DOWAホールディングス株式会社様基幹業務システム再構築(Step2)  
画面開発ガイドライン  
0.50版

改訂履歴

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版 | 改訂日 | 改訂箇所 | 改訂内容 | 作成者 | 承認者 |
| 0.50 | 2015/12/01 |  | 新規作成 | 駒田 |  |

目次

[1. はじめに 5](#_Toc443148999)

[1.1. 目的 5](#_Toc443149000)

[1.2. 対象読者 5](#_Toc443149001)

[1.3. 前提知識 5](#_Toc443149002)

[2. 開発環境 6](#_Toc443149003)

[2.1. 開発環境 6](#_Toc443149004)

[2.1.1. Intra-mart 7.2 Debug Server 6](#_Toc443149005)

[2.1.2. Orcale 11g 6](#_Toc443149006)

[2.1.3. Eclipse 6](#_Toc443149007)

[2.1.4. JDK 6](#_Toc443149008)

[2.2. ブラウザ 6](#_Toc443149009)

[3. アプリケーションの構成 7](#_Toc443149010)

[3.1. フレームワークの利用 7](#_Toc443149011)

[3.2. アプリケーション構成のレイヤー 7](#_Toc443149012)

[3.3. クライアント層の構成 7](#_Toc443149013)

[3.3.1. CSS 8](#_Toc443149014)

[3.3.2. CSJS 8](#_Toc443149015)

[3.4. プレゼンテーション層の構成 8](#_Toc443149016)

[3.4.1. JSP 10](#_Toc443149017)

[3.5. ビジネス層の構成 10](#_Toc443149018)

[3.6. インテグレーション層の構成 12](#_Toc443149019)

[4. アプリケーションの実装 14](#_Toc443149020)

[4.1. クライアント層 14](#_Toc443149021)

[4.1.1. 画面遷移の方法 14](#_Toc443149022)

[4.1.2. 登録や更新時のサーバー送信 15](#_Toc443149023)

[4.1.3. JsRenderの利用 15](#_Toc443149024)

[4.2. プレゼンテーション層 15](#_Toc443149025)

[4.2.1. Formクラスの実装について 15](#_Toc443149026)

[4.2.2. 基本マスタ、基本マスタ(取引先)の補完 17](#_Toc443149027)

[4.3. ロジック層 19](#_Toc443149028)

[4.4. サービス層 19](#_Toc443149029)

[4.4.1. サービス実装の構成 19](#_Toc443149030)

[4.4.2. 登録、更新者コード、登録更新日時等の補完 21](#_Toc443149031)

[4.4.3. Oracle Char型に対する対応 21](#_Toc443149032)

[4.4.4. Entity、ServiceGeneratedの自動生成 24](#_Toc443149033)

[4.4.5. 排他制御(SELECT ～ FOR UPDATE)メソッド 24](#_Toc443149034)

[4.5. 層をまたいだ制御 24](#_Toc443149035)

[4.5.1. Form、Dto、ModelのFiledのデータ型について 24](#_Toc443149036)

[4.5.2. データの移送について 25](#_Toc443149037)

[4.5.3. 一覧のページングについて 25](#_Toc443149038)

[4.5.4. ログインユーザ情報の利用について 31](#_Toc443149039)

[4.5.5. 例外処理 31](#_Toc443149040)

[4.6. セキュリティ 32](#_Toc443149041)

[4.6.1. SQLインジェクション対策 32](#_Toc443149042)

[4.6.2. クロスサイト・スクリプティング対策 34](#_Toc443149043)

[4.6.3. クロスサイト・リクエストフォージェリ対策 34](#_Toc443149044)

[5. 実装イメージの解説 37](#_Toc443149045)

[5.1. 画面の初期表示 37](#_Toc443149046)

[5.1.1. 画面遷移イメージ 37](#_Toc443149047)

[5.1.2. ソースの構成 37](#_Toc443149048)

[5.1.3. プレゼンテーション層 38](#_Toc443149049)

[5.1.4. ロジック層 53](#_Toc443149050)

[5.1.5. サービス層 59](#_Toc443149051)

[5.2. 画面の登録／更新／削除機能 60](#_Toc443149052)

[5.2.1. 画面遷移イメージ 60](#_Toc443149053)

[5.2.2. ソースの構成 61](#_Toc443149054)

[5.2.3. クライアント層 61](#_Toc443149055)

[5.2.4. プレゼンテーション層 62](#_Toc443149056)

[5.2.5. ロジック層 70](#_Toc443149057)

[5.2.6. サービス層 74](#_Toc443149058)

[5.3. 単票画面の一覧表示機能 74](#_Toc443149059)

[5.3.1. 実装イメージ 74](#_Toc443149060)

[5.3.2. プレゼンテーション層(値の取得と、templateの展開) 74](#_Toc443149061)

[5.3.3. クライアント層 78](#_Toc443149062)

[5.3.4. プレゼンテーション層(一覧の更新データ取得) 79](#_Toc443149063)

[5.4. 一覧画面の検索機能 79](#_Toc443149064)

[5.4.1. 画面遷移イメージ 79](#_Toc443149065)

[5.4.2. ソースの構成 80](#_Toc443149066)

[5.4.3. クライアント層 80](#_Toc443149067)

[5.4.4. プレゼンテーション層 82](#_Toc443149068)

[5.4.5. ロジック層 91](#_Toc443149069)

[5.4.6. サービス層 95](#_Toc443149070)

# はじめに

本項では本文書の目的/対象読者/前提知識について記述する。

## 目的

本文書はDOWAホールディングス株式会社様(以降、DOWA様)の基幹業務システム再構築(Step2)(以降、本システム)の品質、メンテナンス性を一定レベルで保ち、各開発者が業務機能の実装に注力できるよう、画面機能の標準実装を説明することを目的とする。

本文書では、画面機能の標準的な実装を定義するが、本標準の考慮外の事項によって、実装が妨げられるのは本末転倒である。標準的な実装に全て則る必要はないが、標準的な実装外で実装する場合は、十分に考慮の上行うこと。また、コードレビュー担当者は、標準外の認識をもって、慎重にレビューを行うこと。

## 対象読者

本文書は本システムを構築するアプリケーションエンジニア、プログラマー、アプリケーションアーキテクトを対象とする。

## 前提知識

本文書は以下に関しての知識を有していることを前提とする。

　・intra-mart

　・Java

　・JavaScript

　・HTML

# 開発環境

本項では、画面開発の開発環境について記述する。

## 開発環境

### Intra-mart 7.2 Debug Server

開発時はAPサーバーとして、Intra-mart 7.2のDebugServerを利用する。開発環境パックに導入済みであり、開発環境パックの前提の通りに利用すること。

### Orcale 11g

開発時は、Oracle 11gを利用する。Oracleのサイトからダウンロードし、開発用PCにインストールすること。インスタンスはUTF8で構築すること。

### Eclipse

開発時は、IDEとしてEclipseを利用する。開発環境パックに導入済みであり、開発環境パックの前提の通りに利用すること。また、別ドキュメント(eclipse環境設定.xlsx)に、CheckStyleの設定等を記載している。必ず設定を確認して仕様すること。

### JDK

開発時は、JDK 1.6.0\_30を利用する。Eclipse自体は、Java 8系で動作させ、ビルドはJDK 6系で行う。ビルドパス、実行構成のJDKバージョンには、十分注意すること。

## ブラウザ

開発、テスト時は、必ず以下のブラウザで確認を行うこと。

・Internet Explorer 11

※上記以外のブラウザおよびモバイル用ブラウザについては対象外とする。

# アプリケーションの構成

本項では、画面開発における画面構成について記述する。

## フレームワークの利用

本システムでは、intra-mart 7.2上に、SAStrutsフレームワークと、S2JDBC(Seaserプロジェクト)を利用して画面を開発する。

参照(SAStruts公式)

<http://sastruts.seasar.org/>

参照(S2JDBC公式)

<http://s2container.seasar.org/2.4/ja/s2jdbc.html>

## アプリケーション構成のレイヤー

本システムでは、サーバー側の処理を「プレゼンテーション層」、「ビジネス層」、「インテグレーション層」の3つのレイヤーに分けて管理する。本システムでは、これに「クライアント層」を加え、4つのレイヤーとしてアプリケーション構成を管理する。



図 3‑1　アプリケーション構成のレイヤー

## クライアント層の構成

クライアント層は、CSS(Cascading Style Sheets)、CSJS(Client Side JavaScript)から構成する。以下の役割を担う。

・画面の描写、および、画面制御

クライアント層は、以下の言語で開発する。

・HTML

・CSS

・JavaScript

クライアント層では、原則的に、動的な画面制御のみを行う。静的な画面制御(表示モードの切り替え等)は、プレゼンテーション層にて実施する。

### CSS

基本的に、intra-mart標準のCSS、もしくは、共通CSS(common.css)を利用すること。

システム全般で統一的なデザインになるよう留意する。また、共通CSSの整備を進めることによって、ほとんどの画面において、実装が不要になるはずである。共通CSSのリファレンスは、SVNリポジトリにコミットされている。リファレンスを参照しながら、実装をおこなうこと。

参照(共通CSSリファレンス)

<http://localhost:8081/imart/dowa/css_ref/index.jsp>

画面特有のデザインがある場合は、1画面において1ファイルのCSSファイルを作成してもよい。

基本設計、詳細設計工程で作成されたモックアップは、あくまで、デザイン、機能の確認のために作成されているだけであり、開発のために作成されたものではない。そのため、モックアップを参考にしてよいが、正しいデザインで実装すること。

### CSJS

原則的に、JQueryを利用して実装する。JQueryでできることは、JQueryで実装すること。

システム全般で動作を統一するため、共通CSJSを整備する。また、必要に応じて、共通CSJSへの追加のリクエストをすること。 基本的にJSP上にはCSJSを記載しない。必ず、外部ファイルとしてCSJSを作成し実装を行う。外部ファイルは、1画面に置いて、1ファイルとする。また、機能間のfunction名の重複を避けるため、名前空間を利用すること。

整備中。

参照(共通CSJSリファレンス)

// TODO:

## プレゼンテーション層の構成

プレゼンテーション層は、Actionから構成する。以下の役割を担う。

・リクエストに対する共通的な制御を行う

・ユーザーの入力値に対しての妥当性検証(バリデーション)を行う

・ユーザーからの要求を受け付け、ロジックを起動する

・ロジックの結果に基づいて適切な画面を構築する

プレゼンテーション層は、以下の言語で開発する。

・Java



図 3‑2　プレゼンテーション層の構成

プレゼンテーション層は、上記のとおり、Form、Action、Model、JSPによって構成する。

以下に作成粒度を記述する。

表 ‑　プレゼンテーション層の役割と粒度

|  |  |  |
| --- | --- | --- |
| 主要クラス | 役割 | 粒度 |
| Form | 画面からアクションへ入力値を引き渡すクラス | 1アクションに対して1 |
| Action | リクエストを処理し、ビジネス層へ処理を依頼する。処理結果を受け取り返却する。 | 1画面に対して原則1 |
| Model | アクションから、JSPへ入力値を引き渡すクラス | 1画面に対して1 |
| JSP | 画面のレイアウトを定義する。 | 1画面に対して1 |

プレゼンテーション層の実処理は、Actionクラスに記載する。

ビジネス層、インテグレーション層は、画面に依存するものでないため、原則的にJavaEEモジュール(HttpSession、HttpServletRequest、HttpServletResponse)にアクセスしない。そのため、これらのモジュールにアクセスするのは、「Action」と「JSP」の役割となる。

原則的に、Actionクラスは画面に対し、1:1で実装し、画面のボタン等の処理は、Actionクラスのメソッドして実装するが、Actionクラスがあまりにも肥大化する場合は、Actionクラスを分割してもよい。

### JSP

HTMLを利用して実装する。正しいHTMLで実装すること。

基本設計、詳細設計工程で作成されたモックアップは、あくまで、デザイン、機能の確認のために作成されているだけであり、開発のために作成されたものではない。そのため、モックアップを参考にしてよいが、正しいマークアップで実装すること。

JSPは、Apache Tilesを利用して構造化を行う。Apache Tilesを利用しての構造化の実装については、「xx\_JSPの実装について\_0.50.docx」を参照すること。

本システムでは、共通的な制御(表示制御や、基本マスタ検索ダイアログ等)が多い。この部分は、taglibによって共通化を実施する。リファレンスを参考に実装を行う。

参照(共通Taglibリファレンス)

http://52.68.179.228/svn/1510\_hanbai\_sanpai/document/40\_製造（M）/01\_調査・検討/開発ガイドライン/タグライブラリ一覧.xlsx

