intra-mart WebPlatform Ver.7.2

SAStruts+S2JDBC 開発・運用環境セットアップガイド

2012/11/09 第 8 版

＜＜　変更履歴　＞＞

変更年月日	変更内容
2011/04/01	初版
2011/04/25	第 2 版 ファイルアップロード対応のため更新
2011/07/29	第 3 版 ・2.2.1 「SAStruts+S2JDBC環境構築ツール」を利用したライブラリの配置と設定について更新 ・3.2.1 「SAStruts+S2JDBC環境構築ツール」を利用したライブラリの配置と設定について更新 ・A.2.2.4 SAStrutsフィルターマッピング定義①についてLuxuryResponseWriterFilterを追加 ・A.2.2.5 SAStrutsフィルターマッピング定義②について設定を追加 ・A.2.5 <%im_path%>/doc/imart/WEB-INF/classes/creator.diconについてFunctionCreatorを追加 ・A.2.6 <%im_path%>/doc/imart/WEB-INF/classes/customizer.dicon についてFunctionCreatorを追加 ・A.2.11.1 global-exceptions について設定内容を修正 ・A.2.11.3 message-resources について設定内容を変更
2011/09/30	第 4 版 2.2 iWP 7.2.4 のリリース環境構築ツールの入手方法、及び実行方法を修正
2011/11/18	第 5 版 3.2.5 IM-Workflow利用時の設定に関する記述を追加
2012/03/16	第 6 版 ・2.2.3 IM-Workflow利用時の設定に関する記述を追加 ・1 はじめに について注意事項を追加
2012/08/03	第 7 版 ・1.3 注意事項に Action クラスの命名ルールに関する記述の追加 ・3.2.2 ルートパッケージの設定に関する記述の追加 ・A.2.3 app.dicon に imart.dicon インクロードに関する記述を追加 ・A.2.2.6 バッチ利用時の web.xml の設定に関する記述を追加 ・A.2.2.7 Web サービス利用時の web.xml の設定に関する記述を追加
2012/11/09	第 8 版 ・1.3 注意事項に Action クラスの命名ルールに関する記述を英小文字に限定するように修正

＜＜　目次　＞＞

1 はじめに

本ドキュメントは、intra-mart WebPlatform(Resin)および intra-mart DebugServer 上で Seasar プロダクトの SAStruts と S2JDBC を組合せた開発環境の構築、および運用環境の構築方法について記載しています。

※本ドキュメントでは、intra-mart WebPlatform(JBoss)および intra-mart AppFramework 上での構築については記載されておりません。

1.1 用語解説

intra-mart WebPlatform Ver.7.2	以下、**IWP** と略します。 **IWP** をインストールしたディレクトリを**<%im_path%>**と略します。
intra-mart DebugServer Ver7.2	intra-mart e Builder で利用するデバッグ専用のサーバです。 以下、**imDS** と略します。 **imDS** をインストールしたディレクトリを**<%ds_path%>**と略します。
intra-mart Server Manager	システム全体を管理するサーバです。以下、**imSM** と略します。 **imSM** をインストールしたディレクトリを**<%sm_path%>**と略します。
Application Runtime	アプリケーションの実行エンジンです。以下、**AppRuntime** と略します。 **AppRuntime** をインストールしたディレクトリを**<%app_path%>**と略します。
intra-mart e Builder Ver7.2	intra-mart 対応アプリケーション作成支援ツールです。 以下、**imEB** と略します。 **imEB** をインストールしたディレクトリを**<%eb_path%>**と略します。
Seasar2	Seasar2 は DI (Dependency Injection) と AOP (Aspect Oriented Programming) をサポートした軽量コンテナです。 詳細は http://www.seasar.org/をご覧ください。
SAStruts	Super Agile に Struts と Seasar2 で開発するためのフレームワークです。 詳細は http://sastruts.seasar.org/をご覧ください。
S2JDBC	流れるようなインターフェースと 2Way-SQL をサポートした O/R Mapping フレームワークです。Seasar2 プロダクトの一部として提供されています。 詳細は http://s2container.seasar.org/2.4/ja/s2jdbc.htmlをご覧ください。

1.2 前提条件

- intra-mart 上で SAStruts を利用する場合、SAStruts のアプリケーションは、他の Servlet と衝突しないように、スラッシュで終わるリクエストに対してマッピングします。これ以外の設定での動作検証は行っておりません。
  - [参考] web.xml での設定内容

```
<filter-mapping>
    <filter-name>routingfilter</filter-name>
    <url-pattern>*/</url-pattern>
    <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

- S2JDBC が利用するトランザクションマネージャとデータソースは、それぞれ以下を利用するように設定します。これ以外は検証しておりません。
  - トランザクションマネージャ ※jta-10.dicon で設定

```
<component class="org.seasar.extension.tx.adapter.JTAUserTransactionAdapter"/>
```

  - データソース ※jdbc.dicon で設定

```
<component name="dataSource" class="jp.co.intra_mart.framework.extension.seasar.util.AutoDetectedDataSource"/>
```

※ログイングループごとにデータソースを切り替えるための設定です。

1.3 注意事項

- intra-mart 上で SAStruts を利用する場合、S2Struts が利用できなくなります。**IWP** をインストールすると標準で S2Struts のライブラリ **s2-struts-1.3.1.jar** がインストールされ、S2Struts が利用することができました。しかし、S2Struts と SAStruts は共存できません。それぞれのライブラリが存在すると、実行時に SAStruts の RoutingFilter 内で NoSuchMethodError が発生します。これは S2Struts のライブラリ **s2-struts-1.3.1.jar** に含まれる org.seasar.struts.util.RequestUtil クラスに getPath メソッドが存在しないためです。同じクラスは SAStruts のライブラリに含まれており、そのクラスは getPath メソッドが定義されています。

- データソースに AutoDetectedDataSource を利用した場合、S2JDBC からシステムデータソースを利用することができません。

- WebPlatform において、Web サーバに Apache または IIS を利用する場合、Web サーバコネクタ経由で SAStruts のアプリケーションを実行します。Apache または IIS から WebPlatform に連携できるようにするために、routingfilter と同じ url-pattern に対してサーブレット SAStrutsGatewayServlet をマッピングします。サーブレット SAStrutsGatewayServlet は通常実行されません。存在しないアクションがリクエストされたときに実行されます。そのため、SAStrutsGatewayServlet が実行された場合、HTTP エラーコード 404 を送信します。
  - [参考] web.xml での設定内容

```
<!-- SAStruts Gateway servlet and servlet-mappings start -->
<servlet>
    <servlet-name>SAStrutsGatewayServlet</servlet-name>
    <servlet-class>jp.co.intra_mart.framework.extension.seasar.struts.servlet.SAStrutsGatewayServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>SAStrutsGatewayServlet</servlet-name>
    <url-pattern>*/</url-pattern>
</servlet-mapping>
<!-- SAStruts Gateway servlet and servlet-mappings end -->
```

- SAStruts+S2JDBC でアプリケーション開発を行う場合、URL 設計が重要になります。コンテキストパス以降の URL は、アクションクラスのパッケージ、クラス、メソッド名に直接関連します。
  たとえば、ホスト名が"localhost"、コンテキストパスが"imart"のとき、http://localhost/imart/**foo/bar/** というリクエスト URL に対してマッピングされるクラスおよびメソッドは、以下の優先順位で決まります。
  1. jp.co.sample.app.action.**foo.BarAction#index()**
  2. jp.co.sample.app.action.**foo.IndexAction#bar()**
  3. jp.co.sample.app.action.**foo.bar.IndexAction#index()**

  ※上記例は Seasar2 の SMART Deploy における Java ルートパッケージが"jp.co.sample.app"の場合

  また、以下のような Action クラスを命名すると、URL から呼び出した場合 InvocationTargetExceptio が発生し、正常に動作しなくなります。
  1. XxxYyyAction のように、Action クラス名を複数の単語でキャメル形式にした場合
  2. XYyyAction のように、Action クラスの前に大文字を複数置いた後、小文字をつなげた形式にした場合

  そのため Action クラスは、先頭を大文字、その後を小文字英字で命名する必要があります。
  (例) XxxyyyAction、XyyyAction

- SAStruts の urlPattern を利用してパスからパラメータを受け取る場合、注意が必要です。urlPattern とは、アクションメソッドに指定するアノテーション「@Execute」の属性 urlPattern です。urlPattern に指定する URL パターンの最後にはスラッシュを指定しません。しかし、リクエストする URL は必ずスラッシュで終わるようにしてください。これは、routingfilter がスラッシュで終わる URL にマッピングされているためです。たとえば、前注意事項例のアクションメソッド jp.co.sample.app.action.**foo.BarAction#index()**に対して **urlPattern** を下記のように指定されているとします。このアクションメソッドに対してパラメータ param1 と param2 にそれぞれ"value1"と"value2"を渡すには、以下の URL となります。

  http://localhost/imart/**foo/bar/value1/value2/**

  - @Execute で urlPattern = bar/{param1}/{param2} を指定

