[Mediator] アプリケーション構築手順

1. Mediator アプリケーションファイル配置構成

1.1. Mediator アプリケーションファイル配置位置

<<ディレクトリ構成>>

```
Web アプリケーションコンテキストルート(例：～¥myserver¥sample)
 |
 ├─error
 |      CertifyError.jsp
 |      DBLockError.jsp      共通エラー画面 (システム毎に対応が必要)
 |      SystemError.jsp
 |      TimeOutError.jsp
 ├─js
 |      common.js            共通 JavaScript
 |
 ├─WEB-INF
 |      |   mediator-taglib.tld      Mediator タグライブラリ定義ファイル
 |      |   struts-bean.tld
 |      |   struts-config.xml
 |      |   struts-html.tld
 |      |   struts-logic.tld
 |      |   web.xml
 |      ├─classes
 |      |      ApplicationResources.properties
 |      |   config.properties
 |      |   db.properties
 |      |   log.properties
 |      |   system.properties          Mediator プロパティ定義ファイル
 |      |   action-forward.xml         Mediator XML 定義ファイル
 |      |   bean-mapping.xml
 |      |   logic-mapping.xml
 |      |   logic-plugin.xml
 |      |   logic-sequence.xml
 |      |   validate-mapping.xml
 |      |   format-text-mapping.xml
 |      └─lib
 |          jads-dbadapter-10.jar
 |          jads-mediator-10-10-11.jar      Mediator ライブラリ
 |          jads-sharelib-10.jar
 |
 |            activation.jar
 |            classes12.jar
 |            struts.jar                    commons-beanutils.jar
 |            commons-collections.jar       commons-digester.jar
 |            commons-fileupload.jar        commons-lang.jar
 |            commons-logging.jar           commons-validator.jar
 |            CustomValidatorRule.jar
 |            jaxp-api.jar                  jdbc2_0-stdext.jar
 |            mail.jar
 |            sax.jar                       xalan.jar
 |            xerces.jar                    xercesImpl.jar
 └─src
```

＊上記は Struts1.1 をベースにしたものです。

＊Java ソース類は src 配下としますが、src 配置位置については任意です。

＊src 配下のパッケージ構成については、省略します。

1.2. Mediator 各ディレクトリ・ファイル説明

1.2.1. error ディレクトリ

共通のエラー画面を配置します。

(注) 但し、各システム開発毎に対応した画面レイアウトに修正する必要があります。

JSP ファイル名	画面説明
CertifyError.jsp	不正な画面遷移時のエラー画面
DBLockError.jsp	データベースアクセス時の排他エラー画面
SystemError.jsp	システムエラー発生時のエラー画面
TimeOutError.jsp	セッションタイムアウト発生時のエラー画面

1.2.2. js ディレクトリ

共通の JavaScript(common.js)を配置します。

Mediator では、画面からの要求(request)を ActionPath、command パラメータ、mode パラメータをキーに業務を実行、次画面遷移を行ないます。Common .js では、主にこの ActionPath、command、mode を操作する関数(function)を実装しています。

1.2.3. web.xml ファイル(WEB-INF 配下)

Mediator では、web.xml に下記の設定をして下さい。

① サーブレットマッピング

② エンコーディングフィルタークラスを使用する場合(Servlet API2.3 以降使用時)

```
<web-app>
----- 抜粋 -----
  <filter>
    <filter-name>encodingFilter</filter-name>
    <filter-class>com.e_jads.sharelib.web.filter.EncodingFilter</filter-class>
    <init-param>
      <param-name>encoding</param-name>
      <param-value>MS932</param-value>     ・・・・・・  [文字コードの指定はシステム毎に設定して下さい]
    </init-param>
  </filter>

----- 抜粋 -----

    <servlet-name>action</servlet-name>
    <servlet-class>com.e_jads.mediator.base.servlet.ServletController</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
<web-app>
```

1.2.4. struts-config.xml ファイル(WEB-INF 配下)

Mediator では、struts-config.xml に下記の設定をして下さい。

① grobal-forwards 設定

Mediator では、この設定を元に例外発生時共通のエラー画面にフォワードします。

1.3.2.config.properties と関係していますので、設定時には注意してください。

② action タグの type 属性

Mediator では、struts の Action クラスを継承した Action クラスの実装は行ないません。

Action クラスは、com.e_jads.mediator.base.logic.LogicDistributor に統合されますので、

type 属性には、下記のように設定して下さい。

但し、struts の Action クラスを継承した Action クラスを作成し、type 属性に設定することで、

struts 上で動作は致します。

```
<struts-config>
  <!-- ========== Data Source Configuration =============================== -->
  <!-- ========== Form Bean Definitions =================================== -->
  <!-- ========== Global Forward Definitions ============================== -->
----- 抜粋 -----
  <global-forwards type="org.apache.struts.action.ActionForward">
    <!-- ログイン画面 -->
    <forward name="system.login" path="/index.html"/>
    <!-- システムエラー画面 -->
    <forward name="system.error" path="/error/SystemError.jsp"/>
    <!-- セッションエラー画面 -->
    <forward name="session.error" path="/error/TimeOutError.jsp"/>
    <!-- 認証エラー画面 -->
    <forward name="certify.error" path="/error/CertifyError.jsp"/>
    <!-- サブ画面用セッションエラー画面 -->
    <forward name="subsession.error" path="/error/SubTimeOutError.jsp"/>
    <!-- サブ画面用システムエラー画面 -->
    <forward name="subsystem.error" path="/error/SubSystemError.jsp"/>
    <!-- サブ画面用認証エラー画面 -->
    <forward name="subcertify.error" path="/error/SubCertifyError.jsp"/>
  </global-forwards>
  <!-- ========== Action Mapping Definitions ============================== -->
  <action-mappings>

----- 抜粋 -----

    <!-- ログイン -->
    <action path="/Login" type="com.e_jads.mediator.base.logic.LogicDistributor" name="LoginForm"
                          scope="request" validate="true" input="/Login.jsp">
      <forward name="success" path="/Login.jsp"/>
      <forward name="menu" path="/MainMenu.do"/>
      <forward name="sample" path="/Sample.do"/>
    </action>
  </action-mappings>
```

1.2.5. mediator-taglib.tld ファイル(WEB-INF 配下)

Mediator が提供するタグライブラリの定義ファイルです。

1.2.6. ApplicationResource.properties (classes 配下)

Mediator では、下記の設定をして下さい。

① システムエラー共通メッセージ

Mediator では、この設定を元に例外発生時共通のエラー画面に設定されたエラーメッセージを表示します。

1.3.2.config.properties と関係していますので、設定時には注意してください。

```
error.system.exception=<li><font size="2">アプリケーション起動時にエラーが発生しました。</font></li>
error.exception=<li><font size="2">例外が発生しました。</font></li>
error.base.exception=<li><font size="2">致命的なエラーが発生しました。</font></li>
error.db.exception=<li><font size="2">ＤＢ操作時に致命的なエラーが発生しました。</font></li>
error.parameter.exception=<li><font size="2">不正なパラメータが渡されています。</font></li>
error.application.exception=<li><font size="2">アプリケーション実行時にエラーが発生しました。/font></li>
db.error.lock=<li><font size="2">テーブルをロックする際にエラーが発生しました。</font></li>
db.error.duplicate=<li><font size="2">一意制約違反が発生しました。</font></li>
db.error.notfound=<li><font size="2">該当マスタデータがありません。</font></li>
db.error.parameter=<li><font size="2">ＤＢ操作時に致命的なエラーが発生しました。</font></li>
```

1.2.7. Mediator プロパティファイル・XML ファイル(classes 配下)

●Mediator では以下のプロパティファイルを使用します。

properties ファイル名	概要
system.properties	システム基本設定プロパティ
config.properties	アプリケーション動作設定プロパティ
db.properties （注）	DBコネクション動作設定プロパティ
dbadapter.properties （注）	JNDI－DB接続設定プロパティ
log.properties	ログ出力動作設定プロパティ

＊詳細はプロパティファイル仕様書を参照

(注) システムで使用する DB 接続方法により使い分けてください。

●Mediator では以下のＸＭＬファイルを使用してアプリケーションの動作を定義します。

XML ファイル名	概要
action-forward.xml (注)	アクション遷移定義ファイル
bean-mapping.xml (注)	メソッドキャッシュ定義ファイル
logic-mapping.xml (注)	ロジック動作定義ファイル
logic-plugin.xml (注)	プラグイン定義ファイル
logic-sequence.xml (注)	シーケンス実行定義ファイル
validate-mapping.xml (注)	バリデート動作定義ファイル
format-text-mapping.xml	テキスト入出力定義ファイル