## ビジネス層の構成

ビジネス層は、Logic、Componentから構成する。以下の役割を担う。

・ビジネスロジック(値の検証や計算)を行う

・データアクセスのインターフェイスを公開する

ビジネス層は、以下の言語で開発する。

・Java



図 3‑3 ロジック層の構成

ロジック層は、上記のとおり、Dto、Logic、Component、ResultDtoによって構成する。

以下に作成粒度を記述する。

表 ‑ ロジック層の役割と粒度

|  |  |  |
| --- | --- | --- |
| 主要クラス | 役割 | 粒度 |
| Dto | アクションからロジックへ入力値を引き渡すクラス | ロジックの1メソッドに対して1 |
| Logic | ビジネスロジックを処理する。必要に応じてインテグレーション層を呼び出す。 | ※1 |
| Component | ビジネスロジックを構成するうえでの部品。 | 任意 |
| ResultDto | ロジックからアクションへ返却値を引き渡すクラス | ロジックの1メソッドに対して1 |

※1 Logicの粒度は、「1つの業務」とする。Logicのメソッドの粒度は、1つの業務に対しての「1つの操作」とする。例えば、「車両登録」業務の場合、「車両登録」業務で1Logicとする。「車両登録」を「登録する」、「削除する」、「検索する」でLogic内のメソッドとする。

Logicは業務単位で作成する。そのため、Logicは、クラスが肥大化しやすい。

Logicクラスはビジネスロジックの入り口として考え、実処理は、Componentクラスに実装する。Componentクラスは、開発者の任意の単位で作成してよい。

## インテグレーション層の構成

インテグレーション層は、Serviceから構成され、以下の役割を担う。

・データベースにアクセスする

インテグレーション層は、以下の言語で開発する。

・Java



図 3‑4 インテグレーション層の構成

インテグレーション層は、上記のとおり、Entity、Condition、Serviceによって構成する。

以下に作成粒度を記述する。

表 ‑ インテグレーション層の役割と粒度

|  |  |  |
| --- | --- | --- |
| 主要クラス | 役割 | 粒度 |
| Entity | ロジックからサービスへ入力値を引き渡すクラス(テーブル、取得データと同一の構成) | 1テーブルに対して1  1SQLに対して1 |
| Condition | ロジックからサービスへ入力値を引き渡すクラス(条件句等) | 1SQLに対して1 |
| Service | データベースアクセスを提供するインターフェイス | 1テーブルに対して1 |

DBへの問い合わせはS2JDBCを利用するが、Serviceクラス以外では、JDBCManagerを利用しない。また、Serviceクラス内で、単純な構成で、かつ、10メソッド実行以内での「流れるようなインターフェイス」は許容する。ただし、単純な公正でない、または、10行を超える場合は、SQLファイルを作成することによって実装を行う。

参照(流れるようなインターフェースと脱CoC)

<http://s2container.seasar.org/2.4/ja/s2jdbc_abstract.html#%E6%B5%81%E3%82%8C%E3%82%8B%E3%82%88%E3%81%86%E3%81%AA%E3%82%A4%E3%83%B3%E3%82%BF%E3%83%BC%E3%83%95%E3%82%A7%E3%83%BC%E3%82%B9%E3%81%A8%E8%84%B1CoC>

# アプリケーションの実装

本項では、アプリケーションの実装方法について記載する。

## クライアント層

本項では、クライアント層特有の実装方法について記載する。

### 画面遷移の方法

画面遷移は、Formのsubmitか、基本的に共通部品として実装されたCSJSを利用して遷移します。

以下は、Formのsubmitによる画面遷移の例です。

<script>

// 本来は、CSJSファイルに記載する

　function transfer() {

$(‘#form’).submit();

}

</scrip>

<form method=”POST” action=”URL”id=”form”>

</form>

以下は、共通部品として実装されたCSJSを利用しての画面遷移です。

<script>

// 本来は、CSJSファイルに記載する

　function transfer() {

var data = {};

data.parameterA = ‘AAAA’;

dowa.sd.core.request.transitionTo($(‘#form’).attr(‘action’), data);

}

</scrip>

<form method=”POST” action=”URL”id=”form”>

</form>

### 登録や更新時のサーバー送信

登録や更新時のサーバー送信は、基本的に共通部品として実装されたCSJSを利用して遷移します。

<script>

// 本来は、CSJSファイルに記載する

　function onClickRegisterBtn() {

// 登録用FORMのactionをFormにコピーする

$('#form).attr('action', $('#editForm').attr('action'));

// FORMと登録後の遷移先を引数に、共通部品を呼び出す

dowa.sd.core.request.ajaxSubmit($('#form'), $('#readForm').attr('action')); }

</scrip>

<form method=”POST” id=”form”>

<input type=”text”name=”aaaa” value=”AAAA”/>

</form>

<form method=”POST” action=”登録ActionURL”id=”editForm”>

</form>

<form method=”POST” action=”遷移先ActionURL”id=”readForm”>

</form>

### JsRenderの利用

一覧部分は、JsRender(CSJSのテンプレートエンジン)を利用して実装する。

参照(JsRender公式)

<http://www.jsviews.com/>

実装方法は、実装イメージの解説を参照のこと。

## プレゼンテーション層

本項では、プレゼンテーション層特有の実装方法について記載する。

### Formクラスの実装について

Formは、以下の基準で実装を行う。。

■画面の項目にあたるfieldは、以下のイメージの順序でFormクラスに定義する。



図 4‑1　Formクラスのfield順

これは、FormクラスのField順にValidatorが実行されるため、fieldのエラーを決まった順序で出力するためである。

■Validatorのアノテーションは、Validator一覧の昇順

/\*\* 製品部門 . \*/

Validator一覧の昇順に並べる。

@Required(arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.003") )

@Maxbytelength(maxbytelength = 3, arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.003") )

public String spart;

これは、fieldに指定した順にValidatorが実行されるため、「必須」「形式チェック」等のValidatorの種別ごとの順序を統一するためである。

■画面の項目にあたらないfield(hidden項目)は、画面項目にあたるものの下にまとめて記述する。

画面項目のメンテナンス時に、修正しやすいようにするため。

■Formクラスには、lombokのToStringアノテーションを付与する。

@ToString

public class EditForm implements Serializable {

エラー発生時にログ出力するため。

### 基本マスタ、基本マスタ(取引先)の補完

基本マスタ、基本マスタ(取引先)を簡易に利用できるよう、「基本マスタ補完」機能を利用することができる。

基本マスタ、基本マスタ(取引先)の補完は、以下の契機で行われる。

・Actionクラス呼出し前後のInterceptorにおいて、publicメソッド動作後

・ResponseUtils#writeにおいて、引数をJSON文字列に変換しHttpServletResponseに書き込む直前

基本マスタ補完機能は、以下のアノテーションによって基本マスタ、基本マスタ(取引先)の内容を補完する。

表 ‑ 基本マスタ、基本マスタ(取引先)の補完用アノテーション

|  |  |  |
| --- | --- | --- |
| アノテーション名 | アノテーションが付与できるデータ型 | 用途 |
| BaseMasterSupplementInfo | List<HashMap<String, String>> | プルダウンリスト、チェックボックス、ラジオボタン等 |
| BaseMasterSupplementName | String | 検索入力時の名称等 |

#### BaseMasterSupplementInfo

以下のように、アノテーションを設定することによって、基本マスタの内容が補完される。

/\*\* 運搬事業者プルダウン. \*/

@BaseMasterSupplementInfo(cdtyp = "001", spart = "01", targetkbn = "1", valueField = "ujscd",

nmflg = true)

private List<Map<String, String>> ujsCdSelect;

このアノテーションは、List<Map<String, String>>型にしか適用できない。この型はdowaui:select、dowaui:checkbox、dowaui:radioにそのまま設定できる。

BaseMasterSupplementInfoは、以下の引数を持つ。

表 ‑　BaseMasterSupplementInfoアノテーションの引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| targetkbn | 取得区分(1:基本マスタ、2:基本マスタ(得意先) | String | 必須 |
| spart | 製品部門 | String | 必須 |
| kunnr | 得意先コード(取得区分が2:基本マスタ(得意先)の場合は必須) | String | 任意 |
| cdtyp | マスタ種別 | String | 任意 |
| ms1kb | 区分1 | String | 任意 |
| ms2kb | 区分2 | String | 任意 |
| ms3kb | 区分3 | String | 任意 |
| ms4kb | 区分4 | String | 任意 |
| ms5kb | 区分5 | String | 任意 |
| bsdtField | 検索基準日(未入力の場合、システム日付を検索基準日とする) | String | 任意 |
| nmflg | 名称フラグ(true:名称を取得、false、略称を取得、defaultはfalse) | boolean | 任意 |
| valueField | 値が設定されるFiled名(selected等を付加するために利用) | String | 必須 |

#### BaseMasterSupplementName

以下のように、アノテーションを設定することによって、基本マスタの内容が補完される。

/\*\* 車種名称. \*/

@BaseMasterSupplementName(spart = "ZZ", cdtyp = "12", ms1kb = "5", targetkbn = "1",

valueField = "syscd")

private String mstnm;

このアノテーションは、String型にしか適用できない。BaseMasterSupplementNameは、以下の引数を持つ。

表 ‑ BaseMasterSupplementNameアノテーションの引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| targetkbn | 取得区分(1:基本マスタ、2:基本マスタ(得意先) | String | 必須 |
| spart | 製品部門 | String | 必須 |
| kunnr | 得意先コード(取得区分が2:基本マスタ(得意先)の場合は必須) | String | 任意 |
| cdtyp | マスタ種別 | String | 任意 |
| ms1kb | 区分1 | String | 任意 |
| ms2kb | 区分2 | String | 任意 |
| ms3kb | 区分3 | String | 任意 |
| ms4kb | 区分4 | String | 任意 |
| ms5kb | 区分5 | String | 任意 |
| bsdtField | 検索基準日(未入力の場合、システム日付を検索基準日とする) | String | 任意 |
| nmflg | 名称フラグ(true:名称を取得、false、略称を取得、defaultはfalse) | boolean | 任意 |
| valueField | 値が設定されるFiled名(mstCdとして設定して値を取得する) | String | 必須 |

## ロジック層

本項では、ロジック層特有の実装方法について記載する。

## サービス層

本項では、サービス層特有の実装方法について記載する。

### サービス実装の構成

以下にServiceの継承関係を記述する。

図 4‑2　サービス実装の構成

各テーブルのServiceクラスは、ServiceGeneratedを継承する形で実装する。ServiceGeneratedには、以下のメソッドを自動生成する。

表 ‑　ServiceGeneratedに自動生成されるメソッド

|  |  |
| --- | --- |
| メソッド | 説明 |
| findById | PrimaryKeyでSelectを行います |
| lockByIdUpdateDatetime | PrimaryKeyと更新日付でSelect For Updateを行います。Select結果を戻しません。 |
| findAndLockByIdUpdateDatetime | PrimaryKeyと更新日付でSelect For Updateを行います。Select結果を返却します。 |
| findAllOrderById | PrimaryKeyをソート順として、全件を取得します |

ServiceGeneratedは、テーブルのカラムが変更になったときに、上書きできるよう、src/main/generated以下に配置する。各Serviceは上書きされないよう、src/main/java以下に配置すること。

表 ‑　AbstractServiceGeneratedのメソッド

|  |  |
| --- | --- |
| メソッド | 説明 |
| updateExcludesNull | Nullの項目を除いた形でUpdateを行う |
| insertExcludesNull | Nullの項目を除いた形でInsertを行う |
| update | Nullの項目も含めて、Updateを行う |
| insert | Nullの項目も含めて、Insertを行う |
| getPagingResultList | ページング用一覧取得 |

### 登録、更新者コード、登録更新日時等の補完

以下の項目は、AbstractServiceGeneratedの登録、更新メソッドを利用する際にEntityの値を自動的に補完します。

表 ‑　補完対象フィールド

|  |  |
| --- | --- |
| フィールド | 説明 |
| trscd | 登録者コード |
| trkid | 登録機能ＩＤ |
| trdat | 登録日時 |
| ksscd | 更新者コード |
| kskid | 更新機能ＩＤ |
| ksdat | 更新日時 |

AbstractServiceGeneratedの登録、更新メソッドを利用しない場合は、jp.co.dowa.sd.core.helper.DowaSdEntityHelperを利用して補完すること。

バッチ等の処理では、ログインユーザ≠更新者コードの場合があります。

### Oracle Char型に対する対応

Oracle Char型は、固定長文字列型である。登録、更新項目は固定長未満の長さが指定された場合Database側でスペースをpaddingされるが、Where句へ値をBindする際は固定長未満の値を指定すると、スペースがpaddingされず、条件としてmatchしないことがある。

そのため、Serviceのメソッド実行前にInterceptorで入力値に対するスペースのpadding、また、Serviceのメソッドの実行後にInterceptorでtrim処理を行う。



図 4‑3　OracleCharInterceptorの動き

OracleCharInterceptorは、以下のFieldに対して、スペースのpadding、trimを行う。実装時には留意して指定をすること。

#### Entityアノテーションが付与されているクラスにおいて、Columnアノテーションが付与されており、columnDefinition=”char(X)”が指定されているFiled。

DatabaseのカラムがChar型の場合、Entity出力の際に自動生成される。自動生成されないConditionクラス等を作成する場合には、注意すること。

/\*\*

\* 車両マスタエンティティクラス

\*

\* @author $Author: $

\* @version $Revision: $ $Date: $

\*/

@Entity

@Generated(value = {"S2JDBC-Gen 2.4.46", "org.seasar.extension.jdbc.gen.internal.model.EntityModelFactoryImpl"})

public class Zswma0 implements Serializable {

/\*\* serialVersionUID \*/

private static final long serialVersionUID = 1L;