```
@Execute(validator = false, urlPattern = bar/{param1}/{param2})
public void index() {
  // 処理
}
```

2 開発環境の構築

本節では、intra-mart 上で Seasar プロダクトの SAStruts と S2JDBC を組合せた開発環境の構築方法を説明します。
構築の手順は以下の通りです。

1. デバッグサーバのインストール
2. ライブラリの配置と設定
3. e Builder のインストールと起動
4. e Builder でアプリケーションの開発

2.1 デバッグサーバのインストール

本節で説明するマシン構成は、下記の状況を想定してインストールする場合の <例> となっています。

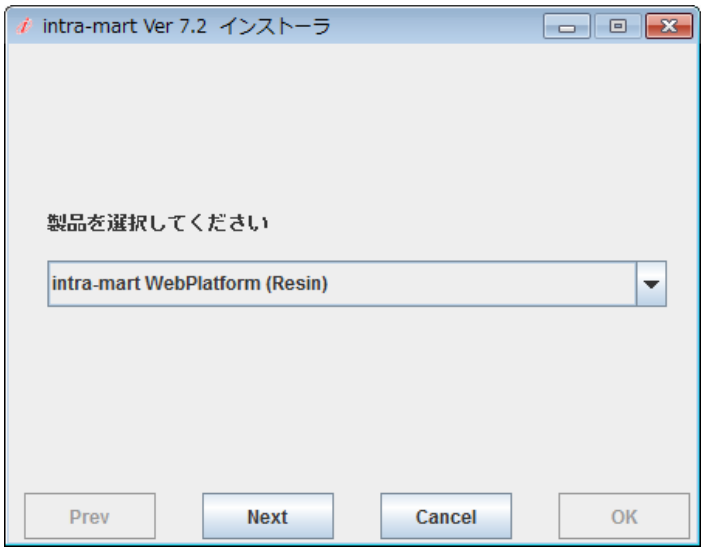
- OS : Windows
- サーバモジュールの文字コード : UTF-8
- ウェブブラウザへ送信する文字コード : UTF-8
- 製品の種類 : DebugServer

intra-mart DebugServer は、スタンドアロン環境、かつ、ウェブサーバとして intra-mart HTTP Server を利用することが前提となります。
文字コードは、運用環境を想定して設定することをお奨めいたします。
また imEB のプロジェクト作成時に、imDS をインストールした際の文字コードを指定する必要があります。

2.1.1 インストーラの起動と操作

インストーラの起動および操作は以下のように行います。本ガイドでは imDS Ver.7.2 に準拠します。

1. java.exe コマンドにパスが通っていることを確認します。
2. エクスプローラで本製品の CD-ROM があるディレクトリに移動します。
3. iwp_iaf¥install ディレクトリへ移動します。
4. **setup.jar** をダブルクリックし起動します。
   (**setup.bat** をダブルクリックしても起動できます。)
5. インストーラの画面が表示されます。



6. 画面の設定項目を以下の手順で設定して、[Next]ボタンをクリックし、最後に[OK]をクリックするとインストールが開始されます
7.

手順	入力
製品を選択してください (1:intra-mart WebPlatform (Resin)　2:intra-mart WebPlatform (JBoss) 3:intra-mart AppFramework 4:intra-mart AppFramewok(for Development 5:intra-mart DebugServer)?	5
サーバモジュールをインストール(y/n)?	y
JDK のホームディレクトリを入力してください	パスをフルパスで入力してください
インストール先を入力してください	パスをフルパスで入力してください (この場所を%ds_path%と表現します)
サーバモジュールの文字コードを選択してください (1:Windows-31J 2:Shift_JIS 3:EUC-JP 4:UTF-8)?	4（※1）
ウェブブラウザへ送信する文字コードを選択してください (1:Windows-31J 2:Shift_JIS 3:EUC-JP 4:UTF-8)?	4（※1）
このホストのアドレスを入力してください	192.168.0.2（※2）
HTTP サーバのアドレスを入力してください	192.168.0.2 (※3)
HTTP サーバのポート番号を入力してください	8081（※4）
BPMS の HTTP サーバのアドレスを入力してください	192.168.0.2 (※5)
BPMS の HTTP サーバのポート番号を入力してください	8082（※6）
Server Manager が使用するポート番号を入力してください	49152（※6）
Service Platform の ID を入力してください	APP:192.168.0.2:8081（※7）
Service Platform の初期ヒープサイズ (-Xms) [MB] (例:128)	128　（※8）
Service Platform の最大ヒープサイズ (-Xmx) [MB] (例:256)	256　（※8）
サンプルをインストール(y/n)?	y
この構成でよろしいですか(y/n)?	y

注釈（※）の詳細については、**「2.1.2 注意事項」**を参照してください。

2.1.2 注意事項

インストールの際に必要となる、入力項目の注意点を説明します。

（※1）多言語環境を構築する際は、「サーバモジュールの文字コード」、および、
「ウェブブラウザへ送信する文字コード」に**「UTF-8」**を選択してください。

（※2）Server Manager または **Service Platform** のアドレスを入力する項目では、

必ず **マシンの IP アドレス** を入力してください。
「localhost」と入力してしまうと、intra-mart は動作しません。
（例）**192.168.0.2**

（※3）HTTP サーバ のアドレスを入力する項目では、必ず **マシンの IP アドレス** を入力してください。
（例）**192.168.0.2**

（※4）HTTP サーバのポートを入力する項目では、intra-martHTTP サーバの HTTP サーバのポート番号を入力してください。

なお、Oracle11g を標準でインストールすると、ポート番号「**8080**」を使用してしまうため、
imDS を同じコンピュータにインストールした場合に、ネットワークのポート設定が衝突してサーバが起動できないことがあります。インストールを行う際は、**ポート番号が重複しないように**、設定には十分注意してください。

（※5） BPMS の HTTP サーバのアドレスを入力する項目では、BPMS に当てるサーバの IP アドレスを入力してください。

（※6）Server Manager はデフォルトではポート番号「49152」を使用します。
このポート番号は Windows Vista など、OS によっては既に使用されている場合がありますので、
その場合は、インストール後に別のポート番号を指定してください。
インストール後に、Server Manager のポート番号を変更する場合は、
<%im_path%>/conf/imart.xml を編集してください。
記入例

```
<intra-mart>
    <administration>
        <host address="192.168. 0. 1"/>
        <network port="49152" timeout="30">
. . .
</intra-mart>
```

デフォルトポート番号一覧

サーバ/OS	ポート番号
Oracle HTTP Server	8080
BPMS	8080
Windows Vista / Windows 7	49152

（※7） Service Platform の ID とは、intra-mart のサーバを**ユニークに判定するための ID** です。

（※8）ヒープサイズに関する注意事項について
初期ヒープサイズ、最大ヒープサイズについて
マニュアルに記載されている値は、あくまでサンプル値です。
この値につきましては、環境にあわせて変更していただく必要がございます。
ヒープサイズが小さすぎますと OutOfMemory が発生いたしますのでご注意ください。

2.2 ライブラリの配置と設定

本節では SAStruts+S2JDBC を利用した開発を行う上で必要となるライブラリの配置と設定について説明します。
以下の手順で行ってください。

1. 「SAStruts+S2JDBC 環境構築ツール」を利用したライブラリの配置と設定
2. データベース接続に関する設定

2.2.1 「SAStruts+S2JDBC 環境構築ツール」を利用したライブラリの配置と設定

「SAStruts+S2JDBC 環境構築ツール」を利用することで、必要なライブラリの配置や設定を行えます。本ツールでは、既存のファイルを上書きします。本ツールは、新規にインストールした Server Manager および Application Runtime に対して行うことを前提としています。

「SAStruts+S2JDBC 環境構築ツール」を実行すると以下のようにファイルが追加、更新、削除されます。

```
<%ServerManager%>/
    └─conf/
        └─message/
                [A] sastruts-validator-message_ja.properties

<%ApplicationRuntime%>/
    └─doc/
        └─imart/
            ├─j2ee/
            │  └─document/
            │      └─error/
            │          [A] s2_application_error.jsp
            │          [A] s2_system_error.jsp
            │
            └─WEB-INF/
                │  [U] struts-config.xml
                │  [U] validator-rules.xml
                │  [U] web.xml
                │
                ├─classes/
                │    [U] app.dicon
                │    [U] convention.dicon
                │    [U] creator.dicon
                │    [U] customizer.dicon
                │    [U] im_hotdeploy.dicon
                │    [U] j2ee.dicon
                │    [U] jta-10.dicon
                │    [A] s2jdbc.dicon
                │
                └─lib/
                    [A] commons-lang-2.4.jar
                    [A] geronimo-ejb_3.0_spec-1.0.jar
                    [A] geronimo-interceptor_3.0_spec-1.0.jar
                    [A] geronimo-jpa_3.0_spec-1.0.jar
                    [A] geronimo-jta_1.1_spec-1.0-tsr.jar
                    [A] geronimo-jta_1.1_spec-1.0.jar
                    [A] im_eb_components.jar
                    [D] s2-struts-1.3.1.jar
                    [A] sa-struts-1.0.4-sp8.jar
```

※ [A]・・・追加されるファイル　[U]・・・更新されるファイル　[D]・・・削除されるファイル
※更新されるファイルおよび削除されるファイルに関して、必要に応じてバックアップしてください。

2.2.1.1 「SAStruts+S2JDBC 環境構築ツール」の入手

IWP7.2 パッチ番号 04 以降のパッチを適用してください。適用することで「SAStruts+S2JDBC 環境構築ツール」がインストールされます。IWP7.2 パッチ番号 04 以降のパッチを適用できないお客様は、以下の URL より「SAStruts+S2JDBC 環境構築ツール」をダウンロードしてください。

http://www.intra-mart.jp/download/product/v72_tool/iwp_iaf/for_sastruts+s2jdbc.zip

2.2.1.2 「SAStruts+S2JDBC 環境構築ツール」の配置と編集

IWP7.2 パッチ番号 04 以降をインストールされたお客様は、2.2.1.3 にお進みください。

2.2.1.1 で URL から「SAStruts+S2JDBC 環境構築ツール」を入手した方はこの操作を行う必要があります。

1. ダウンロードした「SAStruts+S2JDBC 環境構築ツール」の圧縮ファイル for_sastruts+s2jdbc.zip を解凍
2. 解凍してできた for_sastruts+s2jdbc フォルダ内の ServerManager/conf フォルダを**<%ds_path%>**に上書きコピー
3. 解凍してできた for_sastruts+s2jdbc フォルダ内の ApplicationRuntime/bin フォルダを**<%ds_path%>**に上書きコピー
4. 以下のファイルをテキストエディタで開き、変数 IM_APP_RUNTIME_HOME に**<%ds_path%>**を設定。このとき、パスの区切り文字はスラッシュで記述してください。円記号(バックスラッシュ)を利用すると正常に動作しません。
   ファイル：**<%ds_path%>/bin/tools/build/for_sastruts+s2jdbc/setup.bat**
   例：set IM_APP_RUNTIME_HOME=C:/imart

2.2.1.3 「SAStruts+S2JDBC 環境構築ツール」の実行

下記の bat ファイルを実行してください。

ファイル：**<%ds_path%>/bin/tools/build/for_sastruts+s2jdbc/setup.bat**

2.2.2 データベース接続に関する設定

はじめに、S2JDBC のデータベース接続に関する設定を行ってください。次に e Builder からデバッグサーバを起動後に、IWP のセットアップガイド 5 節「データベース接続の設定」を参考に、データベース接続に関する設定を行ってください。

2.2.2.1 s2jdbc.dicon の設定

<%ds_path%>/doc/imart/WEB-INF/classes/s2jdbc.dicon を編集します。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="jdbc.dicon"/>
  <include path="s2jdbc-internal.dicon"/>
  <component name="jdbcManager" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
      <property name="maxRows">0</property>
      <property name="fetchSize">0</property>
      <property name="queryTimeout">0</property>
      <!-- TODO
          dialect(≒DB 方言) を1つだけ有効にしてください。
          ここでは、ログイングループデータソース用の dialect
          を1つだけ有効にします。
          ※複数のログイングループを利用する場合、各ログイン
            グループのデータソースには、同じ dialect となる
            データベースを使用してください。
      -->

      <!--
      <property name="dialect">db2390Dialect</property>
      <property name="dialect">db2400Dialect</property>
      <property name="dialect">db2Dialect</property>
      <property name="dialect">derbyDialect</property>
```

```
        <property name="dialect">firebirdDialect</property>
        <property name="dialect">h2Dialect</property>
        <property name="dialect">hsqlDialect</property>
        <property name="dialect">interbaseDialect</property>
        <property name="dialect">maxdbDialect</property>
        <property name="dialect">mssql2005Dialect</property>
        <property name="dialect">mssqlDialect</property>
        <property name="dialect">mysqlDialect</property>
        <property name="dialect">oracleDialect</property>
        <property name="dialect">postgreDialect</property>
        <property name="dialect">standardDialect</property>
        <property name="dialect">sybaseDialect</property>
        -->
  </component>
</components>
```