＊詳細は XML 定義ファイル仕様書を参照

(注)ファイルの配置(ファイルパス)は config.properties にて定義します。

1.3. Mediator 各種定義ファイル説明

1.3.1. system.properties(システム基本設定プロパティ)

```
#-------------------------------------------------------------------------
#システム設定ファイル
#-------------------------------------------------------------------------
#######状態遷移パラメータ名定義#######
#[任意]機能名のパラメータ属性名
command.name=command                    ..............................  ①
#[任意]動作モード名のパラメータ属性名
mode.name=mode                          ..............................  ②
#######メソッド定義#######
#[必須]Actionメソッドのprefix
action.method.prefix=logic              ..............................  ③
#[任意]Checkerメソッドのprefix
checker.method.prefix=check             ..............................  ④
```

① Request の機能コードパラメータ名設定(必須)

リクエストから機能コードを取得するためのパラメータ名です。

② Request の動作モードパラメータ名設定(必須)

リクエストから動作モードを取得するためのパラメータ名です。

③ ユーザーが定義するロジックメソッドの Prefix 設定(任意)

業務ロジックとして実行されるメソッドの Prefix です。デフォルト値は‘logic’です。

Mediator では、この設定により Prefix+機能コード+動作モードから業務ロジックとして

実行されるメソッドを判別し、メソッドを実行しています。

例) Prefix:logic

機能コード:command = Login

動作モード:mode = Init

実行されるロジックのメソッド名:logicLoginInit

④ ユーザーが定義する入力検証処理メソッドの Prefix 設定(任意)

validator で使用されるチェックメソッドの Prefix です。メソッドオブジェクトのキャッシュ目的で使用されます。

デフォルト値は‘check’です。

この設定は config.properties の file.validate.name が定義されている場合のみ有効となります。

＊その他、設定項目が定義されています(任意)が、詳細はプロパティファイル仕様書を参照下さい。

1.3.2. config.properties(アプリケーション動作設定プロパティ)

```
#-----------------------------------------------------------------------
#  システム設定ファイル
#-----------------------------------------------------------------------
#######メッセージリソース定義#######                    ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・  ①
#[任意]無効なセッションのエラーのメッセージリソース名
res.session.invalid=error.session.invalid
#[任意]不正なログインのエラーのメッセージリソース名
res.session.notlogin=error.session.notlogin

#[必須]アプリケーション起動時のException例外のメッセージリソース名
res.system.exception=error.system.exception
#[必須]アプリケーション実行結果のException例外のメッセージリソース名
res.exception=error.exception
#[必須]アプリケーション実行結果の致命的エラーのメッセージリソース名
res.base.exception=error.base.exception
#[必須]アプリケーション実行結果のＤＢエラーのメッセージリソース名
res.db.exception=error.db.exception
#[必須]アプリケーション実行結果の不正パラメータエラーのメッセージリソース名
res.parameter.exception=error.parameter.exception
#[必須]アプリケーション実行結果のその他エラーのメッセージリソース名
res.application.exception=error.application.exception
#[必須]アプリケーション実行結果のテーブル／レコードロック中のメッセージリソース名
res.db.lock.exception=db.error.lock
#[必須]アプリケーション実行結果の一意制約違反のメッセージリソース名
res.db.duplicate.exception=db.error.duplicate
#[必須]アプリケーション実行結果の該当データなしのメッセージリソース名
res.db.notfound.exception=db.error.notfound
#[必須]アプリケーション実行結果のパラメータエラーのメッセージリソース名
res.db.parameter.exception=db.error.parameter

#######画面遷移フォワード名定義#######                  ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・  ②
#[任意]ログイン画面のフォワード名
fwd.system.login=system.login
#[任意]セッションタイムアウトなどのセッションエラー発生時のフォワード名
fwd.session.invalid=session.error
#[任意]サブ画面上でセッションタイムアウトなどのセッションエラー発生時のフォワード名
fwd.session.invalid./CustomReference=subsession.error
fwd.session.invalid./ItemReference=subsession.error
fwd.session.invalid./DispatchDetail=subsession.error
fwd.session.invalid./CalcCharter=subsession.error
fwd.session.invalid./CalcAttach=subsession.error
fwd.session.invalid./CalcExcess=subsession.error
#[必須]システムエラー発生時のフォワード名
fwd.system.error=system.error
#[必須]サブ画面上でシステムエラー発生時のフォワード名
fwd.system.error./CustomReference=subsystem.error
fwd.system.error./ItemReference=subsystem.error
fwd.system.error./DispatchDetail=subsystem.error
fwd.system.error./CalcCharter=subsystem.error
fwd.system.error./CalcAttach=subsystem.error
fwd.system.error./CalcExcess=subsystem.error

#######ＤＢ接続環境定義#######                          ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・  ③
#[任意]データーベース接続プール名の指定
db.pool.name=main
#[任意]データーベース接続タイムアウトの指定
db.connection.timeout=60
#######Action動作定義#######                           ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・  ④
#[任意]Action->Action遷移の定義ファイル名
file.forward.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/action-forward.xml
#[任意]Logicマッピング定義ファイル名
file.logic.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/logic-mapping.xml
#[任意]Logicシーケンス定義ファイル名
file.sequence.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/logic-sequence.xml
```

```
#######ActionForm動作定義#######                                    ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ⑤
#[任意]validateマッピング定義ファイル名
file.validate.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/validate-mapping.xml
#[任意]validate定義ファイル格納ディレクトリ
directory.validate.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/

#######DataBean動作定義#######                                      ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ⑥
#[任意]beanマッピング定義ファイル
file.mapping.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/bean-mapping.xml

#######Plugin定義#######                                            ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ⑦
#[任意]Pluginマッピング定義ファイル
file.plugin.name=file:///Tomcat-webapps/nuwh_phase2_info07/nuwh/WEB-INF/classes/logic-plugin.xml

#######運用モード定義#######                                        ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ⑧
# service=true  ※true:運用モード  false:開発モード  デフォルトはfalse
# エラーレポート等の出力設定を行います。
service=true
```

① メッセージリソース定義(各設定は全て必須)

Mediator フレームワーク内、または業務ロジック側から各種 Exception が通知(throw)された場合のエラーメッセージを設定します。

また、設定されているそれぞれの Key は ApplicationResources.properties に登録されているリソース名となります。

② 画面遷移フォワード名定義(必須設定あり)

Mediator フレームワーク内、または業務ロジック側から各種 Exception が通知(throw)された場合のフォワード名を設定します。

また、設定されているそれぞれの Key は struts-config.xml の GLOBAL_ERROR に登録されているフォワード名となります。

fwd.system.error ・・・ 全ての例外が発生した時に遷移するフォワード名です。この設定は必須になります。

《画面遷移フォワード名特殊設定》

フォワード先の設定には、アクションパス、機能コード(command)、動作モード(mode)を組み合わせることにより任意にフォワード先の指定が可能です。

【パターン 1】fwd.session.invalid=session.error

このパターンでは、session.error にフォワードします。

【パターン 2】fwd.session.invalid./XXXX=session.errorA

このパターンでは、./ XXXX パス内では session.errorA にフォワードします。

【パターン 3】fwd.session.invalid./XXXX.Execute=session.errorB

このパターンでは、/ XXXX パス内で、command が Execute のときのみ session3.errorB にフォワードします。

【パターン4】fwd.session.invalid./XXXX.Execute.Init=session.errorC

このパターンでは、/ XXXX パス内で、command が Execute、mode が Init のときのみ session3.errorC にフォワードします。

③ DB接続環境定義

システムで使用するデータベースの接続環境を設定します。

設定キーワード	設定	説明
db.pool.name	任意	システムで使用するデータベースコネクションプール名を設定します。 com.e_jads.sharelib.dbconnection.GenericConnectionPool を使用する場合、 この設定が db.properties に登録されていなければなりません。
db.jndi.name	任意	システムで使用するデータベース JNDI 接続名を設定します。 システムで JNDI 接続による DB アクセスを行なう場合、db.properties は作成せず、 dbadapter.properties を作成して下さい。
db.connection.timeout	任意	データベース接続時のプール獲得最大待ち時間(秒単位)を設定します。(0～n の数字) この設定は、db.pool.name が設定されている場合に有効です。 com.e_jads.sharelib.dbconnection.GenericConnectionPool を使用する場合、 設定時間を超えた場合、XSDBException が発生します。