/\*\* 製品部門 \*/

@Id

@Column(name = "SPART", columnDefinition = "char(2)", nullable = false, unique = false)

public String spart;

#### Serviceの引数にて、CharTypeアノテーションが付与されていているString型

DatabaseのカラムがChar型の場合、ServiceGenerated出力の際に自動生成される。

Serviceクラスにメソッドを追加する際は、注意すること。

public abstract class Zswma0ServiceGenerated extends AbstractServiceGenerated<Zswma0> {

/\*\*

\* 識別子でエンティティを検索します。

\*

\* @param spart

\* 識別子

\* @param ujscd

\* 識別子

\* @param syacd

\* 識別子

\* @param vehic

\* 識別子

\* @return エンティティ

\*/

public Zswma0 findById(@CharType(2) final String spart, @CharType(10) final String ujscd, @CharType(4) final String syacd, @CharType(1) final String vehic) {

return this.select().id(spart, ujscd, syacd, vehic).getSingleResult();

}

### Entity、ServiceGeneratedの自動生成

Entity、ServiceGeneraterdは、「s2jdbc-gen」を使って出力しますが、以下のプロジェクトから出力してください。S2jdbc-gen自体にカスタマイズを入れています。また、テンプレートファイルを修正しています。

<http://52.68.179.228/svn/1510_hanbai_sanpai/product/dowa_generate>

### 排他制御(SELECT ～ FOR UPDATE)メソッド

ServiceGeneraterdには、排他制御用の以下のメソッドを用意している。必要に応じて利用すること。

・lockByIdUpdateDatetime(返却値なし)

・findAndLockByIdUpdateDatetime(返却値あり)

これらのメソッドは、各テーブルの主キーと、更新日時を条件として、行排他を行う。レコードが存在しない場合(他者によって更新済み)、レコードが排他された場合(SQLエラー：ORA-61000が返却、他者がLock中)の場合、アプリケーション例外として例外をthrowする。

この例外をそのまま、画面に返却することにより、画面に警告がでる仕組みとしている。

## 層をまたいだ制御

### Form、Dto、ModelのFiledのデータ型について

データ移送に利用するbeanのFiledのデータ型については、基本的に以下の考え方で実装する。

#### Form

Formは、画面から渡された値を保持するため、すべてのFieldをString型で定義する。

#### Model

Modelは、JSPへ渡す値を保持するため、すべてのFieldをString型で定義する。

ただし、以下の場合は、String型以外を利用してもよい。

* 画面の制御に利用する情報(JSP上でifの条件句に利用)は、String型以外のJavaプリミティブ型を利用してよい。
* タグライブラリへのInputになる情報は、String型以外を利用してよい。

#### Dto、ResultDto

Dto、ResultDtoは、Logicの入出力になるため、業務のあるべきデータ型を利用する。数値の情報はBigDecimal型を利用する等。

ただし、画面入力日付は、データベース上、Char(8)で表すため、Dtoレイヤーでは、yyyyMMddの形式で保持する。

Actionレイヤーにて形式を変換し、Dtoへ移送すること。

### データの移送について

データの移送は、基本的に、org.seasar.framework.beans.util.Beansクラスを利用して行う。日付、数値等のConverterが実装されているため、必要に応じて利用すること。また、画面入力の日付型(yyyy/MM/dd)をデータベース定義の日付型(yyyyMMdd)に変換するためのConverterを用意したため、利用すること。

参照(Seaser Beans)

<http://s2container.seasar.org/2.4/ja/beans.html>

表 ‑　共通で実装したConverter

|  |  |
| --- | --- |
| Converter | 説明 |
| dateInputFormatConverter | yyyyMMddをyyyy/MM/ddに変換する |
| dateSystemFormatConverter | yyyy/MM/ddをyyyyMMddに変換する |

　共通で実装したConverterの利用イメージ。

// DtoをModelへ移送

Beans.copy(findResultDto, this.sd04211Model).dateConverter("yyyy/MM/dd hh:mm:ss", "updata")

.dateInputFormatConverter("yksdt", "ykedt").execute();

// Formから、Dtoへ移送

final Sd04211RegisterDto registerDto =

Beans.createAndCopy(Sd04211RegisterDto.class, this.editForm)

.dateSystemFormatConverter("yksdt", "ykedt").execute();

### 一覧のページングについて

一覧のページングは、以下の構成で実現する。



図 4‑4　一覧のページングの仕組み(画面→DB)

画面(JSP、CSJS)では、共通のtaglib、csjs部品を利用して、「次へ」「前へ」ボタンを配置する。また、ページング情報送信用のhiddenに値を設定する。

Actionではページング情報送信用のhidden情報を受け取り、Serviceまで移送する。Serviceでは、AbstractServiceGeneratedにページング用の共通関数が設定されており、共通関数を呼び出すことで、データ取得を行う。

Actionクラスでは、ページ内表示件数を設定する必要がある。



図 4‑5　一覧のページングの仕組み(DB→画面)

DBから取得したデータは、画面に移送して返却するが、ページング制御結果情報は、リクエストに属性として設定することで画面へ返却する。(パラメータとして返却することも可能。複数の一覧がある画面は、パラメータで返却することになる。)

画面まで返却されたページング制御結果情報は、共通taglibまたは、共通CSJSに設定することにより、再度利用することができる。

以下は、共通部品に関する説明である。

#### Tablib　dowaui:pagingControl

「次へ」「前へ」ボタンを配置する。pagingParameterIdを設定し、dowaui:pagingParameterと連携させること。

<dowaui:pagingControl onClick="dowa.sd.sd04222.search();" id="pagingCongorlTop" pagingParameterId="pagingParameter" />

表 ‑　dowaui:pagingControlの入力引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| id | Htmlのid属性 | String | 必須 |
| onClick | ボタン押下時のJavaScriptイベント(「次へ」「前へ」は同じイベントが設定される) | String | 必須 |
| parameter | ページング結果情報(設定されない場合、リクエストの属性値を利用する) | PagingControlResult | 任意 |
| pagingParameterId | 連携されるPagingParameterのid | String | 必須 |

#### Taglib　dowaui:pagingDisplayNumber

ページ数を表示する。

<dowaui:pagingDisplayNumber id="pagingNumberTop"/>

表 ‑　dowaui:pagingDisplayの入力引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| id | Htmlのid属性 | String | 必須 |
| parameter | ページング結果情報(設定されない場合、リクエストの属性値を利用する) | PagingControlResult | 任意 |

#### Taglib dowaui:pagingParameter

ページング用のhiddenパラメータ送信フィールドを作成する。

送信パラメータは、「pagingControl.currentPage」という形で送信されるが、prefixを設定することで、接頭子を与えることができる(1画面内に複数のページングが必要な場合に利用する) 。

<dowaui:pagingParameter id="pagingParameter" />

表 ‑ dowaui:pagingParameterの入力引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| id | Htmlのid属性 | String | 必須 |
| prefix | Htmのname属性接頭子 | String | 必須 |
| parameter | ページング結果情報(設定されない場合、リクエストの属性値を利用する) | PagingControlResult | 任意 |

#### CSJS(JQuery UI Widget) pagingControl

dowaui:pagingControlのCSJS(JQuery UI Widget)実装。

コンストラクタ

$('#pagingCongorlTop').pagingControl({

pagingParameterId : "pagingParameter",

total : "0",

from : "0",

to : "0",

currentPage : "0",

sortKey : "null",

sortOrder : "null"

});

表 ‑ CSJS pagingControlの入力引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| total | 総件数 | String | 必須 |
| from | ページャのFromの件数 | String | 必須 |
| to | ページャのToの件数 | String | 必須 |
| currentPage | 現在ページ数 | String | 必須 |
| sortKey | ソートキー | String | 必須 |
| sortOrder | ソート種別(Asc or Desc) | String | 必須 |

setControl関数

$('#pagingCongorlTop').pagingControl('setControl', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

"to" : resultObject.pagingControl.numTo,

"currentPage" : resultObject.pagingControl.currentPage,

"sortKey" : resultObject.pagingControl.sortKey,

"sortOrder" : resultObject.pagingControl.sortOrder,

});

入力引数は、コンストラクタと同様。

#### CSJS(JQuery UI Widget)　pagingNumber

dowaui:pagingNumberのCSJS(JQuery UI Widget)実装。

コンストラクタ

$('#pagingNumberTop').pagingNumber({

total : "0",

from : "0",

to : "0"

});

表 4‑12　CSJS pagingNumberの入力引数

|  |  |  |  |
| --- | --- | --- | --- |
| 引数 | 説明 | データ型 | 必須／任意 |
| total | 総件数 | String | 必須 |
| from | ページャのFromの件数 | String | 必須 |
| to | ページャのToの件数 | String | 必須 |

setNumber関数

$('#pagingNumberTop').pagingNumber('setNumber', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

"to" : resultObject.pagingControl.numTo

});

入力引数は、コンストラクタと同様。

#### jp.co.dowa.sd.core.dto.PagingControl

画面からDBへ値を移送するためのDtoクラス。Dtoクラスではあるが、例外的に、どのレイヤーでも利用してよい。

#### jp.co.dowa.sd.core.dto.PagingControlResult

DBから画面へ値を移送するためのDtoクラス。Dtoクラスではあるが、例外的にどのレイヤーでも利用してよい。

#### jp.co.dowa.sd.service.AbstractServiceGenerated#getPagingResultList

移送されたPagingControlクラスの情報をもとに、PagingControlResultクラスを生成し、総件数、DBからデータを取得し返却する。

### ログインユーザ情報の利用について

ログインユーザ情報のは、jp.co.dowa.sd.util.ContextUtilsから取得して利用すること。ログインユーザ情報は、LoginInterCeptorにて取得して設定している。

intra-martのAPIからも取得できるが、特定の処理の場合(バッチ処理等)に読み替える必要が出てくるデータが存在する。その内容をLoginInterceptorやAPI内で補完するため、必ずAPI経由で取得すること。

ContextsUtilsからは、以下の情報が取得できる。

表 ‑　ContextsUtilsから取得できるデータ

|  |  |  |
| --- | --- | --- |
| 引数 | 説明 | データ型 |
| userId | ログインユーザID | String |
| locale | ロケール | Locale |

追加すべき情報がある場合は、駒田まで連絡ください。

### 例外処理

例外は、「システム方式設計書」のとおり、アプリケーション例外と、システム例外の2つに分類する。

表 ‑　例外の分類

|  |  |
| --- | --- |
| 例外分類 | 説明 |
| アプリケーション例外 | ユーザーが再度、同じ手順を実行すると正常終了する可能性がある例外  ・入力値の妥当性検証(バリデーション)エラー  ・入力値の業務整合性チェックエラー  ・排他エラー等 |
| システム例外 | 回復不可能な例外  ・ハードウェア故障  ・サーバーダウン  ・アプリケーションとして想定しないデータ整合性エラー等 |

#### アプリケーション例外の実装

アプリケーション例外は、以下の通り実装する。

アプリケーション例外の実装

final int result = this.zswma0Service.updateExcludesNull(entity);

if (result != 1) {

throw new DowaSdApplictaionException(“message”);

}

アプリケーション例外は、エラー文言を画面に表示するため、インスタンス生成時に必ずメッセージを指定すること。また、例外をcatchしてthrowする場合は、causeを必ず設定する。

#### システム例外の実装

システム例外は、以下の通り実装する。

final int result = this.zswma0Service.updateExcludesNull(entity);

if (result != 1) {

throw new DowaSdSystemException();

}

システム例外は、エラー文言を画面に表示しないため、インスタンス生成時にメッセージを指定する必要はない。また、例外をcatchしてthrowする場合は、causeを必ず設定する。

#### 例外の制御

例外がthrowされた場合は、Actionクラスの前後で動くInterceptorにてcatchをし、エラー制御が自動的に行われる。

## セキュリティ

### SQLインジェクション対策

S2JDBCのJdbcManagerを利用し、データアクセスを行うこと。また、JdbcManagerを利用する際に、画面から送信された値を、文字列として結合してSQL文として生成してはならない。

OK

this.select().id(spart, mtmcd, mesno).where(

"name = ?",

name); // nameが画面送信値

NG

this.select().id(spart, mtmcd, mesno).where(

"name = " + name); // nameが画面送信値

### クロスサイト・スクリプティング対策

JSPの実装において、サーバからの送信値を利用する箇所では、必ず無害化(sanitizing)を行わなければならない。

無害化の例：EL式f:h()にて無害化を実施

<input type="text" id="matnr" name="matnr" value="${f:h(sd02101Model.matnr)}" maxlength="18"/>