ログイングループデータソースで利用する DB のダイアレクトを設定してください。
DB のダイアレクトに関する詳細は下記サイトをご覧ください。
http://s2container.seasar.org/2.4/ja/s2jdbc_setup.html

2.2.3 IM-Workflow 利用時の設定

利用する imDS に IM-Workflow をインストールされている場合は jdbc.dicon のデータソース設定を以下の用に設定してください。この設定は IM-Workflow 7.2.6 より設定が可能になります。この設定によって、非同期で案件終了処理、到達処理を実行した場合でも、データソースが取得できるようになります。

```
<!-- from intra-mart -->
<component name="dataSource"
    class="jp.co.intra_mart.system.workflow.extension.seasar.ImwAutoDetectedDataSource">
</component>
```

2.3 e Builder のインストールと起動

imEB をインストールし、起動してください。
詳細は別紙「intra-mart e Builder Ver.7.2 セットアップガイド」をご覧ください。

2.4 e Builder でアプリケーションの開発

SAStruts と S2JDBC を組合せた開発では、サーバを再起動せずアプリケーションの開発を進めていくことができます。業務スケルトンを利用することでさらに「すぐ動くアプリケーション」を開発することができます。
ここでは、業務スケルトンを利用して「ブランクアプリ」を作成、実行する手順を紹介します。

また、e Builder の詳細については、e Builder 付属のマニュアルをご覧ください。e Builder のマニュアルは以下よりご覧いただけます。

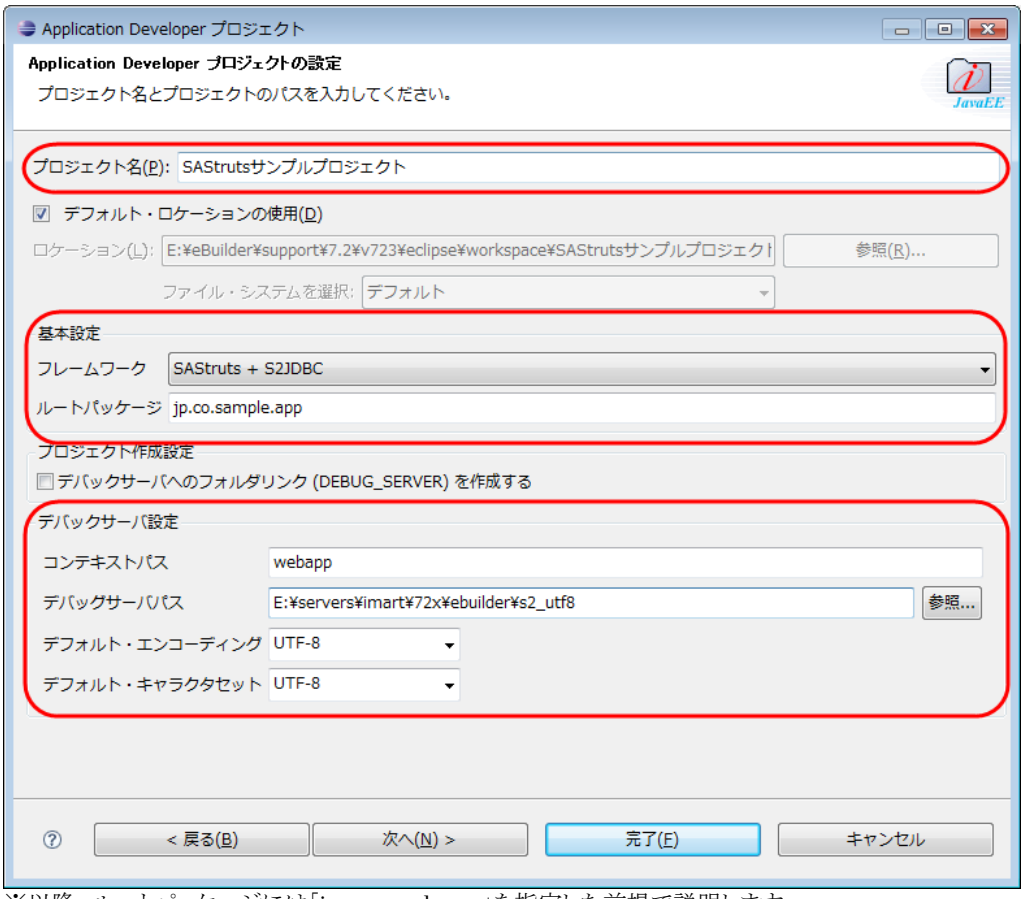
1. 起動した e Builder (eclipse) のウィンドウメニュー[ヘルプ]-[ヘルプ目次]をクリック
2. ヘルプメニュー[intra-mart e Builder Ver.7.2 ヘルプ]をクリック

2.4.1 [参考] ブランクアプリを動かしてみよう

「ブランクアプリ」とは、SAStruts の簡単なフォーム画面(JSP)とアクションクラス等の Java ファイルから構成されるアプリケーションです。「ブランクアプリ」は業務スケルトンから作成できます。以下、業務スケルトンから「ブランクアプリ」を作成し、実行するまでの手順を紹介します。

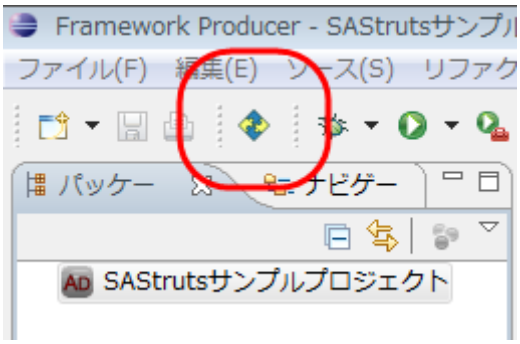
1. e Builder を起動してください。
2. Application Developer プロジェクトを作成してください。

ウィンドウメニュー[ファイル]-[新規]-[プロジェクト]から[intra-mart Application Developer]-[Application Developer プロジェクト]を選択し、下記のように必要な情報を入力してください。