④ Action 動作定義

⑤ ActionForm 動作定義

⑥ DataBean 動作定義

⑦ Plugin 定義

Mediator で使用される各種アプリケーション動作定義 XML ファイルのファイル配置位置を設定します。

(1.2.7.参照)

⑧ 運用モード定義

Mediator では独自のエラーレポート画面出力機能を提供しています。

開発モード(false)では、エラー発生時に各種定義情報、マッピング定義情報等詳細な情報を

画面出力します。システム開発時、運用時で切替えてください。

1.3.3. db.properties(データベース接続設定プロパティ)

```
#-------------------------------------------------------------------------
#  ＤＢ接続プール設定ファイル
#-------------------------------------------------------------------------

#[必須]ＪＤＢＣドライバ名                    ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ①
drivers=oracle.jdbc.driver.OracleDriver
#[必須]ＵＲＬ
main.url=jdbc:oracle:oci:@ORACLE.XXX.XX.XX   ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ②

#[任意]ユーザー名                            ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ③
main.user=SAMPLE
#[任意]パスワード
main.password=SAMPLE

#[必須]最小接続プール数                      ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ ④
main.minpool=1
#[必須]最大接続プール数
main.maxpool=3
```

① JDBC ドライバークラス名(必須)

使用する JDBC ドライバークラスの完全修飾クラス名を設定します。

複数登録する場合はカンマ区切りで設定します。

② JDBC ドライバークラス名(必須)

JDBC ドライバーへの接続 URL 名を設定します。

{プール定義名}.url で定義してください。

プール定義名は config.properties で設定されたプール名です。

③ ユーザー名、パスワード(任意)

データベース接続ユーザー/パスワードを設定します。

{プール定義名}.user、{プール定義名}.password で定義してください。

プール定義名は config.properties で設定されたプール名です。

④ 接続プール数(必須)

最小プール数 ・・・・ 初期プールとして確保するプール数を指定します。

0 であれば初期プールは作成されません。

最大プール数 ・・・・同時接続を許可する接続数を指定します。この値を超える同時接続は許可されません。

尚、0 を設定した場合には制限なしとなります。

{プール定義名}.minpool、{プール定義名}.maxpool で定義してください。

プール定義名は config.properties で設定されたプール名です。

1.3.4. log.properties(ログ出力動作設定プロパティ)

```
#-----------------------------------------------------------------------
#  ログ出力設定ファイル
#-----------------------------------------------------------------------
#[必須]実行トレースファイルの出力ディレクトリ／ファイル名                ・・・・・・・・ ①
trace.file.name=D:/log/nuwh/nuwh_trace_info07.log
#[必須]実行トレースファイルの最大サイズ：２０ＭＢ（ＫＢ単位）            ・・・・・・・・ ②
trace.file.size=20480
#[必須]実行トレースファイルの出力モード                                  ・・・・・・・・ ③
#  none          =       出力なし
#  release       =       WARNING+ERRORのみ出力
#  info          =       INFO+WARNING+ERRORのみ出力
#  debug         =       DEBUG+INFO+WARNING+ERRORのみ出力
#  all           =       DUMP+DEBUG+INFO+WARNING+ERRORの全て出力
trace.file.mode=all
#trace.file.mode=release
```

① トレースログファイル出力ディレクトリ/ファイル名

トレース情報を出力するファイル名を指定します。

物理パスでルートからディレクトリ／ファイル名を指定します。

② トレースログファイルサイズ指定

トレース情報の最大ファイルサイズを Kbyte 単位で指定します。

③ トレースログ出力モード

ログの出力モードを設定します。

出力モード	ログ区分
none	ログ出力は行われません
release	[ERROR] [WARNING]
info	[ERROR] [WARNING] [INFO]
trans	[ERROR][WARNING] [INFO][TRANS]
debug	[ERROR] [WARNING] [INFO] [TRANS] [DEBUG]
all	[ERROR] [WARNING] [INFO] [TRANS] [DEBUG] [DUMP]

ログ区分	区分説明
[ERROR]	Exception が発生した場合などの致命的なエラー
[WARNING]	動作には影響がない警告レベルのエラー
[INFO]	動作を示すメッセージ
[TRANS]	トランザクション実行メッセージ
[DEBUG]	任意のデバッグ出力
[DUMP]	オブジェクトデータのダンプトレース出力

＊その他エラー情報ファイル出力設定、エラー通知メール設定、ガベージコレクタ起動設定が行なえます。詳細はプロパティファイル仕様書を参照下さい。

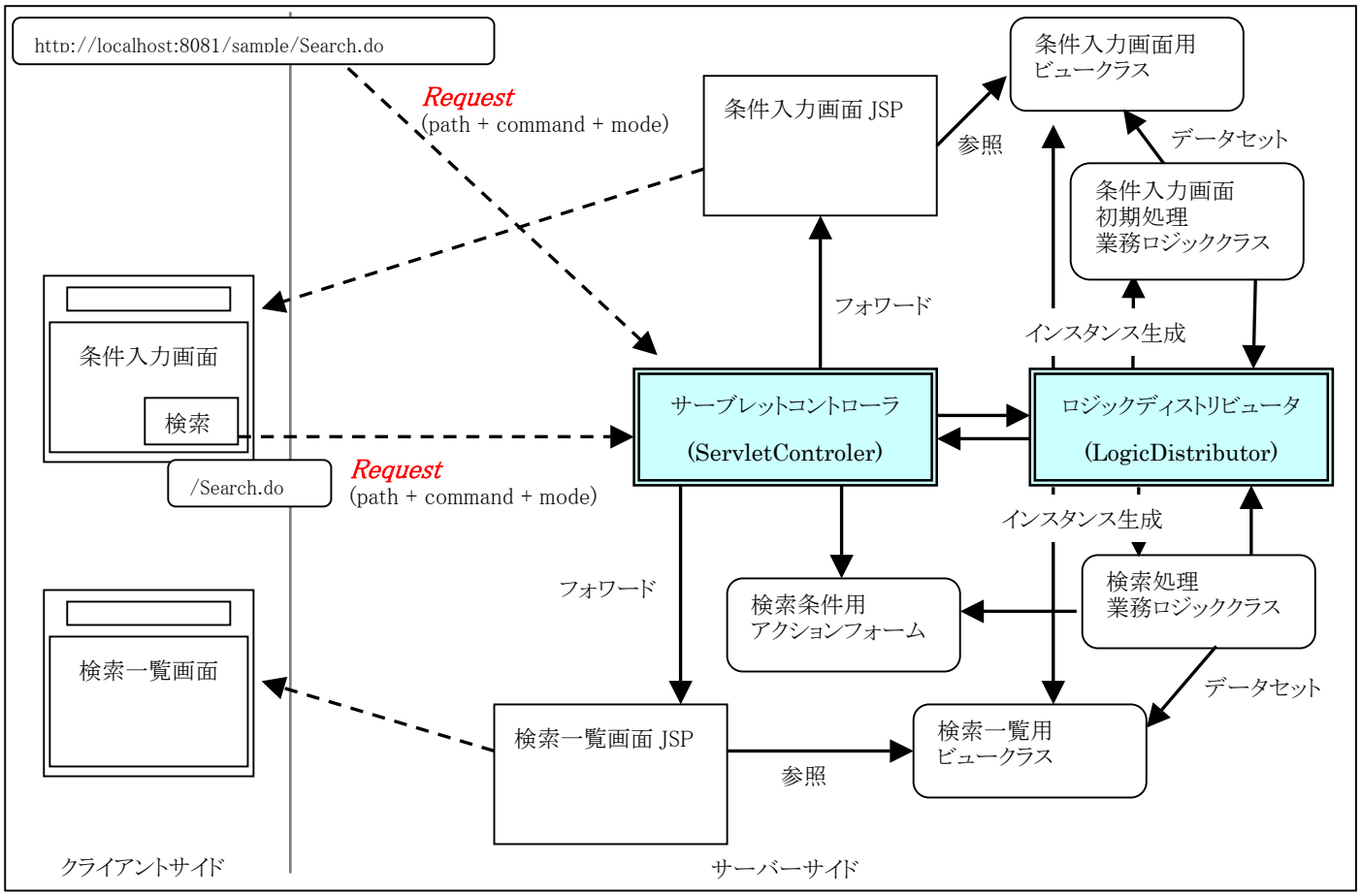
1.4. Mediator 各種アプリケーション動作定義ファイル説明

Mediator では、Web アプリケーション(ブラウザ)からの要求(Request)に対し、Struts の struts-config.xml による
アクション遷移マッピングに加え、その際に実行される業務ロジッククラス、メソッドを Mediator が提供する
独自の定義ファイル上で制御します。
加えて、業務ロジックを順番に呼出すことも独自の定義ファイル上に定義することにより、業務ロジックの
複数実行が定義ファイル上で可能になります。
また、業務ロジック処理後の画面遷移もこの定義ファイル上に論理定義します。
struts-config.xml では、使用するアクションフォーム Bean およびそのスコープ、input パス、validate の指定、
独自の定義ファイルに論理定義されたフォワード名に対する物理パスを定義します。
Mediator では定義上、画面遷移に対する制御も独自の定義ファイル上で制御しているということになります。