※taglib等のロジック内部で無害化されていない部分がありますが、そのうち直します。

### クロスサイト・リクエストフォージェリ対策

クロスサイト・リクエストフォージェリの対策は、「登録」「更新」等のデータベース更新を伴う処理のみを対象とする。データベース更新を伴わない処理は、対策を行わない。

クロスサイト・リクエストフォージェリの対策は、以下の概念で行う。



図 4‑6　クロスサイト・リクエストフォージェリ対策の概念

上記の概念を実現するうえで、以下の部品を用意している。それを利用して実装すること。

#### Tokenを発行し、Sessionに格納

Token制御用のIntercetorを用意している。Actionクラスに、SaveTokenアノテーションを付与すると、自動的にTokenを発行しSesseionに格納する。

/\*\*

\* 車両台帳登録 初期表示 Action.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

@SaveToken

public class IndexAction {

#### Tokenを画面のform部分にhiddenで設定

dowaui:formタグを用意している。左記タグを利用してformを生成すること。ただし、画面送信に利用するformのみに利用すること。

dowaui:formタグを利用する。

<dowaui:form id="sd04211Form" name="sd04211Form" method="post">

・・・入力項目等

</dowaui:form>

<form id="sd04211EditForm" name="sd04211EditForm" method="post" action="${f:url('/sd04211/edit/')}">

こちらは、URLの生成に利用しているだけなので、dowaui:formを利用しない。

</form>

<form id="sd04211ReadForm" name="sd04211ReadForm" method="post" action="${f:url('/sd04211/')}">

</form>

#### Sessionに格納されたTokenと画面から捜査員されたTokenを比較し、チェック

Token制御用のIntercetorを用意している。Actionクラスに、TokenValidアノテーションを付与すると、自動的にTokenをチェックする。エラーの場合は、システムエラー画面に遷移する。

/\*\*

\* 車両台帳登録 登録 Action.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

@TokenValid

public class EditAction {

# 実装イメージの解説

本項では、前項で述べた、各レイヤーについての詳細を記述する。

## 画面の初期表示

車両台帳登録の初期表示部分の実装イメージを例示する。

### 画面遷移イメージ

以下のように、遷移してくる想定で実装する。



図 5‑1　車両台帳登録の初期表示時の画面遷移

### ソースの構成

以下のようにソースを構成する。なお、初期表示時は、他画面から遷移してくるため、クライアント層は利用しない。また、サービス層は、Service以外は自動生成のため記載しない。

|  |  |  |
| --- | --- | --- |
| レイヤー | 主要クラス | ソース |
| プレゼンテーション層 | Action | jp.co.dowa.sd.action.sd04211.IndexAction |
| Form | jp.co.dowa.sd.form.sd04211.IndexForm |
| Model | jp.co.dowa.sd.model.sd04211.Sd04211Model |
| JSP | /jp/co/dowa/sd/sd04211/index.jsp |
| ロジック層 | Logic | jp.co.dowa.sd.logic.sd04211.Sd04211Logic |
| Component | jp.co.dowa.sd.component.sd04211.Sd04211DbAccessComponent |
| Dto | jp.co.dowa.sd.dto.sd04211.Sd04211FindDto |
| ResultDto | jp.co.dowa.sd.dto.sd04211.Sd04211FindResultDto |
| サービス層 | Service | jp.co.dowa.sd.service. Zswma0Service |

### プレゼンテーション層

#### Action

/\*

Copyright表記を入れること。

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.action.sd04211;

import jp.co.dowa.sd.core.exception.DowaSdApplicationException;

import jp.co.dowa.sd.core.exception.DowaSdValidateException;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindResultDto;

import jp.co.dowa.sd.enums.ScreenTypeEnum;

import jp.co.dowa.sd.form.sd04211.IndexForm;

ちゃんとEclipseの設定をしてあれば、自動的にきれいに並びます。

import jp.co.dowa.sd.logic.sd04211.Sd04211Logic;

import jp.co.dowa.sd.model.sd04211.Sd04211Model;

import org.apache.commons.lang.StringUtils;

import org.seasar.framework.beans.util.Beans;

import org.seasar.struts.annotation.ActionForm;

import org.seasar.struts.annotation.Execute;

import javax.annotation.Resource;

/\*\*

\* 車両台帳登録 初期表示 Action.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

JSPの指定はwebapps/WEB-INF/view/の下から記載する。

\*/

public class IndexAction {

private static final String INDEX = "/jp/co/dowa/sd/sd04211/index.jsp";

@ActionForm

@Resource

public IndexForm indexForm;

@Resource

public Sd04211Logic sd04211Logic;

public Sd04211Model sd04211Model;

/\*\*

\* indexメソッド.

\*

\* @return 遷移先

初期表示の場合、エラーが発生した際戻り先がないため、システムエラー画面へ遷移する。

\* @throws DowaSdValidateException 選択必須入力チェックエラー

\* @throws DowaSdApplicationException アプリケーション例外

\*/

@Execute(input = "/jp/co/dowa/sd/core/systemerror.jsp")

public String index() throws DowaSdValidateException, DowaSdApplicationException {

// 選択必須入力チェック

if (!StringUtils.equals(this.indexForm.screenType, ScreenTypeEnum.REGIST.getValue())) {

// 製品部門

if (StringUtils.isEmpty(this.indexForm.spart)) {

// TODO:例外メッセージの送信方法

throw new DowaSdValidateException();

エラーの場合のメッセージ送信方法は、検討中。

例外クラスにメッセージを詰め、Interceptorで自動的に展開するつもり。

選択必須等、画面にまつわる関連チェックは、Actionでチェックする。

}

// 運搬事業者コード

if (StringUtils.isEmpty(this.indexForm.ujscd)) {

// TODO:例外メッセージの送信方法

throw new DowaSdValidateException();

}

// 車両コード

if (StringUtils.isEmpty(this.indexForm.syacd)) {

// TODO:例外メッセージの送信方法

throw new DowaSdValidateException();

}

// 車両枝番

if (StringUtils.isEmpty(this.indexForm.vehic)) {

// TODO:例外メッセージの送信方法

throw new DowaSdValidateException();

}

画面操作共通チェック(アクセス権限のチェック)は、共通処理として実装予定。今は未実装でOKです。

}

// 画面操作共通チェック

// TODO:未実装

// 画面返却値の作成(引継ぎ項目コピー)

this.sd04211Model = Beans.createAndCopy(Sd04211Model.class, this.indexForm).execute();

// 更新モードの場合、初期表示情報を取得する

if (!StringUtils.equals(this.indexForm.screenType, ScreenTypeEnum.REGIST.getValue())) {

final Sd04211FindDto findDto =

org.seasar.framework.beans.util.Beansで、同じfiled名の値はコピーできます。積極的に使ってください。

また、値の変換、除外設定等もできます。以下、参照のこと。

<http://s2container.seasar.org/2.4/ja/beans.html>

Beans.createAndCopy(Sd04211FindDto.class, this.indexForm).execute();

final Sd04211FindResultDto findResultDto = this.sd04211Logic.find(findDto);

Beans.copy(findResultDto, this.sd04211Model).dateConverter("yyyy/MM/dd hh:mm:ss", "updata")

.dateInputFormatConverter("yksdt", "ykedt").execute();

}

return INDEX;

}

}

#### Form

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.form.sd04211;

import lombok.ToString;

import org.seasar.struts.annotation.Arg;

import org.seasar.struts.annotation.Required;

import java.io.Serializable;

/\*\*

\* 車両台帳登録 初期表示 Form.

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

LombokのToStringアノテーションを付与します。

\*/

@ToString

public class IndexForm implements Serializable {

/\*\*

\* コンストラクタ.

\*/

画面項目の順序に従って

Fieldを並べてください。

public IndexForm() {}

/\*\* 製品部門 . \*/

public String spart;

/\*\* 運搬事業者コード. \*/

public String ujscd;

以下にValidatorの一覧があります。

http://52.68.179.228/svn/1510\_hanbai\_sanpai/document/40\_製造（M）/01\_調査・検討/開発ガイドライン/Validator一覧.xlsx

足りないものがあったら、駒田まで。

/\*\* 車両コード. \*/

public String syacd;

/\*\* 車両枝番号. \*/

public String vehic;

/\*\* 画面モード. \*/

@Required(arg0 = @Arg(key = "画面モード", resource = false) )

public String screenType;

/\*\* 前画面検索条件. TODO:項目名未定. \*/

public String preSearchConditions;

}

#### Model

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.model.sd04211;

import jp.co.dowa.sd.interceptor.anotation.BaseMasterSupplementInfo;

import jp.co.dowa.sd.interceptor.anotation.BaseMasterSupplementName;

import lombok.Data;

import java.io.Serializable;

import java.util.List;

import java.util.Map;

/\*\*

\* 車両台帳登録 Model.

\*

基本マスタ取得のアノテーションは、以下のように利用してください。

ClassにDataアノテーションを付けます。そうすると、Lombokが見えないところでgetter、setterを作ってくれます。(アウトラインで見えます。)

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

@Data

public class Sd04211Model implements Serializable {

/\*\*

\* コンストラクタ.

\*/

public Sd04211Model() {}

/\*\* 運搬事業者プルダウン. \*/

@BaseMasterSupplementInfo(cdtyp = "001", spart = "01", targetkbn = "1", valueField = "ujscd",

nmflg = true)

private List<Map<String, String>> ujsCdSelect;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 運搬事業者名称. \*/

private String name6;

/\*\* 製品部門名称. \*/

private String spart;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号. \*/

private String vehic;

/\*\* 最終更新日. \*/

private String updata;

/\*\* 車両番号コード. \*/

private String sytno;

/\*\* 販売組織コード. \*/

private String nyrvk;

/\*\* 販売組織名称. \*/

private String name1;

/\*\* 車種コード. \*/

private String syscd;

/\*\* 車種名称. \*/

@BaseMasterSupplementName(spart = "ZZ", cdtyp = "12", ms1kb = "5", targetkbn = "1",

valueField = "syscd")

private String mstnm;

/\*\* 最終更新者. \*/

private String hksha;

/\*\* 有効期間\_有効開始日. \*/

private String yksdt;

/\*\* 有効期間\_有効終了日. \*/

private String ykedt;

/\*\* 削除フラグ\_フラグ. \*/

private boolean ykedtOver;

/\*\* 最大積載量\_重量. \*/

private String sdsam;

/\*\* 車両重量\_重量. \*/

private String syaam;

/\*\* 車両総重量\_重量. \*/

private String srsam;

/\*\* 指図番号\_指図. \*/

private String aufnr;

/\*\* 画面モード. // TODO:項目名未定. \*/

private String screenType;

/\*\* 前画面検索条件. TODO:項目名未定. \*/

private String preSearchConditions;

}

#### JSP

<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ taglib prefix="imarttag" uri="http://www.intra-mart.co.jp/taglib/foundation/imarttag"%>

<%@ taglib prefix="imart" uri="http://www.intra-mart.co.jp/taglib/core/standard"%>

<%@ taglib prefix="html" uri="http://struts.apache.org/tags-html"%>

Tilesでlayout.jspを読み込みます。

Common.css、Jquery等が自動的にloadされます。

<%@ taglib prefix="f" uri="http://sastruts.seasar.org/functions"%>

<%@ taglib prefix="tiles" uri="http://struts.apache.org/tags-tiles"%>

<%@ taglib prefix="dowaui" uri="http://dowa.co.jp/core/tags"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<tiles:insert template="/WEB-INF/view/jp/co/dowa/sd/layout/layout.jsp" flush="true">

<tiles:put name="title" value="${dowaui:message('jp.co.dowa.sd.sd04211.caption.001')}${dowaui:message('jp.co.dowa.sd.common.caption.001')}" />

titleタグになります。ポップアップ等は設定が必要です。

<tiles:put name="header" type="string">

<script src="${f:url('/dowa/csjs/sd04211/sd04211.js')}"></script>

Tiles:put name=”header”内は、

headerタグ内に設定されます。

<script id="disposalPermitTemplate" type="text/x-jsrender">

</script>

</tiles:put>

Tiles:put name=”content”内は、bodyタグ内に設定されます。

<tiles:put name="content" type="string">

<div class="ui-form-container">

<div class="ui-title">

<imarttag:imartTitleBar title="${dowaui:message('jp.co.dowa.sd.sd04211.caption.001')}" icon="/images/standard/title.gif" />

ボタンへのイベントの割り付けは、基本的にjsファイルでやりますが、ここは、できないため、

直接hrefに入れてよいです。

<imarttag:imartToolbarFrame>

<imarttag:imartToolbarLeft>

<c:if test="${sd04211Model.screenMode=='1'}">

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.001')}"

icon="/images/icons/16x16/fugue-icons/shadow/application-task.png" href="Javascript:dowa.sd.sd04211.onClickRegisterBtn();" />

<span class="separate\_bar"></span>

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.003')}"

icon="/images/icons/16x16/fugue-icons/shadow/application-task.png" href="javascript:dowa.sd.sd04211.onClickConfirmBtn();" />