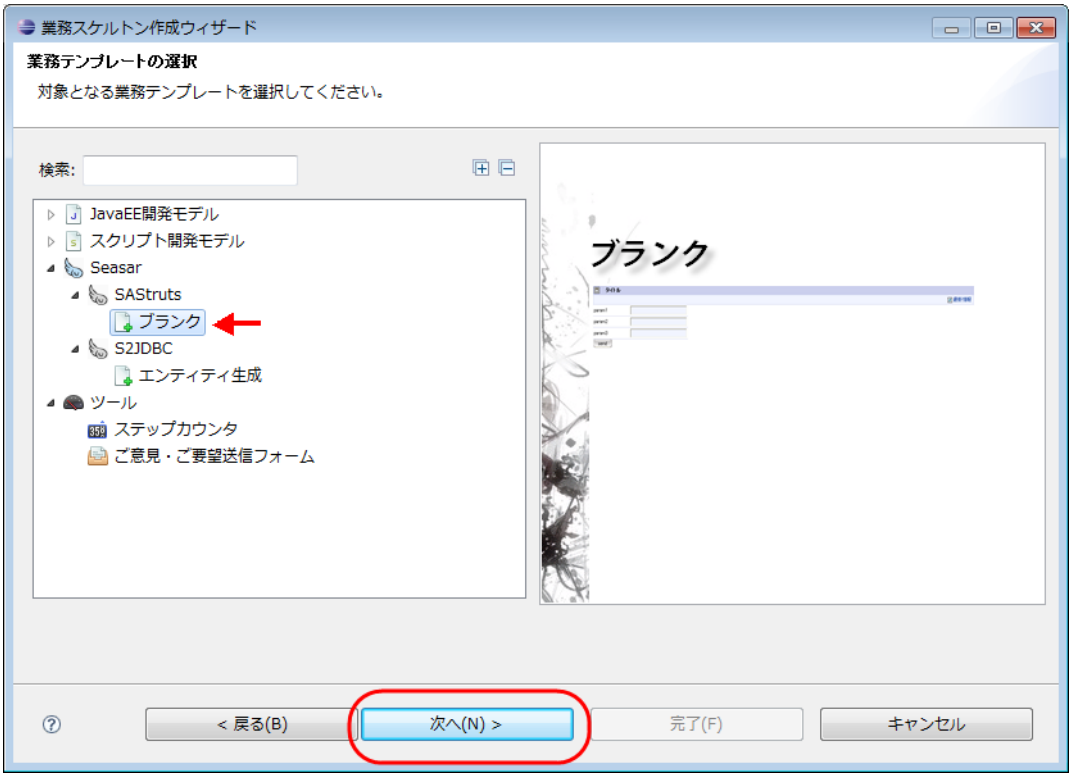


※以降、ルートパッケージには「jp.co.sample.app」を指定した前提で説明します。

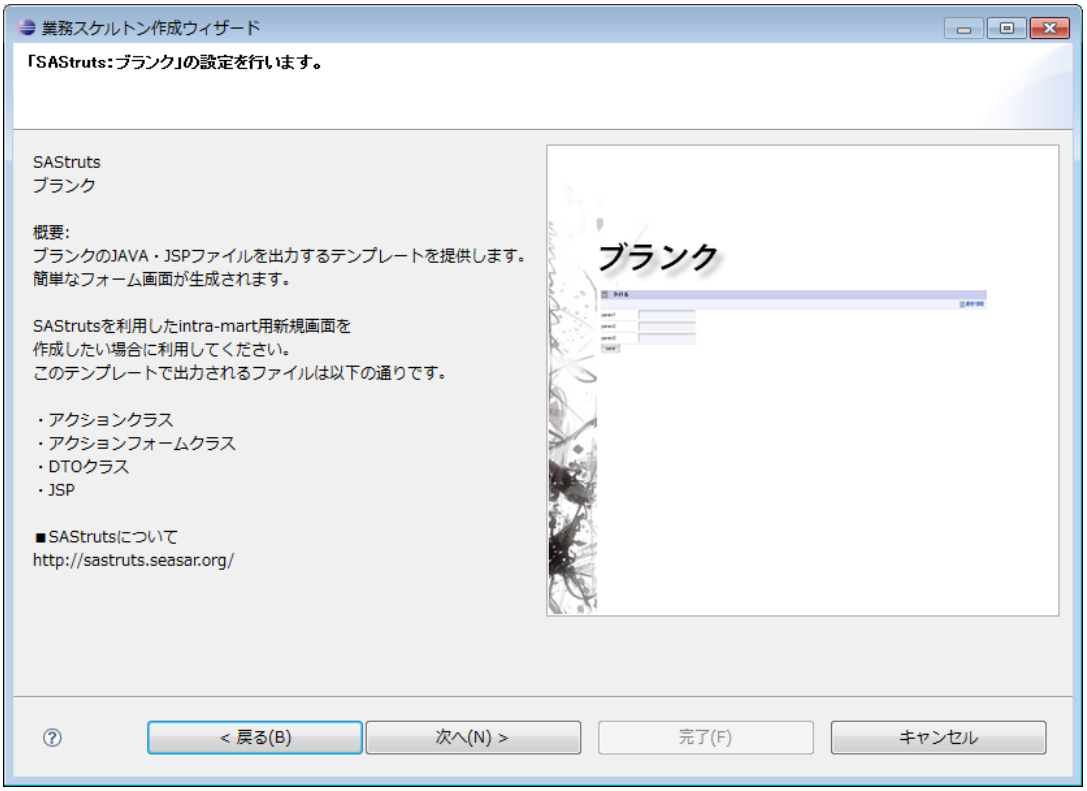
3. 作成できた Application Developer プロジェクトを選択し、下図アイコンをクリックして業務スケルトンウィザードを開いてください。



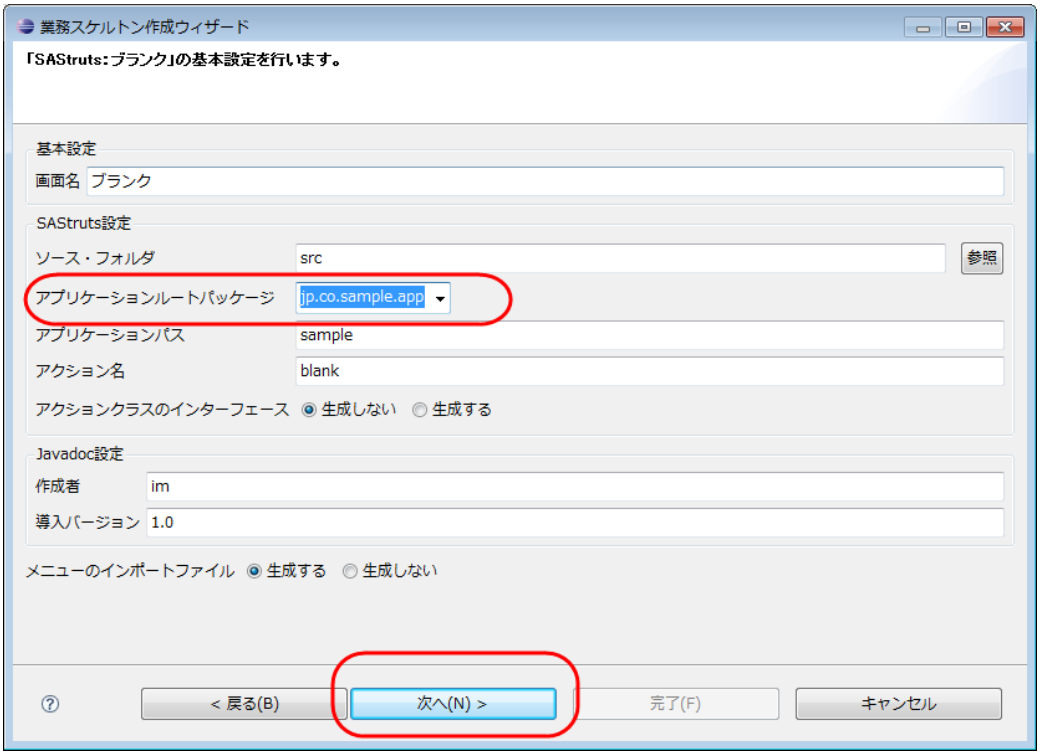
4. 起動した業務スケルトンウィザードから「業務テンプレートの選択」ページまで進み、
[Seasar]-[SAStruts]-[ブランク]を選択して次へ進んでください。



5. 説明のページが表示され、次へ進んでください。



6. 「基本設定」ページで、アプリケーションルートパッケージに「jp.co.sample.app」を選択して次へ進んでください。



※上記設定では、http://localhost/imart/**sample/blank/** でアクセスできるアプリケーションを作成できます。

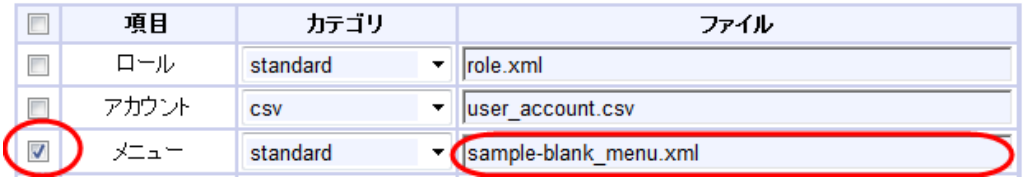
7. 「設定項目の確認」ページで設定に誤りがないか確認し、問題なければ「完了ボタン」をクリックしてください。クリックすると、以下のファイルが生成されます。

- src/jp/co/sample/app/form/sample/BlankForm.java
- src/jp/co/sample/app/dto/sample/BlankDto.java
- src/jp/co/sample/app/action/sample/BlankAction.java
- webapp/sample/blank/_index.jsp
- storage/sample-blank_menu.xml

8. デバッグサーバを起動してください。ログイングループが作成されていない場合、システム管理者でログインし、ログイングループを作成してください。またデータベース接続が行われていない場合設定してください。詳細な説明および手順は、IWP のセットアップガイド 5 節「データベース接続の設定」および 6 節「intra-mart へのログイン」をご覧ください。