1.4.1. logic-mapping.xml (ロジック動作定義ファイル)

logic-mapping.xml では、Web アプリケーション(ブラウザ)からの要求(Request)に対して実行される
業務ロジッククラス、メソッドの定義ならびに業務処理終了後のフォワード先の定義を行ないます。

▼ 図1



Mediator コンポーネント　作成する 業務コンポーネント　作成する JSP

図1では、検索条件入力画面を呼出し、検索条件入力後、「検索ボタン」を押下し検索結果一覧画面を呼出すアプリケーションです。(全体イメージとして、各コンポーネントの関係の概要を記しています。)
このアプリケーションのロジック動作定義ファイルは下記のようになります。

logic-mapping.xml

```
----- 抜粋 -----

<mapping path="/Search" plugin="adminCertify" >
  <!-- 検索条件入力画面 -->
  <logic command="Condition" mode="Disp" class="sample.logic.SearchLogic"
         def-forward="init" viewClass=" sample.view.SearchView" viewScope="session"
         viewContinue="false"/>
  <!-- 検索実行 -->
  <logic command="Search" mode="" class=" sample.logic.SearchLogic"
         def-forward="list" viewClass=" sample.view.SearchListView" viewScope="session"
         viewContinue="true"/>
</mapping>
----- 抜粋 -----
<!-- ▼ 登録 ▼ -->
  <mapping path="/Entry" plugin="salonOfficeCertify">
    <logic command="Entry" mode="" sequence="EntrySearch" viewClass="sample.EntryView"
                              viewScope="request" viewContinue="false" def-forward="result"/>
</mapping>
```

struts-config.xml

```
----- 抜粋 -----
<action path="/Search" type="com.e_jads.mediator.base.logic.LogicDistributor" name="SearchForm"
                                      scope="request" validate="true" input="/sample/error.jsp">
  <forward name="init" path="/sample/SearchCondition.jsp"/>
  <forward name="list" path="/sample/SearchList.jsp"/>
</action>
----- 抜粋 -----
<action path="/Entry" type="com.e_jads.mediator.base.logic.LogicDistributor" name="EntryForm"
                      scope="request" validate="true" input="/sample/EntryError.jsp">
  <forward name="list" path="/sample/SearchList.jsp"/>
</action>
```

1. 検索条件入力画面呼出し

クライアントサイドより、URL：http://localhost:8081/sample/Search.do、command：Condition、mode：Disp の Request(要求)があった場合、

① path、command、mode から logic-mapping 定義情報を取得します。

```
<mapping path="/Search" plugin="adminCertify" >
  <!-- 検索条件入力画面 -->
  <logic command="Condition" mode="Disp" class="sample.logic.SearchLogic"
         def-forward="init" viewClass=" sample.view.SearchView" viewScope="session"
         viewContinue="false"/>
```

属性	名称	説明/備考
class	実行する業務ロジッククラス	Sample.logic.SearchLogic クラス
-	実行する業務ロジックメソッド	Sample.logic.SearchLogic#logicConditionDisp() [業務ロジックメソッド Prefix]+[command]+[mode]からメソッドを特定し、メソッドを呼出します。
viewClass	実行する業務ロジックメソッドに引き渡される(引数となる) View クラス	Sample.view.SearchView クラス
viewScorp	View クラスの格納スコープ	指定されたスコープに、このオブジェクトを格納します。
viewContinue	View クラスの継続指定	業務ロジック実行時に、View クラスの初期化を行ないます。
def-forward	フォワード名設定	業務ロジック処理完了後の遷移先を指定します。

上記設定により、下記のメソッドが呼出され、画面遷移します。

[実行メソッド]

Sample.logic.SearchLogic#logicConditionDisp(SampleForm form,ActionErrors errors,SearchVeiw objView)

[フォワード先]

/sample/SearchCondition.jsp

2. 検索処理実行

クライアントサイドより、「検索ボタン」を押下し、URL：http://localhost:8081/sample/Search.do、command：Search の Request(要求)があった場合、

```
<mapping path="/Search" plugin="adminCertify" >
  <!-- 検索実行 -->
  <logic command="Search" mode="" class="sample.logic.SearchLogic"
         def-forward="list" viewClass="sample.view.SearchListView" viewScope="session"
         viewContinue="true"/>
```

上記設定により、下記のメソッドが呼出され、画面遷移します。

[実行メソッド]

Sample.logic.SearchLogic#logicSearch(SampleForm form,ActionErrors errors,SearchListVeiw objView)

[フォワード先]

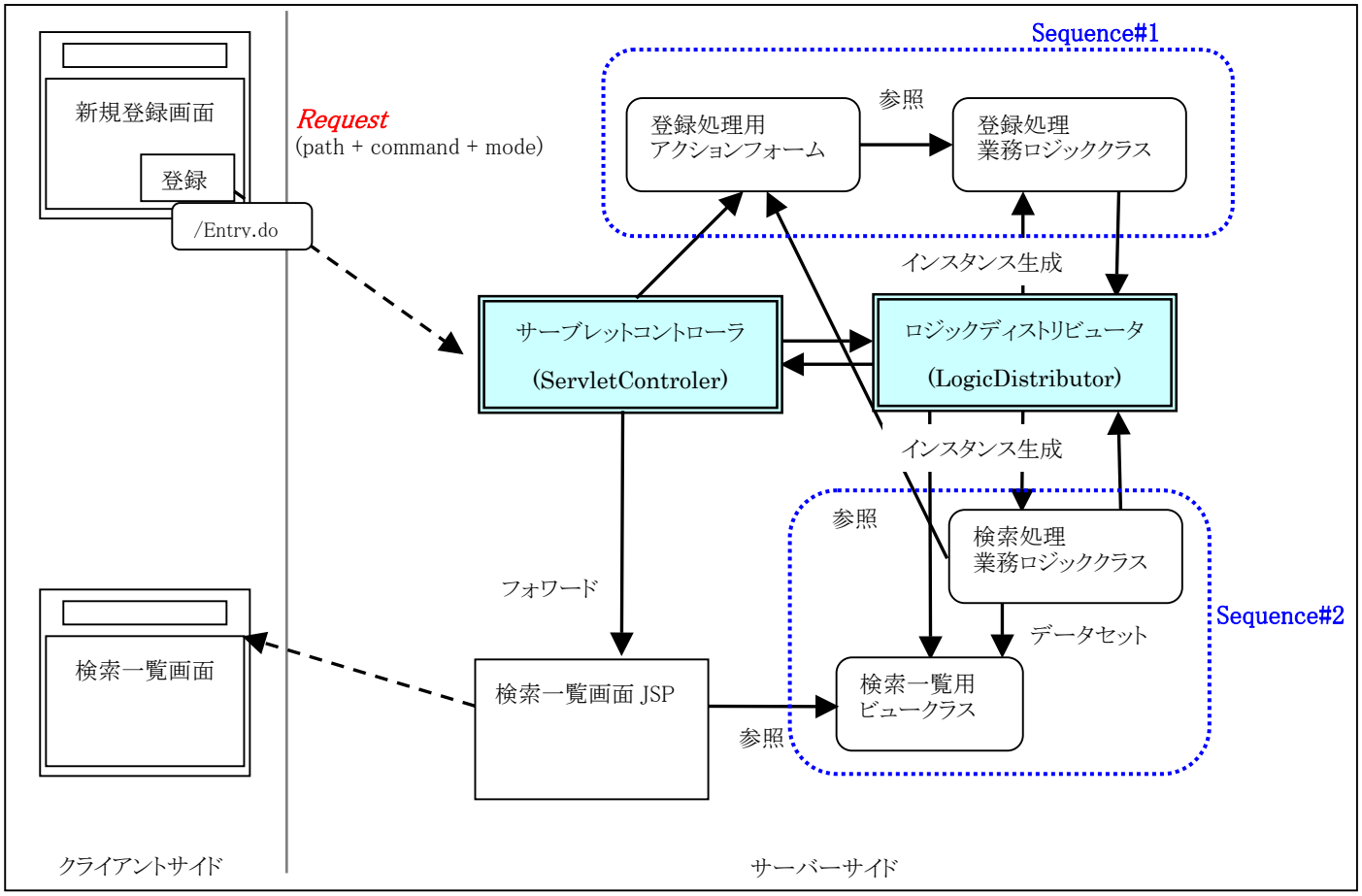
/sample/ SearchList.jsp

1.4.2. logic-sequence.xml (ロジックシーケンス実行定義ファイル)

