出】
2016-01-28 10:53:21,040 INFO (nj.commm.sql.CUtil:461) -
SELECT count(*)
FROM test_department
WHERE
  dp_cd in ('A1','A2','B1') --コードリスト(配列の場合はdp_cds.length
  AND dp_cd='B1'

2016-01-28 10:53:21,119 INFO (nj.commm.sql.CUtil:604) - 【SQL文出力: 改ページ】
2016-01-28 10:53:21,119 INFO (nj.commm.sql.CUtil:605) -
SELECT *
FROM (
  --テスト_部門の抽出
  SELECT
    row_number() over (
      ORDER BY dp_update_dt
      asc ) as rowno,
    dp_cd          AS "dp_cd"          --コード
    , dp_name      AS "dp_name"        --名称
    , dp_leader_cd AS "dp_leader_cd"    --部長コード
    , dp_tel       AS "dp_tel"         --電話
    , dp_fax       AS "dp_fax"         --ファクス
    , dp_update_dt AS "dp_update_dt"    --更新日時
  FROM test_department
  WHERE
    dp_cd in ('A1','A2','B1') --コードリスト(配列の場合はdp_cds.length
    AND dp_cd='B1'
) tbl
WHERE rowno > 0 AND rowno <= 5
ORDER BY "dp_update_dt"
asc

2016-01-28 10:53:21,135 DEBUG (nj.commm.sql.Result:88) - [type] = -5

```


※検索及び改ページには、下記のソースを参照。

```
/njsam/src/nj/action/ReferSale.java  
/njsam/src/nj/entity/ReSalePage.java  
/njsam/WebContent/jsp/sale/sale-refer.jsp  
/njsam/WebContent/jsp2/sale/sale-refer.jsp  
/njsam/WebContent/jsp/include/pager.jsp  
/njsam/WebContent/jsp2/include/pager.jsp
```

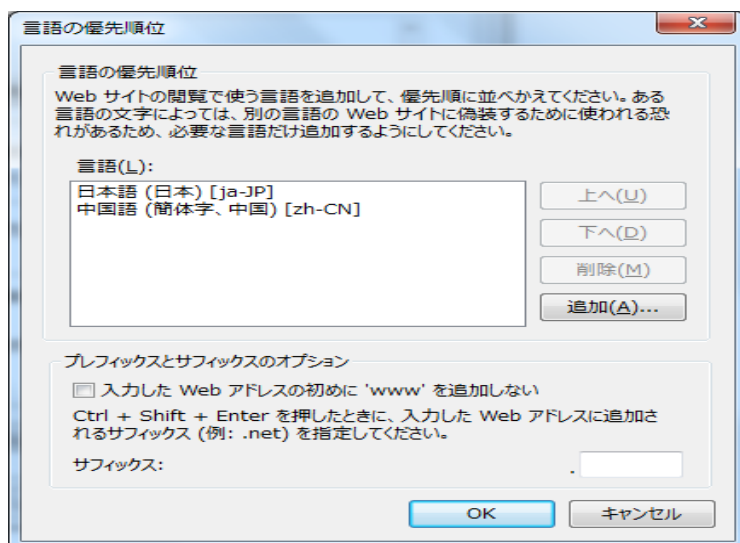
※詳細には下記の njcomm-1.00 API 仕様を参照。

解凍したファイル njweb_japanese_1.00.zip の中の njcomm-api を参照 (index.html で起動)。

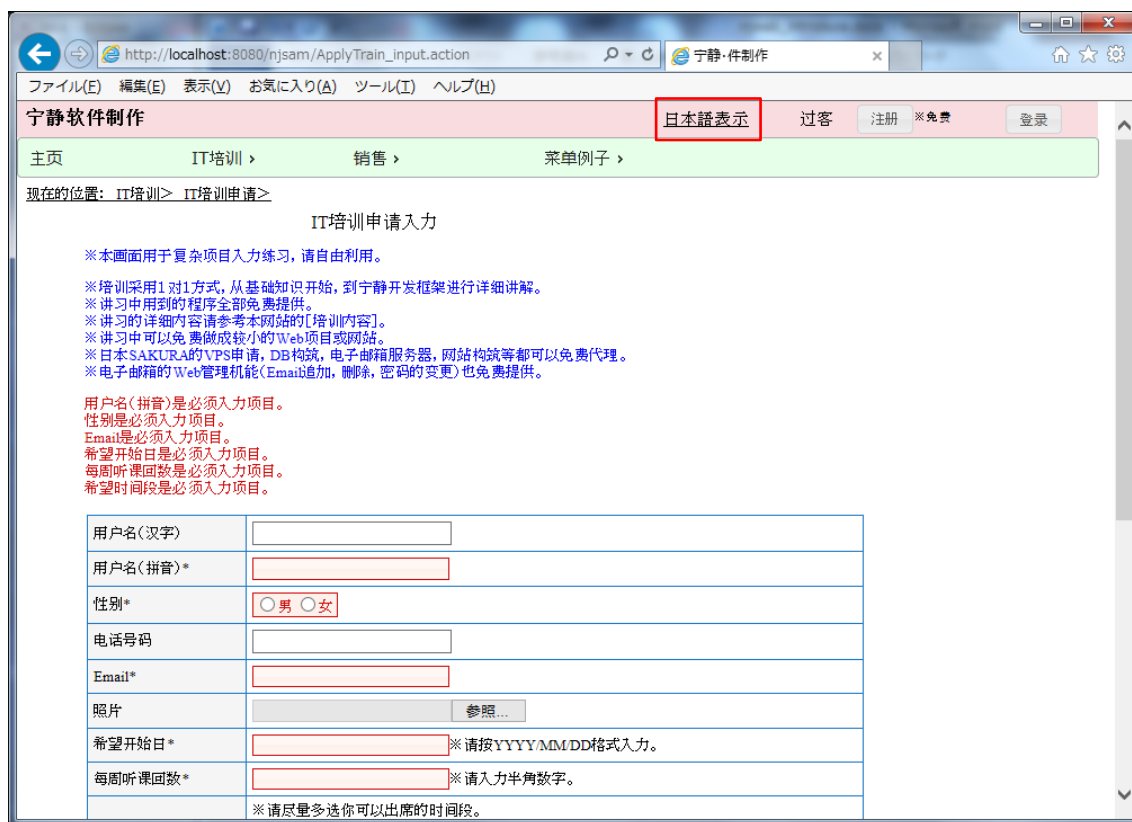
1 7. 多国言語

寧靜フレームワーク多国言語がサポートされています。もちろん、単一言語にも問題ありません。初画面の表示にはブラウザに設定された最優先言語とします。この後には画面で言語の変更ができます。一旦変更された後には、ブラウザを閉じるまでに、変更した言語を続きます。

(図 1 7 - 1 多国言語設定)



(図 17-2 多国言語切替)



※ 多国言語には、下記のソースを参照。