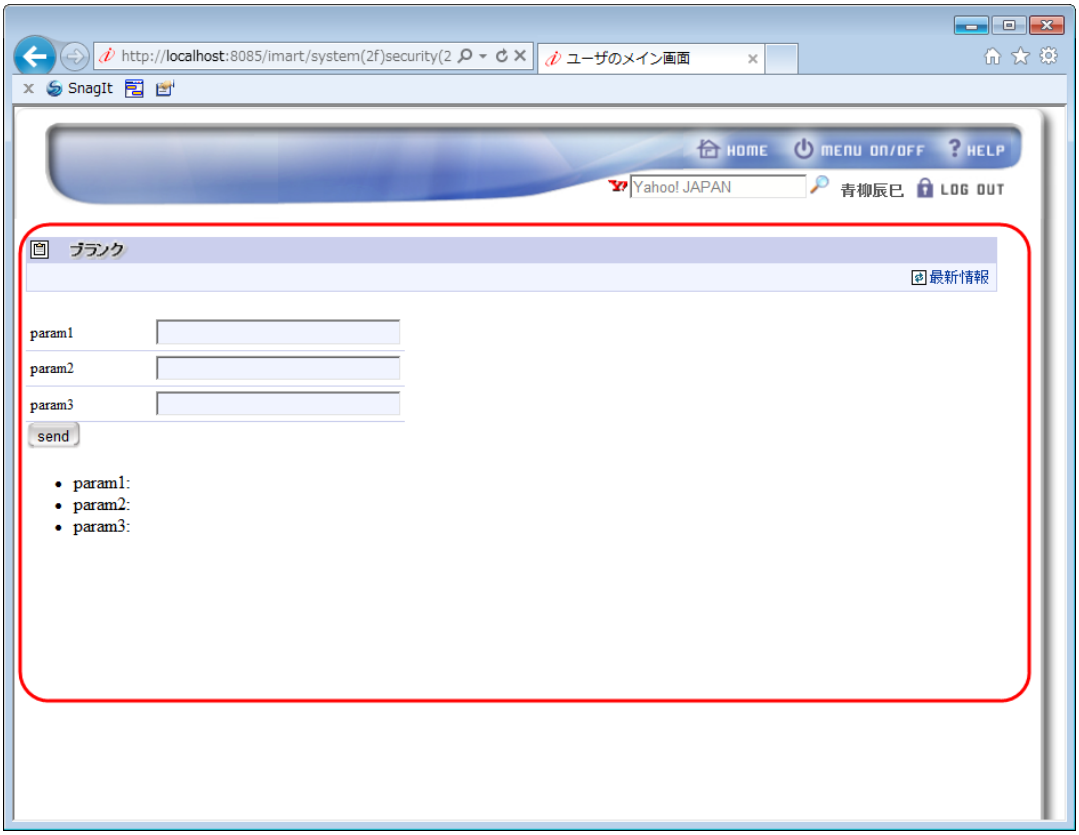
9. グループ管理者でログインし、メニューを登録してください。生成されたメニュー設定ファイル「storage/sample-blank_menu.xml」をインポートすることでメニューを登録できます。その手順は以下の通りです。

1. ファイル sample-blank_menu.xml を<%ds_path%>/storage/フォルダに配置
2. グループ管理者でログイン
3. メニュー[ログイングループ管理]-[アクセスセキュリティ情報入出力]-[インポート]をクリック
4. 項目[メニュー]をチェックし、ファイルに sample-blank_menu.xml を指定し、[インポート]ボタンをクリック



5. 「インポートが成功しました」と結果が表示されれば完了。

10. 一般ユーザでログインし、登録された「ブランク」メニューをクリックすると、以下のページが表示されます。



3 IWP 運用環境構築

本節では、2 節で開発した SAStruts+S2JDBC のアプリケーションを intra-mart WebPlatform 上で実行するための環境構築について説明します。構築の手順は以下の通りです。

1. IWP のインストール
2. インストールした Application Runtime にライブラリの配置と設定

またWebサーバコネクタを利用する場合、「3.3 [Webサーバコネクタ利用時] SAStrutsGatewayServletの設定」を行ってください。

3.1 IWP のインストール

IWP のインストールは、IWP のセットアップガイドを参考に行ってください。IWP のセットアップガイドは、「intra-mart 製品最新情報ダウンロードページ」にございます。

http://www.intra-mart.jp/download/product/index.html

3.2 Application Runtime にライブラリの配置と設定

本節では SAStruts+S2JDBC のアプリケーションを実行するためのライブラリの配置と設定について説明します。以下の手順で行ってください。

1. 「SAStruts+S2JDBC 環境構築ツール」を利用したライブラリの配置と設定
2. データベース接続に関する設定
3. hotdeploy から cooldeploy に変更

上記以外の設定に関しては、IWP のセットアップガイドをご覧ください。

3.2.1 「SAStruts+S2JDBC 環境構築ツール」を利用したライブラリの配置と設定

「SAStruts+S2JDBC 環境構築ツール」を利用することで、必要なライブラリの配置や設定を行えます。本ツールでは、既存のファイルを上書きします。本ツールは、新規にインストールした Server Manager および Application Runtime に対して行うことを前提としています。

「SAStruts+S2JDBC 環境構築ツール」を実行すると以下のようにファイルが追加、更新、削除されます。

```
<%ServerManager%>/
    └─conf/
        └─message/
                [A] sastruts-validator-message_ja.properties

<%ApplicationRuntime%>/
    └─doc/
        └─imart/
            ├─j2ee/
            │  └─document/
            │      └─error/
            │          [A] s2_application_error.jsp
            │          [A] s2_system_error.jsp
            │
            └─WEB-INF/
                │  [U] struts-config.xml
                │  [U] validator-rules.xml
                │  [U] web.xml
                │
                ├─classes/
                │    [U] app.dicon
                │    [U] convention.dicon
                │    [U] creator.dicon
                │    [U] customizer.dicon
                │    [U] im_hotdeploy.dicon
                │    [U] j2ee.dicon
                │    [U] jta-10.dicon
                │    [A] s2jdbc.dicon
                │
                └─lib/
                      [A] geronimo-ejb_3.0_spec-1.0.jar
                      [A] geronimo-interceptor_3.0_spec-1.0.jar
                      [A] geronimo-jpa_3.0_spec-1.0.jar
                      [A] geronimo-jta_1.1_spec-1.0-tsr.jar
                      [A] geronimo-jta_1.1_spec-1.0.jar
                      [A] im_eb_components.jar
                      [D] s2-struts-1.3.1.jar
                      [A] sa-struts-1.0.4-sp8.jar
```

※ [A]・・・追加されるファイル　[U]・・・更新されるファイル　[D]・・・削除されるファイル

※更新されるファイルおよび削除されるファイルに関して、必要に応じてバックアップしてください。

3.2.1.1 「SAStruts+S2JDBC 環境構築ツール」の入手

「SAStruts+S2JDBC環境構築ツール」は以下のURLより最新版をダウンロードしてください。

http://www.intra-mart.jp/download/product/v72_tool/iwp_iaf/for_sastruts+s2jdbc.zip

※本ツールは「intra-mart 製品最新情報ダウンロードページ」より入手できます。

3.2.1.2 「SAStruts+S2JDBC 環境構築ツール」の配置と実行

以下の手順で bat ファイルを実行してください。

1. ダウンロードした「SAStruts+S2JDBC 環境構築ツール」の圧縮ファイル for_sastruts+s2jdbc.zip を解凍
2. 解凍してできた for_sastruts+s2jdbc フォルダ内の ServerManager/conf フォルダを**<%sm_path%>**に上書きコピー
3. 解凍してできた for_sastruts+s2jdbc フォルダ内の ApplicationRuntime/bin フォルダを**<%app_path%>**に上書きコピー
4. 以下のファイルをテキストエディタで開き、変数 IM_APP_RUNTIME_HOME に**<%app_path%>**を設定。このとき、パスの区切り文字はスラッシュで記述してください。円記号(バックスラッシュ)を利用すると正常に動作しません。
   ファイル： **<%app_path%>/bin/tools/build/for_sastruts+s2jdbc/setup.bat**
   例： set IM_APP_RUNTIME_HOME=C:/imart/app
5. setup.bat を実行

3.2.2 ルートパッケージの設定

<%ds_path%>/doc/imart/WEB-INF/classes/convention.dicon の設定をします。

SMART deploy にルートパッケージを登録してください。

以下は「jp.co.example.app」というルートパッケージを登録する例です。

```
<!--
  <component class="org.seasar.framework.convention.impl.NamingConventionImpl">
-->
  <component class="jp.co.intra_mart.framework.extension.seasar.convention.IMNamingConventionImpl ">
      <initMethod name="addRootPackageName">
          <arg>"org.seasar.framework.container.warmdeploy"</arg>
      </initMethod>
      <initMethod name="addRootPackageName">
          <arg>"jp.co.example.app"</arg>
      </initMethod>
  </component>
```

※SMART deploy の詳細は以下のサイトをご覧ください。

http://s2container.seasar.org/2.4/ja/S2.4SmartDeploy.html

3.2.3 データベース接続に関する設定

はじめに、S2JDBC のデータベース接続に関する設定を行ってください。次に IWP を起動後に、IWP のセットアップガイド 5 節「データベース接続の設定」を参考に、データベース接続に関する設定を行ってください。

3.2.3.1 s2jdbc.dicon の設定

<%ds_path%>/doc/imart/WEB-INF/classes/s2jdbc.dicon を編集します。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="jdbc.dicon"/>
  <include path="s2jdbc-internal.dicon"/>
```

```
  <component name="jdbcManager" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
      <property name="maxRows">0</property>
      <property name="fetchSize">0</property>
      <property name="queryTimeout">0</property>
      <!-- TODO
          dialect(≒DB 方言) を1つだけ有効にしてください。
          ここでは、ログイングループデータソース用の dialect
          を1つだけ有効にします。
          ※複数のログイングループを利用する場合、各ログイン
            グループのデータソースには、同じ dialect となる
            データベースを使用してください。
      -->

      <!--
      <property name="dialect">db2390Dialect</property>
      <property name="dialect">db2400Dialect</property>
      <property name="dialect">db2Dialect</property>
      <property name="dialect">derbyDialect</property>
      <property name="dialect">firebirdDialect</property>
      <property name="dialect">h2Dialect</property>
      <property name="dialect">hsqlDialect</property>
      <property name="dialect">interbaseDialect</property>
      <property name="dialect">maxdbDialect</property>
      <property name="dialect">mssql2005Dialect</property>
      <property name="dialect">mssqlDialect</property>
      <property name="dialect">mysqlDialect</property>
      <property name="dialect">oracleDialect</property>
      <property name="dialect">postgreDialect</property>
      <property name="dialect">standardDialect</property>
      <property name="dialect">sybaseDialect</property>
      -->
  </component>
</components>
```

ログイングループデータソースで利用する DB のダイアレクトを設定してください。

DB のダイアレクトに関する詳細は下記サイトをご覧ください。

http://s2container.seasar.org/2.4/ja/s2jdbc_setup.html

3.2.4 hotdeploy から cooldeploy に変更

運用環境では必ず SMART deploy の「cooldeploy」をご利用ください。

cooldeploy を利用するためには、<%app_path%>/doc/imart/WEB-INF/classes/s2container.dicon を以下のように編集します。

■ 編集前

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="im_hotdeploy.dicon"/>
</components>
```