logic-sequence.xml では、Web アプリケーション(ブラウザ)からの要求(Request)に対して実行される
業務ロジッククラス、メソッドを実行順に複数定義することにより、業務ロジックの複数呼び出しを行ないます。

図1のアプリケーションに登録画面を追加し、登録処理後一覧データを再表示する機能を追加します。

▼ 図 2



1. 検索条件入力画面呼出し

クライアントサイドより、URL：http://localhost:8081/sample/Entry.do、command：Entry、mode：
の Request(要求)があった場合、

① path、command、mode から logic-mapping 定義情報を取得します。

```
<!-- ▼ 登録 ▼ -->
  <mapping path="/Entry" plugin="salonOfficeCertify">
    <logic command="Entry" mode="" sequence="EntrySearch" viewClass="sample.SearchListView"
                           viewScope="request" viewContinue="false" def-forward="list"/>
</mapping>
```

このマッピングでは、sequence 属性が設定されているため、この sequence 名に該当するシーケンス定義を
logic-sequence.xml より取得します。

属性	名称	説明/備考

sequence	ロジックシーケンス名	シーケンス定義されたロジックを実行する場合に設定します。

logic-sequence.xml

```
<logic-sequence>
----- 抜粋 -----

<sequence name=" EntrySearch">
  <call command="Entry" mode="" class="sample.logic.EntryLogic"
                                formUse="true" viewUse="false" errorAbort="true"/>
  <call command="Search" mode="" class="sample.logic.SearchLogic"
                                                formUse="true" viewUse="true"/>
</sequence>

----- 抜粋 -----
</logic-sequence>
```

<<sequence エレメント>>

属性	名称	説明/備考
name	ロジックシーケンス名	業務ロジックのシーケンス実行する定義名です。

<<call エレメント>>

属性	名称	説明/備考
name	ロジックシーケンス名	業務ロジックのシーケンス実行する定義名です。
command	機能コード	メソッド生成時に使用されます。
mode	動 作 モード	メソッド生成時に使用されます。
class	実行ロジッククラス名	
formUse	ActionForm 使用フラグ	実行される業務ロジックに ActionForm を引き渡す場合に設定します。 引き渡される ActionForm は、struts-config.xml の name 属性で 定義したクラスです。
viewUse	View 使用フラグ	実行される業務ロジックに View クラスを引き渡す場合に設定します。 引き渡される View は、logic-mapping.xml の viewClass 属性で 定義したクラスです。
errorAbort	エラー実行中止フラグ	シーケンス実行中に ActionErrors にエラーが格納された時点で処理を 中止する場合に設定します。

上記設定により、下記のメソッドが順に呼出され、画面遷移します。

[実行メソッド]

1. Sample.logic.EntryLogic#logicEntry(EntryForm form,ActionErrors errors,)
2. Sample.logic.EntryLogic#logicSearch(EntryForm form,ActionErrors errors,SearchListView objView)

[フォワード先]

/sample/ SearchList.jsp

1.4.3. action-forward.xml (アクション遷移定義ファイル)

Mediator では、Web アプリケーション(ブラウザ)からの要求(Request)に対して、業務ロジックメソッドを実行しますが、業務ロジック処理完了後、フォワード先としてアクションを呼出したい場合の定義ファイルです。

logic-mapping.xml

```
<logic-mapping>
----- 抜粋 -----
<mapping path="/Sample" plugin="clientCertify">
  <logic command="Search" mode="Init" class="sample.logic.SampleLogic"        ........  ①
                viewClass="sample.SampleView" viewScope="request" viewContinue="false" />
  <case id="1" act-forward="sample1"/>                                        ........  ②
  <case id="2" act-forward="sample2"/>                                        ........  ③
</mapping>

----- 抜粋 -----
<mapping path="/ NextSample1" plugin="clientCertify">
  <logic command="Entry1" mode="Execute1" class="sample.logic.NextSampleLogic1"  ....  ④
                viewClass="sample.NextSampleView1" viewScope="request"
                          viewContinue="false" def-forward="success"/>
</mapping>
<mapping path="/ NextSample2" plugin="clientCertify">
  <logic command="Entry2" mode="Execute2" class=" sample.logic.NextSampleLogic2" ....  ⑤
                viewClass="sample.NextSampleView2" viewScope="request"
                          viewContinue="false" def-forward=" success "/>
</mapping>
```

<< case エレメント >>

属性	名称	説明/備考
id	実行ステータス ID	業務ロジック実行後のステータスを設定します。
act-forward	アクションフォワード名	別の業務ロジックにフォワードする場合に設定します。

action-forward.xml

```
<action-forward>
----- 抜粋 -----
    <logic class="sample.logic.SampleLogic" command="Search" mode="Init">  ........  ⑥
      <forward name="sample1" command="Entry1" mode="Execute1"/>           ........  ⑦
      <forward name="sample2" command="Entry2" mode="Execute2"/>           ........  ⑧
    </logic>
----- 抜粋 -----
</action-forward>
```

<< logic エレメント >>

属性	名称	説明/備考
class	業務ロジッククラス名	遷移元の業務ロジッククラス名を設定します。
command	機能コード	遷移元の機能コードを設定します。
mode	動作モード	遷移元の動作モードを設定します。

struts-config.xml

```
<action-mapping>
----- 抜粋 -----

<action path="/Sample" type="com.e_jads.mediator.base.logic.LogicDistributor" name="SampleForm"
                                    scope="session" validate="true" input="/sample/error.jsp">
  <forward name="sample1" path="/NextSample1.do"/>                                    ........  ⑨
  <forward name="sample2" path="/NextSample2.do"/>                                    ........  ⑩
</action>
----- 抜粋 -----
<action path="/NextSample1" type="com.e_jads.mediator.base.logic.LogicDistributor" ......  ⑪
            name="NextSampleForm1" scope="request" validate="true" input="/sample/error.jsp">
  <forward name="success" path="/sample/next.jsp"/>
</action>
----- 抜粋 -----
<action path="/NextSample2" type="com.e_jads.mediator.base.logic.LogicDistributor" ......  ⑫
            name="NextSampleForm2" scope="request" validate="true" input="/sample/error.jsp">
  <forward name="success" path="/sample/next.jsp"/>
</action>

</action-mapping>
```

Web アプリケーション(ブラウザ)から下記の要求(Request)があった場合 (①)

path:/Sample、command:Search、mode:Init から

[実行ロジックメソッド] sample.logic.SampleLogic#logicSearchInit()

[フォワード先] 業務ロジック実行結果が「1」の場合、アクションフォワードの「sample1」(②)

業務ロジック実行結果が「2」の場合、アクションフォワードの「sample2」(③)

＊Mediator では、フォワード指定が act-forward 属性で定義されているため、下記の情報から
次に呼出される業務ロジックを判定します。

1) act-forward 属性で定義されたフォワード名(② or ③)から struts-config.xml を参照し、
遷移元の path から遷移先を特定します。(⑨ or ⑩)

2) 遷移元の path、command、mode、act-forward から action-forward.xml を参照し、
遷移先に引き渡す command、mode を特定します。(⑥⑦⑧)

3) 1)、2)の結果より、path、command、mode から遷移先の業務ロジックメソッドが実行されます。
(④⑪ or ⑤⑫)