</c:if>

<c:if test="${sd04211Model.screenMode=='2'}">

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.002')}"

icon="/images/icons/16x16/fugue-icons/shadow/application-task.png" href="Javascript:dowa.sd.sd04211.onClickUpdateBtn();" />

<span class="separate\_bar"></span>

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.003')}"

icon="/images/icons/16x16/fugue-icons/shadow/application-task.png" href="javascript:dowa.sd.sd04211.onClickConfirmBtn();" />

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.004')}"

icon="/images/icons/16x16/fugue-icons/shadow/application-task.png" href="javascript:dowa.sd.sd04211.onClickCorrectionBtn();" />

</c:if>

</imarttag:imartToolbarLeft>

<imarttag:imartToolbarRight>

<imarttag:imartIcon name="${dowaui:message('jp.co.dowa.sd.common.caption.005')}"

icon="/images/icons/16x16/agt\_back.png" href="javascript:sd.mockup.sd04211.back()" />

</imarttag:imartToolbarRight>

</imarttag:imartToolbarFrame>

</div>

<form id="sd04211Form" name="sd04211Form" method="post">

<div id=errorDiv>

<span id=msg> <html:errors />

</span>

</div>

<input type="hidden" id="screenMode" name="screenMode" value="${sd04211Model.screenMode}" />

CSSは共通CSSリファレンスを見ながら設定してください。

Taglibは、taglib一覧を見ながら設定してください。

<table class="ui-form">

<colgroup>

<col class="wd-13">

<col class="wd-37">

<col class="wd-13">

<col class="wd-37">

</colgroup>

<tr>

<c:if test="${sd04211Model.screenMode=='1'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.002')}</label></th>

<td><dowaui:generalPurposeSearchSingle value="${f:h(sd04211Model.ujscd)}" id="ujscd" name="ujscd" maxlength="10"

HTMLに出力される部分はエスケープをしてください。

class="wd-23" nameValue="${f:h(sd04211Model.name6)}" namearea\_name="name6" namearea\_class="wd-60"

namearea\_maxlength="40" screenMode="${f:h(sd04211Model.screenMode)}" /></td>

</c:if>

<c:if test="${sd04211Model.screenMode!='1'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.002')}</label></th>

<td><dowaui:generalPurposeSearchSingle value="${f:h(sd04211Model.ujscd)}" id="ujscd" name="ujscd" maxlength="10"

class="wd-23" nameValue="${f:h(sd04211Model.name6)}" namearea\_name="name6" namearea\_class="wd-60"

namearea\_maxlength="40" screenMode="3" disabled/></td>

</c:if>

<c:if test="${sd04211Model.screenMode=='1'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.003')}</label></th>

<td><dowaui:productDeptSelect value="${f:h(sd04211Model.spart)}" iphkb="" gmkcd="" id="spart" name="spart" width="200px"/></td>

</c:if>

<c:if test="${sd04211Model.screenMode!='1'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.003')}</label></th>

<td><dowaui:productDeptSelect value="${f:h(sd04211Model.spart)}" iphkb="" gmkcd="" id="spart" name="spart" width="200px" disabled/></td>

</c:if>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode=='1'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.004')}</label></th>

<td>

<input type="text" class="ui-text wd-10" name="syacd" id="syacd" maxlength="4" value="${f:h(sd04211Model.syacd)}" />

<input type="text" class="ui-text wd-5" name="vehic" id="vehic" maxlength="1" value="${f:h(sd04211Model.vehic)}" />

</td>

</c:if>

<c:if test="${sd04211Model.screenMode!='1'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.004')}</label></th>

<td>

<input type="text" class="ui-text-readonly wd-10" name="syacd" id="syacd" maxlength="4" value="${f:h(sd04211Model.syacd)}" readonly />

<input type="text" class="ui-text-readonly wd-5" name="vehic" id="vehic" maxlength="1" value="${f:h(sd04211Model.vehic)}" readonly />

</td>

</c:if>

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.005')}</label></th>

<td>

<input type="text" class="ui-text-readonly wd-27" name="updata" id="updata" maxlength="20" value="${f:h(sd04211Model.updata)}" readonly />

</td>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.006')}</label></th>

<td><input type="text" class="ui-text wd-42" name="sytno" id="sytno" maxlength="20" value="${f:h(sd04211Model.sytno)}" /></td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.006')}</label></th>

<td><input type="text" class="ui-text-readonly wd-42" name="sytno" id="sytno" maxlength="20" value="${f:h(sd04211Model.sytno)}" readonly /></td>

</c:if>

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.007')}</label></th>

<td><dowaui:salesDeptSearchSingle value="${f:h(sd04211Model.nyrvk)}" id="nyrvk" name="nyrvk" class="wd-10"

maxlength="4" nameValue="${f:h(sd04211Model.name1)}" namearea\_class="wd-34" namearea\_maxlength="16" screenMode="3" disabled/></td>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.008')}</label></th>

<td><dowaui:baseMasterSearchMulti id="syscd" name="syscd" value="${f:h(sd04211Model.syscd)}" class="wd-10"

maxlength="3" nameValue="${f:h(sd04211Model.mstnm)}" namearea\_class="wd-35" namearea\_maxlength="40"

namearea\_name="mstnm" spart="ZZ" cdtyp="12" ykdt="20150202" screenMode="${f:h(sd04211Model.screenMode)}" acquisitionDest="1" chooseFunction="test"/></td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.008')}</label></th>

<td><dowaui:baseMasterSearchMulti id="syscd" name="syscd" value="${f:h(sd04211Model.syscd)}" class="wd-10"

maxlength="3" nameValue="${f:h(sd04211Model.mstnm)}" namearea\_class="wd-35" namearea\_maxlength="40"

namearea\_name="mstnm" spart="ZZ" cdtyp="12" screenMode="${f:h(sd04211Model.screenMode)}" acquisitionDest="1" disabled/></td>

</c:if>

<th>${dowaui:message('jp.co.dowa.sd.sd04211.caption.009')}</th>

<td><input type="text" class="ui-text-readonly wd-27" name="hksha" id="hksha" maxlength="12" value="${f:h(sd04211Model.hksha)}" readonly /></td>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.010')}</label></th>

<td>

<input type="text" class="ui-text wd-22" name="yksdt" id="yksdt" maxlength="10" value="${f:h(sd04211Model.yksdt)}">

<imart:floatingCalendar element="document.sd04211Form.yksdt">

<imarttag:imartIcon icon="/images/icons/16x16/fugue-icons/shadow/calendar-blue.png" />

</imart:floatingCalendar>

<imarttag:imartIcon icon="/images/icons/16x16/deletecell.png" href="javascript:void(0)" name=""

attr="onclick='dowa.sd.core.common.clearInputTd(this, \"yksdt\")'" />

<input type="text" class="ui-text wd-22" name="ykedt" id="ykedt" maxlength="10" value="${f:h(sd04211Model.ykedt)}">

<imart:floatingCalendar element="document.sd04211Form.ykedt">

<imarttag:imartIcon icon="/images/icons/16x16/fugue-icons/shadow/calendar-blue.png" />

</imart:floatingCalendar>

<imarttag:imartIcon icon="/images/icons/16x16/deletecell.png" href="javascript:void(0)" name=""

attr="onclick='dowa.sd.core.common.clearInputTd(this, \"ykedt\")'" />

</td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.010')}</label></th>

<td>

<input type="text" class="ui-text-readonly wd-22" name="yksdt" id="yksdt" maxlength="10" value="${f:h(sd04211Model.yksdt)}" readonly>

<label>～</label>

<input type="text" class="ui-text-readonly wd-22" name="ykedt" maxlength="10" value="${f:h(sd04211Model.ykedt)}" readonly>

</td>

</c:if>

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.011')}</label></th>

<td><input type="checkbox" name="ykedtOver" id="ykedtOver" disabled <c:if test="${sd04211Model.ykedtOver}">checked</c:if> /></td>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.012')}</label></th>

<td colspan="3">

<input type="text" class="ui-text wd-6" name="sdsam" id="sdsam" maxlength="5" value="${f:h(sd04211Model.sdsam)}">

<label>kg</label>

</td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.012')}</label></th>

<td colspan="3">

<input type="text" class="ui-text-readonly wd-6" name="sdsam" id="sdsam" maxlength="5" value="${f:h(sd04211Model.sdsam)}" readonly>

<label>kg</label>

</td>

</c:if>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.013')}</label></th>

<td colspan="3">

<input type="text" class="ui-text wd-6" name="syaam" id="syaam" maxlength="5" value="${f:h(sd04211Model.syaam)}">

<label>kg</label></td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.013')}</label></th>

<td colspan="3">

<input type="text" class="ui-text-readonly wd-6" name="syaam" id="syaam" maxlength="5" value="${f:h(sd04211Model.syaam)}" readonly>

<label>kg</label></td>

</c:if>

</tr>

<tr>

<c:if test="${sd04211Model.screenMode!='3'}">

<th><label class="ui-require">${dowaui:message('jp.co.dowa.sd.sd04211.caption.014')}</label></th>

<td colspan="3">

<input type="text" class="ui-text wd-6" name="srsam" id="srsam" maxlength="5" value="${f:h(sd04211Model.srsam)}">

<label>kg</label>

</td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.014')}</label></th>

<td colspan="3">

<input type="text" class="ui-text-readonly wd-6" name="srsam" id="srsam" maxlength="5" value="${f:h(sd04211Model.srsam)}" readonly>

<label>kg</label>

</td>

</c:if>

</tr>

<tr>

<th><label>${dowaui:message('jp.co.dowa.sd.sd04211.caption.015')}</label></th>

<c:if test="${sd04211Model.screenMode!='3'}">

<td colspan="3">

<dowaui:generalPurposeSearchSingle value="${f:h(sd04211Model.aufnr)}" id="aufnr" name="aufnr" maxlength="12"

class="wd-12" screenMode="${f:h(sd04211Model.screenMode)}" />

</td>

</c:if>

<c:if test="${sd04211Model.screenMode=='3'}">

<td colspan="3">

<dowaui:generalPurposeSearchSingle value="${f:h(sd04211Model.aufnr)}" id="aufnr" name="aufnr" maxlength="12"

class="wd-12" screenMode="${f:h(sd04211Model.screenMode)}" disabled />

</td>

</c:if>

</tr>

</tbody>

</table>

</form>

<form id="sd04211EditForm" name="sd04211EditForm" method="post" action="${f:url('/sd04211/edit/')}"></form>

<form id="sd04211ConfirmForm" name="sd04211ConfirmForm" method="post" action="${f:url('/sd04211/edit/confirm/')}"></form>

<form id="sd04211ReadForm" name="sd04211ReadForm" method="post" action="${f:url('/sd04211/')}"></form>

</div>

</tiles:put>

</tiles:insert>

### ロジック層

#### Logic

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.logic.sd04211;

import jp.co.dowa.sd.component.sd04211.Sd04211DbAccessComponent;

import jp.co.dowa.sd.core.exception.DowaSdSystemException;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindResultDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211RegisterDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211UpdateDto;

import jp.co.dowa.sd.entity.Zswma0;

import jp.co.dowa.sd.service.Zswma0Service;

import jp.co.dowa.sd.util.DateUtils;

import org.seasar.framework.beans.util.Beans;

import javax.annotation.Resource;

/\*\*

\* 車両台帳登録 業務ロジック.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

public class Sd04211Logic {

@Resource

Sd04211DbAccessComponent sd04211DbAccessComponent;

@Resource

public Zswma0Service zswma0Service;

/\*\*

\* 車両台帳を取得する.

\*

\* @param findDto パラメータ

\* @return 処理結果

\*/

public Sd04211FindResultDto find(final Sd04211FindDto findDto) {

final Zswma0 zswma0 = this.zswma0Service.findById(findDto.getSpart(), findDto.getUjscd(),

findDto.getSyacd(), findDto.getVehic());

if (null == zswma0) {

// TODO:エラー処理

throw new DowaSdSystemException();

}

final Sd04211FindResultDto result =

Beans.createAndCopy(Sd04211FindResultDto.class, zswma0).execute();

final int periodCheckResult =

DateUtils.periodCheck(DateUtils.getSystemDate(), result.getYksdt(), result.getYkedt());

if (periodCheckResult != 0) {

result.setYkedtOver(true);

}

return result;

}

#### Component

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.component.sd04211;

import jp.co.dowa.sd.entity.Zswma0;

import jp.co.dowa.sd.util.DateUtils;

/\*\*

\* 車両台帳登録 DBアクセス関連.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

public class Sd04211DbAccessComponent {

/\*\*

\* 表示用更新情報を補完する.

\*

\* @param entity 車両台帳Entity

\*/

public void completeUpdateInfo(final Zswma0 entity) {

// TODO:表示用更新組織、更新者の取得

entity.hjkvk = "dep1";

entity.hksha = "aoyagi";

entity.updata = DateUtils.getSystemDate();

}

}

#### Dto

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.dto.sd04211;

import lombok.Data;

import java.io.Serializable;

ClassにDataアノテーションを付けます。そうすると、Lombokが見えないところでgetter、setterを作ってくれます。(アウトラインで見えます。)

/\*\*

\* 車両台帳登録 find用Dto.