■ 編集後 ※インクルードするファイルを im_hotdeploy.dicon から cooldeploy.dicon に変更

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="cooldeploy.dicon"/>
</components>
```

※SMART deploy の詳細は以下のサイトをご覧ください。

http://s2container.seasar.org/2.4/ja/S2.4SmartDeploy.html

hotdeployモードの場合、hotdeployfilterがリクエスト処理を直列化するため、1つのAppRuntimeは同時に1つのリクエストしか処理できません。そのため、hotdeployモードは運用環境には適しません。
※hotdeployとcooldeployの切り替えには、env.txtを利用した方法もございます。詳細は上記サイトをご覧ください。

3.2.5 IM-Workflow 利用時の設定

利用するIWPにIM-Workflowをインストールされている場合はjdbc.diconのデータソース設定を以下の用に設定してください。この設定はIM-Workflow 7.2.6より設定が可能になります。この設定によって、非同期で案件終了処理、到達処理を実行した場合でも、データソースが取得できるようになります。

```
<!-- from intra-mart -->
<component name="dataSource"
    class="jp.co.intra_mart.system.workflow.extension.seasar.ImwAutoDetectedDataSource">
</component>
```

3.3 [Web サーバコネクタ利用時] SAStrutsGatewayServlet の設定

Web サーバコネクタを利用して、Web サーバの Apache または IIS から WebPlatform に連携できるようにするためには、routingfilter と同じ url-pattern に対してサーブレット SAStrutsGatewayServlet をマッピングしてください。
※この設定は標準で有効になっています。

■ web.xml

```
<!-- SAStruts Gateway servlet and servlet-mappings start -->
    <servlet>
        <servlet-name>SAStrutsGatewayServlet</servlet-name>
        <servlet-class>jp.co.intra_mart.framework.extension.seasar.struts.servlet.SAStrutsGatewayServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>SAStrutsGatewayServlet</servlet-name>
        <url-pattern>*/</url-pattern>
    </servlet-mapping>
<!-- SAStruts Gateway servlet and servlet-mappings end -->
```

3.3.1 [参考] SAStrutsGatewayServlet を設定する理由

resin の Web サーバコネクタが Apache/IIS から AppRuntime に処理を委譲するのは、サーブレット・マッピング(servlet-mapping)に定義された URL パターン(url-pattern)にマッチした場合です。そのため、routingfilter と同じ url-pattern に対してサーブレット SAStrutsGatewayServlet を設定しています。

付録 A　SAStruts+S2JDBC 環境構築ツール

「SAStruts+S2JDBC 環境構築ツール」の仕様について説明します。

A.1　SAStruts+S2JDBC の実行に必要なライブラリの追加と削除

IWP をインストールすると、Struts および Seasar2 関連のライブラリがインストールされます。SAStruts を利用するためには、S2Struts と SAStruts の一部クラスが競合するため、S2Struts を削除します。

- ライブラリの削除
  - **<%im_path%>/doc/imart/WEB-INF/lib/s2-struts-1.3.1.jar**

次に、SAStruts の実行に必要なライブラリを追加します。

- ライブラリの追加
  - **<%im _path%>/doc/imart/WEB-INF/lib/geronimo-ejb_3.0_spec-1.0.jar**
  - **<%im _path%>/doc/imart/WEB-INF/lib/geronimo-interceptor_3.0_spec-1.0.jar**
  - **<%im _path%>/doc/imart/WEB-INF/lib/geronimo-jpa_3.0_spec-1.0.jar**
  - **<%im _path%>/doc/imart/WEB-INF/lib/geronimo-jta_1.1_spec-1.0-tsr.jar**
  - **<%im_path%>/doc/imart/WEB-INF/lib/geronimo-jta_1.1_spec-1.0.jar**
  - **<%im_path%>/doc/imart/WEB-INF/lib/im_eb_components.jar**
  - **<%im_path%>/doc/imart/WEB-INF/lib/sa-struts-1.0.4-sp8.jar**
    ※ライブラリ sa-struts は最新版をご利用ください。

次に、Action でエラーが発生した場合に共通的に遷移させるエラーページを配置します。

- エラーページ(JSP)の配置
  - **<%im _path%>/doc/imart/j2ee/document/error/s2_application_error.jsp**
  - **<%im _path%>/doc/imart/j2ee/document/error/s2_system_error.jsp**

※本エラーページは、struts-config.xml で指定しています。

A.2　各種設定

A.2.1　<%sm_path%>/conf/message/sastruts-validator-message_ja.properties

- 以下の内容で上記ファイルを作成してください。これはバリデータから利用するメッセージ定義です。

```
# バリデーションエラー表示フォーマット
errors.header=<ul>
errors.prefix=<li>
errors.suffix=</li>
errors.footer=</ul>

# バリデータエラーメッセージ
# バインド変数:
## {0}: 論理名
## {1}: 空文字 or パラメータインデックス ※同名のパラメータが複数ある場合にそのインデックス ※1 から開始
## {2}: 入力値
## {3}: 制限値情報 ※バリデータにより異なる
## {4}: 制限値情報 ※バリデータにより異なる

# @Required
errors.required=「{0}{1}」は必ず入力してください。
message.inputerror.required=「{0}{1}」は必ず入力してください。
```

```
# @Number
errors.number=「{0}{1}」は不正な数字です。正しく入力してください。
message.inputerror.number=「{0}{1}」は不正な数字です。正しく入力してください。
# @ByteType
errors.byte=「{0}{1}」は不正な数字(byte)です。正しく入力してください。
message.inputerror.byte=「{0}{1}」は不正な数字(byte)です。正しく入力してください。
# @ShortType
errors.short=「{0}{1}」は不正な数字(short)です。正しく入力してください。
message.inputerror.short=「{0}{1}」は不正な数字(short)です。正しく入力してください。
# @IntegerType
errors.integer=「{0}{1}」は不正な数字(int)です。正しく入力してください。
message.inputerror.integer=「{0}{1}」は不正な数字(int)です。正しく入力してください。
# @LongType
errors.long=「{0}{1}」は不正な数字(long)です。正しく入力してください。
message.inputerror.long=「{0}{1}」は不正な数字(long)です。正しく入力してください。
# @FloatType
errors.float=「{0}{1}」は不正な数字(float)です。正しく入力してください。
message.inputerror.float=「{0}{1}」は不正な数字(float)です。正しく入力してください。
# @DoubleType
errors.double=「{0}{1}」は不正な数字(double)です。正しく入力してください。
message.inputerror.double=「{0}{1}」は不正な数字(double)です。正しく入力してください。

# @DateType
errors.date=「{0}{1}」は不正な日付です。正しく入力してください。
message.inputerror.date=「{0}{1}」は不正な日付です。正しく入力してください。
# @EmailType
errors.email=「{0}{1}」は不正なメールアドレスです。正しく入力してください。
message.inputerror.email=「{0}{1}」は不正なメールアドレスです。正しく入力してください。
# @UrlType
errors.url=「{0}{1}」は不正な URL です。正しく入力してください。
message.inputerror.url=「{0}{1}」は不正な URL です。正しく入力してください。
# @CreditCardType
errors.creditcard=「{0}{1}」は不正なクレジットカード番号です。正しく入力してください。
message.inputerror.creditcard=「{0}{1}」は不正なクレジットカード番号です。正しく入力してください。
# @Mask
errors.invalid=「{0}{1}」は不正なパターンです。正しく入力してください。
message.inputerror.invalid=「{0}{1}」は不正なパターンです。正しく入力してください。

# @Minlength
errors.minlength=「{0}{1}」は{3}文字以上で入力してください。
message.inputerror.minlength=「{0}{1}」は{3}文字以上で入力してください。
# @Maxlength
errors.maxlength=「{0}{1}」は{3}文字以下で入力してください。
message.inputerror.maxlength=「{0}{1}」は{3}文字以下で入力してください。
# @Minbytelength
errors.minbytelength=「{0}{1}」は{3}バイト以上で設定してください。
message.inputerror.minbytelength=「{0}{1}」は{3}バイト以上で設定してください。
# @Maxbytelength
errors.maxbytelength=「{0}{1}」は{3}バイト以下で設定してください。
message.inputerror.maxbytelength=「{0}{1}」は{3}バイト以下で設定してください。

# @IntRange, @LongRange, @FloatRange, @DoubleRange, @DateRange
errors.range=「{0}{1}」は範囲外です。({3}～{4})
message.inputerror.range=「{0}{1}」は範囲外です。({3}～{4})


#  入力項目ラベル
# labels.param1=項目 1
# labels.param2=項目 2
# labels.param3=項目 3
```

※errors.xxx と message.inputerror.xxx には同じメッセージを設定してください。errors.xxx はアノテーションのデフォルトメッセージキーです。message.inputerror.xxx は validator-rules.xml で指定するバリデータのデフォルトメッセージキーです。

A.2.2 <%im_path%>/doc/imart/WEB-INF/web.xml

A.2.2.1 S2Struts 関連の設定の削除

S2Struts ライブラリを削除したことにより、S2Struts に関する以下の設定を削除するか、コメントアウトしてください。

■ フィルター定義