1.4.4. logic-plugin.xml (プラグイン定義ファイル)

Mediator では、logic-mapping.xml の path タグまたは logic タグに plugin 属性を設定することにより、
業務ロジック実行前にプラグインの呼出しが可能です。

logic-plugin.xml

```
<logic-plugin>
  <plugin name="sample" class="sample.logic.plugin.SampleLogic" method="logicSample">
    <case id="false" action="forward" name="certify.error" resource="error.certify.exception" />
    <case id="true" action="rewind" name="certify.error" resource="error.certify.exception" />
  </plugin>
</logic-plugin>
```

<< plugin エレメント >>

属性	名称	説明/備考
name	プラグイン名	logic-mapping.xml の plugin 属性で定義された名称と同一になるよう設定して下さい。
class	プラグイン実行クラス名	プラグイン実行クラスを指定して下さい。
method	プラグイン実行クラスメソッド名	プラグイン実行クラスメソッドを指定して下さい。

<< case エレメント >>

属性	名称	説明/備考
id	実行ステータス ID	logic-mapping.xml の plugin 属性で定義された名称と同一になるよう設定して下さい。
action	プラグイン実行後動作指定	・forward：name 属性に指定されたフォワード先に遷移します。 ・rewind： プラグイン実行クラスメソッドの引数である ActionErrors にエラーが格納された場合に、logic-mapping.xml に input 属性が指定されていればその遷移先に、指定されていなければ struts-config.xml に指定された input 属性の遷移先にフォワードします。
name	フォワード名	struts-config.xml に定義された global-forward 名を設定して下さい。
resource	リソース名	ApplicationResource.properties に定義されたリソース名を設定して下さい。

logic-mapping.xml

```
<logic-mapping>

----- 抜粋 -----

<mapping path="/ NextSample1" plugin="sample">
  <logic command="Entry1" mode="Execute1" class="sample.logic.NextSampleLogic1"
                 viewClass="sample.NextSampleView1" viewScope="request"
                           viewContinue="false" def-forward="success"/>
</mapping>
<mapping path="/ NextSample2" >
  <logic command="Entry2" mode="Execute2" class=" sample.logic.NextSampleLogic2"
          viewClass="sample.NextSampleView2" viewScope="request" viewContinue="false"
                                              plugin="sample" def-forward=" success "/>
</mapping>

----- 抜粋 -----

</logic-mapping>
```

1.4.5. bean-mapping.xml (メソッドキャッシュ定義ファイル)

Mediator では、アプリケーションのパフォーマンスを考慮する目的で、bean-mapping.xml に設定された
クラスメソッドをアプリケーションサービス起動時にキャッシュ(データを一時的に保管して再利用する)します。

bean-mapping.xml

```
<bean-mapping>
  <package name="sample.data">
    <bean name="SampleInformationData" type="data" declared="true"/>
    <bean name="SampleListData" type="data" declared="true"/>
    <bean name="SampleData" type="data" declared="true"/>
  </package>
  <package name="sample.rec">
    <bean name="SampleInfomationListRec" type="record" declared="true"/>
    <bean name="SampleListRec" type="record" declared="true"/>
  </package>
</bean-mapping>
```

<< package エレメント >>

属性	名称	説明/備考
name	パッケージ名	キャッシュ対象となるパッケージ名を指定します。

<< bean エレメント >>

属性	名称	説明/備考
name	クラス名	キャッシュ対象となるクラス名を設定して下さい。
type	クラス種別	data ： prefix が"set"/"get"/"_frm"のメソッドをキャッシュします。 record ： prefix が"set"/"get"/"_frm"/"_set"/"_get"/"_def"のメソッドを キャッシュします。 bean ： prefix が"set"/"get"/"_frm"のメソッドをキャッシュします。 この場合、declared 属性は無効となります。
declared	スーパークラスのキャッシュ有無	name 属性で設定したクラスのスーパークラスをキャッシュ対象とする場合、 設定して下さい。

1.4.6. format-text-mapping.xml (CSV ファイル出力フォーマット定義ファイル)

Mediator では、CSV ファイルダウンロード機能を提供しています。

CSV ファイル出力フォーマット定義ファイルに各種設定を行なうことにより、Mediator ではこの定義に沿った CSV ファイルを出力します。

この XML ファイルは、Mediator で指定したタグ、属性を使用して作成します。

ファイル名の指定はありません。任意に命名して下さい。

また、ファイルの配置位置につきましても、プログラムで管理します。

format-text-mapping.xml

```
<format-text-mapping>
  <output class="sample.data.SampleData">
    <title>
      <column value="店舗名称"/>
      <column value="商品名称"/>
      <column value="メーカー名称"/>
      <column value="JANコード"/>
      <column value="前月在庫"/>
      <column value="当月入庫"/>
      <column value="当月出庫"/>
      <column value="当月在庫"/>
      <column value="最終入庫日"/>
      <column value="最終出庫日"/>
    </title>
    <item>
      <property name="SALON_NM"/>
      <property name="ITEM_NM"/>
      <property name="SIIRE_NM"/>
      <property name="ITEM_JAN_CD"/>
      <property name="STOCK_ZEN_NUM"/>
      <property name="CUR_RECEIPT_NUM"/>
      <property name="CUR_SHIPMENT_NUM"/>
      <property name="STOCK_NUM"/>
      <property name="RECEIPT_DT"/>
      <property name="SHIPMENT_DT"/>
    </item>
  </output>
<format-text-mapping>
```

<< output エレメント >>

属性	名称	説明/備考
class	クラス名	テキストファイル出力時に読み出し先となるデータクラス名を指定します。

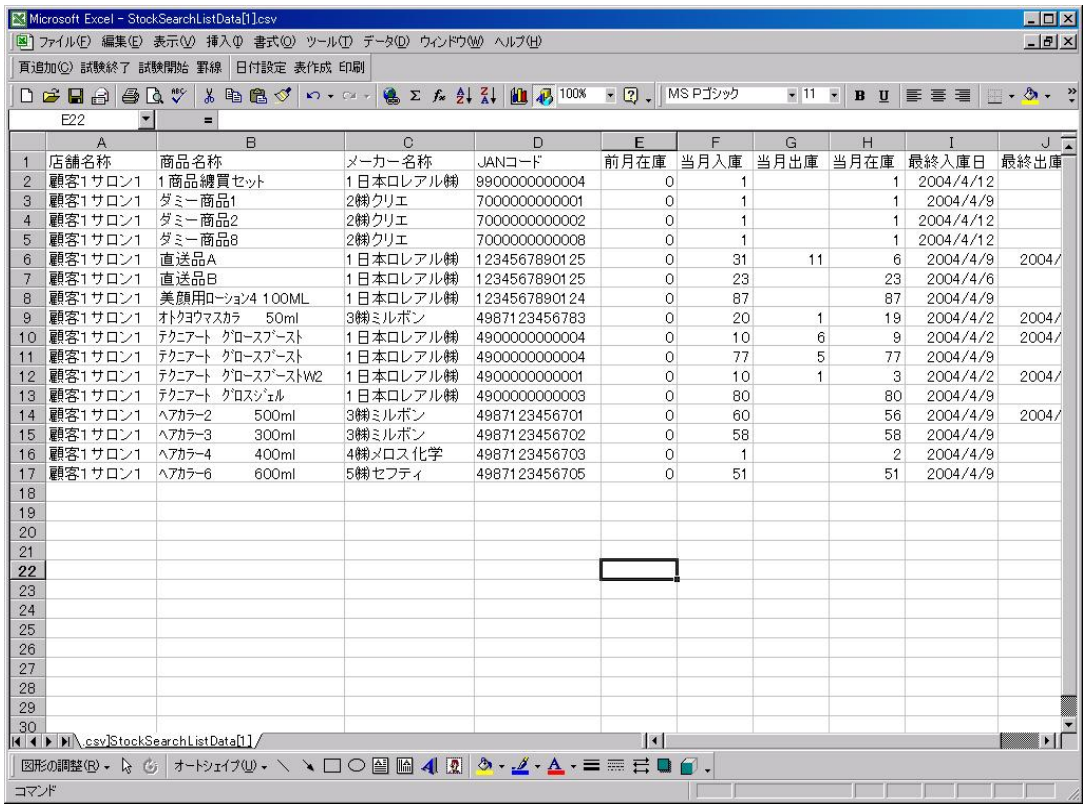
<< column エレメント >> ・・・ title タグ内に記述して下さい。