\*

\*/

@Data

public class Sd04211FindDto implements Serializable {

/\*\*

\* コンストラクタ.

\*/

public Sd04211FindDto() {}

/\*\* 製品部門 . \*/

private String spart;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号 // TODO:項目名未定. \*/

private String syaSeqno;

}

#### ResultDto

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.dto.sd04211;

import lombok.Data;

import java.io.Serializable;

import java.sql.Timestamp;

/\*\*

\* 車両台帳登録 find用Dto.

\*

\*/

@Data

public class Sd04211FindResultDto implements Serializable {

/\*\* 製品部門 . \*/

private String spart;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号. \*/

private String vehic;

/\*\* 更新日時 . \*/

private Timestamp updata;

/\*\* 車両登録番号. \*/

private String sytno;

/\*\* 販売組織. \*/

private String nyrvk;

/\*\* 車種コード. \*/

private String syscd;

/\*\* 最終更新者 . \*/

private String hksha;

/\*\* 有効開始日. \*/

private String yksdt;

/\*\* 有効終了日. \*/

private String ykedt;

/\*\* 削除フラグ. TODO:DB項目にない。 \*/

private boolean ykedtOver;

/\*\* 最大積載量. \*/

private String sdsam;

/\*\* 車両重量. \*/

private String syaam;

/\*\* 車両総重量. \*/

private String srsam;

/\*\* 指図番号. \*/

private String aufnr;

}

### サービス層

#### Service

package jp.co.dowa.sd.service;

import jp.co.dowa.sd.condition.Zswma0.Zswma0R01Condition;

import jp.co.dowa.sd.entity.Zswma0;

import javax.annotation.Generated;

/\*\*

\* {@link Zswma0}のサービスクラスです。

\*

\*/

@Generated(

ServiceGeneratedを継承して使います。

value = {"S2JDBC-Gen 2.4.46",

"org.seasar.extension.jdbc.gen.internal.model.ServiceModelFactoryImpl"},

date = "2015/12/03 19:16:42")

public class Zswma0Service extends Zswma0ServiceGenerated {

private static final String ZSWMA\_R001 = "ZSWMA\_R001.sql";

/\*\*

\* 車両台帳を取得します.

\*

\* @param condition 取得条件

\* @return 車両台帳データ

基本的にSQLファイルを使ってSQLを実行します。

\*/

public Zswma0 r001(final Zswma0R01Condition condition) {

return this.selectBySqlFile(Zswma0.class, Zswma0Service.ZSWMA\_R001, condition)

.getSingleResult();

}

}

## 画面の登録／更新／削除機能

車両台帳登録の登録部分の実装イメージを例示する。

### 画面遷移イメージ

以下のように、遷移してくる想定で実装する。



図 5‑2　車両台帳登録の登録時の画面遷移

### ソースの構成

以下のようにソースを構成する。なお、初期表示時は、他画面から遷移してくるため、クライアント層は利用しない。また、サービス層は、Service以外は自動生成のため記載しない。

表 ‑ 車両台帳登録の初期表示時のソース構成

|  |  |  |
| --- | --- | --- |
| レイヤー | 主要クラス | ソース |
| クライアント層 | CSJS | /dowa/csjs/sd04211/sd04211.js |
| プレゼンテーション層 | Action | jp.co.dowa.sd.action.sd04211.EditAction |
| Form | jp.co.dowa.sd.form.sd04211.IndexForm |
| Form | jp.co.dowa.sd.form.sd04211.EditForm |
| JSP | /jp/co/dowa/sd/sd04211/index.jsp |
| ロジック層 | Logic | jp.co.dowa.sd.logic.sd04211.Sd04211Logic |
| Dto | jp.co.dowa.sd.dto.sd04211.Sd04211RegisterDto |
| サービス層 | Service | jp.co.dowa.sd.service. Zswma0Service |

### クライアント層

#### CSJS

var dowa = dowa || {};

名前空間の宣言をします。

dowa.sd = dowa.sd || {};

dowa.sd.sd04211 = {};

EL式での補完ができないため、JSP側にURLを書いてください。

(function($) {

dowa.sd.sd04211.onClickEditBtn = function() {

$('#sd04211Form').attr('action', $('#sd04211EditForm').attr('action'));

共通部品ですが、内部でFormをserializeしてサーバに送信します。

dowa.sd.core.request.ajaxSubmit($('#sd04211Form'), $('#sd04211ReadForm').attr('action'));

};

dowa.sd.sd04211.onClickComfirmBtn = function() {

dowa.sd.core.request.ajaxSubmit($('#sd04211Form'));

};

})(jQuery);

### プレゼンテーション層

#### Action

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.action.sd04211;

import jp.co.dowa.sd.core.exception.DowaSdValidateException;

import jp.co.dowa.sd.dto.sd04211.Sd04211RegisterDto;

import jp.co.dowa.sd.enums.ScreenTypeEnum;

import jp.co.dowa.sd.form.sd04211.EditForm;

import jp.co.dowa.sd.form.sd04211.IndexForm;

import jp.co.dowa.sd.logic.sd04211.Sd04211Logic;

import net.arnx.jsonic.JSON;

import org.apache.commons.lang.StringUtils;

import org.seasar.framework.beans.util.Beans;

import org.seasar.struts.annotation.ActionForm;

import org.seasar.struts.annotation.Execute;

import org.seasar.struts.util.ResponseUtil;

import java.util.HashMap;

import java.util.Map;

import javax.annotation.Resource;

/\*\*

\* 車両台帳登録 登録 Action.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

public class EditAction {

@ActionForm

@Resource

public EditForm editForm;

@Resource

public Sd04211Logic sd04211Logic;

/\*\*

入力エラーの場合は、左記のURLへ遷移させます。このJSPは、結果をJSONの形に成形して、画面に返却します。

\* indexメソッド.

\*

\* @return 遷移先

\* @throws DowaSdValidateException 選択必須入力チェックエラー

\*/

@Execute(input = "/jp/co/dowa/sd/core/jsonerror.jsp")

public String index() throws DowaSdValidateException {

// 共通権限チェック

// TODO:未実装

// RQ-3 4の業務処理を行う

// TODO:未実装

if (StringUtils.equals(this.editForm.screenType, ScreenTypeEnum.REGIST.getValue())) {

final Sd04211RegisterDto registerDto =

FormのデータをDtoに移送し、Logicを呼び出します。

Beans.createAndCopy(Sd04211RegisterDto.class, this.editForm).execute();

this.sd04211Logic.register(registerDto);

} else if (StringUtils.equals(this.editForm.screenType, ScreenTypeEnum.EDIT.getValue())) {

画面入力データを次画面(同じ画面の初期表示を参照モードで開く)向けに格納します。

System.out.println("更新モード");

}

// 画面返却値の作成(引継ぎ項目コピー)

final IndexForm indexForm = Beans.createAndCopy(IndexForm.class, this.editForm).execute();

indexForm.screenType = ScreenTypeEnum.READ.getValue();

final Map<String, Object> responseData = new HashMap<String, Object>();

responseData.put("error", false);

responseData.put("data", indexForm);

データをResponseに詰め、返却します。ここは、共通化する予定。

final String jsonText = JSON.encode(responseData);

ResponseUtil.write(jsonText, "application/json");

return null;

}

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.action.sd04211;

import jp.co.dowa.sd.core.exception.DowaSdValidateException;

import jp.co.dowa.sd.dto.sd04211.Sd04211RegisterDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211UpdateDto;

import jp.co.dowa.sd.enums.ScreenTypeEnum;

import jp.co.dowa.sd.form.sd04211.EditForm;

import jp.co.dowa.sd.form.sd04211.IndexForm;

import jp.co.dowa.sd.logic.sd04211.Sd04211Logic;

import jp.co.dowa.sd.util.ResponseUtils;

import org.apache.commons.lang.StringUtils;

import org.seasar.framework.beans.util.Beans;

import org.seasar.struts.annotation.ActionForm;

import org.seasar.struts.annotation.Execute;

import javax.annotation.Resource;

/\*\*

\* 車両台帳登録 登録 Action.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

public class EditAction {

@ActionForm

@Resource

public EditForm editForm;

@Resource

public Sd04211Logic sd04211Logic;

/\*\*

入力エラーの場合は、左記のURLへ遷移させます。このJSPは、結果をJSONの形に成形して、画面に返却します。

\* indexメソッド.

\*

\* @return 遷移先

\* @throws DowaSdValidateException 選択必須入力チェックエラー

\*/

@Execute(input = "/jp/co/dowa/sd/core/jsonerror.jsp")

public String index() throws DowaSdValidateException {

// 共通権限チェック

// TODO:未実装

// RQ-3 4の業務処理を行う

// TODO:未実装

if (StringUtils.equals(this.editForm.screenType, ScreenTypeEnum.REGIST.getValue())) {

FormのデータをDtoに移送し、Logicを呼び出します。

final Sd04211RegisterDto registerDto =

Beans.createAndCopy(Sd04211RegisterDto.class, this.editForm)

.dateSystemFormatConverter("yksdt", "ykedt").execute();

this.sd04211Logic.register(registerDto);

} else if (StringUtils.equals(this.editForm.screenType, ScreenTypeEnum.EDIT.getValue())) {

final Sd04211UpdateDto updateDto = Beans.createAndCopy(Sd04211UpdateDto.class, this.editForm)

.dateSystemFormatConverter("yksdt", "ykedt").execute();

画面入力データを次画面(同じ画面の初期表示を参照モードで開く)向けに格納します。

this.sd04211Logic.update(updateDto);

}

// 画面返却値の作成(引継ぎ項目コピー)

final IndexForm indexForm = Beans.createAndCopy(IndexForm.class, this.editForm).execute();

indexForm.screenType = ScreenTypeEnum.READ.getValue();

ResponseUtils.write(false, indexForm);

データをResponseに詰め、返却します。JSONの場合、returnがnullなので注意。

return null;

}

}

}

#### Form

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.form.sd04211;

import lombok.ToString;

import org.seasar.struts.annotation.Arg;

import org.seasar.struts.annotation.Maxbytelength;

import org.seasar.struts.annotation.Required;

import java.io.Serializable;

以下にValidatorの一覧があります。

http://52.68.179.228/svn/1510\_hanbai\_sanpai/document/40\_製造（M）/01\_調査・検討/開発ガイドライン/Validator一覧.xlsx

足りないものがあったら、駒田まで。

/\*\*

\* 車両台帳登録 登録Form.

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

\*/

@ToString

public class EditForm implements Serializable {

/\*\*

\* コンストラクタ.

\*/

public EditForm() {}

/\*\* 製品部門 . \*/

@Required(arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.003") )

@Maxbytelength(maxbytelength = 3, arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.003") )

public String spart;

/\*\* 運搬事業者コード. \*/

@Required(arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.002") )

public String ujscd;

/\*\* 車両コード. \*/

@Required(arg0 = @Arg(key = "jp.co.dowa.sd.sd04211.caption.004") )

public String syacd;

/\*\* 車両枝番号. \*/

public String vehic;

/\*\* 更新日時 . \*/

public String ksdat;

/\*\* 車両登録番号. \*/

@Required

public String sytno;

/\*\* 販売組織. \*/

public String nyrvk;

/\*\* 車種コード. \*/

@Required

public String syscd;

/\*\* 最終更新者 . \*/

public String kssnm;

/\*\* 有効開始日. \*/

@Required

public String yksdt;

/\*\* 有効終了日. \*/

@Required

public String ykedt;

/\*\* 削除フラグ. TODO:DB項目にない。 \*/

public String ykedtOver;

/\*\* 最大積載量. \*/

@Required

public String sdsam;

/\*\* 車両重量. \*/

@Required

public String syaam;

/\*\* 車両総重量. \*/

@Required

public String srsam;

/\*\* 指図番号. \*/

@Required

public String aufnr;

/\*\* 画面モード. \*/

@Required

public String screenType;

/\*\* 前画面検索条件. TODO:項目名未定. \*/

public String preSearchConditions;

public OrderForm[] orderForm;

}

#### JSP

初期表示と同一のものを利用する。5.1.3.4)JSPを参照のこと。

### ロジック層

#### Logic

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.logic.sd04211;

import jp.co.dowa.sd.component.sd04211.Sd04211DbAccessComponent;

import jp.co.dowa.sd.core.exception.DowaSdSystemException;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211FindResultDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211RegisterDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211UpdateDto;

import jp.co.dowa.sd.entity.Zswma0;

import jp.co.dowa.sd.service.Zswma0Service;

import jp.co.dowa.sd.util.DateUtils;

import org.seasar.framework.beans.util.Beans;

import javax.annotation.Resource;

/\*\*

\* 車両台帳登録 業務ロジック.