```
<filter>
    <filter-name>s2strutsfilter</filter-name>
    <filter-class>org.seasar.struts.filter.S2StrutsFilter</filter-class>
</filter>
```

■ フィルターマッピング定義

```
<filter-mapping>
    <filter-name>s2strutsfilter</filter-name>
    <servlet-name>action</servlet-name>
</filter-mapping>
```

A.2.2.2 sastruts.VIEW_PREFIX (オプション)

JSP などの View 用のファイルを置くディレクトリを sastruts.VIEW_PREFIX で指定します。 ブラウザから直接アクセスされないように/WEB-INF 配下のディレクトリを指定すると良いとされています。

設定例:

■ コンテキスト初期化パラメータ

```
<context-param>
    <param-name>sastruts.VIEW_PREFIX</param-name>
    <param-value>/WEB-INF/view</param-value>
</context-param>
```

A.2.2.3 SAStruts フィルター定義

■ routingfilter を追加してください。※IWP 標準設定では s2filter および hotdeployfilter は定義済です。

```
<!-- SAStruts filters start -->
    <filter>
        <filter-name>s2filter</filter-name>
        <filter-class>org.seasar.framework.container.filter.S2ContainerFilter</filter-class>
    </filter>

    <filter>
        <filter-name>hotdeployfilter</filter-name>
        <filter-class>org.seasar.framework.container.hotdeploy.HotdeployFilter</filter-class>
    </filter>

    <filter>
        <filter-name>routingfilter</filter-name>
        <filter-class>org.seasar.struts.filter.RoutingFilter</filter-class>
        <init-param>
            <param-name>jspDirectAccess</param-name>
            <param-value>false</param-value>
        </init-param>
    </filter>
<!-- SAStruts filters end -->
```

A.2.2.4 SAStruts フィルターマッピング定義①

■ 下記フィルターマッピングを追加してください。※運用環境では hotdeployfilter を無効にしてください。

```
<!-- SAStruts filter-mappings start -->
    <filter-mapping>
        <filter-name>ResponseMonitoringFilter</filter-name>
        <url-pattern>*/</url-pattern>
```

```
    </filter-mapping>
    <filter-mapping>
        <filter-name>RequestLogFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>TransitionLogFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>ExceptionHandlingFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>RequestQueryLengthMonitoringFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>RequestControlFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>RequestCharacterEncodingFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>ResponseCharacterEncodingFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>LuxuryResponseWriterFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>HTTPContextHandlingFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>JSSPContextFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>SessionFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>URLAccessFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>IntramartLocaleFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>StrutsConnectFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>ActiveSessionFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>DuplicateLoginHandlingFilter</filter-name>
        <url-pattern>*/</url-pattern>
    </filter-mapping>
    <filter-mapping>
        <filter-name>s2filter</filter-name>
        <url-pattern>*/</url-pattern>
```

```
        <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>INCLUDE</dispatcher>
    </filter-mapping>
<!--
    <filter-mapping>
        <filter-name>hotdeployfilter</filter-name>
        <url-pattern>*/</url-pattern>
        <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>INCLUDE</dispatcher>
    </filter-mapping>
-->
    <filter-mapping>
        <filter-name>routingfilter</filter-name>
        <url-pattern>*/</url-pattern>
        <dispatcher>REQUEST</dispatcher>
    </filter-mapping>
<!-- SAStruts filter-mappings end -->
```

A.2.2.5 SAStruts フィルターマッピング定義②

■ 開発環境でホットデプロイを利用する場合、ActionServlet に対して、以下のように**青文字**の箇所を追加、設定してください。※運用環境では hotdeployfilter を無効にしてください。本設定がされていない場合、ホットデプロイ対象クラスのインスタンスをセッションから取得すると ClassCastException が発生します。

[変更前]

```
    <filter-mapping>
        <filter-name>s2filter</filter-name>
        <servlet-name>action</servlet-name>
    </filter-mapping>
```

[変更後]

```
    <filter-mapping>
        <filter-name>s2filter</filter-name>
        <servlet-name>action</servlet-name>
        <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>INCLUDE</dispatcher>
    </filter-mapping>
    <filter-mapping>
        <filter-name>hotdeployfilter</filter-name>
        <servlet-name>action</servlet-name>
        <dispatcher>REQUEST</dispatcher>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>INCLUDE</dispatcher>
    </filter-mapping>
```

A.2.2.6 SAStruts フィルターマッピング定義③

■ バッチ処理の中で Seasar のコンポーネントを利用する際は、HTTPActionEventListener サーブレットに s2filter を追加する必要があります。以下のように**青文字**の個所を追加、設定してください。

```
    <filter-mapping>
        <filter-name>JSSPContextFilter</filter-name>
        <servlet-name>HTTPActionEventListener</servlet-name>
    </filter-mapping>
    <filter-mapping>
        <filter-name>s2filter</filter-name>
        <servlet-name>HTTPActionEventListener</servlet-name>
    </filter-mapping>
```

A.2.2.7 SAStruts フィルターマッピング定義④

■ Web サービス処理の中で Seasar のコンポーネント利用する際は、AxisServlet サーブレットに s2filter を追加する必要があります。以下のように青文字の個所を追加、設定してください。

```
<filter-mapping>
    <filter-name>JSSPContextFilter</filter-name>
    <servlet-name>AxisServlet</servlet-name>
</filter-mapping>
<filter-mapping>
    <filter-name>s2filter</filter-name>
    <servlet-name>AxisServlet</servlet-name>
</filter-mapping>
```

A.2.2.8 SAStruts サーブレット定義①

■ 青文字の箇所を追加、設定してください。

```
<!-- Seasar2 servlets begin -->
<servlet>
    <servlet-name>s2servlet</servlet-name>
    <servlet-class>org.seasar.framework.container.servlet.S2ContainerServlet</servlet-class>
    <init-param>
        <param-name>configPath</param-name>
        <param-value>app.dicon</param-value>
    </init-param>
    <init-param>
        <param-name>debug</param-name>
        <param-value>false</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
</servlet>
<!-- Seasar2 servlets end -->
<!-- S2Struts servlets begin -->
<servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <!--
    <servlet-class>org.seasar.struts.servlet.S2ActionServlet</servlet-class>
    -->
    <init-param>
        <param-name>config</param-name>
        <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <init-param>
        <param-name>configFactory</param-name>
        <param-value>org.seasar.struts.config.S2ModuleConfigFactory</param-value>
    </init-param>
    <init-param>
        <param-name>debug</param-name>
        <param-value>2</param-value>
    </init-param>
    <init-param>
        <param-name>detail</param-name>
        <param-value>2</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
<!-- S2Struts servlets end -->
```

A.2.2.9 SAStruts サーブレット定義②

- Web サーバの Apache または IIS から WebPlatform に連携できるようにするために、routingfilter と同じ url-pattern に対してサーブレット SAStrutsGatewayServlet をマッピングしてください。

```
<!-- SAStruts Gateway servlet and servlet-mappings start -->
    <servlet>
        <servlet-name>SAStrutsGatewayServlet</servlet-name>
        <servlet-class>jp.co.intra_mart.framework.extension.seasar.struts.servlet.SAStrutsGatewayServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>SAStrutsGatewayServlet</servlet-name>
        <url-pattern>*/</url-pattern>
    </servlet-mapping>
<!-- SAStruts Gateway servlet and servlet-mappings end -->
```

A.2.3 <%im_path%>/doc/imart/WEB-INF/classes/app.dicon

以下の設定を追加してください。

```
<include path="convention.dicon"/>
<include path="aop.dicon"/>
<include path="s2jdbc.dicon"/>
<include path="j2ee.dicon"/>
<component name="actionMessagesThrowsInterceptor"
    class="org.seasar.struts.interceptor.ActionMessagesThrowsInterceptor"/>
```

また、SAStruts フレームワークでのアプリケーション開発の際に jp.co.intra_mart.framework.base.util.UserInfo オブジェクトを利用したい場合、app.dicon に imart.dicon をインクルードするように設定をする必要があります。

```
<include path="convention.dicon"/>
<include path="imart.dicon"/>
<include path="aop.dicon"/>
<include path="s2jdbc.dicon"/>
<include path="j2ee.dicon"/>
<component name="actionMessagesThrowsInterceptor"
    class="org.seasar.struts.interceptor.ActionMessagesThrowsInterceptor"/>
```

A.2.4 <%im_path%>/doc/imart/WEB-INF/classes/convention.dicon

A.2.4.1 [補足] NamingConventionImpl を拡張した IMNamingConventionImpl について

IMNamingConventionImpl は、HotDeploy モードのときに、必ず HotdeployClassLoader か Class を取得するように拡張したクラスです。NamingConventionImpl を利用した場合、以下の問題があり、この問題を解消するために IMNamingConventionImpl クラスを提供しています。

- JSP において、JSP が HotDeploy 対象クラスを参照するときに、そのクラスが HotDeploy 対象から外れてしまう
- Action 実装クラスで、Logic インターフェースの変数に対して Logic 実装クラスのインスタンスを自動バインドできない
- FrameworkTestCase を継承したテストケースをホットデプロイできない

A.2.5 <%im_path%>/doc/imart/WEB-INF/classes/creator.dicon

以下の設定を追加してください。

```
<component class="org.seasar.struts.creator.FormCreator"/>
<component class="jp.co.intra_mart.product.eb.seaser.creator.FunctionCreator"/>
```

※FunctionCreator は e Builder Application Producer より生成されるコードで利用する Creator です。

A.2.6 <%im_path%>/doc/imart/WEB-INF/classes/customizer.dicon

以下の設定を追加してください。