属性	名称	説明/備考
value	カラム項目名	出力されるテキストデータの先頭行に value 値で指定されている項目名を挿入します。

<< property エレメント >>・・・ item タグ内に記述して下さい

属性	名称	説明/備考
name	カラム項目名	データ出力時に class 属性で指定されたデータクラスから呼出されるメソッド名(prefixを除く)を指定します。

〈CSV 出力結果イメージ〉



Microsoft Excel - StockSearchListData[1].csv

	A	B	C	D	E	F	G	H	I	J
1	店舗名称	商品名称	メーカー名称	JANコード	前月在庫	当月入庫	当月出庫	当月在庫	最終入庫日	最終出庫
2	顧客1サロン1	1商品縺買セット	1日本ロレアル㈱	9900000000004	0	1		1	2004/4/12	
3	顧客1サロン1	ダミー商品1	2㈱クリエ	7000000000001	0	1		1	2004/4/9	
4	顧客1サロン1	ダミー商品2	2㈱クリエ	7000000000002	0	1		1	2004/4/12	
5	顧客1サロン1	ダミー商品8	2㈱クリエ	7000000000008	0	1		1	2004/4/12	
6	顧客1サロン1	直送品A	1日本ロレアル㈱	1234567890125	0	31	11	6	2004/4/9	2004/
7	顧客1サロン1	直送品B	1日本ロレアル㈱	1234567890125	0	23		23	2004/4/6	
8	顧客1サロン1	美顔用ﾛｰｼｮﾝ4 100ML	1日本ロレアル㈱	1234567890124	0	87		87	2004/4/9	
9	顧客1サロン1	ｵﾄｸﾖｳﾏｽｶﾗ 50ml	3㈱ミルボン	4987123456783	0	20	1	19	2004/4/2	2004/
10	顧客1サロン1	ﾃｸﾆｱｰﾄ ｸﾞﾛｰｽﾌﾟｰｽﾄ	1日本ロレアル㈱	4900000000004	0	10	6	9	2004/4/2	2004/
11	顧客1サロン1	ﾃｸﾆｱｰﾄ ｸﾞﾛｰｽﾌﾟｰｽﾄ	1日本ロレアル㈱	4900000000004	0	77	5	77	2004/4/9	
12	顧客1サロン1	ﾃｸﾆｱｰﾄ ｸﾞﾛｰｽﾌﾟｰｽﾄW2	1日本ロレアル㈱	4900000000001	0	10	1	3	2004/4/2	2004/
13	顧客1サロン1	ﾃｸﾆｱｰﾄ ｸﾞﾛｽｼﾞｪﾙ	1日本ロレアル㈱	4900000000003	0	80		80	2004/4/9	
14	顧客1サロン1	ﾍｱｶﾗｰ2 500ml	3㈱ミルボン	4987123456701	0	60		56	2004/4/9	2004/
15	顧客1サロン1	ﾍｱｶﾗｰ3 300ml	3㈱ミルボン	4987123456702	0	58		58	2004/4/9	
16	顧客1サロン1	ﾍｱｶﾗｰ4 400ml	4㈱メロス化学	4987123456703	0	1		2	2004/4/9	
17	顧客1サロン1	ﾍｱｶﾗｰ6 600ml	5㈱セフティ	4987123456705	0	51		51	2004/4/9	

1.4.7. validate-mapping.xml (バリデート動作定義ファイル)

struts では、送信データのチェック処理を記述するための検証メソッドとして、アクションフォーム Bean に「validate」が用意されています。

Mediator では、この validate メソッドにロジックを実装するかわりに、定義ファイル上にアクションフォーム Bean の機能コード(command)、動作モード(mode)単位に各プロパティのチェック処理を定義します。

validate-mapping.xml

```
<validate-mapping>

----- 抜粋 -----

  <package name="sample">
    <form name="SampleForm">
      <action name="sampleSearchCategoryValidate" command="Search" mode="Category"/>
      <action name="sampleSearchItemValidate" command="Search" mode="Item"/>
      <action name="sampleSearchNameValidate" command="Search" mode="Name"/>
      <action name="sampleSearchCodeValidate" command="Search" mode="Code"/>
      <action name="sampleSearchOrderValidate" command="Search" mode="Order"/>
      <action name="sampleSearchShipmentValidate" command="Search" mode="Shipment"/>
    </form>

----- 抜粋 -----

<validate-mapping>
```

<< package エレメント >>

属性	名称	説明/備考
name	パッケージ名	バリデート対象となるパッケージ名を指定します。

<< form エレメント >>

属性	名称	説明/備考
name	フォーム名	バリデート対象となるアクションフォーム Bean 名を指定します。

<< action エレメント >>

属性	名称	説明/備考
name	バリデート名	バリデートロジック定義に設定されているバリデート名を指定します。 バリデートロジック定義 ：validate-{アクションフォーム Bean クラス名}
command	機能コード	
mode	動作モード	

1.4.8. validate-{アクションフォーム Bean クラス名}.xml (バリデートロジック定義ファイル)

validate-mapping.xml(**1.4.7.参照**)でアクションフォーム Bean 毎の機能コード、動作モード単位に定義された name 属性に紐付くバリデート処理のロジック定義をします。

各アクションフォーム Bean のバリデートロジック定義ファイル名は、下記のように指定して下さい。

ファイル名：validate-{アクションフォーム Bean クラス名}.xml

1.4.7 を例に sampleForm 用のバリデートロジック定義ファイルを定義します。

validate-SampleForm.xml

```
<?xml version="1.0" encoding="Shift_JIS"?>
<validate-SampleForm>
  <validate name="sampleSearchCategoryValidate" ........ ①
        checker="com.e_jads.mediator.validator.util.ValidateUtil" resourceTopKey="validateError.app">
    <target property="sampleCd">
      <check method="Property" msgKey="dispFlg" >
        <param const="SAMPLE_CD"/>
        <param sessionName="sample.SampleSearchCategoryView"
                                                sessionProperty="sampleSelectListDataCollection" />
      </check>
    </target>
  </validate>
  <validate name="sampleSearchItemValidate"  ........ ②
        checker="validator.SampleValidator" resourceTopKey="validateError.app">
    <target property="queryValue">
      <check method="SampleItemQueryValue" split="true" splitChar="-" msgKey="dispFlg" >
        <param const="SAMPLE_CD1"/>
        <param const="SAMPLE_CD2"/>
        <param sessionName="sample.SampleSearchCategoryView1"
                                                sessionProperty="SampleListDataCollection1" />
        <param sessionName="sample.SampleSearchCategoryView2"
                                               sessionProperty="SampleListDataCollection2" />
      </check>
    </target>
  </validate>
  <validate name="sampleSearchNameValidate"   ........ ③
        checker="com.e_jads.sharelib.util.CheckUtil" resourceTopKey="validateError.simple">
    <target property="sampleNm">
      <check method="Value" msgKey="checkNull" msgParamKey="item" msgParam1="SampleNm" break="true"/>
      <check method="Length" msgKey="checkMaxLength" msgParamKey="item" msgParam1="SampleNm"
                                                                  msgConst2="28">
        <param const="1"/>
        <param const="28"/>
      </check>
      <check checker="validator.ValidatorCommon" method="HtmlEncodeChar" msgKey="checkChar"
                                                          msgParamKey="item" msgParam1="SampleNm" />
    </target>
  </validate>
  <validate name="sampleSearchOrderValidate"   ........ ④
        checker="com.e_jads.sharelib.util.CheckUtil" resourceTopKey="validateError.simple">
    <target property="orderNo" pass="true">
      <check method="Length" msgKey="checkMaxLength" msgParamKey="item" msgParam1="OrderNo"
                                                                        msgConst2="7">
        <param const="1"/>
        <param const="7"/>
      </check>
      <check method="Number" msgKey="checkAsciiNumber" msgParamKey="item" msgParam1="OrderNo" />
    </target>
    <target property="orderSpan">
      <check method="OrEquals" msgKey="checkInvalid" msgParamKey="item" msgParam1="Span" >
        <param const="1,2,3" />
      </check>
    </target>
  </validate>
```