\*

\* @version $Revision$

\* @author ndha

\* @since 1.0

\*/

public class Sd04211Logic {

@Resource

Sd04211DbAccessComponent sd04211DbAccessComponent;

@Resource

public Zswma0Service zswma0Service;

/\*\*

\* 車両台帳を登録する.

\*

\* @param registerDto パラメータ

DTOからEntityにデータを移送しServiceを呼び出します。

\*/

public void register(final Sd04211RegisterDto registerDto) {

final Zswma0 entity = Beans.createAndCopy(Zswma0.class, registerDto).execute();

this.sd04211DbAccessComponent.completeUpdateInfo(entity);

final int result = this.zswma0Service.insert(entity);

if (result != 1) {

throw new DowaSdSystemException();

}

return;

}

/\*\*

\* 車両台帳を更新する.

\*

\* @param updaterDto パラメータ

\*/

public void update(final Sd04211UpdateDto updaterDto) {

final Zswma0 entity = Beans.createAndCopy(Zswma0.class, updaterDto).execute();

this.sd04211DbAccessComponent.completeUpdateInfo(entity);

final int result = this.zswma0Service.updateExcludesNull(entity);

if (result != 1) {

throw new DowaSdSystemException();

}

return;

}

}

#### Dto

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.dto.sd04211;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 車両台帳登録 登録Form.

ClassにDataアノテーションを付けます。そうすると、Lombokが見えないところでgetter、setterを作ってくれます。(アウトラインで見えます。)

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

\*/

@Data

public class Sd04211RegisterDto implements Serializable {

/\*\* 製品部門 . \*/

private String spart;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号 // TODO:項目名未定. \*/

private String syaSeqno;

/\*\* 車両番号. \*/

private String sytno;

/\*\* 販売組織. \*/

private String nyrvk;

/\*\* 車種. \*/

private String syscd;

/\*\* 有効開始日. \*/

private String yksdt;

/\*\* 有効終了日. \*/

private String ykedt;

/\*\* 最大積載量. \*/

private String sdsam;

/\*\* 車両重量. \*/

private String syaam;

/\*\* 車両総重量. \*/

private String srsam;

/\*\* 指図番号. \*/

private String aufnr;

}

### サービス層

Zswma0Serviceが継承しているAbstractServiceGeneratedの実装を利用するため、記載しない。

## 単票画面の一覧表示機能

受注伝票入力の「処理品・その他・明細」部分の実装イメージを例示する。

### 実装イメージ

単票の場合は、Ajaxアクセスで一覧部分を実装するのではなく、JSON文字列を画面に配置し、その値を、JsRenderのテンプレートにバインドするかたちで実装する。



図 5‑3 単票画面の一覧部分の実装イメージ

### プレゼンテーション層(値の取得と、templateの展開)

#### Action

IndexAction#indexの抜粋

Beans.copy(findResultDto, this.sd04211Model).dateConverter("yyyy/MM/dd hh:mm:ss", "updata")

.dateInputFormatConverter("yksdt", "ykedt").execute();

final List<OrderModel> orderModelList = new ArrayList<OrderModel>();

for (int i = 0; i < 3; i++) {

final OrderModel orderModel = new OrderModel();

orderModel.setSpart("ZZ");

orderModel.setNsgam(i + "-1");

orderModel.setSekam(i + "-2");

orderModel.setMenge(i + "-3");

orderModel.setMeins(i + "-4");

orderModel.setJpckb(i + "-5");

orderModel.setPayme(i + "-6");

JSONICを利用して、オブジェクトをJSON化する。JSON文字列をModelに設定する。

orderModel.setBill(i + "-7");

orderModel.setMnfno(i + "-8");

orderModel.setAufnr(i + "-9");

orderModelList.add(orderModel);

}

final String disposalPermitJsonString =

JSON.escapeScript(orderModelList.toArray(new OrderModel[orderModelList.size()]));

this.sd04211Model.setDisposalPermitJsonString(disposalPermitJsonString);

#### JSP

index.jspの抜粋

template部分

<script id="disposalPermitTemplate" type="text/x-jsrender">

<tr>

<td rowspan="2">

Type=”text/x-jsrender”を設定する。Temlpateであることを明示する。

<input type="checkbox" />

</td>

<td>

<input type="checkbox" />

</td>

<td>

<dowaui:generalPurposeSearchSingle id="orderForm[{{>idx}}].spart" name="orderForm[{{>idx}}].spart" value="{{>spart}}" maxlength="10" class="wd-23"

nameValue="" namearea\_class="wd-60" namearea\_maxlength="40"

searchiconOnclick="test();" screenType="" inJsRender="true" />

</td>

<td colspan="2">

Taglibの引数も、JsRenderの置換が可能。Taglibには、inJsRender=”true”を設定してください。(Scriptタグ重複の対応が入っています。)

<dowaui:itemSearchSingle value="" id="" name="" maxlength="10" class="wd-23"

nameValue="" namearea\_name="" namearea\_class="wd-60" namearea\_maxlength="40"

searchiconOnclick="test();" screenType="" inJsRender="true" />

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].nsgam" name="orderForm[{{>idx}}].nsgam" value="{{>nsgam}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].nsgam" name="orderForm[{{>idx}}].nsgam" value="{{>nsgam}}" class="ui-text-readonly wd-60" readonly />

JsRenderのif文等が使えます。必要に応じて利用してください。

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].sekam" name="orderForm[{{>idx}}].sekam" value="{{>sekam}}" class="ui-text wd-60"/>

{{else}}

<input type="text" id="orderForm[{{>idx}}].sekam" name="orderForm[{{>idx}}].sekam" value="{{>sekam}}" class="ui-text-readonly wd-60" readonly />

actionFormのfield名[配列添え字].fieldクラスのメンバで配列型をActionFormに渡すことができます。

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].menge" name="orderForm[{{>idx}}].menge" value="{{>menge}}" class="ui-text wd-60" />/<input type="text" id="orderForm[{{>idx}}].meins" name="orderForm[{{>idx}}].meins" value="{{>meins}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].menge" name="orderForm[{{>idx}}].menge" value="{{>menge}}" class="ui-text-readonly wd-60" readonly />/<input type="text" id="orderForm[{{>idx}}].meins" name="orderForm[{{>idx}}].meins" value="{{>meins}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td rowspan="2">

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].jpckb" name="orderForm[{{>idx}}].jpckb" value="{{>jpckb}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].jpckb" name="orderForm[{{>idx}}].jpckb" value="{{>jpckb}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].payme" name="orderForm[{{>idx}}].payme" value="{{>payme}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].payme" name="orderForm[{{>idx}}].payme" value="{{>payme}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].bill" name="orderForm[{{>idx}}].bill" value="{{>bill}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].bill" name="orderForm[{{>idx}}].bill" value="{{>bill}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].mnfno" name="orderForm[{{>idx}}].mnfno" value="{{>mnfno}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].mnfno" name="orderForm[{{>idx}}].mnfno" value="{{>mnfno}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td>

{{if screenType == 1}}

<input type="text" id="orderForm[{{>idx}}].aufnr" name="orderForm[{{>idx}}].aufnr" value="{{>aufnr}}" class="ui-text wd-60" />

{{else}}

<input type="text" id="orderForm[{{>idx}}].aufnr" name="orderForm[{{>idx}}].aufnr" value="{{>aufnr}}" class="ui-text-readonly wd-60" readonly />

{{/if}}

</td>

<td rowspan="2">

<input type="text" id="" name="" />

</td>

</tr>

</script>

index.jspの抜粋

JSON格納部分

<input type="hidden" id="disposalPermitJsonString" value="${f:h(sd04211Model.disposalPermitJsonString)}" />

### クライアント層

#### CSJS

JSON文字列をparse

$(document).ready(function() {

var disposalPermitData = ImJson.parseJSON($('#disposalPermitJsonString').val());

for (var i = 0; i < disposalPermitData.length; i++ ) {

disposalPermitData[i].idx = i;

disposalPermitData[i].screenType = ($('#screenType').val() == "read") ? 0 : 1;

var template = $.templates('#disposalPermitTemplate');

screenTypeを0 or 1に変換。なぜかJsRender Template側でStringの比較ができなかったため。

var htmlOutput = template.render(disposalPermitData[i]);

$('#disposalPermitRowTbody').append(htmlOutput);

}

});

Templateを取得し、値をbind。生成したhtmlをappendする。

### プレゼンテーション層(一覧の更新データ取得)

#### Form

EditFormの抜粋

public OrderForm[] orderForm;

## 一覧画面の検索機能

車両台帳の画面の検索部分のイメージを例示する。

### 画面遷移イメージ

以下のように、遷移してくる想定で実装する。



図 5‑4　車両台帳の画面遷移

### ソースの構成

以下のようにソースを構成する。

表 ‑　一覧画面検索時のソース構成

|  |  |  |
| --- | --- | --- |
| レイヤー | 主要クラス | ソース |
| プレゼンテーション層 | Action | jp.co.dowa.sd.action.sd04222.SearchAction |
| Form | jp.co.dowa.sd.form.sd04222.SearchForm |
| Model | jp.co.dowa.sd.model.sd04222.SearchResultModel |
| JSP | /jp/co/dowa/sd/sd04222/index.jsp |
| ロジック層 | Logic | jp.co.dowa.sd.logic.sd04211.Sd04211Logic |
| Dto | jp.co.dowa.sd.dto.sd04211.Sd04211SearchDto |
| Dto | jp.co.dowa.sd.dto.sd04211.Sd04211SearchResultDto |
| サービス層 | Service | jp.co.dowa.sd.service.Zswma0Service |
| クライアント層 | CSJS | /dowa/csjs/sd04222/sd04222.js |

### クライアント層

#### CSJS

var dowa = dowa || {};

dowa.sd = dowa.sd || {};

ajaxSendの第二引数にCallback関数を指定する。

第三引数がエラー時、第四引数が正常、エラー両方だが、利用することはほんとんどないはず。

dowa.sd.sd04222 = {};

$(function() {

dowa.sd.sd04222.search = function() {

$('#sd04222Form').attr('action', $('#sd04222SearchForm').attr('action'));

dowa.sd.core.request.ajaxSend($('#sd04222Form')

// done

,function(resultObject, textStatus, jqXHR) {

一覧部分をいったんクリア。

$('#searchResultListTbody').empty();

for (var i = 0; i < resultObject.data.length; i++ ) {

resultObject.data[i].idx = i;

var template = $.templates('#searchResultTemplate');

var htmlOutput = template.render(resultObject.data[i]);

返却値をTemplateに展開しAappendする。

$('#searchResultListTbody').append(htmlOutput);

$('#syacd\_' + i + '\_link').click({'idx' : i}, function (event) {

var transferInfo = {

'spart' : $('#spart\_' + event.data.idx).val(),

'ujscd' : $('#ujscd\_' + event.data.idx).val(),

'syacd' : $('#syacd\_' + event.data.idx).val(),

一覧中のリンクのイベントを追加する。

'vehic' : $('#vehic\_' + event.data.idx).val(),

'screenType' : 'read',

'preSearchConditions' : ''

}

dowa.sd.core.request.transitionTo($('#sd04222TransferForm').attr('action'), transferInfo);

});

}

$('#pagingNumberTop').pagingNumber('setNumber', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

Paging系部品への値設定。

"to" : resultObject.pagingControl.numTo

});

$('#pagingNumberBottom').pagingNumber('setNumber', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

"to" : resultObject.pagingControl.numTo

});

$('#pagingCongorlTop').pagingControl('setControl', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

"to" : resultObject.pagingControl.numTo,

"currentPage" : resultObject.pagingControl.currentPage,

"sortKey" : resultObject.pagingControl.sortKey,

"sortOrder" : resultObject.pagingControl.sortOrder,

});

$('#pagingCongorlBottom').pagingControl('setControl', {

"total" : resultObject.pagingControl.total,

"from" : resultObject.pagingControl.numFrom,

"to" : resultObject.pagingControl.numTo,

"currentPage" : resultObject.pagingControl.currentPage,

"sortKey" : resultObject.pagingControl.sortKey,

"sortOrder" : resultObject.pagingControl.sortOrder,

});

});

};

$('#searchBtn').click(function() {

dowa.sd.sd04222.search();

});

});

### プレゼンテーション層

#### Action

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.action.sd04222;

import jp.co.dowa.sd.core.exception.DowaSdValidateException;

import jp.co.dowa.sd.dto.sd04211.Sd04211SearchDto;

import jp.co.dowa.sd.dto.sd04211.Sd04211SearchResultDto;

import jp.co.dowa.sd.form.sd04222.SearchForm;

import jp.co.dowa.sd.logic.sd04211.Sd04211Logic;

import jp.co.dowa.sd.model.sd04222.SearchResultModel;

import jp.co.dowa.sd.util.ResponseUtils;

import org.seasar.framework.beans.util.Beans;

import org.seasar.struts.annotation.ActionForm;

import org.seasar.struts.annotation.Execute;

import java.util.ArrayList;

import java.util.List;

import javax.annotation.Resource;

/\*\*

\* 車両台帳 検索 Action.