```
<component name="actionCustomizer" class="org.seasar.framework.container.customizer.CustomizerChain">
    <initMethod name="addAspectCustomizer">
        <arg>"aop.traceInterceptor"</arg>
    </initMethod>
    <initMethod name="addAspectCustomizer">
        <arg>"actionMessagesThrowsInterceptor"</arg>
    </initMethod>
    <initMethod name="addCustomizer">
        <arg>
            <component class="org.seasar.framework.container.customizer.TxAttributeCustomizer"/>
        </arg>
    </initMethod>
    <initMethod name="addCustomizer">
        <arg>
            <component class="org.seasar.struts.customizer.ActionCustomizer"/>
        </arg>
    </initMethod>
</component>

<component name="formCustomizer" class="org.seasar.framework.container.customizer.CustomizerChain">
</component>

<component name="logicCustomizer" class="org.seasar.framework.container.customizer.CustomizerChain">
    <initMethod name="addCustomizer">
        <arg>
            <component class="org.seasar.framework.container.customizer.TxAttributeCustomizer"/>
        </arg>
    </initMethod>
</component>

<component name="serviceCustomizer" class="org.seasar.framework.container.customizer.CustomizerChain">
    <initMethod name="addAspectCustomizer">
        <arg>"aop.traceInterceptor"</arg>
    </initMethod>
```

```
    <initMethod name="addCustomizer">
        <arg>
            <component class="org.seasar.framework.container.customizer.TxAttributeCustomizer"/>
        </arg>
    </initMethod>
</component>

<component name="functionCustomizer" class="org.seasar.framework.container.customizer.CustomizerChain">
    <initMethod name="addCustomizer">
        <arg>traceCustomizer</arg>
    </initMethod>
    <initMethod name="addCustomizer">
        <arg>
            <component class="org.seasar.framework.container.customizer.TxAttributeCustomizer"/>
        </arg>
    </initMethod>
</component>
```

A.2.7 <%im_path%>/doc/imart/WEB-INF/classes/im_hotdeploy.dicon (開発用)

開発環境では以下のように コメントを外し、有効にしてください。

```
<components>
  <include path="convention.dicon"/>
  <include path="customizer.dicon"/>
  <include path="creator.dicon"/>
  <component class="org.seasar.framework.container.hotdeploy.HotdeployBehavior"/>
</components>
```

A.2.8 <%im_path%>/doc/imart/WEB-INF/classes/j2ee.dicon

以下ように requiredTx に対する設定を変更してください。

```
<component name="requiredTx"
    class="org.seasar.extension.tx.RequiredInterceptor">
    <initMethod name="addRollbackRule">
        <arg>@jp.co.intra_mart.framework.system.exception.ApplicationException@class</arg>
    </initMethod>
    <initMethod name="addRollbackRule">
        <arg>@jp.co.intra_mart.framework.system.exception.SystemException@class</arg>
    </initMethod>
    <initMethod name="addRollbackRule">
        <arg>@java.lang.RuntimeException@class</arg>
    </initMethod>
</component>
```

A.2.9 <%im_path%>/doc/imart/WEB-INF/classes/jta-10.dicon

以下の設定を追加してください。

```
<component name="TransactionSynchronizationRegistry"
    class="org.seasar.extension.jta.TransactionSynchronizationRegistryImpl"/>

<!-- UserTransction と TransactionSynchronizationRegistry を利用する制限付き TransactionManager -->
<component name="TransactionManager" class="javax.transaction.TransactionManager">
    @org.seasar.extension.j2ee.JndiResourceLocator@lookup("java:comp/TransactionManager")
</component>
```

A.2.10 <%im_path%>/doc/imart/WEB-INF/classes/s2jdbc.dicon

s2jdbc.dicon は IWP に含まれておりません。以下の内容のファイルを作成・配置し、ログイングループデータソー

スで利用する DB のダイアレクトを設定してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE components PUBLIC "-//SEASAR//DTD S2Container 2.4//EN"
  "http://www.seasar.org/dtd/components24.dtd">
<components>
  <include path="jdbc.dicon"/>
  <include path="s2jdbc-internal.dicon"/>
  <component name="jdbcManager" class="org.seasar.extension.jdbc.manager.JdbcManagerImpl">
      <property name="maxRows">0</property>
      <property name="fetchSize">0</property>
      <property name="queryTimeout">0</property>
      <!-- TODO
          dialect(≒DB 方言) を1つだけ有効にしてください。
          ここでは、ログイングループデータソース用の dialect
          を1つだけ有効にします。
          ※複数のログイングループを利用する場合、各ログイン
            グループのデータソースには、同じ dialect となる
            データベースを使用してください。
      -->

      <!--
      <property name="dialect">db2390Dialect</property>
      <property name="dialect">db2400Dialect</property>
      <property name="dialect">db2Dialect</property>
      <property name="dialect">derbyDialect</property>
      <property name="dialect">firebirdDialect</property>
      <property name="dialect">h2Dialect</property>
      <property name="dialect">hsqlDialect</property>
      <property name="dialect">interbaseDialect</property>
      <property name="dialect">maxdbDialect</property>
      <property name="dialect">mssql2005Dialect</property>
      <property name="dialect">mssqlDialect</property>
      <property name="dialect">mysqlDialect</property>
      <property name="dialect">oracleDialect</property>
      <property name="dialect">postgreDialect</property>
      <property name="dialect">standardDialect</property>
      <property name="dialect">sybaseDialect</property>
      -->
  </component>
</components>
```

DB のダイアレクト関する詳細は以下のサイトをご覧ください。

http://s2container.seasar.org/2.4/ja/s2jdbc_setup.html

A.2.11 <%im_path%>/doc/imart/WEB-INF/struts-config.xml

A.2.11.1 global-exceptions

■ Action でのエラーをハンドリングするためのエラーハンドラーを設定してください。

```
<global-exceptions>
    <!-- ApplicationException -->
    <exception key="errors.application"
        handler="jp.co.intra_mart.framework.extension.seasar.struts.exception.ApplicationRuntimeExceptionHandler"
        type="jp.co.intra_mart.framework.extension.seasar.struts.exception.ApplicationRuntimeException"
        path="/j2ee/document/error/s2_application_error.jsp" />
    <!-- SystemException -->
    <exception key="errors.system"
        handler="jp.co.intra_mart.framework.extension.seasar.struts.exception.SystemRuntimeExceptionHandler"
        type="jp.co.intra_mart.framework.extension.seasar.struts.exception.SystemRuntimeException"
        path="/j2ee/document/error/s2_system_error.jsp" />
    <!-- Exception -->
    <exception key="errors.exception"
        handler="jp.co.intra_mart.framework.extension.seasar.struts.exception.SystemRuntimeExceptionHandler"
        type="java.lang.Exception"
        path="/j2ee/document/error/s2_system_error.jsp" />
```

```
</global-exceptions>
```

A.2.11.2 controller

- maxFileSize および bufferSize はそれぞれ要件に合わせて適切な数値を設定してください。
- processorClass および multipartClass にそれぞれ拡張クラスを指定してください。

```
<controller
    maxFileSize="1024K"
    bufferSize="1024"
    processorClass="jp.co.intra_mart.framework.extension.seasar.struts.action.IMS2RequestProcessor"
    multipartClass="jp.co.intra_mart.framework.extension.seasar.struts.upload.IMS2MultipartRequestHandler"/>
```

A.2.11.2.1 [補足] IMS2RequestProcessor および IMS2MultipartRequestHandler について

processorClass および multipartClass は、それぞれ次の目的で標準クラスを拡張しています。ファイルアップロード機能において、以下の例のように、<input type="file"/> タグの name 属性値が同じリクエストパラメータでも ActionForm にマッピングされるように拡張しています。このとき ActionForm のマッピング先のフィールドは、FormFile インターフェースのリスト型(List<FormFile>)にもマッピングできるように拡張しています。

- JSP

```
<s:form>
   <input type="file" name="foo" />
   <input type="file" name="foo" />
   <input type="file" name="bar" />
</s:form>
```

- ActionForm

```
public FormFile[] foo;
public List<FormFile> bar;
```

参考:クラス構成は以下の通りです。

- IMS2RequestProcessor は org.seasar.struts.action.S2RequestProcessor を継承しています。
- IMS2MultipartRequestHandler は org.seasar.struts.upload.S2MultipartRequestHandler を継承しています。

A.2.11.3 message-resources

- 下記設定を追加してください。

```
<message-resources parameter=""
    factory="jp.co.intra_mart.framework.extension.seasar.struts.util.IMPropertyMessageResourcesFactory"/>
```

上記設定により、メッセージの取得は jp.co.intra_mart.foundation.security.message.MessageManager を利用します。この API は、メッセージを**<%sm_path%>**/conf/message/フォルダにあるプロパティファイルから取得します。

A.2.11.4 plug-in

- 下記設定を追加してください。

```
<plug-in className="org.seasar.struts.validator.S2ValidatorPlugIn">
    <set-property
      property="pathnames"
      value="/WEB-INF/validator-rules.xml"/>
</plug-in>
```

A.2.12 <%im_path%>/doc/imart/WEB-INF/validator-rules.xml

「SAStruts+S2JDBC 環境構築ツール」に含まれる validator-rules.xml では、バリデータメソッドに S2FieldChecks を拡張した IMFieldChecks を設定しています。IMFieldChecks は、以下の拡張を行っています。

- ログインユーザのロケールに応じたエラーメッセージを取得
- 複数パラメータの場合にメッセージに対してパラメータのインデックス(1～)を渡せるように拡張

この機能を利用するためには、struts-config.xmlでmessage-resourcesにIMPropertyMessageResourcesFactoryを指定してください。詳細は「A.2.11.3 message-resources」をご覧ください。

intra-mart WebPlatform Ver.7.2
SAStruts+S2JDBC 開発・運用環境セットアップガイド

2012/11/09 第 8 版

TEL: 03-5549-2821
FAX: 03-5549-2816
E-MAIL: info@intra-mart.jp
URL: http://www.intra-mart.jp/