```

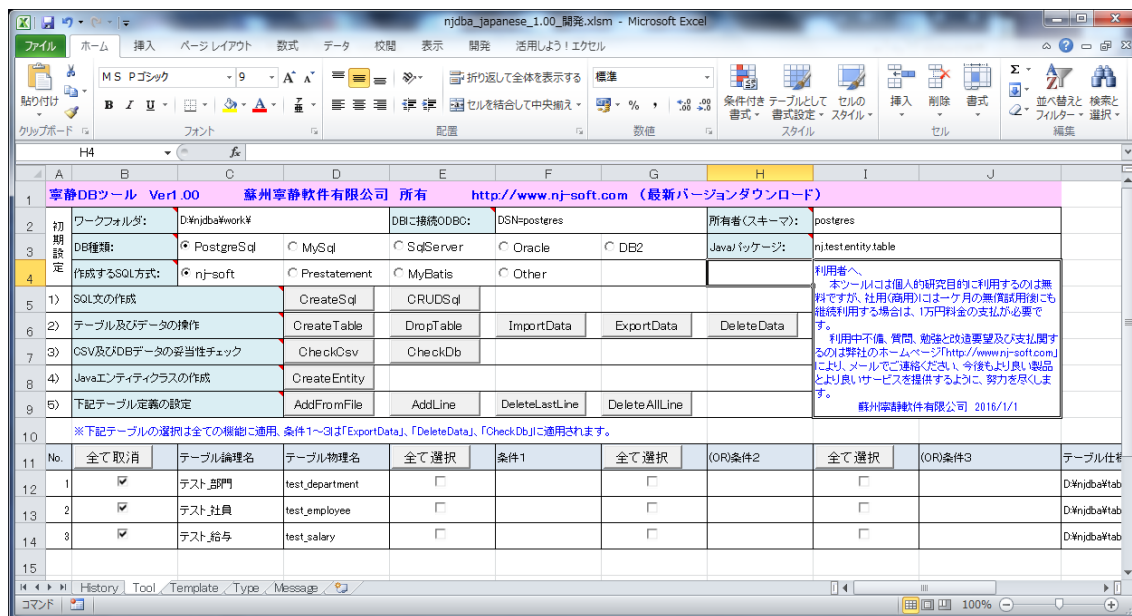
/njsam/src/nj/action/ApplyTrain. java
/njsam/src/nj/action/chk/Check. java
/njsam/src/nj/action/chk/Message. java
/njsam/WebContent/jsp/train/train-input. jsp
/njsam/WebContent/jsp2/train/train-input. jsp
/njsam/WebContent/jsp/include/messenger. jsp
/njsam/WebContent/jsp2/include/messenger. jsp
    
```

1 8. 寧靜 DB ツールとの連携

寧靜フレームワークは寧靜 DB ツールと連携されています。寧靜 DB ツール作成した動的 SQL 文や、Java エンティティクラスはそのまま使えるため、たいへん効率よくプロジェクトの開発がスムーズにできます。

寧靜 DB ツールは Excel の VBA で作成、テーブル仕様書を基づいて、DB に関する操作とデータのチェックや、SQL 文と Java エンティティクラスの自動作成ができます。このツールにも PostgreSQL、MySQL、SqlServer、Oracle、DB2 など五つのデータベースがサポートされています。

(図 1 8 - 1 寧靜 DB ツールとの連携)



※詳細には下記の寧靜 DB ツール仕様を参照。

<http://www.nj-soft.com/njweb/Download.action?baseInfo.menuId=3030&baseInfo.curPage=0>

(全文完了)

蘇州寧靜軟件有限公司 所有

<http://www.nj-soft.com>

2016年5月1日 労働の日