① sampleSearchCategoryValidate

<< validate エレメント >>

属性	名称	説明/備考
name	バリデート名	バリデート定義名を指定します。
checker	チェッカークラス名	バリデートロジックが実装されたチェッカークラス名を指定します。
resourceTopKey	リソース KEY の上部	ApplicationResource.propertis に定義されたリソースキーの Top 部を指定します。 下位エレメントの resourceSubkey 属性、msgKey 属性との組み合わせから リソースキーが特定されます。

<< target エレメント >>

属性	名称	説明/備考
property	プロパティ名	バリデート対象となるメインプロパティ名を指定します。 validate タグの name 属性で指定された ActionForm の getter メソッド名の prefix を除いた名前です。

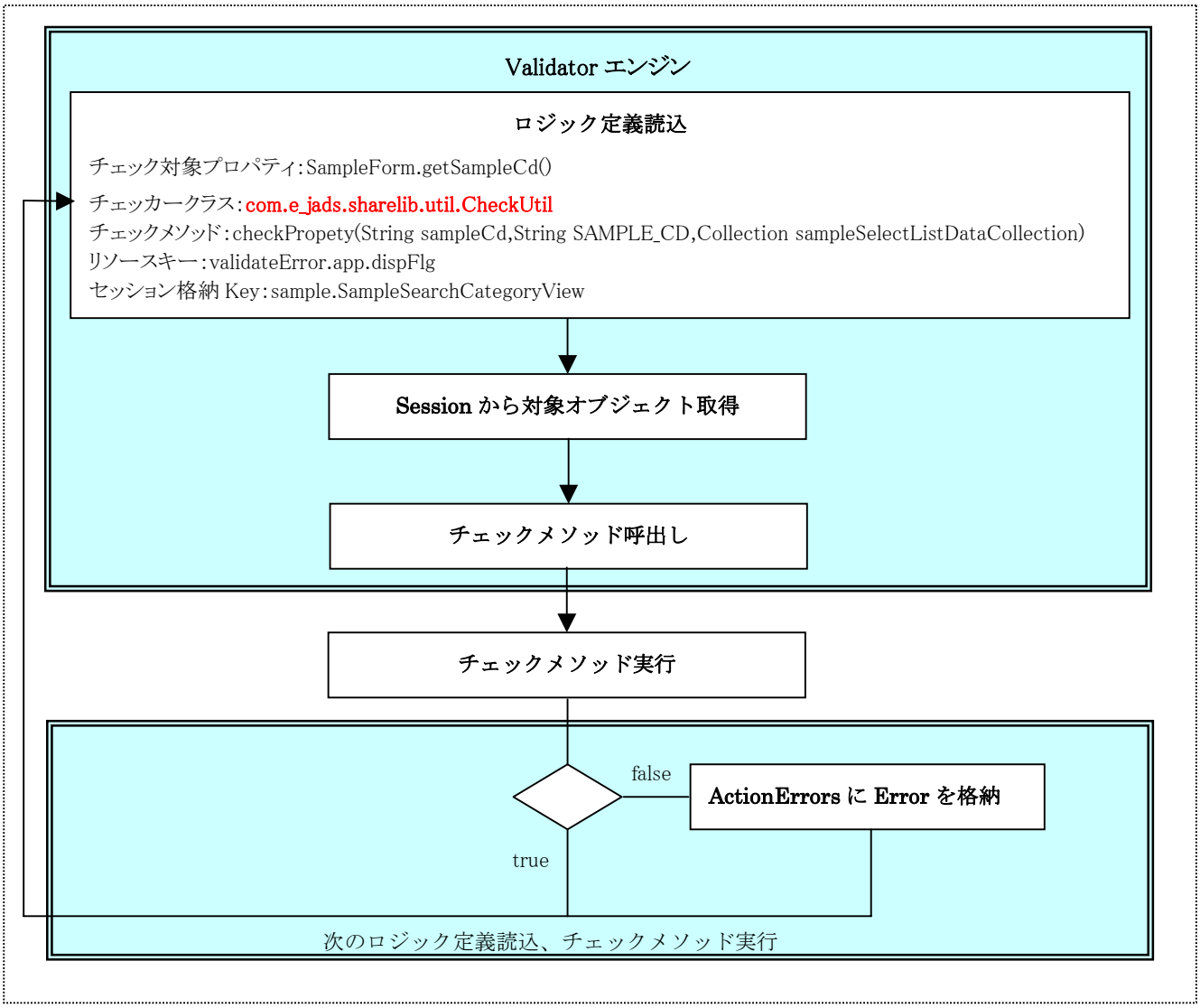
<< check エレメント >>

属性	名称	説明/備考
method	チェックメソッド名	validate タグの checker 属性で指定されたクラスに実装されたメソッド名を指定します。
msgKey	チェッカークラス名	ApplicationResource.propertis にアクセスするためのリソース KEY を指定します。

<< param エレメント >>

属性	名称	説明/備考
const	固定値パラメータ指定	任意の固定文字列を指定します。
sessionName	セッション格納 KEY	Session 上に格納されているオブジェクトの格納 Key 名を指定します。
sessionProperty	セッション格納オブジェクト getter メソッド名	Session 上に格納されているオブジェクトの getter メソッド名(prefix は除く)を 指定します。

上記設定により、下記のようにチェック処理が実行されます。



＊ com.e_jads.sharelib.util.CheckUtil とは

Mediator が提供するバリデート用の API です。 詳細は JavaDoc を参照下さい。

② sampleSearchItemValidate

<< check エレメント >>

属性	名称	説明/備考
split	プロパティ分割設定	property 属性で取得した値を分割して処理したい場合に指定します。 splitChar 属性で指定された文字で分割し、チェッカーメソッドに String 型の配列で引き渡されます。
splitChar	プロパティ分割文字	split 属性が true の場合に区切り文字として使用される文字を指定します。

ロジック定義読込結果

チェック対象プロパティ:SampleForm.getQueryValue()
チェッカークラス:validator.SampleValidator
チェックメソッド:checkSampleItemQueryValue (String[] value,String SAMPLE_CD1, String SAMPLE_CD2,
Collection sampleSelectListDataCollection1, Collection sampleSelectListDataCollection2)
リソースキー:validateError.app.dispFlg
セッション格納 Key:sample. SampleSearchCategoryView1
セッション格納オブジェクト getter メソッド名:get SampleListDataCollection1 ()
セッション格納 Key:sample. SampleSearchCategoryView2
セッション格納オブジェクト getter メソッド名:get SampleListDataCollection2 ()

* validator.SampleValidator とは

Mediator が提供するバリデート用の API 以外にシステムに依存するチェック処理が必要な場合、システムに依存するチェッカークラスを作成し、バリデートロジック定義ファイルに設定することにより呼出しが可能です。

③ sampleSearchNameValidate

<< check エレメント >>

属性	名称	説明/備考
msgParamKey	パラメータリソース KEY の TOP 部	ActionErrors に格納されるエラーメッセージにバインドされるパラメータメッセージにアクセスするためのリソース KEY の TOP 部を指定します。MsgParam 属性との組み合わせからリソースキーを特定します。
msgParam1	メッセージパラメータリソース KEY	ActionErrors に格納されるエラーメッセージにバインドされるパラメータメッセージにアクセスするためのリソース KEY を指定します。msgParam1～msgParam4 属性まで指定できます。
msgConst2	メッセージパラメータ文字列	ActionErrors に格納されるエラーメッセージにバインドされるパラメータメッセージにアクセスするためのリソース KEY を指定します。msgConst1～msgConst4 属性まで指定できます。
break	バリデート中止設定	この指定をした場合、実行されたチェック処理結果が false の時に以降の check エレメントの処理を行わず、次の target エレメントへ処理が移行します。

ロジック定義読込結果

チェック対象プロパティ:SampleForm.getSampleNm()

第 1 チェック
チェッカークラス:com.e_jads.sharelib.util.CheckUtil
チェックメソッド:checkQueryValue (String sampleNm)
リソースキー:validateError.simple.checkNull
リソースパラメータキー(第1):sample.sampleNm

第 1 チェック
チェッカークラス:com.e_jads.sharelib.util.CheckUtil
チェックメソッド:checkQueryValue (String sampleNm)
リソースキー:validateError.simple.checkNull
リソースパラメータキー(第1):sample.sampleNm

第 1 チェック
チェッカークラス:com.e_jads.sharelib.util.CheckUtil
チェックメソッド:checkQueryValue (String sampleNm)
リソースキー:validateError.simple.checkNull
リソースパラメータキー(第1):sample.sampleNm