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

\*/

public class SearchAction {

@ActionForm

@Resource

public SearchForm searchForm;

@Resource

public Sd04211Logic sd04211Logic;

/\*\*

\* indexメソッド.

\*

\* @return 遷移先

\* @throws DowaSdValidateException 選択必須入力チェックエラー

\*/

@Execute(input = "/jp/co/dowa/sd/core/jsonerror.jsp")

public String index() throws DowaSdValidateException {

// 共通権限チェック

// TODO:未実装

final Sd04211SearchDto sd04211SearchDto =

Beans.createAndCopy(Sd04211SearchDto.class, this.searchForm).execute();

sd04211SearchDto.getPagingControl().limit = 10;

画面引数に対してlimitを設定する。Limitの保持方法が未決定なので、とりあえず、直接指定でOK。できるようならばInterceptorで補完したい。

final List<Sd04211SearchResultDto> sd04211SearchResultDtoList =

this.sd04211Logic.search(sd04211SearchDto);

final List<SearchResultModel> searchResultModelList = new ArrayList<SearchResultModel>();

for (final Sd04211SearchResultDto sd04211SearchResultDto : sd04211SearchResultDtoList) {

final SearchResultModel searchResultModel =

Beans.createAndCopy(SearchResultModel.class, sd04211SearchResultDto).execute();

searchResultModelList.add(searchResultModel);

}

// 画面返却値の作成

ResponseUtils.write(false, searchResultModelList);

return null;

}

}

#### Form

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.form.sd04222;

import jp.co.dowa.sd.core.dto.PagingControl;

import java.io.Serializable;

/\*\*

\* 車両台帳 検索Form.

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

\*/

@ToString

public class SearchForm implements Serializable {

/\*\* 製品部門 . \*/

public String spart;

/\*\* 運搬事業者コード. \*/

public String ujscd;

/\*\* 車両コード. \*/

public String syacd;

/\*\* 有効.TODO:項目名未定 \*/

画面引数に対してlimitを設定する。Limitの保持方法が未決定なので、とりあえず、直接指定でOK。できるようならばInterceptorで補完したい。

public String enable;

/\*\* 削除済み.TODO:項目名未定 \*/

public String deleteFlag;

/\*\* ページング制御 情報. \*/

public PagingControl pagingControl;

}

#### Model

package jp.co.dowa.sd.model.sd04222;

import jp.co.dowa.sd.core.dto.PagingControlResult;

import lombok.Data;

/\*\*

\* 車両台帳 検索結果Model.

\*

\* @version $Revision$

\* @author t.komadas

\* @since 1.0

\*/

@Data

public class SearchResultModel {

/\*\* 製品部門. \*/

private String spart;

/\*\* 運搬事業者コード. \*/

public String ujscd;

/\*\* 運搬事業者名称. \*/

private String name6;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号. \*/

private String vehic;

/\*\* 車両番号コード. \*/

private String sytno;

/\*\* 有効期間\_有効開始日. \*/

private String yksdt;

/\*\* 有効期間\_有効終了日. \*/

private String ykedt;

/\*\* 最大積載量\_重量. \*/

private String sdsam;

/\*\* 車両重量\_重量. \*/

private String syaam;

/\*\* 車両総重量\_重量. \*/

private String srsam;

/\*\* 指図番号\_指図. \*/

private String aufnr;

/\*\* 削除フラグ\_フラグ. \*/

private boolean ykedtOver;

/\*\* ページング制御パラメータ. \*/

private PagingControlResult pagingControlResult;

}

#### JSP

<%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>

<%@ taglib prefix="imarttag" uri="http://www.intra-mart.co.jp/taglib/foundation/imarttag"%>

<%@ taglib prefix="imart" uri="http://www.intra-mart.co.jp/taglib/core/standard"%>

<%@ taglib prefix="html" uri="http://struts.apache.org/tags-html"%>

<%@ taglib prefix="f" uri="http://sastruts.seasar.org/functions"%>

<%@ taglib prefix="tiles" uri="http://struts.apache.org/tags-tiles"%>

<%@ taglib prefix="dowaui" uri="http://dowa.co.jp/core/tags" %>

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<tiles:insert template="/WEB-INF/view/jp/co/dowa/sd/layout/layout.jsp" flush="true">

<tiles:put name="title" value="収集運搬車両台帳一覧"/>

<tiles:put name="header" type="string">

<script src="${f:url('/dowa/csjs/sd04222/sd04222.js')}"></script>

<script id="searchResultTemplate" type="text/x-jsrender">

<tr>

<td><input type="text" class="ui-text-readonly" value="{{>name6}}" readonly /></td>

<td><a href="javascript:void;" id="syacd\_{{>idx}}\_link">{{>syacd}}</a>

<input type="hidden" id="spart\_{{>idx}}" value="{{>spart}}" />

<input type="hidden" id="ujscd\_{{>idx}}" value="{{>ujscd}}" />

<input type="hidden" id="syacd\_{{>idx}}" value="{{>syacd}}" />

<input type="hidden" id="vehic\_{{>idx}}" value="{{>vehic}}" />

</td>

<td><input type="text" class="ui-text-readonly" value="{{>vehic}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>sytno}}" readonly /></td>

<td></td>

<td><input type="text" class="ui-text-readonly" value="{{>yksdt}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>ykedt}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>sdsam}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>syaam}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>srsam}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>aufnr}}" readonly /></td>

<td><input type="text" class="ui-text-readonly" value="{{>ykedtOver}}" readonly /></td>

</tr>

</script>

</tiles:put>

<tiles:put name="content" type="string">

<div class="ui-form-container">

<div class="ui-title">

<imarttag:imartTitleBar title="収集運搬車両台帳一覧" icon="/images/standard/title.gif"/>

</div>

<form id="sd04222Form" name="sd04222Form" method="post">

<div id=errorDiv>

<span id=msg>

<html:errors/>

</span>

</div>

<table class="ui-form">

<colgroup>

<col class="wd-12">

<col class="wd-38">

<col class="wd-12">

<col class="wd-38">

</colgroup>

<tr>

<th><label class="ui-require">製品部門</label></th>

<td>

<dowaui:productDeptSelect value="${sd04222Model.spart}" iphkb="" gmkcd="" id="spart" name="spart" width="200px"/>

</td>

<th><label>状態</label></th>

<td>

<input type="checkbox" name="enable">有効&nbsp;<input type="checkbox" name="deleteflag" >削除済み

</td>

</tr>

<tr>

<th><label>運搬事業者コード</label></th>

<td>

<dowaui:generalPurposeSearchSingle value="${sd04222Model.ujscd}" id="ujscd" name="ujscd" maxlength="10" class="wd-23"

nameValue="" searchiconOnclick="test();" screenType="regist" displayType="2" />

</td>

<th><label>車両コード</label></th>

<td>

<dowaui:generalPurposeSearchSingle value="${sd04222Model.syacd}" id="syacd" name="syacd" maxlength="10" class="wd-23"

nameValue="" searchiconOnclick="test();" screenType="regist" displayType="2" />

</td>

</tr>

</table>

<div style="text-align:center;"><input type="button" value="検索" id="searchBtn"/><input type="button" value="クリア" class="ml-10" id="clearBtn"/></div>

<dowaui:pagingControl onClick="dowa.sd.sd04222.search();" id="pagingCongorlTop" pagingParameterId="pagingParameter" />

<dowaui:pagingDisplayNumber id="pagingNumberTop"/>

<table class="ui-table mt-10">

ページング用部品を設定する。

<thead>

<tr>

<th><label>運搬事業者名称</label></th>

<th><label>車両コード</label></th>

<th><label>車両枝番</label></th>

<th><label>車両登録番号</label></th>

<th><label>車種名称</label></th>

<th><label>有効開始日</label></th>

<th><label>有効終了日</label></th>

<th><label>最大積載量</label></th>

<th><label>車両重量</label></th>

<th><label>車両総重量</label></th>

<th><label>指図番号</label></th>

<th><label>削除フラグ</label></th>

</tr>

</thead>

ページング用部品を設定する。

<tbody id="searchResultListTbody">

</tbody>

</table>

<dowaui:pagingControl onClick="dowa.sd.sd04222.search()" id="pagingCongorlBottom" pagingParameterId="pagingParameter" />

<dowaui:pagingDisplayNumber id="pagingNumberBottom"/>

<dowaui:pagingParameter id="pagingParameter" />

</form>

<form id="sd04222SearchForm" name="sd04222SearchForm" method="post" action="${f:url('/sd04222/search/')}">

</form>

<form id="sd04222TransferForm" name="sd04222TransferForm" method="post" action="${f:url('/sd04211/')}">

</form>

</div>

</tiles:put>

</tiles:insert>

### ロジック層

#### Logic

Sd04211Logic#search

/\*\*

\* 車両台帳を検索する.

\*

\* @param searchDto パラメータ

\* @return 検索された車両台帳

\*/

public List<Sd04211SearchResultDto> search(final Sd04211SearchDto searchDto) {

final Zswma0R001Condition condition = new Zswma0R001Condition();

condition.spart = searchDto.getSpart();

condition.ujscd = StringUtils.isEmpty(searchDto.getUjscd()) ? null : searchDto.getUjscd();

condition.syacd = StringUtils.isEmpty(searchDto.getSyacd()) ? null : searchDto.getSyacd();

final List<Zswma0> resultList =

this.zswma0Service.r001(condition, searchDto.getPagingControl());

final List<Sd04211SearchResultDto> resultDtoList = new ArrayList<Sd04211SearchResultDto>();

for (final Zswma0 result : resultList) {

final Sd04211SearchResultDto resultDto =

Beans.createAndCopy(Sd04211SearchResultDto.class, result).execute();

resultDtoList.add(resultDto);

}

return resultDtoList;

}

#### Dto

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.dto.sd04211;

import jp.co.dowa.sd.core.dto.PagingControl;

import lombok.Data;

import java.io.Serializable;

/\*\*

\* 車両台帳 Seach用Dto.

\*

\* @version $Revision$

\* @author t.komada

\* @since 1.0

\*/

@Data

public class Sd04211SearchDto implements Serializable {

/\*\* 製品部門 . \*/

private String spart;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 有効.TODO:項目名未定 \*/

private String enable;

/\*\* 削除済み.TODO:項目名未定 \*/

private String deleteFlag;

/\*\* ページング制御情報. \*/

ページング情報を移送する。

private PagingControl pagingControl;

}

#### ResultDto

/\*

\* (C) DOWA HOLDINGS Co., Ltd. 2015

\*/

package jp.co.dowa.sd.dto.sd04211;

import jp.co.dowa.sd.core.dto.PagingControlResult;

import lombok.Data;

import java.io.Serializable;

import java.sql.Timestamp;

/\*\*

\* 車両台帳登録 find用Dto.

\*

\*/

@Data

public class Sd04211SearchResultDto implements Serializable {

/\*\* 製品部門 . \*/

private String spart;

/\*\* 運搬事業者コード. \*/

private String ujscd;

/\*\* 車両コード. \*/

private String syacd;

/\*\* 車両枝番号. \*/

private String vehic;

/\*\* 更新日時 . \*/

private Timestamp updata;

/\*\* 車両登録番号. \*/

private String sytno;

/\*\* 販売組織. \*/

private String nyrvk;

/\*\* 車種コード. \*/

private String syscd;

/\*\* 最終更新者 . \*/

private String hksha;

/\*\* 有効開始日. \*/

private String yksdt;

/\*\* 有効終了日. \*/

private String ykedt;

/\*\* 削除フラグ. TODO:DB項目にない。 \*/

private boolean ykedtOver;

/\*\* 最大積載量. \*/

private String sdsam;

/\*\* 車両重量. \*/

private String syaam;

/\*\* 車両総重量. \*/

private String srsam;

/\*\* 指図番号. \*/

private String aufnr;

}

### サービス層

#### Service

package jp.co.dowa.sd.service;

import jp.co.dowa.sd.condition.zswma0.Zswma0R001Condition;

import jp.co.dowa.sd.core.dto.PagingControl;

import jp.co.dowa.sd.entity.Zswma0;

import java.util.List;

import javax.annotation.Generated;

/\*\*

\* {@link Zswma0}のサービスクラスです。

\*

\*/

@Generated(

value = {"S2JDBC-Gen 2.4.46",

"org.seasar.extension.jdbc.gen.internal.model.ServiceModelFactoryImpl"},

date = "2015/12/03 19:16:42")

public class Zswma0Service extends Zswma0ServiceGenerated {

private static final String R002 = "zswma0\_r002.sql";

/\*\*

\* 車両台帳を取得します.

\*

\* @param condition 取得条件

\* @return 車両台帳データ

\*/

public List<Zswma0> r001(final Zswma0R001Condition condition, final PagingControl pagingControl) {

return this.getPagingResultList(Zswma0.class, Zswma0Service.R002, condition, pagingControl);

}

AbstractServiceGeneratedに実装したgetPagingResultListをcallしてください。

}